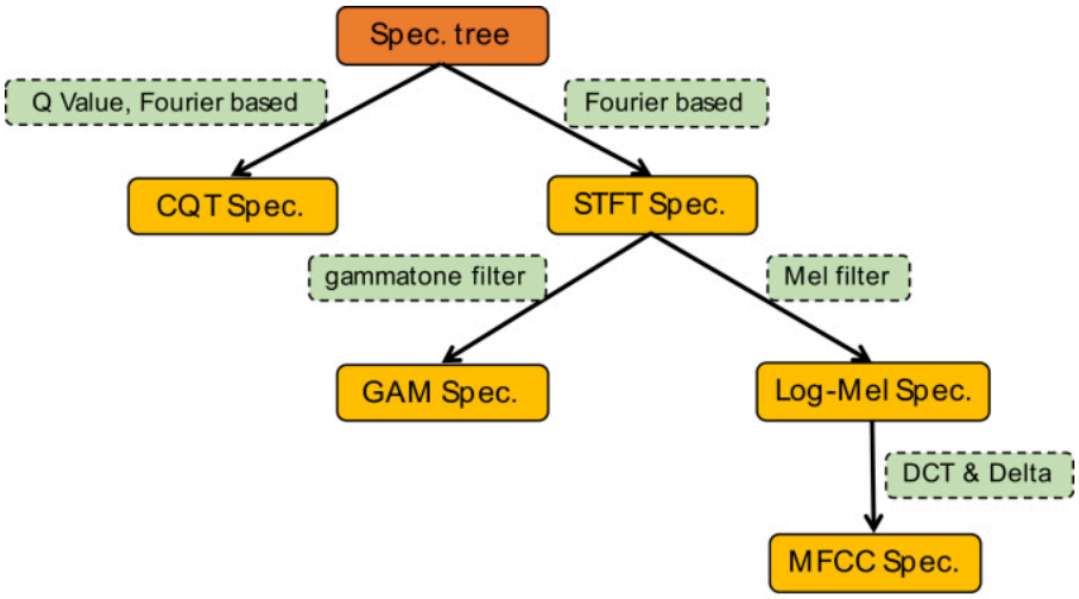
Audio Emotion Detection

Link: https://github.com/Oversoul73/DS-Capstone-Spring-2024/tree/Nagini

MFCCs (Mel-Frequency Cepstral Coefficients) and Mel Spectrograms are two commonly used acoustic feature extraction methods in audio processing and analysis.

MFCCs (Mel-Frequency Cepstral Coefficients):
MFCCs are a compact representation of the spectral features of an audio signal. They are derived from the Mel Spectrogram through the following steps:
Take the Discrete Cosine Transform (DCT) of the log-magnitude Mel Spectrogram.
The resulting coefficients are the MFCCs, representing the audio's short-term power spectrum in a decorrelated form.
MFCCs are commonly used as features in speech recognition systems, as they can effectively capture the spectral envelope of the speech signal, which is important for distinguishing different speech sounds. They are also used in music information retrieval applications, such as genre classification and audio similarity measures.
In summary, Mel Spectrograms provide a time-frequency representation of the audio signal that is perceptually relevant, while MFCCs provide a compact, decorrelated representation of the spectral features. Both are widely used in audio processing and analysis tasks.

Mel Spectrograms:
A Mel Spectrogram is a time-frequency representation of an audio signal that uses the Mel scale, which is a perceptual scale of pitches judged by listeners to be equal in distance from one another. The Mel Spectrogram is obtained by the following steps:
Take the Short-Time Fourier Transform (STFT) of the audio signal to get the spectrogram.
Map the linear frequency scale of the spectrogram to the Mel scale, which is a nonlinear scale that better approximates the human auditory system's response.
Take the logarithm of the Mel-scaled spectrogram to emphasize the lower frequency components.
The Mel Spectrogram provides a time-frequency representation of the audio signal that is more perceptually relevant than a standard linear-scale spectrogram. It is commonly used as input to machine learning models for tasks like speech recognition, music genre classification, and audio event detection.



Dataset:

RAVDESS contains 1440 files: 60 trials per actor x 24 actors = 1440. The RAVDESS contains 24 professional actors (12 female, 12 male), vocalizing two lexically-matched statements in a neutral North American accent. Speech emotions includes calm, happy, sad, angry, fearful, surprise, and disgust expressions. Each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression.

Models:

Logistic Regression Model:

Logistic regression is a statistical modeling technique used to predict the probability of a binary or categorical outcome variable based on one or more predictor variables. Unlike linear regression, which is used to predict continuous outcomes, logistic regression is specifically designed for binary classification problems.

The key aspects of logistic regression are:
Outcome Variable: The dependent variable in logistic regression is binary or categorical, taking on values such as 0/1, true/false, or pass/fail.
Probability Prediction: Logistic regression models the probability of the outcome variable occurring as a function of the predictor variables. The output is a probability between 0 and 1.

Logit Transformation: Logistic regression applies a logit transformation to the probability, which allows the model to overcome the limitation of a binary outcome being bounded between 0 and 1.
Odds Ratio: The model coefficients are typically reported as odds ratios, which represent the change in the odds of the outcome occurring given a one-unit change in the predictor variable.
Model Evaluation: Logistic regression models are evaluated using metrics like accuracy, precision, recall, F1-score, and area under the ROC curve (AUC-ROC).

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a specialized type of artificial neural network designed for processing grid-like data, such as images. They are widely used in computer vision and image recognition tasks.
Key aspects of CNNs:
Convolutional Layers: CNNs consist of convolutional layers that apply a set of learnable filters (or kernels) to the input image. These filters detect low-level features like edges and shapes, and higher-level features like object parts.
Pooling Layers: Pooling layers reduce the spatial size of the feature maps, making the representations more manageable and robust to small translations in the input.
Fully Connected Layers: After the convolutional and pooling layers, the network typically has one or more fully connected layers that perform high-level reasoning based on the extracted features.
Parameter Sharing: CNNs leverage the concept of parameter sharing, where the same set of weights (filters) are applied across the entire input image. This reduces the number of trainable parameters compared to a fully connected network.
Translation Invariance: CNNs are designed to be invariant to the location of features in the input image, making them effective for tasks like object recognition and classification.
Applications: CNNs have been successfully applied to a wide range of computer vision tasks, including image classification, object detection, semantic segmentation, and image generation.

Autoencoders and Variational Autoencoders

Autoencoders and Variational Autoencoders (VAEs) are neural network architectures used for data compression and representation learning.
Autoencoders

Autoencoders consist of an encoder that compresses the input data into a lower-dimensional latent space, and a decoder that reconstructs the original input from the latent representation.
The goal is to learn an efficient encoding of the data that allows for accurate reconstruction, effectively compressing the input.
Autoencoders can be used for dimensionality reduction, feature extraction, and data denoising.
Variational Autoencoders
VAEs extend the basic autoencoder by modeling the latent space as a probability distribution rather than a single point.
The encoder outputs the parameters (mean and variance) of a probability distribution in the latent space, rather than a single latent vector.
This probabilistic latent representation allows VAEs to generate new samples by sampling from the learned latent distribution, enabling them to be used as generative models.
VAEs are trained by optimizing an objective that balances reconstruction accuracy and the regularization of the latent space to match a prior distribution (typically a standard normal distribution).

In summary, autoencoders learn efficient data representations through compression, while variational autoencoders add a probabilistic component to the latent space, enabling them to generate new samples in addition to encoding and decoding data.

Here we used:

- Logistic Regression on MFCC and Mel Spec Features
- CNNs on MFCC and Mel Spec Features
- Autoencoders and Variational Autoencoders on whichever features are giving better performance in the above models

The results we got for each code blocks are presented here:

Logistic Regression: MFCCs

```
scores, cmatrix = LogisticRegressionPipeline(X1,y1)
```

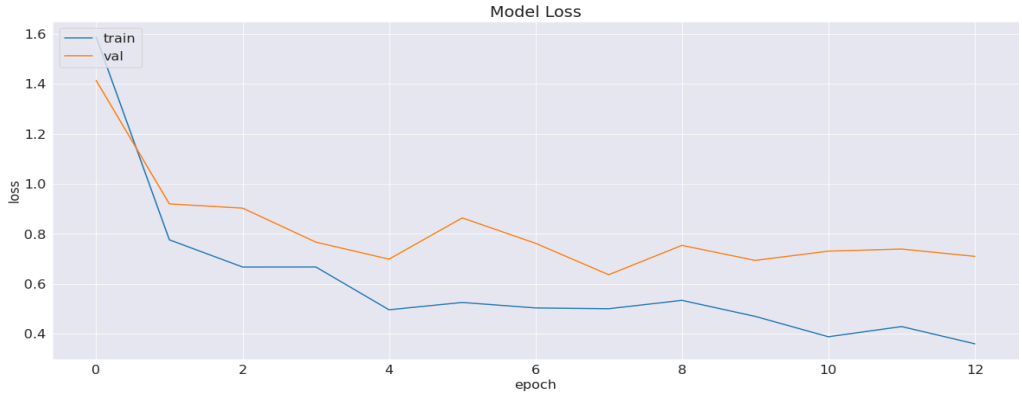
```
Training Performance
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       154
           1       0.99      0.97      0.98       153

    accuracy                           0.98       307
   macro avg       0.98      0.98      0.98       307
weighted avg       0.98      0.98      0.98       307

----------------------------------------
Test Performance
              precision    recall  f1-score   support

           0       0.82      0.87      0.85        38
           1       0.86      0.82      0.84        39

    accuracy                           0.84        77
   macro avg       0.84      0.84      0.84        77
weighted avg       0.85      0.84      0.84        77

----------------------------------------
5-Folds Scores:  [0.64935065 0.74025974 0.67532468 0.68831169 0.80263158]
----------------------------------------
5-Folds Average Score:  0.7111756664388242
```

Logistic Regression: Mel Spectrogram

```
scores, cmatrix = LogisticRegressionPipeline(X2,y2)
```

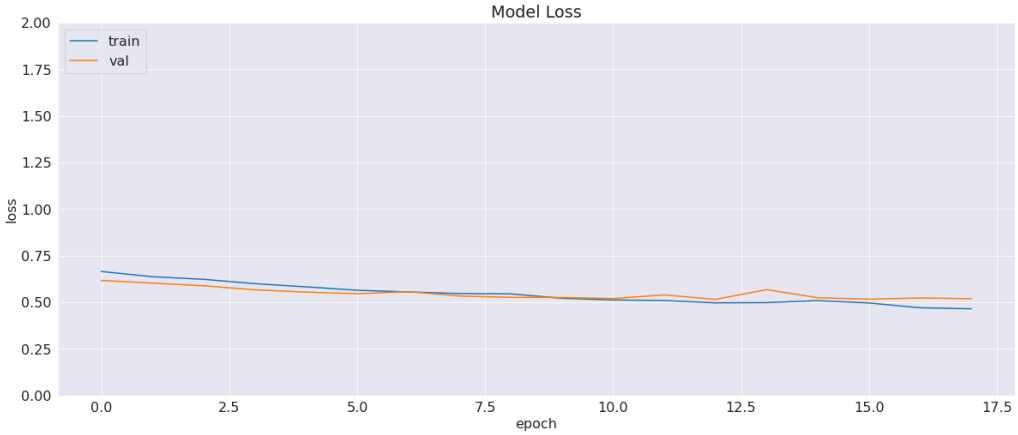
```
Training Performance
              precision    recall  f1-score   support

           0       0.90      0.68      0.77       154
           1       0.74      0.93      0.82       153

    accuracy                           0.80       307
   macro avg       0.82      0.80      0.80       307
weighted avg       0.82      0.80      0.80       307

----------------------------------------
Test Performance
              precision    recall  f1-score   support

           0       0.74      0.61      0.67        38
           1       0.67      0.79      0.73        39

    accuracy                           0.70        77
   macro avg       0.71      0.70      0.70        77
weighted avg       0.71      0.70      0.70        77

----------------------------------------
5-Folds Scores:  [0.67532468 0.74025974 0.67532468 0.72727273 0.71052632]
----------------------------------------
5-Folds Average Score:  0.7057416267942583
```

CNN: MFCCs



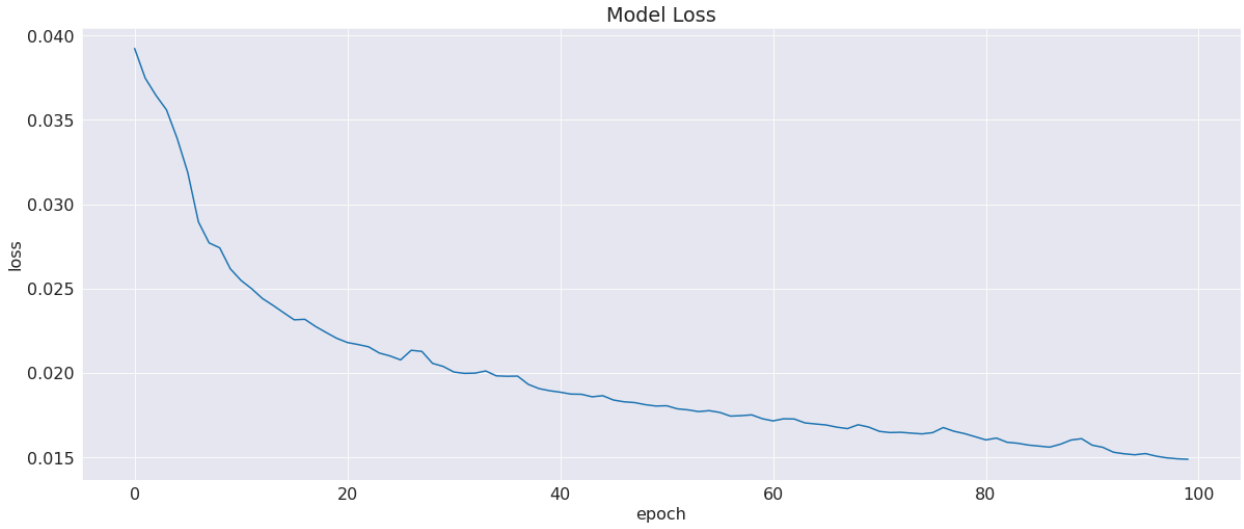
	precision	recall	f1-score	support
0	0.70	0.55	0.62	38
1	0.64	0.77	0.70	39
accuracy			0.66	77
macro avg	0.67	0.66	0.66	77
weighted avg	0.67	0.66	0.66	77

CNN: Mel Spectrogram



	precision	recall	f1-score	support
0	0.53	0.42	0.47	38
1	0.53	0.64	0.58	39
accuracy			0.53	77
macro avg	0.53	0.53	0.53	77
weighted avg	0.53	0.53	0.53	77

Autoencoder:



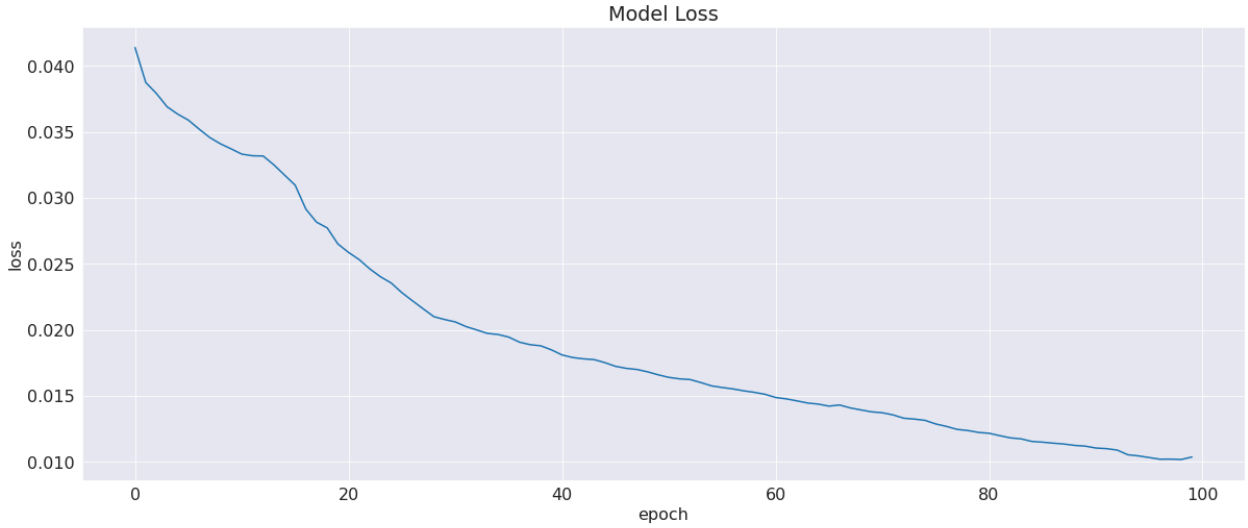

```
model.evaluate(X_happy,X_happy)
```

```
6/6 [==============================] - 0s 4ms/step - loss: 0.0148
0.014842587523162365
```

```
model.evaluate(X_sad,X_sad)
```

```
6/6 [==============================] - 0s 4ms/step - loss: 0.0193
0.019280999898910522
```

Variational Autoencoders:

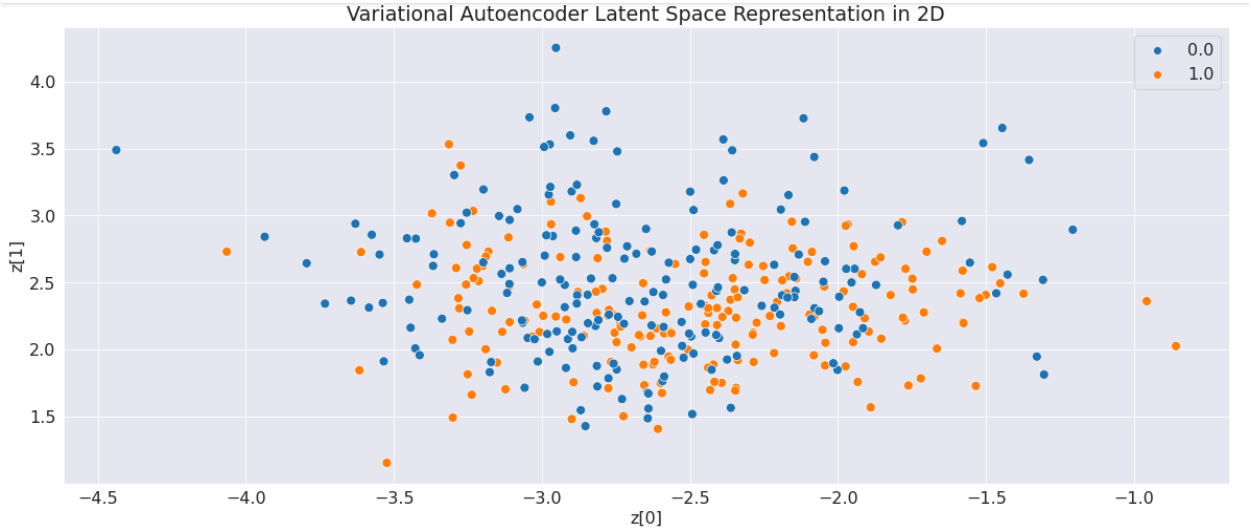



```
model.evaluate(X_happy,X_happy)
```

```
6/6 [==============================] - 0s 7ms/step - loss: 0.0098
0.009753730148077011
```

```
model.evaluate(X_sad,X_sad)
```

```
6/6 [==============================] - 0s 6ms/step - loss: 0.0235
0.023536866530776024
```



Conclusion:

The Logistic Regression Model showed promising results when utilizing MFCC Features, achieving respectable classification accuracy and F1-score. Similarly, CNNs demonstrated effectiveness in leveraging MFCC Features. Hence, for autoencoders, only MFCCs are utilized as features.

When it comes to autoencoders, Variational Autoencoders excel in reconstructing audio samples with lower reconstruction loss compared to traditional autoencoders. This superiority stems from their ability to capture the distribution of happy audio samples within a 128-dimensional latent space.

We have taken some standard machine learning models like logistic regression, CNN, autoencoders, and variational encoders to evaluate the performance of different audios of actors of different ages.