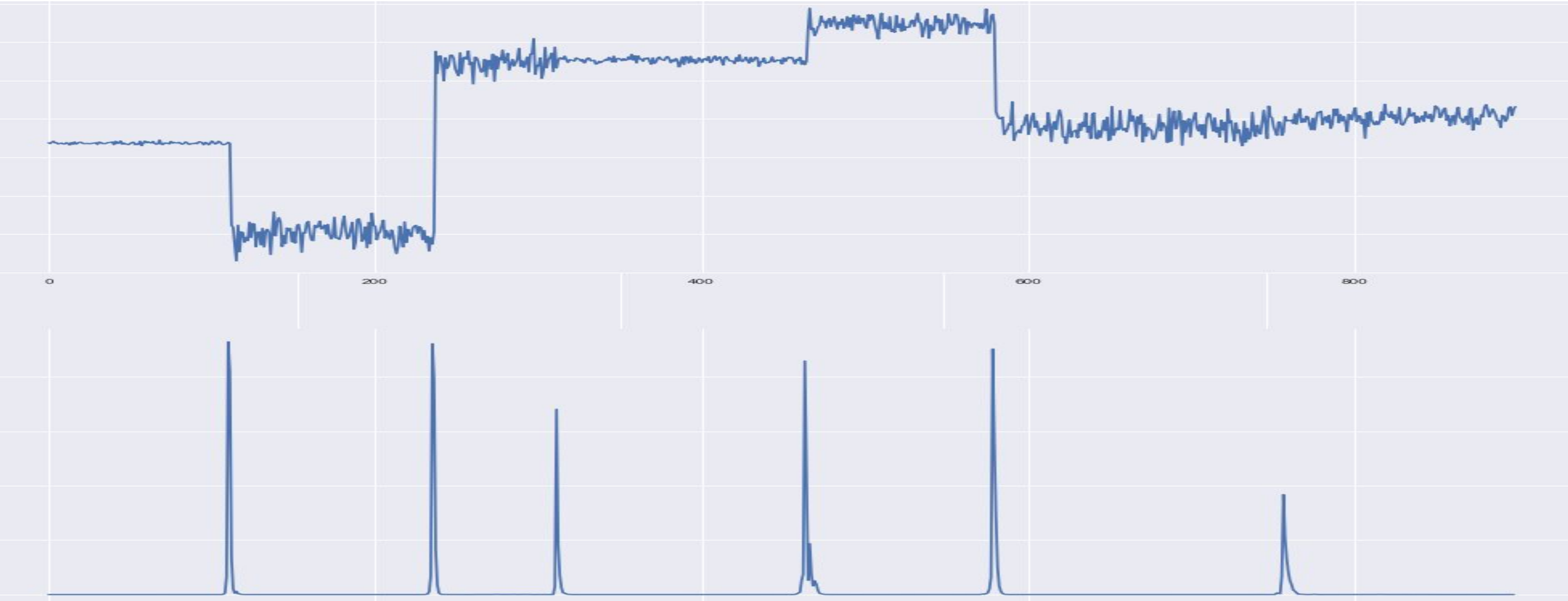
NLP Group

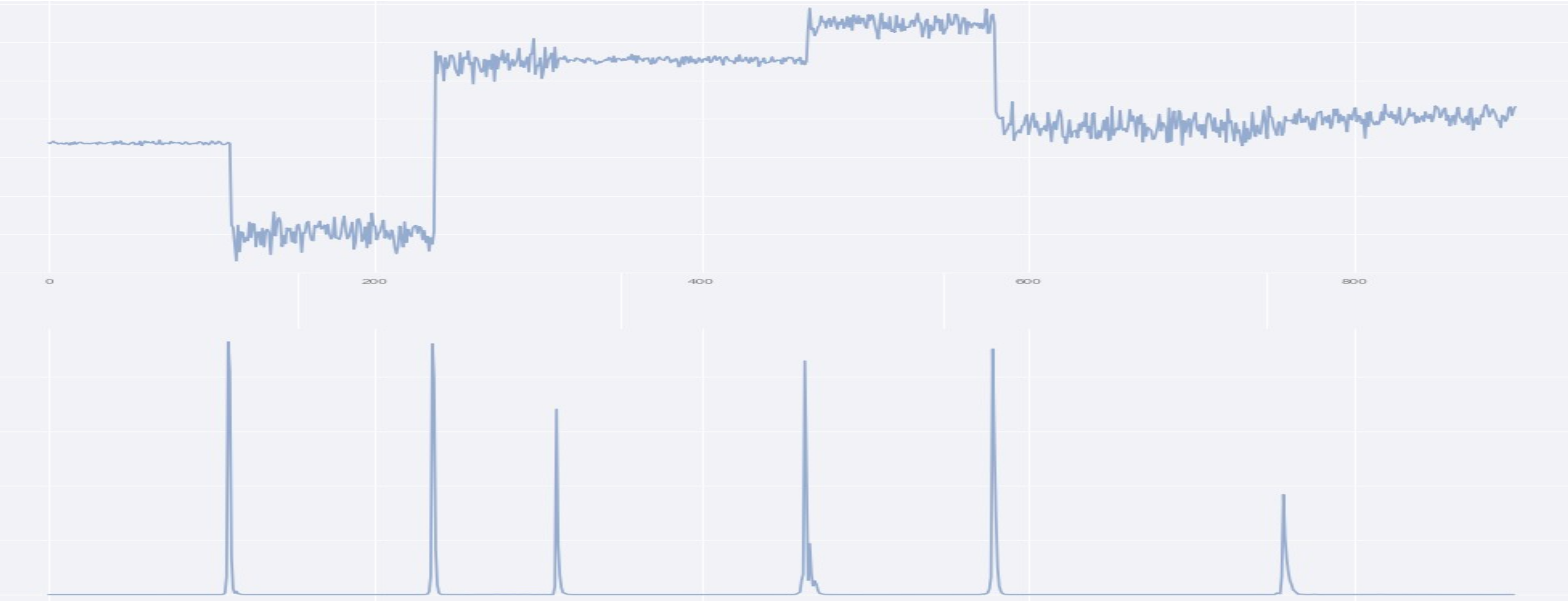
Joe Mosby
Michael Hill
Bhargava Samanthula
Matthew Lewine
Hrisheek Radhakrishnan
Sanath Nagaraj
Aniruddha Das
Anthony D’Achille
Sarah Chen

Bayesian Change Point Detection

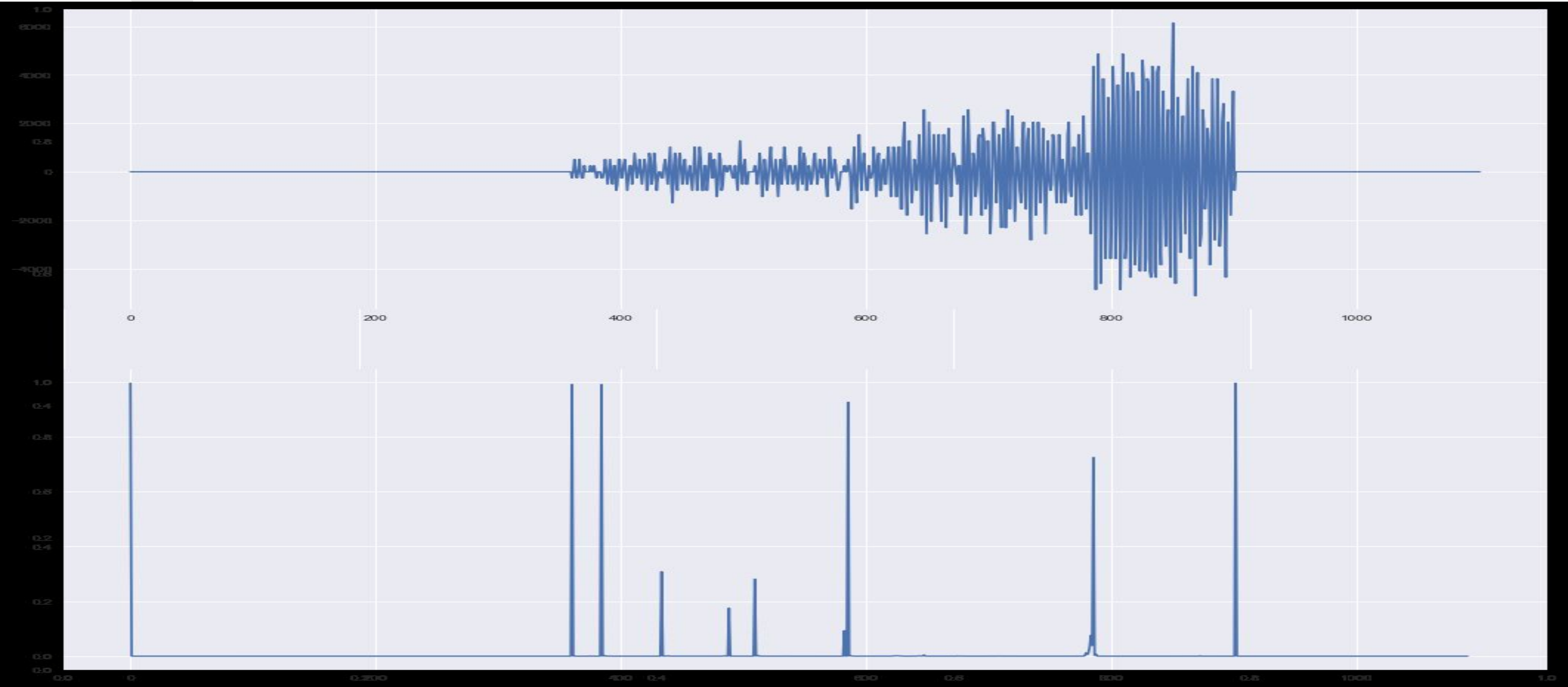


How can this be used?

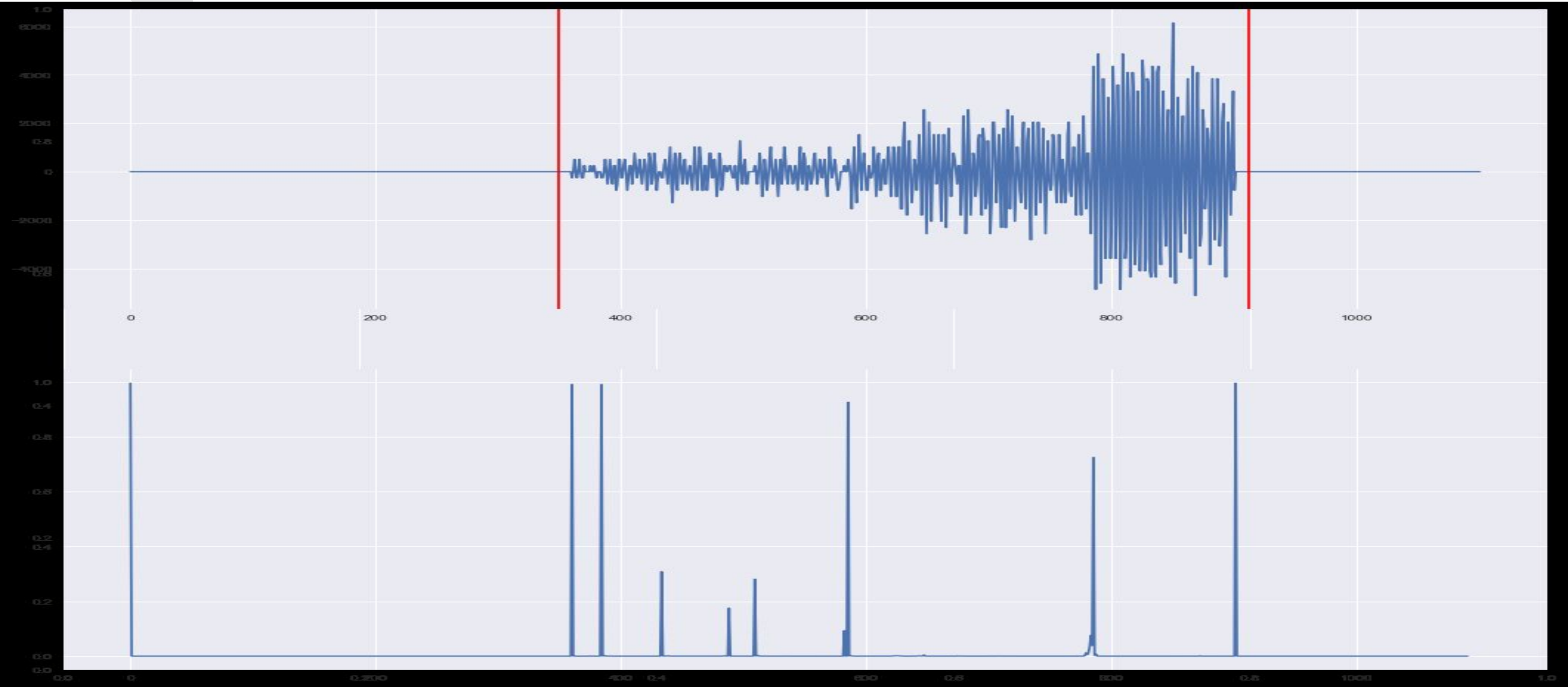
- Stream Data → Stream Data
- Stream Data → Feature Data



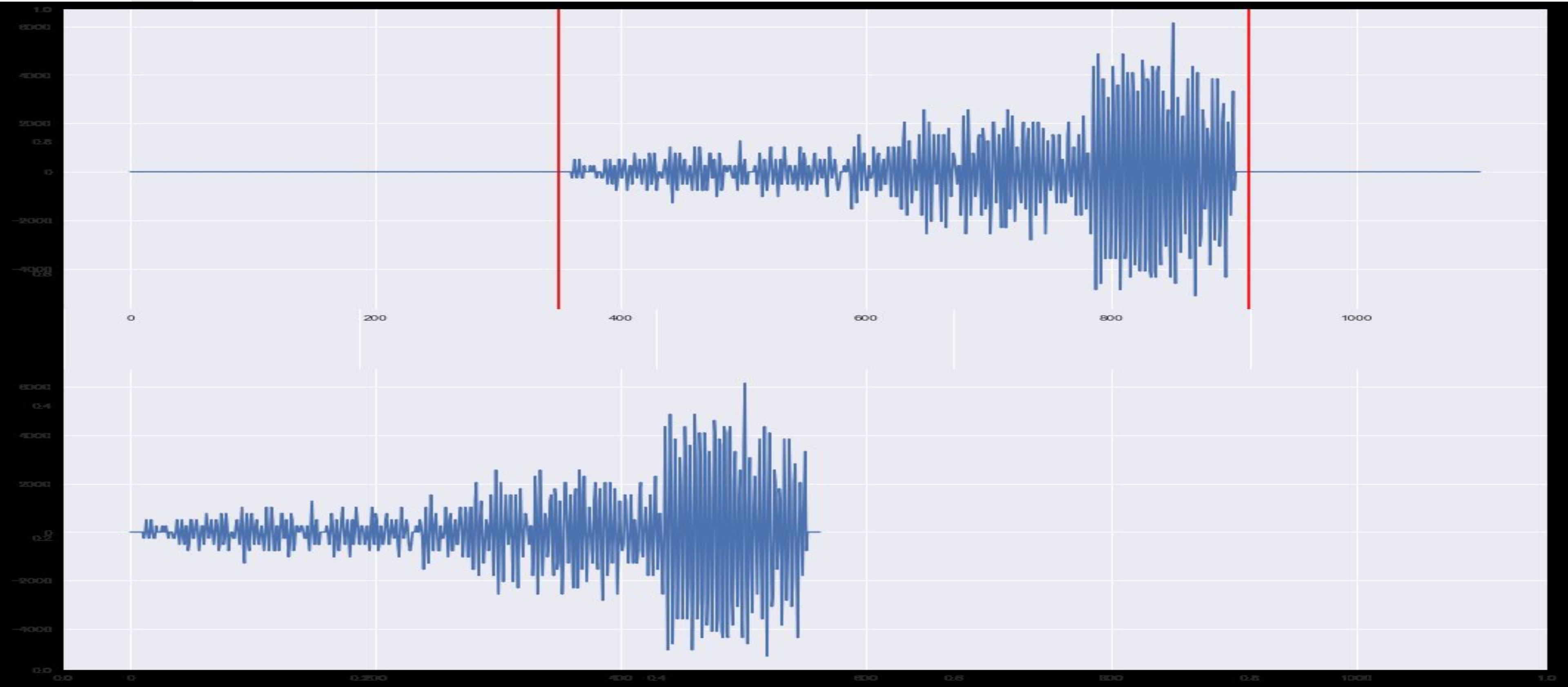
Applying CPD to trim audio (short signal)



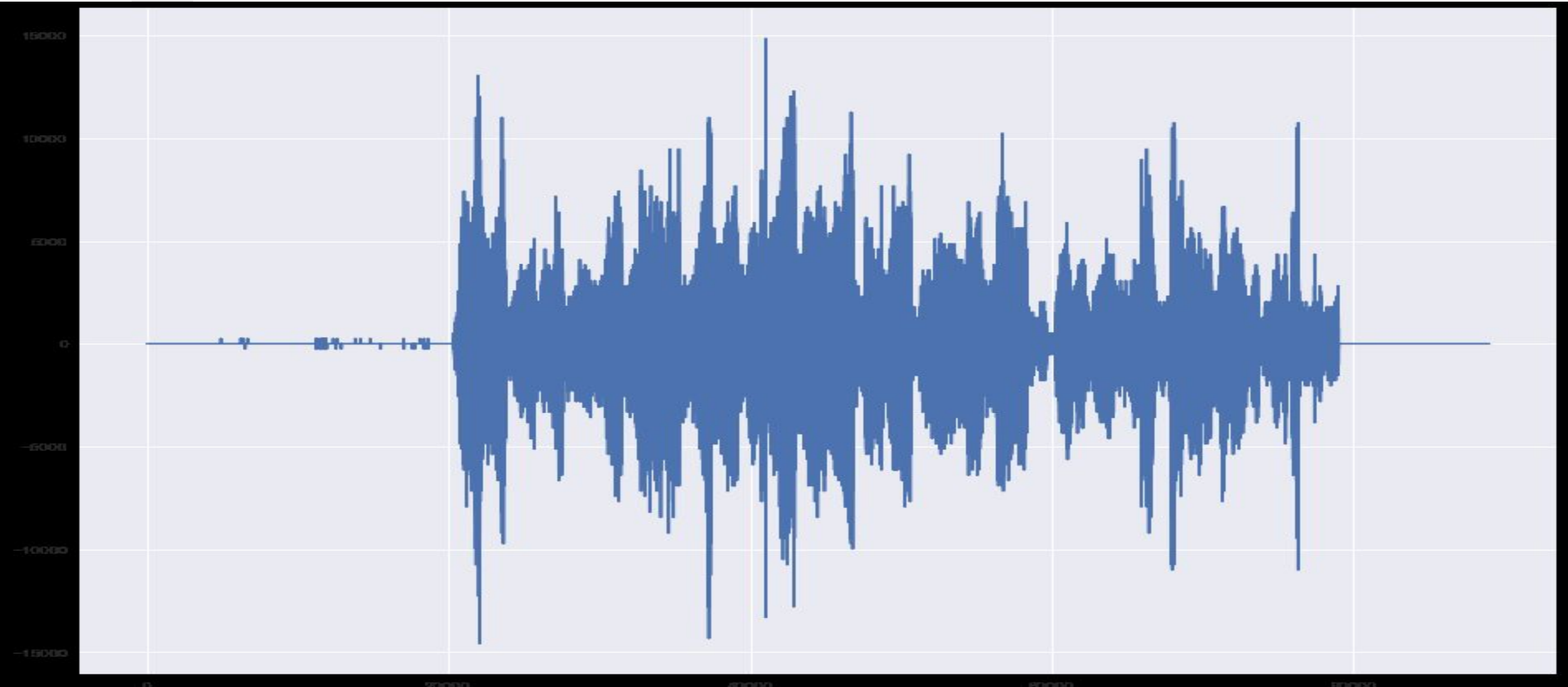
Applying CPD to trim audio (short signal)



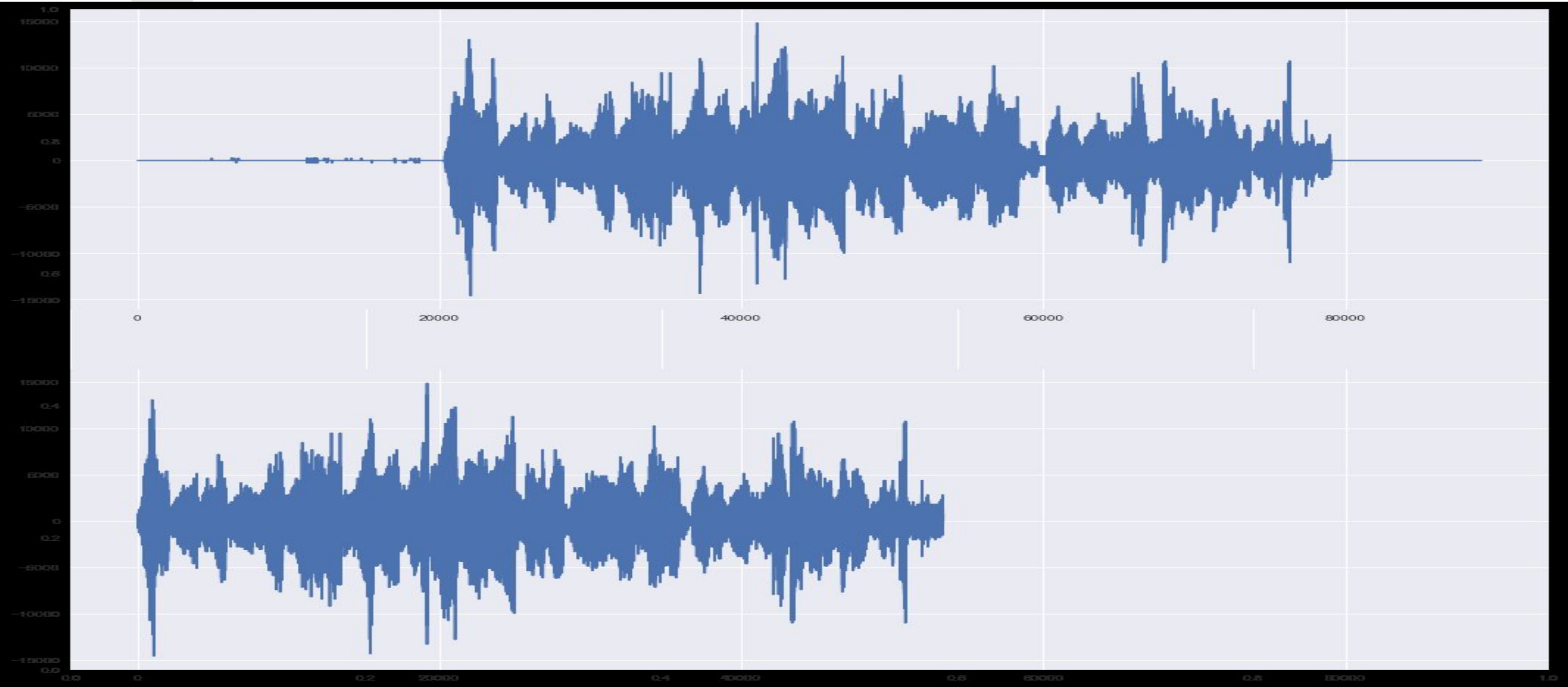
Applying CPD to trim audio (short



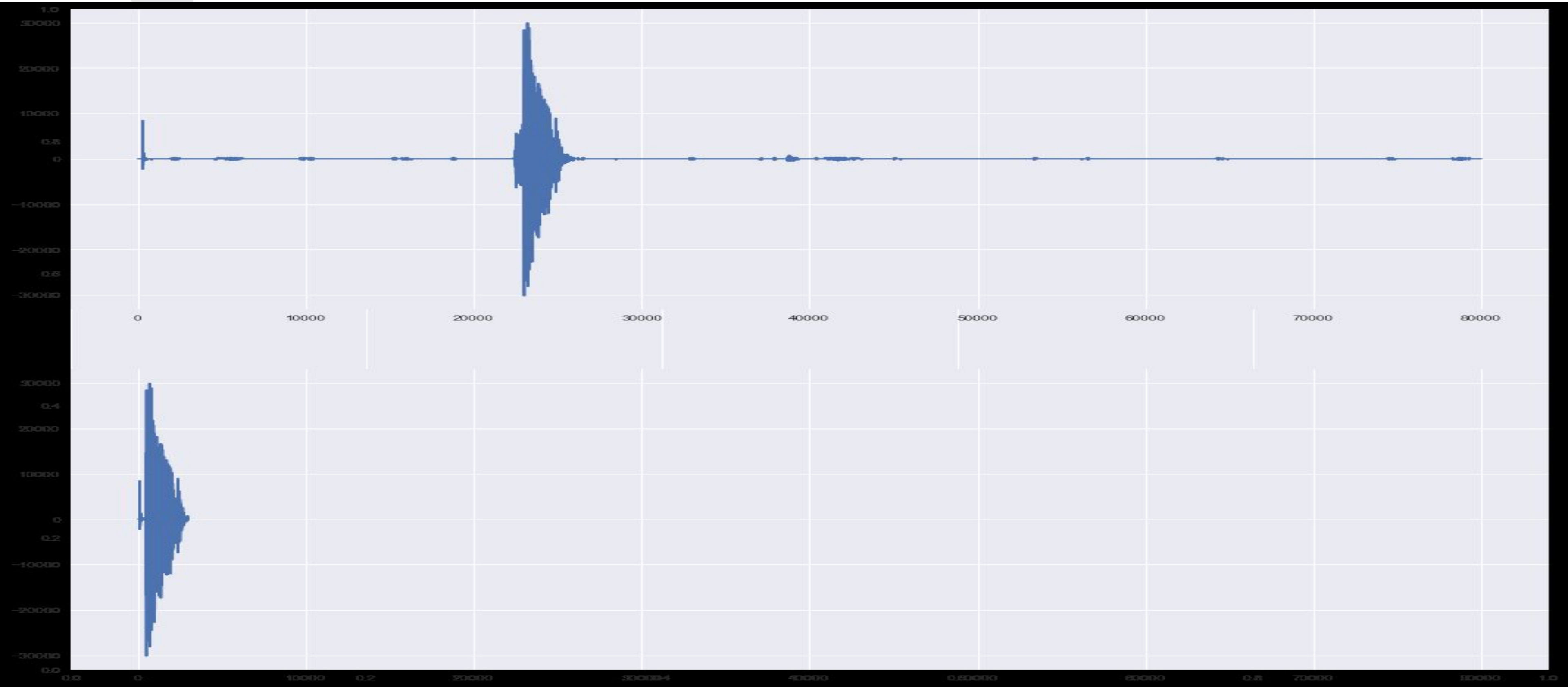
Applying CPD to trim audio (long signal)



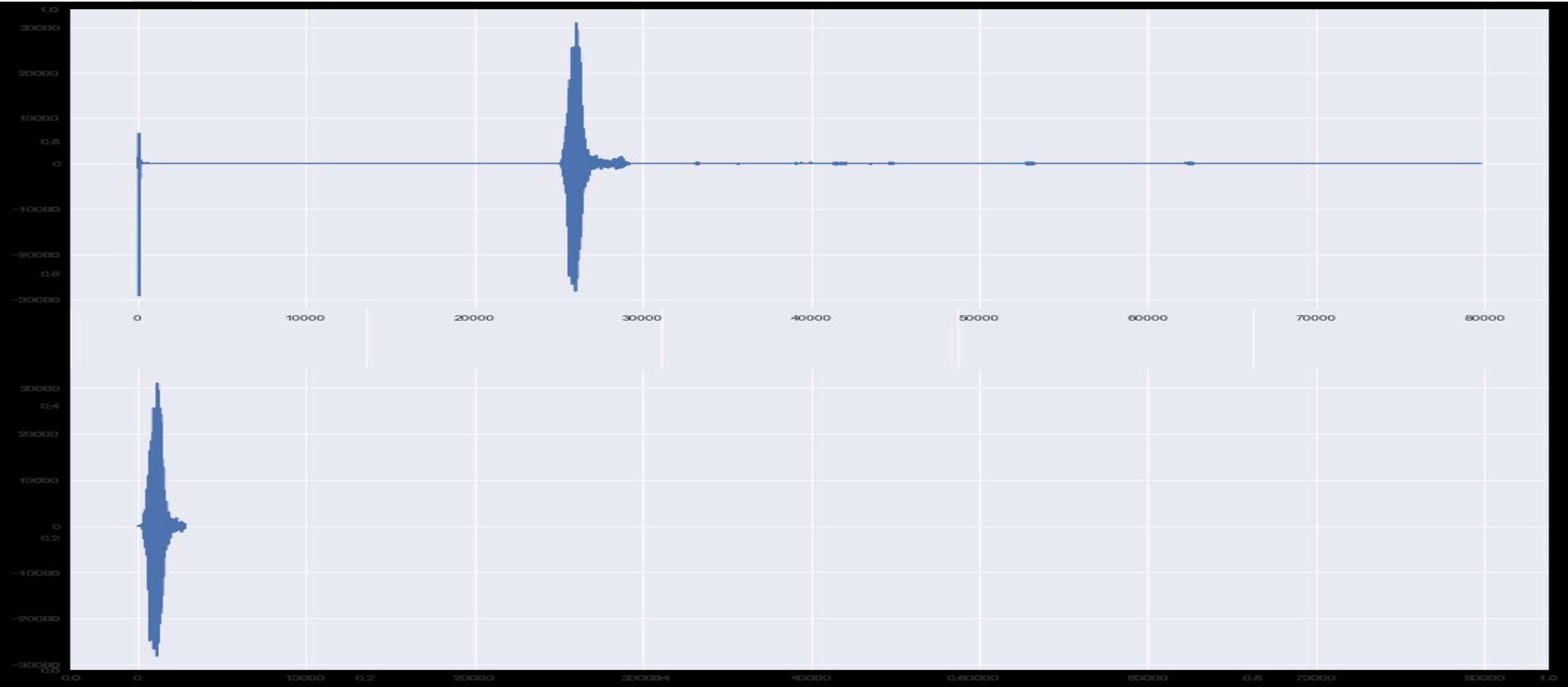
Applying CPD to trim audio (long signal)



Result for speech with “no”

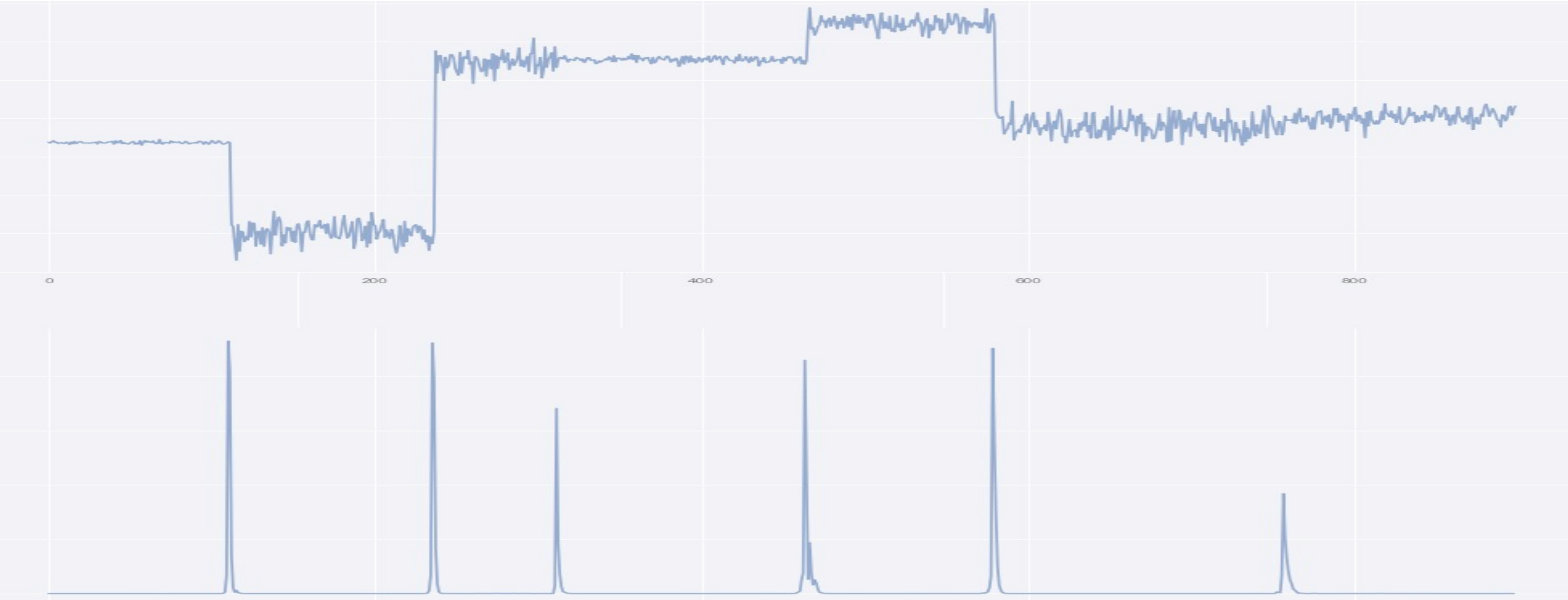


Result for speech with “yes”

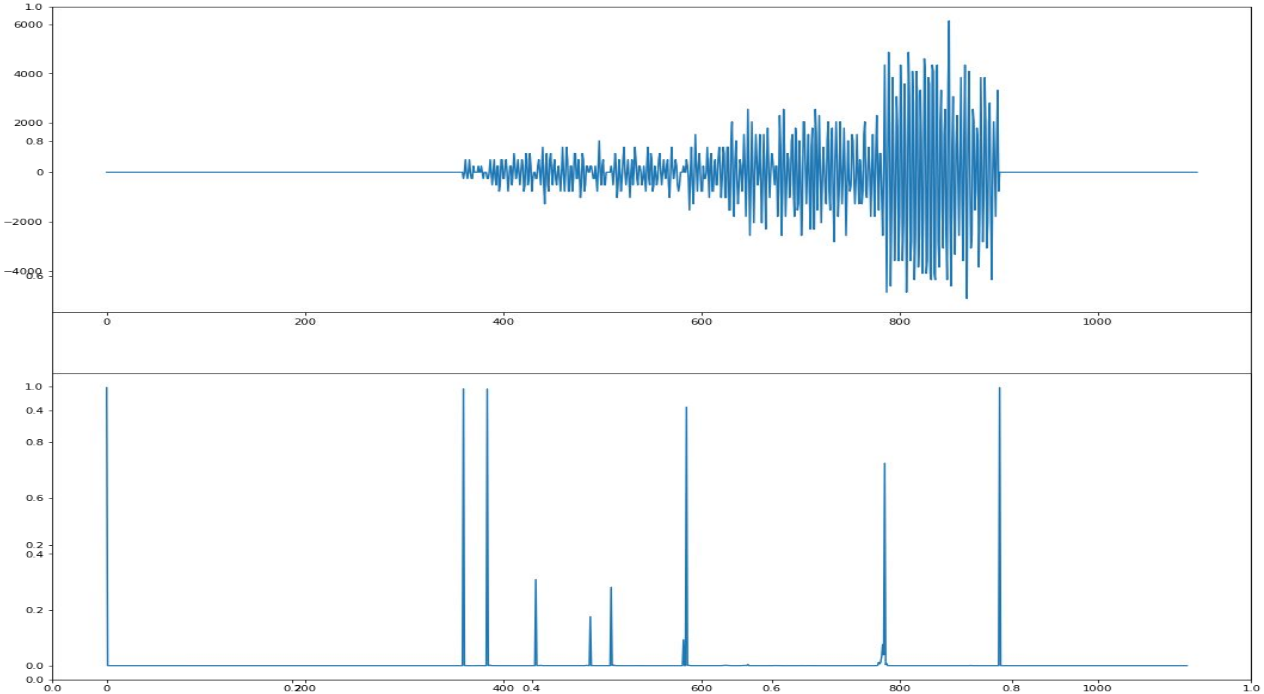


How else can this be used?

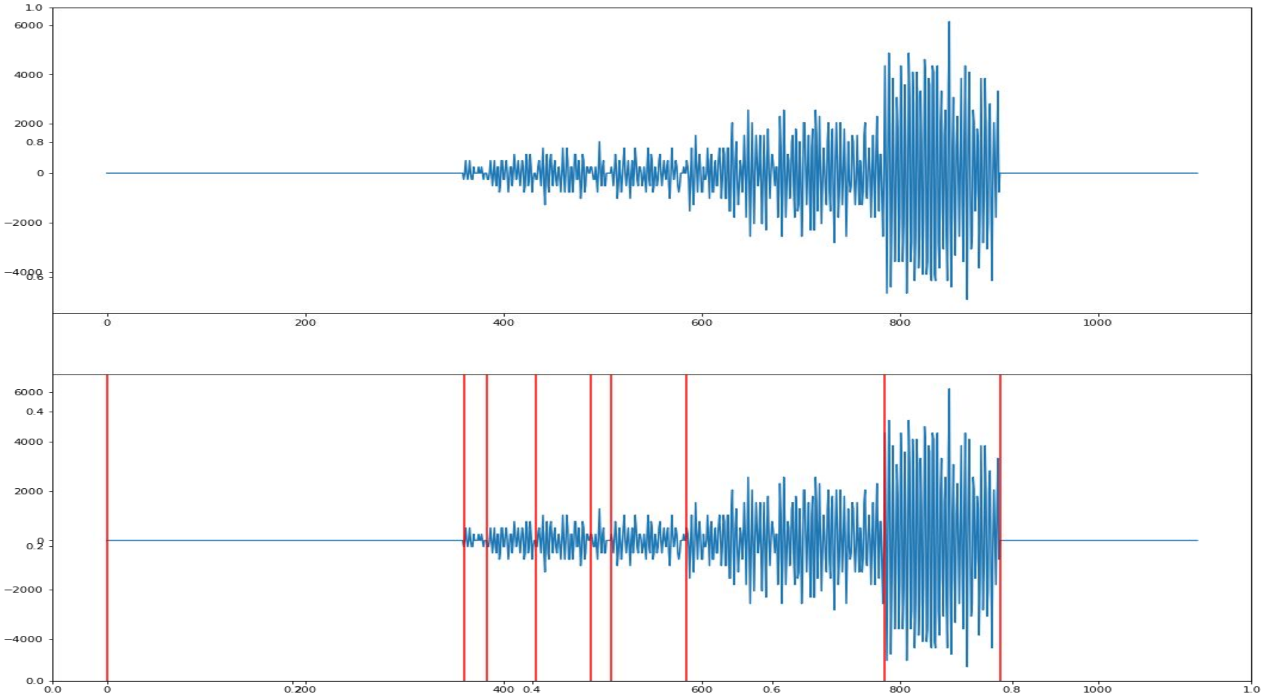
- Stream Data → Stream Data
- Stream Data → Feature Data



Change Point for Feature Data

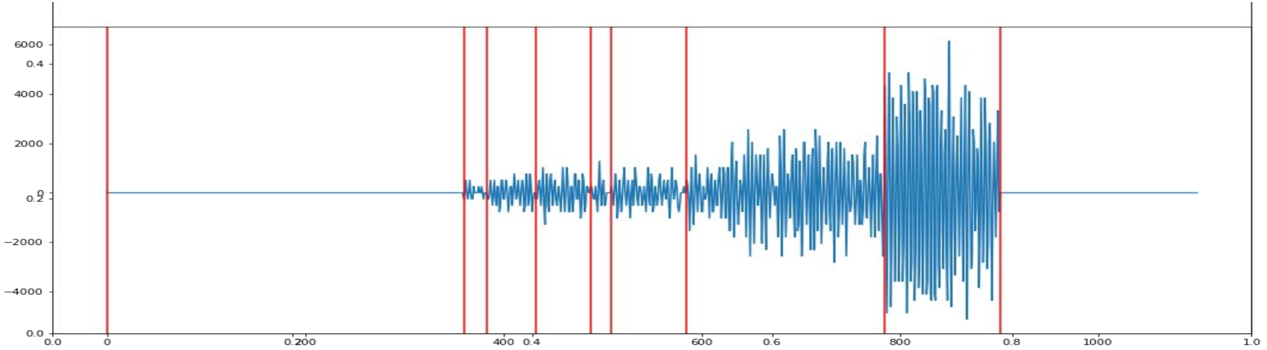


Change Point for Feature Data



Change Point for Feature Data

Sample Feature Data for Below Audio								
Change Pts	0	360	384	433	488	509	585	785
Normalized Change Pts	0.00	0.33	0.35	0.39	0.44	0.46	0.53	0.71
Avg. Freq (Hz)	0.0	2063.6	1860.1	1785.5	1879.0	1762.0	1857.1	1910.5

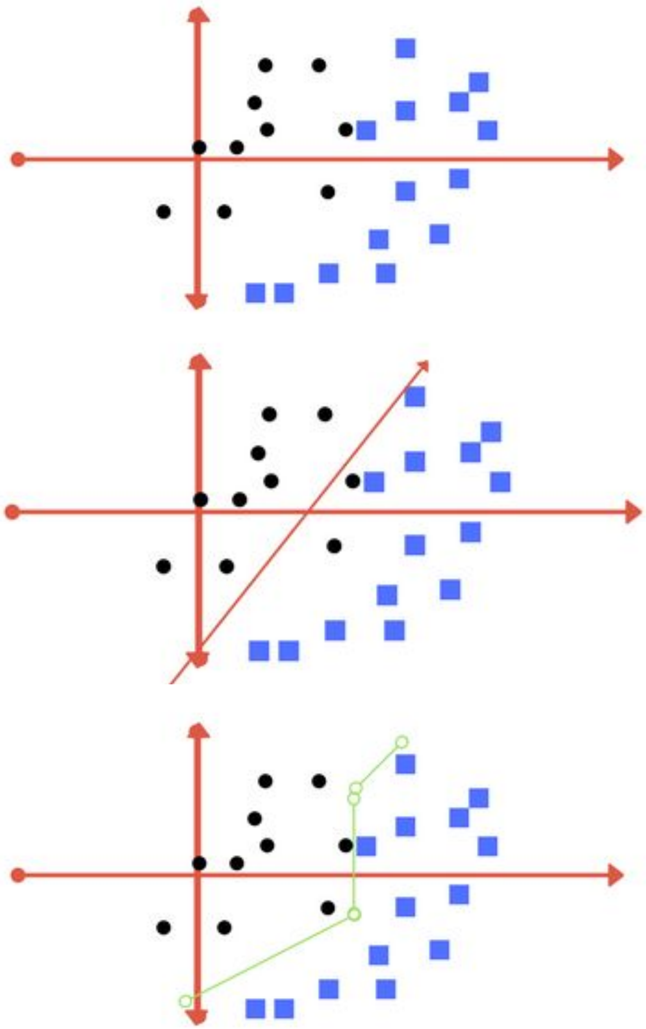


SVM-based Segmentation

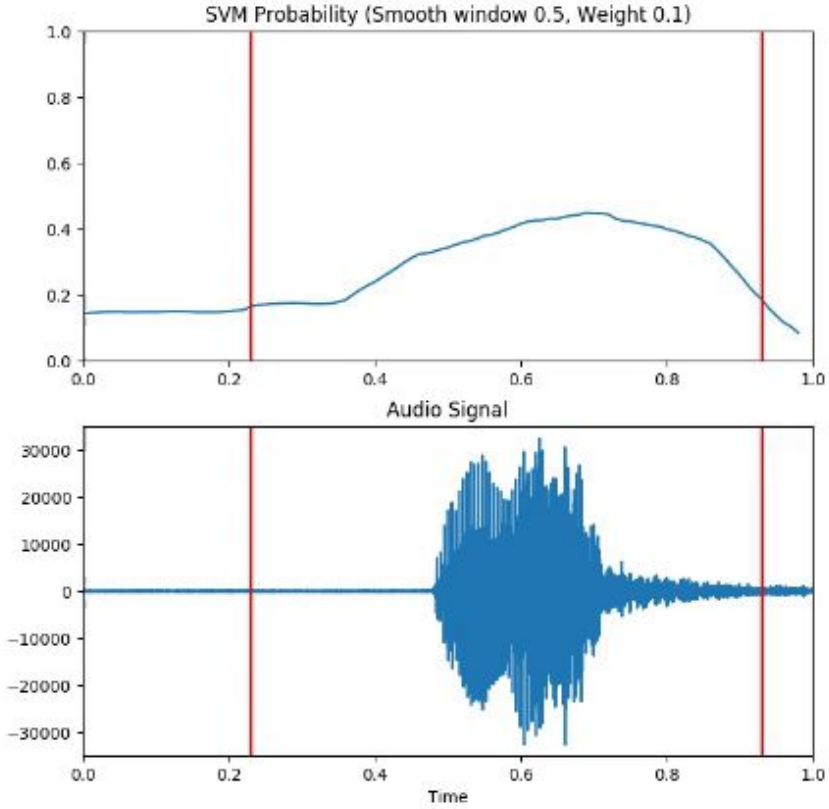
- **Input**: uninterrupted audio recording
- **Output**: Segment endpoints that correspond to audio events
- Semi-supervised approach
  - SVM trains on 10% of highest and 10% of lowest energy frames (The sum of squares of the signal values, normalized by the respective frame length)
  - SVM outputs probabilistic evaluation
- Parameters:
  - Signal
  - Sampling frequency (from audio file)
  - Short-term window size (0.020 s) and step (0.010 s)
  - Window (in seconds) used to smooth the SVM probabilistic sequence
  - Weighting factor between 0 and 1 that specifies how strict to threshold intervals

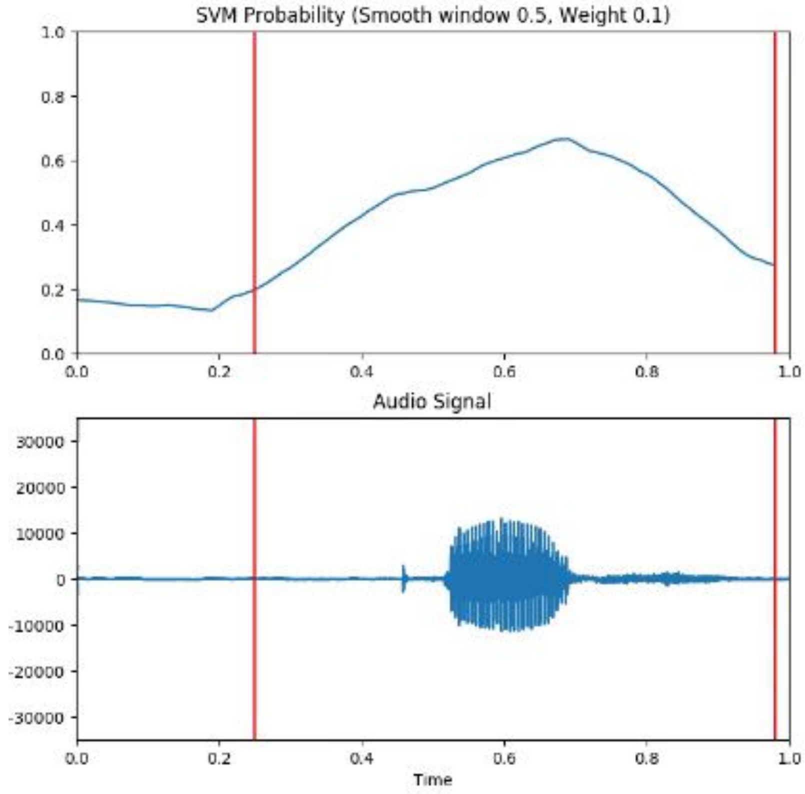
SVM

Given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In 2D space, this hyperplane is a line dividing a plane in two parts where in each class lay in either side

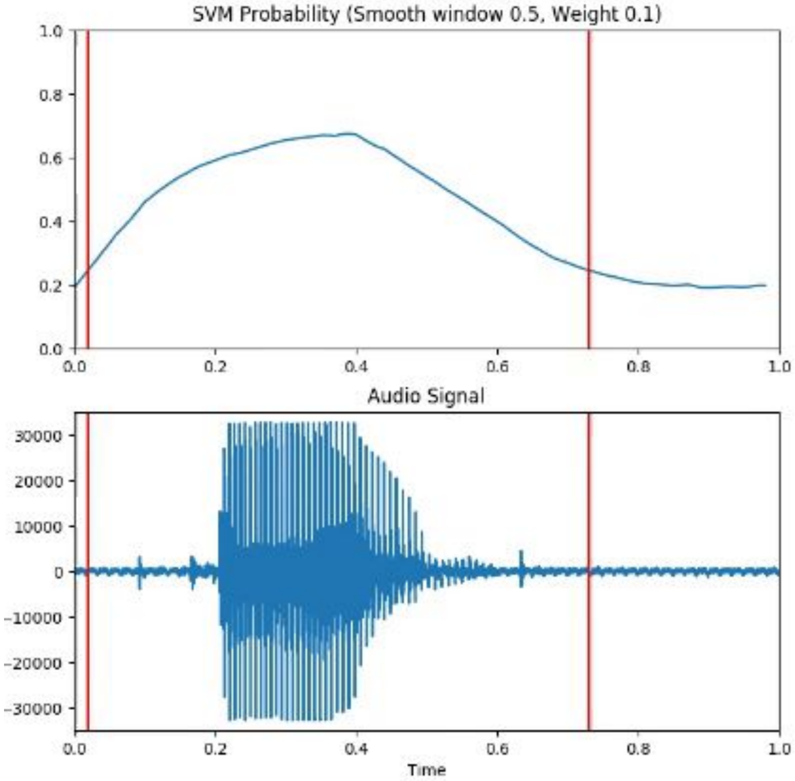


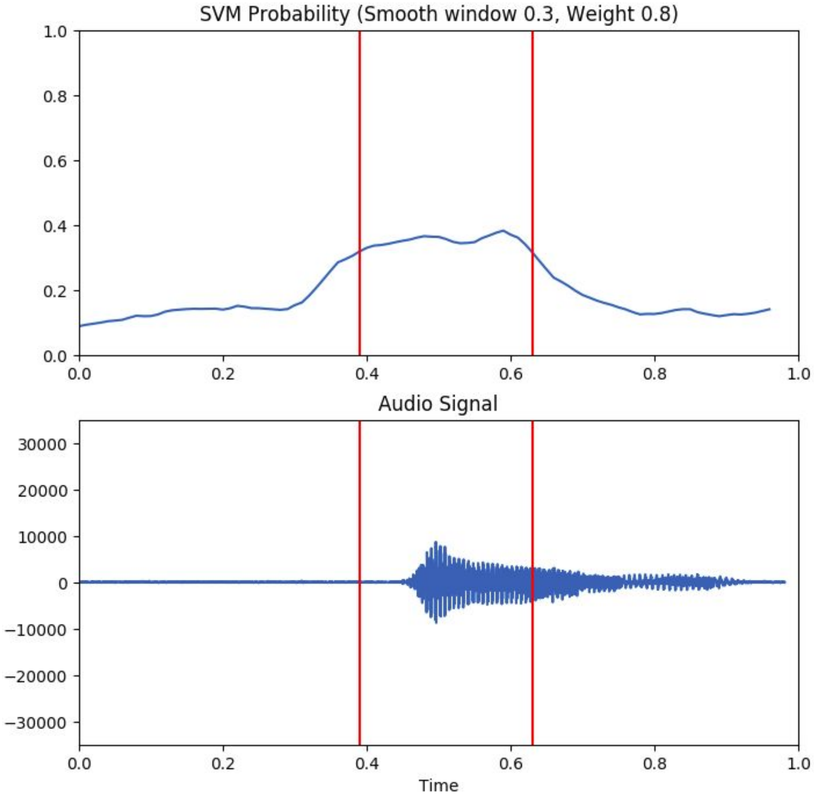
Word: “off”

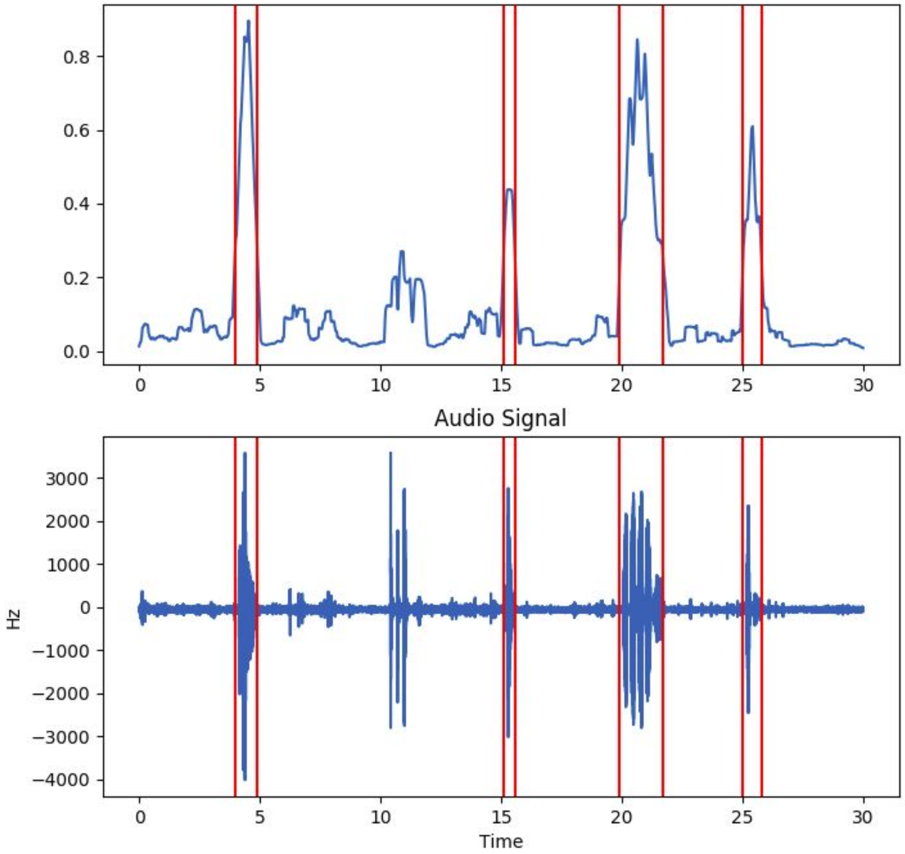

SVM Probability (Smooth window 0.5, Weight 0.1)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Audio Signal
30000
20000
10000
0
−10000
−20000
−30000
0.0
0.2
0.4
0.6
0.8
1.0
Time


SVM Probability (Smooth window 0.5, Weight 0.1)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Audio Signal
30000
20000
10000
0
-10000
-20000
-30000
0.0
0.2
0.4
0.6
0.8
1.0
Time

Word: “on”


SVM Probability (Smooth window 0.5, Weight 0.1)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Audio Signal
30000
20000
10000
0
−10000
−20000
−30000
0.0
0.2
0.4
0.6
0.8
1.0
Time


SVM Probability (Smooth window 0.3, Weight 0.8)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
Audio Signal
30000
20000
10000
0
−10000
−20000
−30000
0.0
0.2
0.4
0.6
0.8
1.0
Time


0.8
0.6
0.4
0.2
0.0
0
5
10
15
20
25
30
Audio Signal
3000
2000
1000
0
−1000
−2000
−3000
−4000
Hz
0
5
10
15
20
25
30
Time

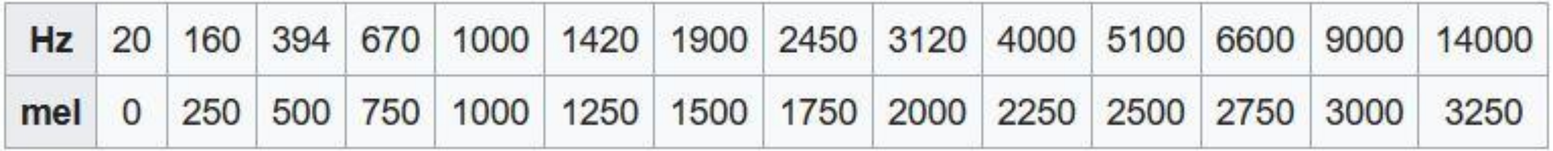
Mel-Frequency Cepstrum Coefficients (MFCC’s)

- A set of features widely used for speech recognition
- Rather than directly using sound frequencies, it relies on the Mel Scale

Mel Scale

- Pitch is determined by a sound’s frequency, measured in Hertz
- Humans do not perceive pitch as a linear function of frequency
  - To the human ear, doubling the frequency, doesn’t mean the pitch sounds twice as high
- Mel Scale - a perceptual scale of pitches judged by listeners to be equal in distance from one another

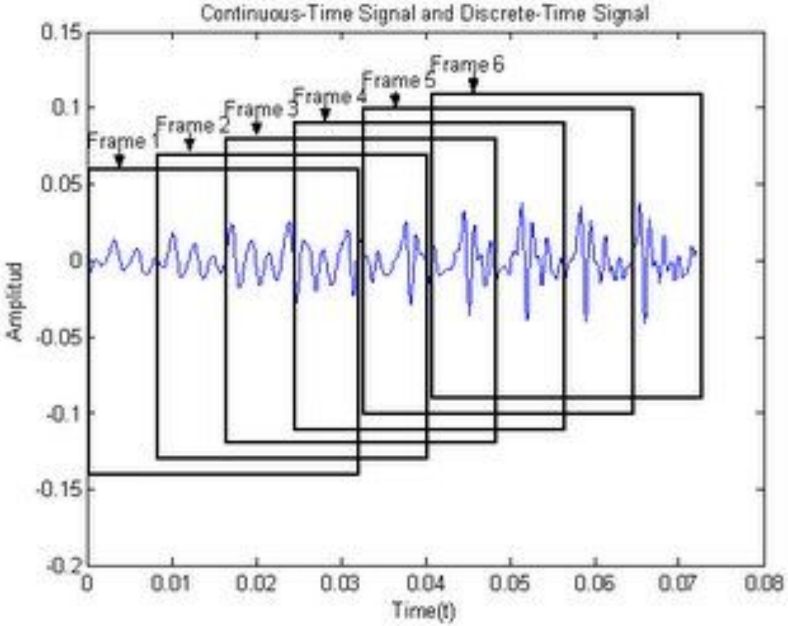
$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right) = 1127 \ln\left(1 + \frac{f}{700}\right)$$



Hz	20	160	394	670	1000	1420	1900	2450	3120	4000	5100	6600	9000	14000
mel	0	250	500	750	1000	1250	1500	1750	2000	2250	2500	2750	3000	3250

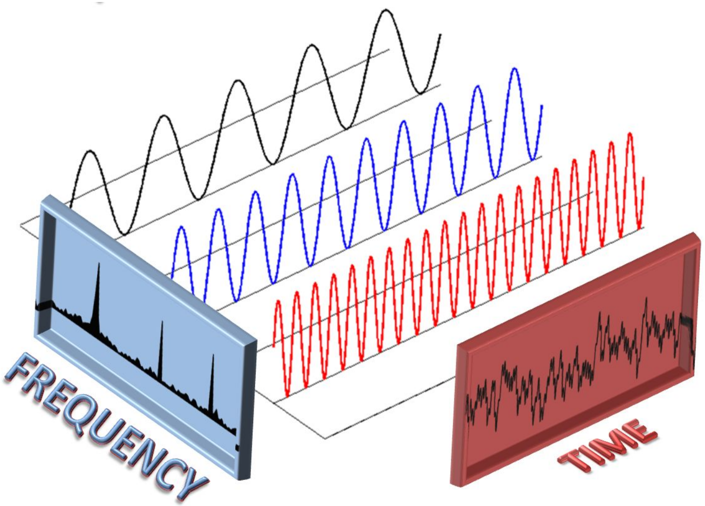
Derivation of MFCC’s

1. Segment the signal into a set of overlapping frames



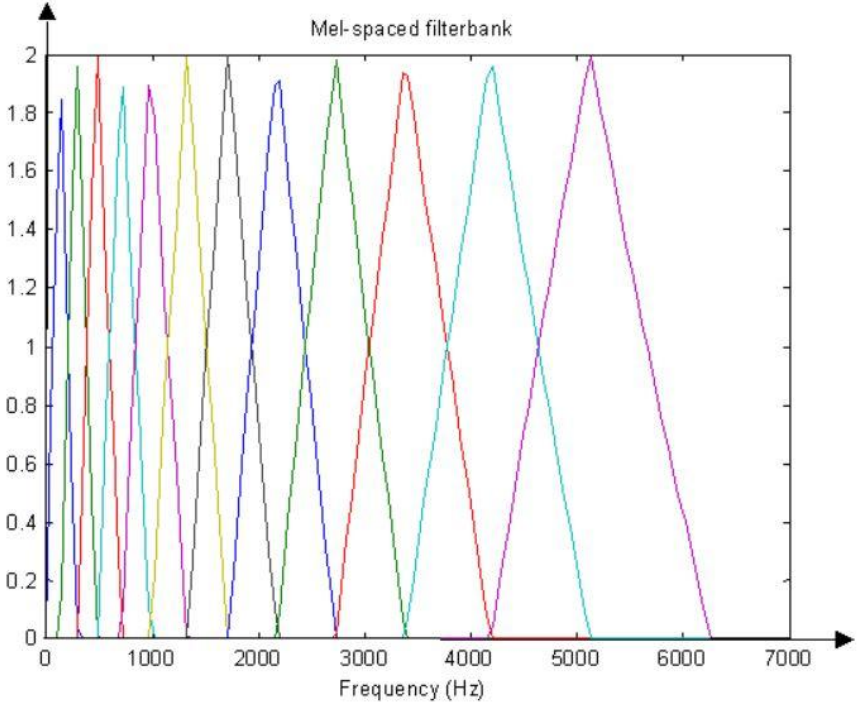
Derivation of MFCC’s

1. Segment the signal into a set of overlapping frames
2. Calculate the Fourier Transform on a frame of the signal



Derivation of MFCC's

1. Segment the signal into a set of overlapping frames
2. Calculate the Fourier Transform on a frame of the signal
3. Compute the signal power through a bank of filters tuned to mel-scaled frequencies



Derivation of MFCC’s

1. Segment the signal into a set of overlapping frames
2. Calculate the Fourier Transform on a frame of the signal
3. Compute the signal power through a bank of filters tuned to mel-scaled frequencies
4. Take the logs of the signal powers at each of the mel banks
5. Apply the discrete cosine transform on the list of log mel powers

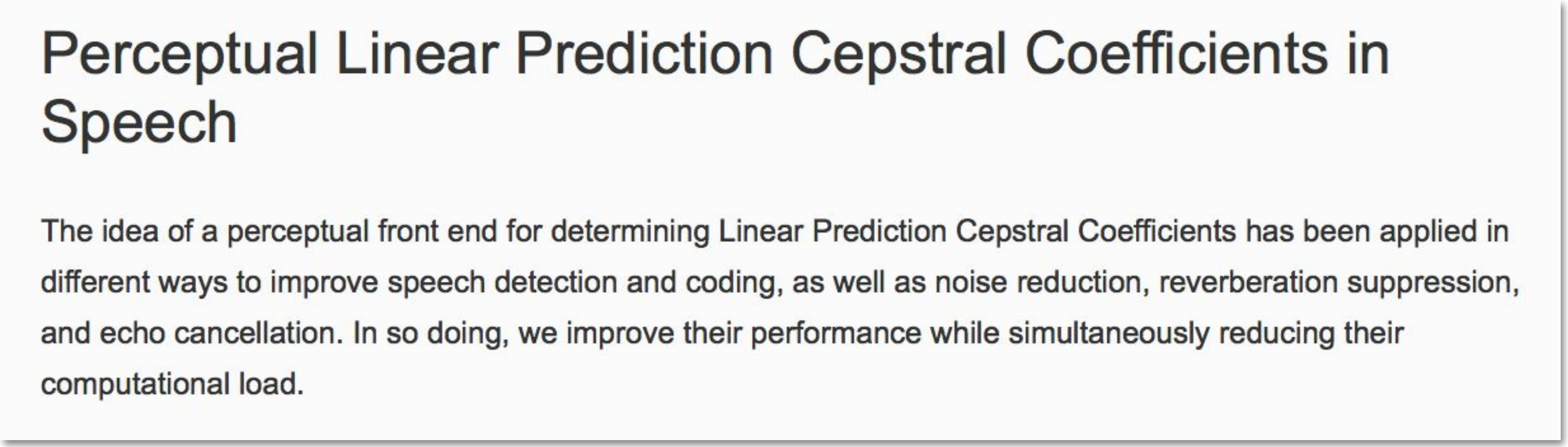
$$X_k = \frac{1}{2}(x_0 + (-1)^k x_{N-1}) + \sum_{n=1}^{N-2} x_n \cos\left[\frac{\pi}{N-1}nk\right] \qquad k = 0, \ldots, N-1$$

Modulated Parameters

- Four primitives: MFCC, Logfbank, MFCC_mod, Logfbank_mod
- MFCC: computes Mel Frequency Cepstral Coefficients of an audio signal (from python_speech_features)
- Logfbank: computes Log Mel-Filterbank energies of an audio signal
- MFCC_mod and Logfbank_mod: allowed EMADE to modulate default arguments by multiplying by a float value
    - Window Length (in seconds)
    - Window step (time between windows)
    - Number of Cepstrum to return
    - Number of filters in filterbank
    - Size of the FFT (number of bins in the analysis window)

Perceptual Linear Prediction (PLP)

- Focus on improving accuracy while also reducing outside/extra noise
- Mel-frequency cepstrum coefficient



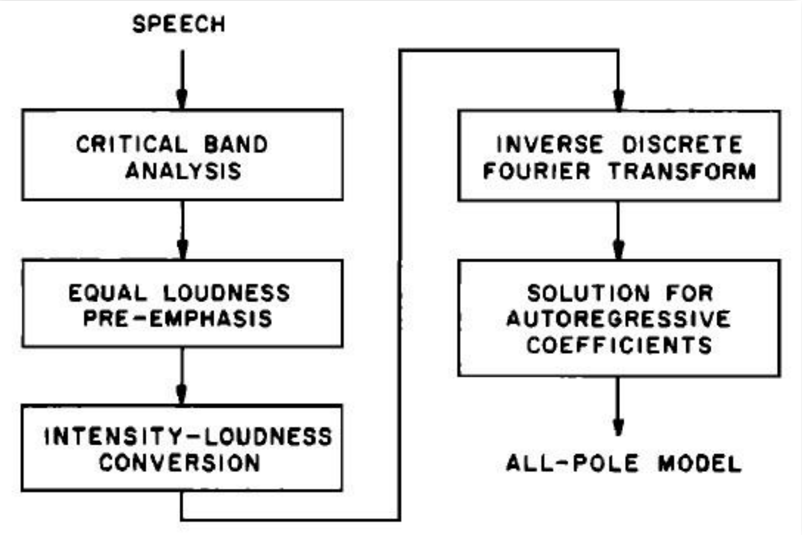
Perceptual Linear Prediction Cepstral Coefficients in Speech

The idea of a perceptual front end for determining Linear Prediction Cepstral Coefficients has been applied in different ways to improve speech detection and coding, as well as noise reduction, reverberation suppression, and echo cancellation. In so doing, we improve their performance while simultaneously reducing their computational load.

https://www.vocal.com/perceptual-filtering/perceptual-linear-prediction-cepstral-coefficients-in-speech/

Perceptual Linear Prediction (PLP)

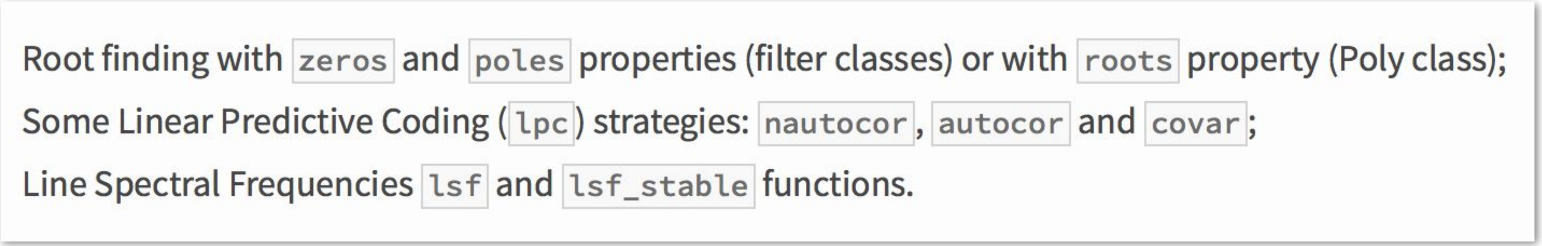
- Computationally efficient and yields a low-dimensional representation of speech. **Useful for speaker-independent automatic speech recognition.**
    - Used in telephone applications and voice response systems
    - Doesn’t require one person to train their voice multiple times; used by anyone but with a limited/smaller number of recognized words
- More consistent with human hearing than conventional Linear Prediction (LP).
- Python Package: audiolazy 0.6 → real-time expressive digital signal processing


SPEECH
CRITICAL BAND ANALYSIS
EQUAL LOUDNESS PRE-EMPHASIS
INTENSITY-LOUDNESS CONVERSION
INVERSE DISCRETE FOURIER TRANSFORM
SOLUTION FOR AUTOREGRESSIVE COEFFICIENTS
ALL-POLE MODEL

Perceptual Linear Prediction (PLP)

- Tried implementing into EMADE - unsuccessful due to a few reasons
  - More complicated than realized - for me and EMADE
  - Not as efficient/accurate as expected (but probably because of incorrect implementation)

- What about EMADE needs to change so it can support PLP?



Root finding with `zeros` and `poles` properties (filter classes) or with `roots` property (Poly class);

Some Linear Predictive Coding (`lpc`) strategies: `nautocor`, `autocor` and `covar`;

Line Spectral Frequencies `lsf` and `lsf_stable` functions.

Perceptual Linear Prediction (PLP)

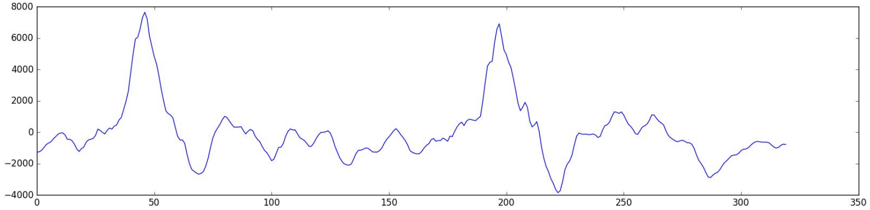
- Next Steps:
  - Continue working on PLP since it can prove to be very useful (especially in terms of speaker-independence)
- What to work on in EMADE?
  - Accurate implementation required (more work and expertise)
  - Fix efficiency issues

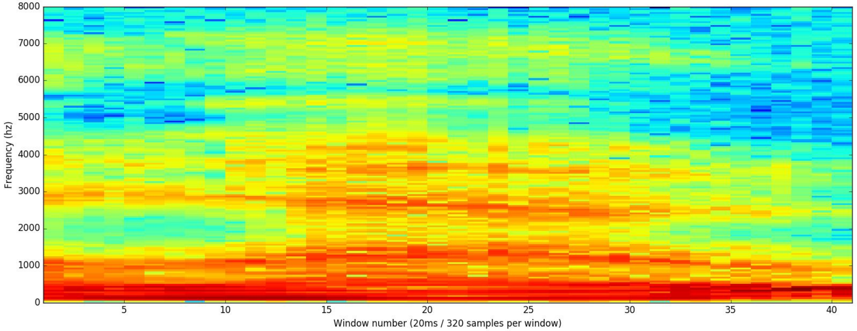
Neural Networks for Speech Recognition

- NN’s - layers of interconnected nodes that “learn” tasks by analyzing examples
- First applied to speech recognition around 1980, but did not get much attention until decades later
    - Large increases in data and computing power showed the promise of NN’s with speech
- Strategy:
    - Take a sampling of a recording, composed of the amplitude values of the sound wave at each moment in time
    - Preprocess the data to reduce complexity and define the sampling by its major components
    - Learn the neural network on these values to classify them as words

NN - Preprocessing

- Take Samplings of recordings, composed of 16,000 amplitude values, are very complex
  - The top image on the right is 20ms of data
- Fourier Transform:
  - Break apart a complex sound wave into the simple sound waves that make it up
  - Sum the energy of each simple wave within each chunk
  - Using the sum, score each frequency band by importance





NN - Classification

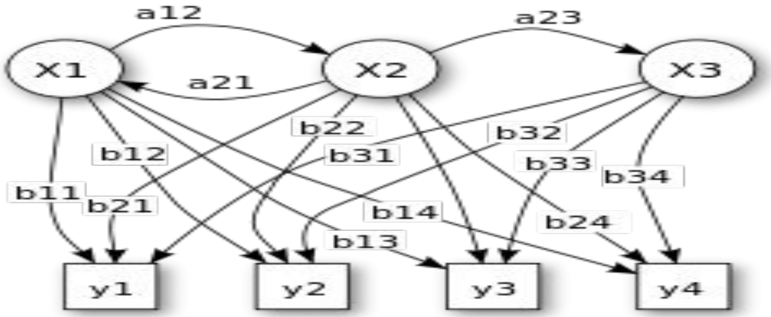
- SciKit - Multilayer Perceptron
  - alpha = 0.0001, constant learning rate, momentum = 0.9
  - 16,000 node input layer (one for each feature/time subsection), 2 100 node hidden layers, 1 node output layer (classifies yes or no whether the sound wave matches the word)
- Keras - Deep Neural Network
  - Dense Layers - each node in one layer is connected to each node in the next
  - Same input and output layers, 2 128 node hidden layers

NN - Next Steps

- Recurrent Neural Network - can use internal state to process sequences of inputs
  - Essentially has a memory that it can use to influence future predictions
  - Used to split up the classification of spoken words into individual spoken letters, where the previously classified letters in the word influence the next letter to be classified
    - i.e. With the word “Hello”, if “Hell” is already classified, then the next letter has a higher chance of being classified as “o” because it makes sense for “o” to come after “Hell”
- Fine tune parameters

Hidden Markov Models

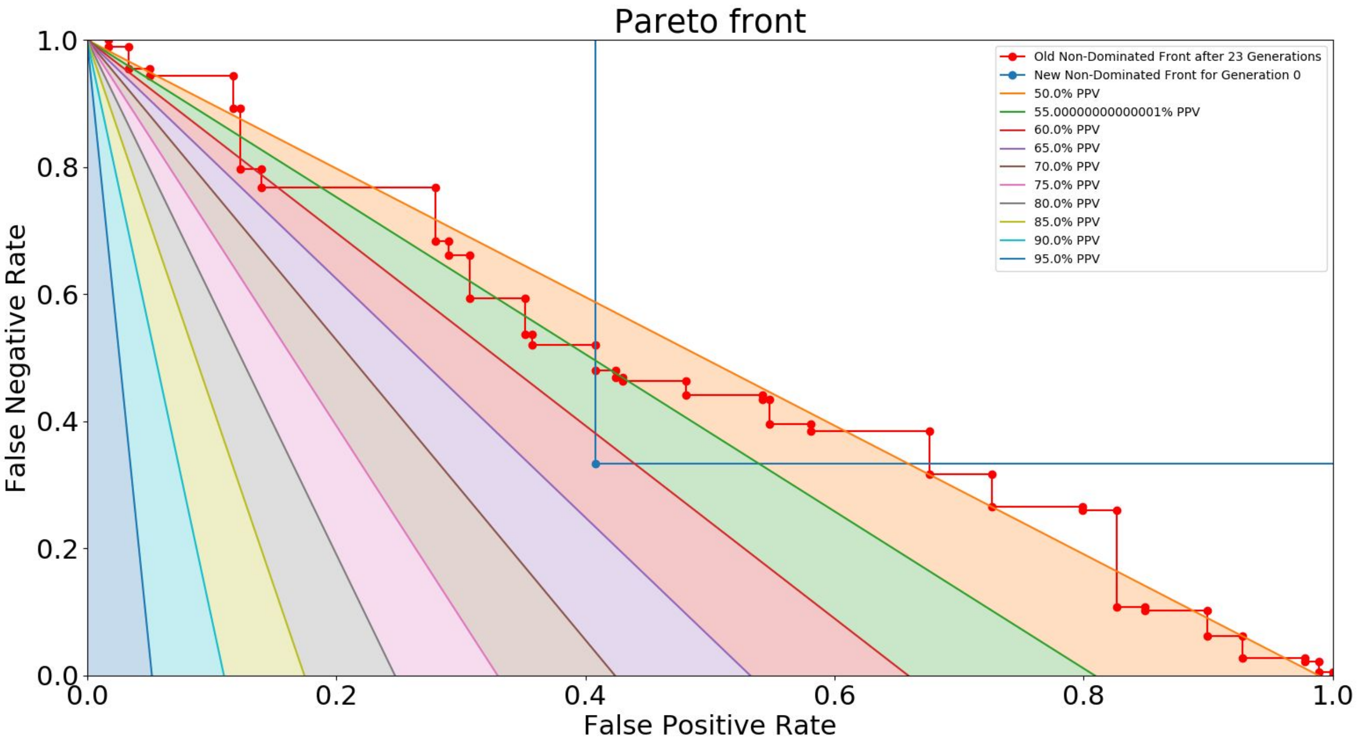
- Used HMMLearn python package
- 1) For each target class, initialize an HMM
- 2) Train each model with all of the instances of its respective class in the training data
- 3) Score each model on each instance of the testing data and set the target value of each test instance as the name of the HMM model with the highest score



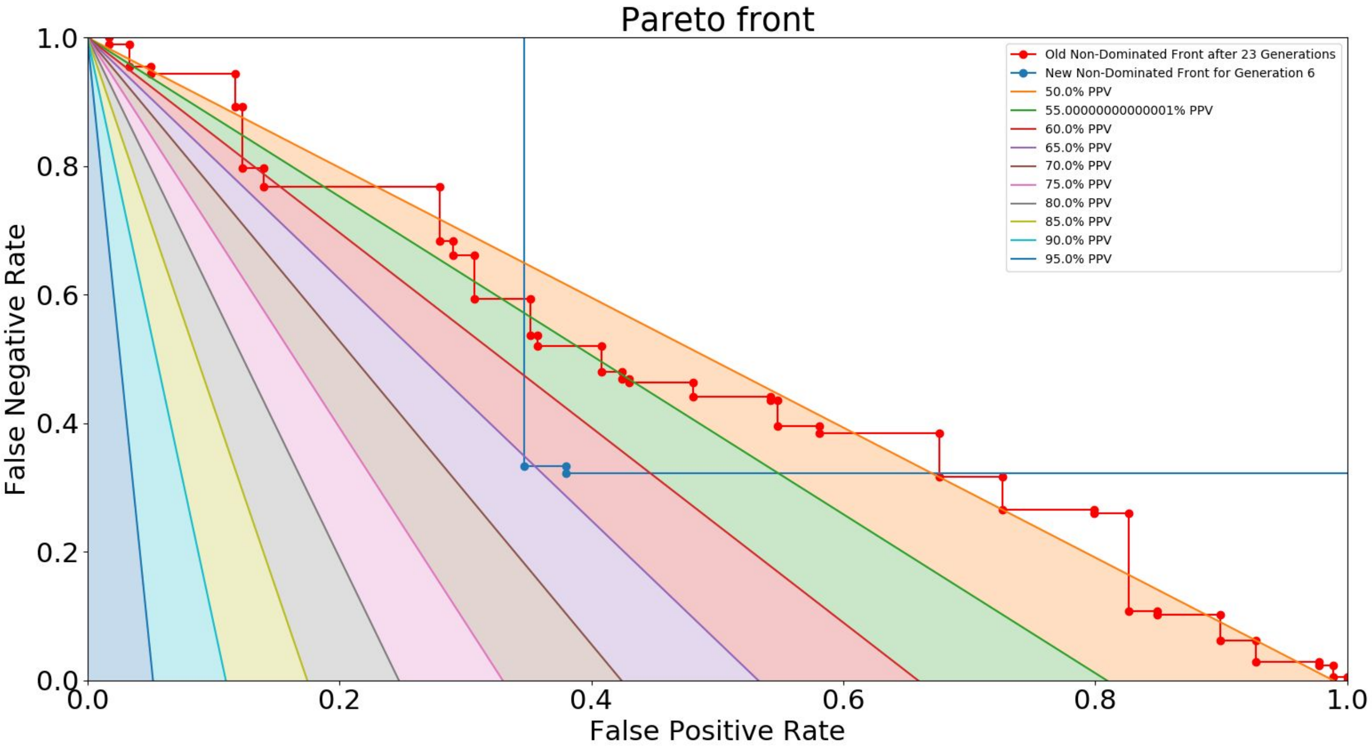
Non-negative Matrix Factorization

- Used for Feature Extraction
- NMF generates a set of topics that represent weighted sets of co-occurring terms. The discovered topics form a basis that provides an efficient representation of the original documents.
- Useful when there are many attributes, particularly when the attributes are ambiguous or are not strong predictors. By combining attributes NMF can display patterns, topics, or themes which have importance.
- P. SMaragdis (2004) “Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs“

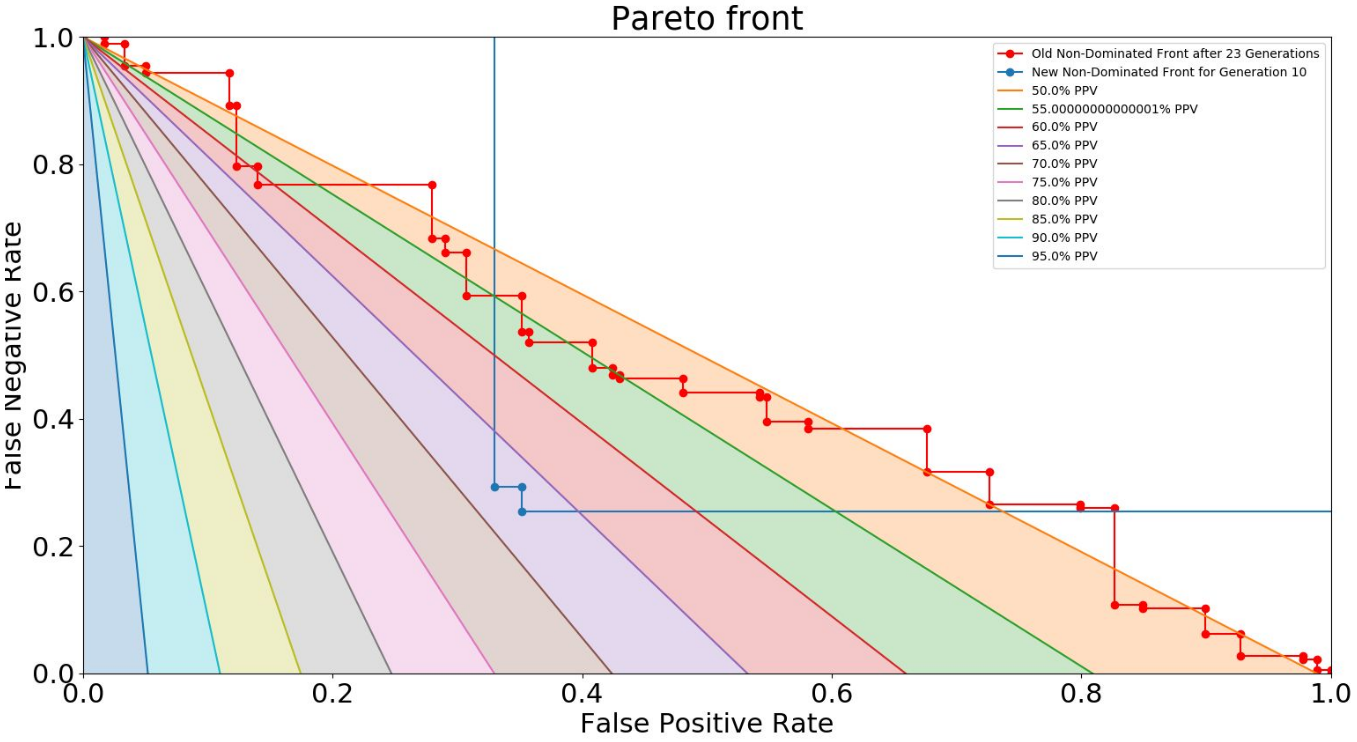
MFCC with Decision Tree Regression



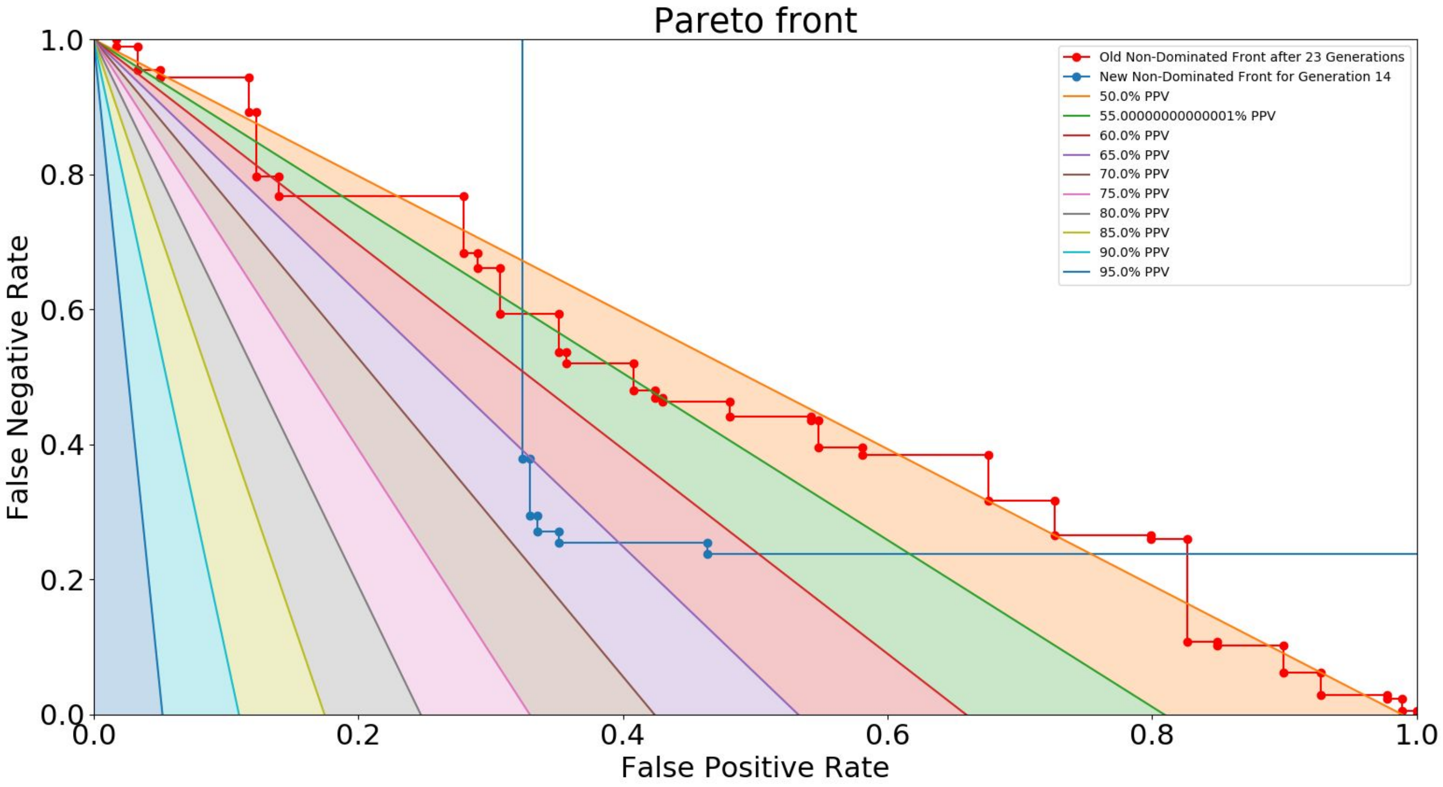
Generation 6



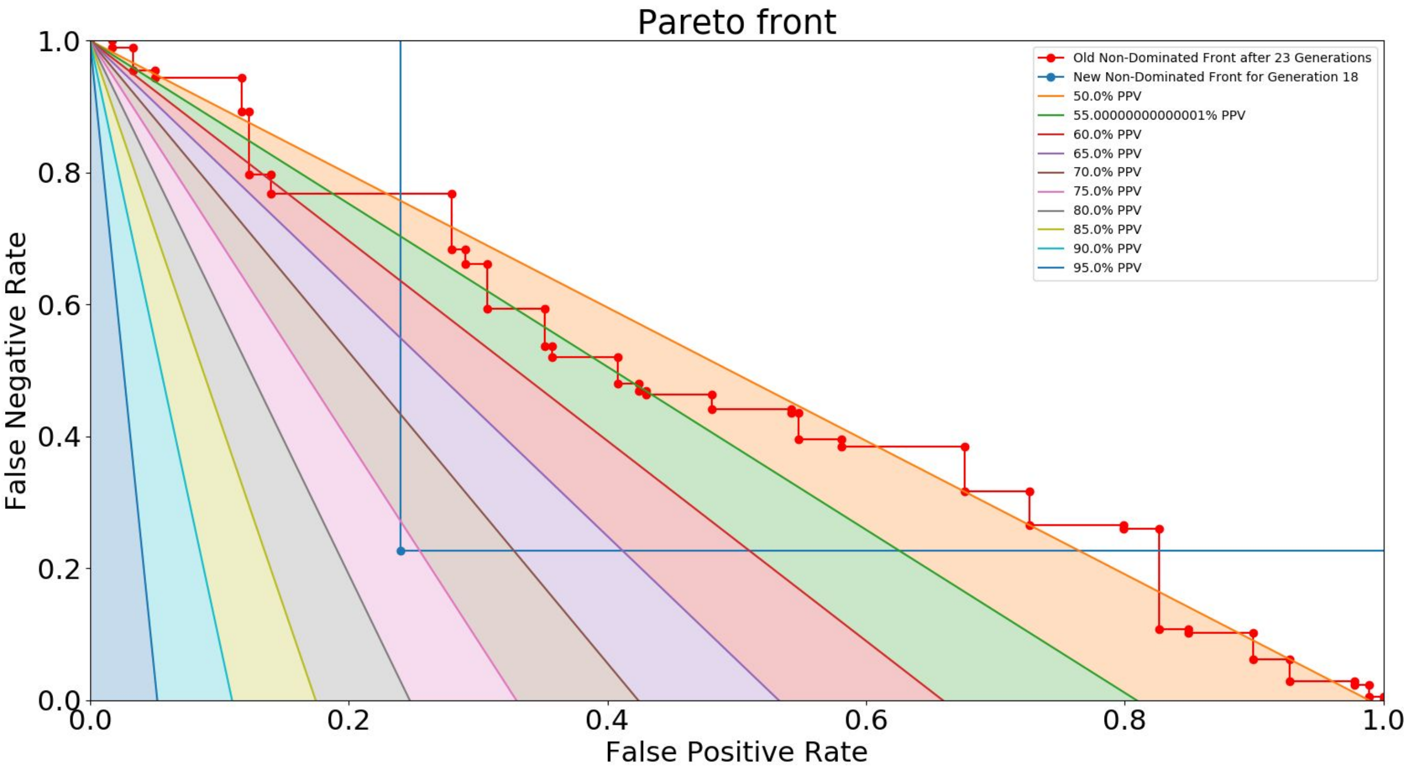
Generation 10



Generation 14



Generations 18-20



Non-Dominated Algorithm

SingleLearner(BaggedLearner(myMFCC(ARG0, passTriState(2)), learnerType('random_forest_regression', {'n_estimators': 100, 'criterion': 0})), learnerType('decision_tree_regression', {'class_weight': 0, 'n_estimators': 100, 'criterion': 0}))

Thank you!