

Sentiment Analysis in Songs

Clustering: Fall 2016

Mohammed Musaddiq

December 17, 2016

1 Problem Description

To cluster songs based on their lyrical content and obtain a break-up of constituent and dominant sentiments in the lyrics. This process would assist in potentially identifying the overall mood or preference of a particular listener based on a song or playlist the person chooses to listen to.

Proposed Goals:

1. Obtain the break-up of constituent and dominant sentiments in the given songs/playlist based on the lyrical content
2. Assuming each cluster represents one of three sentiment categories - positive, negative and neutral. Cluster the songs/playlist to do the following:
 - Observe cluster sizes/densities and compare to determine the dominant sentiment indicated by the clustering
 - Conclude the mood/preference of the listener for the given set of songs so it could be used to suggest potentially similar songs from elsewhere

2 Data Sources and Literature Review

The primary source of data for this project is the musiXmatch dataset [1] (referred to as MXM dataset in the subsequent text), the official lyrics collection for the Million Song Dataset, available at: <http://labrosa.ee.columbia.edu/millionsong/musixmatch>.

The MXM dataset provides lyrics for many tracks. The lyrics come in bag-of-words format: each track is described as the word-counts for a dictionary of the top 5,000 words across the set. The dataset also provides the full list of stemmed words and the total word counts, i.e. all the words that were seen at least once. There are 498,134 unique words, for a total of 55,163,335 occurrences. The 5,000 words in the dataset account for 50,607,582 occurrences, so roughly 92%.

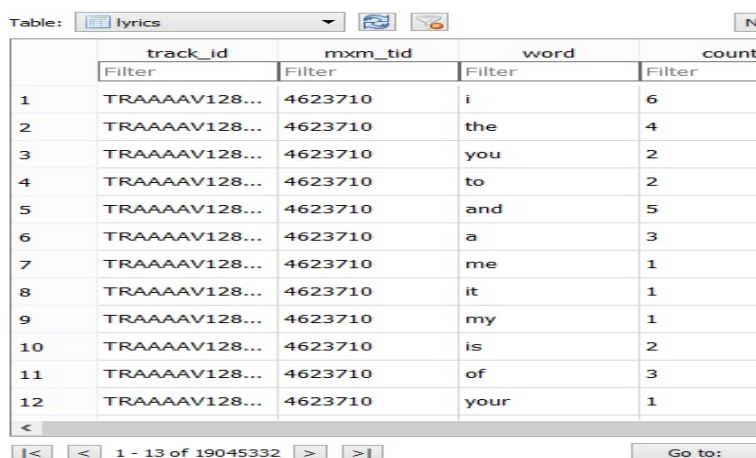
For choosing the 5,000 words, MXM normalized the word counts by the number of word occurrences in each song. Thus, it is not the top 5,000 of this file. The list is noisy, the top 5,000 words are clean, but for the rest, no guarantee has been given.

Finally, for working with visualizing lyrics, stemmed ones are annoying. Therefore, a list of unstemmed words has also been provided with their stemmed version. For instance, it maps 'victori' → 'victory'.

3 My work

Data Processing

I obtained the SQLite version of the dataset [2] since it is faster and more convenient to deal with as a SQLite database. For viewing and editing database files, I used the DB Browser for SQLite [3].



	track_id	mxm_tid	word	count
1	TRAAAV128...	4623710	i	6
2	TRAAAV128...	4623710	the	4
3	TRAAAV128...	4623710	you	2
4	TRAAAV128...	4623710	to	2
5	TRAAAV128...	4623710	and	5
6	TRAAAV128...	4623710	a	3
7	TRAAAV128...	4623710	me	1
8	TRAAAV128...	4623710	it	1
9	TRAAAV128...	4623710	my	1
10	TRAAAV128...	4623710	is	2
11	TRAAAV128...	4623710	of	3
12	TRAAAV128...	4623710	your	1

Figure 1: The *lyrics* table: contains word-counts for each track in the database

Since the original *lyrics* table is too large for processing at once, I extracted two smaller sets from it, namely *playlist1* and *playlist2*, each containing 40 songs selected arbitrarily.

First, I wanted to drop stopwords from the input, that is, high-frequency words like *the*, *to* and *also* since they usually have little lexical content, and their presence in the lyrics fails to distinguish it from other instances.

For this filtration, I used NLTK's stopwords corpus [4].

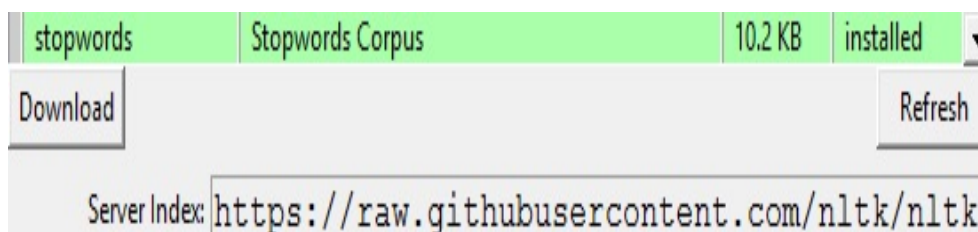


Figure 2: NLTK's stopwords corpus

As mentioned earlier, the *lyrics* table contains stemmed version of the lyrics. For example, "cry", "cried", and "crying" are treated as instances of the same thing, rather than treating them as distinct, unrelated tokens. So, for this example, all these words are mapped to "cri".

For my approach (as would become evident in the forthcoming stages), I wanted to convert these to an unstemmed form (exact conversion is not required since every unstemmed variant would mostly have the same sentiment score).

For the aforementioned conversion, I used the stemmed \rightarrow unstemmed mapping file provided by MXM [5]. Here's a section of the mapping from the file:

```
lullabi<SEP>lullaby
nightmar<SEP>nightmare
situat<SEP>situation
infin<SEP>infinity
redempt<SEP>redemption
sunshin<SEP>sunshine
puzzl<SEP>puzzle
```

Figure 3: stemmed \rightarrow unstemmed mapping

The last concern about the current input is that the lyrics often contain words from other languages (even though the song itself is predominantly in English).

So, for ensuring filtering of words outside the English language, I used NLTK's words corpus [4].

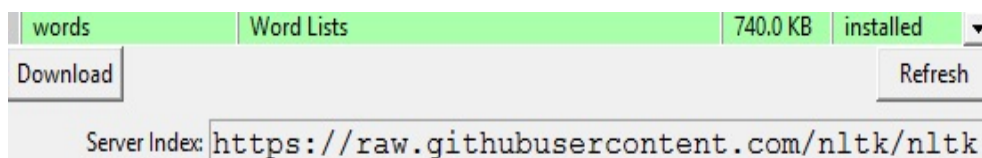


Figure 4: NLTK's words corpus

Data Analysis

Now that the input has been massaged appropriately, I wanted to do sentiment analysis at the word-level in each song lyric.

For this purpose, I used the approach followed in NLTK's Vader SentimentAnalyzer [6, 7].

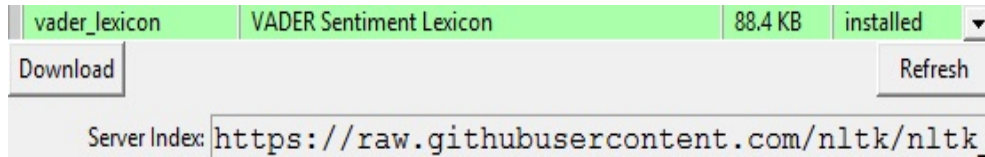


Figure 5: NLTK’s Vader Sentiment Analyzer

Following is a sample for a song whose sentiment scores have been calculated based on its lyrics:

```

Lyrics
know time get feel make way take away back think try won't ever lies much last thought break put far waste tired trust throw b
uild dust swear self tension deceit

Sentiment Scores
{'neg': 0.299, 'neu': 0.563, 'pos': 0.138, 'compound': -0.6776}

```

Figure 6: Sentiment scores for a song based on its lyrics

Based on the sentiment scores available, the input can be set up for clustering. The number of rows will be the number of songs and the number of features will be 4 (i.e. 'neg' negative score, 'neu' neutral score, 'pos' positive score and 'compound' overall sentiment score for that particular song).

Now that the data is available for clustering, I intended to run *k-means* with number of clusters as 3, the hope being that songs are clustered into three categories, according to the dominant sentiment: positive, negative or neutral.

First, I shall cluster *playlist1*:

```

playlist1 : 40 songs (each having 4 features)
[[ 0.086  0.755  0.16  0.5106]
 [ 0.19  0.609  0.201 -0.0634]
 [ 0.149  0.673  0.178  0.25 ]
 [ 0.321  0.389  0.29  -0.1486]
 [ 0.  0.923  0.077  0.3612]
 [ 0.118  0.582  0.3  0.8979]
 [ 0.28  0.466  0.254 -0.1027]
 [ 0.272  0.532  0.197 -0.6369]
 [ 0.377  0.445  0.178 -0.9264]
 [ 0.205  0.569  0.225  0.5778]
 [ 0.  0.652  0.348  0.9442]
 [ 0.31  0.69  0.  -0.8689]
 [ 0.157  0.494  0.349  0.7351]
 [ 0.164  0.623  0.213  0.312 ]
 [ 0.306  0.359  0.335  0.6096]
 [ 0.27  0.656  0.075 -0.8934]
 [ 0.  0.612  0.388  0.9545]
 [ 0.177  0.748  0.075 -0.7964]
 [ 0.146  0.541  0.314  0.9042]
 [ 0.137  0.664  0.198  0.6486]
 [ 0.109  0.729  0.162  0.5994]
 [ 0.109  0.524  0.367  0.8851]
 [ 0.204  0.581  0.215 -0.25 ]
 [ 0.167  0.42  0.413  0.9466]
 [ 0.178  0.524  0.298  0.8214]
 [ 0.359  0.455  0.186 -0.9201]
 [ 0.123  0.513  0.364  0.8689]
 [ 0.  0.923  0.077  0.3612]
 [ 0.16  0.671  0.169  0.3818]
 [ 0.314  0.537  0.149 -0.8032]
 [ 0.326  0.522  0.151 -0.8555]
 [ 0.137  0.698  0.165 -0.0258]
 [ 0.126  0.654  0.22  0.8176]
 [ 0.239  0.524  0.237 -0.0258]
 [ 0.125  0.512  0.363  0.9459]
 [ 0.104  0.798  0.098 -0.4019]
 [ 0.477  0.47  0.053 -0.9885]
 [ 0.207  0.563  0.23  0.5389]
 [ 0.106  0.757  0.137 -0.0516]
 [ 0.233  0.393  0.374  0.7717]]

```

Figure 7: *playlist1*

cluster centers computer by kmeans

```

[[ 0.3202222  0.56166667  0.11822222 -0.85436667]
 [ 0.14066667  0.55888889  0.30044444  0.77655556]
 [ 0.158      0.66423077  0.17776923  0.04587692]]

```

Figure 8: Clustering *playlist1* using kmeans

Based on Figure 8, I was able to ascertain that kmeans computed the cluster centers as I had hoped, i.e. for negative, positive and neutral sentiments respectively.

Next, I went ahead with plotting the clusters as shown here:

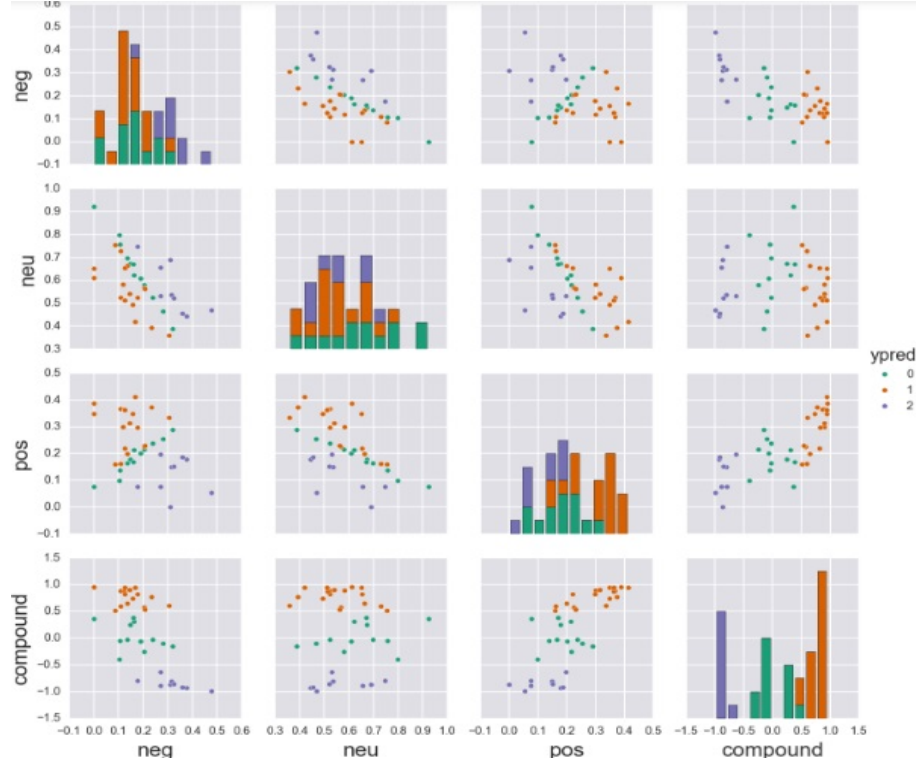


Figure 9: Plot for *playlist1*

From the scatter plot, I observed the following:

- Cluster density/members seem to be the most for the red points i.e. *cluster 1* according to the legend.
- The red points seem to be tending towards the positive sentiment direction when the plots along the vertical for the "pos" feature are observed, possibly hinting towards domination of positive sentiment.
- Furthermore, *cluster 0* i.e. the green points seem to be overwhelmed by the other two clusters, indicating that these could potentially be the sentiment that is not so pronounced for the given set of songs

For supporting the observations I made so far, I wanted to check the numerical statistics based on the *kmeans_labels* computed, which revealed the following information:

Cluster Composition:

cluster 0 (green color, supposedly the negative sentiment songs): 9

cluster 1 (red color, supposedly the positive sentiment songs): 18

cluster 2 (blue color, supposedly the neutral sentiment songs): 13

Therefore, I could infer that *playlist1* containing 40 songs was predominantly a positive sentiment set.

Now, I shall cluster *playlist2*:

```
playlist2 : 40 songs (each having 4 features)
[[ 0.251  0.652  0.098 -0.8689]
 [ 0.19  0.609  0.201 -0.0634]
 [ 0.149  0.673  0.178  0.25 ]
 [ 0.299  0.563  0.138 -0.6776]
 [ 0.321  0.389  0.29 -0.1486]
 [ 0.  0.923  0.077  0.3612]
 [ 0.276  0.606  0.118 -0.8238]
 [ 0.323  0.383  0.294 -0.4404]
 [ 0.28  0.466  0.254 -0.1027]
 [ 0.272  0.532  0.197 -0.6369]
 [ 0.205  0.569  0.225  0.5778]
 [ 0.  0.652  0.348  0.9442]
 [ 0.377  0.445  0.178 -0.9264]
 [ 0.31  0.69  0. -0.8689]
 [ 0.157  0.494  0.349  0.7351]
 [ 0.164  0.623  0.213  0.312 ]
 [ 0.306  0.359  0.335  0.6096]
 [ 0.27  0.656  0.075 -0.8934]
 [ 0.  0.612  0.388  0.9545]
 [ 0.177  0.748  0.075 -0.7964]
 [ 0.204  0.581  0.215 -0.25 ]
 [ 0.167  0.42  0.413  0.9466]
 [ 0.178  0.524  0.298  0.8214]
 [ 0.359  0.455  0.186 -0.9201]
 [ 0.123  0.513  0.364  0.8689]
 [ 0.  0.923  0.077  0.3612]
 [ 0.16  0.671  0.169  0.3818]
 [ 0.314  0.537  0.149 -0.8032]
 [ 0.326  0.522  0.151 -0.8555]
 [ 0.137  0.698  0.165 -0.0258]
 [ 0.329  0.4  0.271 -0.7003]
 [ 0.239  0.524  0.237 -0.0258]
 [ 0.104  0.798  0.098 -0.4019]
 [ 0.477  0.47  0.053 -0.9885]
 [ 0.217  0.604  0.179 -0.8271]
 [ 0.224  0.616  0.161 -0.8752]
 [ 0.106  0.757  0.137 -0.0516]
 [ 0.239  0.548  0.212 -0.5719]
 [ 0.237  0.623  0.14 -0.7579]
 [ 0.235  0.722  0.043 -0.743 ]]
```

Figure 10: *playlist2*

Cluster centers computer by kmeans

```
[[ 0.1625  0.65308333  0.18441667  0.08319167]
 [ 0.2808  0.5785  0.1408 -0.768865 ]
 [ 0.142  0.517875  0.34  0.8072625 ]]
```

Figure 11: Clustering *playlist2* using kmeans

Based on Figure 11, I was able to ascertain once again that kmeans computed the cluster centers as I had hoped, i.e. for neutral, negative and positive sentiments respectively.

Next, I went ahead with plotting the clusters as shown here:

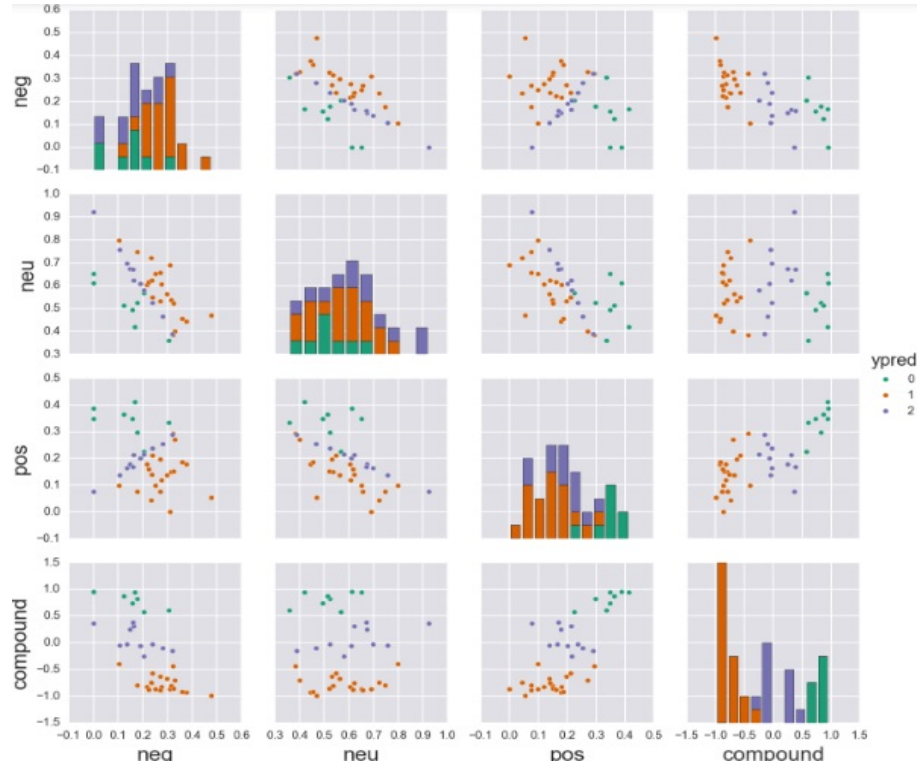


Figure 12: Plot for *playlist2*

From the scatter plot, I observed the following:

- Cluster density/members seem to be the most for the red points i.e. *cluster 1* according to the legend.
- The red points seem to be tending towards the negative sentiment direction when the plots along the vertical for "*neg*" feature are observed, possibly hinting towards domination of negative sentiment.
- Furthermore, *cluster 2* i.e. the blue points seem to be overwhelmed by the other two clusters, indicating that these could potentially be the sentiment that is not so pronounced for the given set of songs

For supporting the observations I made so far, I wanted to check the numerical statistics based on the *kmeans_labels* computed, which revealed the following information:

Cluster Composition:

cluster 0 (green color, supposedly the neutral sentiment songs): 12
cluster 1 (red color, supposedly the negative sentiment songs): 20
cluster 2 (blue color, supposedly the positive sentiment songs): 8

Therefore, I could infer that *playlist2* containing 40 songs was predominantly a negative sentiment set.

4 Discussion

Goals proposed/achieved:

1. Obtain the break-up of constituent and dominant sentiments in the given songs/playlist based on the lyrical content

Achievement: As seen in case of *playlist1* and *playlist2*, I have provided the sentiment composition for each collection of songs and also deduced the dominant sentiment based on the numerical information available in the form of *kmeans_labels* that were computed.

2. Assuming each cluster represents one of three sentiment categories - positive, negative and neutral.

Achievement: As seen in in case of *playlist1* and *playlist2*, kmeans was able to set up cluster centers representing each of the sentiment categories.

Cluster the songs/playlist to do the following:

- Observe cluster sizes/densities and compare to determine the dominant sentiment indicated by the clustering

Achievement: As seen in in case of *playlist1* and *playlist2*, the cluster densities for majority and minority sentiments were visually evident in the scatter plots which enabled the analysis of the given set of songs.

- Conclude the mood/preference of the listener for the given set of songs so it could be used to suggest potentially similar songs from elsewhere

Achievement: Based on the numerical and visual inferences available from the clusterings, I was able to successfully conclude the overall mood/preference of the listener for each given playlist of songs. However, suggesting other similar songs based on my conclusion is something that I would like to do in the future as an extension to the current logic - a possible approach could be to pick a random song from the universal song collection and see which cluster it falls into and decide accordingly whether it could align with the listener's known preferences and ultimately provide suggestions.

5 Bibliography

References

- [1] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.
- [2] Source: http://labrosa.ee.columbia.edu/millionsong/sites/default/files/AdditionalFiles/mxm_dataset.db
- [3] Source: <http://sqlitebrowser.org/>
- [4] Source: <http://www.nltk.org/book/ch02.html>
- [5] Source: http://labrosa.ee.columbia.edu/millionsong/sites/default/files/mxm_reverse_mapping.txt
- [6] Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.
- [7] Source: http://www.nltk.org/_modules/nltk/sentiment/vader.html
- [8] Other reference of relevance: <https://prezi.com/pzl-yt3epq4a/using-word-vector-cluster-for-sentiment-analysis/>