Deniz Soral
1 December 2024
Link to Self Score
Link to Colab
Link to Notes Doc

Data Finding

Finding the 'perfect' dataset took me a while this time around. I had similar criteria as the last MAs (something I can understand, varied, malleable, and somewhat interesting). I ended up running through quite a few different datasets until settling on audio samples of birds. The data was initially just pure audio, which isn't super workable, but I knew I could derive a ton of features from it by processing the audio files. The data also had varied targets (with about 113 different species of bird being sampled) which meant the audio should be varied to be able to create clusters.

Feature Derivation

This was definitely my favorite part of the process. I nearly instantly knew that I wanted to take these audio files and turn them into spectrograms, which are visualizations of the spectrum of frequencies created by sound over time.

Spectrograms are pretty simple, they are graphs of the intensity (color) of a specific frequency (y-axis) at a given time (x-axis) in an audio file. Example below:

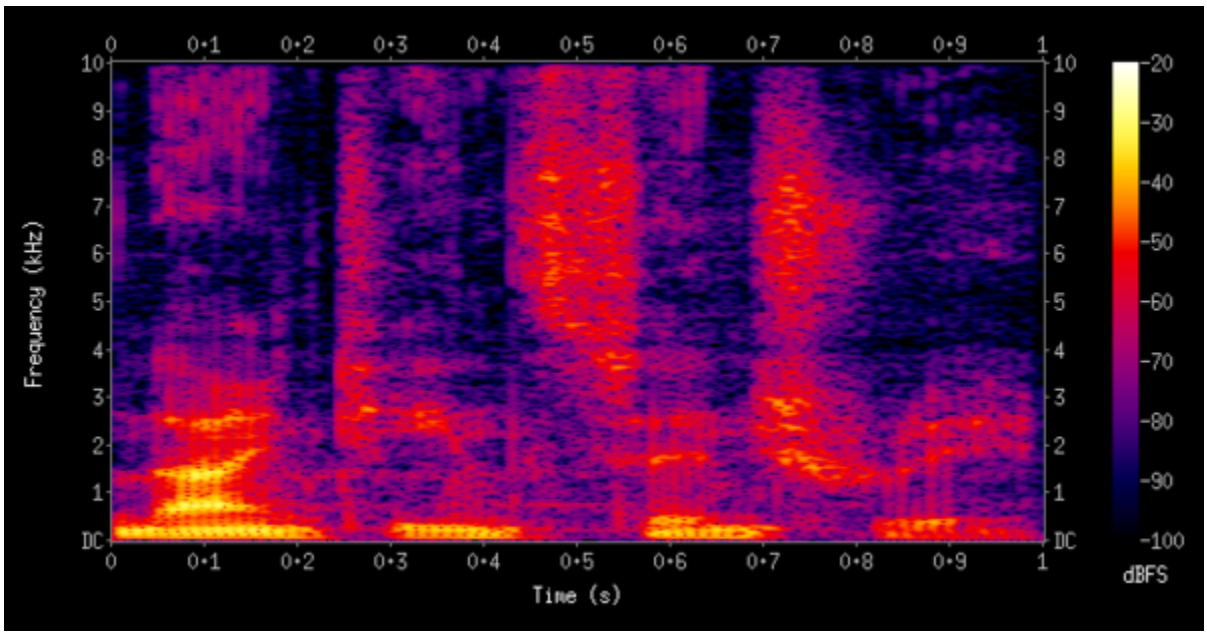


Figure: *Spectrogram of the spoken words "nineteenth century". (From Wikipedia).*

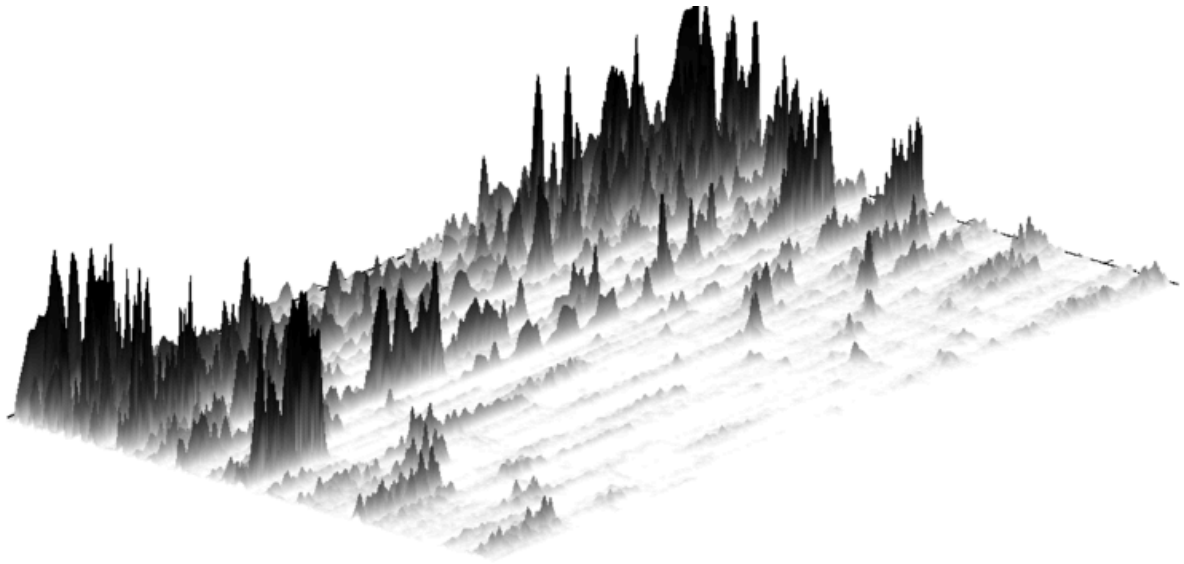


Figure: *This is probably my favorite way to think about spectrograms! It's taking each frequency and squishing it into a 2d graph with the coloring acting as its own axis. (Also from Wikipedia).*

I created 7 features (scrapped two later, so ended up only training with 5), each derived using librosa by processing audio files into a spectrogram and storing results in a separate DataFrame. Each feature should represent a different part of what makes a sound distinct and unique from other sounds (I hope, I'm sadly not a audio expert).

- **Spectral centroid** finds the center of mass of a sound (which is just the weighted mean, but it's cool to visualize it in terms of the intensity shown on a spectrogram). The spectral centroid is highly correlated to the timbre or *color* of a sound, which is what allows us to distinguish between a flute and violin playing the same notes, making it useful for separating widely different bird sounds (such as rattling/rumbling versus high pitched chirps). Typically, the spectral centroid is a line, so I took the mean of all datapoints on the line to finalize the feature. Here's an example of a spectral centroid taken from librosa's documentation:

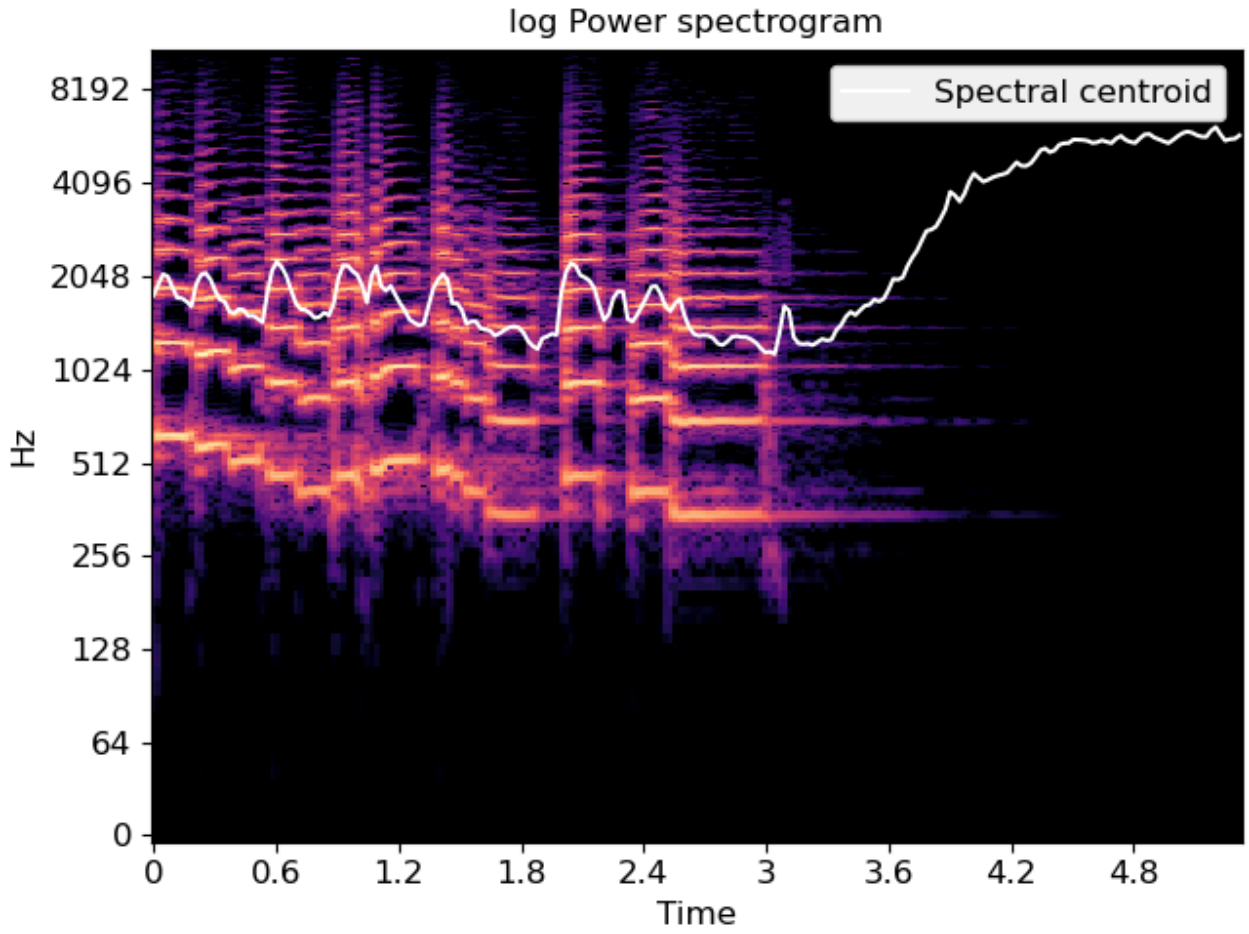


Figure: *Note how the centroid trails up at the end because of the silence. As far as I can tell (and after listening to a few files) the data I'm working with doesn't seem to have this trailing silence, which should avoid this issue.*

- **Spectral bandwidth** is based off of a spectral centroid. It finds the *weighted* standard deviation of frequencies around the spectral centroid (note that all of these features are weighted, which means they're probably good for making these broad generalizations about a sound). The stdev isn't taken around a single point though, it's taken point-by-point based on the original line form of a spectral centroid, which means I took the mean of all the stdevs. This feature is good at distinguishing consistency of vocalization, e.g. a bird with one super annoying note that wakes you up at 5 a.m. or a bird that produces varied pitches to create a *complex* song). Here's another example, using the same spectrogram as above (thanks librosa!):

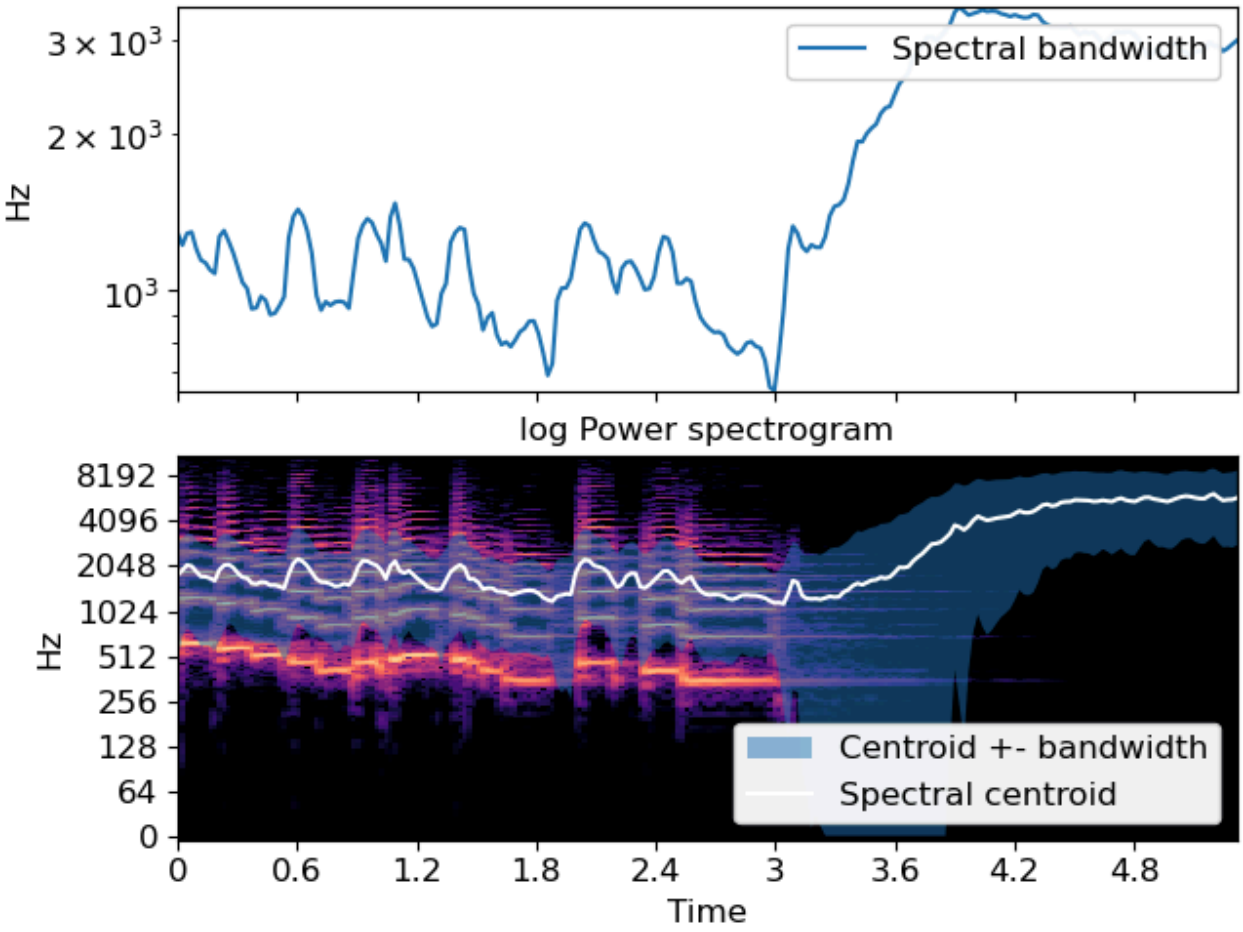


Figure: *Note how there's quite a bit of variation in this compared to the centroid, which probably causes the mean to lose some of the data's nuance. Taking the mean is probably the only way to stop the model from skewing in favor of this one specific feature though, so I have to live with it.*

- **Zero crossing rate** is how often the frequency of a sound crosses between positive and negative (literally crossing the 0 axis) over a given time (in this case, the whole audio file). So high frequency sounds will have high ZCRs and low frequency sounds will have low ZCRs—basically measuring oscillation. This is useful in determining if abrupt rattling or clicking noises are being made, or if they're smoother, tonal noises. I will note that this is less about spectrograms, and more so about the general frequency/amplitude of a sound.
- **Spectral rolloff** is probably the weirdest one, but also quite helpful. It essentially finds the area in which most *spectral energy* (intensity) is concentrated. It's calculated by summing the lowest frequency sounds then moving upwards in frequency until 85% (can be different, but typically 85%) of all intensity has been summed/used. Wherever you stop is the rolloff frequency. Again, rolloff is a line so I took the mean to simplify it into a feature. For bird sounds, this is helpful to not only detect lower vs higher frequencies (or

at least somewhat, this feature makes a very broad claim about frequency), distinguish tonal versus noisy/broadband birds, but also remove outliers or 'noise' (likely produced by unwanted environmental sounds) in audio files by cutting them off—thus improving the quality of the model.

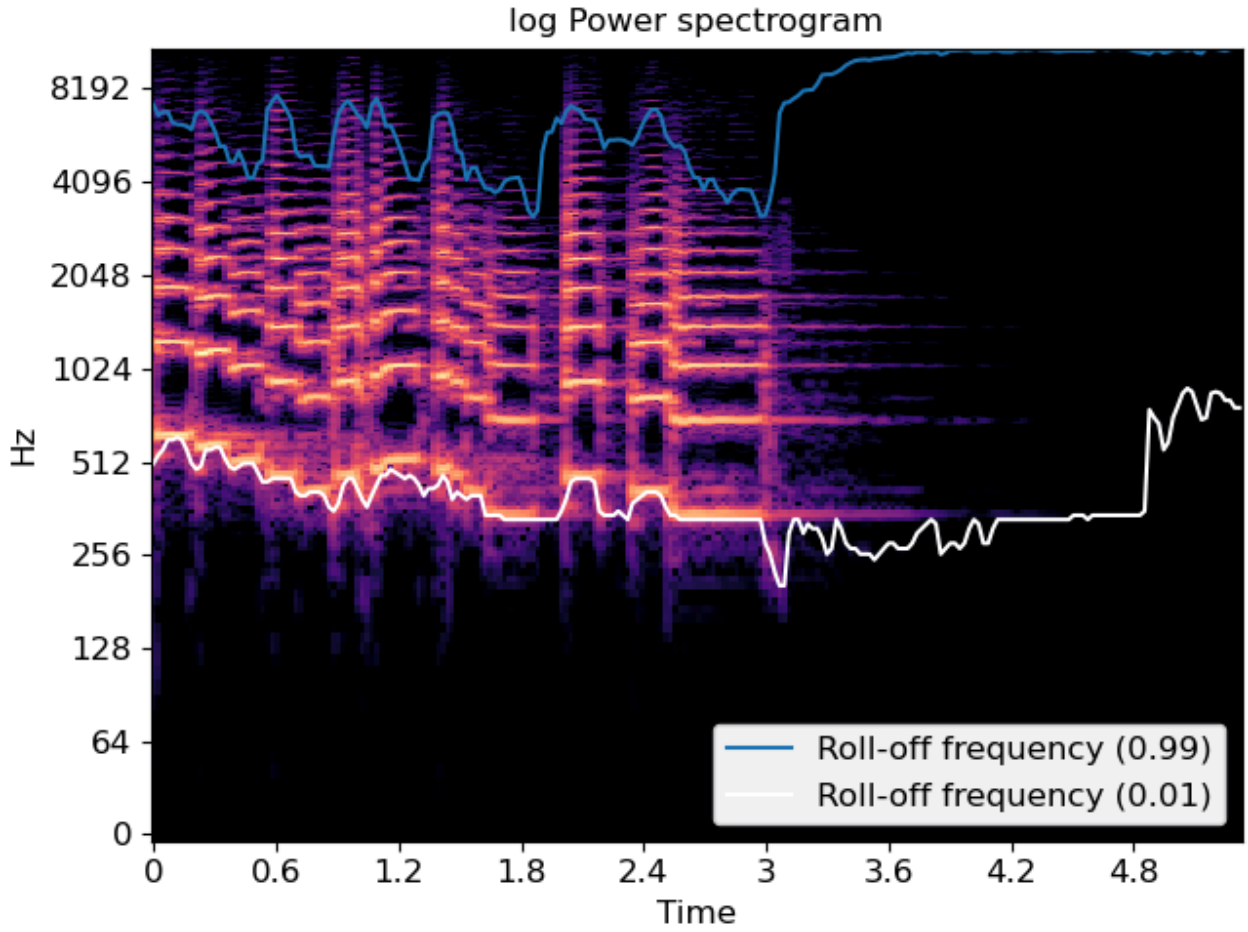


Figure: *Note that the parameters they used here were 99% and 1%, so the cutoff is pretty marginal.*

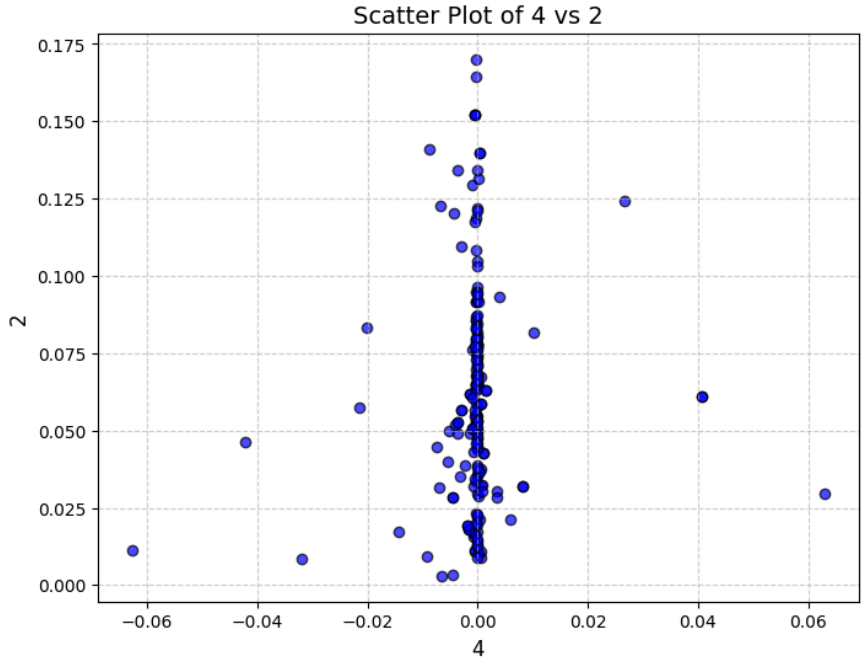
- **Fundamental frequency** is essentially the "pitch" of a sound—or the first harmonic of a waveform (essentially the lowest, and loudest, proper note). It's the main tone we hear in a given sound, so it ignores any overtones that are also produced. This is probably the most straightforward way to distinguish between bird class; it's just a matter of saying "oh wow this bird sounds like an A while this one sounds like a C, must be different species!"

I did scrap the **Mel-Frequency Cepstral Coefficient** for a couple of reasons. While it is probably one of the more useful (and commonly used) it ultimately didn't fit a clustering model. What it does do is super cool though, I didn't bother to understand it fully, but here's what I know: it separates a file into many chunks of about 20 frames each, then does some math, applies

filters to them that separate and amplify frequencies to show how human ears perceive the sound and how that perception *changes over time*. The key bit is that its purpose is to show each frame individually and show how it changes, meaning there's no good way for me to condense it into one feature (as it outputs as a bunch of vectors) without losing the integrity/purpose of the feature. And if I were to use it as multiple features, it would skew the model too far towards this one statistic, forgoing the other features.

I also scrapped the **Harmonic ratio**, which is derived by dividing the amount of harmonic energy by the total energy in an audio file (again, remember that energy is similar to intensity). Harmonic energies are just frequencies that are tonal and similar to the fundamental frequencies (this is confusing terminology, it's just sounds that have a proper frequency: singing in a choir would be harmonic, whereas me banging on pans is not harmonic). This can be used to distinguish 'noisy' bird calls from song-like bird calls (similar to a couple of the other features here, but this is especially good as catching irregularities in sound).

- Quick sidebar: it would have been interesting to use this feature to actually eliminate datapoints. If a audio file has a low HR it likely means it has a ton of background/environmental noise, which probably makes it bad for the model as the actual bird sound is obscured. This is actually why HR is often used in commercial background audio suppressors! I didn't do this (because I'm lazy and didn't realize this) but, in hindsight, it would have been cool and made for a more accurate model.
- Also here's why I scrapped it, I have no clue why it turned out like this, but the other features should compensate for differentiating noisy vs harmonic (every single feature vs HR looked like this):



While these features do have some things in common (such as differentiating between high and low, harmonic vs noisy) I believe that they cover enough of what I could have analyzed and are each unique in some aspects that my model should create a (kinda) cohesive representation of each bird call while not being overly complicated.

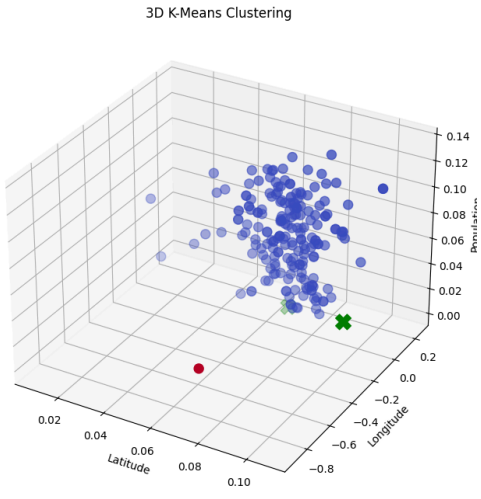
Feature Engineering (actual ML now! no more audio yap)

I just want to preface this all by saying that I lost a lot of data from the original in the final dataset I worked with. This is because I was only able to process 200 files before my colab crashed (and I can't be bothered to run it natively or workaround this constraint in colab). I select the data from the full dataset with a random state of 42 (which makes it reproducible). I'm not quite sure if this is biased towards one species over another (especially considering that the model isn't even in terms of how much each species is represented either). The randomization should hopefully avoid any extra bias not already present in the dataset though.

After this, I normalized my data using numpy. It was super super skewed before this, with some features ranging in the thousands from others in the 0.0000001s. This should hopefully have stopped clusters from being solely formed on these large features and instead forming on all features equally.

I then removed outliers from my dataset in two ways. First, I removed any data points that were 2 standard deviations away from their assigned centroid, then I removed any outliers that got their own clusters (datapoints in clusters with two or fewer points were removed). In this scenario, I believe that the outliers weren't somehow crazy birds with insanely different calls, but rather poorly gathered bird calls that were polluted by other noises, such as machinery, high winds, or other animals. About 5-7 data points were removed in each run of the model. Here are some before and afters of outlier removal:

With Outliers



Without Outliers

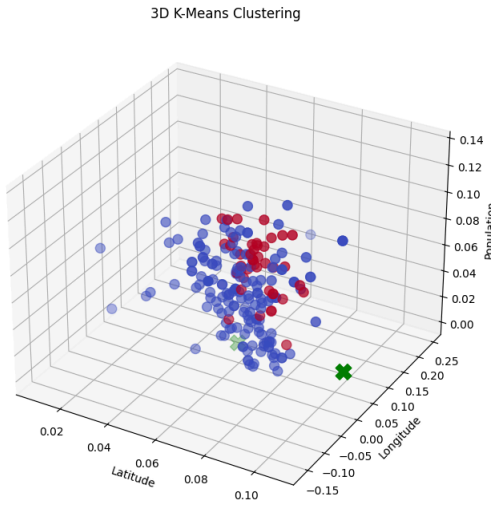


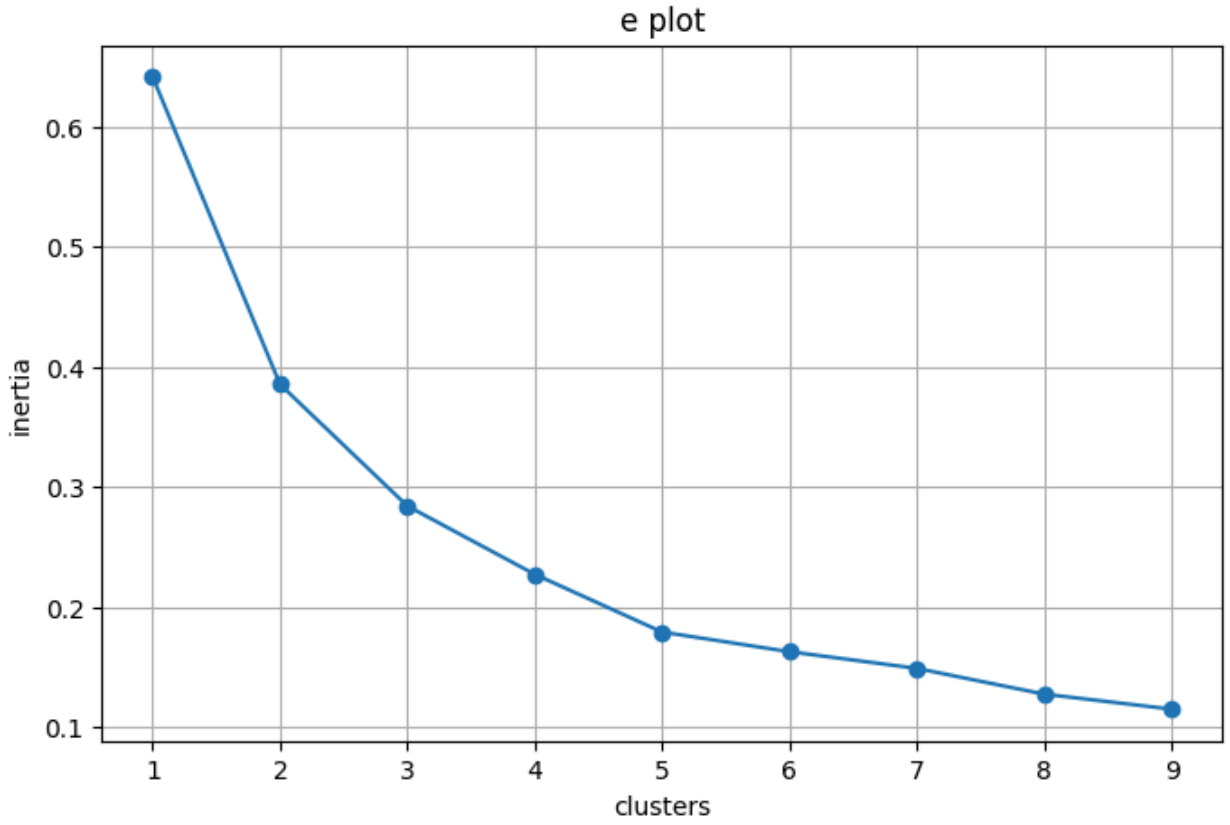
Figure: *Note that there are only two clusters here for now, I'll talk about this later but I'm still not fully sure if I should have used 2 or 3. Also note that these images were generated before I dropped the harmonic ratio, so they may not be fully accurate—but the point still stands.*

Brief Rationale for Model Type

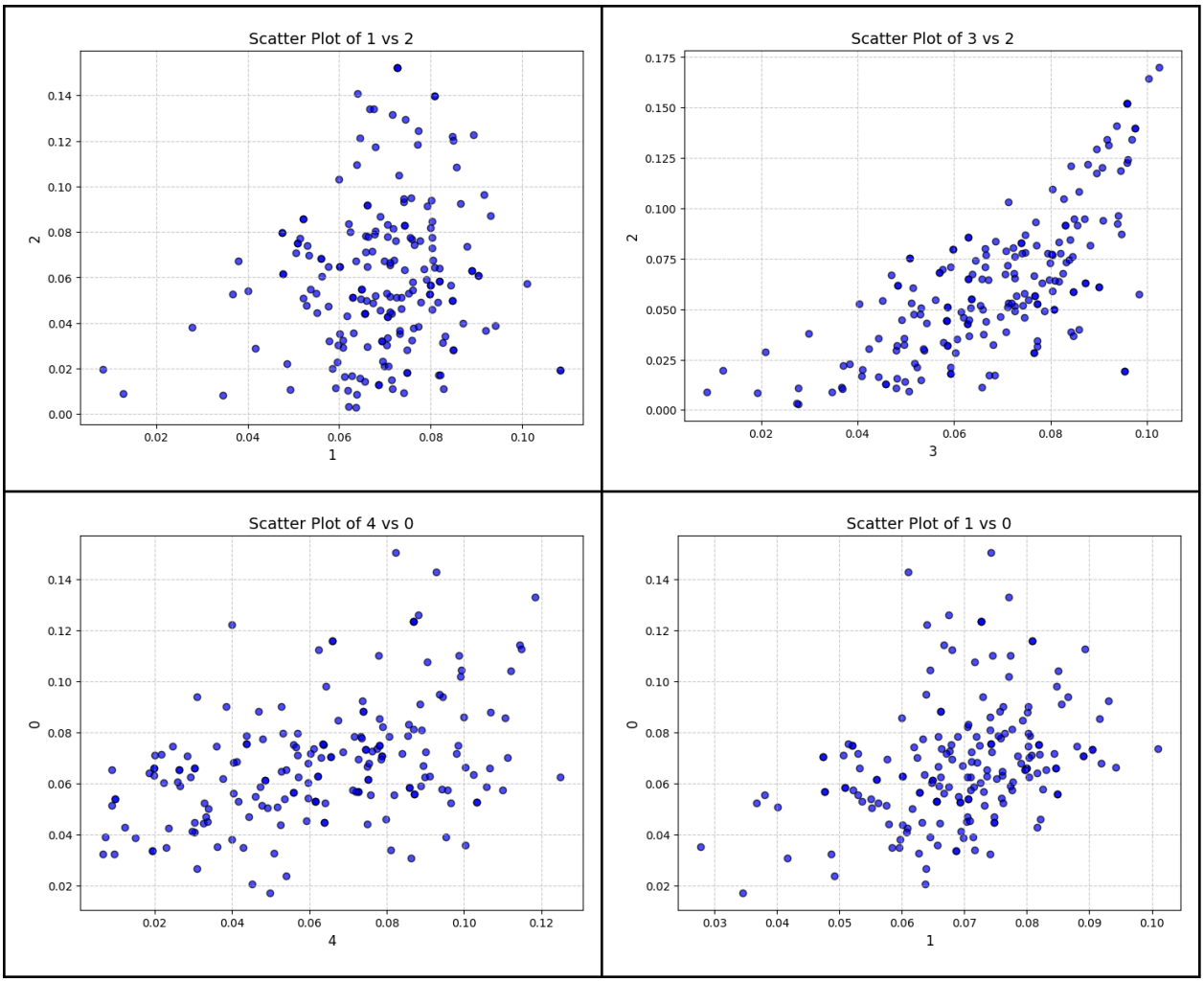
I probably could have used db-scan here, but I think k-means was slightly more suitable because the data is pretty close together (which I believe k-means is better at differentiating between). Not a huge difference though, I just like k-means)

k-Means Parameters

I determined my k using the elbow-plot method (I forget what it's called). Graphing the inertia (how much the centroids move around) versus the amount of centroids.



3 looks like the proper k to pick here (because that's where you preserve efficiency without overcomplicating the model). But I ran with two for a while (because I hadn't dropped HR yet, which made the graph look disgustingly different). After I checked out the 2d feature vs feature graphs 3 definitely seemed the correct option. Here are some of those 2d graphs below (it would be hard to draw 2 or 4 logical clusters for most of these):



I also used k-means++, which, to my understanding, just runs the model a bunch of times (I chose 10) and finds what the ideal placement for centroids is based on the consistency between models.

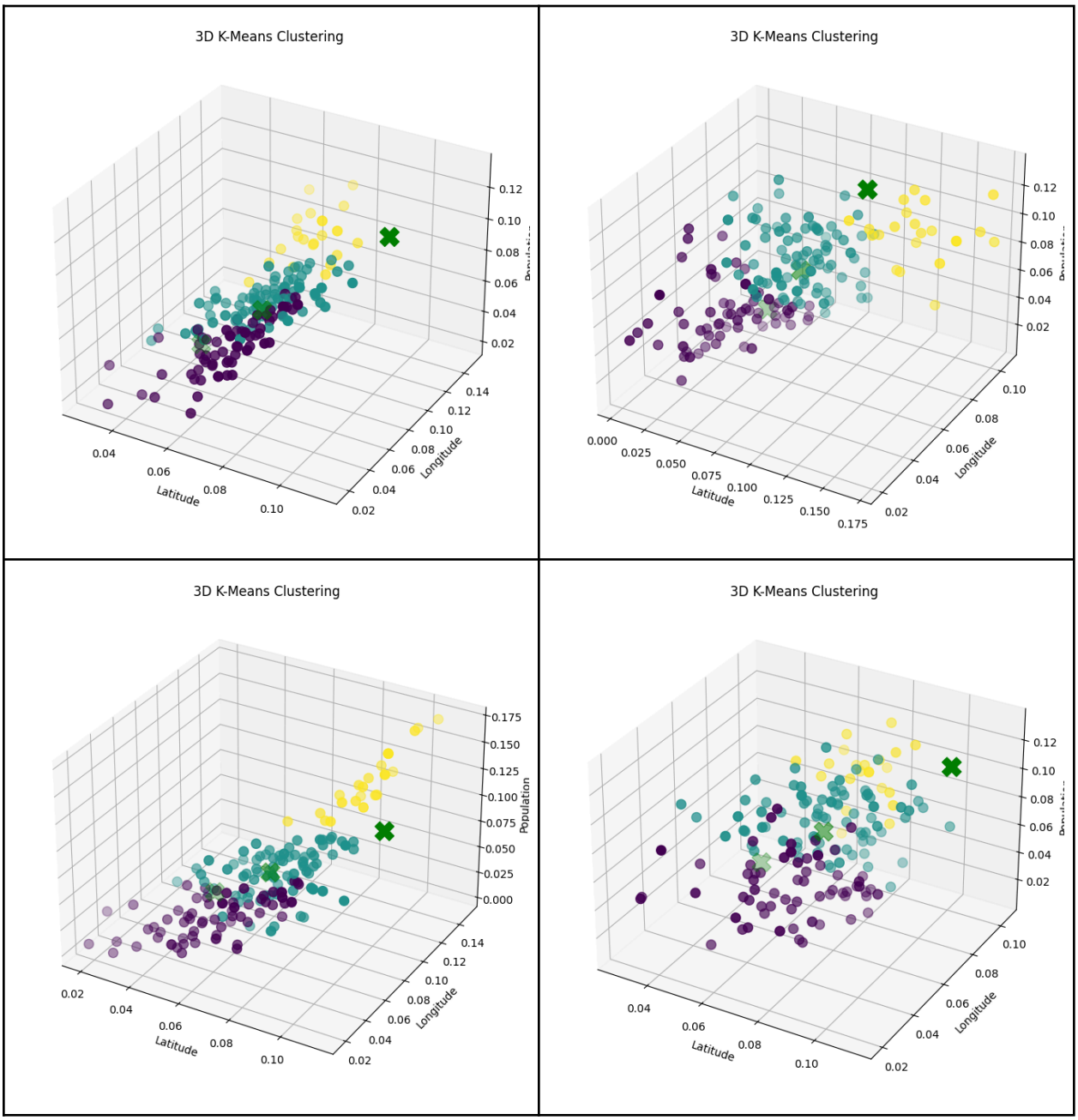
Validation

Regarding ground truth, I don't think it's necessarily useful for this model. While I did have one, there are just so many bird species that I have no prior information on their similarities that it's essentially impossible for me to tell if clusters were made in the right spot. Sure, I can say that it's cool how it grouped all of the andean finches together, but I believe the point of the model is to make interspecies connections (as opposed to intraspeices connections, because it's already pretty obvious that two roseate spoonbills have the same call). I'm also lazy and don't have time!

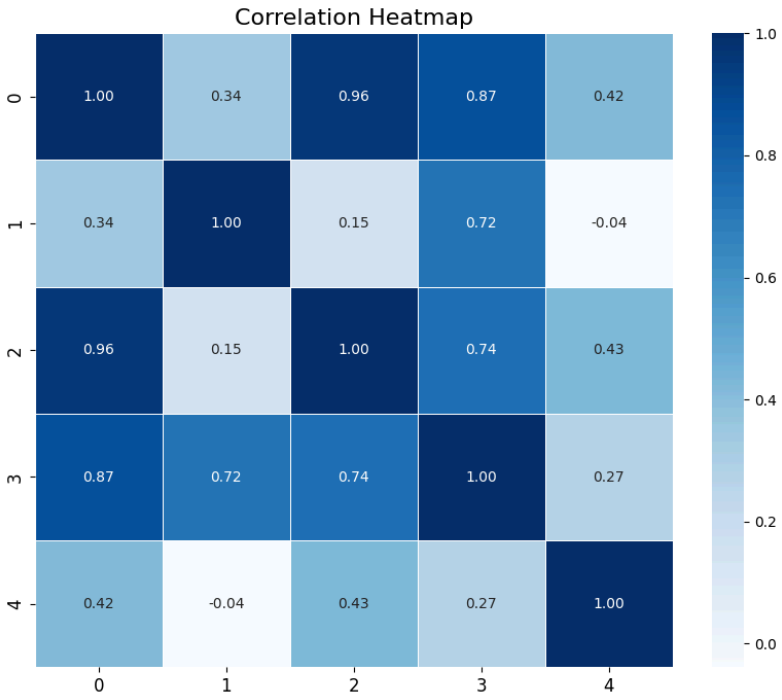
Using k-means++ already ensures that my model isn't getting my clusters horribly wrong and guarantees a degree of consistency—which provides solid validation (I also checked, and the centroid locations are consistent). Furthermore, all the engiennering I've done up until now

(elbow plot, normalization, ensuring varied features) adds to the validity of the general claims my model is making.

I also did a simple visual test—do the clusters look solid and proper. I say that they look pretty good, here are a few examples:



Also, to just fully make sure that my features are somewhat unique, I created a correlation map between them. Ideally, everything has low correlation (not super duper low, but low enough). This would mean that my features are super unique and not repetitive, meaning no bias is made towards a specific characteristic of a sound.



Yeah, this isn't perfect. This is probably due to the fact that my features are similar: measuring noisiness vs harmony, tonal vs broadband, oscillations, low vs high frequency, are all quite similar in reality. This does mean that I probably could have removed some features (such as 3, which is the spectral rolloff) that have high correlations with all other features. In the end though, I think that it doesn't have a huge impact on the model's bias/accuracy, as it still seems to be creating distinct clusters, still has low correlations in other areas, and the correlations isn't high enough (except for the 0.96 and 0.87) to be substantially redundant.

Ethics

This model's goal is to make broad generalizations (as most ML models do) to group bird sounds and species based on those sounds—not to identify specific species or calls, as the model is being fed many many species and only making 3 groups. This is for a few reasons.

First, most features are the mean of many datapoints that form a line, making them lose some nuance in sacrifice of usability.

Second, as discussed above, the high correlation between some variables means that there's some bias towards specific features of a call, so birds that don't have that element in their calls will likely not fit into this model. But this shouldn't make a huge difference when trying to broadly categorize calls—as such birds will likely fall into the same cluster—it just means that delineating between those birds won't work.

Third, like with most clustering models though, there's a pretty big gray zone in the center of all the datapoints where clusters grow super close to each other (especially because my features are averages). This does mean that the model is poor at differentiating sounds that fall into these areas (typical bird calls without special flares etc.).

Very quickly, much of what was said in the validation section can also be applied to ethics. Everything done to improve the model and reduce bias also factors towards the ability of the model to make *generalizations*.

Dropping the wide majority of the bird dataset probably also affects the real-world applicability of the model. Furthermore, because some birds were overrepresented in the original dataset, my randomization likely overrepresented a couple species (which again, probably doesn't match the real world), meaning that the model may tend to group more birds into these overrepresented categories.

Some Future Steps

I would love to have played around with and explored other audio processing techniques to derive more features (which are hopefully also unique!). I found it super interesting, and really enjoyed learning about these weird things I can do to data (whereas in my past projects, gathering and deriving data was essentially non-existent, but I suppose derivation matters a lot more in unsupervised learning).

I would also have liked to test the model with different bird calls (post-validation exploration!) that I know are classified as similar/different and tested its decision making. Sadly, I didn't have much time for this, and in my limited time I couldn't find data for this.

Doing more validation (probably with some fancy mathematical coefficients) would have been cool as well, besides just looking at elbow plots and data correlation.