Moodify

A music search engine by

🙂❤️😭🤯👻🎹😵

Rock, Saru, Vincent, Walter

Explore music through mood

Create a Web App that recommends songs based on how the user is feeling

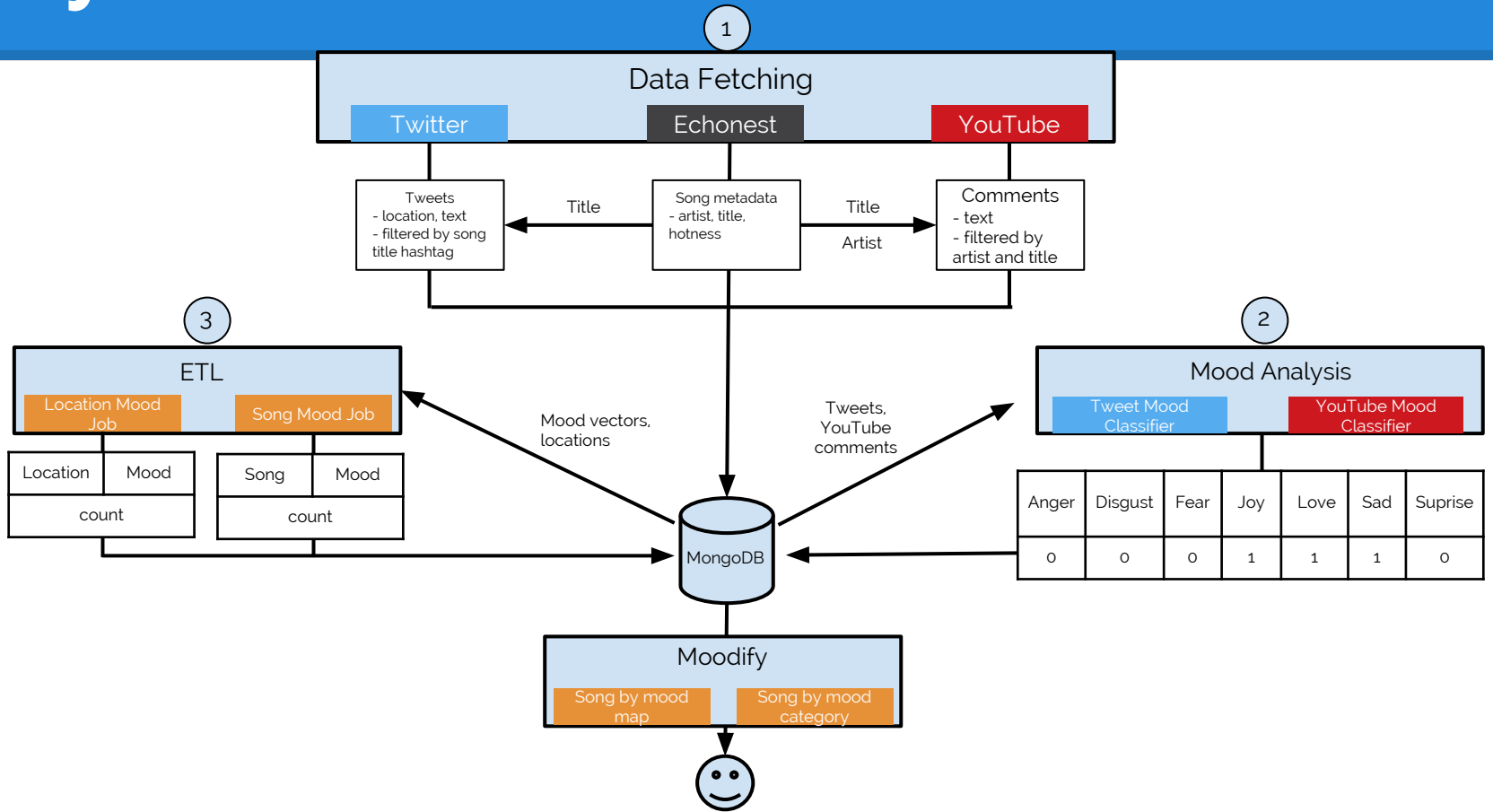
- 7 supported moods

Joy	Love	Sad	Surprise	Fear	Anger	Disgust
						

Functions of our Product:

1) Search songs by mood categories
2) Explore moods by regions in an interactive map

System Architecture



Data Acquisition and Preparation

Fetch Data:

1. Grab a list of trending songs from the Echonest music repository (http://the.echonest.com/) based on 'hotness' score.
   a. ~ 14,000 unique songs
2. Get Tweets and Youtube comments
   a. Queried social media sites based on song title and artist name
   b. Use geo-enabled tweets to get tweet's location, Youtube API does not have info on location

Mood and Location Analysis:

1. Manually create training data to use Multinomial Naive Bayes Classifiers to categorize songs by mood

   a. Process is run separately for Tweets and Youtube comments due to the different nature of language/sentence structure
   b. Each comment is associated with a 7-bit vector containing 0/1, depending if the song expresses a mood
   c. Use Multinomial Naive Bayes Algorithm and training data to generate mood vectors for each tweet/comment.

2. Use Geopy's Nominatim to gather more precise location information on the tweet's coordinates.

Model Building and Implementation

Extract Transform Load (ETL):

1. MapReduce Job 1: Aggregate all the moods associated with a song
   a. Mapper outputs song_id (key) and 7-bit mood vector (value)
   b. For each key, reducer sums each element of the vector and divides each aggregated element count by the total number of mood vectors for that song (total mood score/total tweets for a song)
   c. Result => 7bit probability vector, where each element in the vector indicates the probability of that mood being expressed in a certain song.

2. MapReduce Job 2: Aggregate all the moods associated with a region/country
   a. Mapper outputs location, 7-bit mood vector as the key (value is 1) ie. **["nova scotia,canada", "joy"] 1**
   b. Reducer counts the keys and this value will be displayed in our front-end app ie. **["nova scotia,canada", "joy"] 107**

Ranking songs by Mood:

1. Calibrate for songs that are not as popular or are too popular compared to other songs.

2 For each song (from MapReduce Job 1), take each mood's score from the 7bit probability vector x hotness score(Echonest)
   a. MF-SH score of each mood of a song is stored back in a 7bit vector -> MongoDB

Example: {"love": 0.11, "joy": 0.019, "sad": 0.037, "disgust": 0.028, "anger": 0.037, "surprise": 0.084, "fear": 0.009}

Data Storage and Organization

Data-Acquisition Phase:

1. Local MongoDB store Twitter and Echonest data

2. AWS S3 buckets stores Youtube comments

Post Data-Acquisition Phase:

1. Remote MongoDB that runs on t2.micro EC2 instance (part of AWS free tier services).

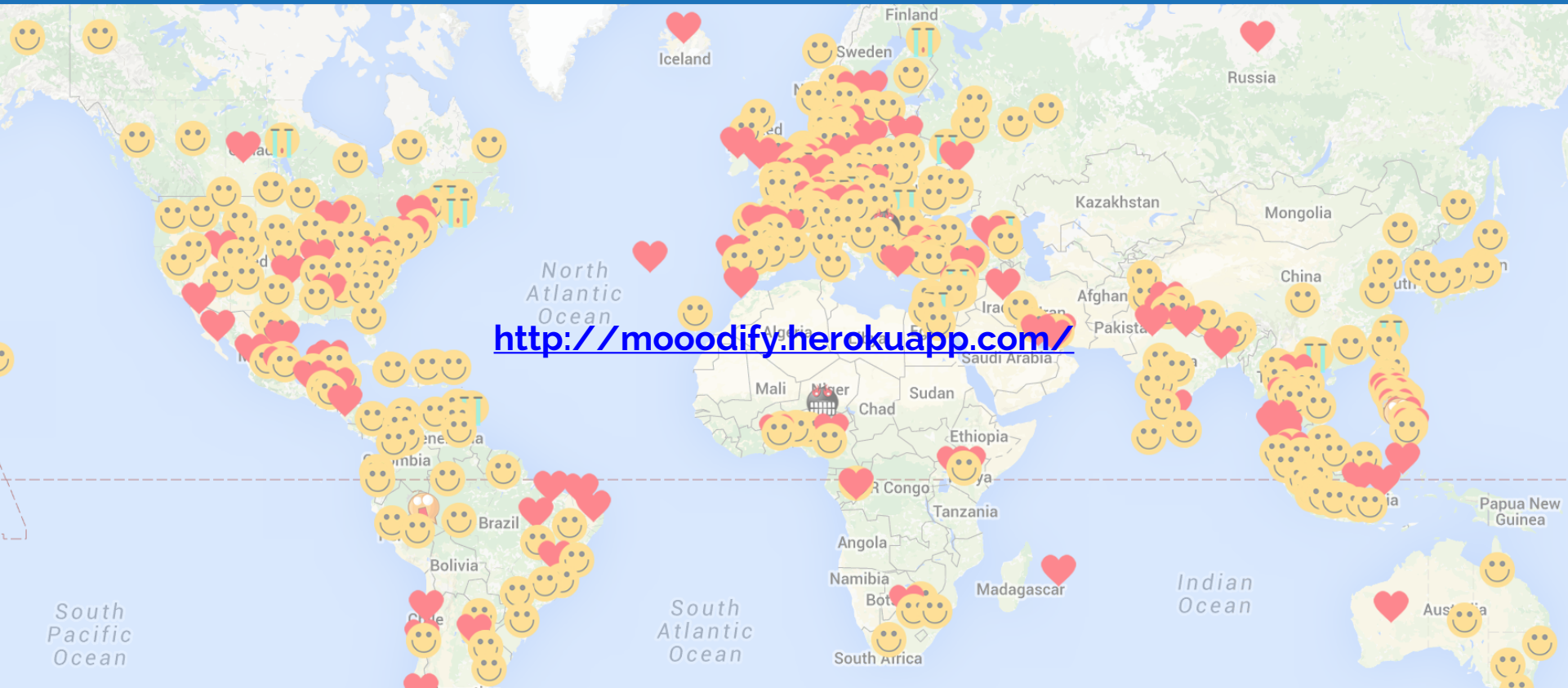
echonest_songs	tweets_v2	youtube_comments	location_moods
- id - title - artists_name - song_hotttnesss - youtube_mf_sh - tweet_mf_sh	- id - song_id - text - coordinates - user - love - joy - sad - disgust - anger - surprise - fear	- id - song_id - text - love - joy - sad - disgust - anger - surprise - fear	- location - love - joy - sad - disgust - anger - surprise - fear - longitude - latitude

Data Summary

Data type	Steps	# of records	Size	Collections
Echonest	Before cleaning	31,398	30.19MB	echonest_song
	After cleaning duplicates	14,487	24.38MB	echonest_songs
Twitter	First iteration	1,785,149 (12,861 unique songs)	846.56MB	tweets
	Second iteration	575,089 (10,700 unique songs)	278.93MB	tweets_v2
YouTube	First iteration	739,134 (11,355 unique songs)	224.09MB	youtube_comments
Location mood	ETL	427 (14,272 unique tweets)	0.1MB	location_moods

Database size: 1.37GB

Result


http://mooodify.herokuapp.com/
Finland
Sweden
Iceland
Russia
Kazakhstan
Mongolia
China
North Atlantic Ocean
Mali
Sudan
Chad
Ethiopia
Tanzania
Angola
Namibia
Madagascar
Brazil
Bolivia
Papua New Guinea
Indian Ocean
South Pacific Ocean
South Atlantic Ocean
South Africa

Conclusion

Easy to grab relevant comments from YouTube

- Simply filtered by song title and artist

Extremely hard to fetch relevant tweets from Twitter

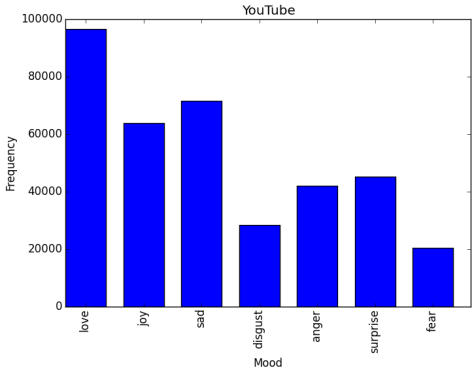
- Filtering by song title and artist gets lots of junk
- Ended up filtered by title hashtag
- Distribution is still skewed

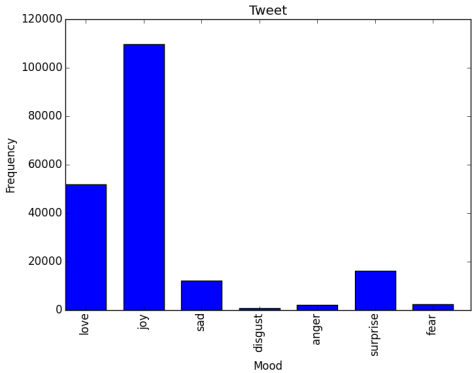
Challenges in Training Data Phase:

- Subjective bias- grey area when classifying moods on a song's comment
  - ie. does this comment reflect surprise and joy?
- Some moods were hard to find in comments
- Tried to get an equal amount of comments reflecting each mood.

Improvement

- More data sources
- Weight and combine mf-sh from each source
- Automate mood score generation pipeline
  - Refresh data every week





Q&A

Thanks! 

http://mooodify.herokuapp.com/

(3 **o**'s in the moo**o**dify)