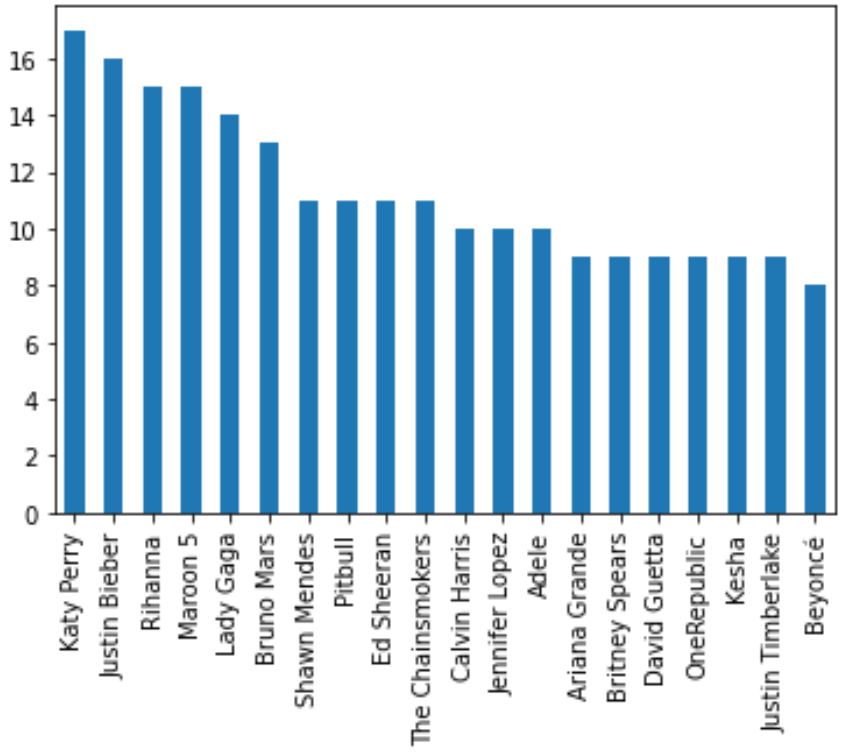
Hello! For my final project, I decided to evaluate a dataset with the top Spotify songs from 2010-2019 (pretty much the decade!). I decided to use this because I've been more on a Spotify kick lately (who was your top artist in your Spotify wrapped? – seriously, if this is Heman reading this, message me on Discord, haha!). I've had to get a lot of help on this because I am super picky about how things look and making sure everything has a place and purpose in the project—and the things I wanted to do sometimes required some functions and objects I didn't already know. Anyway, I worked really hard on this and found some new functions, objects (value_counts () is a really cool one to use!), and attributes to use to help make my visualizations look and be what I was looking for! Here goes:

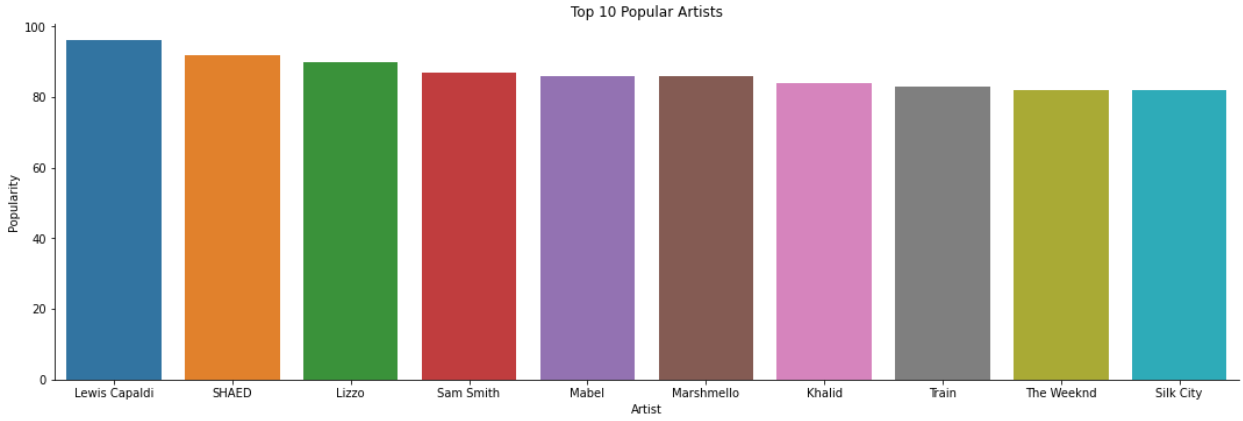
I started my project with the usual: importing different things that I might need to analyze my data and reading my dataset into my project. I had to write something I didn't quite understand here (the encoding part), but a CA told me to do it because my dataset is large. Nothing special here to talk about.

What I did next was that I wanted to take the number of times that an artist was listed on my csv and put them on a bar chart, which was much easier to do once a friend told me about value_counts (it literally solved all my problems):



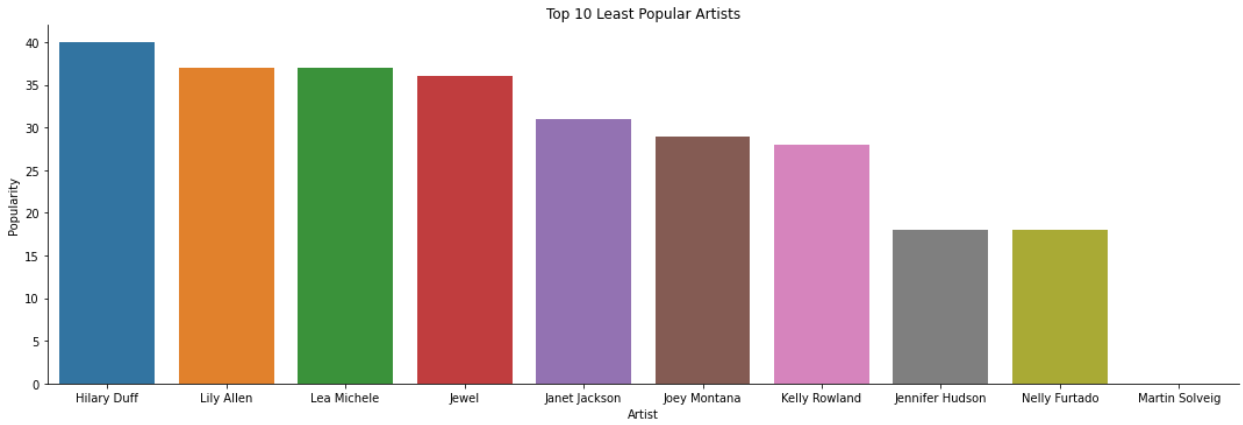
As you can see here, Katy Perry appeared the most on the hits throughout the decade, with Bieber at a close second. Do you see any artists you like in here?

That was super interesting to look at. However, is Katy Perry the most "popular" **artist** from the decade? Turns out that popularity was a column given numerically in my dataset, and so that is what I wanted to work with next:



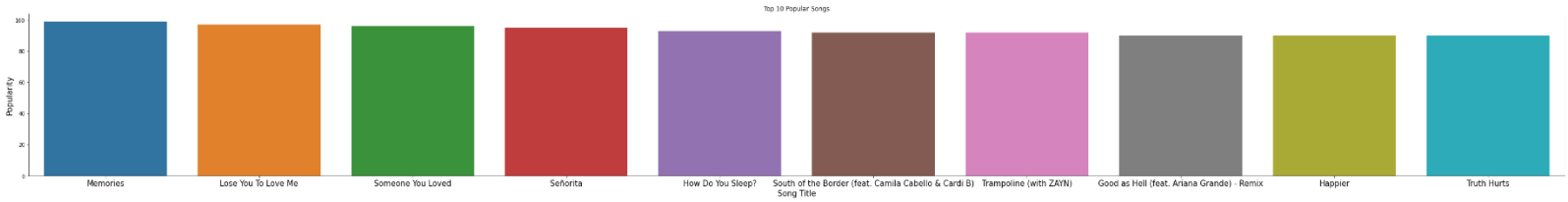
This is different than seeing what artists have the "most" hits; this will be looking at the artists that had hits with the most popularity through 2010-2019. I basically took the artists' names and their popularity and grouped by artist, and sorted by popularity, and then listed them in descending order listing the first 10. Later, I discovered catplot, which was basically able to create my bar plot with ease (I believe aspect changes the size). I then was able to figure out how to label graphs to make them look a little more cohesive. Looks like Lewis Capaldi is at the top unless I'm mistaken.

My next one is similar, and it is simply the top 10 least popular artist of the artists with hits (so not the least popular artist in general). Here's the result:



I did this one pretty much the same way, but I took the tail instead. This time, it looks like the least popular artist would be coming from right to left and be Martin Solveig.

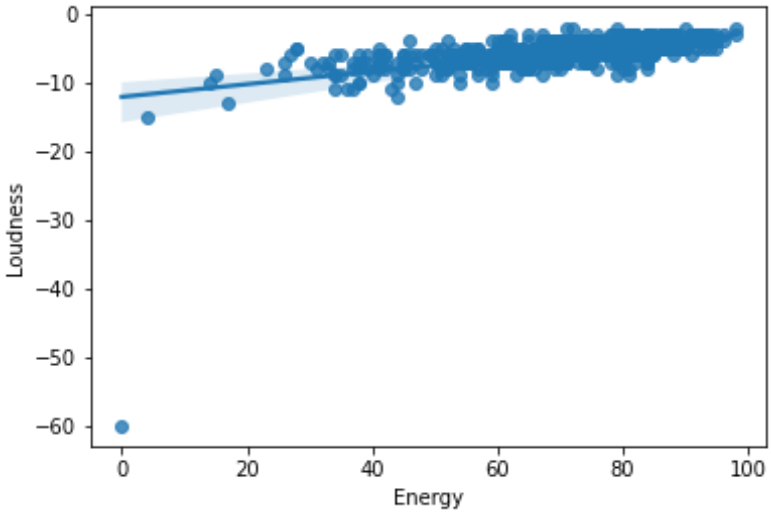
Next, I decided to find the top 10 popular **songs** from 2010-2019. I found the popularity of each song and was able to get the highest numbers in popularity and then groupby title to create this next bar plot (I'm actually a fan of bar plots):



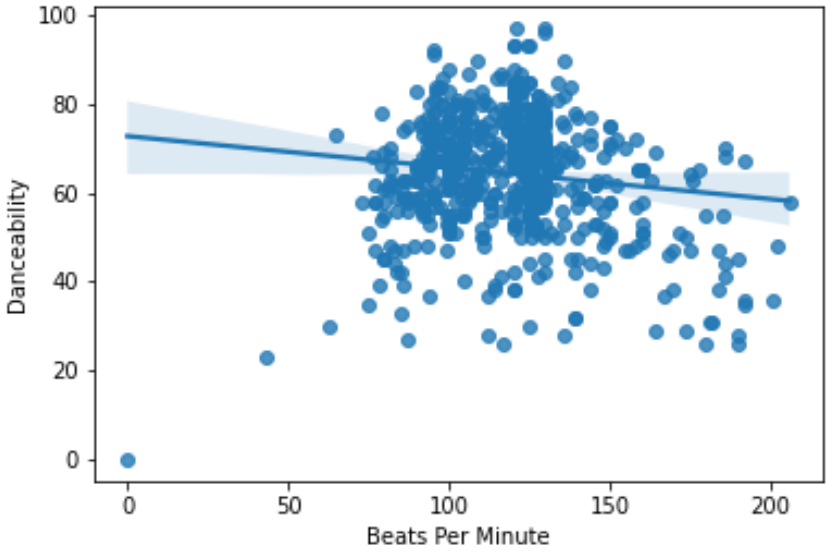
Might have to zoom in to read this one. For some reason, no matter what font size and aspect I used, it is very small to read, and I just couldn't make that work (maybe it's just wide). This is like how I did the most and least popular artist but changed to "title". I was curious about what it would say, and it looks like "Memories" by Maroon5 is the winner here with Lizzo picking up at the end of the top 10 list.

Next, I did an evaluation of one of my favorite artists, The Weeknd. I pulled from the artist column "The Weeknd" to show everything about them I could find in this database. You'll have to look yourself this time, but his genre is Canadian contemporary R&B, his beats per minute were about average for this dataset (I have df describe slightly below), and more. I also called to collect how many times he popped in the database, which looks like 5 times in total. I also described df down below as I wanted to compare some of the information from what I got.

Next, something pretty cool: scatter plots. I thought I would have some variety to break off from the sheer amount of bar plots. In my dataset, I also have columns titled around energy, loudness, beats per minute, danceability, valence, acoustics, and more that are available to use and compare. For my first plot, I decided to compare Loudness (dB) and Energy (nrgy) of the song to see how they correlate.

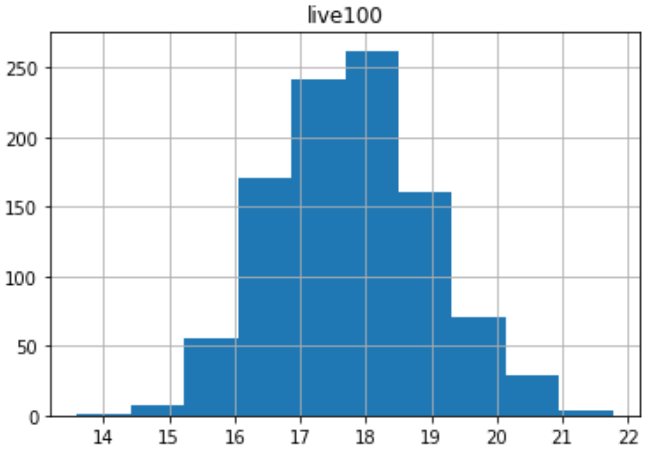


It appears that loudness and energy are positively correlated (more energy leads to more loudness). That makes sense. Next, I compared Beats Per Minute (bpm) and Danceability (dnce) to see what I get here, and here's the result:



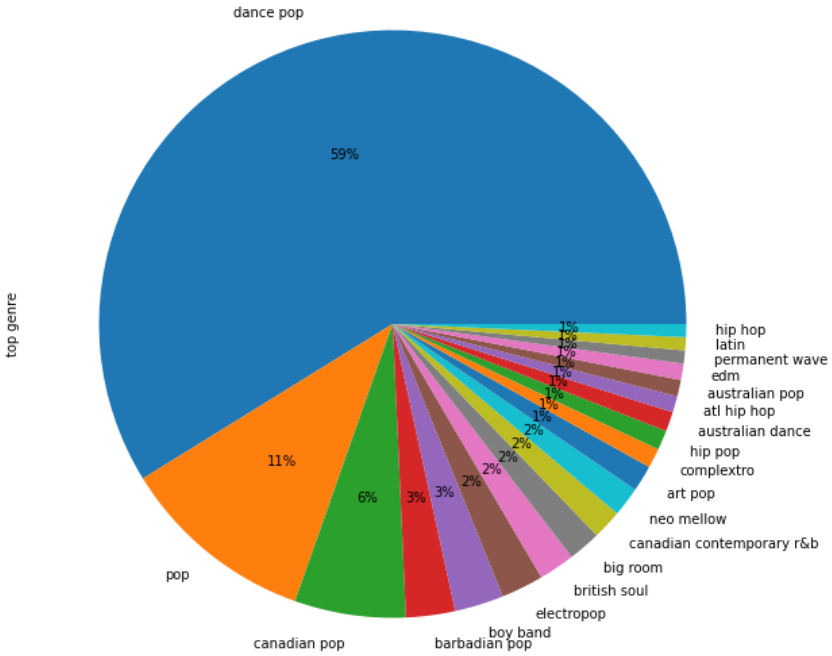
Looks a tad different than what I imagined it would. I think this would technically be a negative correlation, and there's probably a sweeter spot around 100 for bpm and in between 60-80 for danceability to most songs. I think the r would be close to 0 in this case.

I also decided to create a simulation of one of my columns, liveliness or "live". I just simply wanted to see how it would work out, so here it is:

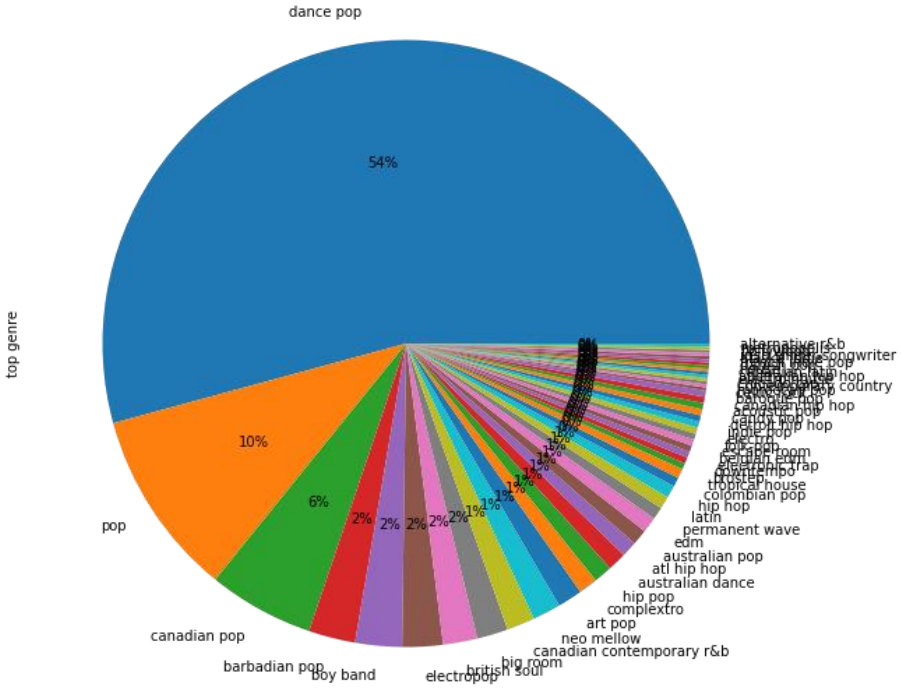


Looks like normal distribution for liveness. I started by creating a function to simulate the average of any column and number I put in, and then I simulated liveness 1000 times.

Next, I wanted to create a pie chart (yes, I know we haven't done that before) to show how genres popped up in our dataset and I wanted it converted to percentages so that it would be a good visualization of the genres in the dataset. Admittedly, Google helped me out on this one (autopct? Figsize? Who knew?) and showed me multiple different ways this could go down. Here's what I did:



It's clear that "dance pop" is the most popular genre from the decade. There are some others to look at though! I decided to take the top 20 because it looked like this otherwise:



Yeah, that's not very readable. But apparently it equals 100%, so it's the most correct one. Anyway, for the first one, I took the top genre's value counts, made it into a pie chart, took the top 20, and then automatically made the resulting number into a percentage (autopct).

Overall, I think that's that for the project! I tinkered a bit in doing simulations, z-tests, and maybe even some regression but I felt that my project a.) got very long and b.) didn't necessarily need those tests to show that I learned adequately in STAT107! I think that while I used some new objects and attributes in this project (for example: value_counts()), I think that it's important that I show that I learned *how* to use it than just followed directions online when you search *value_counts() python*, which I believe I did! I really enjoyed this class and to whoever is reading this, have a great break!