Instructions: Replace all bracketed text with text for your project. Feel free to add. Leave the document structure in place. Delete these instructions.

[Hadi Baajour, 96316680]
CIS4930 Individual Coding Assignment
Spring 2023

1. Problem Statement

One of the most ground-breaking inventions created in the 21st century is voice assisted devices seen in almost every home you walk into. Companies like Amazon, Google, and Apple have created their own voice activated devices to help people access information on the internet quicker and easier than before.

A problem faced by voice activated devices is that sometimes these devices are not trained enough to detect the emotion of the individual speaking to it. This may be a problem as the device would not know how to properly respond to the person correctly depending on their emotion when asking the question.

In order to solve this problem, we must create a machine learning model which is trained with different audio files that have their corresponding emotion passed in with it. Emotions can be anger, sad, happy, and fear and passing them into a model will help us create a device that can detect human emotion through their speech. This will help in significantly improving the quality of answers given by voice activated devices.

2. Data Preparation

To create the model, we must first analyze the data being passed through and ways to clean it up before extracting features. Firstly, I started off by splitting the data from the main folder into training and testing folders. I split the data by 70-30, where 70% was used for training and 30% was used for testing. This was done for each of the four emotions, and they were all split up into their respective folders.

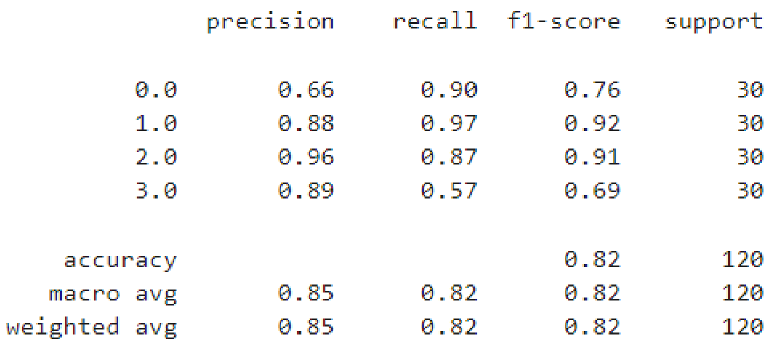
Next, I began exploring the data to analyze different aspects of the audio files that could be useful for me to know for future development. This included finding out whether the data was split appropriately, graphing the time and frequency domain graphs for certain audio files, and listening to random audio files from the four emotions to better grasp how each audio file sounds. Listening to the audio files was done by importing the IPython.display library and using its Audio() function to display it.

After taking a look at the audio files and their respective graphs, I began extracting the acoustic features from each audio file. I did this by concatenating each emotions audio file path to one list and their respective results on another. I then passed them into a function called features() which I created that loops through the first list and *uses each path to extract each of its audios features. It makes separate data frames for features such as loudness, mel-frequency cepstral coefficients, zero crossing rate, chroma, and mel spectrogram and then concatenates all of these features into one big data frame (I used the librosa library to extract these features). After creating the data frame, we alter the scale of each column to be from a range of -1 to 1 using the sklearn preprocessing library called MinMaxScaler. From there we take the average of each column, convert it to a numpy array, and append it to a list named matrix which will be passed into our model.*

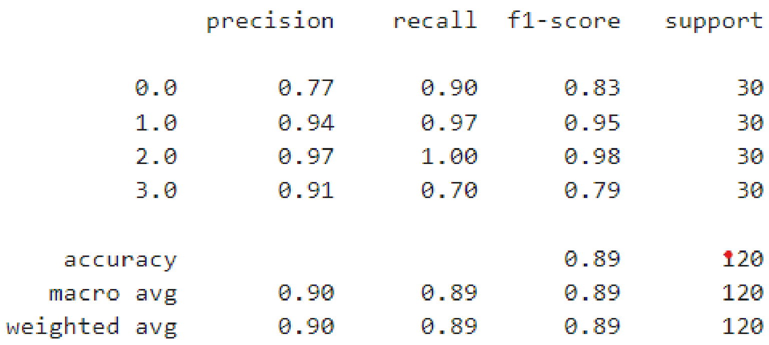
3. Model Development

- Model Training
    - *For the training phase, I trained three different models: Support Vector Classifier, Gaussian Naïve Bayes, and Random Forest Classifier. I used these three models specifically because they work best when it comes multi-class classification problems. SVC, Gaussian NB, and RFT can handle non-linear decision boundaries, while LR assumes a linear relationship between the input features and the output variable. If the relationship between the input and output variables is non-linear, then SVM, Random Forests, and Gaussian NB might provide better results.*
    - *Splitting training and testing data was done previously and explained above.*

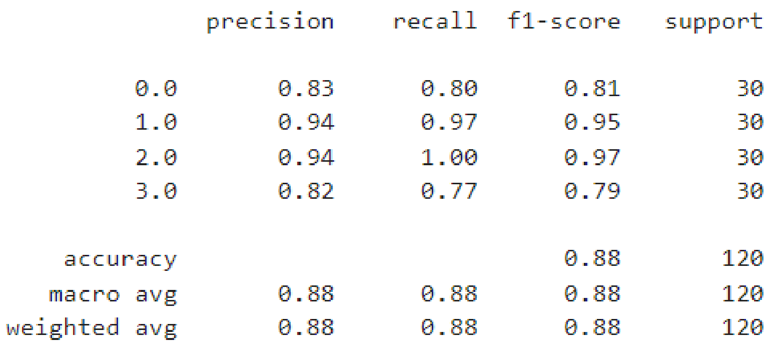
- Model Evaluation



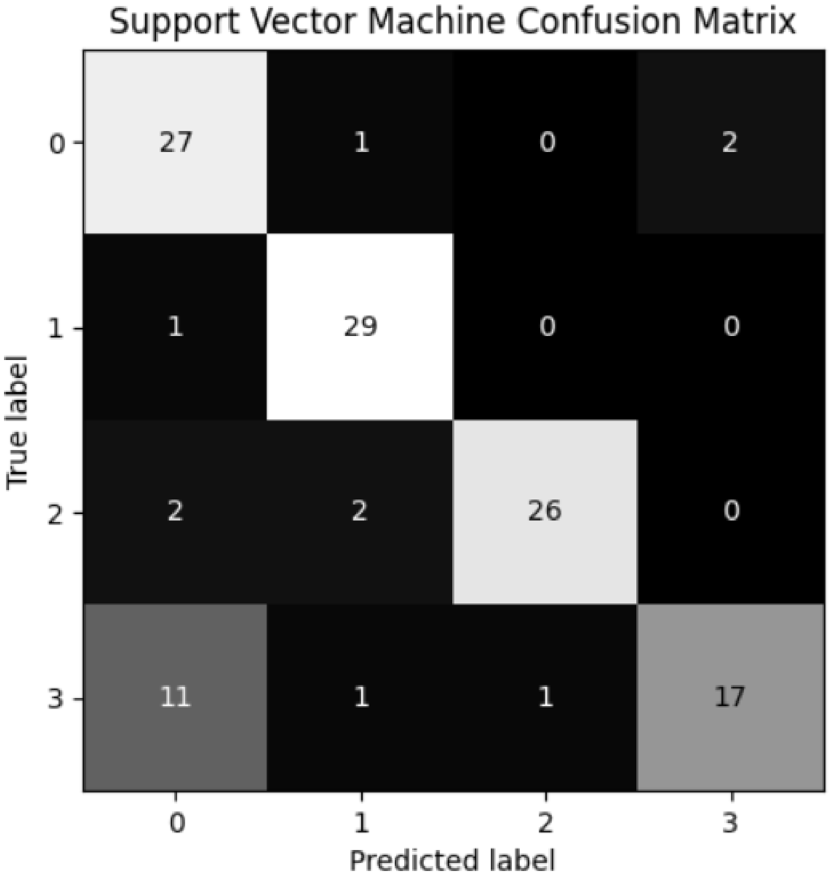
```
              precision    recall  f1-score   support

         0.0       0.66      0.90      0.76        30
         1.0       0.88      0.97      0.92        30
         2.0       0.96      0.87      0.91        30
         3.0       0.89      0.57      0.69        30

    accuracy                           0.82       120
   macro avg       0.85      0.82      0.82       120
weighted avg       0.85      0.82      0.82       120
```



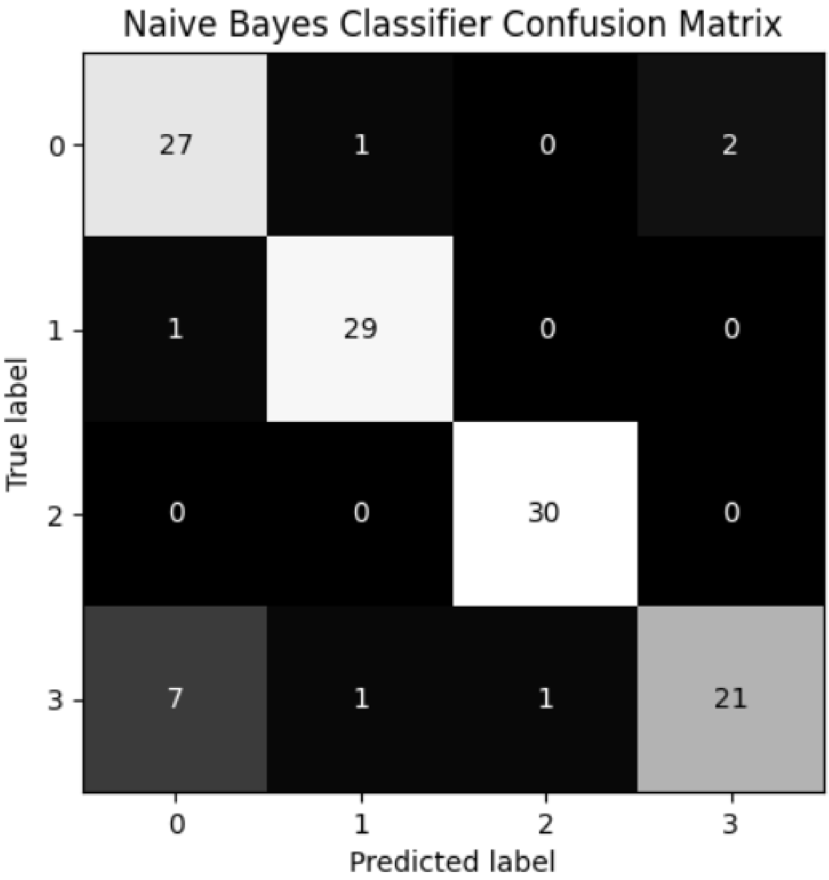
```
              precision    recall  f1-score   support

         0.0       0.77      0.90      0.83        30
         1.0       0.94      0.97      0.95        30
         2.0       0.97      1.00      0.98        30
         3.0       0.91      0.70      0.79        30

    accuracy                           0.89       120
   macro avg       0.90      0.89      0.89       120
weighted avg       0.90      0.89      0.89       120
```



```
              precision    recall  f1-score   support

         0.0       0.83      0.80      0.81        30
         1.0       0.94      0.97      0.95        30
         2.0       0.94      1.00      0.97        30
         3.0       0.82      0.77      0.79        30

    accuracy                           0.88       120
   macro avg       0.88      0.88      0.88       120
weighted avg       0.88      0.88      0.88       120
```

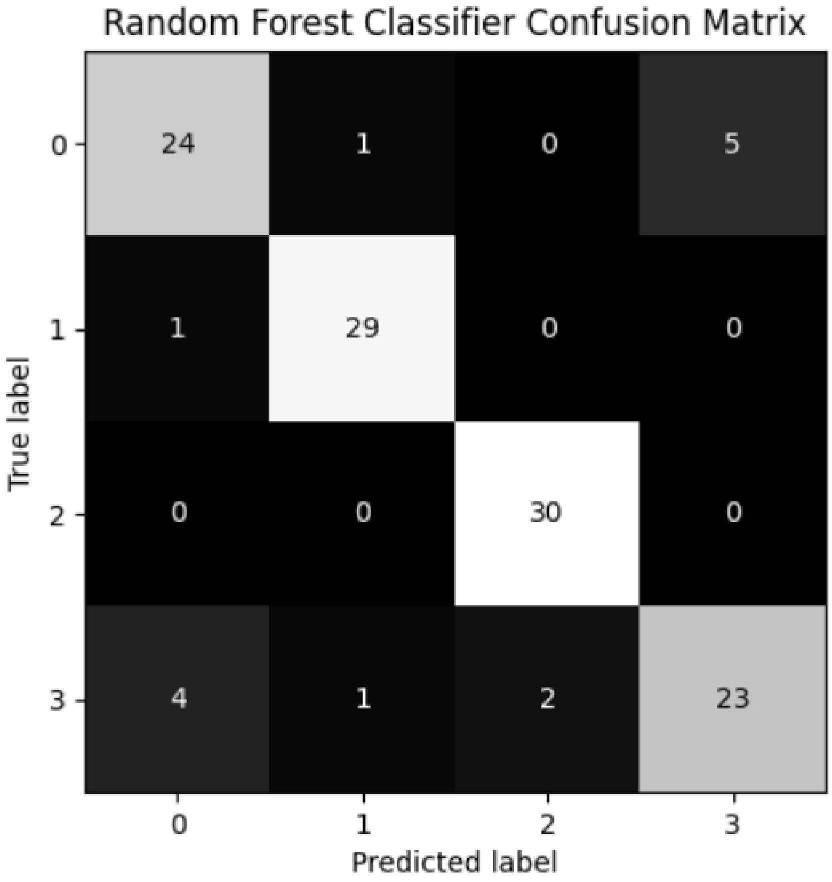


Support Vector Machine Confusion Matrix



Naive Bayes Classifier Confusion Matrix

	0	1	2	3
0	27	1	0	2
1	1	29	0	0
2	0	0	30	0
3	7	1	1	21



Random Forest Classifier Confusion Matrix

	0	1	2	3
0	24	1	0	5
1	1	29	0	0
2	0	0	30	0
3	4	1	2	23

4. Discussion

- *Yes, my model performs pretty well given the small amount of data used to train and test the model. The accuracy of each model is above 80% and at times goes to 90%. The precision is also in the 80 and 90 percent range indicating how good it is. The recall and the f1 score also do good given the small data sample size provided.*
- *My model fixes the problem given in the problem statement well because it can almost always accurately detect what emotion the user speaks with, allowing devices to better respond to its user.*

- *One challenge I met was displaying and being able to listen to the audio using the IPython.display library. For some reason when you run the Audio() function inside another function it does not work. I had to do a bit of research but eventually found out you also need to import the display() function from IPython.display in order to the Audio() function to work.*

- *Another challenge I met was figuring out which library could allow me to scale my data. I quickly figured out that the sklearn library which I have already imported contains a function called MinMaxScaler, which easily allowed me to scale my data*

- *I really liked this project as it taught me as lot of interesting information on how to use and extract information from audio files which can assist me in creating different machine learning models which I can use for any programming projects down the line. I also really enjoyed learning more about different python libraries which would benefit me in creating more python applications much more easier than before.*

5. Appendix

- *https://github.com/hadiplays/Assignment-3*
- *https://www.youtube.com/watch?v=TsQL-sXZOLc*
- *https://pressbooks.pub/sound/chapter/sound-graphs/*
- *https://www.headphonesty.com/2019/07/sample-rate-bit-depth-bit-rate/*
- *https://learn.flucoma.org/reference/mfcc/*
- *https://towardsdatascience.com/getting-to-know-the-mel-spectrogram-31bca3e2d9d0*
- *https://www.researchgate.net/post/Why_we_take_only_12-13_MFCC_coefficients_in_feature_extraction*