9.5/10

I think this is a great project idea in general -- really like the idea of capitalizing on the Spotify API. My only real concern is how will you evaluate whether your clusters are any good? I suppose you may just have to do this qualitatively, but perhaps you can think of some other way (you could check, perhaps if your clusters correlate to known genres, assuming you could somehow match songs to genres automatically). Anyway, I look forward to seeing more!

Jon Zimbel
CS 4100

Final Project Proposal

I would like to work with the Spotify web API to determine the types of music that a given user likes. The API has an endpoint, audio-features, that returns a list of features for a track. An example response:

```
{
  "acousticness": 0.284,
  "analysis_url":
"https://api.spotify.com/v1/audio-analysis/4WzVh7aGOYraYtdpOscafh",
  "danceability": 0.65,
  "duration_ms": 242093,
  "energy": 0.791,
  "id": "4WzVh7aGOYraYtdpOscafh",
  "instrumentalness": 0.491,
  "key": 1,
  "liveness": 0.0668,
  "loudness": -5.775,
  "mode": 0,
  "speechiness": 0.0311,
  "tempo": 120.048,
  "time_signature": 4,
  "track_href":
"https://api.spotify.com/v1/tracks/4WzVh7aGOYraYtdpOscafh",
  "type": "audio_features",
  "uri": "spotify:track:4WzVh7aGOYraYtdpOscafh",
  "valence": 0.895
}
```

The values that are possibly useful are shown in black text. All highlighted values are on a scale from 0 to 1. Exceptions:
`"key"`: the key the track is in, in Pitch Class notation.
`"loudness"`: overall loudness of the track, in dB. Ranges from -60 (silent) to 0 (very loud).
`"mode"`: modality of the track. 0 = minor, 1 = major.
`"tempo"`: overall estimated tempo of the track, in BPM.
`"time_signature"`: estimated overall time signature, or beats per measure, of the track.

I plan to use only the 7 highlighted values as dimensions for categorizing music. I would like to use clustering to find clusters of tracks with similar feature values. I could then feed the program a new track that isn't already in the library, and based on its features' Euclidean distance from the nearest cluster centroid, determine how likely it is that the user will like that new track.

On which clustering strategy to use, I am not quite sure yet. I do not think K-means would be a good choice, because I would have to know how many clusters of similar tracks I expect to find before actually building the clusters themselves. My best option is either either agglomerative or density-based clustering. If I had to choose one without doing any further research or data exploration on a sample set of tracks, I would most likely use agglomerative clustering since the algorithm seems simple and should serve the purpose of the project well enough.