Emotion Sensitive Music Broadcasting by Analysing Facial Expressions using Machine Learning

MSc Research Project
Master of Science in Data Analytics

Christy Lyona Joseph Vijayan
Student ID: x21202583

School of Computing
National College of Ireland

Supervisor: Taimur Hafeez

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Christy Lyona Joseph Vijayan
Student ID:	x21202583
Programme:	Master of Science in Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Taimur Hafeez
Submission Due Date:	14/08/2023
Project Title:	Emotion Sensitive Music Broadcasting by Analysing Facial Expressions using Machine Learning
Word Count:	XXX
Page Count:	26

Signature:	
Date:	13th August 2023

Emotion Sensitive Music Broadcasting by Analysing Facial Expressions using Machine Learning

Christy Lyona Joseph Vijayan
x21202583

Abstract

Music has the power to be a medicine and a medium to inspire, rejuvenate and help oneself commit to a task at hand. To enjoy a seamless musical experience has been in focus from the invention of the Microphone itself. However, it is still a struggle in this Modern age of Technology. Hence, using the Top Edge technologies available to us at our fingertips to better the experience and facilitate a new way of enhancing this feeling with complex Recurrent Neural Networks and Machine Learning does seem to be a requirement than a privilege.

Using a complex CNN network and training the model using a modified VGG16 and training was carried out on over 10 thousand images and after 75 epochs and 13 hours an acceptable accuracy of 67% was achieved. With the introduction to Multithreading live image detection and detection of most dominant emotions were achieved and this information was then fed in a simple Machine Learning Model namely Gausian Naive Based to predict and recommend the top 10 songs for the two strongest feelings displayed by the user. For the last piece in this project I have used the opensource SoundCloud where the songs are fed in via api calls to SoundCloud via the webdrivers.

Finally with over 67% accuracy in Facial Recognition using a modified version of VGG16 and over 78% of accuracy in recommending the songs according to the users two dominant emotions and feeding these songs into the SounCloud and playing them I have created a complete Music Player that plays songs based on the users emotion and tries to elevate it to the best possible condition over time.

1 Introduction

Physiognomy is the practise of asserting or in a much bolder sense judging ones character or personality from their face (Ahonen; 2014). The great mind of Charles Darvin concluded his research by stating that human emotions can be observed and studied to understand ones 'True Self', he also stated that by observing a humans facial emotions and comparing it to an animal, mainly a domesticated animal, we can conclude the theory of evolution from a lesser species (Darwin and Prodger; 1998). With all this information freely available to us, the study of Emotion and Sentiments has taken a steep rise in the recent years. With advancement in the field of Natural Language Processing, sentiment analysis on various platforms have increased multi-folds.

Music therapy is a complete science where psychologists use music as a medium of therapy for their patients. However, the proposed methodology of this paper is not to determine the change of ones emotion by inducing music, but to reverse engineer the entire process. **To what extent can facial expression analysis be used to predict music preferences and enhance the music listening experience through emotion-sensitive music broadcasting using machine learning techniques?** With ongoing innovation and collaboration, the future of personalized music is full of potential and exciting possibilities.

The effect of music and its need in todays world has been established, it is now time to look at it from a financial point of view. Amazon Music has over 55 million customers and over 90 million songs, with only over 13% of the worlds global streaming market it has seen an increase of 104% active listeners just by the introduction of Alexa and Echo Dot from 2019 till this day. Just one technological advancement has helped amazon music to gain a staggering uptick in users, however it is completely annihilated while comparing it to Spotify.

Spotify has over 456 million active users, that is 12 times that of Amazon Music, after adding the songs, podcasts and record labels it has over 118 million audio tracks, with over 34 million as net income and a 9.6 billion dollar revenue Spotify is one of the leading music applications currently.

The major difference is easy of access and a playlist that never seems to end. Spotify has a seamless musical experience due to the amazing algorithm that accounts for every click, every song heard and the duration of the song heard, every song skipped and the list goes on. Hence, investing and presenting newer approaches to such a vast and financially rewarding field is definitely a good strategy.

2 Related Work

Facial expressions are a universal and reliable indicator of emotional states, and recent advances in machine learning have enabled researchers to develop systems for automated facial expression analysis. Music has long been recognized for its ability to serve as both a remedy and an inspiration. Physiognomy is the practise of asserting or in a much bolder sense judging ones character or personality from their face Browne (2018)

The invention of the microphone has helped to enhance the music experience. However, in today's modern age the development of complex Machine Learning techniques offers the potential to revolutionize the way we experience music. For instance, a study by Sharma et al. (2021) developed a system which uses a deep learning model to extract audio features from the user's music library and analyze the lyrics of the songs to identify the emotional content. The emotional content is then mapped to a set of predefined emotional states, which are used to recommend music tracks that match the user's current emotional state.

The paper by Pise et al. (2022) discusses methods for facial expression recognition with applications in challenging situations. The authors provide an overview of the different techniques used for facial expression recognition, including traditional computer vision methods and deep learning-based methods and also highlights some of the challenges faced when recognizing facial expressions in challenging situations, such as low lighting conditions, occlusions, and facial expressions that are not easily distinguishable. The objective of this proposed research is to develop an Emotion Sensitive Music Broadcasting system that utilizes facial expression analysis to predict music preferences.

Sadikoğlu and Idle Mohamed (2022) proposes a facial expression recognition system that uses CNN to improve accuracy. Their model consists of multiple convolutional layers followed by max-pooling, batch normalization, and fully connected layers. The authors also apply data preprocessing techniques such as histogram equalization, data augmentation, and normalization to enhance the performance of their model. To evaluate their model, the authors conduct experiments on benchmark datasets such as the Facial Expression Recognition 2013 (FER2013) dataset, which contains over 35,000 facial images of seven emotion classes (angry, disgust, fear, happy, sad, surprise, and neutral). They compare their model's performance with other state-of-the-art methods and achieve an accuracy of 70.24%, which outperforms other methods on the FER2013 dataset.

2.1 Previous work on emotion recognition using facial expressions

Facial Recognition grew in popularity in the early 1960's, all thanks to Woody Bledsoe, Helen Chan Wolf and Charles Bisson for their contribution in this field in 1964 and '65. Due to funding from an unnamed intelligence agency, most of their work was never published. However, they did not fail in setting down a great milestone.

Facial Recongnition was frowned upon as it was seen to be a leap in the direction of anti-freedom and stalking purposes. It wasnt until the leate 1980's that Sirovich and Kirby began applying linear algebra to the problem of facial recognition to use it for the purpose of biometric scanning and hence a potential security measure.

With the new found usage of Facial Recognition, this science grew and branched out soon by 1957, this was when most presume facial analysis with the use of machine learning algorithms was born.

Today with Holistic Matching, Feature-based, Skin Texture Analysis and Thermal Camera Edge detection methods we can capture and predict the persons identity using a dataset in real time. Feature Recognition helps identify if the individual is resting, awake, and also predict how that individuals face will be 20 years later! From matching scars or minute details to match a persons image from when they were a child to their older self it is all possible using Feature-Based detection. Holistic Matching is the most simplest and sophistic security system employed at every smart phone users fingertips. The development in the past 30 years can be stated as in 1980's the fastest facial recognition systems could recognize an image in just under 1 second, today FaceStation2 Suprema holds the banner of the fastest face recognition with up-to 3000 successful matches per second.

First, Mollahosseini et al. (2019) introduced the AffectNet database, which is a large-scale database of facial expressions that aims to enable research on facial expression recognition in the wild. The database contains over 1 million images of facial expressions, each labeled with valence (positive or negative) and arousal (intensity of the emotion). The images were collected from various sources, including the internet, television shows, and movies, to capture a wide range of facial expressions that occur in real-world situations. The authors used this dataset to train a deep neural network for facial expression recognition, achieving state-of-the-art performance on several benchmarks. The AffectNet database is a significant contribution to the field of affective computing, as it provides a large, diverse dataset for training and testing facial expression recognition algorithms. The dataset has been widely used in subsequent research, including in studies that aim to improve the accuracy and efficiency of facial expression recognition.

Second, Luo et al. (2021) proposed a facial expression recognition system based on machine learning models. Their research aims to provide an effective and accurate solution for facial expression recognition by exploring the performance of different machine learning algorithms on the FER2013 dataset. The authors began by explaining the FER2013 dataset, which contains over 35,000 images of facial expressions labeled with one of seven different emotions. They then discussed the feature extraction process, which involves converting the images into a set of numerical features that can be used for training and classification. The authors compared the performance of several machine learning models, including decision tree, K-nearest neighbor, support vector machine, and deep learning models, such as convolutional neural networks.

The experimental results showed that the deep learning models outperformed traditional machine learning algorithms. Specifically, the authors found that the convolutional neural network (CNN) achieved the highest accuracy of 66.78%, which is a considerable improvement compared to other machine learning models. The authors further explored the impact of data augmentation techniques on the performance of their proposed method and showed that data augmentation could improve the accuracy of the CNN model.The authors concluded that the proposed system can effectively recognize facial expressions and provide a promising solution for applications that require emotion recognition, such as music broadcasting.

In comparison to Mollahosseini et al. (2019), Luo et al. (2021) focused more on the comparison of different machine learning algorithms rather than on creating a new dataset. Both studies, however, are related to emotion recognition using facial expressions, which is also the focus of our research question on emotion-sensitive music broadcasting using machine learning algorithms.

2.2 Previous work on music recommendation systems

Now that it is established that a sentiment analysis is possible on a target under observation and it is established that it can be used for the betterment of oneself, the main question arises; Why should one invest this technology in a simple device as a Music Player or Recommender. This answer is simple and will be answered throughout the course of this paper.

1. Capital: Successful and well known companies like Spotify, Apple Music and YouTube all invest a mighty sum of their capital which is around 1.2 Billion according to Krysik in improving their Recommender Engines. With state laws to restrict data leaks, it has become an even more significant that utmost care in technological safety is taken. Hence, this style of recommendation if introduced can be revolutionizing.

2. Health and Well-Being: Music has proved to be of potential advantage in the field of treatment of patients diagnosed with Post Traumatic Stress (Landis-Shack et al.; 2017) or Anxiety. With the right set of mind an individual has been proven to show more focus and moldable by the doctor or psychiatrist compared to an agitated or distracted patient.

3. Therapeutic approach: The approach of Landis-Shack et al. has confirmed the advantage of music in treatment of patients detected with Post Traumatic Stress. May it be a simple domestic violence case in a minor or a Medal clad war hero, music induced emotional manipulation has been effective in almost all cases. It

causes Muscle and Mind relaxation during sessions of physical training or just silent meditation.

The paper by He et al. (2018) compares two different recommendation techniques: neural collaborative filtering (NCF) and matrix factorization (MF). NCF is a deep learning-based approach that can capture complex user-item interactions, while MF is a classic technique that decomposes the user-item matrix into latent factors. The paper evaluates the performance of both methods on large-scale datasets and shows that NCF outperforms MF in terms of recommendation accuracy. This work is significant for our research question on emotion-sensitive music broadcasting because it highlights the importance of using advanced machine learning techniques to capture subtle user preferences and emotions. NCF's ability to learn from user-item interactions and capture non-linear relationships between them is particularly relevant for our research.

The by Devoogdt et al. (2019) proposes a session-based music recommendation method based on recurrent neural networks (RNNs). The authors use a dataset of user sessions, where each session consists of a sequence of songs played by the user, and train an RNN to predict the next song in the session. The paper shows that their method outperforms several baseline methods in terms of recommendation accuracy and diversity. This work is significant for our research question because it highlights the importance of considering the temporal context of user listening behavior when recommending music. By modeling the sequential patterns of user sessions, the proposed method can capture the dynamic nature of user emotions and preferences over time.

3 Requirements and Design of the CNN Model

This section introduces the dataset and pre-processing of the dataset to be used, in my case the FER 2013 dataset provided by google. This section also gives a brief introduction to the roadmap of this project, the methodology used and the end goal of my Auto Playlist Generator.

This section is sub divided in 3 major parts:

1. Data Preparation.

2. Classification.

3. Key Aspects.

3.1 Data Preparation.

Here the steps of selecting the dataset, preparing the dataset for our particular observation and observing how the dataset is panned out and its various features will be addressed. This section is just an introduction to the Deep Learning CNN network creation.

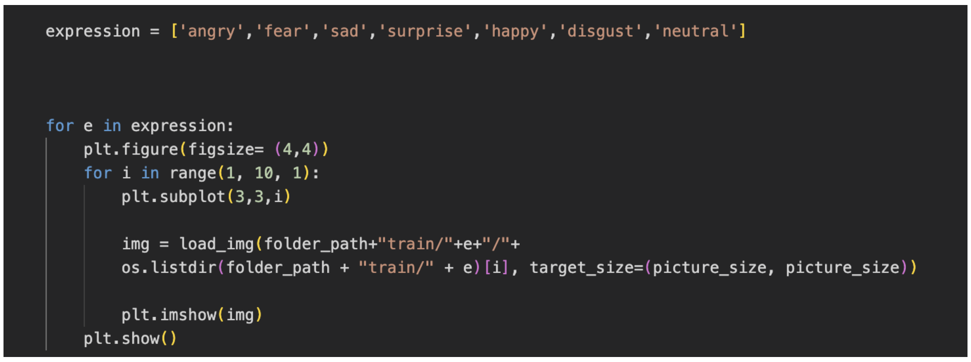
3.1.1 Data Gathering: FER 2013.

The Dataset selected for the CNN model is the famous FER 2013 dataset. It is an open source licensed dataset freely available on Kaggle. This dataset has images of 48x48 pixels grayscale each. The images are automatically registered and centered such as they occupy the same amount of space. The dataset is split into Training and Test directories.

Each Directory has 7 other Directories of 7 different emotions namely Angry, Disgust, Fear, Happy, Neutral, Sad and Surprised.

3.1.2 Data Pre-Processing.

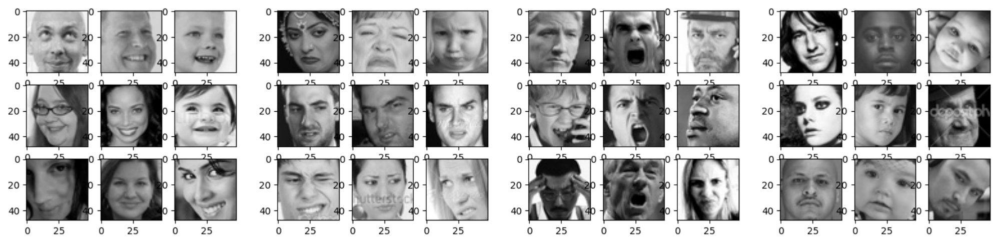
Now that the format of the dataset used is understood we can move ahead to reading and viewing the dataset. This is accomplished by the Keras library in python. We use the Keras Pre-Processing library to accomplish this, we read the images from the training dataset with respect to their emotion. For this we create a string of the names of the emotions with respect to the Directory names present. Using the load_img command provided by Keras Preprocessing we load these images and display them using another common package namely Math Plot. (Fig. 1).



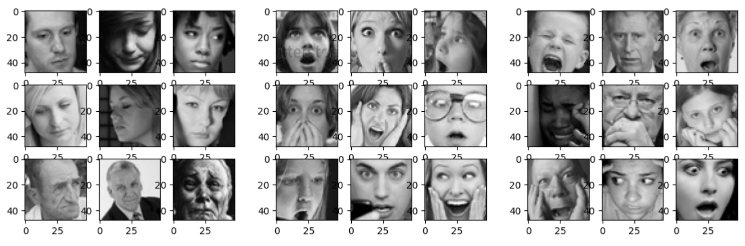
```
expression = ['angry','fear','sad','surprise','happy','disgust','neutral']



for e in expression:
    plt.figure(figsize= (4,4))
    for i in range(1, 10, 1):
        plt.subplot(3,3,i)

        img = load_img(folder_path+"train/"+e+"/"+
        os.listdir(folder_path + "train/" + e)[i], target_size=(picture_size, picture_size))

        plt.imshow(img)
    plt.show()
```

Figure 1: Using Keras Load Image to read the Dataset.

On Execution on the above code (Fig. 1) 7 different plots each of size 4x4 pixels and consisting of 9 first images present of each individual directories are displayed and can be observed below in Fig. 2.



(a) Happy (b) Disgust (c) Angry (d) Neutral



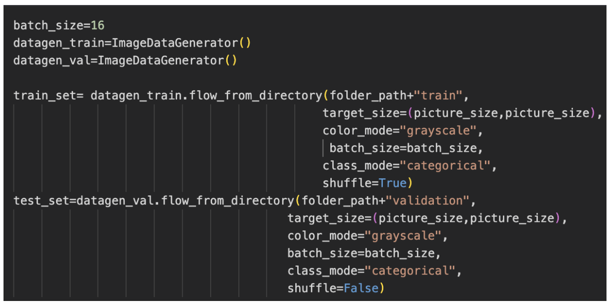
(e) Sad (f) Surprised (g) Fear

Figure 2: Images of Emotions in the Dataset.

For this project as we plan on using VGG16 modified we now convert all images to an input size of 48 pixels as the input of VGG16 is 48. If the images are observed in depth as in Fig. 13c the first image can be categorized in Angry and also in Neutral as the emotion isn't clearly portrait. Hence, these small details need to be taken care of and this is done in the modelling and classification stage.

3.2 Classification.

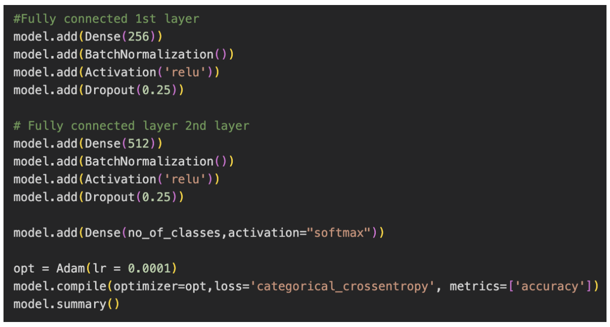
For classification to be carried out and prediction of the emotion to be done we first need to create a model. We have already decided to use VGG16 as our base model so all the images are set to a size of 48 pixels. We select a batch size of 16 and take the test and train set as a set of grayscale images of size 48 through ImageDataGenerator function provided by Keras again. This can be seen in Fig. 3



```
batch_size=16
datagen_train=ImageDataGenerator()
datagen_val=ImageDataGenerator()

train_set= datagen_train.flow_from_directory(folder_path+"train",
                                              target_size=(picture_size,picture_size),
                                              color_mode="grayscale",
                                               batch_size=batch_size,
                                              class_mode="categorical",
                                              shuffle=True)
test_set=datagen_val.flow_from_directory(folder_path+"validation",
                                        target_size=(picture_size,picture_size),
                                        color_mode="grayscale",
                                        batch_size=batch_size,
                                        class_mode="categorical",
                                        shuffle=False)
```

Figure 3: Creating Train And Test sets.

We can now proceed with the modelling. As there are 7 possible outputs of 7 different emotions we will be having 7 different classes. We will be creating 4 CNN layers and the first 2 layers will be fully connected. For our purpose the optimizer selected is as Adam with a learning rate of 0.0001 and for validation we will be using the accuracy as a metrics. The image below shows the entire creation of the model.

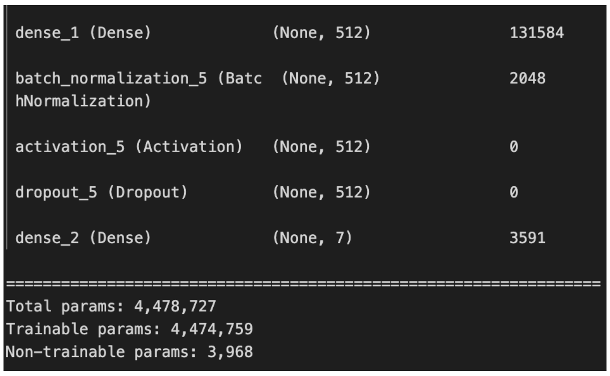


```
#Fully connected 1st layer
model.add(Dense(256))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.25))

# Fully connected layer 2nd layer
model.add(Dense(512))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.25))

model.add(Dense(no_of_classes,activation="softmax"))

opt = Adam(lr = 0.0001)
model.compile(optimizer=opt,loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

Figure 4: Model Creation.

The summary of the model provides detailed layers shape and parameters trainable in each layer, however we are only interested in the final 4 layers that we introduced in the VGG16 format to predict our output. The image in Fig. 5 is the layers inserted by running the code block in Fig. 4. The last layer has a shape of 7 that refers to our emotion classes of 7 different emotions ie. Angry, Disgust, Happy, Fear, Sad, Surprised and Neutral.

Now that the final layer has the right output classes we go ahead and fit this model to the training dataset. Thanks to Keras again we have libraries available to monitor the Episodes(aka. Epocs) and callbacks to attain and early stop if a sufficient accuracy or



```
dense_1 (Dense)             (None, 512)               131584

batch_normalization_5 (Batc  (None, 512)              2048
hNormalization)

activation_5 (Activation)   (None, 512)               0

dropout_5 (Dropout)         (None, 512)               0

dense_2 (Dense)             (None, 7)                 3591

=================================================================
Total params: 4,478,727
Trainable params: 4,474,759
Non-trainable params: 3,968
```

Figure 5: Summary of the Layers inserted.

validation accuracy is reached. Here we will create a checkpoint if the validation accuracy has shown sufficient increase however making sure that the model is not over-fitted.

One of the most crucial task in Deep Learning is using the right number of Epocs or Episodes, fewer Episodes generate poor accuracy however extreme number of Episodes or Epocs cause the problem of Over-fitting, although the accuracy will be astonishingly high, any image out of the normal will cause the model to predict completely wrong results; in simple terms the model will be rigid to a certain type of dataset and will under perform greatly for any other dataset (Rice and Wong; 2020).

Hence for this exact reason we choose two important parameters:

1. Early Stopping.
2. Reduce Learning Rate.

Early stopping is defined in the Fig. 6. Here the inputs are set to monitor the val_loss, simply if the loss is below a threshold the model is moving into over-fitting and to stop that from happening we stop the epocs early. For this project i have set 48 epocs to be the limit however after 17 executions the max epocs reached is 12. Hence, if the dataset is changed 48 epocs will be a good reference.

For the Early Stopping we will monitor the Validation Loss function. This function is the value of the cost function for my cross-validation data. This is a better choice over loss function as Loss should always be decreasing as the model accuracy improves, but a significant stale state or increase in Validation Loss can signify that the model is now fitting perfectly and any more tuning to this dataset will cause Over-fitting.

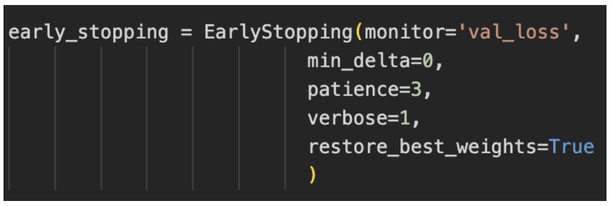
Minimum Delta parameter defines the starting point of this value, we set it as 0 as we introduce no bias and we want to stop the epocs at the time of Over-fitting when Validation Loss is drastically low and not before or after over-fitting has reached.

Patience is set to 3, this indicates that if there are 3 continuous epocs with no change in the monitored value, here Validation Loss the epocs will stop early.

Verbose will be set to 1 as we need an indication, here a message to show when the callback takes an action to stop the epocs and why was it stopped.

Finally, restore best weights is set to True as we want to keep in memory the weights related to the lowest Validation accuracy before our model began over-fitting, not at the brink of it. If early stopping is called due to patience variable then too the lowest Validation Loss.

Reduce Learning Rate shares some similar inputs as that of Early Stopping (Fig. 7). These parameters are nothing but triggering parameters that are called to stop the epocs



```
early_stopping = EarlyStopping(monitor='val_loss',
                               min_delta=0,
                               patience=3,
                               verbose=1,
                               restore_best_weights=True
                               )
```

Figure 6: Early Stopping Definition.

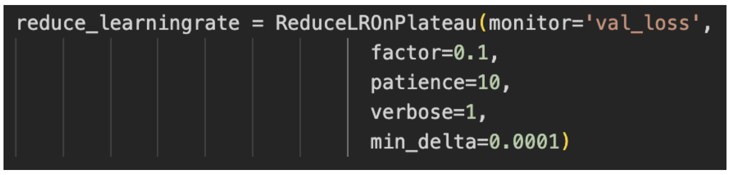
mid-way to avoid waste of computation power, time and avoid Over-fitting.

Patience and Verbose are already understood in Early stopping and have the same functionality here as well.

Factor is set to 0.1 and this resembles the reduction of the learning rate as the model begins to improve, this is given by a simple formula:

$$new\_LR = LR * factor \tag{1}$$

$$Here: \quad new\_LR = 0.001 * 0.1 \tag{2}$$



```
reduce_learningrate = ReduceLROnPlateau(monitor='val_loss',
                                        factor=0.1,
                                        patience=10,
                                        verbose=1,
                                        min_delta=0.0001)
```

Figure 7: Reduce Learning Rate Definition.

This will happen every time there is a significant change in the Validation Loss.

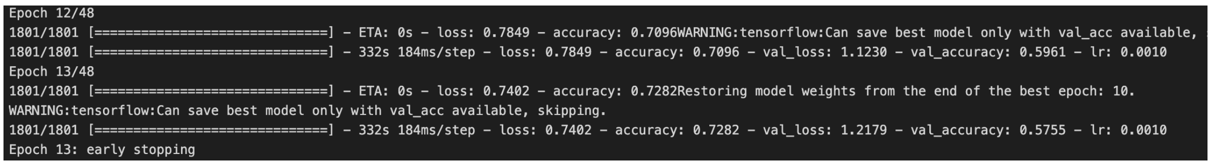
Finally, Min_Delta is nothing but the lowest bracket that the learning rate should reach, hence if learning rate goes below 0.001, the epocs will be stopped.

The learning rate will only be reduced if:

1. Significant Reduction is Monitored in the Validation Loss.

2. No Changes for 10 Epocs as Patience is set to 10.

3. Increase in Validation Loss is noticed, clear indication of Over-fitting.



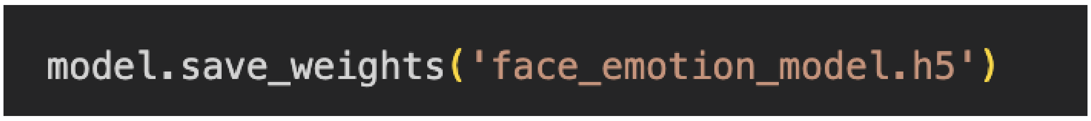
```
Epoch 12/48
1801/1801 [==============================] - ETA: 0s - loss: 0.7849 - accuracy: 0.7096WARNING:tensorflow:Can save best model only with val_acc available,
1801/1801 [==============================] - 332s 184ms/step - loss: 0.7849 - accuracy: 0.7096 - val_loss: 1.1230 - val_accuracy: 0.5961 - lr: 0.0010
Epoch 13/48
1801/1801 [==============================] - ETA: 0s - loss: 0.7402 - accuracy: 0.7282Restoring model weights from the end of the best epoch: 10.
WARNING:tensorflow:Can save best model only with val_acc available, skipping.
1801/1801 [==============================] - 332s 184ms/step - loss: 0.7402 - accuracy: 0.7282 - val_loss: 1.2179 - val_accuracy: 0.5755 - lr: 0.0010
Epoch 13: early stopping
```

Figure 8: Early Stopping due to Increase in Validation Loss.

3.3 Key Aspects.

Some Key Aspects of this Modelling plan can be mentioned as:

1. Training takes around 5 hours for just 10 epocs, the maximum of 13 epocs were reached which took 7 hours and 17 minutes.

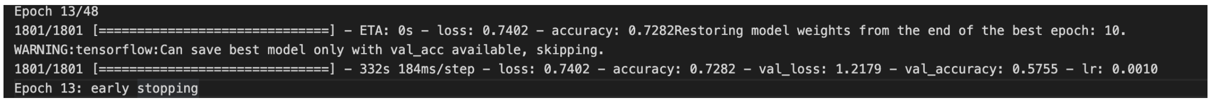


```
model.save_weights('face_emotion_model.h5')
```

Figure 9: Saving the weights of the best trained Epocs.

2. To avoid retraining of the entire model, the weights are stored in an h5 file that can be used to predict the output without training the entire model again.

3. An Accuracy after 13 epocs was concluded at 72% and early stop was triggered as the Model was entering Over-fitting due to an increase in the Validation Loss.



```
Epoch 13/48
1801/1801 [==============================] - ETA: 0s - loss: 0.7402 - accuracy: 0.7282Restoring model weights from the end of the best epoch: 10.
WARNING:tensorflow:Can save best model only with val_acc available, skipping.
1801/1801 [==============================] - 332s 184ms/step - loss: 0.7402 - accuracy: 0.7282 - val_loss: 1.2179 - val_accuracy: 0.5755 - lr: 0.0010
Epoch 13: early stopping
```

Figure 10: Accuracy of Face Recognition Model.

4. The Accuracy and Loss are plotted using Mat-Plot library to visualize the loss and accuracy of the model and gain a fair understanding of how accurately the model will perform.

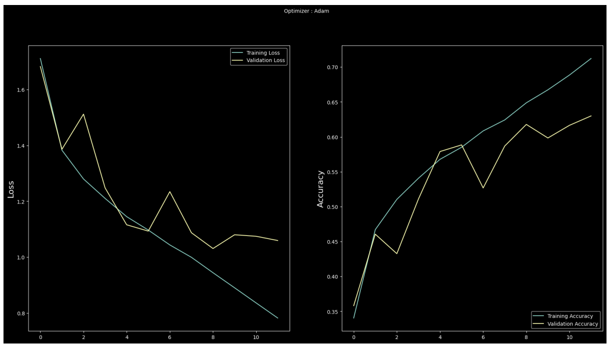
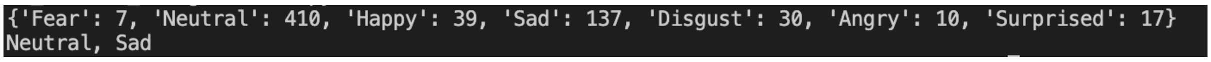


Figure 11: Comparison of Training and Validation Loss and Accuracy.

5. Detection of all emotions are done in real time and all the 7 emotions are correctly and accurately detected, the executed output is shown below in 13.

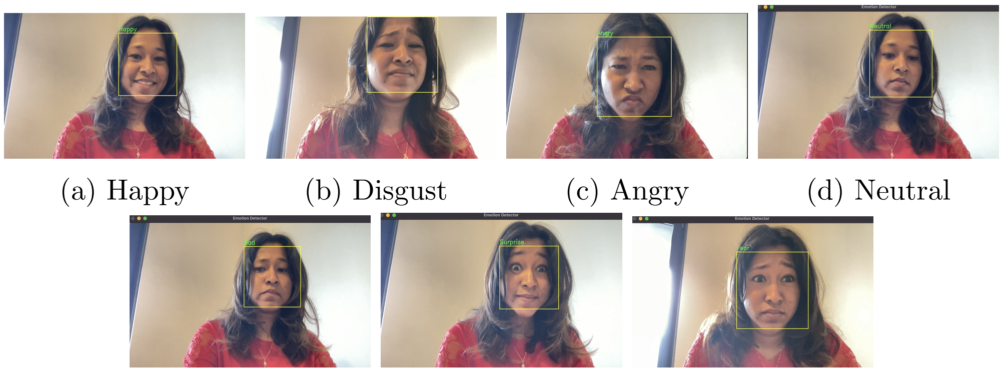
6. All these outputs are recorded per second and a counter function is set up to record the emotions and return the Two most prominent emotions. This can be seen in 12. The numbers represent the number of seconds the user displayed his/her emotion. For testing and evaluation purpose, i have tried to display all 7 emotions and 12 shows how long i held that emotion.

7. The end of the CNN and this section occurs when the two Prominent emotions are retrieved and this data is ready to be now fed as input to the RNN model where using a dataset we shall predict the songs that would help improve the users experience. This algorithm is explained in the Next Section.



```
{'Fear': 7, 'Neutral': 410, 'Happy': 39, 'Sad': 137, 'Disgust': 30, 'Angry': 10, 'Surprised': 17}
Neutral, Sad
```

Figure 12: Emotions in an Array with Prominent two Emotions Detected.



(a) Happy (b) Disgust (c) Angry (d) Neutral

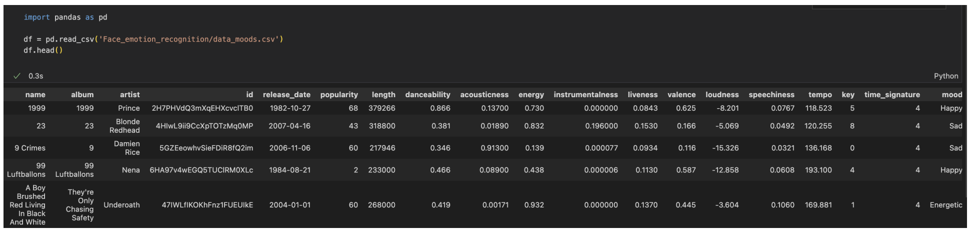
(e) Sad (f) Surprised (g) Fear

Figure 13: Output of CNN Emotion Detection in Real-Time.

4 Requirements and Design of the RNN Model

4.1 Understanding the Design.

The Music Mood Dataset is a open dataset available at Kaggle provided by Spotify. The columns present in this dataset are a total of 19 in number however not all are useful for our purpose. The entire dataset is shown in Fig. 14. With over 700 rows at our disposal and with fairly clean data completely free of N/A Values and the issue of outliers as the data has been marginalised in vocals and decibel units. It comprises of music from 650 Albums and encompasses 672 different artists in total! Another special observation of this dataset is its inputs that we will consider are all numerical representation of the songs attributes like Popularity, length, dance-ability, loudness, energy, acoustics, instrumentalness, liveliness, valence, speechiness, temp and key. In the data preparation section we will see the steps taken to prepare this data for creation of the RNN based recommender system. The modelling section will cover the models used for training the RNN and the reason for selecting Gaussian Naive Based algorithm for our prediction over Logistic Regression, Service Vector Classifier and Decision Tree Classifier. The reason for dropping few columns and retaining the rest will also be discussed in Data Preparation.



```python
import pandas as pd

df = pd.read_csv('Face_emotion_recognition/data_moods.csv')
df.head()
```

✓ 0.3s Python

name	album	artist	id	release_date	popularity	length	danceability	acousticness	energy	instrumentalness	liveness	valence	loudness	speechiness	tempo	key	time_signature	mood
1999	1999	Prince	2H7PHVdQ3mXqEHXcvclTB0	1982-10-27	68	379266	0.866	0.13700	0.730	0.000000	0.0843	0.625	-8.201	0.0767	118.523	5	4	Happy
23	23	Blonde Redhead	4HIwL9ii9CcXpTOTzMq0MP	2007-04-16	43	318800	0.381	0.01890	0.832	0.196000	0.1530	0.166	-5.069	0.0492	120.255	8	4	Sad
9 Crimes	9	Damien Rice	5GZEeowhvSieFDiR8fQ2im	2006-11-06	60	217946	0.346	0.91300	0.139	0.000077	0.0934	0.116	-15.326	0.0321	136.168	0	4	Sad
99 Luftballons	99 Luftballons	Nena	6HA97v4wEGQ5TUCIRM0XLc	1984-08-21	2	233000	0.466	0.08900	0.438	0.000006	0.1130	0.587	-12.858	0.0608	193.100	4	4	Happy
A Boy Brushed Red Living In Black And White	They're Only Chasing Safety	Underoath	47IWLfIKOKhFnz1FUEUIkE	2004-01-01	60	268000	0.419	0.00171	0.932	0.000000	0.1370	0.445	-3.604	0.1060	169.881	1	4	Energetic

Figure 14: Raw Dataset.

Now that we have this dataset I will move on and present to you the overall idea of this project from the beginning of the CNN we covered in section 4 to the RNN we will cover in this section. If you refer to the flowchart in Fig. 15 you will understand that there are two datasets and two different neural networks at play here. First a simple CNN that takes the live feed of the individual and gets the top two emotions of that individual. Second the Simple RNN that predicts the Mood of a song. Using both the Networks outputs we call a Recommend_Song() function that with the use of the emotions of the individual and mood of the song provides the top 5 Popular songs to uplift or improve the individuals mood.

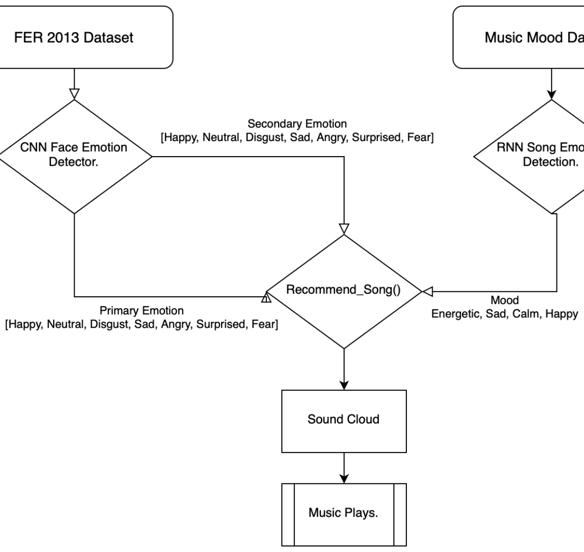

FER 2013 Dataset
Music Mood Dataset
CNN Face Emotion Detector.
Secondary Emotion [Happy, Neutral, Disgust, Sad, Angry, Surprised, Fear]
RNN Song Emotion Detection.
Recommend_Song()
Primary Emotion [Happy, Neutral, Disgust, Sad, Angry, Surprised, Fear]
Mood Energetic, Sad, Calm, Happy
Sound Cloud
Music Plays.

Figure 15: Flowchart for the Project.

The use of SoundCloud and the details of each unit will be explained as the sections progress. We have already seen the FER2013 Dataset and the CNN Face Emotion Detector. This section we will learn in depth about the Music Mood Dataset and the RNN Song Emotion Prediction.

4.2 Introduction to SoundCloud.

Before we enter the technicality of creation of the RNN Network, let us understand what is SoundCloud and why I have chosen Sound Cloud as our Music player and not created a Default Music Player.

4.2.1 Problems in Creating and Using a Default Music Player.

Creation of a Music Player that runs system independent is a simple and time efficient process. However, there were many minor details that could cause an hindrance.

1. Licenses: The simple task of Web Scrapping famous websites like Spotify or Google Music would be an efficient way to create a database of songs. However, playing the song on a device and distributing it would be a huge legal problem as the many songs have copyrights and are not allowed to be distributed freely.

2. Data Size: The data size of songs mainly mp3 will be of great sizes. Some long songs or EDM's which stand for Electronically Dance Music are more than 3 gigabytes in size. Transferring these songs and storing them would require a cloud storage and that would again be a security issue as this could be used to distribute the songs individually and again defeat the purpose of this project.

3. Speed of the Prediction: The prediction once commenced would take more than minutes to pull up the music from the database and play it in the Music Player, again this would again defeat the purpose of the music player causing more irritation to the user.

4.2.2 Pros of Using a Artist-First Platform.

SoundCloud is a platform established in the the year 2007. It is a Artist-First Platform that mainly means they empower independent artists and financially aid artists for their craftsmanship (Weissman; 2021).

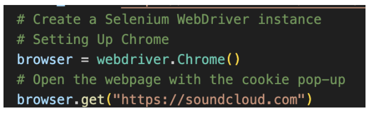
- All the music on Sound Cloud is free to use and available to everyone having the internet. Hence one of the major drawbacks of a Local Music Player of Licenses is eliminated.
- As API calls will be made to find and play the song, there will be no reason for storage and distribution issues to develop. Hence, another big drawback can be solved.
- As API calls take micro seconds to complete, time delay will be only hardware based and can easily be improved via software changes.
- Finally, the artists who are the creators of the songs will be given their recognition and reward and the user of this product will have a seamless music experience. A Win-Win situation for all.

After weighing all the parameters mentioned above, I decided to continue the Music Player based on a Artist-Based Platform to facilitate the smooth execution of the program and deliver the best possible outcome.

4.2.3 Creating the SoundCloud Music Player.

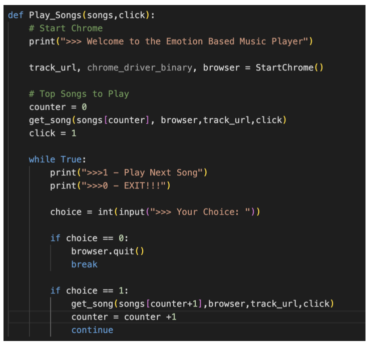
The SoundCloud web-application can easily be called by using the web-driver provided by Selenium and taking control of Google Chrome. It is an automated driver that allows us to pass API requests and get data and post data onto any website of our choosing.

1. First we need to set up Chrome. For this the location of the chrome driver is needed and a get request is called to get the website we need to visit, for our purpose its the home page of SoundCloud (refer 16a).
2. The track URL in fig.16a is the query we will append the name of the song we wish to play and pass once SoundCloud is up and running.
3. The function Play_Songs takes in two parameters songs and click, Click is the counter that facilitates the On Click Mouse operation in the code. It allows the music to be played and accepting privacy policies when pop-ups appear. The songs is a list of songs predicted, this is the inputs to the query to call in the songs from SoundCloud.
4. The Play_Songs function also provide the user with a continuous loop in which the user can easily switch from the current song to the next song by pressing the key 1. This is just an resemblance key for testing purpose and can be replaced by the next



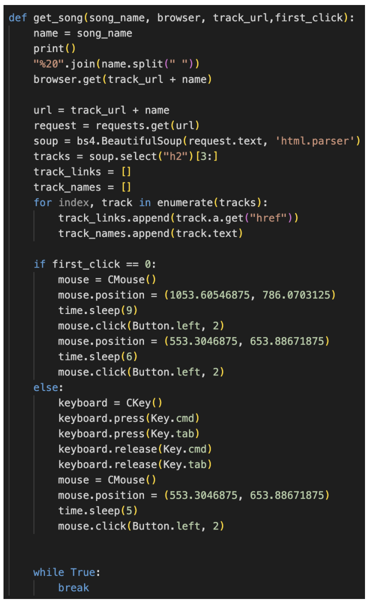
```python
# Create a Selenium WebDriver instance
# Setting Up Chrome
browser = webdriver.Chrome()
# Open the webpage with the cookie pop-up
browser.get("https://soundcloud.com")
```

(a) Setup for Google Chrome.



```python
def Play_Songs(songs,click):
    # Start Chrome
    print(">>> Welcome to the Emotion Based Music Player")

    track_url, chrome_driver_binary, browser = StartChrome()

    # Top Songs to Play
    counter = 0
    get_song(songs[counter], browser,track_url,click)
    click = 1

    while True:
        print(">>>1 - Play Next Song")
        print(">>>0 - EXIT!!!")

        choice = int(input(">>> Your Choice: "))

        if choice == 0:
            browser.quit()
            break

        if choice == 1:
            get_song(songs[counter+1],browser,track_url,click)
            counter = counter +1
            continue
```

(b) Play_Songs Function



```python
def get_song(song_name, browser, track_url,first_click):
    name = song_name
    print()
    "%20".join(name.split(" "))
    browser.get(track_url + name)

    url = track_url + name
    request = requests.get(url)
    soup = bs4.BeautifulSoup(request.text, 'html.parser')
    tracks = soup.select("h2")[3:]
    track_links = []
    track_names = []
    for index, track in enumerate(tracks):
        track_links.append(track.a.get("href"))
        track_names.append(track.text)

    if first_click == 0:
        mouse = CMouse()
        mouse.position = (1053.60546875, 786.0703125)
        time.sleep(9)
        mouse.click(Button.left, 2)
        mouse.position = (553.3046875, 653.88671875)
        time.sleep(6)
        mouse.click(Button.left, 2)
    else:
        keyboard = CKey()
        keyboard.press(Key.cmd)
        keyboard.press(Key.tab)
        keyboard.release(Key.cmd)
        keyboard.release(Key.tab)
        mouse = CMouse()
        mouse.position = (553.3046875, 653.88671875)
        time.sleep(5)
        mouse.click(Button.left, 2)


    while True:
        break
```

(c) Get_Songs Function.

Figure 16: Three Important Functions for Creating the Music Player.

button on the device. Fig. 16b also sets the Click to 1 as after the first run there are no cookie and permission requests to accept hence allowing us to only change between tabs and play music using the mouse click which is also automated.

5. With this we come to our final function namely the get_songs function that gets the name of the songs, passes the argument and switches between tabs using the keyboard shortcuts and clicks the play button on the webpage to play the song track. This can be visualised in Fig 16c

4.3 Creating the Music Recommender using RNN.

This simple machine learning technique is fairly easy to implement and basic to understand. Our Target Variable to classify will be the mood of each song, this will be carried out by considering the inputs or the features of each song. Weights will be assigned depending on the values of each feature and the prediction will be made.

To understand the modelling, prediction and working of the entire RNN network let us begin from the dataset and getting a deeper understanding of it.

4.3.1 Data Preparation.

With a total of 672 different artists, for the entire year of 2020 the Music Mood dataset also gives a numerical representation of the individual songs Popularity, length, danceability, loudness, energy, acoustics, instrumental, liveliness, valence, speech, temp and key. These are all used to predict the next column entry that being the "mood" of the song. To understand the selection process of each column a brief understanding of each column is needed and that is listed below.

- Columns Selected.

1. Popularity: This gives a numerical representation of the up votes received divided by down votes in the spotify application for the particular song. Hence, the larger the number the more famous the song.
2. Dancebility: This value gives a numerical representation of the beats and lyrics present in the song that decrees the song suitable to dance upon.
3. Acousticness: This value gives a numerical representation of the nature of the song, if it is loud and musical or silent and lyrical. Higher the value more louder and musical the song.
4. Energy: This value gives the beats per seconds in a numerical form. The more beats/sec, the more energetic the song hence the higher this value tends to be.
5. Instrumentalness: This value gives a numerical understanding of how instrumental based the target song is. Lower this value more soulful and single toned music the song is.
6. Liveliness: This value is very similar to Energy however this also takes into account lyrical count and pitch. Higher Energy with higher pitch will cause to give a higher Liveliness score.
7. valence: Higher the electronic music involved in the song, higher the valence representation.
8. Loudness: This score represents the overall pitch of the song.
9. Speechness: This score represents the lyrics per second. A son with more words per second for instance a rap song will have a higher score than melodious soulful song.
10. Tempo: This numerical value represents the Tempo or the Pace of the song.
11. Key: This value represents the musical note or the key the song is recorded in.
12. TimeSignature: This timestamp gives the day month and year of the song recorded and published to the people.

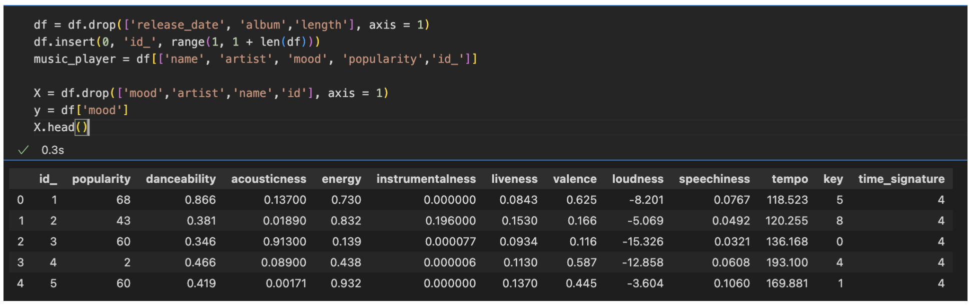
This can be done by referring to the code snippet in Fig 17

- Columns Dropped.

1. Album: This string value gives the album the songs belongs to. For our purpose we need no reference to the album as we wish to predict the mood depending on the song itself and not the package or album it comes in. Also one-hot encoding of this table would stretch the dataset unnecessarily. Hence, to avoid complexity and the sheer uselessness of this data i have dropped this column.
2. Release Date: This timestamp gives us just the release date of the song. This is a repetitive value in comparison to the time signature chosen which is more descriptive and precise. Hence, this column data is dropped.
3. length: This numerical value gives us the length of the song, again this data is not needed for our emotion based recommender hence is dropped.

- Target Variable.

1. Mood: This is our target variable column that is mainly categorical and are in 4 categories namely: 'Happy','Sad','Energetic' and 'Calm'. Our goal is to predict the mood depending purely on the attributes of the song. These can be seen in Fig(18)



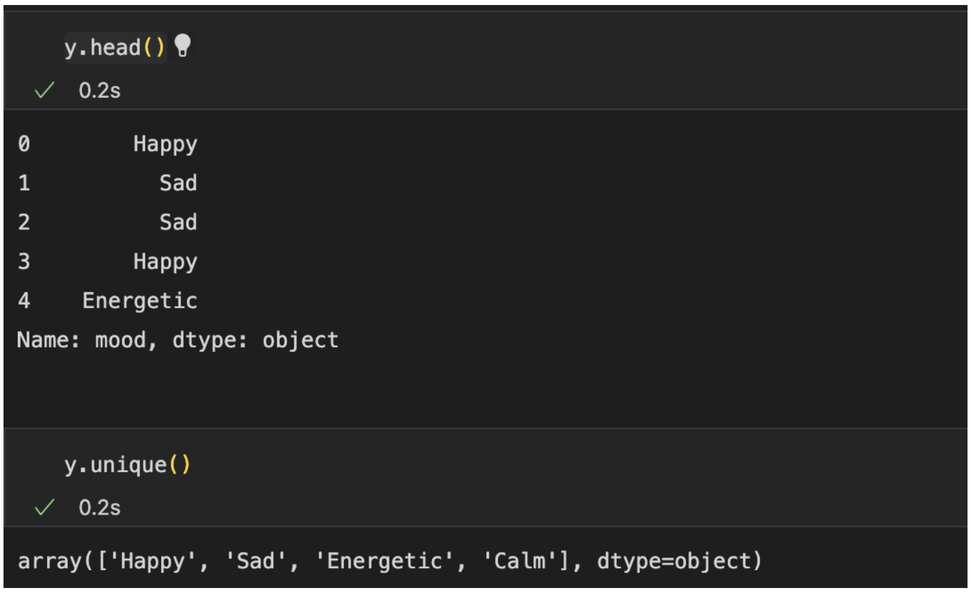
```
df = df.drop(['release_date', 'album','length'], axis = 1)
df.insert(0, 'id_', range(1, 1 + len(df)))
music_player = df[['name', 'artist', 'mood', 'popularity','id_']]

X = df.drop(['mood','artist','name','id'], axis = 1)
y = df['mood']
X.head()
```

✓ 0.3s

	id_	popularity	danceability	acousticness	energy	instrumentalness	liveness	valence	loudness	speechiness	tempo	key	time_signature
0	1	68	0.866	0.13700	0.730	0.000000	0.0843	0.625	-8.201	0.0767	118.523	5	4
1	2	43	0.381	0.01890	0.832	0.196000	0.1530	0.166	-5.069	0.0492	120.255	8	4
2	3	60	0.346	0.91300	0.139	0.000077	0.0934	0.116	-15.326	0.0321	136.168	0	4
3	4	2	0.466	0.08900	0.438	0.000006	0.1130	0.587	-12.858	0.0608	193.100	4	4
4	5	60	0.419	0.00171	0.932	0.000000	0.1370	0.445	-3.604	0.1060	169.881	1	4

Figure 17: Input Columns with ID variable.

With this detailed understanding of the columns taken into consideration and the columns dropped, we now move on to a much deeper understanding of how the rows are mapped. As we have not used the names of the song again due to one-hot encoding issue as our prediction is of a regression-classifier, we have no logical connection between the song attributes and names. Hence, to solve this i introduce an id_ variable that creates a simple id count for each row. This can help us map the name of the predicted row to the attributes of the predicted row and the target variable being the mood of the song. The code snippet in Fig 17 gives you the clear understanding of how this is accomplished and Fig 18 shows you the Target variable to be predicted and its unique values or unique classes.



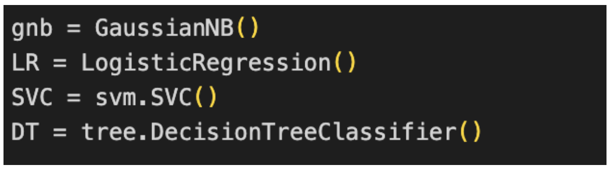
```
y.head()
✓ 0.2s

0        Happy
1          Sad
2          Sad
3        Happy
4    Energetic
Name: mood, dtype: object

y.unique()
✓ 0.2s

array(['Happy', 'Sad', 'Energetic', 'Calm'], dtype=object)
```

Figure 18: Output or Target Column.

4.3.2 Modelling.

Once the preparation is completed we can move onto modelling. I will be using multiple regression models over this dataset and judge their performance by the accuracy of each of the models. The best model proven to be is by a very slim margin. A few minute basic steps of Outlier detection and Removal of Duplicate or N/A values are skipped as this dataset has no such irregularities. We will be including 4 models here, namely

1. Gaussian Naive Bayes.
2. Logistic Regression.
3. SVC or as commonly known Support Vector Classifier.
4. Decision Tree Classifier.

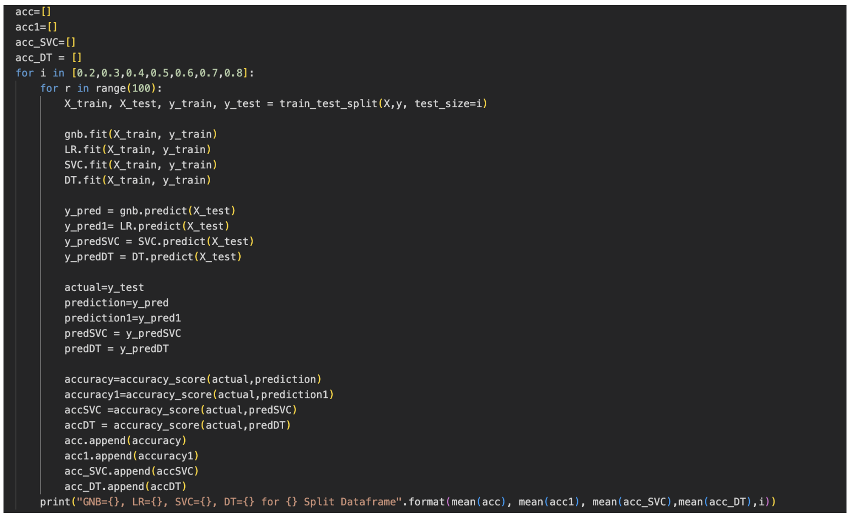


```
gnb = GaussianNB()
LR = LogisticRegression()
SVC = svm.SVC()
DT = tree.DecisionTreeClassifier()
```

Figure 19: Models used.

Amongst these Gaussian Naive Bayes surprisingly out-performs each and every other model. For designing the model, i have created 2 empty lists for accuracy, one that holds all the accuracy values for the current model under test and feeds the average of the accuracy after 100 Monte-Carlo runs into the next list that holds the average of each and every other model after the 100 Monte-Carlo runs.

A loop is called that runs this code 100 times, in this loop the below mentioned instructions are carried out in order 100 times to see the performance of the model at different scenarios with different inputs and gauge by returning an average accuracy throughout all the 100 unique and different scenarios.



```
acc=[]
acc1=[]
acc_SVC=[]
acc_DT = []
for i in [0.2,0.3,0.4,0.5,0.6,0.7,0.8]:
    for r in range(100):
        X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=i)

        gnb.fit(X_train, y_train)
        LR.fit(X_train, y_train)
        SVC.fit(X_train, y_train)
        DT.fit(X_train, y_train)

        y_pred = gnb.predict(X_test)
        y_pred1= LR.predict(X_test)
        y_predSVC = SVC.predict(X_test)
        y_predDT = DT.predict(X_test)

        actual=y_test
        prediction=y_pred
        prediction1=y_pred1
        predSVC = y_predSVC
        predDT = y_predDT

        accuracy=accuracy_score(actual,prediction)
        accuracy1=accuracy_score(actual,prediction1)
        accSVC =accuracy_score(actual,predSVC)
        accDT = accuracy_score(actual,predDT)
        acc.append(accuracy)
        acc1.append(accuracy1)
        acc_SVC.append(accSVC)
        acc_DT.append(accDT)
    print("GNB={}, LR={}, SVC={}, DT={} for {} Split Dataframe".format(mean(acc), mean(acc1), mean(acc_SVC),mean(acc_DT),i))
```

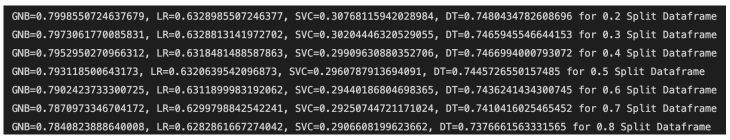
Figure 20: Monte-Carlo Runs with Variable Data Split Size.

- train_test_split is called: This splits the input and output dataframe under consideration randomly. As i have mentioned the split to be at 0.5, it means 50% of the dataframe will be considered for Training and the other 50% is considered for test purpose. However, this split will be completely random and different for all 100 Monet-Carlo runs. To find the best model with the best fit, i have created an outer loop that provides me with a variable ranging from 0.2 to 0.8 with a step of 0.1. This value is then used to set the split size of the dataframe, hence gaining 100 Monet-Carlo runs over the dataframe split differently. Hence, not just ensuring the best accuracy for a Dataframe split differently but also with different dimensions and picking up the best split with the best model. This is found to be Gaussian Naive Bayes for a split of 0.2 (refer fig(21)

- mode.fit is executed: This will fit the split dataframe onto the model of our choice, here for all the four above mentioned models. Hence by doing this during every run each model will be put through the same dataframe split and will be measured on the same conditions.

- model.pred is called and the value is stored: The .pred operation seeks the model to give a prediction based on each of their Mathematical Framework. Again this prediction will be carried out with the same Test Samples for each and every model ensuring the same conditions and inputs for each during the test. This prediction is then stored and used for accuracy calculation.

- accuracy_score is invoked to compare between True and Predicted Values: This is done to calculate the Root Mean Square Error between the actual and predicted value and to give us an understanding by providing us with the accuracy of how accurately each model predicts.



```
GNB=0.7998550724637679, LR=0.6328985507246377, SVC=0.30768115942028984, DT=0.7480434782608696 for 0.2 Split Dataframe
GNB=0.7973061770085831, LR=0.6328813141972702, SVC=0.30204446320529055, DT=0.7465945546644153 for 0.3 Split Dataframe
GNB=0.795295027096312, LR=0.6318481488587863, SVC=0.29909630880352706, DT=0.7466994000793072 for 0.4 Split Dataframe
GNB=0.793118500643173, LR=0.6320639542096873, SVC=0.2960787913694091, DT=0.7445726550157485 for 0.5 Split Dataframe
GNB=0.7902423733300725, LR=0.6311899983192062, SVC=0.29440186804698365, DT=0.7436241434300745 for 0.6 Split Dataframe
GNB=0.7870973346704172, LR=0.6299798842542241, SVC=0.29250744721171024, DT=0.7410416025465452 for 0.7 Split Dataframe
GNB=0.7840823888640008, LR=0.6282861667274042, SVC=0.2906608199623662, DT=0.7376661563331565 for 0.8 Split Dataframe
```

Figure 21: Accuracy according to Models and Different Test Train Split.

5 Implementation and Model Integration

5.1 Integration: Facial Recognition and Music Recommender.

In this chapter we will be connecting the building block together. On one side we have the Facial Emotion Recognition by CNN and on the other we have the Song Mood Prediction.

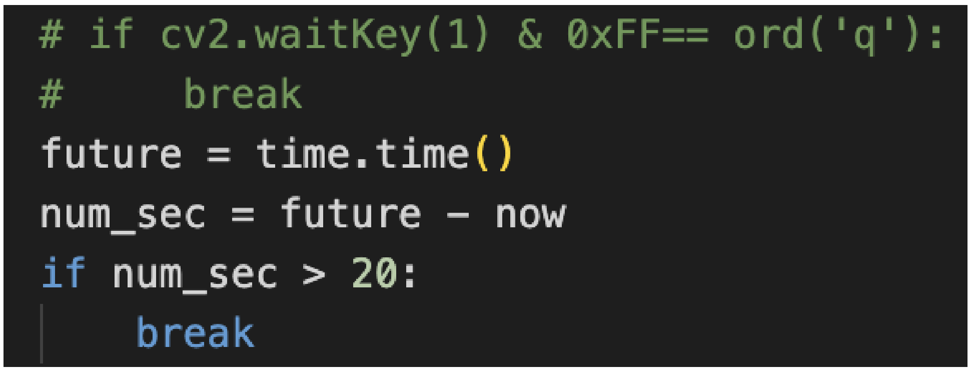
6 Evaluation

The Evaluation will be completed by commenting about the project execution, time efficiency and dependency, final output display and integration ease. All of these evaluations will be detailed when it comes to technical requirements and dependencies.

6.1 Project Execution.

The primary goal of creating this project was to completely automate a music player that takes in user input for a set frame time and returns music to the user without any user interference. The only interference the user needs to involve himself is in:

- Quitting the Face Emotion Reading: This happens normally when the user wants to stop his emotions being read and is ready to play the music, he or she needs to press and release the button 'q'. This is setup just for testing purposes where many more emotions can be captured over variable time spans. However, a limited time window and expression capture is easily possible by just a slight change in the cv2 exit statement and can be seen in fig 22. The commented area in this python code is the original piece of instruction where we can quit the program by pressing the 'q' button (Bradski and Kaehler; 2008). However, by setting two parameters 'now' and 'future' we can manipulate the closing of the emotion detector.



```
# if cv2.waitKey(1) & 0xFF== ord('q'):
#      break
future = time.time()
num_sec = future - now
if num_sec > 20:
    break
```

Figure 22: Modification to close Emotion Reading in 20 seconds.

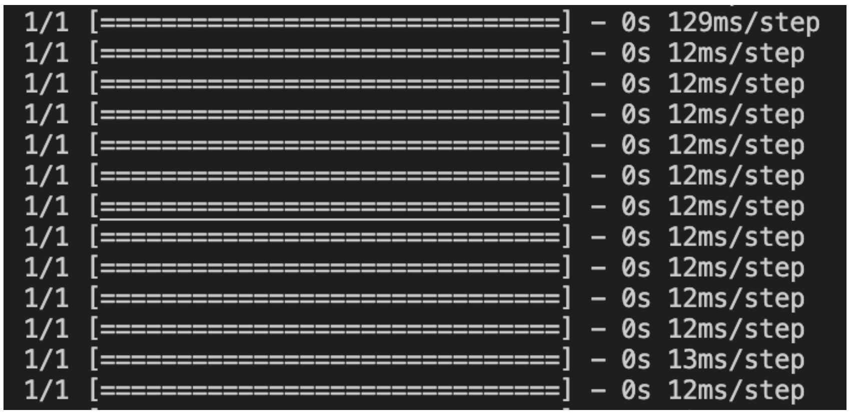
  The first variable 'now' is initiated when the camera is turned on, for a basic Macbook Air with a 8 gigabyte random access memory the wait time for the camera to turn on is 17ms, hence it is significantly fast and can be neglected to instantiate this variable i call the time.time() function that basically captures the time the camera started. Once this is done and the recording has begun we instantiate the future by saying future = time.time() after the first frame is captured, this is checked by the if condition after each and every frame. The moment all the frames until 20 seconds are completed, the escape condition is triggered and the code breaks out of loop causing the camera to turn off and all active windows are then destroyed.

- User input to Quit the entire program or Skip to next song: A User input is required for the end of the usage of the program or to skip the current song and play the next immediately. To be honest, this can not be automated or modified as the user input is required to make such a decision. One way of automating it could be to keep the emotion detection running and on countering a disgust emotion for a certain period of time turning off the music player or skipping the song. The problem of this could be, the user might be influenced not by the music player but by another colleague or person around causing the emotion of disgust to be prominent and causing an error in the algorithm. To void this complexity, i have selected a simple user input of 1- To play next song and 0 - To Terminate the music player.

Except these two above mentioned conditions the entire Song Player is automated and the experience is seamless, with just a maximum of 30 seconds to gather emotions from the user and 10 seconds to play the first song.

6.2 Time Efficiency and Dependency.

The Face Recognition has a max frame rate of 1ms per frame. Keeping in mind system and programming limitations, the frame-rate is still a surprisingly fast and constant 12ms/step with a starting delay of 129ms/step which is system dependent (fig.23). This can be normalised by integrating the functionality code with other systems and cross platform testing however, i have no control over the hardware built up of a system.



```
1/1 [==============================] - 0s 129ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 12ms/step
1/1 [==============================] - 0s 13ms/step
1/1 [==============================] - 0s 12ms/step
```

Figure 23: Emotion Capture Frame-rate.

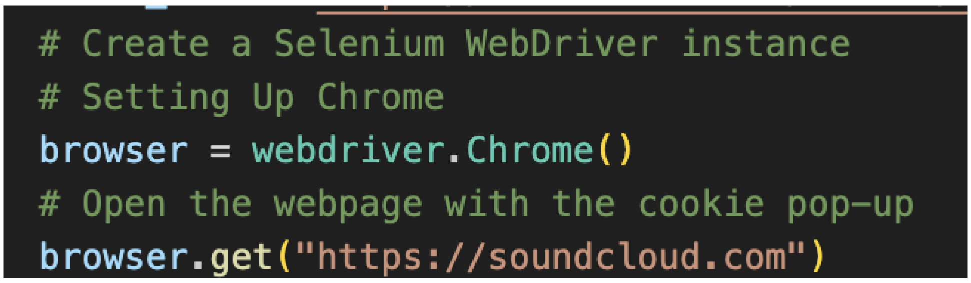
The time dependency of the entire system from initiating the run to playing the first song is a maximum of 43 seconds and a fastest of 39 seconds. Excluding the 30 seconds window of the camera being turned on to get the users input, i would boldly say 9 to 12 seconds to interpret and produce music is quite an achievement.

6.3 Final Output and Display.

As spoken of in length the final output of this project is a webpage namely sound-cloud an Artist First Application that featured over 10 thousand plus artists independent and under labels. As it is an Open Source and free applications i hinder no copyrights or cause any damage to any individuals intellectual property. The use of this application is currently designed for a laptop with internet connection however, the SoundCloud application is available on Android and iOS mobile devices too. This gives a future scope to this project of Integrating this project to these Applications and utilizing their search engines to enhance ones experience and also grow my own database.

6.4 Integration Ease.

The runner function is a simple Micro Service based architecture which calls all the different building blocks in an orderly fashion. Hence, if one wishes to reproduce my work they would only need to download these pieces of code and the datasets and run the code. The entire project is build on Python 3.9.12 Version and it being the latest stable release there would be no issues in recreating the entire project from start to end. As the weights are saved in an h5 file, retraining the model and spending over 7 hours users can just use the existing weights and model to run the entire code in just 39 seconds! SoundCloud being available on the normal web page another dependency is to use a chrome browser.



```
# Create a Selenium WebDriver instance
# Setting Up Chrome
browser = webdriver.Chrome()
# Open the webpage with the cookie pop-up
browser.get("https://soundcloud.com")
```

Figure 24: Changes in Selenium to avoid Browser dependency.

However, selenium supports all browsers and a slight change in the code shown in Fig. 24. Providing the executable file location of the preferred Browser and calling the respective Web-Driver this problem can be easily solved. Other than this there maybe a few import related irregularities however all are available and can be downloaded by simple python based pip arguments.

7 Conclusion and Future Work

7.1 Difficulty in Emotion Recognition.

Emotion Detection using facial analysis is a thought that isn't something new, in-fact (Liu; 1918) worked on something really similar in the year 1918! a small experiment to analyse faces in a classroom which had a section dedicated to how attendance in classrooms can then be taken automatically, also a general interest level of students can be gauged. This did inspire many individual contributors that went on to take this idea to different fields, like young Kokkalis (1919) whose thesis project was an implementation of this very same idea but for the border patrol of his country, which was then surely adopted by the Naval Bases in Monterey.

So this idea is almost a decade old, however, success in this field hasn't been of great lengths. The speed of Face Recognition and the capacity of faces recognised have grown immensely, however minute facial details that describe complex emotion like Love or Depression require a specific environment that isn't economically or dimensionally feasible for many conditions.

In my project the major finding and one of my drawbacks was the sheer complexity of detecting Love as an emotion using only facial analysis. The emotion of Love using a simple Laptop Camera or a above average mobile device camera is just not possible as Iris analysis and Pupil dilation needs to be measured as well which is impossible with a regular compact camera.

The work of (Soleymani et al.; 2016) is complete when it comes to detecting emotions changing with time, however a heavy part of their prediction is based on the EEG signals and facial analysis is used to corroborate the data incoming. Without the EEG signals complex emotions like Love, Depression or Tiredness/Fatigue could not be predicted.

Hence, this limitation of codependency on different technology requirements limited this project scope to simple emotion detection. However, with improvement of camera quality and sharpness of software upgrades in computer vision this is soon achievable.

7.2 Over-fitting Problem Solution.

Over-fitting has always been a problem in Deep Learning, in simple terms Over-fitting is a condition when the model is over trained on a particular dataset such that it introduces a robustness in the prediction such that any change in the dataset or any new item to predict would cause a serious problem for the model and the trained weights would not perform as it should.

For the facial analysis on the FER-2013 dataset, as the models are very advanced now the dataset of over 10 thousand images too were very few and reached an early Over-fitting in just under 4 hours of training.

With no constant monitoring in place this model would easily produce an accuracy of over 90%, however making itself completely robust in the process. To avoid constant monitoring that directly applies strain on the processing system one would require a much larger dataset and propose stopping condition based on time complexity or accuracy requirements, this in turn would converge to equivalent stress on the system as constant monitoring. Hence, the Key Finding here is the trade-off experienced irrespective of the decision taken.

I have chosen to proceed with a smaller dataset as larger dataset proposes a difficulty of storage, and my choice of designing needed this project to be quick and light weighted that can easily be integrated for software collaboration with pre established softwares like Spotify or Google SoundCloud. Hence, an accuracy using VGG-16 modified of 79% is acceptable for my purpose.

7.3 Screen Control Using Mac Os.

System dependency is always proves a bad designing technique. My project has a system based dependency as well when it comes to tab changes and mouse clicks. This is something i encountered and tried to find an alternative with no success. Mac Os having different commands and mouse controls due to its operating system design caused me to adopt different single based solution of using controls by the operating system commands.

However, the solution for this was provided by pynput. This solution i found in the work of (Chua et al.; 2022) where the use of pynput was extensively displayed to control a Personal Desktop by hand gestures that were recorded using Proximity and other similar electronic devices. Chua et al. also integrated a proximity sensor to completely write a sentence on the Microsoft Word application. Therefore, i used a similar approach to complete the tab switches from the current running window to the SoundCloud page, after which with a hard coded time delay of 6seconds simultaneous mouse click would occur to play the songs and accept required security and cookie recommendations. The upside of using pynput is that it actually allows the same instructions to work on all other operating systems and platforms, may it be Mac OS, Red Hat, Windows or Linux.

The end product was a completely environment friendly and System Independent cross verified product that works on any system seamlessly. However, it has to be noted Internet connection dependency due to latency and Processing Input of a system can still be dependencies causing a requirement of higher hardware wait or delay time.

7.4 Neutral Emotion Issue Solution.

The emotions in this project are of 7 different categories ie. Happy, Sad, Neutral, Disgust, Angry, Fear and Surprised. Each emotion has a distinct characteristic and can be improved or influenced with a music of a certain type. The music recommendation is done by 4 emotions namely Sad, Happy, Energetic and Calm.

The problem rises where the major emotion detected is neutral. A happy song might not end up influencing the individual majorly if the individual is already happy but his emotion during the 20 seconds window was majorly neutral. Similarly, a Energetic song may not be the best option if the user has a neutral face but feels Angry in reality.

To address this issue i have selected not one major emotion but two instead. Hence even if neutral is detected for most of the time in the time span when the camera was capturing its input there will be a secondary emotion that will also be captured. This gives me and my system a fair understanding of what that individual is feeling. The major change is setting up the primary emotion and i have taken the design liberty to always swap the first emotion with the secondary emotion if and only if Neutral emotion is detected as the Primary emotion. This project has an immense potential in my opinion. With the limited resources and a time span of 5 months i could complete the ground work of a simple Emotion based Music Player. However, with further improvement this could be an excellent contribution to multi-million dollar industries that are in search for innovative and simple software upgrades to enhance user experience and in return be cost efficient as well. Which i think this project has potential to be and have proposed multiple ways it could be in the next section.

8 Further Improvement.

Now that the project is commenced i can provide a few inputs of my own that can help improve this music player and help create a much Versatile, efficient and sophisticated.

8.1 Improving Versatility.

Versatility is the ability to adapt. AI and Machine Learning are built on the concept of adaptability over robustness. Prediction and Classification based on not rigid but dynamic data. My Music recommendation is build on a dataset with just over 12 thousand inputs including music of the year 2020 only. This can be a starting point to increase the versatility of the project where a larger dataset encompassing song data over a decade with over a thousand different artists can be used. Similarly, the use of a larger more updated and defined dataset instead of the FER2013 can be used to train the weights of the Image Processing making the facial recognition more accurate and precise.

8.2 Improving Efficiency.

Efficiency of any process in an environment is of the utmost importance in this digital era. With the introduction of lightning speeds of internet connections the bottleneck in many cases arises at the software and hardware domain. Hardware dependencies can easily be minimized by software efficiency, for instance image capturing and sharpening of images are two different things, however an IPhone with under 20 megapixel can capture far superior images that any other phone with the same or even slightly higher mega pixels due to the software processes the capturing of the image goes through. Similarly, my Automated Playlist generator can be more efficient if instead of using VGG-16 modified version one would introduce Resnet V2. The Resnet V2 is trained on more than a million images from the ImageNet Database and it classifies the images into 1000 categories. These are similar properties compared to VGG-16 however with a great difference and that is VGG-16 has only 16 deep layers while Resnet has 50 deep layers and is a much accurate and sophisticated model.

8.3 More Sophisticated.

As seen in the above section one of the ways to improve the sophistication is to improve the Image Detection algorithm from VGG-16 to Resnet 50. Another revolutionary idea would be to create a self sustained music player and eradicating the requirement of a cloud based music player all together. The reason this couldn't be accomplished was mentioned in Section 5.2.1 however, if the right Licenses could be attained it wouldn't be a difficult task at all. The entire model could be build on a cloud service like Amazon Web Services or Google Cloud Platform with all the data and songs stored on the cloud itself and a simple web application ready to be deployed. This would make the entire system freely available and enrich the music player with a sophisticated edge. API calls can be made from the web application itself if SoundCloud interface needs to be still kept intact, Heroku provides excellent service in this manner. Another improvement could be a collaboration with other 3rd part softwares for instance Google Music or Spotify itself that has a much bigger song dataset with better quality of songs and options for live lyrics or even Karaoke sing along. Although gaining licences and API calls to Spotify come at a cost, this can be achieved to make the product more user friendly and sophisticated to use. Finally compactness of the music player is of high importance, hence uploading this entire system on an Android or iOS device would definitely be the next step where instead of the web camera to capture the image the cellular device front camera would be used and then the suitable browser shall play the songs recommended.

9 Overall Output: Critique.

As the author Maraboli says in his book (Maraboli; 2020) 'Once your Mindset changes, everything on the outside changes along with it.' what i have done here is to influence ones mindset for reaching a stage of focus and aspiration with the use of a simple technique.

An individual on daily bases as per Howarth uses their smartphone for an average of 3hours and 15minutes, an individual checks his/her phone 58 times and an average music application use time of 37 minutes is recorded. If this 37 minutes would be spent during commute from one place to another, or before an important exam, or at a Gymnasium for motivation and the individual could receive the perfect or close to perfect song they desire by their Music Application that exactly fits their mood or emotion and the following songs just keep on improving their mindset, wouldn't it be an experience or a moment to savor? The answer to this question is in my work. The provision of a seamless experience with a mindset alteration that improves the well-being of an individual and provides the listener the inspiration they are looking for.

By providing a full fledged self dependent application that takes in user input on the go, and uses this dynamic data to predict the song they desire and automatically with no interference play the music that could improve their mindset and self learn the changes in their emotions to rectify the algorithm, I have created a recommender mostly all music providers are seeking to perfect, and my work is just a different approach to meet this requirement and make our technology a trustful and safe environment for everyone.

References

Ahonen, M. (2014). Ancient physiognomy, *Sourcebook for the History of the Philosophy of Mind*, Springer, pp. 623–631.

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*, " O'Reilly Media, Inc.".

Browne, J. (2018). Darwin and the face of madness, *The anatomy of madness*, Routledge, pp. 151–165.

Chua, S., Chin, K., Lim, S. and Jain, P. (2022). Hand gesture control for human–computer interaction with deep learning, *Journal of Electrical Engineering & Technology* **17**(3): 1961–1970.

Darwin, C. and Prodger, P. (1998). *The expression of the emotions in man and animals*, Oxford University Press, USA.

Devoogdt, D., De Turck, F. and Dhoedt, B. (2019). Session-based music recommendation with recurrent neural networks, *Proceedings of the 14th International Conference on Network and Service Management (CNSM)*, IEEE, pp. 1–9.

He, X., Deng, K., Wang, X., Li, Y. and Gong, Y. (2018). Neural collaborative filtering vs. matrix factorization revisited, *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM)*, ACM, pp. 677–686.

Howarth, J. (2022). Time spent using smartphone.
URL: *https://explodingtopics.com/blog/smartphone-usage-stats*

Kokkalis, K. (1919). *Contribution of Artificial Intelligence to Border Security*, PhD thesis, Monterey, CA; Naval Postgraduate School.

Krysik, A. (2022). What is the secret of spotify's recommendation system success?
URL: *https://recostream.com/blog/what-is-the-secret-of-spotify-recommendation-system-success*

Landis-Shack, N., Heinz, A. J. and Bonn-Miller, M. O. (2017). Music therapy for posttraumatic stress in adults: A theoretical review., *Psychomusicology: Music, Mind, and Brain* **27**(4): 334.

Liu, M. (1918). *Video Labeling System for Face Analysis on Classroom Videos*, PhD thesis, Worcester Polytechnic Institute.

Luo, J., Xie, Z., Zhu, F. and Zhu, X. (2021). Facial expression recognition using machine learning models in fer2013, *2021 IEEE 3rd International Conference on Frontiers Technology of Information and Computer (ICFTIC)*, pp. 231–235.

Maraboli, S. (2020). Making the change, *Conflict Recovery: Cultural Humility and Civility in Education* p. 43.

Mollahosseini, A., Hasani, B. and Mahoor, M. H. (2019). Affectnet: A database for facial expression, valence, and arousal computing in the wild, *IEEE Transactions on Affective Computing* **10**(1): 18–31.

Pise, A. A., Alqahtani, M. A., Verma, P., Karras, D. A., Halifa, A. et al. (2022). Methods for facial expression recognition with applications in challenging situations, *Computational Intelligence and Neuroscience* **2022**.

Rice, L. and Wong (2020). Overfitting in adversarially robust deep learning, *International Conference on Machine Learning*, PMLR, pp. 8093–8104.

Sadikoğlu, F. M. and Idle Mohamed, M. (2022). Facial expression recognition using cnn, *2022 International Conference on Artificial Intelligence in Everything (AIE)*, pp. 95–99.

Sharma, V. P., Gaded, A. S., Chaudhary, D., Kumar, S. and Sharma, S. (2021). Emotion-based music recommendation system, *2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 1–5.

Soleymani, M., Asghari-Esfeden, S., Fu, Y. and Pantic, M. (2016). Analysis of eeg signals and facial expressions for continuous emotion detection, *IEEE Transactions on Affective Computing* **7**(1): 17–28.

Suprema (2021). Smart facial recognition terminal: Facestation 2.
URL: *https://www.supremainc.com/en/hardware/face-recognition-terminal-facestation2.asp*

Weissman, M. (2021). Soundcloud helpdesk.
URL: *https://help.soundcloud.com/hc/en-us/articles/115003570488-What-is-SoundCloud-*