Text mining with lyrics

Jure Zajc
Faculty of Computer and Information Science
University of Ljubljana
Večna pot 113, 1000 Ljubljana
Email: jz4314@student.uni-lj.si

Miha Debenjak
Faculty of Computer and Information Science
University of Ljubljana
Večna pot 113, 1000 Ljubljana
Email: md9754@student.uni-lj.si

***Abstract*—For our project we were wondering if there is any correlation between musical lyrics and genres. We created a classifier for music genres based on their lyrics. We also created a classifier for the mood of the songs. With the `LDA` topic model we tried to discover the genres in the database, without any prior knowledge about the songs genres.**

1. Introduction

Natural Language Processing (NLP) is a field in machine learning with which we try to to understand, analyze, manipulate, and potentially generate human language with computers. Its use is more and more common in the every day life.

The problem we were working on for this project was text mining with lyrics. Can we guess a songs genre just by looking at its lyrics? And can we cluster a set of lyrics into genres? It seems like a possible outcome, but we have to keep in mind, that songs from different genres can have very similar lyrics, which can be a problem. We tried to answer these two questions, with the use of different machine learning and NLP techniques.

2. Experiments

Our work is divided into two parts. First part concentrates on making classifiers for genres and the songs mood. The second part is unsupervised learning of the genres on an dataset of songs, whose genres are unknown.

2.1. Dataset

Our data set [1] contained 1200 songs. The train dataset were made out of 1000 songs while the test dataset was made out of 200 songs. We also removed the songs form genres, which were not represented with at least 30 songs. Genre distribution can be seen in Figure 1.

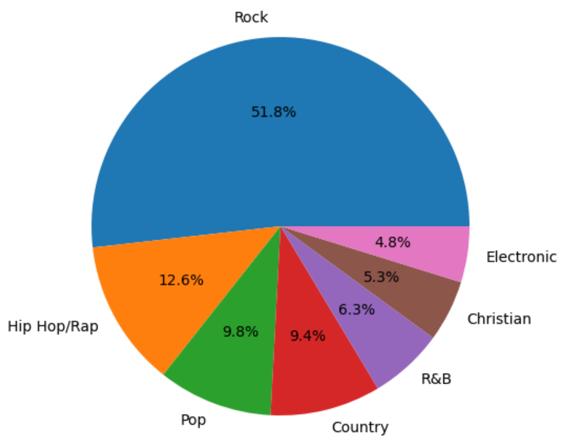


Figure 1. Genre distribution on data set

Each song in the data set contained: file name, artist, title, lyrics, genre, mood and release date. We were only interested in artist, title, lyrics, genre and mood.

2.2. Classification

Our algorithm is described below:

- Create bag of words [2] for train file,
- Create bag of words for test file,
- Remove spaces and add additional stopwords from `nltk` [3] from train and test file,
- Create voctors with CountVectonizer (simultaneously input is stemmed using PorterStemmer) $\longrightarrow$ running $\longrightarrow$ run,
- Fit and transform training file,
- Transform test file,
- Resample training data using `RandomOverSampler`,
- Try to fit to naive bayes and SVM.

The bag of words model is a simplifying representation used in natural language processing and information retrieval. In this model, a text (such as a sentence or a document) is represented as a bag of its words, disregarding grammar and even word order.

Stopwords are english words which do not add much meaning to a sentence and can be found in every text. They can safely be ignored without sacrificing the meaning of the sentence. For example, the words like the, he, have etc. We used stopwords from the nltk library, but we also added some of our own e.g. customStopWords = ["′s", "n′t", "′m", "′re", "′ll", "′ve", "...", "ä", "′′", ′``′, ′--′, "′d", ′el′, ′la′].

CountVectonizer converts a collection of text documents to a matrix of token counts. This implementation produces a sparse representation of the counts using scipy.sparse.csr_matrix. If you do not provide an a-priori dictionary and you do not use an analyzer that does some kind of feature selection then the number of features will be equal to the vocabulary size found by analyzing the data. We set the parameters as follows:

- analyzer = stemmed_words,
- lowercase = true, so all words are lower-cased,
- max_features = 5000,
- stopwords = stopwords from nltk library with custom stopwords.

Almost all actions were made with SciKit [4] which is library for machine learning in Python. We resampled training data, because we wanted to see if the results would be different. Resampling changes the dataset into a more balanced one by adding instances to the minority class or deleting ones from the majority class, that way we should build better machine learning models. We picked the RandomOverSampler method from library imbalanced-learn [5] which over-samples the minority class(es) by picking samples at random with replacement. Then we fitted the original and resampled data to Naive Bayes and SVM classifiers.

2.3. Topic modeling

We then tried to recreate the given genres only by considering the words in lyrics, without any prior knowledge about the genres of the lyrics. This is the goal of topic modeling, which recognizes topics within a set of documents. Because it does not require predefined classes it is an unsupervised technique.

For the implementation of topic modeling we used Latent Dirichlet Allocation (LDA), proposed by Blei et al. [6]. It receives the set of preprocessed texts and the number of topics we want to generate. We can guess the number of topics or measure the topic coherence between the output topics of models with different numbers of topics to determinate the correct or best number of topics. The returned topics are lists of pairs of words and their weight on the given topic. We must then determine the meaning of the returned topics by ourselves. LDA was implemented with the Genism library for Python.

We can then calculate the dominant topic of each text and compare it to the original genre of the text. This way we can see whether the topics represent the original genres or that they contain different genres.

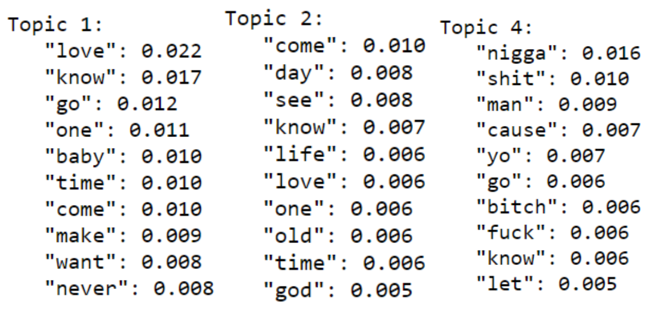
2.4. Results

The results given by our implementations can be seen in Table 1. We can see, that mood has a very higher accuracy score. That is probably because, there are only 2 types of mood in the dataset: Happy and Sad. On the other hand, the results for genre classification were not as good. This can be due to the fact that artists from different genres use similar words in their songs. For example the word Love is most popular word in 3 different genres: Pop, Christian and R&B.

Type	Genre	Mood
Naive Bayes	0.58	0.97
Naive Bayes - Resampled	0.535	0.98
SVM	0.565	0.95
SVM - Resampled	0.585	0.96
Majority classifier	0.518	0.55

TABLE 1. RESULTS ON MOOD AND GENRE

We generated the LDA model for different numbers of topics. Unfortunately in most cases the generated topics did not represent the genres in the dataset. Although there were excetions, with the models generating 3 and 4 topics. In these models one of the topics represented the Hip Hop/Rap genre. We can see 3 out of four topics in Figures 2. The first topic looks like a love topic, the second can maybe be more friendship related, while the third one is clearly Hip Hop/Rap.



```
Topic 1:                  Topic 2:                  Topic 4:
   "love": 0.022             "come": 0.010             "nigga": 0.016
   "know": 0.017             "day": 0.008              "shit": 0.010
   "go": 0.012               "see": 0.008              "man": 0.009
   "one": 0.011              "know": 0.007             "cause": 0.007
   "baby": 0.010             "life": 0.006             "yo": 0.007
   "time": 0.010             "love": 0.006             "go": 0.006
   "come": 0.010             "one": 0.006              "bitch": 0.006
   "make": 0.009             "old": 0.006              "fuck": 0.006
   "want": 0.008             "time": 0.006             "know": 0.006
   "never": 0.008            "god": 0.005              "let": 0.005
```

Figure 2. Generated topics with the LDA with 4 topics.

We can say that the other generated topics actually represent the topics of lyrics and not the genre. That is because different genres can use very similar vocabularies. We can conclude that the Hip Hop/Rap genre has the most unique vocabulary, which makes sense, since it relies heavily on slang.

3. Conclusion

In this project we tried to create a classifier for musical lyrics and generate genres on an unknown se of lyrics. The proposed classifier did not give the best results for genre classification, because different genres can have very similar vocabularies. The mood classifier was much more accurate, but this is also due to the fact it only had two classes.

We manage to recreate one of the genres (Hip Hop/Rap) with the `LDA` topic model. Other genres were mixed in the rest of the generated topics. We conclude that is because Hip Hop/Rap has the most distinct vocabulary, while the other genres mostly talk about similar topics with similar words.

It seems that text mining is not the best approach to defining a songs genre. The most appropriate approach would probably be analyzing the sounds of the songs, as genres are more defined by sound then by lyrics.

References

[1] "musicmood/dataset at master · rasbt/musicmood," https://github.com/rasbt/musicmood/tree/master/dataset, (Accessed on 04.06.2020).

[2] "A gentle introduction to the bag-of-words model," https://machinelearningmastery.com/gentle-introduction-bag-words-model/: :text=A20bag2Dof2Dwords20is,the20presence20of20known20words., (Accessed on 04.06.2020).

[3] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python*. OReilly Media Inc., 2009.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[5] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017. [Online]. Available: http://jmlr.org/papers/v18/16-365.html

[6] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 05 2003.