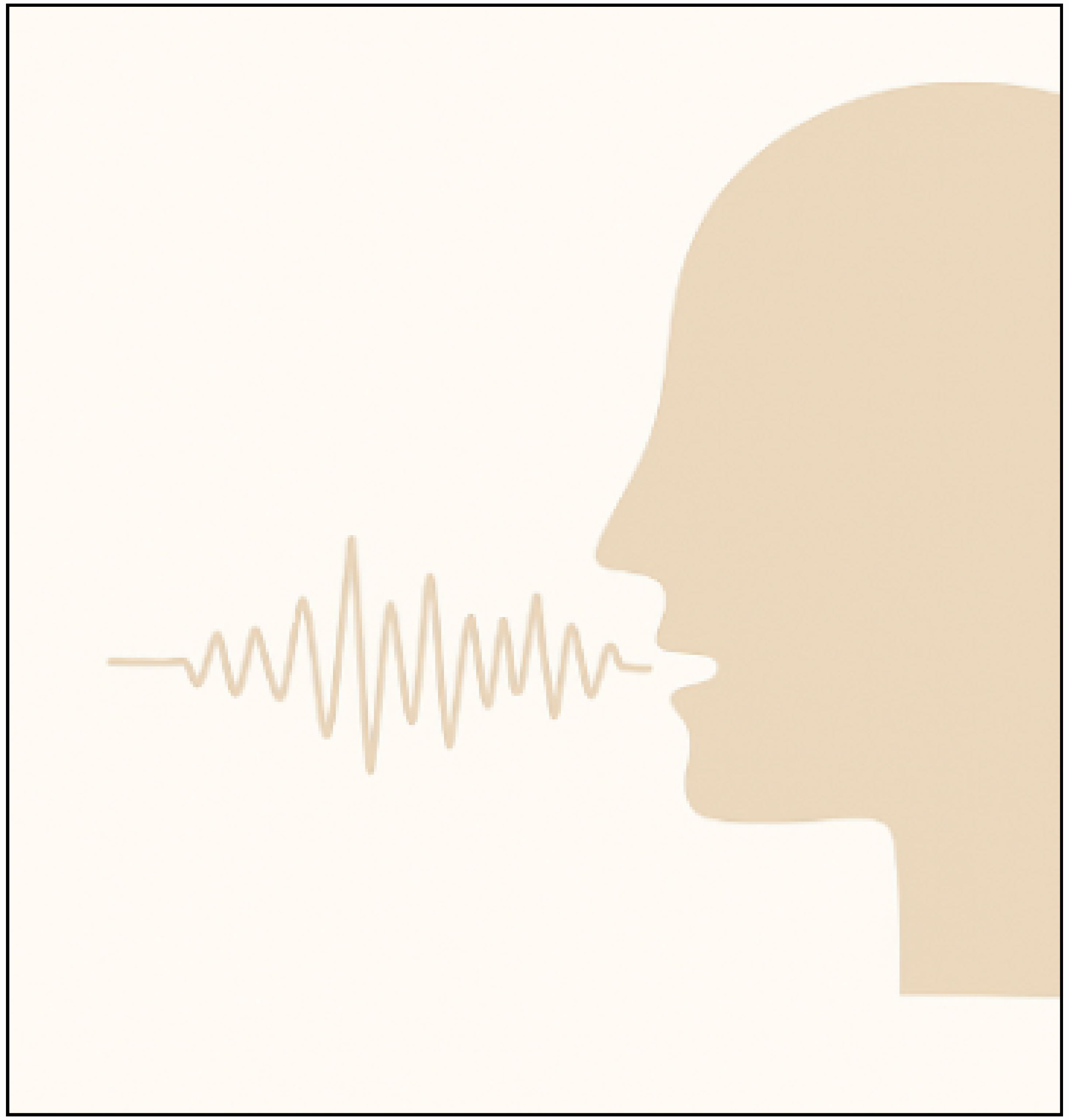
Dysarthria detection

MFCC + Excitation features

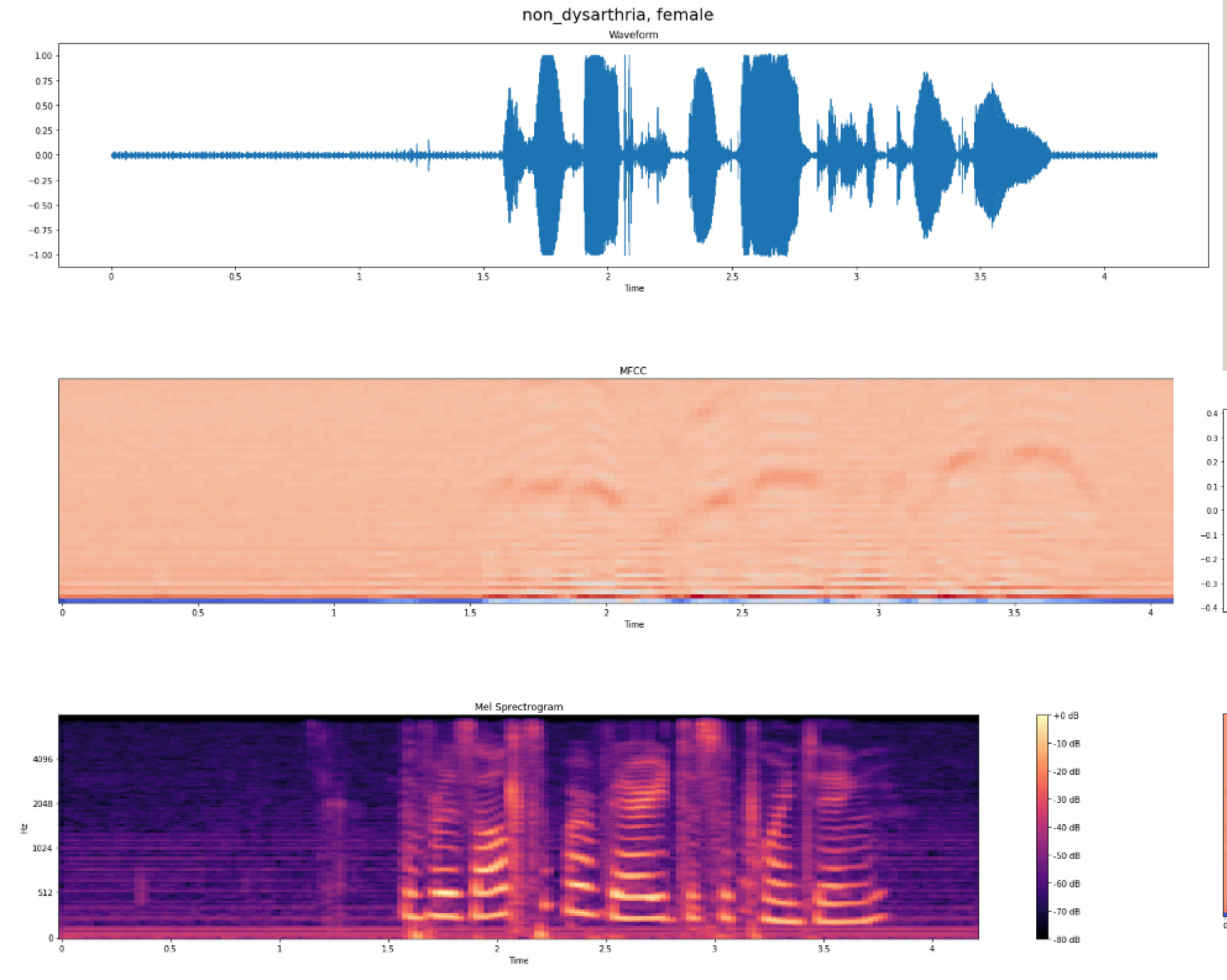


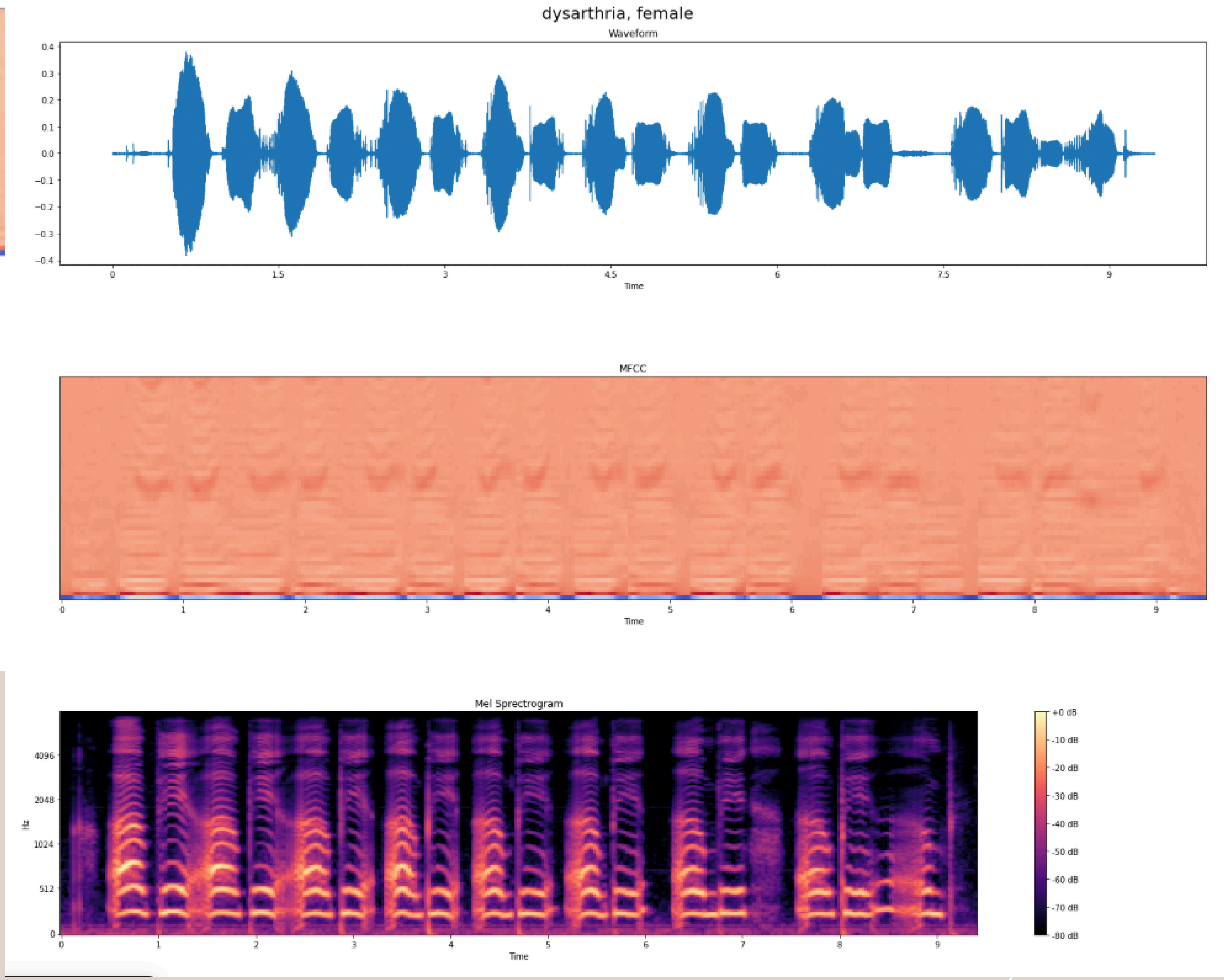
Team 4
Jyothi, Druvitha

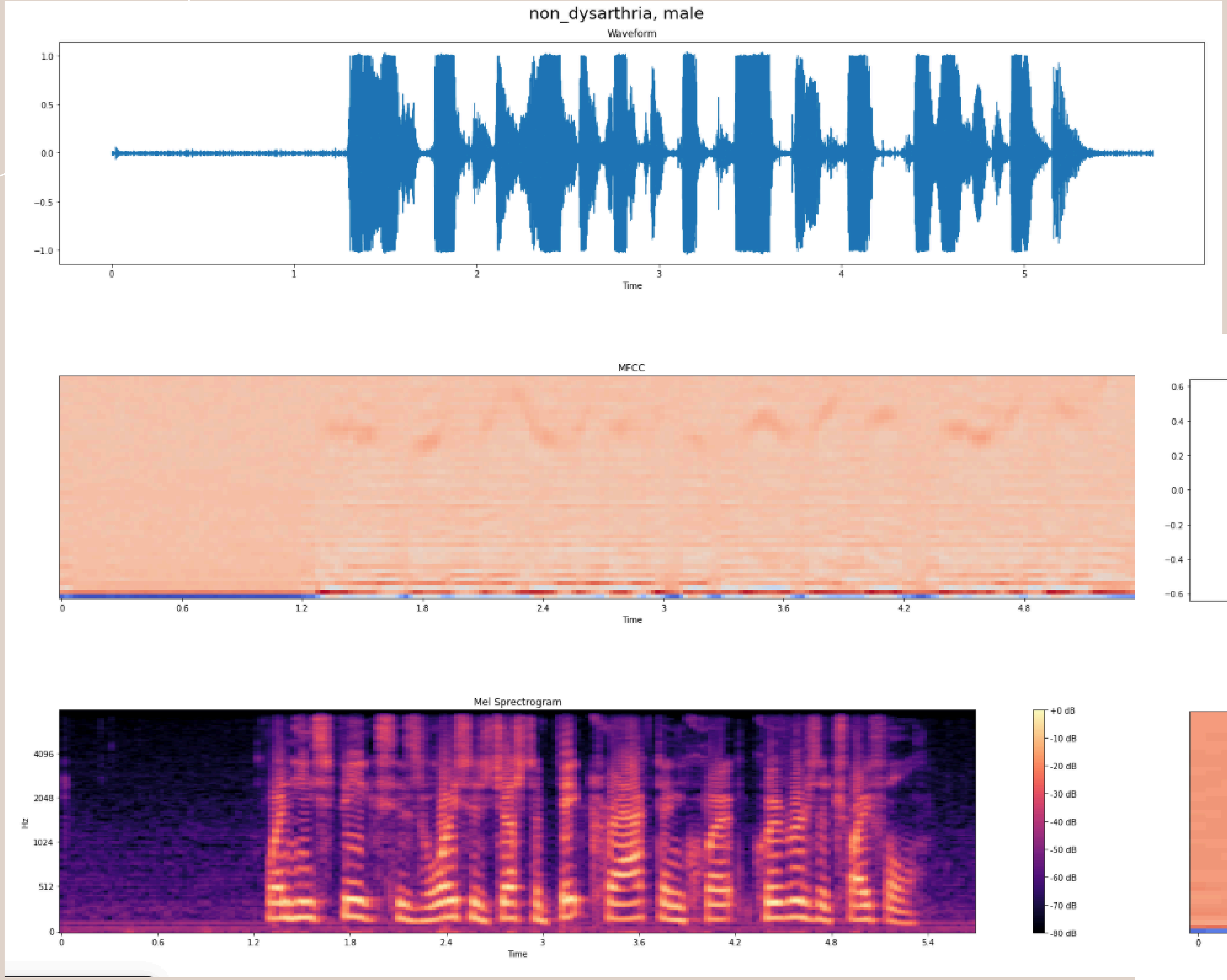
Dysarthria Dataset

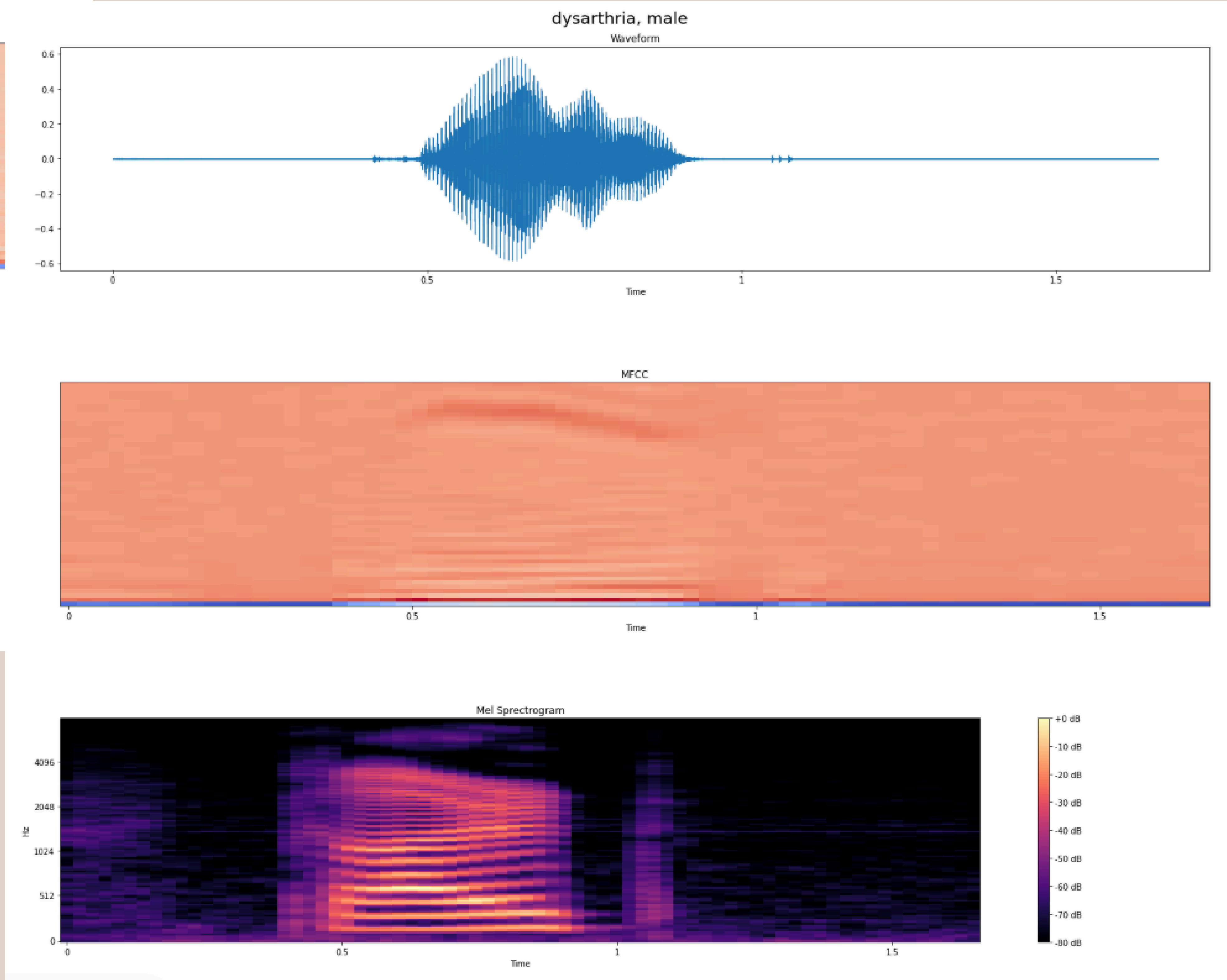
The TORGO Database: Acoustic and articulatory speech from speakers with dysarthria

- dysarthria_female: 500 samples of dysarthric female audio recorded on different sessions.
- dysarthria_male: 500 samples of dysarthric male audio recorded on different sessions.
- non _dysarthria _female: 500 samples of non-dysarthric female audio recorded on different sessions.
- non _dysarthria _male: 500 samples of non-dysarthric male audio recorded on different sessions.


non_dysarthria, female
Waveform
Time
MFCC
Time
Mel Sprectrogram
Hz
Time
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB


dysarthria, female
Waveform
Time
MFCC
Time
Mel Sprectrogram
Hz
Time
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB


non_dysarthria, male
Waveform
Time
MFCC
Time
Mel Sprectrogram
Hz
Time
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB


dysarthria, male
Waveform
Time
MFCC
Time
Mel Sprectrogram
Hz
Time
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB
-60 dB
-70 dB
-80 dB

Setup

Model: Support Vector Machine

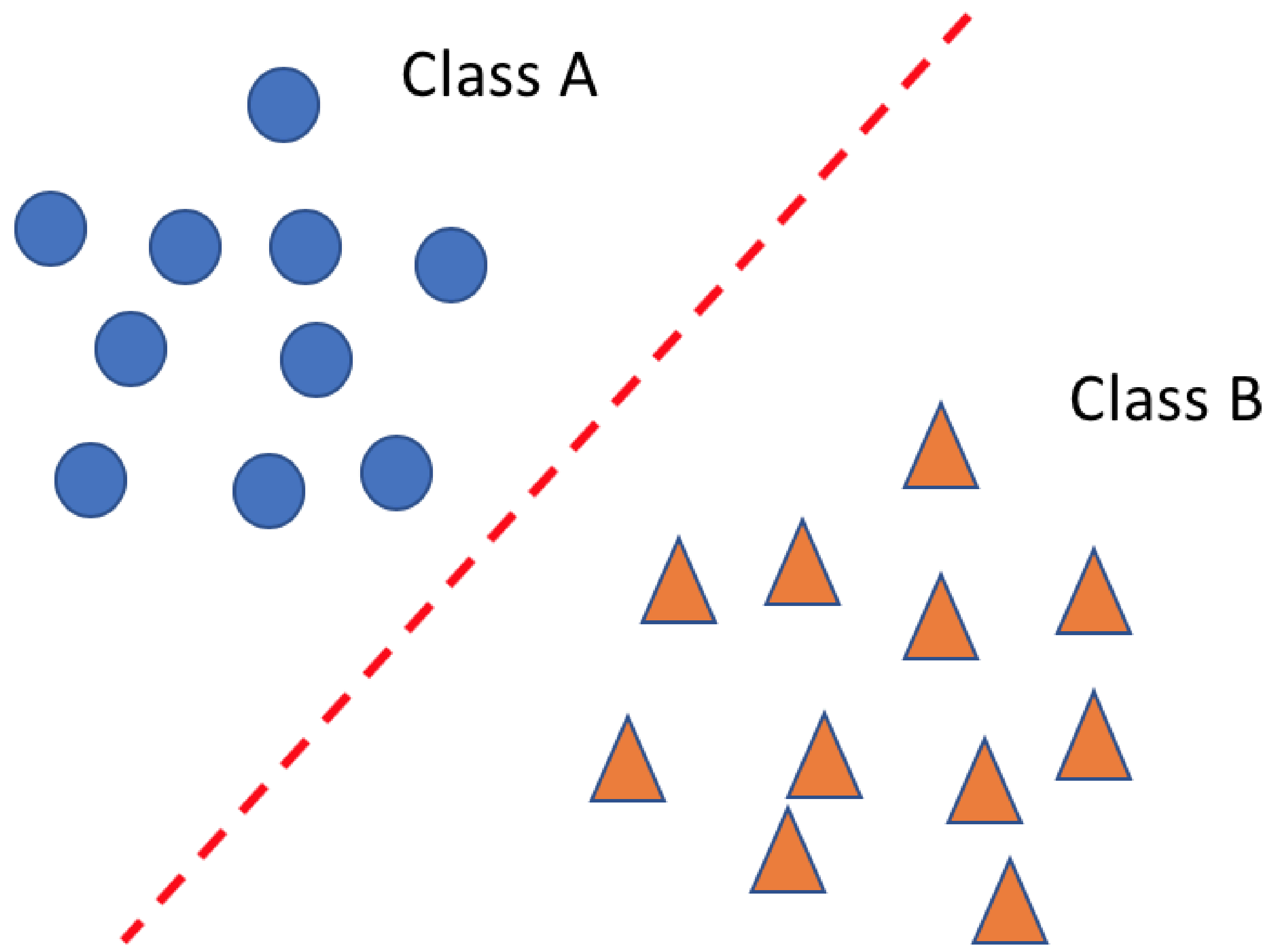
We have total of 2000 data points, which is quite low, thus SVM. Also are suitable for binary classification (can learn non-linearity with kernels like rbf) and prevent overfitting for equally distributed data.

Hyperparameter tuning: GridSearchCV

Using gridsearchCV, we find the best performing (on basis of accuracy of 5-fold validation) on a predefined grid -param_grid.

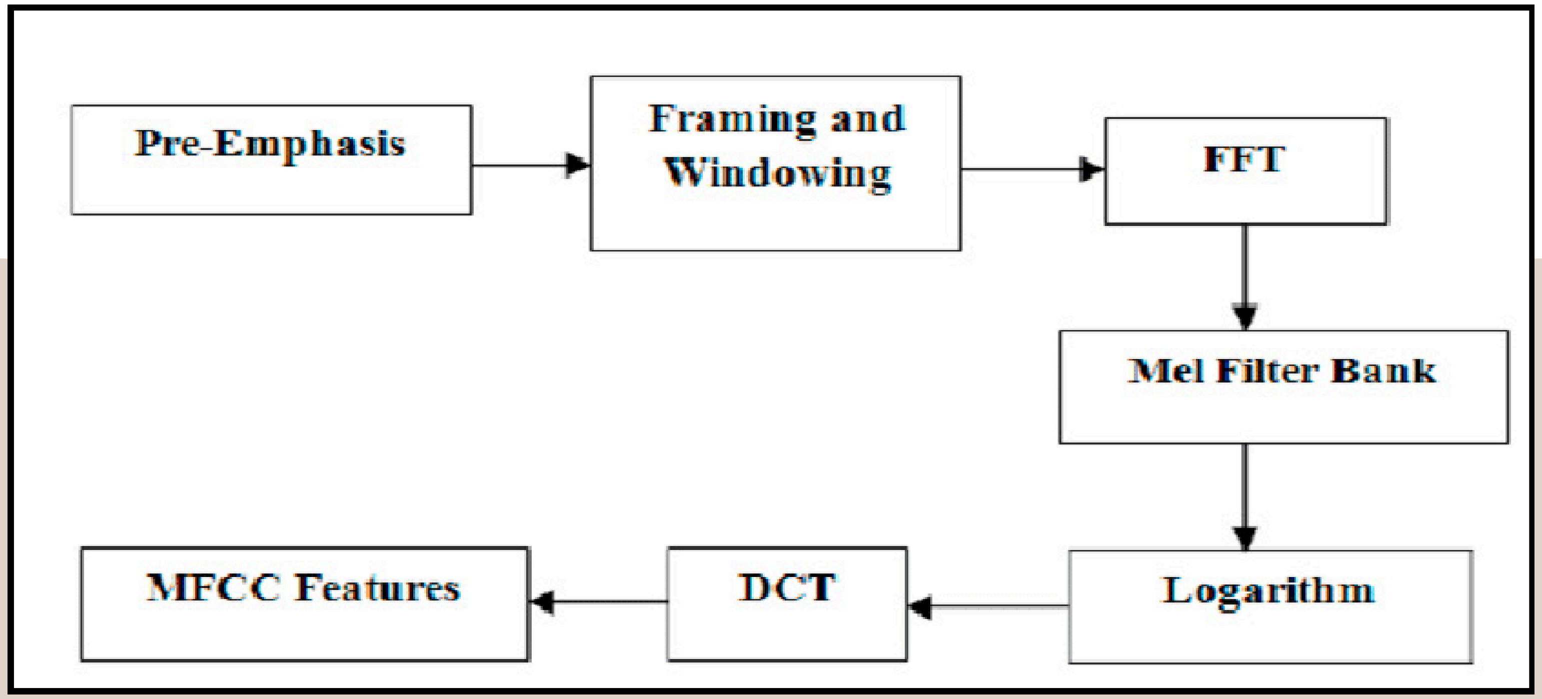
param_grid=[{'C':[0.5,1,10,100, 1000], 'gamma':[10,1,0.1,0.001,0.00001, 0.000001], 'kernel':['rbf'], }]

Here, C is the regularization parameter for allowing misclassifications within a margin. kernel-type: 'rbf' (Radial Basis Function) for non-linear data. gamma: defines the influence of a single training example. small gamma -> a far influence.



MFCC

```
mfccs = librosa.feature.mfcc(y=signal, sr=fs, n_mfcc=n_mfcc)
```



- Extracted **52** MFCC features from speech signals.
- Training Data: X_train consists of 52 MFCC features (mean of all frames) + Gender, while y_train represents Dysarthria classification.
- By hyperparameter Tuning, Optimized SVM parameters:
    - C = 10, gamma = 0.001, kernel = rbf
- **Achieved 98.25% accuracy on the test set.**

Extracting Epoch locations

Reference: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4648930

```
zff_output = zero_frequency_filter(speech, fs)
epochs = detect_epochs(zff_output)
```

Step 1: Difference the speech signal (to remove any time-varying low frequency bias in the signal) $x[n] = s[n] - s[n-1]$

Step 2: Pass the differenced speech signal twice through an ideal resonator at zero frequency. $y_1[n] = -\sum_{k=1}^{2} a_k y_1[n-k] + x[n]$ $y_2[n] = -\sum_{k=1}^{2} a_k y_2[n-k] + y_1[n]$

Step 3: Remove the trend in by subtracting the average over 10 ms at each sample.

$$y[n] = y_2[n] - \frac{1}{2N+1} \sum_{m=-N}^{N} y_2[n+m]$$

Step 4: Negative to positive zero-crossings are the epoch locations!

After extracting epoch locations ⟶

Pitch perturbation features:

- T0 is the pitch period
- Computed Mean F0, Std F0, Jitter, RAP, PPQ, and PPF from detected epochs.
- **Jitter**: Measures frequency variation between cycles.
- **RAP & PPQ**: Local pitch period variations (3-frame & 5-frame).
- **PPF**: Ratio of pitch periods exceeding 0.005s threshold.

Reference: [Dysarthric speech detection from telephone quality speech using epoch-based pitch perturbation features]()

1. Mean F_0 (μ) is computed by:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} F_{0_i} \tag{4}$$

2. Standard deviation of F_0 contour (σ) is computed by:

$$\sigma = \frac{1}{n-1}\sum_{i=1}^{n} (F_{0_i} - \mu)^2 \tag{5}$$

3. Jitter is computed by:

$$jitter = \frac{\frac{1}{n-1}\sum_{i=1}^{n-1} |F_{0_i} - F_{0_{i+1}}|}{\mu} \tag{6}$$

RAP and PPQ are computed similarly to Jitter but with 3 and 5 pitch cycles respectively.

4. PPF is computed by:

$$PPF = \frac{N_{p \geq threshold}}{N} \times 100 \tag{7}$$

The numerator and denominator represents the pitch values greater than the given threshold and the total number of extracted pitch values, respectively.

Results: Pitch perturbation features

- Extracted those 6 PP features from speech signals.
- Trained only on these **6 PP features**.

Achieved 76.75% accuracy on the test set.

- Combining these **6 PP features with 52 MFCC** features.

Achieved 97.5% accuracy on the test set.

After extracting epoch locations ⟶

Other Epoch features:

- Extracts key statistical features from the time intervals (T0) between detected epochs.
- Features:
  - **Minimum, Maximum, Mean, and Median intervals**
  - **Standard Deviation (Variability)**
  - **Interquartile Range (IQR)**
- These features help analyze the periodicity and timing characteristics of the speech signal.

Results: Other Epoch features

- Extracted those 6 epoch based features from speech signals.
- Trained only on these **6 features**.

Achieved 81% accuracy on the test set.

- Combining these **6 features with 52 MFCC features**.

Achieved 97.75% accuracy on the test set.

Observation:
Dysarthric speech often shows uneven pitch periods and unstable timing, which are better captured by these features than just frequency-based measures. While PP features detect small-scale irregularities, they might not fully capture the larger-scale pitch period disruptions seen in dysarthria, making epoch-based features more effective.

Finding Residual Cepstral Coefficients

```
a = librosa.lpc(signal_data, order=order)
residual = signal.lfilter([1] + -1 * a[1:].tolist(), [1], signal_data)
```

$$e(n) = x(n) - \sum_{i=1}^{p} a_i \cdot x(n-i)$$

Step 1: Finding LP residual (after preemphasis)

$$x(n) \approx -\sum_{i=1}^{p} a_i \cdot x(n-i)$$ (Linear Predictive Coding)

The function librosa.lpc(signal_data, order) returns an array of LPC coefficients:

$A = [a_0, a_1, a_2, ..., a_p]$ a_0 is always 1

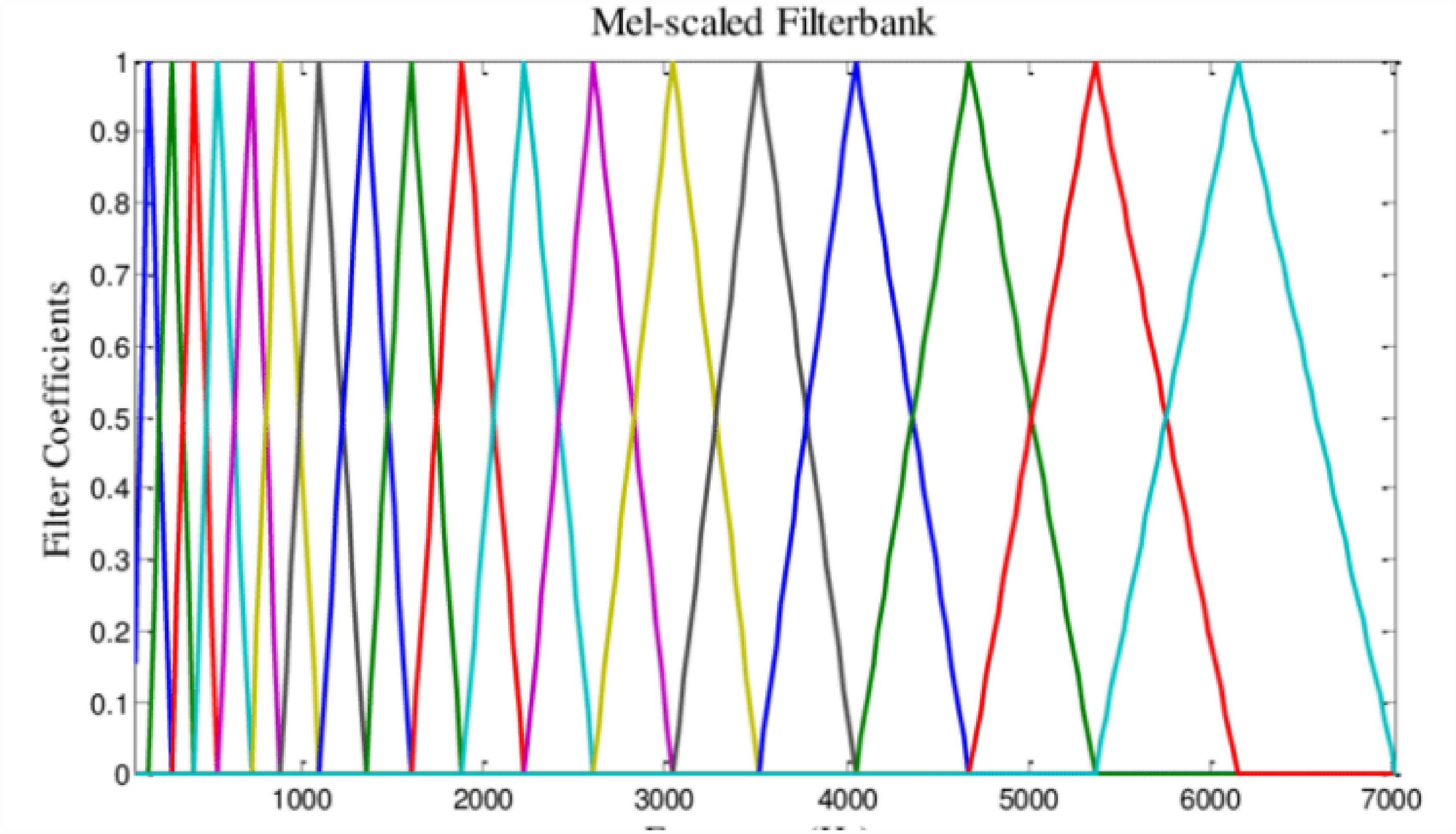
Apply inverse filter

$$H(z) = 1 - \sum_{i=1}^{p} a_i z^{-i}$$ to get lp residual (e[n])

Finding Residual Cepstral Coefficients

```
residual = lp_residual(signal_data)                    (used order 10 here)
rccs = librosa.feature.mfcc(y=residual , sr=fs, n_mfcc=n_rcc)
return np.mean(rccs, axis=1)
```

Step 2: By passing this residual through the MFCCs inbuilt librosa function, we obtain Mel RCC’s



Results: RCCs

- Extracted those 52 RCCs from speech signals.
- Trained only on these **52 RCCs**.

Achieved 98.5% accuracy on the test set.

- Combining these **52 RCCs with 52 MFCC features**.

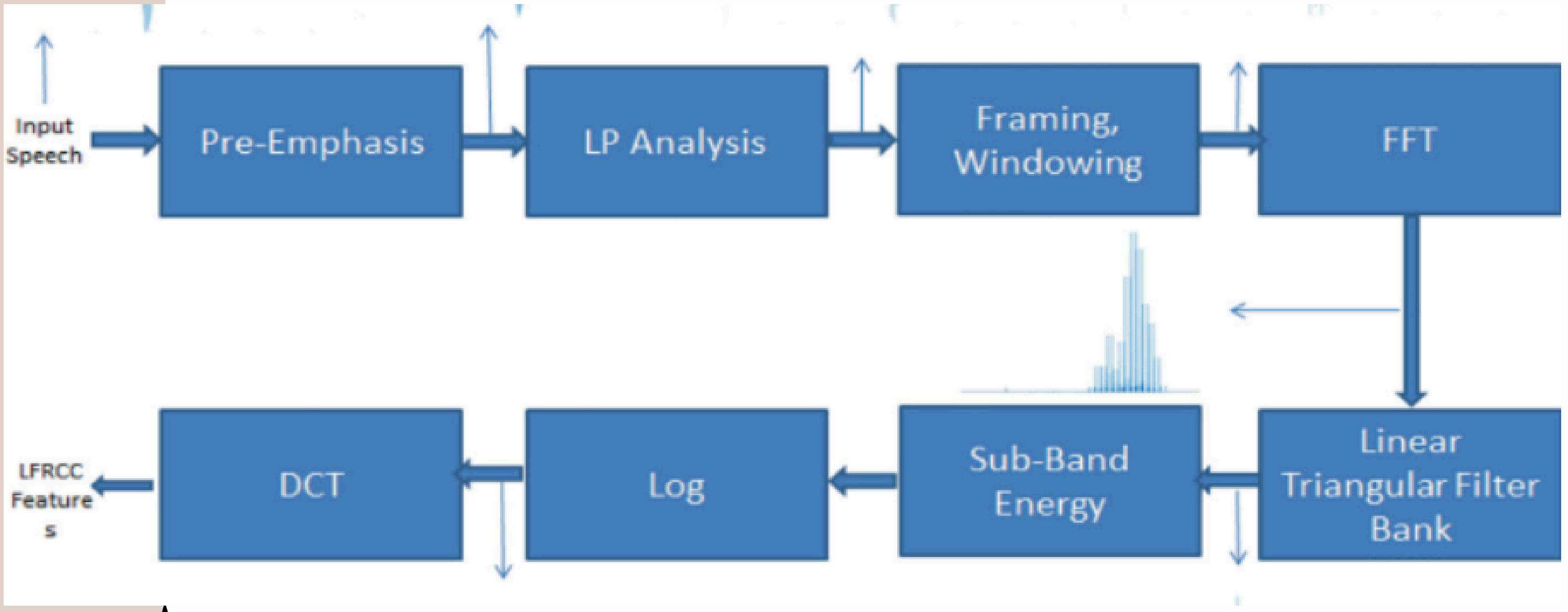
Achieved 98.25% accuracy on the test set.

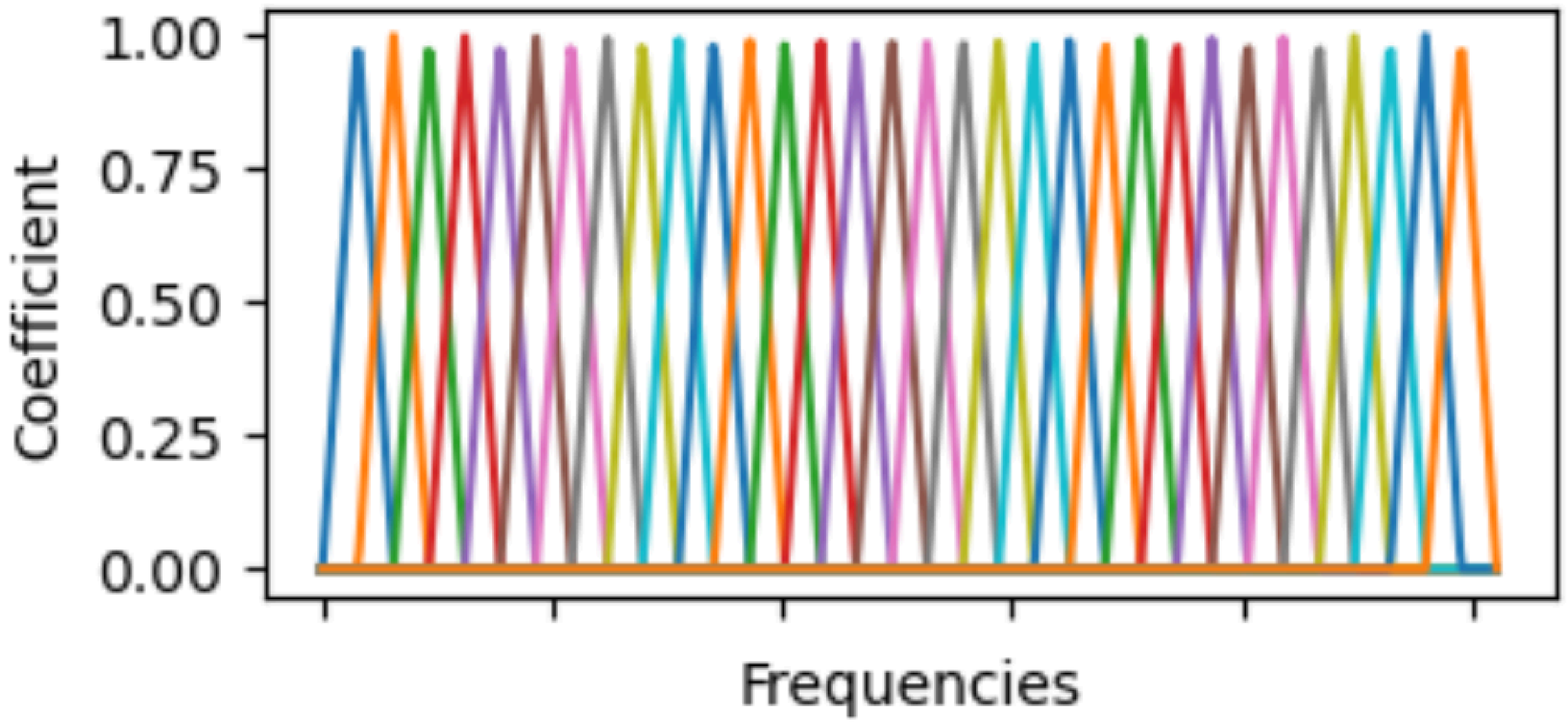
Observation:
RCCs focus on the excitation signal rather than vocal tract resonances, making them highly sensitive to irregularities (breaks, breathiness, etc.) in vocal fold vibration, which are common in dysarthria.

Reference: https://dl.acm.org/doi/10.1007/978-3-031-78498-9_22

Finding LFRCC

Step 1: Finding LP residual (after preemphasis)
Step 2: FFT
Step 3: Linear filter banks (equally spaced triangular)
Step 4: log + DCT





Results: LFRCCs

- Extracted those 52 LFRCCs from speech signals.
- Trained only on these **52 LFRCCs**.

Achieved 87.25% accuracy on the test set.

- Combining these **52 LFRCCs with 52 MFCC features**.

Achieved 98.25% accuracy on the test set.

What next?

Explore more excitation features and implement them.

THANK YOU