0. Choosing a dataset

Outline:

I. Understanding the data

- Dataset: Billboard Hot-100 Songs 2000-2018
- The dataset I chose is from data.world and the data is collected by user @typhon. It includes information on Billboard Hot-100 songs from 2000-2018 and corresponding Spotify data of each song.
- It's a dataset under public domain with an open license.
- Data reconciliation is handled by OpenRefine.

II. Prelimination:

- The raw data has 7573 rows and 31 columns. However, it also contains a great amount of unknown or missing values. Therefore, it still needs cleaning.
- Data Prelimination
- Trend Prelimination
- Relational ER Diagram

III. Ideas of exploring dataset

- Exploring trend of popular songs by analysing genres
- Analysing the correlations between specific genres and its Spotify values(like loudness, energy, acoustiveness etc.)
- Predicting the relationship between lyrics and genres
- Predicting the genres and Spotify values of future popular songs

1. Plan Final Deliverables

Firstly, I checked the dataset structure. My dataset did not contain much reusable textual information, however, only artist name and broad_genre columns could be

enriched with reconciliation in OpenRefine. Most columns contain numeric values or links, which cannot be reconciled or linked. Therefore, I planned to develop a shiny app for data visualization.

2. Shiny App with RStudio

2.1 Packages and Dataset

Packages installed for Shiny app

- shiny
- shinydashboard
- shinydashboard
- ggplot2
- tidyverse
- shinythemes
- plotly
- wordcloud2
- dashboardthemes
- flexdashboard

Dataset

- new.csv: I exported the original dataset in OpenRefine as a csv file.
- artist.csv: I wrote a new csv file based on the new.csv by extracting the information of all artists and their corresponding wiki_uri for later hyperlink embedding.

2.2 Interface Design

I originally planned 3 pages for displaying, and each page would include various graphs.

- Overall Information Visualization Page
    - Dataset overview
    - Popularity of different genres developed by time
    - Wordcloud of top 100 popular artists
- Individual Artist Information Page
    - Search box for selecting artist
    - Information box with linking artists' wikidata page
    - Average musical information (tempo, energy...)

    - Table of songs that were on billboard before
    - Bubble chart which illustrate each song's popularity and genre
- Prediction Page
    - Slidebar for user interact with prediction model
        - Users could select values for different musical variables and the model will predict the ranking (peak ranking) of this "song".

2.3 Approaches

2.3.1 Overall Information Visualization Page

1. Review the summary of the original dataset by grouping rows by year
- I convert the original date column as date format and add a new column called year by extracting year from it.
- I group the data by year and count there are how many records (rows) for each year. Surprisingly, I found the number of records varies a lot for each year (range from 249 for 2001 to 626 for 2017). This means I need to normalize data while I do visualization to avoid biased graphs.

1. Dataset overview
- Steps: I created 4 boxes which show the general information of my dataset (Number of records, artists, songs, genres) to give users an introduction for my dataset.
    - Notes: Comparing the other 3 boxes, Artist one is different and it's more complicated, since I split the artist column, which contains multiple artists' names, into 8 columns in Homework 3.

2. Popularity of different genres developed by time
- Notes: I noticed that there is no genre information for records in 2018, therefore, I filter them out and only create graphs based on data from 2000 to 2017.
- Steps: Firstly, I filter out rows that still do not have broad_genre information. Group data by year and broad_genre, and count the occurrences of different genres in each year. I normalized the popularity percentage by dividing the number of occurrences for certain genres by the number of records in that year. Finally, plot line graphs based on it. Popularity on y axis shows as percentage, which is the percentage of genres for songs on the Billboard (higher means more popular).

3. Wordcloud of top 100 popular artists

- Notes: Most of the artists only appeared once on the graph. It's unrealistic to make a Wordcloud for all artists. Therefore I chose to display top 100 artists after several times of testing.
- Steps: Unlist 8 artist columns and count the frequency for each one. Then create Wordcoud based on it.
- Findings: It's interesting that many artists would collaborate with others, and most of the time, they are not the main artist for this song. Some artists don't have trending songs as main singers in the rank, however, they contributed to many hot songs.

2.3.2 Individual Artist Information Page

1. Create a new csv file for this pages
- A new csv file is necessary because I reconciled multiple artist columns. Each artist would have a different wikidata link, so I cannot store their wikidata information in one row.
- Steps: In OpenRefine, I extract the artists' wikidata number and create corresponding new columns (e.g artist1 & wikidata_uri1), which convert their wikidata number to hyperlinks by adding prefix and domain of the website. This dataset is saved as artist.csv.

1. Search box for selecting artist
- Notes: During searching related work, I found there are some similar projects. However, most of these projects require users to enter specific artist names, and it's definitely not friendly for users. (Imagine you have to enter "The Beatles" so you could successfully find information. It won't work if your input is "beatles", "the Beatles" or "Beatles".) Therefore, I designed the artist input as a box that users could either select values from dropdown menus or type names to search. It could partially match the artist names and display potential ones.
- This search is the most significant input on this page. Graphs and links would change based on the artist names.

2. Information box with linking artists' wikidata page
- Notes: I tried different methods for extracting corresponding wikidata_uri and making a dynamic hyperlink in Shiny App. However, none of those worked, since Shiny requires quotes around hyperlinks and it's hard to achieve in Shiny. Therefore, I went back and changed the artist.csv file by adding single quotes around the uris.
- Steps: Created a dynamic information box. It displays the artist name and link for the wikidata page.

3. Average musical information (tempo, energy...)

- Notes: There are 13 musical information columns, and I chose 8 more important columns to create gauge graphs. Notice these values also need to be normalized.
- Steps: Calculate 8 columns average values for user selected artists. Normalize the average values by (average-min(colName))/(max(colName)-min(colName)). Output would be percentage values. Round it to 1 decimal place and display it on a gauge graph.

4. Table of songs that were on billboard before

- Notes: Again, for some artist who has a great number of songs ranked, I designed this box to be collapsible.
- Steps: Create a table which displays all the ranked songs with their date, title, and rank.

5. Bubble chart which illustrate each song's popularity and genre

- Notes: In this graph, I include 4 variables, which are rank, broad_genre, time, and peak_pos of each song. However, I found it's tricky for displaying since the higher ranking (smaller number) would result in lower places on the graph and smaller size as bubbles. Therefore, I substitute rank and peak_pos as popularity and peak_popularity, which are (101-origianal_value). In this case, a song would show with higher yaxis value and larger size when it's ranked higher.
- Steps: Create a bubble graph for certain artist's all ranked songs.
    - xaxis: time: Year & Month
    - yaxis: Popularity: 101 - rank
    - size: peak_popularity: 101 - peak_pos
    - color: broad_genre

2.3.3 Prediction Page

1. Create model for prediction

- In the R Markdown File I created, I used linear regression for fitting the model to musical variables. After selecting statistically significant variables, I'm not satisfied with the result of model prediction after running testing on it. In the final model I made, only 3% of the rank value could be explained by my variables.

1. Future Improvement

- Poor result of linear regression model indicates following possibilities:
    - Linear regression is not a suitable model for this dataset
    - Making models by adding variables is not promising
    - There is weak correlation between musical information and rank values
- Potential Solutions
    - Try different types of regression models

- Test with various combination of variables
- Run test to check if there is strong correlation between rank value and artist (Assume certain artist would be popular regardless of the musical information influence)