Lyrics and Predictive Analytics

Brendan Herger
Chandrashekar Konda
Layla Martin

March 16, 2015

1 Abstract

In order to further understand how lyrics differentiate musical styles, we trained a variety of statistical models to classify two musical genres. Utilizing a custom built data set containing the most popular R&B and Adult Contemporary songs throughout time, we were able to experiment with parametric and nonparametric machine learning approaches to model text data. Our end classifier was able to correctly identify the genre of songs in a holdout set with 83% accuracy.

2 Introduction and Motivation

Classification problems arise all the time in predictive analytics. Spam filtering, sentiment analysis, information retrieval, and author identification are some common examples. You'll notice that the previous four types of problems involve text rather than numeric or categorical data. In unique cases such as these, features must be engineered in order to extract relevant information from the text; with the hope that a model can be built from these features which will correctly suit the problem at hand.

In order to familiarize ourselves with techniques such as text mining and feature engineering, we took on a problem in which we sought to differentiate between genres of music solely from lyrics. Specifically, we took a subset of Adult Contemporary and R&B songs and built machine learning models to predict which genre a song is most likely to belong to using only the lyrics of that song. We started by acquiring a set of "popular" songs and corresponding lyrics across the Adult Contemporary and R&B genres. We then explored parametric and nonparametric machine learning techniques to model this data and make predictions about the likely genre of a new song.

While this specific problem does not lend itself to a real world application - if lyrics are known, the genre is typically know as well - the process and techniques we have performed can be extended to text classification problems in a variety of other more applicable fields.

3 Data

3.1 Acquiring Popular Songs

The first step in model building was acquiring a reliable set of data. We got "popular" songs from Billboard Magazine, commonly known as the number one magazine in the United States music industry. The Billboard Magazine website, Billboard.com, keeps archives of the weekly number one Billboard chart songs across genres dating back to 1958. We acquired these song titles and corresponding artists for the genres of Adult Contemporary and R&B by using the BeautifulSoup web scraping module in Python. The archives for these two genres existed as far back as 1961 for Adult Contemporary and 1985 for R&B. This gave us a list of every song which had appeared as a chart topper in its respective genre at some point in time.

3.2 Acquiring Lyrics

In order to generate a reliable set of lyrics, we partnered with MusiXmatch, who graciously granted us full access to their API service. By submitting pull requests based on song title and artist, we were able to acquire lyrics for our previously generated set of songs. The success rate on lyrics acquisition was around 90% across both genres. We then had to subset our dataset to only unique songs, since many songs appeared as number one on the Billboard charts across multiple weeks. We further removed all instrumental tracks and those which MusiXmatch did not have lyrics for. Thus we were left with 626 Adult Contemporary and 440 R&B songs and corresponding lyrics.

4 Feature Engineering

Since we aimed to engineer useful features for model building, we had to transform each raw text representation of lyrics. Through a variety of preprocessing techniques within the NLTK library, we were able to engineer a multitude of features, eventually settling on seven to move forward with model building. This preprocessing was usually performed with operations on each set of lyrics in its bag of words representation[1]. Listed below are the final features decided upon:

- Total words
- Average words per line
- Noun density
- Verb density
- Unique word density
- Stop word density
- Curse word density

Because the length of a given song's lyrics will vary widely between songs, the features involving density are normalized as to avoid bias. However, we realized that certain metrics of overall length might also be good predictors of genre and elected to use overall number of words and average number of words per line as features as well. Unique word, stop word, noun, and verb densities were included as exploratory variables. Our group had no prior intuition as to whether certain parts of speech are more common to different genres. However, curse word density was included with an intuition that this feature would be a strong classifier between the two genres - adult contemporary songs generally do not contain curse words, whereas R&B music can be quite explicit.

Whereas most of the above features were calculated using simple string operations in Python, noun and verb density were obtained in a different manner. In these cases, the nouns and verbs first had to be identified from the text. This was accomplished through the NLTK library. This library uses a pre-existing corpus (Penn Treebank corpus) for part of speech tagging[2]. It is possible to overwrite these features, but because the NLTK library is said to achieve 90% or higher accuracy, we did not see the necessity for writing our own part of speech tags.

Principal component analysis was performed on the set of derived features with the hopes of reducing dimensionality of the feature space and identifying important predictors. However, we were able to conclude that none of the features majorly explain the variance throughout genres and it was best to include all seven features in further analysis. Select plots and implications of PCA can be found in Appendix A.

Two features stood out as better classifiers across these two genres than the rest - total words and curse word density. Figure 1 shows the number of words plotted for each observation in the training set. We can see a clear separation in the mean and variance of this feature across genres, signifying that this feature may be a good classifier of genre.

[1] The bag of words representation refers to the breakdown of text into an unordered list of words comprising the text. No component of context is kept from the original document.

[2] A complete list of part of speech tags can be found here, http://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html.

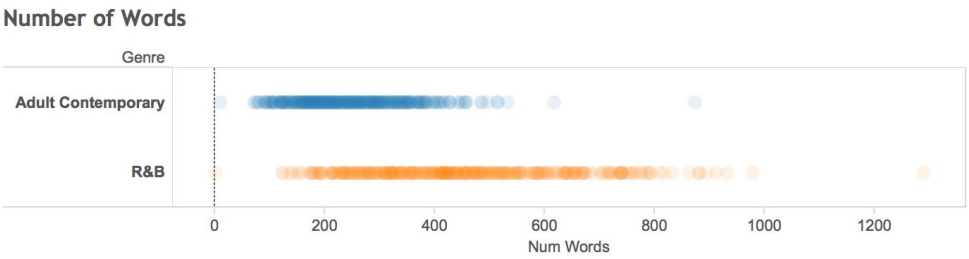

Figure 1: Total lyric words by genre

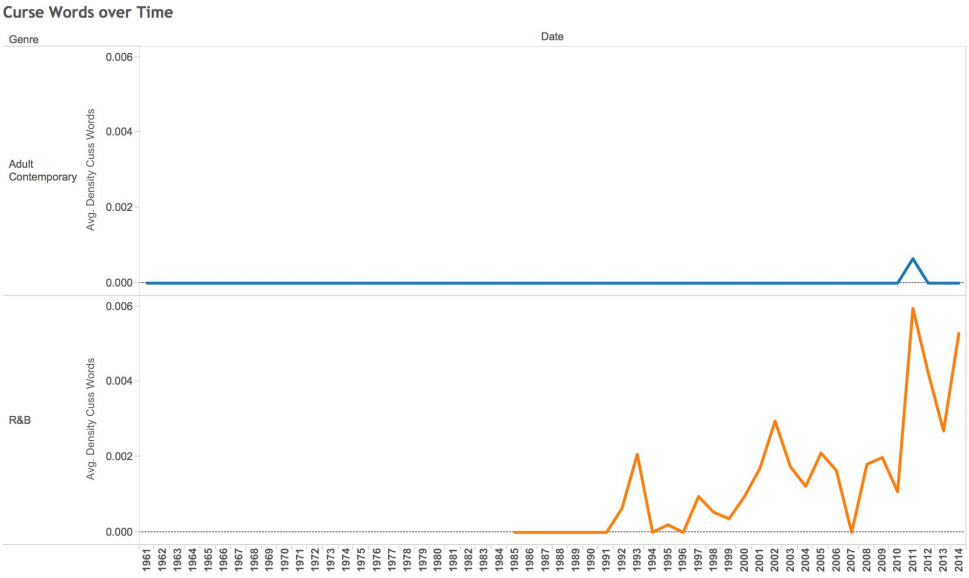

Figure 2: Curse word density by genre over time

The next stand-out feature was curse word density. As we suspected, only a couple songs in the adult contemporary genre contained curse words, whereas many R&B songs did. Recall that we acquired song data by year from the Billboard website. While we did not use year as a predictor variable in any way, an interesting way to visualize curse words was to plot the curse word density of songs over time, seen in figure 2. We notice that the first occurrence of curse words in songs does not even occur for the adult contemporary genre until 2011, while there is a general increase in curse word density throughout time for the R&B genre.

In future model building, these two features ended up being significant at the α=0.05 significance level when used in logistic regression and also were flagged as important predictors in the variable importance plot[3] for the random forest model.

5 Parametric Modeling and Results

The first set of models we developed were the parametric machine learning algorithms implemented using the Sklearn module in Python. All models were trained using 95% of the original data and tested on a 5% holdout sample. A distribution of genre over the training and test sets can be seen in Appendix B.

Within the training data, each model was optimized using a grid search to find optimal hyper parameters using five fold cross-validation. The accuracy was then computed using these parameters on the 5% holdout test set. We chose to optimize accuracy as a scoring metric because there is no penalty for predicting one genre incorrectly over another - we only wish to maximize the number of correct

[3]The variable importance was measured in terms of the increase in gini that would result if the variable was removed from the model.

predictions. We would also like to note that runtime was not a consideration for any models, as our training set was relatively small.

The following table shows the accuracies across all models built, including optimal hyper parameters implemented.

Model	**Hyperparameters**	**Accuracy**
Logistic Regression	L2 regularization	72.2%
Support Vector Classifier	C = 1.0, γ = 1x10^{-5}	74.1%
Random Forest	500 trees, splits based on gini index	76.0%
AdaBoost	100 estimators, learning rate = 1	76.0%
K Nearest Neighbors	19 nearest neighbors	79.6%
Stochastic Gradient Boost	200 estimators, learning rate = 1	83.3%

6 Nonparametric Modeling and Results

While feature engineering can generate important insights into classification, nonparametric approaches often lend themselves more nicely to text analytics. Consider a supervised learning problem in which we have a set of documents whose classes are known. Then, given a new document, we wish to calculate the probability that it belongs to a certain class. Here, the nonparametric Naive Bayes algorithm can be used to assign this document the higher probability class, given that document's words. In this case, the parameter space can increase with the presence of new words in documents, hence taking into account more information than a parametric approach might yield.

We decided to implement the Naive Bayes algorithm on our data set, considering the probability that words occur in the lyrics of songs across genres. We further wished to move past the bag of words approach and take into account some context of the lyrics. This was achieved by utilizing the document's N-gram representation along with individual word occurrence probability. Due to the restrictions with sklearn's vectorizers, we were worried about being able to develop custom blends of N-grams. With this in mind, we instead chose to use the Naive Bayes implementation Brendan created last semester[4]. With this algorithm, we were able to use an existing framework for normalizing text, and had a bit more control over how words were vectorized.

Additionally we decided to implement our own Term Frequency Inverse Document Frequency (TFIDF) classification algorithm, leveraging NLTK's stemming framework for generating words from a set of lyrics.

Accuracy for both algorithms was computed in the same way as the parametric approaches. While TFIDF did not perform as well as we had hoped, our implementation for Naive Bayes competed toe-to-toe with the nighest performing parametric model, outperforming the other models, as shown below:

Model	**Hyperparameters**	**Accuracy**
TF-IDF	None	67.0%
Naive Bayes	Custom blend of 1-, 3-, 5-grams	81.3%

7 Challenges

The main challenges of this project occurred in the acquisition process, feature generation, and Naive Bayes implementation.

While using an API to acquire lyrics gave us a generally clean set of text to work with, this process had some limitations. Namely, the API either produced multiple results per song/artist request or did

[4]Code and full documentation can be found on our project github, goo.gl/YVtW2u

not produce any. The former case occurred because certain songs have multiple versions produced. There were a few undesirable cases which returned instrumental tracks (with no lyrics) instead of the correct version which would have appeared on the Billboard chart. In an effort to make scraping code robust, we could not account for these individual cases and therefore had to throw out multiple observations due to this occurrence. The second case occurred when the API service did not contain lyrics for a particular song in their archives. Again, these observations had to be removed and our data set was further reduced.

Initially we spent some time attempting to bypass the API service and instead scrape lyrics ourselves using BeautifulSoup; however, formatting issues across lyrics yielded even messier results than the API service, and we proceeded using our initial method and a slightly smaller data set.

While feature generation presented less technical challenges, it raised more discussion of importance among features. We spent a significant amount of time researching which features are commonly looked at in text analysis and weighing pros and cons of using those features for genre classification. The majority of features generated were exploratory in nature - we had no reason to believe one genre would be differentiated from another using this feature. The only feature we predicted would do an exceptional job at classification was curse word density. Although we considered using similar metrics, such as indicator variables marking occurrences of specific words, we chose not to proceed in this fashion. We feared that selecting words based on our prior familiarity with the genre would bias our results and inhibit us from extending the project to classify between genres with which we were less familiar. For example, since I am very familiar with the genre of country music, I could speculate that a country song is more likely to contain the words "boots", "truck", and "beer". Therefore an indicator variable of whether an unknown song's lyrics contain one of those words would be a good classifier of a country song over another genre. However, being very unfamiliar with the genre of heavy metal, I would not be able to generate a similar indicator to classify songs of that genre.

In order to avoid bias as described above, we chose to leave those features out of the parametric models. We noted that the nonparametric approaches should capture this information in a better manner when classifying based on the probability of words occurring in different genres.

8 Further Extensions

We are also interested in leveraging this data set beyond the scope of a three week project. The most natural extension to our work would be to generalize beyond two classes. We would like to generate a more complete set of lyrics within a multitude of genres and train classifiers to predict among them. Extending the nonparametric models to this case would be trivial; however, the parametric models would utilize a one-versus-all or one-versus-one approach to predict for a non-binary classification problem.

Additionally, we would like to examine the differences between commercially successful and non-commercially successful songs. This would involve acquiring data in a completely different manner because we would need a variety of popular as well as unpopular songs across genres.

Finally, we would like to attempt author identification pertaining to songs, leveraging available meta data to examine authorship within the scope of artists, genre, and music in general.

9 Conclusions

By implementing a wide variety of both parametric and nonparametric models, we are able to evaluate machine learning techniques on a unique text analysis problem. We have found that Stochastic Gradient Boosting and Naive Bayes allow for the highest accuracy in classifying songs between Adult Contemporary and R&B genre, achieving around 83% accuracy. We can also conclude that of the features we engineered, the most important differentiating between the two classes are the total number of words in the song and the density of curse words.

We acknowledge that with further research into relevant feature extraction and engineering, more accurate results could most likely be obtained for classification between R&B and Adult Contemporary songs. However, overall we are pleased with the results we have obtained for the scope of a three week

project.

A Principal Component Analysis

Below are various plots demonstrating the principal component analysis we performed. PCA was attempted to identify a number of important features which would distinguish the two genres. From this process we can infer that the features of total words, noun density, verb density, and unique word density are responsible for explaining the majority of variance in the data. Figure 1 below demonstrates this fact, where we can see that the first four features explain around 97% of the overall variation in data. Figures 2 and 3 also show a distinct separation between genre along their axes.

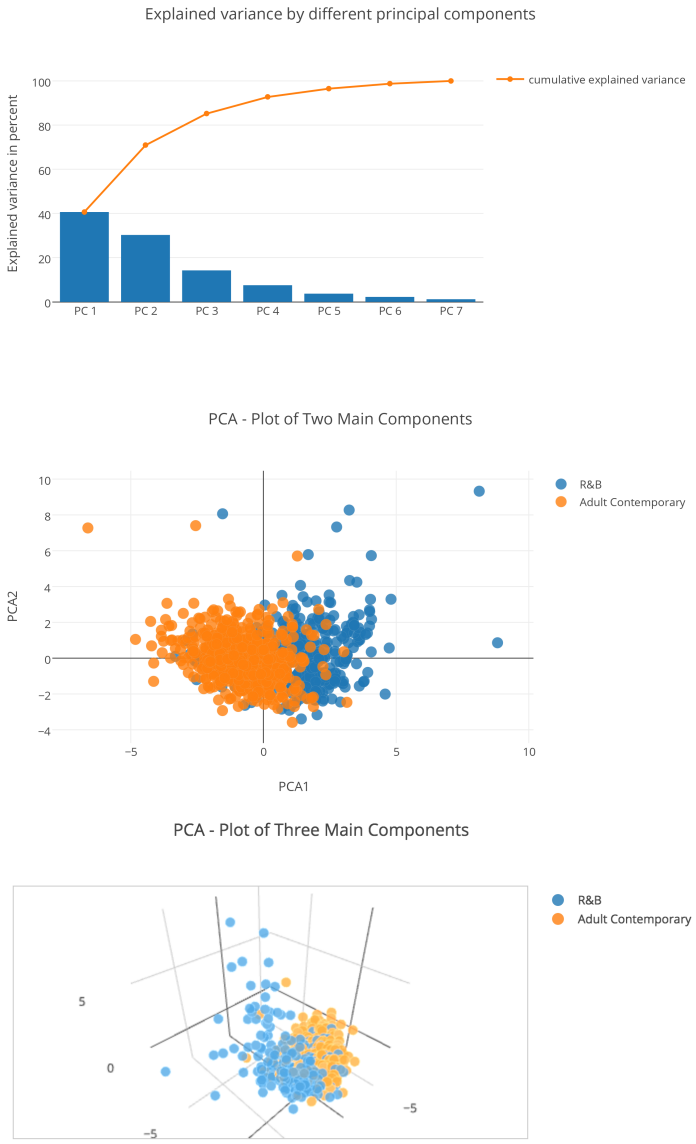

Figure 3: Variation explained by Principle Components

B Exploratory Data Analysis

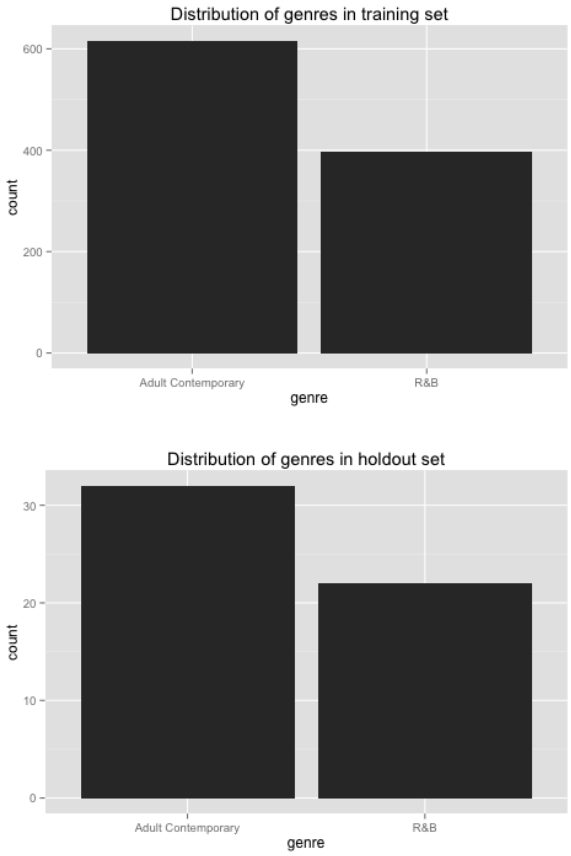

Figure 4: Training and holdout sample genre distributions

C Ranked Model Results

Model	**Hyperparameters**	**Accuracy**
TFIDF	None	67.0%
Logistic Regression	L2 Regularization	72.2%
Support Vector Classifier	C = 1.0, γ = 1x10^{-5}	74.1%
Random Forest	500 trees, splits based on gini index	76.0%
AdaBoost	100 estimators, learning rate = 1	76.0%
K Nearest Neighbors	19 nearest neighbors	79.6%
Naive Bayes	Custom blend of 1-, 3-, 5-grams	83.1%
Stochastic Gradient Boost	200 estimators, learning rate = 1	83.3%