Tune Tracker

Team Members:

- Rebecca Kahn, rkahn@g.clemson.edu, C14523233
- Jake Liguori, jrliguo@g.clemson.edu
- Cierra Oliveira, colivei@clemson.edu, C73041981

Link to the project repository

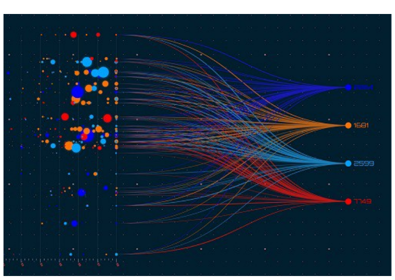
Overview and Motivation

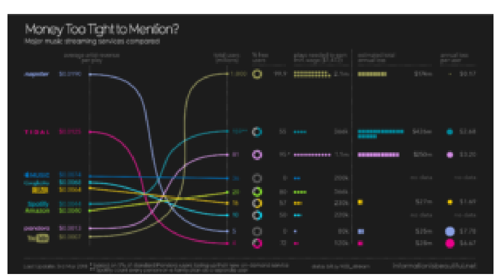
Our visualization project, Tune Tracker, seeks to understand the nature of music popularity. We aimed to learn about past patterns and anticipate future trends in music. We had one central question: **What makes a song popular?** To answer this question, we sought to find music data from a reliable and relevant source that provided attributes about particular songs that we could correlate to popularity. We used a dataset taken from the popular music streaming service Spotify to answer our research question.

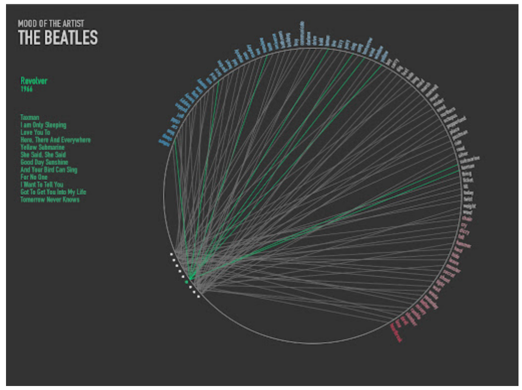
All three of us are users of a music streaming service, and we all regularly listen to music. Because listening to music is a part of our daily lives, we are interested in what drives us to listen to the songs we do. Are there features of popular songs that make them more likely for us to encounter them? We are motivated mostly by our own curiosity around the reasons why music becomes popular and where popularity extends. The implications of our project could potentially extend to those in the music industry as well. For those interested in reaching the charts, understanding the factors that drive popularity is crucial. Our research could also serve fellow music appreciators who wish to learn more about music trends.

Related Work

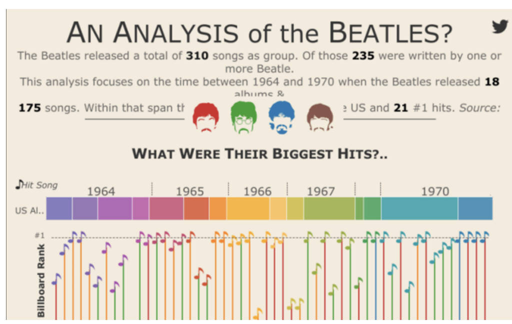
When we first began researching various ways people visualized data regarding music, we discovered very unique and colorful portrayals of different song-related data. Below are some visualizations using lines and force graph applications to display various information about music and one, more specifically, about the Beatles. All of these use lines to make it a more fun interactive experience for the user and we were drawn to the layout as well as the colors.



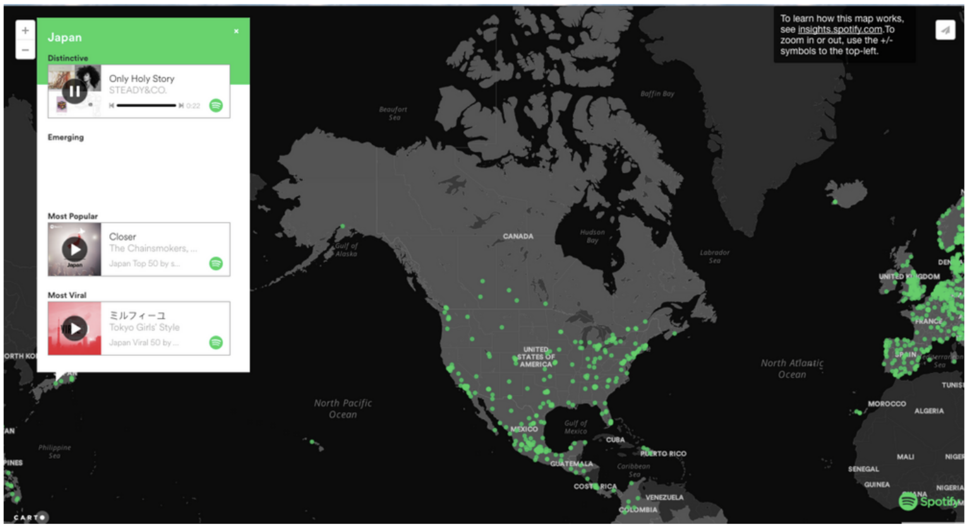




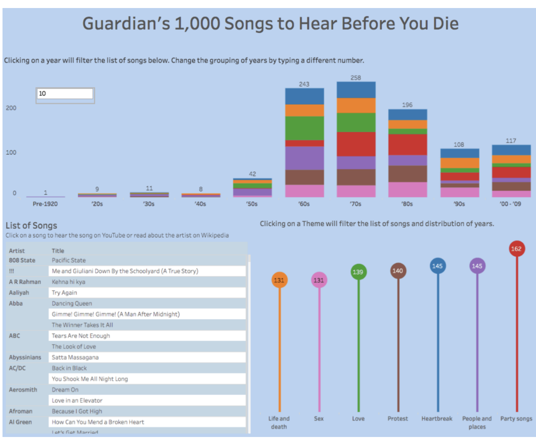
When deciding on other visualizations to put together with our project, we found this analysis of the Beatles that features small music notes and a bar at the top easily depicting the transition of their hit song over the years. We really liked the unique use of the music note and considered having a musical instrument or image to depict a genre of music and display the genre that way rather than just a color. All of these make use of colors to connect to songs which is what inspired us to connect genres to specific colors in our visualization.

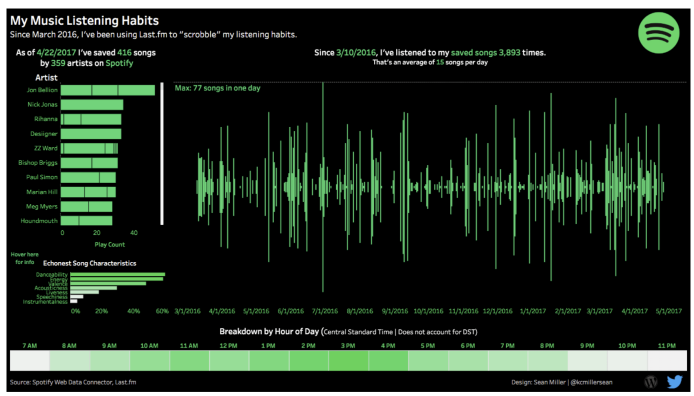


In addition to the visualizations featured above, we really liked Spotify's world map that displayed what songs people were listening to in different parts of the world. This visualization is fun, interactive, and interesting. The colors and song information definitely inspired us with our own visualization. This map is compelling and makes the audience immediately interested as soon as you look at it. This map also made use of the Spotify green color which audiences automatically associate with the streaming service to know where the data is coming from.



Below are some additional graphs we looked at before as well to kind of use as a basis for ours to determine which ones best depict musical data. All of these visualizations led us to using the ones we ultimately decided on for our final submission of Tune Tracker.





Questions

Tune Tracker seeks to answer one central question: **What makes a song popular?** We break this main question down into several sub-questions:

- *How do certain attributes of a song affect its popularity?*
- *How does genre affect the length of popularity?*
- *Do certain genres of music land on music charts more often than others?*
- *Of the top genres of music, what are the top artists?*

We also considered asking another general question: How does music popularity differ by U.S. region? We were interested in learning about which genres of music dominate different areas of the U.S. This question, however, would require additional data. We would have had to scrape data from Apple Music weekly charts for several different regions in the U.S. This data collection would have been much more complicated than using the ready-to-use data from kaggle. Due to time constraints, we decided to move away from this question and, instead, focus on our first central question.

To develop the sub-questions, we evaluated the attributes we had in the data. Because we had several predictors like danceability, acousticness, and energy, it made sense to investigate how these attributes correlate with popularity. There were other interesting facts about songs like the number of times charted which made us curious about the length of popularity.

Toward the beginning of our project, we were also interested in investigating how music popularity has changed over time. Specifically, we wanted to investigate if different genres became popular across different months of the year. Our theory was that different seasons of the year would affect which genres were popular. However, because we only had data for one year, we couldn't make generalizations about patterns over time.

Data

We used a dataset from Kaggle: [Spotify Top 200 Charts (2020-2021)](). This dataset contains all songs that appeared on the Spotify Global Top 200 chart from January 2020 - August 2021. It is hosted on Kaggle and was originally scaped using the Spotify API and spotifycharts.com. From Kaggle, we were able to download a CSV with all 23 attributes and 1,556 items.

We did not need much data cleanup. This data comes directly from Kaggle and is already in a tabular format with clean values, making it easy for us to extract and use data. One instance of cleanup we need is with dates. The data set provides dates in a week format (e.g., 2021-07-23--2021-07-30) and we need to extract months. Additionally, the genre cell for each song contains a list of genres (e.g., ['dance pop', 'pop']), meaning we can clean this attribute by generalizing the songs into broader genres (pop, rock, hip-hop, etc.) or leave it as is and let each subgenre be its own category. Data processing was implemented through D3, where we read and stored the data. Afterwards, helper or accessor functions were written to extract the relevant portion of the element and convert it to the right data type, but in almost all attributes the data was ready to be plotted.

Exploratory Data Analysis

We first looked at several distribution plots of different targets we could use for popularity, such as number of streams, Spotify's 0-100 popularity ranking, and number of weeks on the charts. We found the popularity attribute to be very skewed to the left, which makes sense considering these are charting songs and are likely to be highly popular. The number of streams was skewed to the right, but the distribution still seemed better suited for our target value than popularity.

Similarly, we plotted distribution plots of the attributes and found most were also highly skewed either right or left. This insight influenced several design choices. First, opacity should be reduced to make clusters of points more visible. Second, the aspect ratio should be adjusted to spread clusters out further (i.e., we would need to make the visualization wider to see more isolated points). Finally, given the insight that the distributions of both the x and y variables were skewed, we decided to rethink the scales used on the scatterplots. Rather than a linear scale, we considered and eventually used a logarithmic scale for our scatter plots, which reduces any extreme clustering.

The nature of the data also led us to rethink the intent and purpose of the first visualization. Rather than focus on finding trends, it may just serve better as a tool for seeing where specific songs and genres fall for each attribute. Therefore, the interactions implemented later focused on adding customization to explore the differences in streams and attribute scores as the genre and attributes selected change.

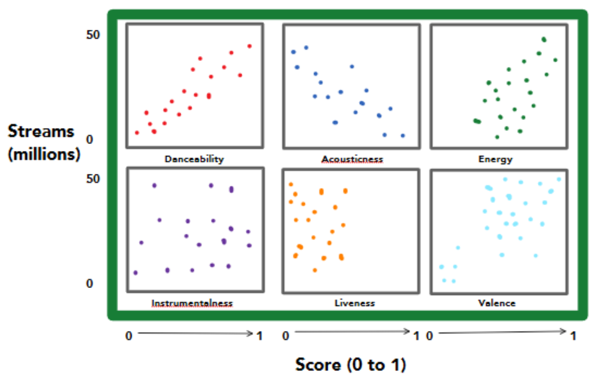
We also explored the genre data since the other two visualizations primarily involve genres. An important insight was that each song was classified by Spotify as several different genres, and Spotify has a lot of subgenres. This would be too many to visualize, so we needed to narrow them. First, we gathered a list of “general genres'' by identifying the most common “general” genres, like pop, rock, and hip-hop. Next, we filtered genre names to match a general genre (e.g., pop-rock, dance pop, pop italiano all fall under the general genre of pop). The insight that Spotify provided a list of genres for each song also determined the implementation of our genre filter. Since certain songs may be classified as multiple general genres, like pop and hip-hop, we felt it would be reasonable to allow them to be shown for both filters. Therefore, the interaction was written to simply check if the genre chosen is contained in the song’s list, rather than if it matches the singular general genre.

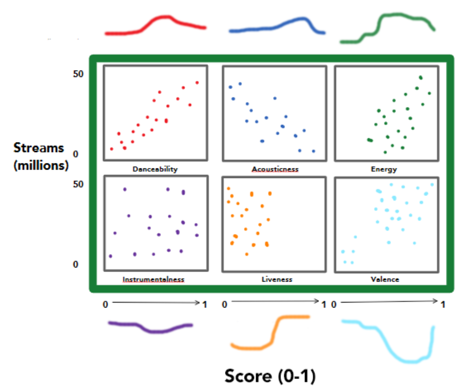
Finally, simple count plots and tables of the top artists and genres provided the data needed for the bubble plot. The top 5 artists of each genre shown in the data exploration were used in the final visualization. However, because of the multiple genre classification mentioned before, there is some overlap in artists (e.g., Drake is top 5 in both Pop and Hip-Hop).

Design Evolution

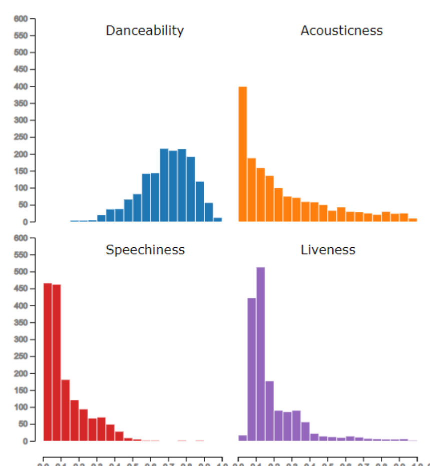
Scatterplot Subplots

We wanted to explore any patterns between Spotify's attributes such as Danceability, Energy, Acousticness, etc. and the number of streams the song reached on the charts. In addition, since the dataset is all charting songs, we wanted to visualize the distribution of these attributes and see where most popular charting songs fell on a 0-1 scale of these attributes. The first design included 6 subplots, one of each attribute, and a second version with distribution plots along the sides.

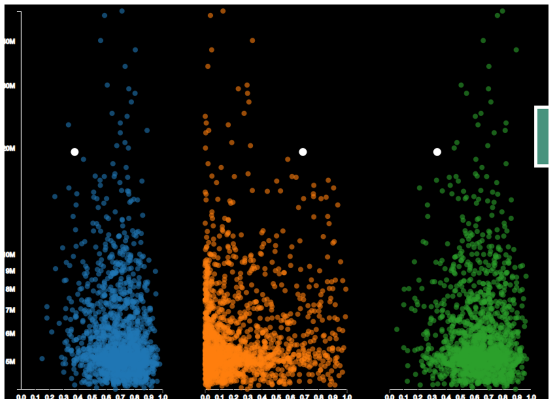




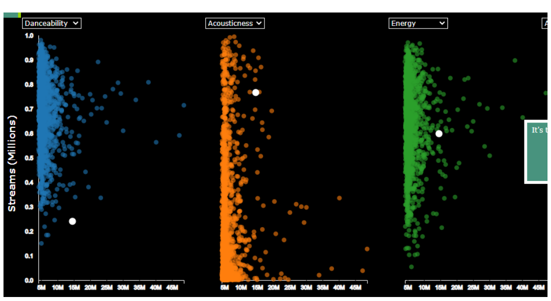
After some additional data exploration, it seemed many of the points on the plots were too clumped together on the y axis (i.e., the distribution of streams was very right skewed). Consequently, we explored another design with distribution plots of each attribute rather than scatter plots. This design did not last long as we felt interactable song points were integral to the project.

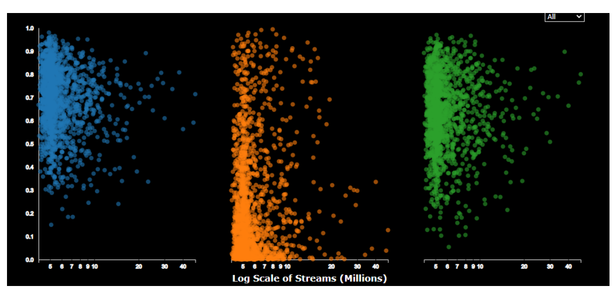


We returned to the scatterplot design, and to remedy the problem with the distribution we changed the stream scale to a log scale. Also, instead of plotting all 6 attributes at once, we cut it down to just three subplots. This way, the y-axis could be lengthened and an interaction could be added for the user to select which of the 6 attributes s/he would like to see. Finally, the opacity of the points was lowered, which allows the user to spot clusters (e.g., a darker orange in the lower left corner indicates a lot of charting songs have low acousticness scores).

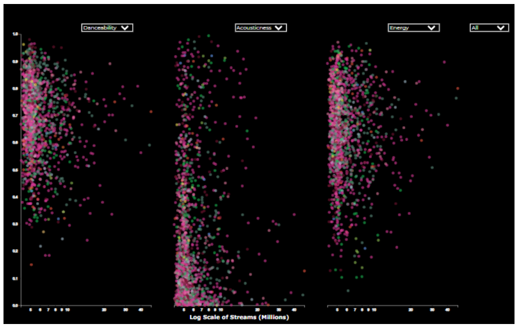


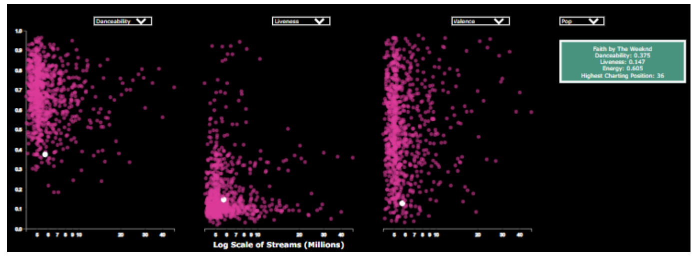
The main issue with this design is it is very difficult to determine the differences between the scores of each attribute. The song selected in the image above has a higher acousticness score (middle) than danceability and energy (left and right), but this is not immediately apparent. By flipping the axes and placing streams on the x axis, the clicked points would differ in height, which is much easier to perceive than the lateral differences shown above. The axes were swapped, and as expected the differences in attribute scores are perceived more easily. The linear scale is shown on the left and the log scale - which was used in the final version - is shown on the right.





Finally, upon receiving feedback after our presentation, we realized the colors in the scatterplot visualization and the bar chart visualization were inconsistent. While the scatterplots were color coded by attribute, the bar chart was color coded by genre, and some of these colors were the same. This creates confusion for the user, as the same color being used for two different purposes will make it unclear what is being indicated across different parts of the website. We decided to color code everything by genre, regardless of attributes selected.

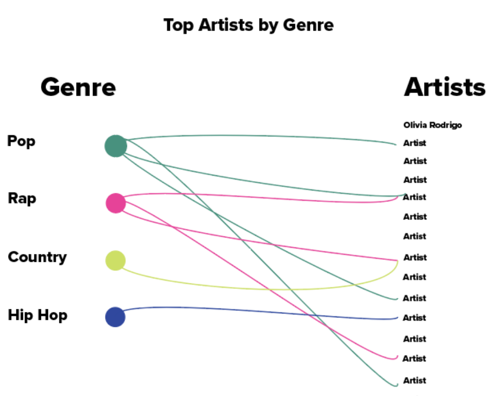




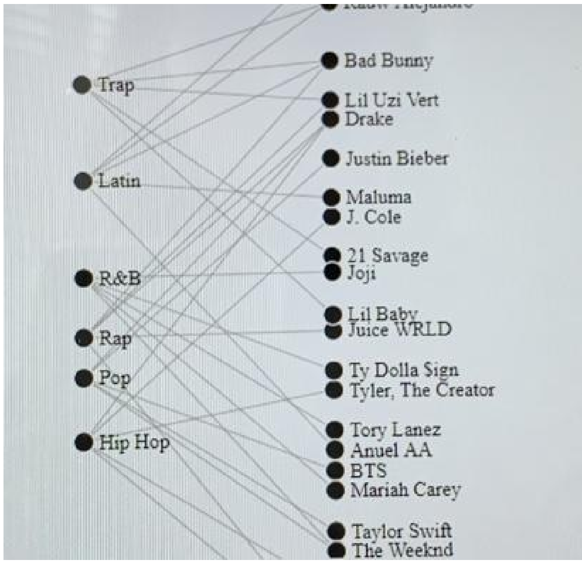
Artist-Genre Network Graph

We wanted to investigate which artists are the most popular amongst each genre. Conceptually, we sought to have entities that represent genres linked to several entities that represent artists. We understood that for some artists, they were associated with multiple genres. We needed a way to link genres to artists that accounted for multiple genres describing an artist. To visualize this, we decided to use a network graph.

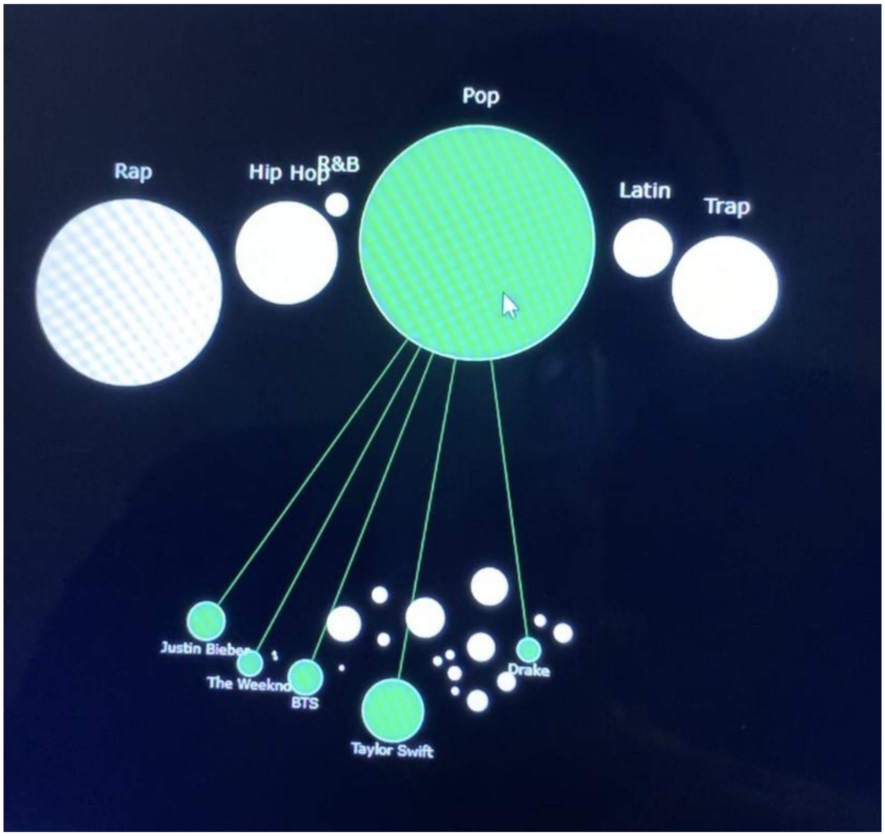
The design process began with a conceptual sketch. We drew several genre nodes that are all aligned vertically. We listed the names of all of the artists that were the top of their genre in vertical alignment, parallel to the genre nodes. All artist names were listed once, even if they were the top of multiple genres. Next, we drew lines that connected genre nodes to their corresponding artists. To distinguish between the different genres, we assigned different colors to each genre node and its links. The first conceptual sketch is included below.



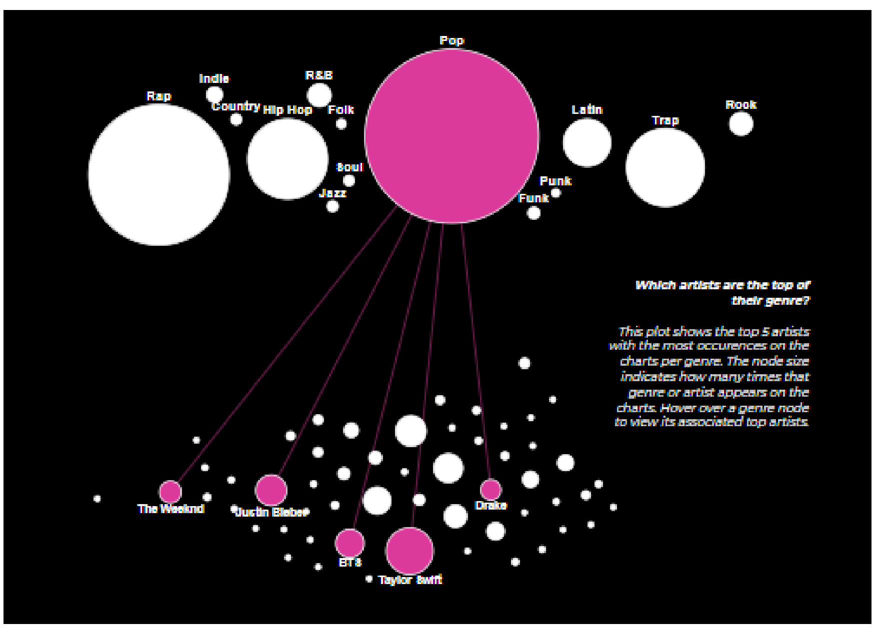
Next, we built our first prototype of the visualization with D3. Again, we created nodes for each genre and aligned them vertically. Unlike the previous design, we also created nodes for each artist. We linked each genre node to its corresponding artist nodes. At this step in the design process, we did not assign colors to each individual genre node. The first prototype is included below.



Next, we considered moving away from a graph with nodes that are vertically aligned in two columns. Instead, we created two force-directed clusters of nodes: one for genres and one for artists. Again, we created links from each genre node to their respective top artists. Unlike the previous design, we decided to not show every link and artist name at once. To avoid clutter, we allowed links and artists names to appear once its corresponding genre was hovered over. We used a green color to distinguish the genre/artist node hovered over from the others. This version of the design is shown below.

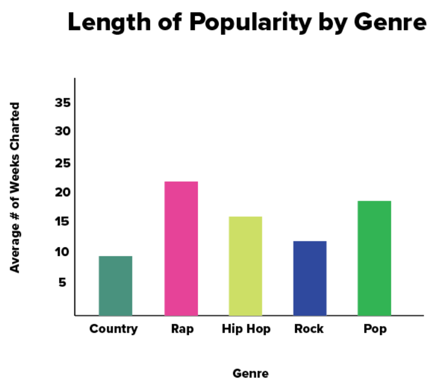


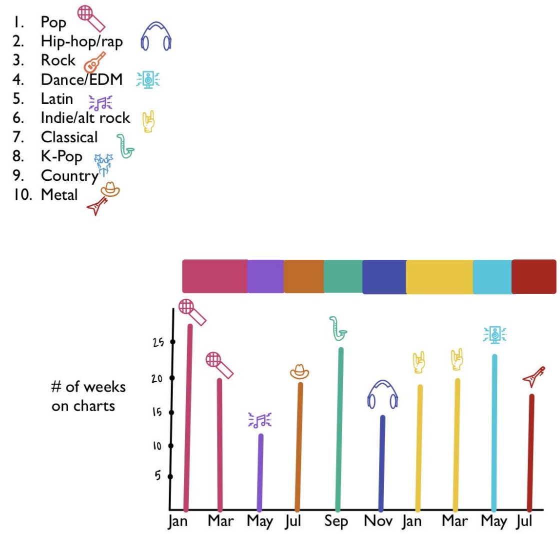
For the final version of our visualization, we used different colors to distinguish between the genres. For consistency and to avoid color clashing, we gave every genre a unique color and used that color on every visualization on the page. The final design is included below.



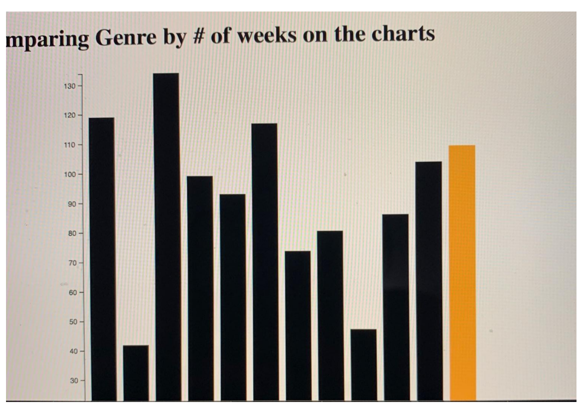
Genre Popularity Bar Graph

We wanted to display the impact genre has on the number of weeks on the charts to connect to the two other visualizations. This way, the audience could see which genres are actually considered most popular and highest charted on the charts. The drawings below represented the original plan we had in mind for creating this visualization so that viewers could compare their genres

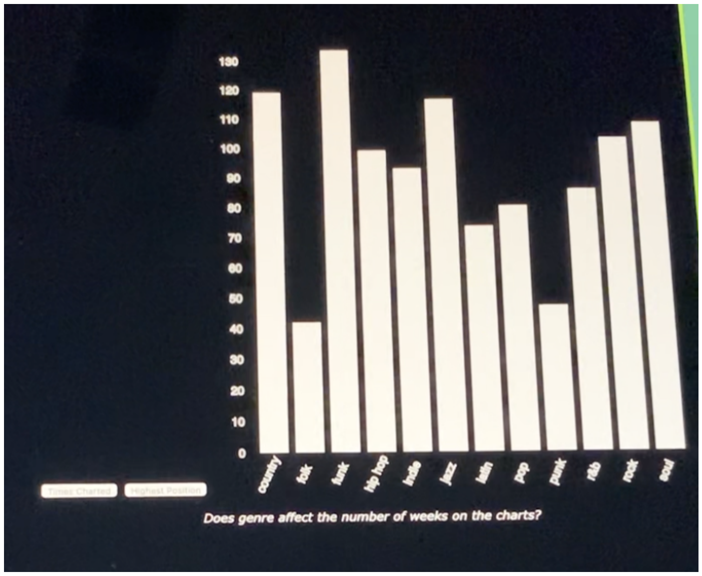




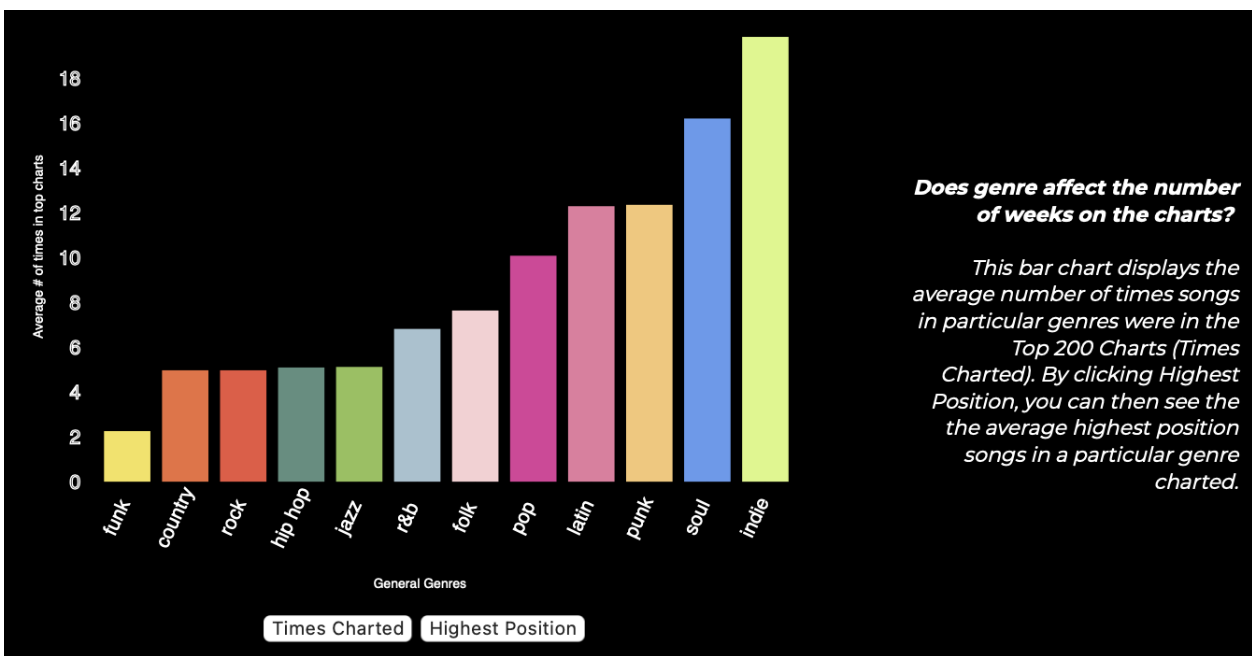
Next was actually implementing this drawing into an actual bar graph using javascript and D3. We got the first general genre of each song and organized all twelve of those genres into groups into which we got an average number of times charted value for each genre.



We decided we wanted to display a little more information for this visualization to display not only more information and a different comparison, but also more colors and more interaction. As you can see in the image below, two buttons have been added to be able to switch between two different datasets (one is average highest position on the charts and the other is average number of times charted for each of the genres).



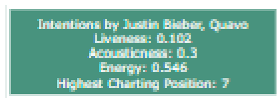
Finally, after messing around with different color schemes, we decided to use these colors and implement them throughout the entire visualization. This bar chart displays all the information in a way that is easy to read and understand from a user perspective while also implementing all the tools from javascript. Originally, it was difficult to fix the x-axis and y-axis and end up actually flipping the y-axis for the “Highest Position” graph, but after playing around with the various features and becoming more familiar with javascript, we were able to develop a fully functional bar graph.

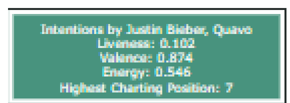


Implementation

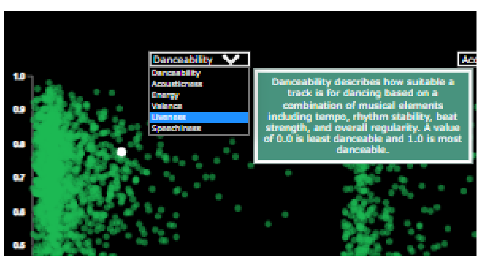
Scatterplot Subplots

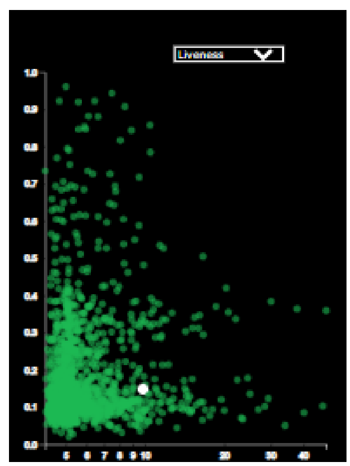
To best visualize the differences in attributes for each point, the user should be able to locate where a song is on each subplot at once. Hovering over a point will highlight it, and clicking it makes the point highlighted and larger on every subplot. The user would also like to know exactly what point they are looking at, so the green text box next to the plot shows the name of the song and its artist, scores, streams, and highest charting position of the highlighted or clicked song. The song info will also adjust as new attributes are selected.





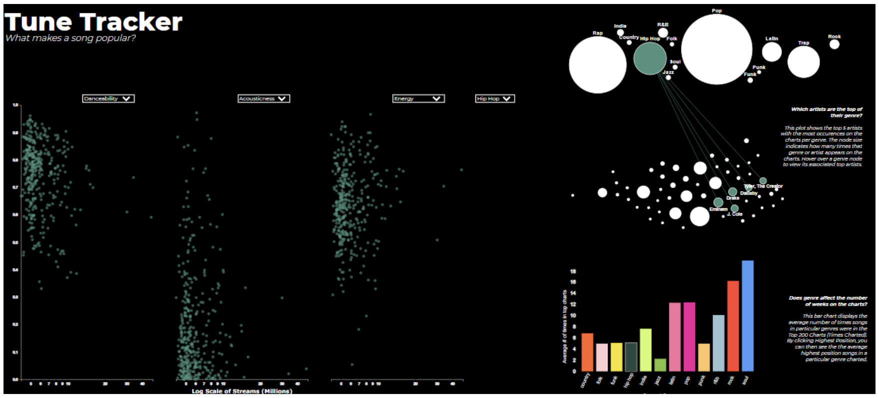
The final version of this visualization includes only 3 subplots, although our initial proposal included 6. To still allow the user to visualize all attributes, dropdown interactions were added to allow the user to change attributes. After a change to a dropdown is detected, the y value of all points is shifted. A transition was added to allow the user to watch a clicked point move to its new value. Additionally, hovering over the dropdown provides a description of the currently selected attribute, as certain attributes may have unclear names.





Global Genre Filter

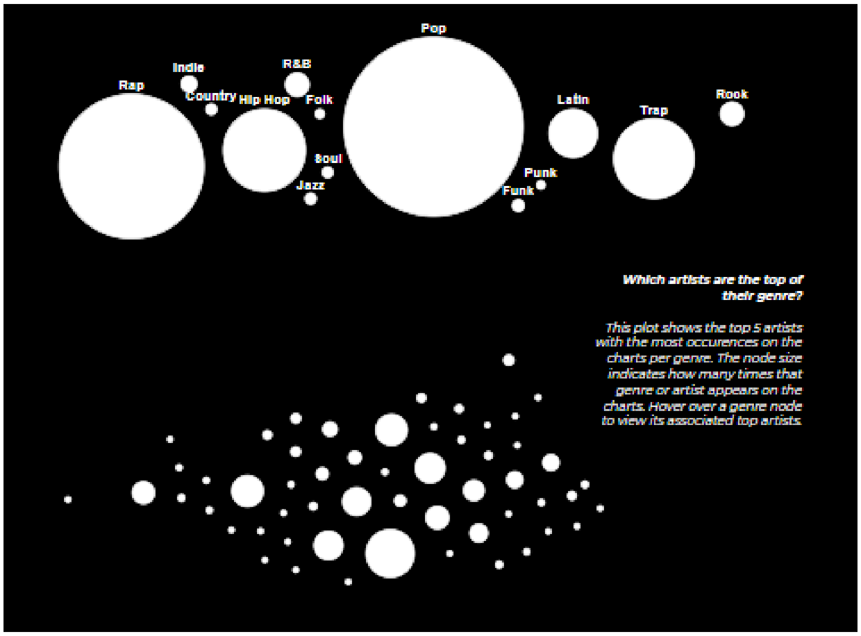
Each visualization has a function for filtering the genre of the entire page. Originally, only the dropdown filter near the scatter plots existed. However, If a user wanted to highlight the genre on the bubble plot, it seemed counterintuitive to force him or her to use an option on another part of the page. Instead, a bubble, bar, or dropdown can be clicked and the entire page will filter. Filtering the entire page 1) prevents any confusion caused by graphs highlighting separate genres and 2) allows the user to easily focus on analyzing a favorite genre. The bar graph does not include 2 genres (Rap and Trap) that the other two do. In the event one of these are clicked, the bar graph filters to “All” (nothing highlighted).



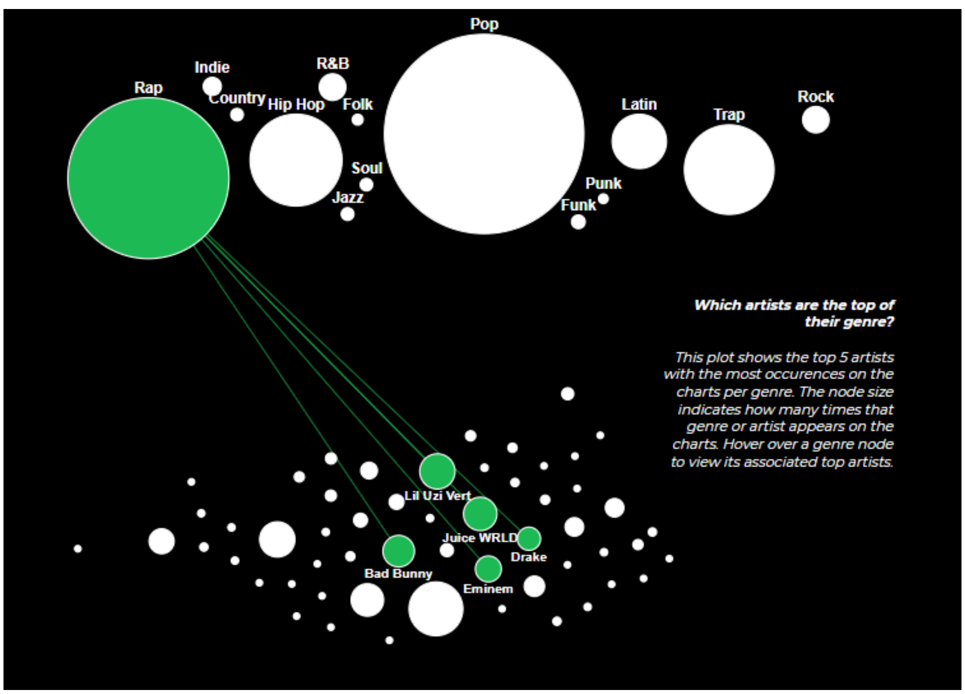
Artist-Genre Network Graph

To display the top artists of each genre, we designed a network graph. We used a force directed graph to visualize genre and artist nodes and their relationships. We created nodes for each genre and artist, and provided a value based on the number of times that genre/artist occured on the charts. We also created links where the source is a given genre and the target is one of its associated top artists.

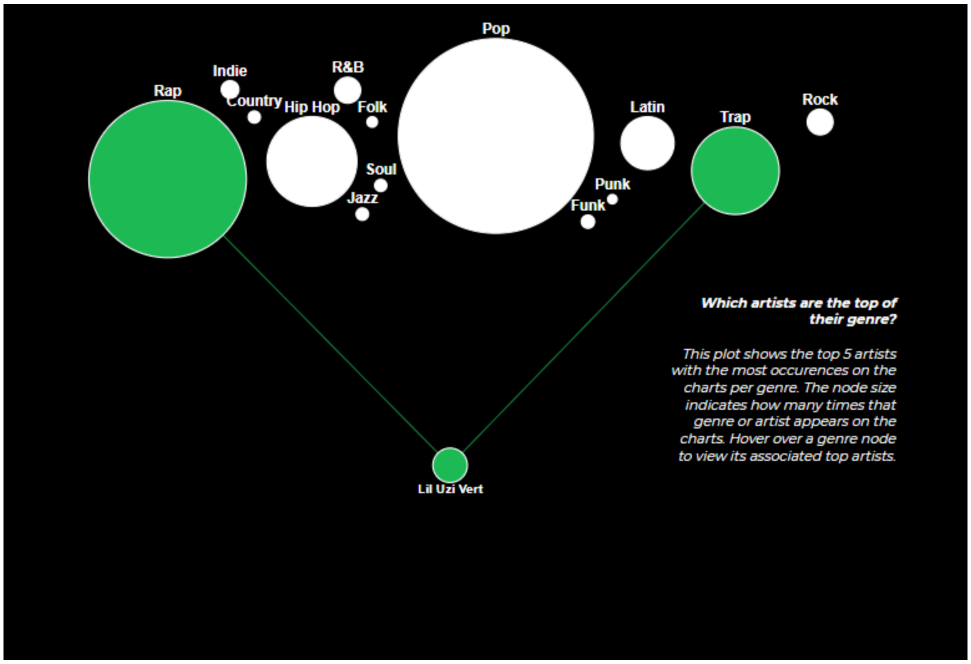
Our graph consisted of several forces. To separate the nodes into 2 clusters, we used the D3 forceY function and assigned the genre nodes a Y value of 200 and artist nodes a value of 700. This created two distinct clusters. We also used a collision force that was a function of the size of the node. In other words, larger nodes had a higher collision force. Lastly, we used a center force to push all of the nodes towards the center of the SVG element. The resulting design is included below.



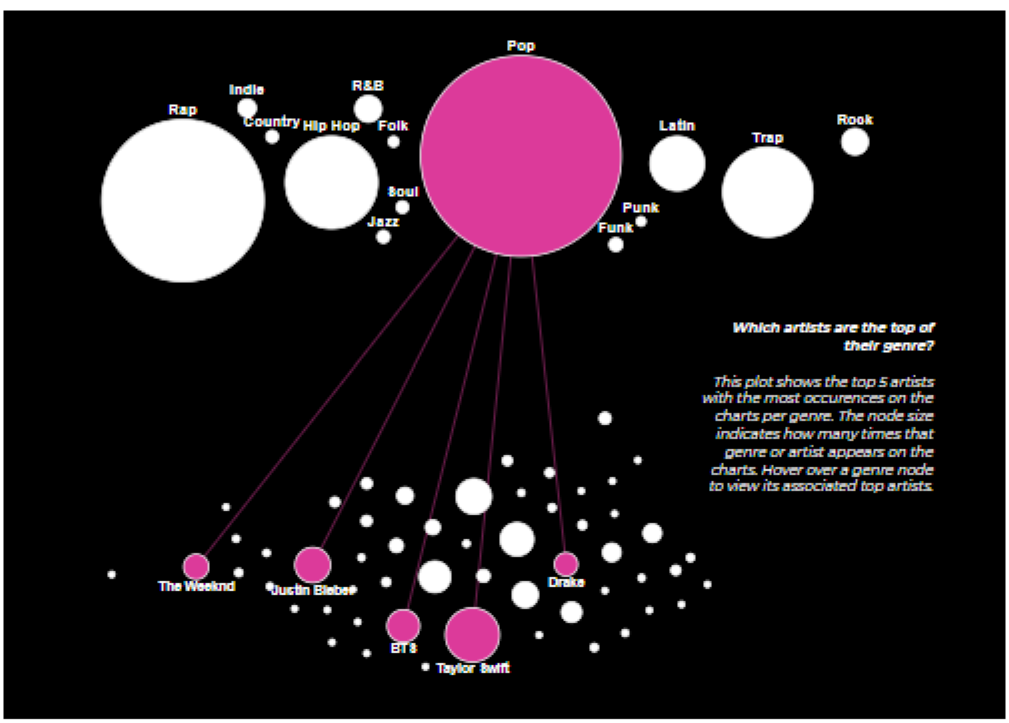
We also created several interactions for this visualization. When uses hovered over a given genre, the genre node, its associated top artist nodes, and the links between the two are highlighted. The names of the connected artists also appear when a genre node is hovered over. Below is an example of the hover-over interaction.



When users hover over an artist, all other artist nodes disappear, and the associated genre or genres are highlighted. The name of the artist and the link between the artist and its genre or genres appear as well. Below is an example of the artist node interaction.

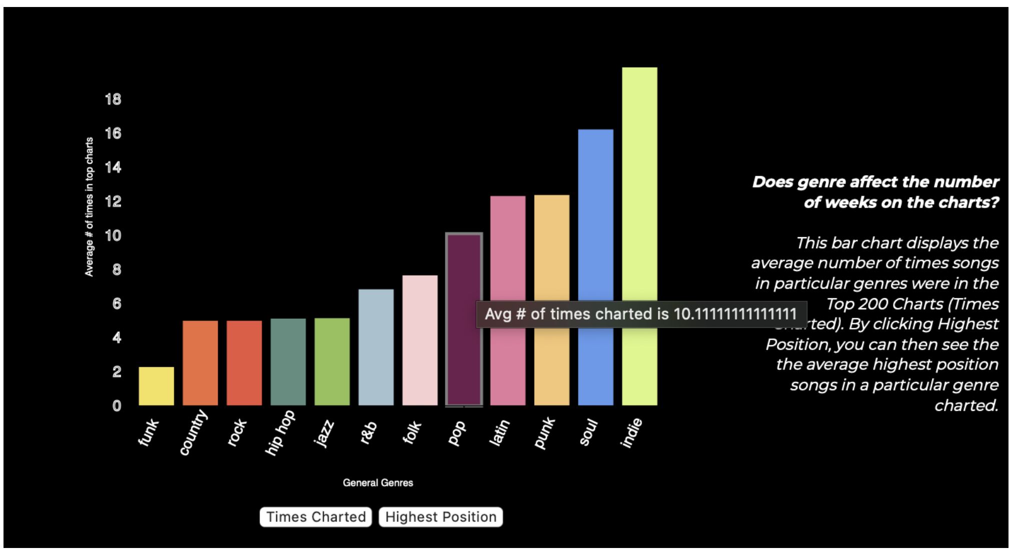


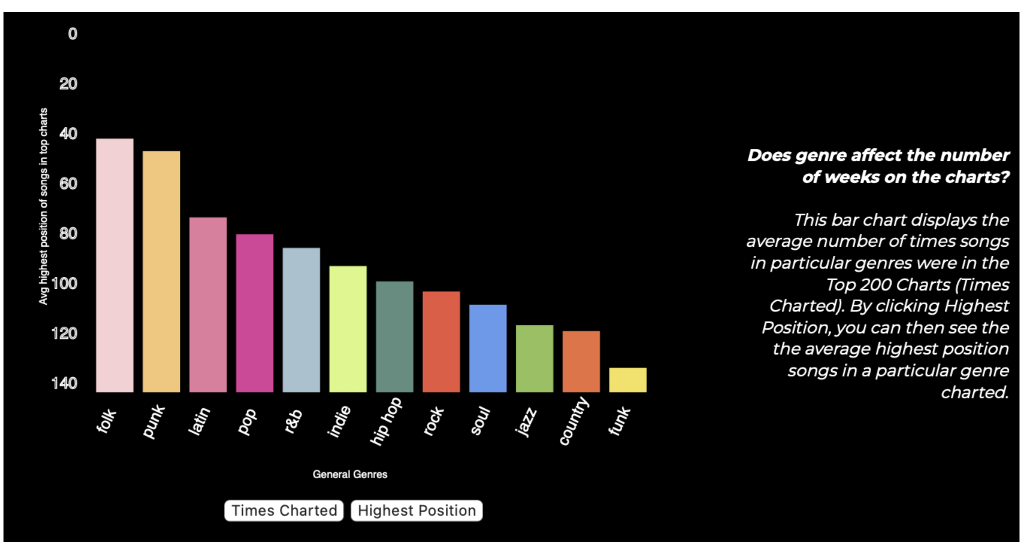
Lastly, when users click on a genre node, the genre turns a specific color that is used to identify the genre globally. Its top artists and its links are also highlighted in that color. Like mentioned before, clicking a genre node triggers a global interaction that filters data for that genre across all visualizations. When users move their mouse away from the clicked genre node the highlight stays, unlike when users hover. Below is an example of when a user clicks on a genre node.



Genre Popularity Bar Graph

To display the impact of genre on the number of weeks on the charts, we switch between two different bar graphs to show the differences between average highest charting position per genre and average number of times charted per genre. When you switch between the two by pressing one of the buttons below, there is a transition with almost every single piece displayed on the bar graph. The x-axis moves around based on where the genre is in the other bar graph, and the y-axis will smoothly flip when switching as well. The bars themselves actually will fade in color and switch the the genre that it actually represents as the x-axis and y-axis are adjusting accordingly. You can also hover over each of the genre bars to see the actual value displayed to further compare and the opacity will change and also add a grey box outline around the bar you hover or even click on. When clicking on a bar, it will remain highlighted and the adjustments can be seen on the other visualizations to represent that one specific genre as well.





Evaluation

Our overarching question was what made a song popular, whether that be certain genres or other attributes. Both the bubble plot and bar chart clearly show some genres have more songs on the charts - and more longevity on the charts - than others. Notably, we learned from our visualizations that Pop and Hip-Hop and unsurprisingly two of the most popular genres. The scatter plots also reinforce this as filtering by genre provides a sense of how many points each genre has, and Pop and Hip-Hop show the most.

For the Spotify attributes (Danceability, Energy, etc.), the nature of the data made it difficult to visualize any obvious relationships between streams and scores. However, given our whole dataset was charting songs, visualizing the distribution of points within each attribute told us how successful charting songs tend to sound. Most have high Danceability and Energy and low Acousticness, Liveness, and Speechiness. Valence, or how positive a song sounds, showed a fairly normal distribution and a low correlation with streams, implying the positivity of a song does not noticeably affect its popularity.

Finally, we also wanted to explore which artists had the most success on the charts. This was accomplished well by visualizing the top 5 artists of every genre. We learned some artists dominate multiple genres, and pop artists like Taylor Swift and Justin Bieber produced the highest number of charting songs. The hover-effect of the scatter plots also allowed for some visualizations of specific artists, as the user can click a point they are interested in and see how the attribute scores and popularity data of the song compare to the rest. Using this feature, we were able to learn what some of the most popular songs and artists of the year were.

One further improvement we could have made with more time is artist filtering. In the same way clicking a genre bubble filters the scatterplots, clicking an artist could highlight all their points on the plots. This would allow deeper analysis of a particular artist or just add another fun element to exploring the dashboard.

We could have also added lines connected highlighted points to show the differences in attribute values. Although it is pretty easy to see the differences already, adding lines may help the user distinguish where the highlighted points actually are in the highly-populated graph. Similarly, a hover effect showing the chosen points and connecting lines on each plot would allow a user to explore several points quickly without clicking them all, although it may have made for a busier visualization.