README

Overview:

In the present natural NLP scenario, most of the work has been done, in the field of text processing but with the growing internet and abundance of video and audio now available to us, We as a team have tried to work and study the processing of these audio data and analyze the emotions of the speaker. Furthermore, we have even tried to identify the gender(Male or female) of the person. Below is a brief description of our work. The datasets we used, features we tried to learn, and different ML models we tried to implement.

Files:

1. **CNN_Best.ipynb**: The best performing CNN model with 9 layers with clear descriptions and comments for easy readability. The code includes the statistical analysis of the model by evaluating the precision, recall and F1-score.
2. **MLP_and_Random_Forest.ipynb**: Complete MLP and Random Forest on the audio features we extracted from the dataset.
3. **SVM.ipynb**: SVM model implementation and accuracy.
4. **Predictions.csv**: Actual and Predicted values on the test set.
5. **Model.json**: Model Architecture file **(Needed for loading the model)**
6. **mfcc_eng_CNN.h5**: Weights of the training dataset **(Needed for loading the model)**
7. X_test.npy, X_train.npy, Y_test.npy, Y_train.npy: Numpy Arrays **(Needed for loading the model)**

Datasets:

The code takes an input of a dataset of around 2000 voice datasets of different actors both male and female with different emotions that are already labeled, which we have used to train our various models in which CNN with 18 layers performed best.

RAVDESS: https://zenodo.org/record/1188976#.X8TreWgzap4
SAVEE: http://kahlan.eps.surrey.ac.uk/savee/Download.html
CREMA-D: https://www.kaggle.com/ejlok1/cremad

Methodology:

Audio Features Extracted:

1. **MFCC-** Extraction technique basically includes dividing the signal in signals, applying the DFT(Fourier transformation), taking the log of the magnitude, and then scaling it on a Mel scale, followed by applying the inverse DCT (Cosine transformation).

2. **Energy-** Root means square energy, using the STFT(Short Time Fourier Transformation) transformation.
3. **Chromagram-** Contains features; each expressing how the pitch content within the divided time chunks is spread over the twelve chroma bands.
4. **Zero_cross-** Rate of sign change(slope) in a signal in each time frame present.
5. **Mel Spectrogram**- Converts the frequencies to the mel scale.
6. **Spectral Rolloff**- Is the frequency below which a specified percentage of the total spectral energy, e.g. 85%, lies.
7. **Spectral Centroid**- Provides the location of the center of mass for the spectrum. In other words, it provides us with the brightness of the audio.
8. **Envelope Derivative Operator**- Frequency Based Energy Operator
9. **Teager-Kaiser operator**- Detects the level of stress, resulting due to the alteration of the airflow during the production of sound.
10. **Pitch**- Pitch of an audio across the time frame.
11. **Harmonic to Noise Ratio (HNR)-** Measures the ratio between the periodic and non-periodic components of the audio. Jitter, shimmer and harmonics to noise ratio detects the involuntary changes on the physical properties of the vocal tract.
12. **PLP(Perceptual Linear Prediction) and RASTA-** Pattern recognition system for speech recognition from audio signals

Note - All these features have been extracted from the *.wav files present in our dataset, all these features define the audio in different sorts of ways. Finding the best combination of features which would really help in identifying the emotion as well as the gender turned out to be a very tedious task.

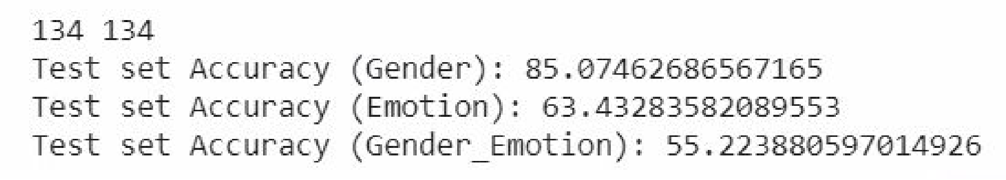
According to our research and implementations the best feature combination was found to be **MFCC, Energy** (dimension: 432 columns). Therefore all the current outputs of our different models have been trained on this set of features.

***Different combinations of features and outputs will be included in our final report.**

Models and Accuracies:

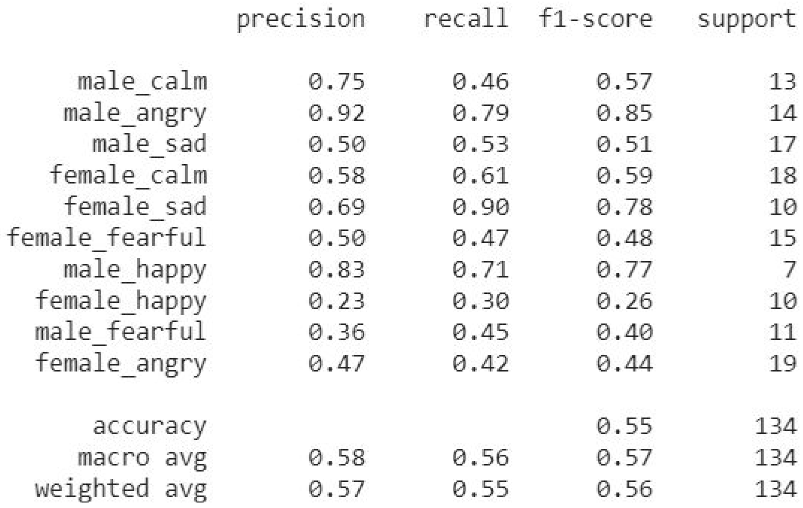
CNN (Present in: CNN_best.ipynb)

CNN had the best performance in our project with the validation set accuracies as,



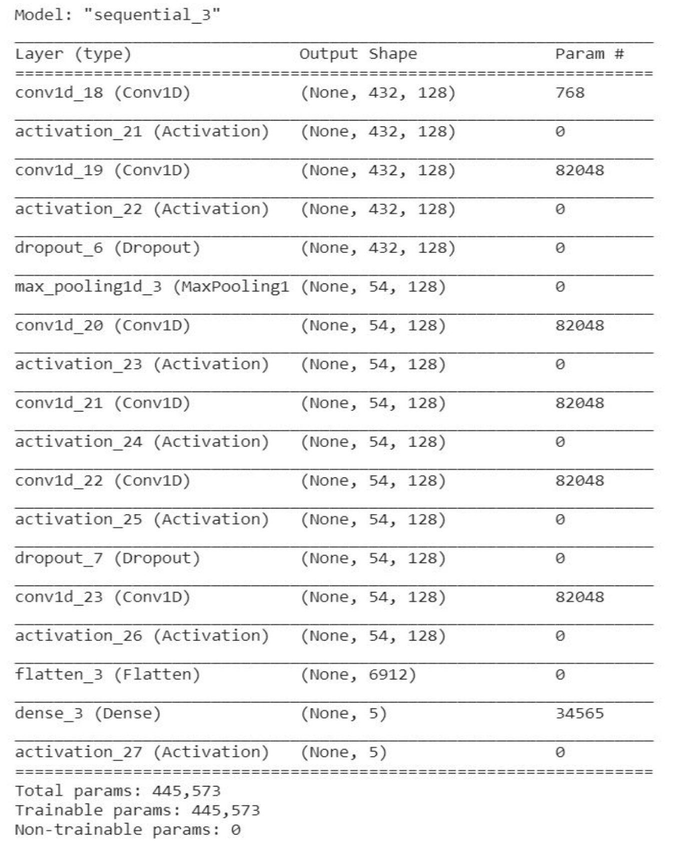
```
134 134
Test set Accuracy (Gender): 85.07462686567165
Test set Accuracy (Emotion): 63.43283582089553
Test set Accuracy (Gender_Emotion): 55.223880597014926
```

CNN was easily able to predict the gender of the audio with **63.43%** accuracy Emotion with the accuracy of **85.07%** Gender and emotion together with the accuracy of **55.22%**.

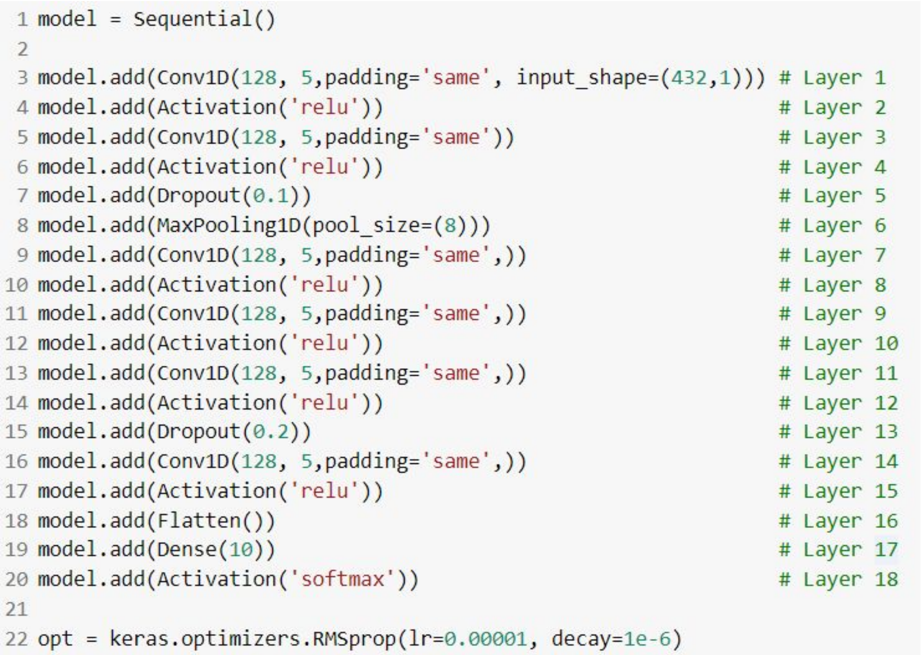


```
                precision    recall  f1-score   support

     male_calm       0.75      0.46      0.57        13
    male_angry       0.92      0.79      0.85        14
      male_sad       0.50      0.53      0.51        17
   female_calm       0.58      0.61      0.59        18
    female_sad       0.69      0.90      0.78        10
female_fearful       0.50      0.47      0.48        15
    male_happy       0.83      0.71      0.77         7
  female_happy       0.23      0.30      0.26        10
  male_fearful       0.36      0.45      0.40        11
  female_angry       0.47      0.42      0.44        19

      accuracy                           0.55       134
     macro avg       0.58      0.56      0.57       134
  weighted avg       0.57      0.55      0.56       134
```

CNN model:-



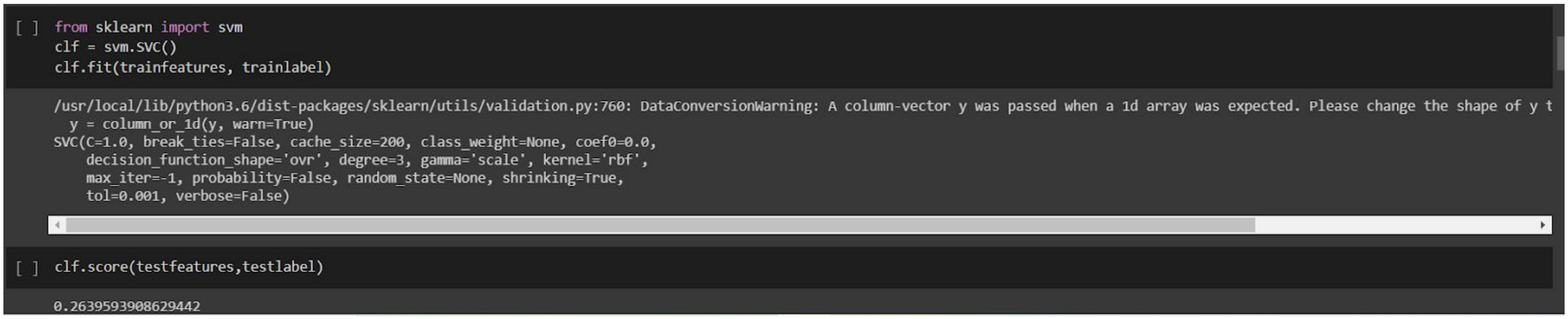
```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_18 (Conv1D)           (None, 432, 128)          768
_________________________________________________________________
activation_21 (Activation)   (None, 432, 128)          0
_________________________________________________________________
conv1d_19 (Conv1D)           (None, 432, 128)          82048
_________________________________________________________________
activation_22 (Activation)   (None, 432, 128)          0
_________________________________________________________________
dropout_6 (Dropout)          (None, 432, 128)          0
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 54, 128)           0
_________________________________________________________________
conv1d_20 (Conv1D)           (None, 54, 128)           82048
_________________________________________________________________
activation_23 (Activation)   (None, 54, 128)           0
_________________________________________________________________
conv1d_21 (Conv1D)           (None, 54, 128)           82048
_________________________________________________________________
activation_24 (Activation)   (None, 54, 128)           0
_________________________________________________________________
conv1d_22 (Conv1D)           (None, 54, 128)           82048
_________________________________________________________________
activation_25 (Activation)   (None, 54, 128)           0
_________________________________________________________________
dropout_7 (Dropout)          (None, 54, 128)           0
_________________________________________________________________
conv1d_23 (Conv1D)           (None, 54, 128)           82048
_________________________________________________________________
activation_26 (Activation)   (None, 54, 128)           0
_________________________________________________________________
flatten_3 (Flatten)          (None, 6912)              0
_________________________________________________________________
dense_3 (Dense)              (None, 5)                 34565
_________________________________________________________________
activation_27 (Activation)   (None, 5)                 0
=================================================================
Total params: 445,573
Trainable params: 445,573
Non-trainable params: 0
```



```
model = Sequential()

model.add(Conv1D(128, 5,padding='same', input_shape=(432,1))) # Layer 1
model.add(Activation('relu'))                                 # Layer 2
model.add(Conv1D(128, 5,padding='same'))                      # Layer 3
model.add(Activation('relu'))                                 # Layer 4
model.add(Dropout(0.1))                                       # Layer 5
model.add(MaxPooling1D(pool_size=(8)))                        # Layer 6
model.add(Conv1D(128, 5,padding='same',))                     # Layer 7
model.add(Activation('relu'))                                 # Layer 8
model.add(Conv1D(128, 5,padding='same',))                     # Layer 9
model.add(Activation('relu'))                                 # Layer 10
model.add(Conv1D(128, 5,padding='same',))                     # Layer 11
model.add(Activation('relu'))                                 # Layer 12
model.add(Dropout(0.2))                                       # Layer 13
model.add(Conv1D(128, 5,padding='same',))                     # Layer 14
model.add(Activation('relu'))                                 # Layer 15
model.add(Flatten())                                          # Layer 16
model.add(Dense(10))                                          # Layer 17
model.add(Activation('softmax'))                              # Layer 18

opt = keras.optimizers.RMSprop(lr=0.00001, decay=1e-6)
```

SVM (Present in - SVM_RandomForest.ipynb)**:**

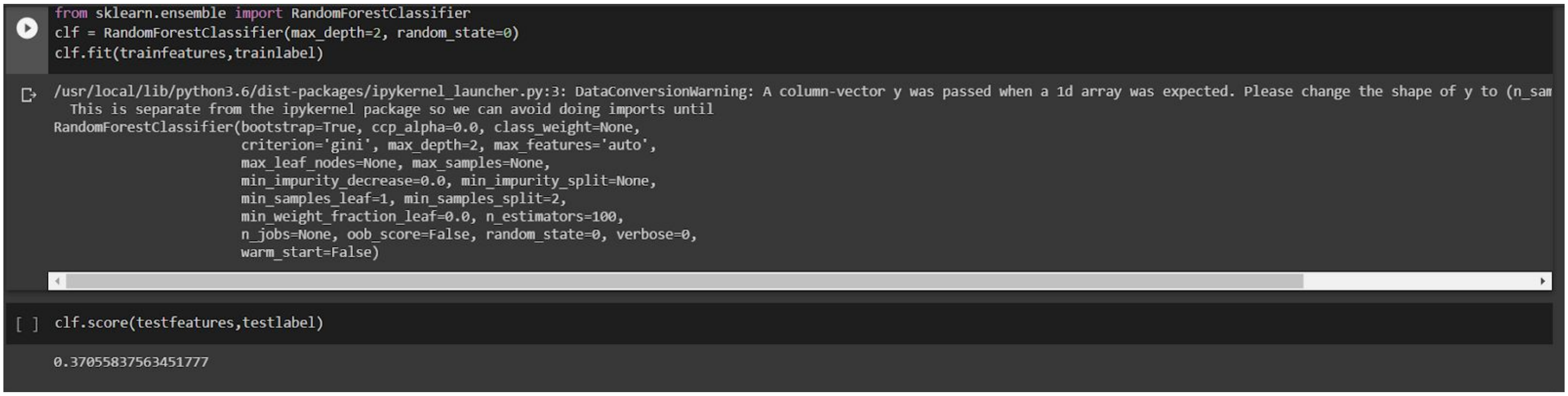
Accuracy of detection gender an emotion together on validation set: **26%**



```
[ ] from sklearn import svm
    clf = svm.SVC()
    clf.fit(trainfeatures, trainlabel)

    /usr/local/lib/python3.6/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y t
      y = column_or_1d(y, warn=True)
    SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
        decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
        max_iter=-1, probability=False, random_state=None, shrinking=True,
        tol=0.001, verbose=False)

[ ] clf.score(testfeatures,testlabel)

    0.2639593908629442
```

RandomForestClassifier (Present in - SVM_RandomForest.ipynb)**:**

Accuracy of detection gender an emotion together on validation set: **37%**

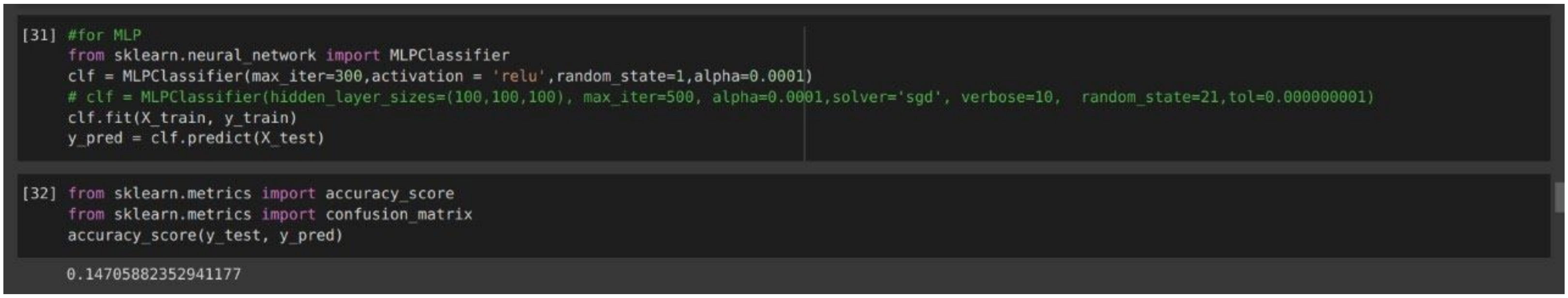


```
    from sklearn.ensemble import RandomForestClassifier
    clf = RandomForestClassifier(max_depth=2, random_state=0)
    clf.fit(trainfeatures,trainlabel)

    /usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sam
      This is separate from the ipykernel package so we can avoid doing imports until
    RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                           criterion='gini', max_depth=2, max_features='auto',
                           max_leaf_nodes=None, max_samples=None,
                           min_impurity_decrease=0.0, min_impurity_split=None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=100,
                           n_jobs=None, oob_score=False, random_state=0, verbose=0,
                           warm_start=False)

[ ] clf.score(testfeatures,testlabel)

    0.37055837563451777
```

MLP (Present in - MLP.ipynb)**:**

Accuracy of detection gender an emotion together on validation set: **15%**

Activation used is relu with alpha=0.0001



```
[31] #for MLP
     from sklearn.neural_network import MLPClassifier
     clf = MLPClassifier(max_iter=300,activation = 'relu',random_state=1,alpha=0.0001)
     # clf = MLPClassifier(hidden_layer_sizes=(100,100,100), max_iter=500, alpha=0.0001,solver='sgd', verbose=10,  random_state=21,tol=0.000000001)
     clf.fit(X_train, y_train)
     y_pred = clf.predict(X_test)

[32] from sklearn.metrics import accuracy_score
     from sklearn.metrics import confusion_matrix
     accuracy_score(y_test, y_pred)

     0.14705882352941177
```