1. Improving the features we use for classification

What we've just performed is feature engineering of our audio data. Next, we'll cover a few more features that are more unique to timeseries data.

2. The auditory envelope

We'll begin by calculating the "envelope" of each heartbeat sound. The envelope throws away information about the fine-grained changes in the signal, focusing on the general shape of the audio waveform. To do this, we'll need to calculate the audio's amplitude, then smooth it over time.

3. Smoothing over time

First, we'll remove noise in timeseries data by smoothing it with a rolling window. This means defining a window around each timepoint, calculating the mean of this window, and then repeating this for each timepoint.

4. Smoothing your data

For example, on the left we have a noisy timeseries as well as an overlay of several small windows. Each timepoint will be replaced by the mean of the window just before it. The result is a smoother signal over time which you can see on the right.

5. Calculating a rolling window statistic

Let's cover how to do this with Pandas. We first use the dot-rolling method of our dataframe, which returns an object that can be used to calculate many different statistics within each window. The window parameter tells us how many timepoints to include in each window. The larger the window, the smoother the result will be.

6. Calculating the auditory envelope

Now that we know how to smooth our data, we can calculate the auditory envelope of our signal. First, we calculate the "absolute value" of each timepoint. This is also called "rectification", because you ensure that all time points are positive. Next, we calculate a rolling mean to smooth the signal. Let's see what these transformations look like.

7. The raw signal

First, we'll take a look at the raw audio signal.

8. Rectify the signal

Next, we take the absolute value of each timepoint.

9. Smooth the signal

Finally, we smooth the rectified signal. The result is a smooth representation of how the audio energy changes over time.

10. Feature engineering the envelope

Once we've calculated the acoustic envelope, we can create better features for our classifier. Here we'll calculate several common statistics of each auditory envelope, and combine them in a way that scikit-learn can use.

11. Preparing our features for scikit-learn

We'll then stack these features together with the same function we've used before. Even though we're calculating the same statistics (avg, standard deviation, and max), they are on different features, and so have different information about the stimulus.

12. Cross validation for classification

Now that our features are defined, lets fit a classifier and see how it performs. We'll use cross-validation in order to train and test the model on different subsets of data. We can use a single

function to combine the steps of splitting data into training and validation sets, fitting the model on training data, and scoring predictions on validation data. Using "cross_val_score" will generate a list of scores across different "splits" of our data.

13. Using cross_val_score

To use it, pass an instance of a scikit-learn model as the first parameter, and the X and y data as second and third parameters. You can configure the strategy that scikit-learn uses to split the data with the CV parameter. Passing an integer will determine the number of splits that are made (and the number of scores generated).

14. Auditory features: The Tempogram

There are several more advanced features that can be calculated with timeseries data. Each attempts to detect particular patterns over time, and summarize them statistically. For example, a tempogram tells us the "tempo" of the sound at each moment. We'll show how to calculate it using a popular tool for audio analysis in Python called librosa.

15. Computing the tempogram

Here we show how librosa can be used to extract the tempogram from an audio array. This tells us the moment-by-moment tempo of the sound. We can then use this to calculate features for our classifier.