Project Report

*Introduction*

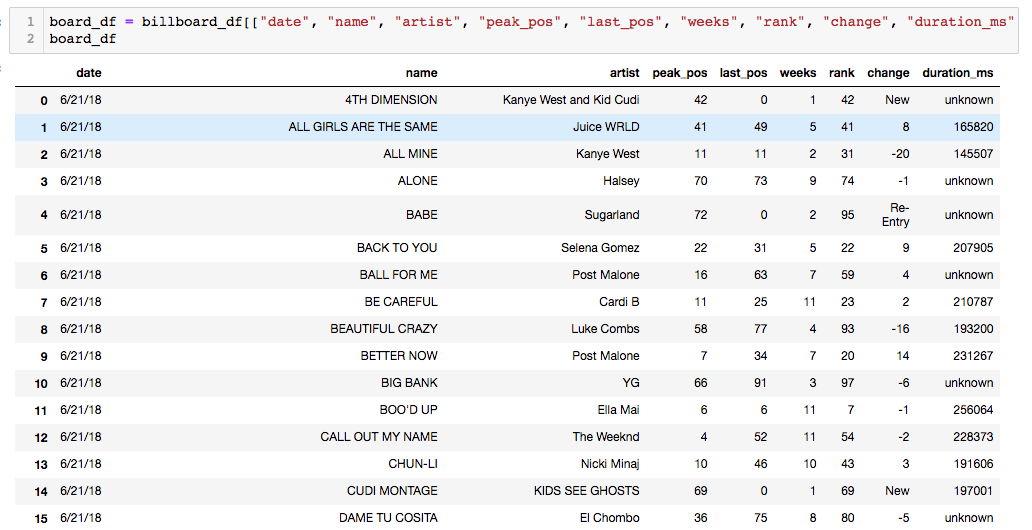
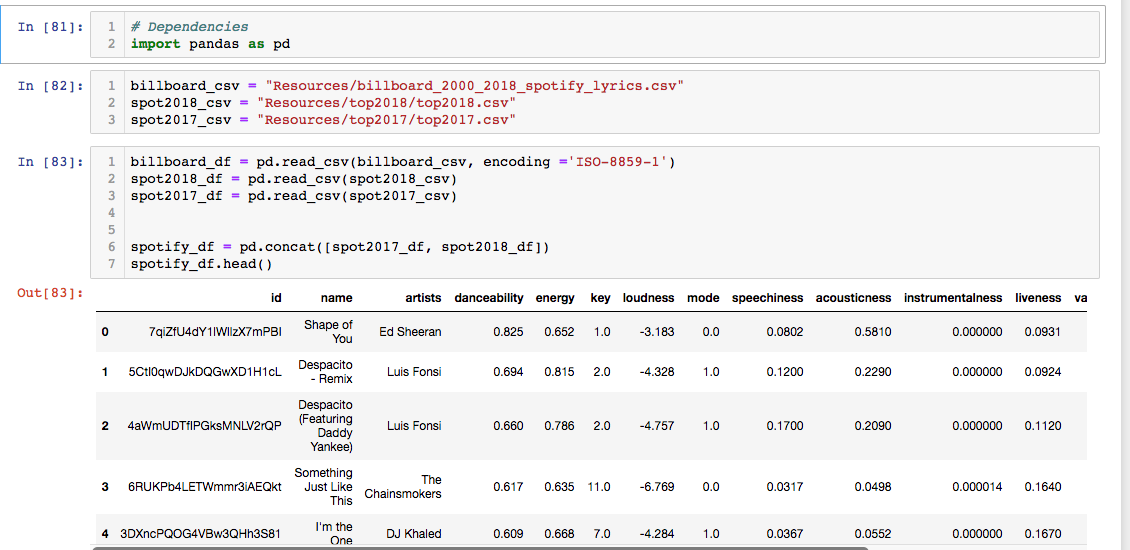


*Figure 1: Spotify and Billboard logos - both are aggregate data on the most popular songs within the US*

Our project involved the creation of a dataset listing top spotify tracks that have also featured on the 100 Hottest Billboard song list (Fig. 1). From combining this data, we wanted to create a larger, more detailed dataset of popular music in 2017-2018.

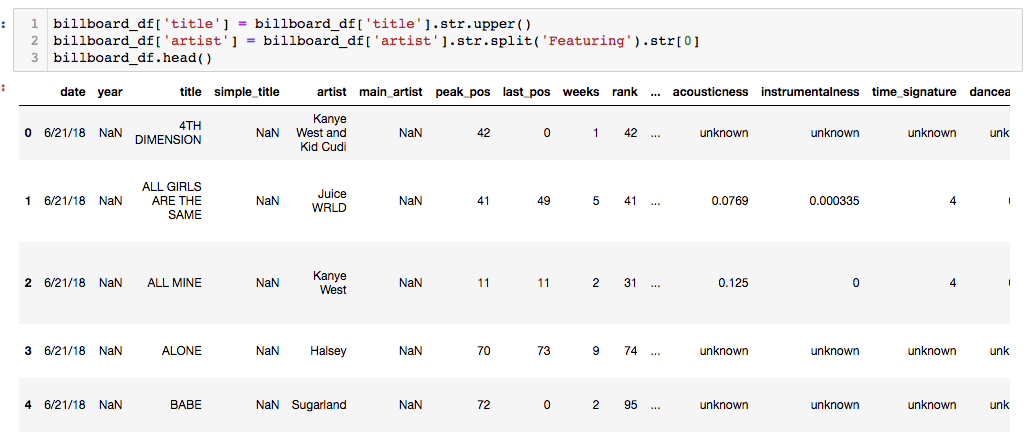
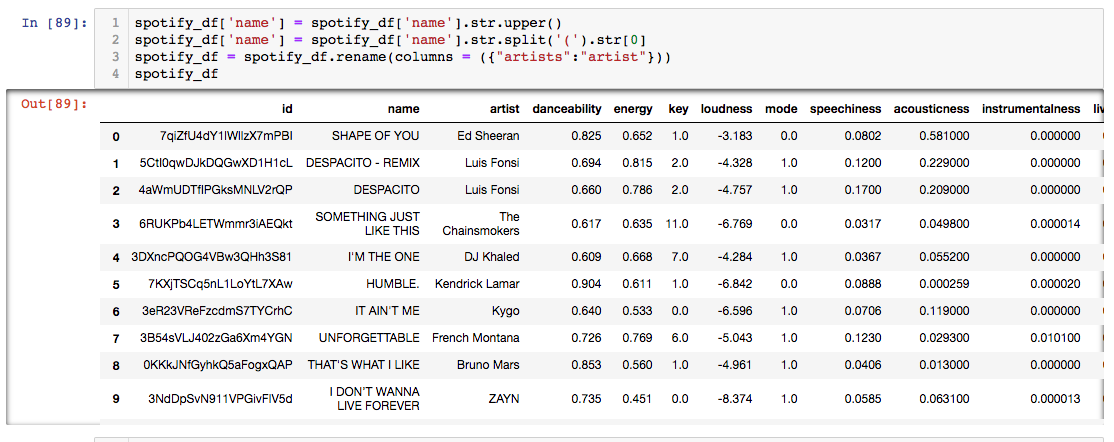
*Method and Results*

In order to appropriately build up our database, we first worked on extracting music datasets from our online sources. We downloaded data on the [top spotify tracks of 2017 and 2018](https://www.kaggle.com/nadintamer/top-spotify-tracks-of-2018) from *Kaggle* and data on the [100 hottest billboard in 2000-2018](https://data.world/typhon/billboard-hot-100-songs-2000-2018-w-spotify-data-lyrics/workspace/file?filename=billboard_2000_2018_spotify_lyrics.csv) from *Data.world*. We focused on this data because we saw matching columns that would hopefully give us the opportunity to accurately and easily combine the data within one table. Using the data collected, we then wrote out python code within a jupyter notebook in order to clean the data for better processing (Fig. 2). Our code used the Pandas library to not only view and better analyse the data but also to clean and join the two CSV datasets.



*Figure 2: Importing both the Spotify and Billboard CSVs to create dataframes for data cleaning*

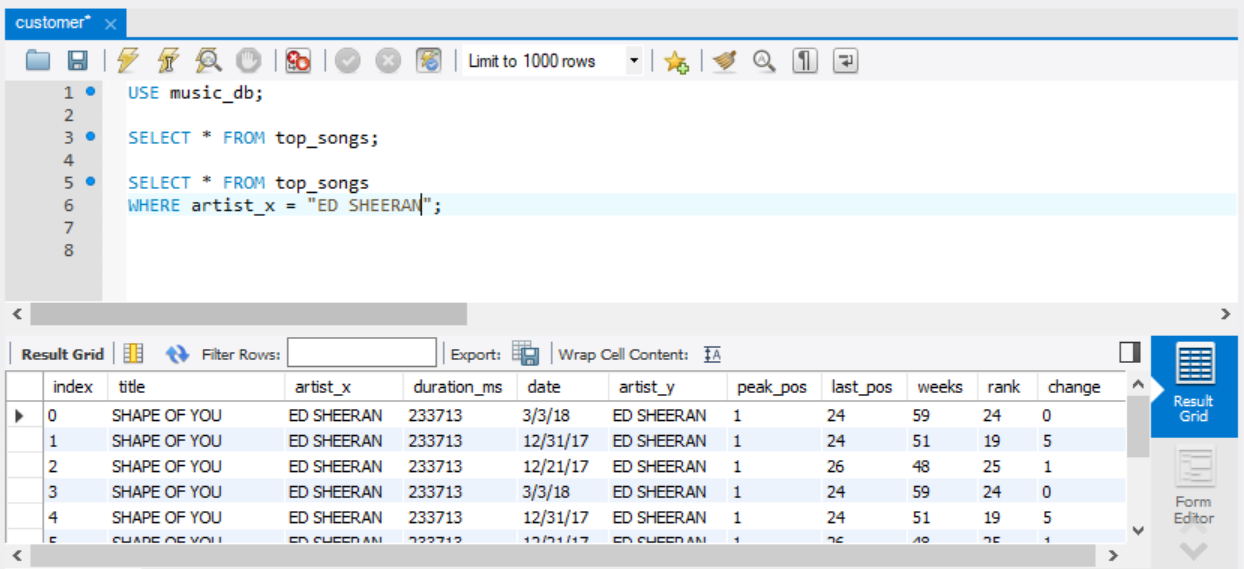
During the cleaning process, we worked on removing any columns that were lacking information or didn’t contribute any relevant information. We also made sure to remove any redundant columns that would bloat our dataset without adding value to future researchers. For example, within the Billboard dataset, we dropped the year, simple\_title, main\_artist, video\_link, broad\_genre and lyrics columns.



*Figure 3: Analysis of the data showed inconsistent text around “featuring” artists. We formatted removed the featured artists to make the data more consistent for future joins*

As we progressed towards joining our datasets, it became clear that the columns we were counting on for our join were not as helpful as we would have liked. Our initial goal was to join the datasets using song titles and artist names but we found that there were inconsistent “featuring artists” text formats. Knowing that these would be our main joining columns, we needed to either make each format consistent or remove them entirely. We decided to try removing all mention of “featuring artists” by splitting the text whenever an open parenthesis appeared, e.g. ‘(featuring..’ (Fig. 3). We thought this would be a good strategy for creating a safe join but after further analysis of the updated dataset, we realised that we couldn’t guarantee that every instance of ‘featuring’ was correctly altered especially when we found an error while glancing through the updated dataframe.

Following our disappointment with song title and artists, we decided to look into using the song duration as our safe bet for joining our datasets. We agreed on duration as our column of choice since it was a more specific marker for each song than song title and artist name. A large number of the Billboard entries were missing a lot of entries in multiple columns. We decided to clean out the data further by removing these rows as they did not add additional value to the larger goal. Once our dataset was completely updated and cleaned, we created a SQL database called *music\_db* and worked on loading the data into tables within the database. Initially we had trouble loading our data into the SQL database due to an encoding issue only to find out that we needed to encode our connection in latin1.

**

*Figure 4: SQL queries to test and confirm that data was correctly loaded into our SQL database*

With our connection was established and our data loaded, we initiated basic queries in order to confirm that the data was correctly loaded into the SQL database (Fig. 4). Our decision to use a relational, SQL database hinged on the fact that we had a clear understanding of what data we had and that weren't worried about any additional/missing columns. We also thought that a SQL table would give future explorers a lot of options in terms of analysing the database we had collated.

*Discussion*

As we processed an combined our data, there were a few issues that we needed to resolve in order to ensure we created a new reliable data source for popular music. The most

important of these issues was ensuring that we joined the datasets in a meaningful way without removing to much of the information found in each source.

We initially tried joining the data using the Spotify URL/ID since that should have been consistent for both CSVs but we found that the larger Billboard dataset had a lot of songs that were missing this information. The team moved on to joining using the artist name but then ran into the issue of inconsistent featuring artist conventions. We went around this issue by entirely removing the mentioned featuring artists as described within our methods. Following the missing and inconsistent data we had observed, it was then important that we join using consistent data to ensure that the correct song data was matched throughout the two CSVs. We decided to complete our join using song duration and cutting out a lot of rows missing data in order to create a complete and informative dataset.

Overall, there was a lot of preparation and analysis required before our datasets were ready for joining and loading. Our team took the time to ensure that we fully understood the data we were working with so that we could create a larger enhanced dataset that could be used by future researchers/explorers.