Spotify and You

CS 439 - Introduction to Data Visualization Final Project

Nick Schiermeier

Presentation of the Data

For my project, I thought it would be interesting to visualize data from Spotify, both data that could be found in general about Spotify as well as my own streaming history. For the general Spotify data, I looked online and downloaded a few different datasets on Kaggle before deciding on one to use for my project. I found the dataset I downloaded interesting because it had a lot of different attributes that I was able to play with and manipulate in order to create unique visualizations. I thought about asking questions such as how are the attributes correlated, how can they be represented in different ways, and what different types of visualization techniques could I use to show these?

For my personal data, I was able to get that through my Spotify account's user settings, and then they emailed it to me about a week later. I thought this data would be interesting to present because it is personal to me, so I wanted to see what I could conclude about my listening tastes. General questions I had were about my listening tastes and seeing how they had changed over time, or what and when I listened to most.

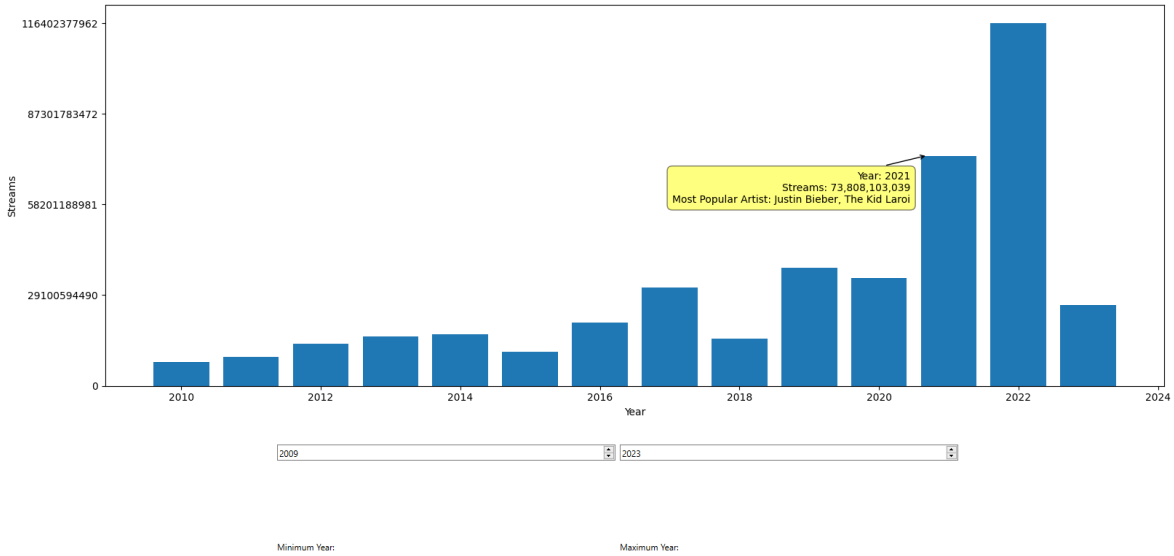
Goals and Objectives

My original goals were to see how my music taste compared to the general public, or how my music taste changed over the last 5 years. I also wanted to see how my listening changed over Covid, or if I listened more when I drive or do homework, etc. However, all of these were contingent on me getting my entire listening history in reasonable time, and if I didn't that it would be in the same format as the 1-year history that would be sent to me sooner. Unfortunately, when I got the full data, it was not in the same format as the 1-year data, so I had to change these goals somewhat, as I didn't have time to reformat the project to accept this new type of file, as well as my interest and goals just changing in general. So my new goals ended up being just visualizing spotify data and visualizing the user's data over a given year, but without any sort of interaction.

Design

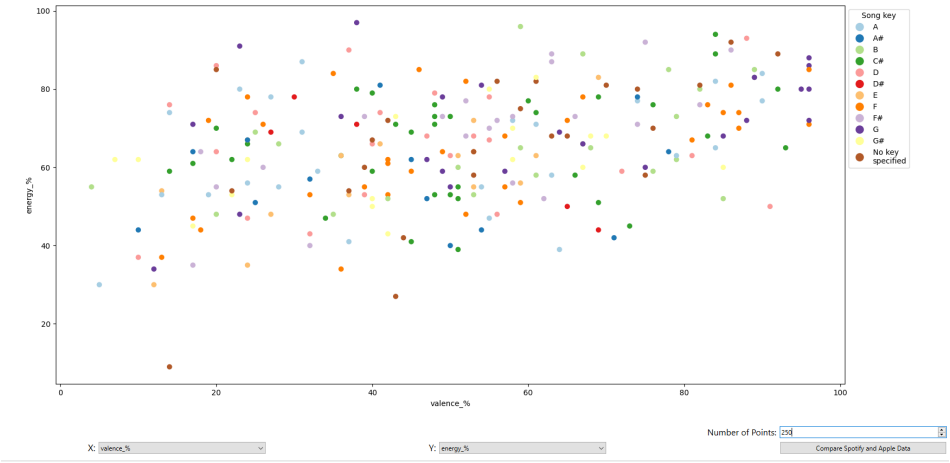
My main design decision was to create 5 tabs with each tab being a different type of visualization (though due to time constraints this only ended up being 4 tabs).

Tab 1 — Song History

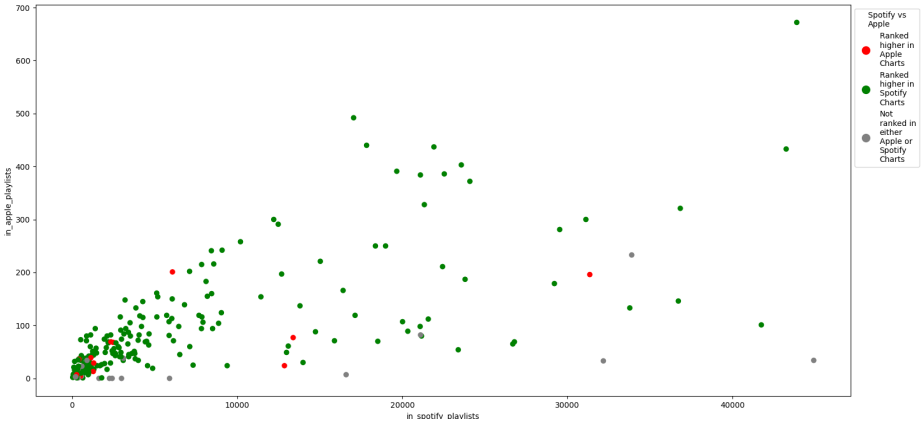


This was my first tab. It visualized the cumulative number of streams for songs by year. I decided to make this a bar chart because it would clearly show an increase or decrease of streams, as well as being able to isolate years easily. One extra aspect I took was including a minimum and maximum year filter, so that you would be able to center your view on a very specific time frame you could be interested in to see how music changed there (eg. the ‘90s). This tab related to my goals of visualizing general spotify data, because I could see how much streams increased in the last decade or so, as well as seeing which artist has the most number of streams for a song they released in a given year, and determine from that the most “popular” artist of that year.

Tab 2 — Scatter Plot



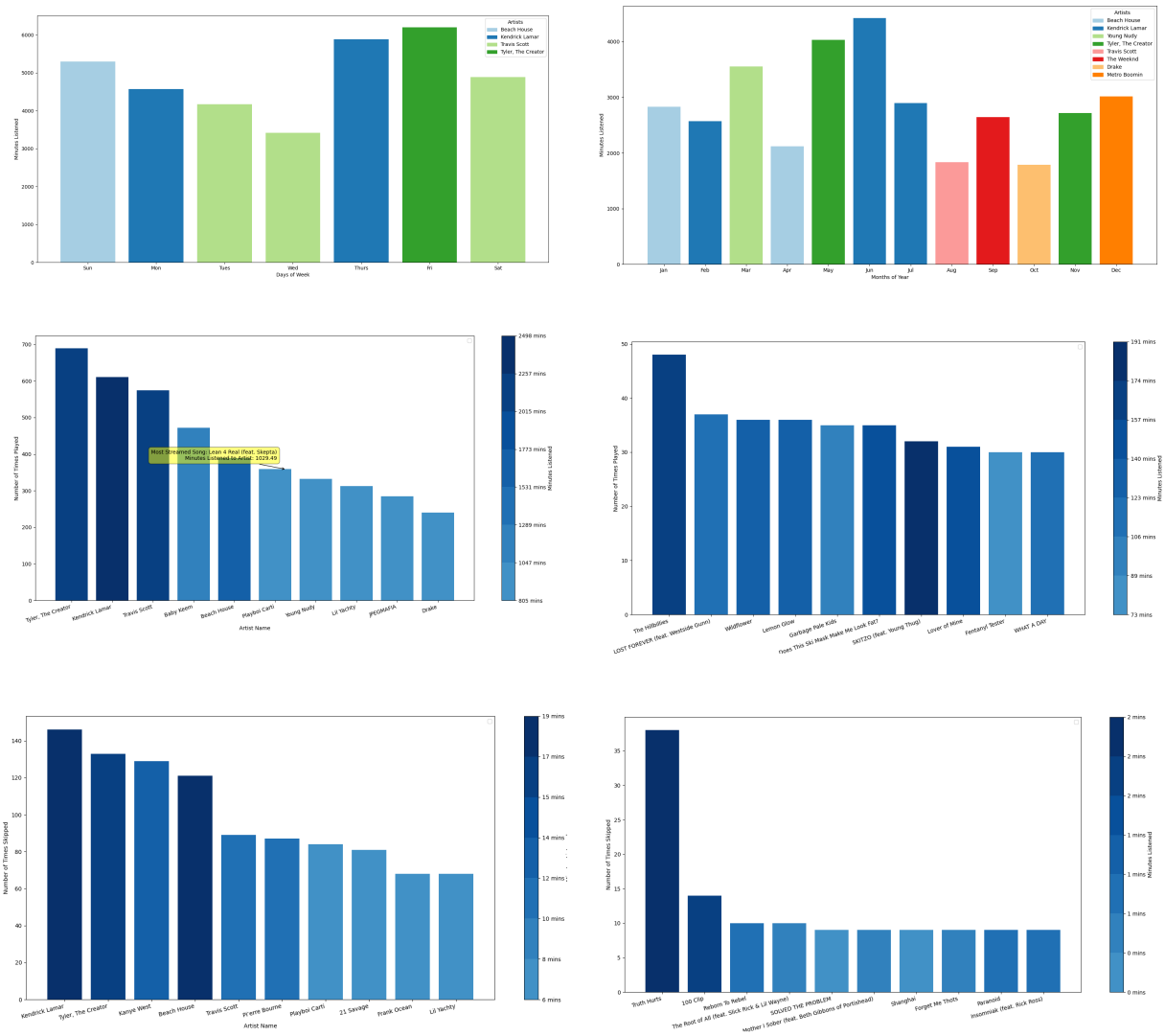
In this tab I introduced a Scatter Plot (very similar to project 2) that allowed users to change different numerical attributes of the x and y axis in order to compare different variables. This met my original goals again of just analyzing spotify data, and being able to see questions such as “do songs with high energy typically perform better in the charts?” There is also an option to reduce the number of points for better visibility.



This tab also contained a way to compare data from Apple and Spotify charts, so you could see where a song performed better and where more users enjoyed it. However, this might’ve been slightly biased since Apple Music keeps its data private for the most part, so there wasn’t much data for Apple Music.

Tab 3 — User Data

(As a quick side note, this is what this tab looks like when you click it for the first time, it lets you open the File Dialog and choose your CSV file. This isn't too terribly important but I'd never used QFileDialog before so I thought it was cool.)



Just to sort of "Rapid Fire" off all of these, since there were 6 different types of visualizations for a user's Spotify Data. For this, I decided to look at a user's listening over days of the week, months of the year, most listened to artists and songs, as well as most skipped artists and songs. I also plotted these as bar charts because it gave each entry a distinct value, and users would be able to compare them while looking at them very closely. For coloring, the first two show the most popular artist for the days of the week and months of the year respectively. I did artists rather than songs since the song would probably be different, but the artists would be similar so there would be more grouping and less different (almost arbitrary) colors.

For the most listened to / skipped artists and track, they were all pretty similar designs, so I'll explain them all together. I colored them in a sort of "density" fashion, where the more minutes listened to an artist, the darker the color of the bar. This seemed to work well, as it's interesting to see that the song or artist I played most was not necessarily the most number of

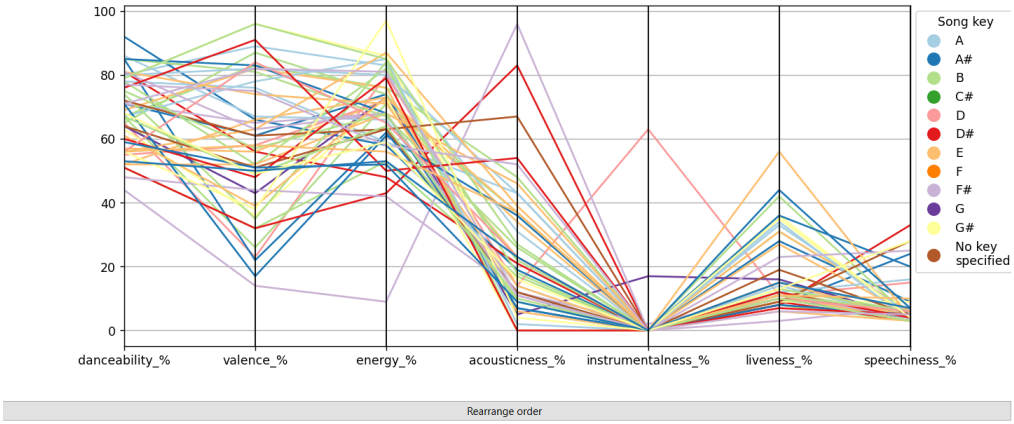
times played. I also added tooltips to all of these to show a bit more info, such as the most listened to song for a particular artist, or the artist of a most listened to song.

Also, this is how you are able to change to the different graphs, and you can include skipped songs as well.

This visualization helped my goals of visualizing a specific user's spotify data. It was an interesting way of representing how a user listened to music over the last year, and seeing who they listened to most. Users were able to see their minutes listened for each artist as well as number of times they listened to a song or artist.

Tab 4 — Similarity

For this tab, I added a parallel coordinates plot that allowed users to see the difference or similarity between the different variables that spotify makes for each song. The goal of this was to make a way to see how closely these values are related to each other, because spotify doesn't give any algorithms for how they obtain these values.



There is also a button to rearrange the order of these variables. I would've liked to make a way where you just drag them at the bottom by their name and it updates in real time, but didn't have time to do that. This would go hand-in-hand with tab 2, because it is also a way to illustrate the same variables, but this shows a continuous change between them and the scatterplot shows it more discreetly.

Challenges

Most of my challenges came from the actual coding process. One of the challenges not related to my coding was that we had this final project over Thanksgiving break, and I wanted to spend time with my friends and family that I hadn't seen in 3 months, so it was hard to find motivation. But I was able to work on it for at least 30 mins each night of break. Aside from my second tab, which was largely taken from project 2, I had to learn how to code everything. Tab 1 came fairly easily, as that was just a bar chart where the attributes were fixed. The only thing I changed there was the axis changing when the user updated the minimum or maximum year. One of the challenges I had here was making sure the user couldn't set the minimum year greater than the maximum, but that was fairly easy to implement as well.

For tab 3, a lot of the challenges came from aggregating the data for the tooltips. I've taken a SQL class before so some of panda's groupby functions made sense, but a lot of the time it was just guessing until I got something right (not too dissimilar from how I did the homework in that SQL class…). Another challenge for this was making the colorbar, or moreso deleting it. I was able to create it without much difficulty, but it was hard to remove it, as whatever I tried didn't work much. I ended up having to recreate my entire figure whenever I changed something. Tab 4 was also fairly difficult to get, as we didn't learn parallel coordinates in class, and it was the only tab where I *had* to make an additional class rather than just putting everything in my main class. This was a bit difficult to figure out. My final challenge came from my presentation as a whole, because I realized I kind of had too much to fit into 6 minutes, so I was rushing and cut some smaller things out from my final presentation as a result.

Outcome

Overall, I was really pleased with how my presentation came out. My final results looked pretty different technically from my original proposal, however it still covered a lot of the same things, maybe just in a different way. I was happy with my outcome that I was able to present in class, but I think before I consider it "final" and upload to my github, there are a few things I wanted to add, such as a "colorblind mode" button for the apple vs spotify data, possibly add the 5th tab that was supposed to show how similar a user's playlists were, add a way to preprocess the data in the code, and a way that compares the user's data to the most popular songs of that year. However, I think for the sake of this project, I was very happy with my results and the data I was able to visualize

Software

The only software I used for this project was Microsoft Visual Studio Code 2022, Microsoft Excel, and an online website that converted .json files into .xlsx files. In excel, there were a few preprocessing steps that I did, such as extracting the Day of the Week and Month of the year, as well as deleting one column called 'return' that was empty for every song, making a

‘minsPlayed’ column that converted the ‘msPlayed’ given to minutes and adding a boolean column called ‘skipped’ that would be true if the song’s playtime in mins was less than .5
In order to run this with a user’s personal spotify data, they would have to request it from their user settings tab, and then do the preprocessing above, but with the data I’m providing for this report that has already been done.

Runtime Instructions

In order to run this program, there are a few preprocessing components that must be done, if using your own data (data I have provided has already done this.) I am assuming that the user has already gotten their 1-year listening history from spotify, and if not, this can be found elsewhere.

1. Go to an online website that converts .json files to .xlsx files. Upload the "StreamingHistoryX" file, where X is a number you wish to use (Spotify cuts off the files every 10000 songs, you could also just copy/paste all of the parts into 1 file for a full comprehensive file). Download this .xlsx file.
2. In the .xlsx file, remove the 'record' column.
3. Add a column to the right of endTime and call it 'date'. Add this following code into the first cell and hit the green square to auto fill all the rows:
   `=LEFT(A2, FIND(" ",A2,1)-1)`
4. Add a column to the right of that called 'weekday' and apply this code to all rows as stated previously:
   `=(CHOOSE(WEEKDAY(B2),"Sun","Mon","Tues","Wed","Thurs","Fri","Sat"))`
5. Add a third column to the right called 'month', and apply the following:
   `=(CHOOSE(MONTH(B2), "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"))`
6. Add another column called 'time':
   `=RIGHT(A2,FIND(" ",A2,1)-6)`
7. To the right on 'msPlayed', add a 'minsPlayed' column with the following code:
   `=H2/60000`
8. Add one final column called 'skipped' with this code:
   `=IF(I2>0.5,"no","yes")`
9. Save this file as a .csv

After preprocessing data, make sure the following versions of python packages are installed:

```
Package            Version
------------------ -----------
attrs              23.1.0
certifi            2023.7.22
click              8.1.7
click-plugins      1.1.1
cligj              0.7.2
colorama           0.c4.6
contourpy          1.1.0
cycler             0.11.0
Fiona              1.9.4.post1
fonttools          4.42.1
geopandas          0.14.0
graphviz           0.20.1
joblib             1.3.2
```

```
kiwisolver        1.4.5
matplotlib        3.7.2
mplcursors        0.5.2
multidict         6.0.4
networkx          3.1
nltk              3.8.1
numpy             1.25.2
packaging         23.1
pandas            2.0.3
Pillow            10.0.0
pip               23.2.1
plotly            5.18.0
pyparsing         3.0.9
pyproj            3.6.1
PyQt5             5.15.10
PyQt5-Qt5         5.15.2
PyQt5-sip         12.13.0
PyQt6             6.5.2
PyQt6-Qt6         6.5.2
PyQt6-sip         13.5.2
PyQtWebEngine     5.15.6
PyQtWebEngine-Qt5 5.15.2
python-dateutil   2.8.2
pytz              2023.3
regex             2023.10.3
shapely           2.0.1
six               1.16.0
tenacity          8.2.3
tqdm              4.66.1
tzdata            2023.3
wordcloud         1.9.2
wordtree          0.1.0
```

In order to run the program, you just need to execute the following command in terminal:
`python .\spotifyvis.py -f .\spotify-2023.csv`
Of course, this would have to be adjusted for your file locations, but make sure you pass the -f flag or else the program will not run.