Final_EDA

November 20, 2019

By: Jack Swisher, Dennis Lin, and Sam Benkelman

0.1 Description of our EDA

For our EDA, we first got data for the songs within each playlist using the Spotify API. The information that we derived from this is described below. We anticipate using the Spotify API further later in our project to get the top playlists to further evaluate the performance of our model.

We do three types of EDA below: * EDA of information about the playlists (duration of playlist, number of songs from a single artist, number of songs in a playlist)

- EDA of audio features for songs, clustered by playlist
- EDA of audio features for general songs

0.2 Description of Data for Songs

From the spotify API, we were able to get the following audio features for each track. Below, we have listed them with a description of what they represent:

- **DANCEABILITY:** Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.
- **ENERGY:** Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy. For example, death metal has high energy, while a Bach prelude scores low on the scale.
- **KEY:** The estimated overall key of the track. Integers map to pitches using standard Pitch Class notation . E.g. 0 = C, 1 = C/D, 2 = D, and so on. If no key was detected, the value is -1.
- **LOUDNESS:** The overall loudness of a track in decibels (dB) ranging from -60 to 0. Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks.
- **MODE:** Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
- **SPEECHINESS:** Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks

- **ACOUSTICNESS:** A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
- **INSTRUMENTALNESS:** Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
- **LIVENESS:** Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. Values range from 0 to 1 with a value above 0.8 providing strong likelihood that the track is live.
- **VALENCE:** A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).
- **TEMPO:** The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.

0.3 EDA Thought Process and Results for data in Playlists

[6]:
```
count    1000000.000000
mean          65.346428
std           53.669358
min            4.000000
25%           25.000000
50%           48.000000
75%           91.000000
max          375.000000
Name: pos, dtype: float64
```

This tells us the distribution of the number of songs in a playlist. The mean is about 65 with a standard deviation of about 54. The minimum number of songs in the playlists that we have is 4 and the maximum is 375.

[8]:
```
count    3.808821e+07
mean     1.741915e+00
std      2.747589e+00
min      1.000000e+00
25%      1.000000e+00
50%      1.000000e+00
75%      2.000000e+00
max      2.450000e+02
Name: pos, dtype: float64
```

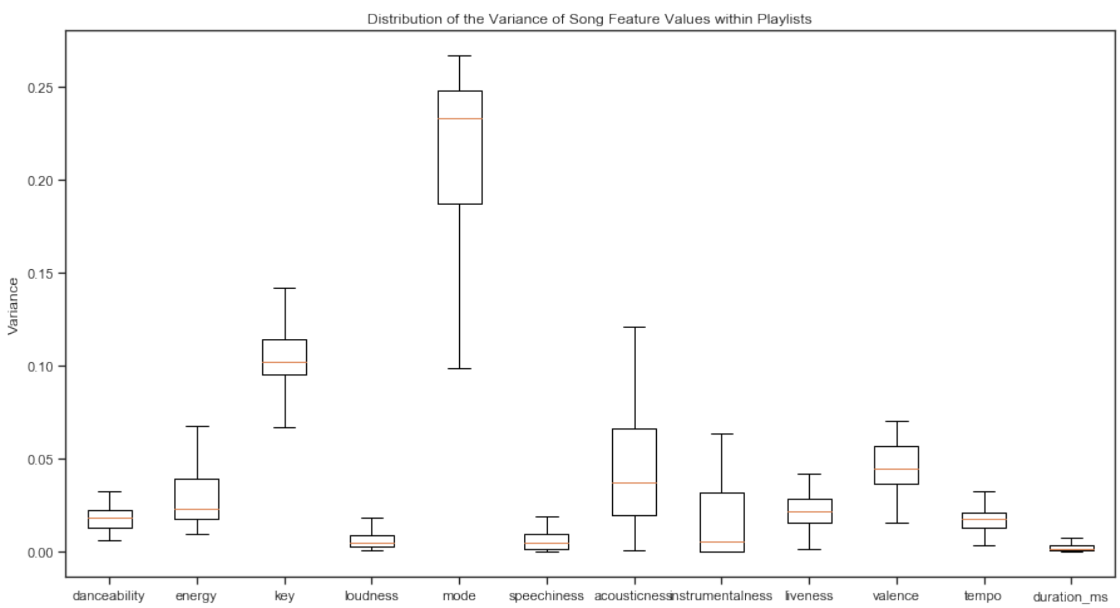
This tells us the distribution of how many songs there are in a playlist by the same artist. The mean is around 2 songs by the same artist. The minimum is 1 song and the maximum is 245 songs by the same artist, which makes sense because someone could have a playlist solely based off of one artist's songs.

[9]:
```
fid  pid
0    0      192.206900
```

```
     1       194.274500
     2       233.999300
     3       482.100967
     4        72.254700
                ...
999  995     141.702883
     996     520.781800
     997     100.252750
     998      48.493467
     999     324.189417
Name: duration_ms, Length: 1000000, dtype: float64
```

This tells us the distribution of the total duration of playlists. The max value is an outlier, which is a playlist that lasts over 10,000 minutes. At first, we thought this may have been an error in the data, but upon further inspection, the playlist just has songs that are mixes, which last a very long time (average around 26 minutes).

0.4 EDA Thought Process and Results on Songs Clustered by Playlists



We also took a look at the variance in features expressed in the songs chosen for inclusion within individual playlists. We took a subset of 50 full playlists from the Spotify API, and from this dataset, found the audio features for each song in each playlist. We then standardized the values (by each feature) so that the variance can be seen at the same scale. We then made the above boxplot, showing the distribution of the variances of each feature across the 50 playlists.

As we can see from the above boxplots, there clearly *are* some features that tend to cluster together within playlists; for instance, danceability has on average a relatively low variance across all songs in a playlist. This means, simply, that most of the songs in any one playlist will tend to have the same values for danceability. This seems intuitively to make sense: playlists are often made with or without dancing in mind, so a song that is less danceable would not fit into a playlist with songs that are very danceable. In other words, playlists may tend to select for danceability. On the other end, acousticness and mode had relatively higher variances, meaning that these features may not necessarily be selected for in particular when people create playlists. These trends from our EDA will be important to consider when moving on with the project.

0.5 EDA Thought Process and Results on Songs in General

For part of our EDA, we decided it would be a good idea to look at the songs within the playlists. We derived two subsets of songs from the playlists with the audio features described above: * One random subset of 50,000 songs * The most popular 70,000 songs (as measured by frequency of apperance in playlists)

We wanted to compare the audio features within these two subsets to compare whether or not a simple random sample of songs would have significantly different features from a subset of the most popular songs.

We did this in two ways: using summary statistics and comparing distributions.

0.6 Comparing Summary Statistics

[13]:

	danceability	energy	key	loudness	mode \
count	49996.000000	49996.000000	49996.000000	49996.000000	49996.000000
mean	0.550618	0.585032	5.269422	-9.655672	0.654912
std	0.184363	0.266107	3.565643	5.621092	0.475402
min	0.000000	0.000000	0.000000	-60.000000	0.000000
25%	0.427000	0.391000	2.000000	-11.915250	0.000000
50%	0.567000	0.623000	5.000000	-8.161000	1.000000
75%	0.688000	0.808000	8.000000	-5.821750	1.000000
max	0.987000	1.000000	11.000000	4.472000	1.000000

	speechiness	acousticness	instrumentalness	liveness \
count	49996.000000	49996.000000	49996.000000	49996.000000
mean	0.090006	0.354872	0.220637	0.208626
std	0.116390	0.354657	0.349503	0.188922
min	0.000000	0.000000	0.000000	0.000000
25%	0.035400	0.022100	0.000000	0.096000
50%	0.047200	0.218000	0.000600	0.128000
75%	0.084400	0.692000	0.421000	0.261000
max	0.964000	0.996000	1.000000	1.000000

	valence	tempo	duration_ms	time_signature
count	49996.000000	49996.000000	4.999600e+04	49996.000000
mean	0.476369	120.128909	2.478844e+05	3.880330
std	0.269302	29.914074	1.605099e+05	0.474176
min	0.000000	0.000000	2.229000e+03	0.000000
25%	0.250000	97.020250	1.842552e+05	4.000000
50%	0.467000	120.019000	2.251600e+05	4.000000
75%	0.695000	138.285000	2.782010e+05	4.000000

max	1.000000	248.733000	5.823661e+06	5.000000

[14]:

	danceability	energy	key	loudness	mode
count	69678.000000	69678.000000	69678.000000	69678.000000	69678.000000
mean	0.585016	0.641262	5.241152	-7.476997	0.667671
std	0.163043	0.221970	3.589362	3.865238	0.471052
min	0.000000	0.000000	0.000000	-60.000000	0.000000
25%	0.478000	0.493000	2.000000	-9.070000	0.000000
50%	0.594000	0.674000	5.000000	-6.612000	1.000000
75%	0.703000	0.821000	8.000000	-4.928000	1.000000
max	0.988000	0.999000	11.000000	1.586000	1.000000

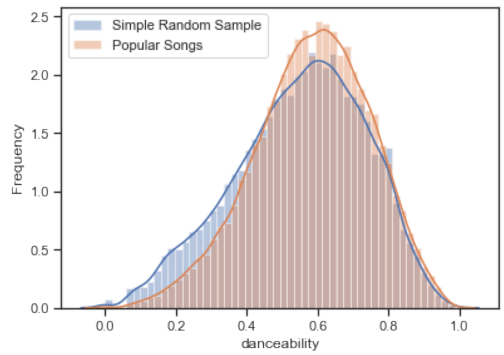
	speechiness	acousticness	instrumentalness	liveness
count	69678.000000	69678.000000	69678.000000	69678.000000
mean	0.088783	0.250946	0.071644	0.193273
std	0.095779	0.290335	0.208811	0.159902
min	0.000000	0.000000	0.000000	0.000000
25%	0.034800	0.018400	0.000000	0.096000
50%	0.048100	0.116000	0.000009	0.127000
75%	0.093200	0.416000	0.002560	0.246000
max	0.962000	0.996000	0.994000	1.000000

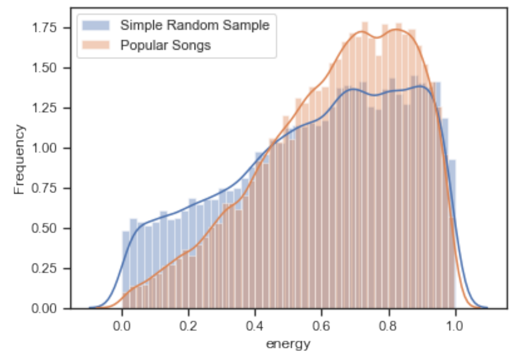
	valence	tempo	duration_ms	time_signature
count	69678.000000	69678.000000	6.967800e+04	69678.000000
mean	0.478198	121.537697	2.347612e+05	3.931270
std	0.244579	28.999635	6.936283e+04	0.352175
min	0.000000	0.000000	4.520000e+03	0.000000
25%	0.282000	98.416500	1.965600e+05	4.000000
50%	0.466000	120.732500	2.250795e+05	4.000000
75%	0.669000	140.000000	2.611330e+05	4.000000
max	0.992000	236.799000	4.195000e+06	5.000000

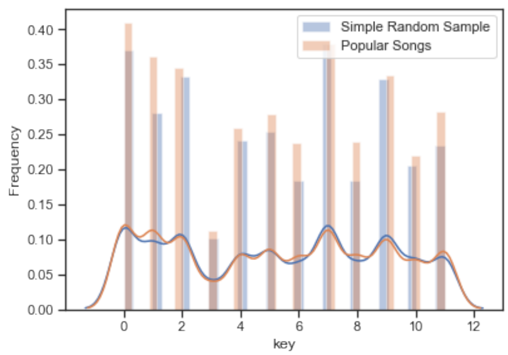
From the summary statistics, we see that the means and standard deviations for most of the features are similar with a few exceptions. Energy has a greater mean in the popular song subset. Loudness, acousticness, instrumentalness, and duration_ms have smaller means and smaller variances in the subset of popular songs.

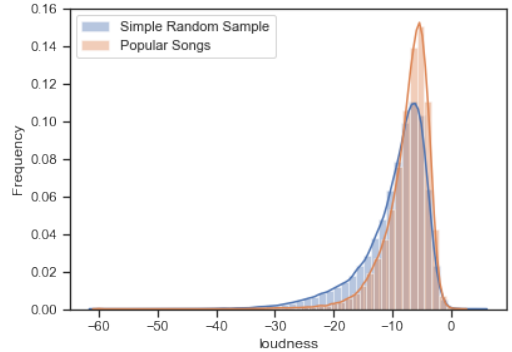
However, we cannot simply rely on summary statistics to tell us everything about the data. Therefore, we will graph the distributions of the data and compare them as well.

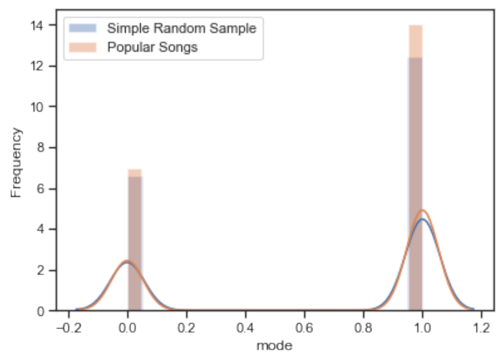
0.7 Comparing Distributions

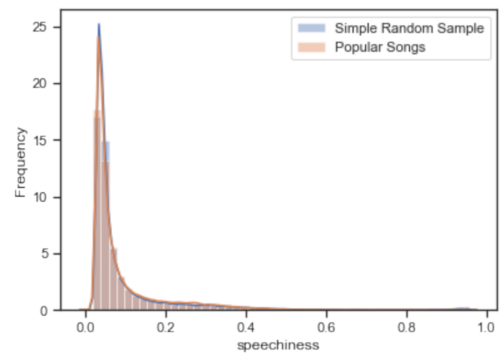


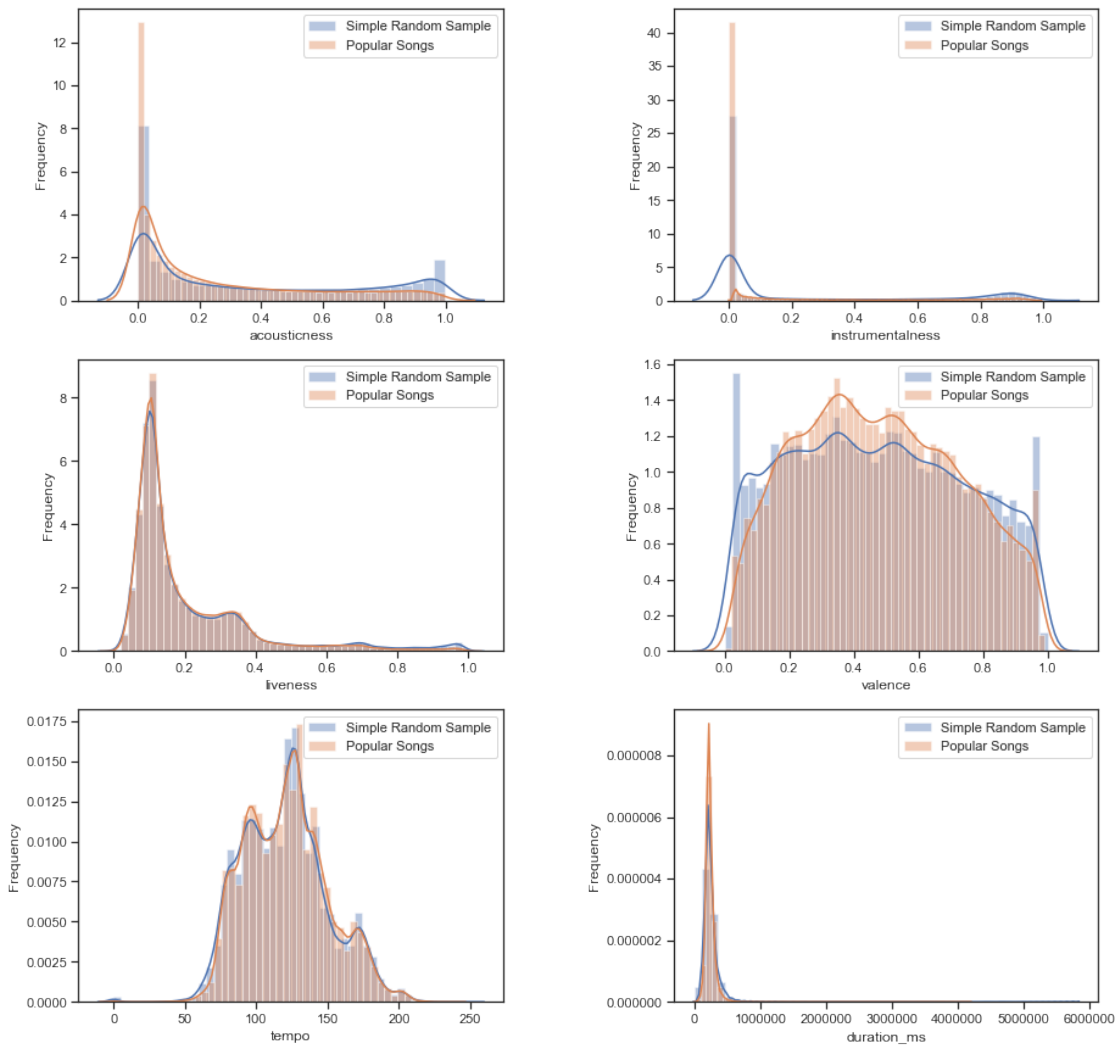












We wanted to check the distribution of audio features in both of these subsets and compare them to see whether or not the distributions of the random subset would be different from the distribution of the most popular songs of the playlists we have.

From the plots above, we see that most of the distributions are not very different. The ones that we identified that are potentially different are: loudness, acousticness, energy, duration_ms, and instrumentalness.

- In the "loudness" plot, we see greater concentration of values around 0 for the popular songs, potentially suggesting that more popular songs are louder.
- In the "acousticness" plot, we see a stronger drop off of values from 0 to 1, suggesting that the distribution of the more popular songs have less acoustic songs.
- In the "instrumentalness" plot, we see a similar steeper drop off of values from 0 to 1, suggesting that the distribution of the more popular songs have less instrumental songs as well.
- In the "energy" plot, we see that there is a greater concentration of larger values in the popular songs distribution. This suggests that more popular songs have greater energy.

- In the "duration_ms" plot, though it hard to see, there is larger spike in smaller values for the popular songs distribution. This suggests that more popular songs are shorter in duration.

More generally, we see that the audio features have different distributions of values. We have a mix of left-skewed (energy, loudness), right-skewed (liveness, instrumentalness, acousticness, speechiness, duration_ms) and normal (danceability, tempo, valence) data.

0.8 Baseline Model

0.8.1 Description / Approach

For our baseline model, we decided to recommend songs from a set of candidate songs according to which of these candidates have the highest average cosine similarity to the set of seed songs. To elaborate, the function is applied pairwise between each candidate and each seed song and then the output is averaged across the seed songs giving an average similarity for each candidate. In other words, playlists are generated using an unsupervised algorithm that attempts to maximize a score function and can generate playlists of any length up to the number of provided candidate songs. For our purposes, we used a collection of around 220,000 candidate songs, which is noteably 1/10th of all unique songs used to create playlists in our dataset.

This approach is motivated by the assumption that different playlists have different distributions of features, perhaps according to genre for example, and that songs that are tonally similar will be a good recommendation for that playlist.

This leaves us with a couple of ways to improve performance. First, our baseline model used `"danceability", "energy", "loudness", "speechiness", "acousticness", "instrumentalness", "liveness", "valence", "mode"` to measure difference, however this selection of features could certainly be tuned for model performance. Second, this similarity function could be changed completely, perhaps using a supervised ML algorithm such as a neural network where the algorithm predicts similarity between songs trained on top spotify playlists, while preserving the structure of our baseline model.

0.8.2 Evaluation

To evaluate our baseline model, we randomly selected 50 playlists and retrieved all of their songs to ensure that we had musical features on all of the playlist entries. Next, we split these playlists into a set of "seed" songs and evaluation songs of sizes 0.75 and 0.25, respectively. Finally, we used our baseline model to predict 500 songs that our model deemed as best matches. We then compared these predictions to the left-out test set and recorded the percentage of the playlist songs that were predicted correctly. Using this metric, our model predicted 2.5% of the test set correctly on average.

Furthermore, we record the positions of the correctly predicted test set where 1 is considered a better match than 500. Looking at this data, we see that there are a few playlists where the model gets many of the test set correct and other playlists where the baseline model predicted none of the original songs. This might suggest that some playlists are tonally similar while others are less so; this may motivate another angle of approach that attempts to use related artists to those found in the seed songs to avoid only suggesting tonally similar songs.

[70]:
```
{(16, 705): [],
 (29, 736): [],
 (52, 746): [],
```

```
(112, 480): [167, 318, 358, 451],
(113, 190): [],
(124, 951): [],
(149, 299): [204],
(164, 985): [96],
(167, 82): [],
(172, 599): [],
(176, 116): [],
(226, 831): [],
(246, 757): [361],
(320, 851): [134, 142],
(334, 229): [],
(341, 453): [266, 486],
(342, 824): [],
(346, 794): [],
(354, 557): [],
(356, 420): [],
(361, 820): [],
(386, 719): [],
(392, 266): [],
(420, 39): [],
(429, 375): [],
(436, 869): [],
(447, 778): [107, 157, 348, 451, 479, 485],
(451, 640): [387],
(508, 314): [],
(526, 566): [],
(560, 929): [],
(680, 108): [196, 215],
(718, 872): [],
(735, 66): [],
(759, 574): [],
(770, 309): [],
(812, 749): [],
(819, 419): [],
(823, 120): [],
(828, 766): [336],
(858, 844): [350, 386],
(859, 769): [],
(863, 519): [],
(865, 866): [],
(892, 979): [],
(897, 643): [],
(921, 300): [],
(964, 992): [],
(965, 905): [],
(998, 452): []}
```

[71]: 0.02505446623093682

0.9 Refining Our Project Questions

Our next project question, at its most fundamental, is how can we improve our baseline model? More specifically:

- What portions of our model can we have supervised learning and which portions of our model should we have unsupervised learning?
- How can we improve our evaluation method? Is there a way to incorporate user feedback - perhaps by evaluating our performance on the top playlists on Spotify?