Machine Learning for Genre Classification of Music Lyrics and Recommendation Systems

Lewin Cary, Ian Scott Knight
Stanford University

Abstract

What is it in a song that allows us to immediately determine: “that’s rap”, or: “that’s country”? For the learned ear, music genre classification comes naturally. In this paper we attempt to teach an untrained computer to predict the genre of a song purely from an examination of the song’s lyrical content, using various machine learning techniques. Can we give good quality recommendations as to what a user should listen to next based purely off the lyrics of a song and nothing else? This paper looks deeply at thousands of songs to determine if there are patterns in the lyrics between genres.

1. Introduction

Music recommendation and recommendation systems in general play a huge role in the way we currently consume media. The experiences given by different applications such as Youtube, Netflix and Spotify rely vastly on consistently bringing high quality content that the user will enjoy to the user. As a result of the scale of these companies, most current industry practices revolve around collaborative filtering approaches. These types of techniques rely on having large amounts of user behavior data and content metadata (i.e. if person A and B like song C, and person B likes song D, then person A will like song D). These techniques work very well, but are limited in that they can only be effective if there are large amounts of user data available, the likes of which only a few companies have access to.

At a high level this paper attempts to examine and extract the latent semantic meaning present within the lyrical content of songs in the context of their genres. This is explored with two interrelated goals. Firstly, we aim to understand the relationship between the lyrics of a song and its genre and will test this by attempting to predict the genres of songs based on their lyrics. Secondly, we will attempt to make relevant and similar music recommendations to users once again purely based on the lyrics of songs.

2. Related Work

There have been several attempts in recent years to use machine learning techniques to classify songs into their respective genres. As mentioned earlier, industry relies mostly on collaborative filtering approaches on user behavior data. However there has been some work done on classifications and recommendations based on the actual content of the media.

These attempts have relied mostly on sonic analysis approaches, and have had limited success. A classifier is built to predict upon the actual audio of the song, using fourier transforms. Previous papers such as Fell and Gossi have used linear classification models trained on music lyrics in a similar fashion to this paper.

3. Gathering Data/Feature Selection

Dataset

Our dataset is derived from the musiXmatch dataset, a subset of the famous Million Song Dataset (MSD) that includes the lyrics of 237,701 songs in addition to their other data as taken from the original MSD. Both datasets are curated by LabROSA, a musicological institution at Columbia University.

The lyrics are stored in a bag-of-words format, where words are represented as positive integers. For efficiency, only the top 5000 most frequently uttered words in the total song vocabulary were included. Therefore, each song's lyrics are delineated by the a list of (token, n) tuples, where n is the number of instances of that token (i.e. word) in the song. We are also given a mapping of integer tokens to their corresponding word, which comes in lemmatized form.

At this point, we vectorized the lyrical data into 5000-dimensional vector representations (one for each token/word), where the value for every dimension is equal to the number of instances of the corresponding token/word in a given song. Therefore, most values are zero.

Preprocessing: Matching musiXmatch Song ID to MSD Song ID

As it exists, the musiXmatch dataset keeps track of songs by a different ID system than that of the original Million Songs Dataset. Since LabROSA does not provide a mapping of musiXmatch song ID to song name/album/artist, it was necessary to do so ourselves by means of the following process:

1. Match musiXmatch song ID to MSD song ID using a mapping provided by LabROSA
2. Match the corresponding MSD song ID to the song name/album/artist using another mapping provided by LabROSA

After completing this, we knew which songs were which in our musiXmatch dataset. By using the data categories included in the MSD dataset, we could now work on collecting genre information and mapping it to musiXmatch song ID.

Preprocessing: Genre Class Creation

We were unable to extract genre information from the MSD without some form of preprocessing, because it does not include "genre" proper as a data category. Instead, it assigns a set of labels called "tags", which could be anything from broad genre (e.g. "rock") to ultra-specific genre (e.g.

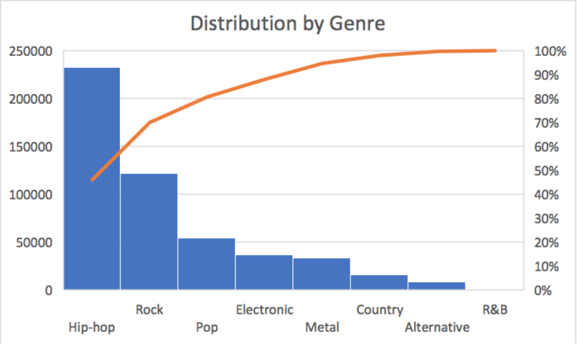
“Australian didgeri-bluegrass”) to something else entirely (e.g. “this goes hard in the paint bro”). These tags are scraped from tags provided real-world listeners on Last.fm.

Due to the large variety of tags, we needed to devise a method for isolating the broad genre tags and assigning them to each song ID as a new data category. The process we implemented is as follows:

1. Remove all tags not included in the pre-selected genres we decided to use for this project:
    a. pop, rock, hip-hop, country, metal, electronic, alternative, r&b
2. Count the number of instances of genre tags for each genre for each song
3. Assign the genre with the most instances to the given song as the song’s genre

Preprocessing: Distribution Correction

At this point, we successfully created a dataset of song IDs to their de facto genre as voted by real-world listeners. However, analysis of the distribution of representation across our 8 genre classes indicated a highly uneven dataset:



Given that “r&b” was extremely underrepresented even compared to the second least common genre (“alternative”), we decided to simply remove it as a genre, leaving 7 genres total.

We then sampled every genre the same number of times as the number of songs in the “alternative” genre, producing a smaller but evenly distributed dataset of 62,706 songs.

Feature Selection

For feature selection, we opted for (1) term frequency – inverse document frequency reweightings (TF-IDF) of the lyrics vectors, as well as (2) the 7 cosine distances of the lyrics vector of a given song from the average vector of each genre. TF-IDF is a well-established reweighting method that has been shown to provide more information for classifiers in many cases, and the latter feature is a novel feature of our own construction.

4. Methods

1. Genre Classification of Lyrics

We implemented three different classifiers in hopes of achieving high-fidelity genre classifications:

a. Logistic Regression
b. Multinomial Naive Bayes
c. Stochastic Gradient Descent

We implemented all three in Python, and narrowed down on hyperparameters using basic hyperparameter tuning.

2. Music Recommendation

The user is prompted to enter the lyrics of song. Once these have been obtained, the lyrics are stemmed using the Porter stemming algorithm. This ensures that all the myriad forms of the same word are converted into one word. This makes sense semantically, as the difference in meaning between "run" and "running" is too nuanced to be picked up by our models.

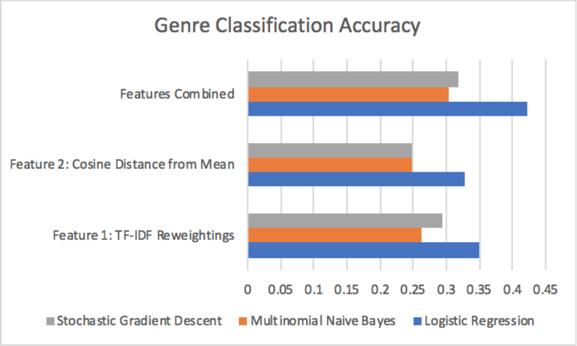
The stemmed lyrics are then placed into the designed vector space model. This model is largely based upon the bag-of-words model given by the Million Song Dataset, which is capped in dimensionality at 5000 (where the 5000 refers to the top 5000 words found most commonly across all the lyrics of songs contained within the dataset).

We then run a cosine similarity metric between the converted VSM of the user input song and each song in the dataset, and then return the top 10 songs to the user as our recommendations.

5. Results and Discussion

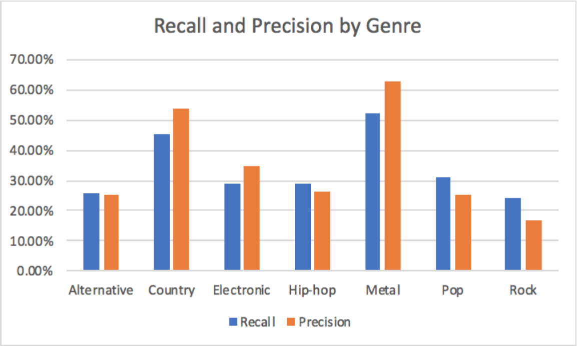
1. Genre Classification of Lyrics

Below are our accuracy results by classifier:



It seems that each feature alone provides reasonable classification accuracy and when combined they produce a better score across classifiers. Moreover, the logistic regression classifier performed the best regardless of the features used.

Below are the accuracy results by genre for our best classifier (logistic regression, features combined):



The genres of metal and country provided the best results, with metal achieving an accuracy around 60% and country around 50%. Other genres produced accuracies between 20~30%. When compared to the accuracy of around 14% produced by uniformly randomly guessing each genre, these results achieve considerable success, especially in the cases of metal and country. As for why these genres allow easier classification, it is inferred that these genres contain lexical characteristics that distinguish them from other genres. Real-world experience of these two genres readily confirms these suspicions. For example, countless jokes are made about how country music universally references beer, trucks, girls or some combination thereof. Similarly, the metal genre is known for its proclivity for grim subject matter and, therefore, fairly distinctive lyrics. As for the other genres, there are no expectations of significantly distinctive lyrical trends. For pop music in particular, it would make sense that it contain fairly nondescript lyrics given its goal of appealing to the largest number of people as possible.

Cleaned Data vs. Uncleaned Data Performance

The uneven dataset technically produced a higher accuracy than the even dataset, but examination of the confusion matrix demonstrated that this was simply due to overprediction of more highly represented classes (e.g. “pop”).

2. <u>Music Recommendation</u>

We found that the cosine similarity metric actually ended up returning relatively high similarity metrics for large numbers of songs (it was not uncommon to return over 10 000 songs with scores over 70% similar). This indicates that there are large numbers of songs that utilize similar vocabularies.

It was difficult to design a quantitative metric to measure the success of the song recommendations. One possible design could be to give recommendations to users, and record if they like the music or not. However this was outside the scale of the resources of this course. Furthermore, just because a user likes the song that the system recommended does not necessarily indicate that the system made a good recommendation. If a system incorrectly classifies a song as ‘rap’ even though it was ‘country’ and then returns another similar rap song to the user and the user happens to like it, this would be by chance rather than by any deliberate decision of the system.

We therefore will make qualitative judgments about the quality of the recommendations the system gave. We found that the quality of recommendations were far better for certain music genres than they were for others in that they were able to consistently recommend other songs that we liked within the same genre. These included ‘metal’ and ‘country’. One possible reason for this is that our work in the classifier described in part 1 revealed that these genres have highly unique vocabularies. Therefore the similarity metrics found between songs with similar vocabulary are exactly the types of inputs that would be accentuated within a cosine algorithm.

6. Conclusion and Future Work

This paper was able to demonstrate the semantic similarities between different genres of music. It was able to capture the idea that certain genres tend towards certain vocabularies, and able to capture this effectively to build recommendation systems.

In the future it would be interesting to examine different classifier models. For example, trying to use a neural network rather than a linear classifier. Additionally, there is much work to be done with different feature selection. This paper was limited by the form of the given data (we were given songs pre-stemmed and in bag-of-words format), but the use of a different dataset would lend itself to bigram and trigram features.

It would also be interesting to combine both parts of the project more concretely. For example, we could run a classifier on the given user song and store the output genre as a high weighted feature within the vector space model. Should the cosine similarity metric come across a different song that fall within the same genre this would lead to high scoring outputs and dramatically increase the accuracy of the recommendations.

Bibliography

Boost up! Sentiment Categorization with Machine Learning Techniques. (2018). *Stanford NLP*. [online] Available at: https://nlp.stanford.edu/courses/cs224n/2009/fp/16.pdf [Accessed 13 Jun. 2018].

Faruqui, M. (2018). Retrofitting Word Vectors to Semantic Lexicons. *Association for Computational Linguistics*. [online] Available at: http://www.aclweb.org/anthology/N15-1184 [Accessed 13 Jun. 2018].

Fell, M. (2018). Lyrics-based Analysis and Classification of Music. *Association of Computational Linguistics*. [online] Available at: http://www.aclweb.org/anthology/C14-1059 [Accessed 13 Jun. 2018].

Gossi, D. (2018). Lyric-based Music Recommendation. [online] Available at: https://www.cse.unr.edu/~mgunes/papers/ComNet16Lyric.pdf [Accessed 13 Jun. 2018].

Hu, X. (2009). LYRIC TEXT MINING IN MUSIC MOOD CLASSIFICATION. *International Society for Music Information Retrieval*.

Logan, B. (2018). Semantic Analysis of Song Lyrics. *International Conference on Multimedia and Expo*. [online] Available at: https://www.semanticscholar.org/paper/Semantic-analysis-of-song-lyrics-Logan-Kositsky/94fc4c86bd48d1b8dc84dfa742669d9e952a1189 [Accessed 13 Jun. 2018].