1. The spectrogram - spectral changes to sound over time

In this lesson, we'll discuss a special case of timeseries features: the spectrogram. Spectrograms are common in timeseries analysis, and we'll cover some basics to help you apply it to your machine learning problems.

2. Fourier transforms

To begin, we'll discuss a key part of the spectrogram: the Fourier Transform. This approach summarizes a time series as a collection of fast- and slow-moving waves. The Fourier Transform (or FFT) is a way to tell us how these waves can be combined in different amounts to create our time series.

3. A Fourier Transform (FFT)

On the left is a raw audio signal, and on the right is the Fourier Transform (or FFT) of the signal. This describes, for a window of time, the presence of fast- and slow-oscillations that are present in a timeseries. The slower oscillations are on the left (closer to 0) and the faster oscillations are on the right. This is a more rich representation of our audio signal.

4. Spectrograms: combinations of windows Fourier transforms

We can calculate multiple fourier transforms in a sliding window to see how it changes over time. For each timepoint, we take a window of time around it, calculate a fourier transform for the window, then slide to the next window (similar to calculating the rolling mean). The result is a description of the fourier transform as it changes throughout the timeseries. This is called a short-time fourier transform or STFT.

5. A Spectrogram Visualized

To calculate the spectrogram, we square each value of the STFT. An example is shown here. Note how the spectral content of the sound changes over time. Because this is speech, we see interesting patterns that correspond to spoken words (e.g. vowels or consonants).

6. Calculating the STFT

We'll use librosa's stft function to calculate a spectrogram. There are many parameters in this process, but we'll focus on the size of the window that is used. We'll calculate the STFT of our audio file, then convert the output to decibels to visualize it more cleanly with specshow (which results in the visualized spectrogram).

7. Calculating the STFT with code

Here's how to compute an STFT with librosa. We first define the size of the window used for the STFT. Next, we calculate the STFT, then convert it to decibels using the amplitude_to_db function, which ensures all values are positive, real numbers. Finally, we use the specshow function, which lets us quickly visualize a spectrogram. This code was used to produce the image shown in the previous slide. Note that we're glossing over some complex details for how spectrograms are calculated, but are focusing on the essentials for the purpose of fitting models.

8. Spectral feature engineering

Each timeseries has a unique spectral pattern to it. This means we can use patterns in the spectrogram to distinguish classes from one another. For example, we can calculate the spectral centroid and bandwidth over time. These describe where most of the spectral energy lies over time.

9. Calculating spectral features

To calculate the spectral centroid and bandwidth, we again turn to librosa. We'll use the spectral_bandwidth and spectral_centroid functions to calculate these values at each moment in time for the spectrogram we've computed. These functions could also accept a raw audio signal (in which case the STFT will be performed first). This visualization code is what produced the figure on the previous slide.

10. Combining spectral and temporal features in a classifier

In this chapter, we've calculated many different kinds of auditory features from our heartbeat sounds. As a final step, we can combine each of the features mentioned before into a single input matrix for our classifier. Here we calculate the mean value of the spectral centroid and bandwidth, and stack these into a single classifier input matrix. In general, as we include more complex features into our model, we'll improve model performance.