École Polytechnique fédérale de Lausanne

COM-480 – Data visualization – Spring 2024

Project : Process Book

Hanwen Zhang
Heling Shi
Jingren Tang

1 Introduction

In the dynamic landscape of the music industry, hit songs not only reflect musical innovation, but also echo cultural change over time. Our visualisation project meticulously documents the evolution of popular music from 2000 to 2019.

We use a series of insightful visualisations to illustrate changes in the popularity of different genres, the rise and fall of artists, and the distinguishing characteristics of hit songs from one year to the next. This work aims to illuminate patterns of musical dominance and evolution, providing users with an interactive exploration of hit songs through engaging visual representations.

2 Problem statement

As outlined in our first milestone, our objective was to identify shifts in preferences for song styles and lyrics across different years, thereby enabling songwriters and listeners to deepen their understanding of musical trends. While this goal remains steadfast, our research has evolved to focus more comprehensively on exploring changes in song styles and genres from multiple perspectives, placing less emphasis on lyrics. This shift allows our investigation to become more focused and in-depth. With this refined approach, our data visualization now aims to answer the following questions :

With our data visualization, we want to answer the following questions :

- How have musical genres evolved over the last two decades ?
- Which artists dominated the music scene in various years ?
- What are the defining characteristics and genres of music that resonate with audiences ? How do they change over time ?

3 Dataset

The dataset "Top Hits Spotify from 2000-2019" encompasses comprehensive information for about 2000 songs, offering a rich resource for analysing the music industry. Each entry in the dataset contains the artist's name, song title, release year, and other key features relevant to its popularity, musical attributes, and lyrical content. Musical features such as danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, genre, and the overall estimated tempo of a track in beats per minute (BPM). The dataset, popular on Kaggle and consisting of approximately 2000 rows, contains no NaN values. Therefore, it minimizes the need for extensive cleaning.

4 Project structure

4.1 Architecture

This project architecture consists of frontend components, data processing, and configuration files. The details are as follows :

4.1.1 Frontend Structure

Frontend Folder Structure :

- `src/`
  - `App.css`, `App.tsx` : Main styles and main component files, defining the overall structure and style of the application.
  - `main.tsx` : Application entry file, responsible for initializing and rendering the main application component.
  - `assets/` : Contains static resource files such as images and fonts.
  - `components/` : Contains functional component files and their styles, such as `BubbleChart.tsx`, `Features.tsx`, `Navbar.tsx`, etc.
  - `picture/` : Contains image resources.

4.1.2 Data Processing

Data Files :

- `songs_normalize.csv`, `songs_normalize.json`, `songs_normalize_cleaned.json` : Store and process the data used in the project.

4.1.3 Configuration Files

Configuration Files :

- `index.html` : Main HTML file defining the basic structure of the webpage.
- `package.json`, `package-lock.json` : Define project dependencies, script commands, and metadata.
- `tsconfig.json`, `tsconfig.node.json` : TypeScript configuration files, defining compilation options and project structure.
- `vite.config.ts` : Vite configuration file, defining the project's build and development server configuration.

4.1.4 Other Files

Other Files :

- `README.md` : Project documentation, including description, installation, and usage instructions.
- `node_modules/` : Contains third-party libraries and modules that the project depends on.
- `public/` : Contains public static resources.

4.2 Tools

- **Node.js** : For running JavaScript code and managing project dependencies.
- **TypeScript** : For type-safe frontend development.
- **Vite** : As the development server and build tool.
- **React** : For building the user interface.
- **Jupyter Notebook** : For data analysis and exploratory data analysis.
- **npm** : For package management and project dependency management.

5 Visualizations

5.1 Welcome Page and Layout

Upon opening this website, users will first see a welcome pop-up window. The pop-up window reads ‘Welcome to the world of music’. By clicking on the ’Explore’ button, users can directly reach our ’songs’ page.

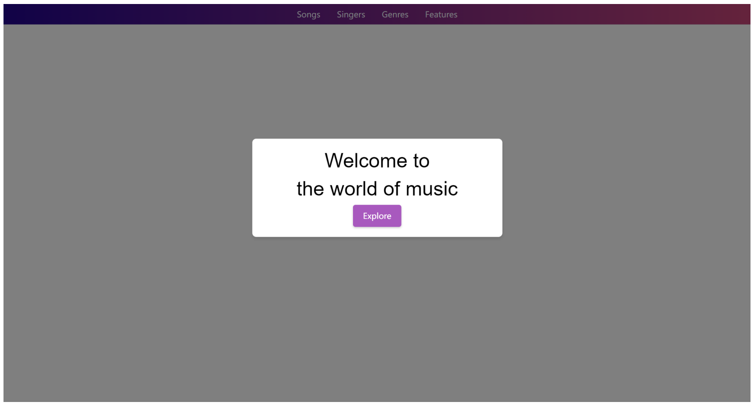


FIGURE 1 – User will first get a welcome pop-up.

Then, with the navigation bar at the top of the page, users can freely switch between our four pages : songs, singers, genres, and features.

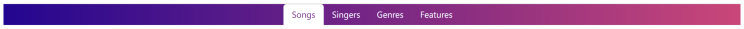


FIGURE 2 – User can switch between pages with the navigation bar.

5.2 Songs

Objective

The purpose of this page is to display popular songs from each year of the dataset, arranged according to a timeline. Attributes displayed include song title, artist, popularity and duration.

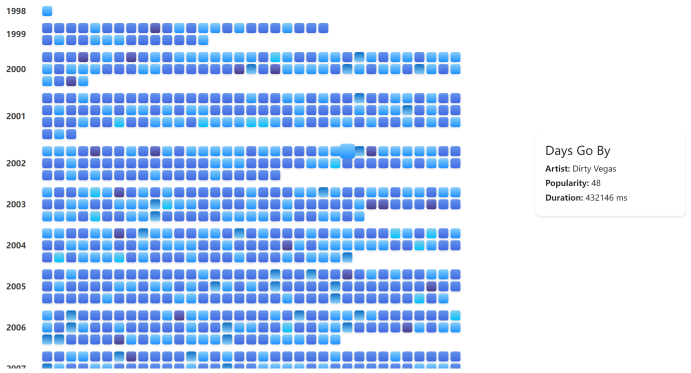
Page Layout

The webpage is divided into three main sections, designed to provide a comprehensive view of the number and popularity of songs released each year :

Top Navigation Bar : Users can switch to different pages, such as Songs, Singers, Genres, and Features, via the top navigation bar.

Song Matrix : The main section of the page displays a matrix where each square represents a song. The color of the square indicates the song's popularity, with a gradient from light blue (low popularity) to dark blue (high popularity). The matrix is organized by year, with each row representing a different year and each square within the row representing a song released that year. When a user clicks on a square, it highlights the song and triggers the display of detailed information about the song on the right side of the page.

Song Details Box : When a user clicks on a square, detailed information about the song, including the title, artist, popularity, and duration, is displayed on the right side. The details box provides a concise summary of the song's key attributes. It is positioned fixedly on the right side, making it accessible without obstructing the view of the song matrix.



Days Go By
Artist: Dirty Vegas
Popularity: 48
Duration: 432146 ms

FIGURE 3 – Yearly Song Popularity Matrix : Users can click on a song to see detailed information, including title, artist, popularity, and duration.

Challenges and Design Decisions

Data Representation :A significant challenge in the design was displaying the number and popularity of songs released each year. To address this, we used a matrix layout grouped by year, allowing users to intuitively see the release patterns of songs each year.

5.3 Singers

Objective

The objective of this page is to show the characteristics of the top five artists with most songs in the dataset. Displayed attributes include their active periods, song releases, popularity, and styles.

Page Layout

This has four main parts, providing a view of the artist's impact and evolution over time :

Timeline : Positioned at the top, the timeline correlates directly with the bar chart. As users hover over specific years on the timeline, it dynamically magnifies this rectangle. The color of the timeline shifts in sync with the bar chart to indicate changes in the leading figures in the music industry over time.

Bar Chart : This main part of the page displays a bar chart where each bar is shaped like a triangle, representing musical waves. The height of each triangle represents the song's popularity. To align with the website's theme, musical notes are placed at the peak of each triangle. The colors of the triangles correspond to different artists, matching the frame colors of the artist's images below, thereby maintaining a cohesive visual theme.

Artist Images : Below the bar chart, images of the artists are displayed. Hovering over an artist's image highlights all songs by that artist in the bar chart by darkening their triangles and enlarges the artist's image. At the same time, the artist's name is displayed at the bottom of their image.

Radar Chart : At the bottom, the radar chart implemented using the react-chartjs-2 library's radar component allows interactive comparison of artist data. Users can can choose to show or hide data for each artist by clicking the legend, facilitating easy comparison. Tempo data is scaled down by a factor of 1/160 to ensure all attributes fit within a 0-1 range.

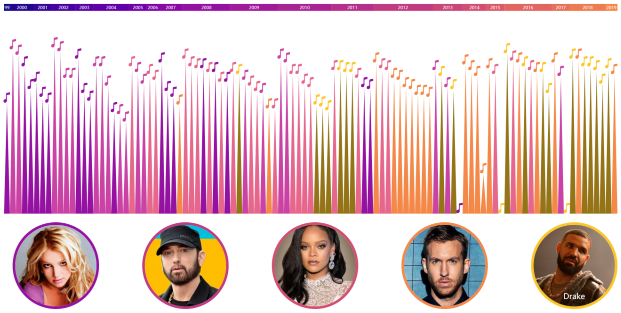


FIGURE 4 – Timeline, bar graphs and artist pictures : User can hover over an artist to see all his/her songs

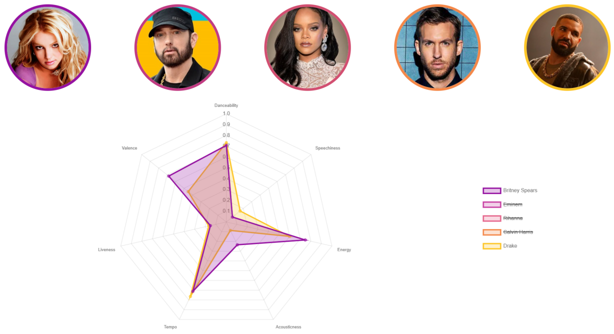


FIGURE 5 – Radar chart : users can choose to keep only some of the artist's data for easy comparison

Challenges and Design Decisions

Data Representation : One of the significant challenges was the varied number of songs per artist per year and instances where multiple songs in the same year could belong to the same artist. There was no readily available bar chart component that can effectively represent this data. Finally, we constructed the bar chart using individual triangles drawn sequentially for each song.

Complex Interactivity : Implementing complex interactive features poses significant challenges. For example, we want to change the color of all triangles that correspond to an artist's songs upon hovering over their image. This functionality is facilitated through a `CircularImages` component equipped with an `onArtistHover` prop. This prop is designed to manage mouse hover events and it takes a `setHoveredArtist` function provided by the `useState` hook to dynamically update the `hoveredArtist` state.When the mouse hovers over an artist's image in the `CircularImages` component, `onArtistHover` is called internally (triggering `setHoveredArtist`) and the artist's name is passed as a parameter. Upon exiting the hover state, it passes `null` to reset the state. The `setHoveredArtist` function updates the `hoveredArtist` state accordingly. If `hoveredArtist` matches the artist of a triangle's song, the triangle's color darkens.

5.4 Genres

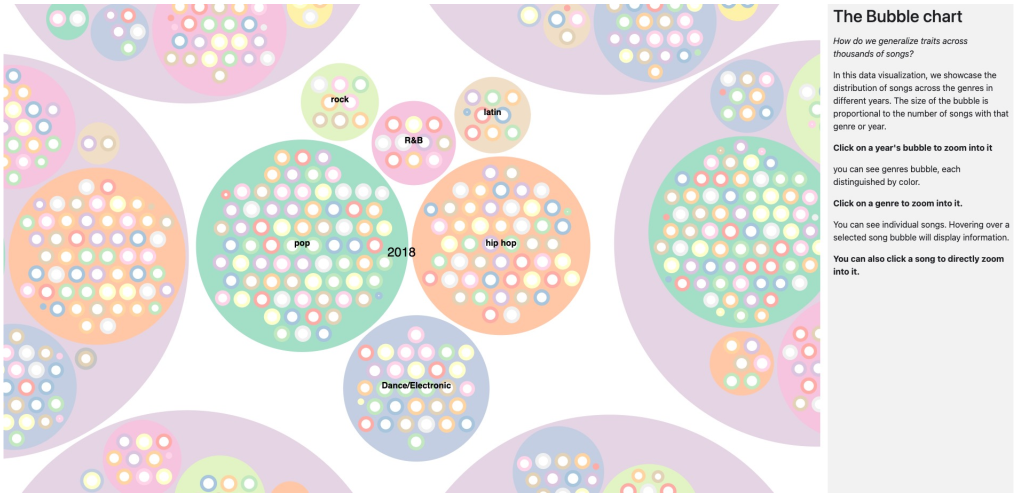
Objective

The purpose of this page is to visualize the distribution of songs across various genres over different years using an interactive bubble chart. This visualization aims to help users understand how genres and song popularity have evolved over time.

Page Layout

Visualization Area : This part contains an interactive bubble chart that allows users to zoom in and out of different hierarchical levels, from years to genres to individual songs. Hovering over a bubble displays a tooltip with detailed information.

Information Panel : This section provides explanations and interaction guides for the visualization, including headings and paragraphs that guide users on how to interact with the chart.



The Bubble chart

How do we generalize traits across thousands of songs?

In this data visualization, we showcase the distribution of songs across the genres in different years. The size of the bubble is proportional to the number of songs with that genre or year.

Click on a year's bubble to zoom into it

you can see genres bubble, each distinguished by color.

Click on a genre to zoom into it.

You can see individual songs. Hovering over a selected song bubble will display information.

You can also click a song to directly zoom into it.

FIGURE 6 – Bubble Chart : Users can Zoom in to see information in different levels, hovering over a song presents the detailed information. The side bar provides navigation.

Challenges and Design Decisions

Data Transformation : Raw song data needed to be transformed into a hierarchical structure suitable for D3.js visualization, grouping songs by year and then by genre.

Aesthetic Design : Handling color levels was a challenge due to the many hierarchical levels. Completely opaque colors would obscure information, while the large number of songs complicated color selection. The solution involved using semi-transparent bubbles with a soft color scheme to maintain visual clarity and ensure clear differentiation between genres. The layout design emphasized the visualization itself, supplemented by an information panel to avoid overwhelming the user.

5.5 Features

Objective

The objective of this page is to provide a comprehensive visualization of various musical features over the years. Displayed attributes include normalized values of features such as danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, and tempo. Implemented using ReactApexChart.

Page Layout

The webpage is divided into two main sections, designed to offer a detailed view of musical features and their changes over time :

Heatmap Chart : The primary section of the page displays a heatmap chart where each cell represents the average value of a specific musical feature for a given year. The color of the cell indicates the value of the feature, with a gradient from low (light color) to high (dark color). This allows users to quickly identify trends and patterns in the data. Users can click on the labels of the y-axis to view a detailed line chart for the selected feature.

Line Chart : When a user clicks on a feature label in the heatmap chart, a line chart is displayed below the heatmap. This line chart shows the yearly trend for the selected feature, providing a more granular view of how the feature has evolved over time.

Challenges and Design Decisions

Data Aggregation : Data is grouped by year and averaged for each feature. The values are then normalized to highlight differences, as the original values had minimal variation, making it difficult to distinguish differences on the heatmap.

Interactive Design : To enhance user engagement, clickable y-axis labels were implemented, which required adding event listeners and managing state changes to render the appropriate line chart.

6 Future work

Integration of Additional Data Sources : To enrich the presentation of our visualizations, we can integrate additional data sources, including lyrics, to enable deeper analyses

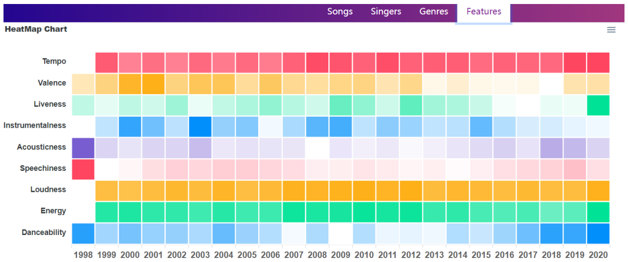


FIGURE 7 – Heatmap : Users can view average feature values for each year and click on feature labels to see detailed trends.

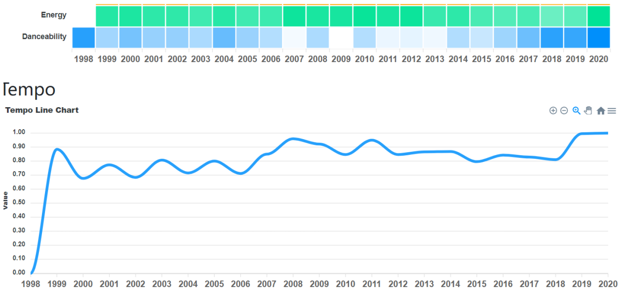


FIGURE 8 – Feature Trend Line Chart : Users can select a feature from the heatmap to display its yearly trend in a line chart for easy comparison.

such as sentiment and word frequency analysis of the lyrics. Additionally, by incorporating other data sources, we could implement audio playback features or display album artwork directly on the webpage. This would allow users to experience the evolution of music styles not only visually but also through auditory engagement.

Advanced Analytical Features : We can employ more sophisticated analytical tools, such as predictive analytics that forecast future trends, and machine learning models to identify underlying patterns and anomalies in music preferences. This will enable us to not only reflect past patterns but also offer predictions about upcoming trends, providing valuable insights for songwriters and artists.

Enhanced Interactivity : Continuing to enhance the interactivity of our visualizations remains a key focus. Future iterations will include the integration of more dynamic elements such as draggable components, zoom capabilities, and customizable filters. These features will allow users to tailor the data display according to their specific interests. For instance, users will have the flexibility to select artists of their interest for comparison, rather than being limited to pre-selected choices.

7 Peer assessement

Hanwen Zhang :

- Global structure
- Song's Page
- Feature's Page

Heling Shi :

- Web app setup
- Global structure and layout (Welcome page, navigation bar)
- Singer's Page

Jingren Tang :

- Genres' page
- Screen cast

8 Conclusion

This project created an interactive web application to visualize the evolution of popular music from 2000 to 2019. By examining musical features, genre changes, and artist dominance, key trends and shifts in popular music were highlighted.

The project addressed how musical genres evolved, which artists dominated different years, and what characteristics made songs popular. Visualizations like the Song Matrix, Bar Chart, Radar Chart, Bubble Chart, and Heatmap provide an intuitive way to explore and understand the data.

Challenges in data representation, interactivity, and design were managed using tools like Node.js, TypeScript, Vite, React, and Jupyter Notebook. These tools helped ensure efficient data processing and a smooth user experience.

Future enhancements could include integrating additional data sources such as lyrics and album artwork, using predictive analytics and machine learning models, and adding features like draggable components, zoom capabilities, and customizable filters.

Overall, this project lays a solid foundation for visualizing popular music's evolution, with exciting potential for further development and deeper insights into the music industry.