Comparing Song Audio Features to Rankings on The Billboard Hot 100

Kevin Carr 501150122
Supervisor: TBA
Date: September 19, 2022

Ryerson University

Table of Contents

Abstract **3**
Project Description 3
Data Sources 3
Tools and Techniques 4
Data Mining 4
Predictive Analytics 5
References **6**
Attachments **7**

Abstract

Project Description

Music streaming services employ data models to characterise audio features for songs, and use this data to recommend songs and playlist to their listeners. This data is provided and publicly available for multiple streaming services, notably Spotify (Spotify, n.d.).

The Billboard Hot 100 has been a music industry standard for approximately 70 years (Wikipedia, 2022). The Billboard Hot 100 contains weekly rankings for songs which are based on sales, plays, and surveys.

The central problem in this project is to utilise the above sources to predict the popularity of a song using audio features. This technique could potentially predict future trends in music or the performance of an individual song. These predictions could be useful to musicians or producers attempting to optimise success, or listeners looking for something new.

Data Sources

The data gathered for this project have been taken from multiple sources and combined. Data was found using the Google dataset search engine (Google, n.d.).

Three of the relevant sources were found on Kaggle.com, a popular online data science community where users can share datasets (Dhruvil Dave, 2021; Malte Grosse, 2022; Rodolfo Figueroa, 2020). Audio features from the large datasets were matched with the list of songs from the Billboard Hot 100.

Not all songs from the Billboard Hot 100 list were found in the audio feature databases obtained from Kaggle.com. Audio features for these missing songs were searched using the Spotify API (Spotify, n.d.). Audio features from songs matched using Spotify API were populated into the project dataset. After combining the datasets, audio features were found for approximately 75% of songs in the Billboard Hot 100 list (approximately 22 thousand of the 30 thousand songs to have appeared in the Billboard Hot 100). In addition, audio features were gathered for approximately 10 million songs.

Tools and Techniques

For this projects, two main techniques will be employed, namely data mining and predictive analytics. The bulk of the analysis will be performed use Python. SQLite will be used in limited cases to query the 8.7 million song database obtained from Kaggle.com (Malte Grosse, 2022). Modules for Python will be utilised on an as needed basis, depending on the analytical techniques being employed.

Data Mining

First, data mining and knowledge discovery will be used to explore the data, cluster audio features, and determine correlations between audio features. The following questions will be considered:

- Which audio features correlate with each other?
- Do popular songs (as identified in the Billboard Hot 100) correspond to correlated combinations of features? Or do popular songs include more unique characteristics?

- Does clustering songs using audio feature lead to any meaningful groupings or insights (for example musical genre)?
- How have these correlations and features changed over time?

For this analysis we will employ unsupervised machine learning techniques such as K-Means Clustering. We will also perform a correlation analysis between features. Additional exploratory data analysis will be performed depending on the insights gained from the clustering and correlation analyses.

Predictive Analytics

Secondly, predictive analytics will be used to attempt to build a predictive model using the data. We will consider the following questions:

- Can we predict performance on the Billboard Hot 100 using audio features?
- Can we predict time series performance using audio features and clusters identified in the data mining portion of the study? I.e., can we predict music trends?
- Can we identify any clusters of audio features that appear popular, but haven't yet made in onto the Billboard Hot 100? I.e., can we predict possible future trends?

For this analysis we will use supervised classification algorithms such as Naïve Bayes or K-Nearest Neighbours. We will evaluate multiple models and compare statistical significance and predictive power between the algorithms.

References

Spotify for Developers. (n.d.). https://developer.spotify.com/documentation/web-api/

Billboard Hot 100. (2022, September 6). Wikipedia. https://en.wikipedia.org/w/index.php?title=Billboard_Hot_100&oldid=1108834581

Google Dataset Search. (n.d.). https://datasetsearch.research.google.com/

Dhruvil Dave. (2021, November 9). Billboard "The Hot 100" Songs [Data set]. Kaggle. https://doi.org/10.34740/KAGGLE/DS/1211465

Malte Grosse. (2022, March 23). 8+ M. Spotify Tracks, Genre, Audio Features [Data set]. Kaggle. https://www.kaggle.com/datasets/maltegrosse/8-m-spotify-tracks-genre-audio-features/

Rodolfo Figueroa. (2020, December 22). Spotify 1.2M+ Songs [Data set]. Kaggle. https://www.kaggle.com/datasets/rodolfofigueroa/spotify-12m-songs

Attachments

Dataset

In [1]:
```
# import modules
import pandas as pd
import numpy as np
import spotipy

# jupyter notebook full-width display
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:100% !important; }</style>"))

# pandas formatting
pd.set_option('display.float_format', '{:.2f}'.format)
# NOTE: underscore separaters ('_') are better than commas (',') because
# numbers with underscores work in Python without any extra effort.
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', 200)
```

Data Sources

The Billboard 100

https://en.wikipedia.org/wiki/Billboard_Hot_100 (https://en.wikipedia.org/wiki/Billboard_Hot_100)

https://www.kaggle.com/datasets/dhruvildave/billboard-the-hot-100-songs (https://www.kaggle.com/datasets/dhruvildave/billboard-the-hot-100-songs)

1.2M Songs with Metadata (csv)

https://www.kaggle.com/datasets/rodolfofigueroa/spotify-12m-songs (https://www.kaggle.com/datasets/rodolfofigueroa/spotify-12m-songs)

8+ M. Spotify Tracks, Genre, Audio Features (SQL)

https://www.kaggle.com/datasets/maltegrosse/8-m-spotify-tracks-genre-audio-features (https://www.kaggle.com/datasets/maltegrosse/8-m-spotify-tracks-genre-audio-features)

Spotify API

https://developer.spotify.com/documentation/web-api/ (https://developer.spotify.com/documentation/web-api/)

https://developer.spotify.com/console/get-search-item (https://developer.spotify.com/console/get-search-item)

https://developer.spotify.com/console/get-audio-features-track/ (https://developer.spotify.com/console/get-audio-features-track/)

https://developer.spotify.com/documentation/web-api/reference/#/operations/get-audio-features (https://developer.spotify.com/documentation/web-api/reference/#/operations/get-audio-features)

Spotipy Library: https://spotipy.readthedocs.io/en/master/ (https://spotipy.readthedocs.io/en/master/)

Data Description and Discussion:

- The Billboard 100 data did not include audio features. It was combined with audio features from the following sources:
  - 1.2M Songs with Metadata (csv format)
  - 8+ M. Spotify Tracks, Genre, Audio Features (SQLite format)
  - Spotify API data gathered via the library Spotipy
- Overall, audio features was gathered for approximately 75% of songs from the Billboard 100.
  - Some songs were excluded based on data repetition issues
    - Typically this was only hard to find songs with very similar names
    - For example searching for 'Metallica The Unforgiven' and 'Metallica The Unforgiven Part 2' yielded the same Spotify id
      - It was determined that excluding these songs was less error-prone than manually fixing the issues
      - Alternatively, we could have kept 1 song. In this case, there is up to a 50% chance that the song is mislabelled, so this option appeared less favourable than dropping both repeat instances.
- A Quality Assurance (QA) check was performed on the final dataset.
  - Audio features from 100 songs were gathered from the Spotify API and compared to the datasets listed above.
  - There were 3 non-trivial issues noted in 2 of the 100 songs:
    - Madonna Live To Tell
      - A significant increase in loudness (~7 dB)
      - Appoximately 1 second different in length
      - All other audio features consistent between data sources
      - Both of these changes appear to result from remastering and re-uploading the track
        - https://artists.spotify.com/help/article/re-uploading-music (https://artists.spotify.com/help/article/re-uploading-music)
    - Lil Wayne Let It All Work Out
      - The key signature was not consistent between the 2 sources
      - The newer source (the API request from Sept 11, 2022) was correct (B major)
      - The SQL database was also different
      - My supposition is that these errors are due to the characteristics of the song:
        - atonal (most notably the sining)
        - detuned (bass pitch automation, and low-fi detuning effects)
  - Overall, there is a large degree of consistency between datasets. Furthermore, inconsistencies are all explainable with reasonable suppositions.

In [2]:
```
desired_formatting = [
    'id', 'song', 'artist',
    'acousticness', 'danceability', 'duration_ms', 'energy',
    'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
    'speechiness', 'tempo', 'time_signature', 'valence'
]

desired_formatting_timeseries = [
    'date',
    'id', 'song', 'artist',
    'rank', 'last-week', 'peak-rank', 'weeks-on-board',
    'acousticness', 'danceability', 'duration_ms', 'energy',
    'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
    'speechiness', 'tempo', 'time_signature', 'valence'
]

# all songs with audio features (combined from 3 sources)
df_10M = pd.read_csv('every_song_with_data.csv')
df_10M = df_10M[desired_formatting]

# all Billboard 100 lists, audio features included where possible
df_B100 = pd.read_csv('all_audio_features_billboard_100.csv')
df_B100 = df_B100[desired_formatting_timeseries]
df_B100['date'] = pd.to_datetime(df_B100['date'])

# all unique songs from the Billboard 100 lists, audio features included where possible
df_B100_songs = pd.read_csv('all_audio_features_billboard_100_songs.csv')
df_B100_songs = df_B100_songs[desired_formatting]
```

Data Description

In [3]:
```
# sizes of the datasets
df_10M.shape, df_B100.shape, df_B100_songs.shape
```

Out[3]:
```
((9595992, 16), (329930, 21), (29681, 16))
```

In [4]:
```
df_B100.date.min(), df_B100.date.max()
```

Out[4]:
```
(Timestamp('1958-08-04 00:00:00'), Timestamp('2021-11-06 00:00:00'))
```

In [5]:
```
df_B100[['rank', 'last-week', 'peak-rank', 'weeks-on-board']].describe().loc[
    ['mean', 'std', 'min', '25%', '50%', '75%', 'max']
]
```

Out[5]:

	rank	last-week	peak-rank	weeks-on-board
mean	50.50	47.59	40.97	9.16
std	28.87	28.05	29.35	7.62
min	1.00	1.00	1.00	1.00
25%	26.00	23.00	13.00	4.00
50%	51.00	47.00	38.00	7.00
75%	76.00	72.00	65.00	13.00
max	100.00	100.00	100.00	90.00

In [6]:
```
# truncate column names so they print better
df_10M.rename(columns=lambda x: x[:4], inplace=True)
df_B100.rename(columns=lambda x: x[:4], inplace=True)
df_B100_songs.rename(columns=lambda x: x[:4], inplace=True)
```

In [7]:
```
df_10M.describe().loc[['mean', 'std', 'min', '25%', '50%', '75%', 'max']]
```

Out[7]:

	acou	danc	dura	ener	inst	key	live	loud	mode	spee	temp	time	vale
mean	0.42	0.53	238209.59	0.54	0.26	5.24	0.21	-10.89	0.66	0.10	118.56	3.84	0.48
std	0.38	0.19	159341.59	0.28	0.37	3.54	0.18	6.36	0.47	0.14	31.03	0.57	0.28
min	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-60.00	0.00	0.00	0.00	0.00	0.00
25%	0.03	0.40	169600.00	0.31	0.00	2.00	0.10	-13.68	0.00	0.04	95.08	4.00	0.23
50%	0.34	0.55	216933.00	0.57	0.00	5.00	0.13	-9.20	1.00	0.05	118.95	4.00	0.47
75%	0.82	0.68	275080.00	0.79	0.64	8.00	0.26	-6.40	1.00	0.08	137.45	4.00	0.71
max	1.00	1.00	19672058.00	1.00	1.00	11.00	1.00	7.23	1.00	0.97	249.99	5.00	1.00

In [8]:
```
df_B100[['acou', 'danc', 'dura', 'ener', 'inst', 'key', 'live', 'loud',
    'mode', 'spee', 'temp', 'time', 'vale']].describe().loc[['mean',
    'std', 'min', '25%', '50%', '75%', 'max']]
```

Out[8]:

	acou	danc	dura	ener	inst	key	live	loud	mode	spee	temp	time	vale
mean	0.28	0.60	226879.65	0.63	0.03	5.22	0.19	-8.61	0.73	0.06	120.40	3.94	0.61
std	0.27	0.15	66552.15	0.20	0.14	3.56	0.16	3.59	0.44	0.07	27.79	0.30	0.24
min	0.00	0.00	30213.00	0.01	0.00	0.00	0.01	-30.35	0.00	0.00	0.00	0.00	0.00
25%	0.04	0.51	183360.00	0.48	0.00	2.00	0.09	-10.97	0.00	0.03	99.93	4.00	0.42
50%	0.18	0.61	221306.00	0.64	0.00	5.00	0.13	-8.15	1.00	0.04	119.00	4.00	0.63
75%	0.47	0.71	258399.00	0.79	0.00	8.00	0.24	-5.79	1.00	0.06	136.00	4.00	0.81
max	1.00	0.99	1561133.00	1.00	0.99	11.00	1.00	2.29	1.00	0.95	241.01	5.00	0.99

In [9]:
```
df_B100_songs.describe().loc[['mean', 'std', 'min', '25%', '50%', '75%', 'max']]
```

Out[9]:

	acou	danc	dura	ener	inst	key	live	loud	mode	spee	temp	time	vale
mean	0.32	0.59	217638.34	0.61	0.04	5.20	0.20	-8.93	0.74	0.07	120.48	3.92	0.61
std	0.29	0.15	68403.26	0.20	0.15	3.56	0.17	3.62	0.44	0.08	28.09	0.33	0.24
min	0.00	0.00	30213.00	0.01	0.00	0.00	0.01	-30.35	0.00	0.00	0.00	0.00	0.00
25%	0.05	0.49	169533.00	0.46	0.00	2.00	0.09	-11.31	0.00	0.03	99.79	4.00	0.42
50%	0.22	0.60	210426.00	0.62	0.00	5.00	0.13	-8.55	1.00	0.04	119.07	4.00	0.64
75%	0.56	0.70	251333.00	0.77	0.00	8.00	0.25	-6.11	1.00	0.06	136.34	4.00	0.81
max	1.00	0.99	1561133.00	1.00	0.99	11.00	1.00	2.29	1.00	0.95	241.01	5.00	0.99

Proportion of Songs With Audio Feature Data:

~75% of songs on the Billboard list are available on Spotify, and weren't removed for data errors

In [10]:
```
# All Billboard 100 lists
# number not null, total, proportion not null
(
    df_B100[df_B100.id.notnull()].shape[0],
    df_B100.shape[0],
    df_B100[df_B100.id.notnull()].shape[0] / df_B100.shape[0]
)
```

Out[10]:
```
(253254, 329930, 0.7675991877064832)
```

In [11]:
```
# All songs from Billboard 100 lists
# number not null, total, proportion not null
(
    df_B100_songs[df_B100_songs.id.notnull()].shape[0],
    df_B100_songs.shape[0],
    df_B100_songs[df_B100_songs.id.notnull()].shape[0] / df_B100_songs.shape[0]
)
```

Out[11]:
```
(22189, 29681, 0.7475826286176341)
```