Genre prediction from lyrics.com

Problem

According to Hypebot.com, in every 24 hours in 2018 roughly 20,000 musical tracks were uploaded to major streaming services. That amounts to approximately a million tracks every six weeks. This enormous amount of data is too much for any single human to sift through in a lifetime. Assuming three minutes a track, a million tracks would take a single individual almost six years to listen to.

In order for music to be made available to individuals it is necessary to perform some degree of automated classification. The task of genre classification (assigning similar music to groups) is one of the most fundamental of these tasks. Broad grouping of music along stylistic similarity allows increased ease of discovery of new artists for listeners, and for marketers it allows identification of potential consumers for music from new bands.

Data

Lyrics.com is a website featuring lyric data from roughly 150,000 artists across 1.3 million songs. Of these 1.3 million songs, roughly 15% have genre tag classifiers. Additional tagging is available for subgenre or audience-specific tags.

In order to acquire the lyric information, a web scraping application was created using BeautifulSoup and the Requests library for python. The 1.3 million songs represented on Lyrics.com were downloaded and stored in a SQLite database object for further analysis using SQLAlchemy.

Exploratory Data Analysis

Initial investigation of the data set showed that of 1.25 million songs scraped from the website, 382,600 had associated lyrics. The vast majority of songs represented appeared without lyric content, either because they represented instrumental tracks or because user catalogued lyrics had not been supplied.

Rock	56226
Pop	46089
Jazz	18897
Electronic	17622
Folk, World, & Country	17236
Hip Hop	14935
Funk / Soul	12355
Latin	7190
Blues	5270
Stage & Screen	3512
Reggae	2853
Classical	889
genre	850
Non-Music	367
Children's	349
Brass & Military	26
genres	8
Ce genre de Mec	5
Ce genre	2
Rapgenre	2
Buy This Song	1
Ce genre Lyrics	1
transgenres	1
Le genre féminin	1

Name: genre, dtype: int64

Figure: Genre representation in the lyrics.com dataset

Genre Curation

When considering genre taggings of songs in the database, 134,673 unique combinations of song title and artist were represented and genre tagged. Roughly 38,000 songs had multiple associated genre tags. Multiply tagged songs had as few as two or as many as eighty associated genres. Of tagged genres, Rock and Pop were by far the most highly represented, with Rock representing 27% of all singly-tagged songs and Pop representing 22%.

Focusing on a subset of the seven most highly represented genre tags further reduced the dataset to 94,000 tracks. By analyzing the distribution of these tracks it was possible to determine the degree of overlap between tracks. The resulting analysis showed the Pop tag to be highly promiscuous, associating with all other tags except Hip Hop. Rock was also highly promiscuous, associating with all tags except for Jazz. Of all the combinations the Rock/Pop tag combination was the most highly convolved.

In order to address this issue, the group of tracks was further decreased to the Rock, Jazz, Hip Hop, and Folk genres. This leaves us with approximately 85,000 tracks with a single genre track, roughly half of which are tagged as Rock tracks.

Language Analysis

We can also assess the language of each track, limiting to English-language tracks. The langdetect package provides tools for matching a large number of languages, English among them. By assessing the first fifty characters of each

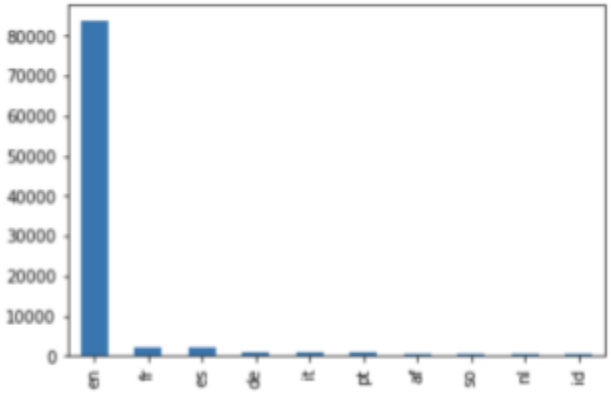

Figure: The genres represented in the dataset skew heavily toward English. Some 10,000 tracks are in other languages.

song we can arrive at a language classification for all tracks in roughly ten minutes of compute time.

This reduces the training set to 74,000 tracks, eliminating 10,000 tracks, most of which are Spanish, French, Dutch, and Italian.

Song Deduplication

In order to check for song lyrics that are represented multiple times in the data, a custom approach was used to iteratively compare sub-blocks of the data. An exhaustive approach using bit vectors to compare all pairwise similarity scores among the 85,000 tracks was found to take nearly six hours to run on one core, but by comparing blocks of 20,000 or fewer records using cosine similarity, this time can be shortened to minutes at the cost of the possibility of a small number of duplicate records remaining in the dataset.

Data is first sorted by track name, then blocks of 20,000 records are compared for pairwise similarity. This process is then repeated multiple times until the number of duplicates converges. Validation with exhaustive duplicate detection shows this method to remove 99% of duplicates in the lyrics.com catalogue. In cases where duplicates were detected, the earliest dated entry was retained and all other entries were discarded.

The resulting set of tracks comprises 41,422 unique tracks, indicating that in fact more than half of the data is duplicated, representing four genres. In previous runs, Jazz was found to be the most highly duplicated of all the genres, with nearly three quarters of tracks comprising covers or standards.

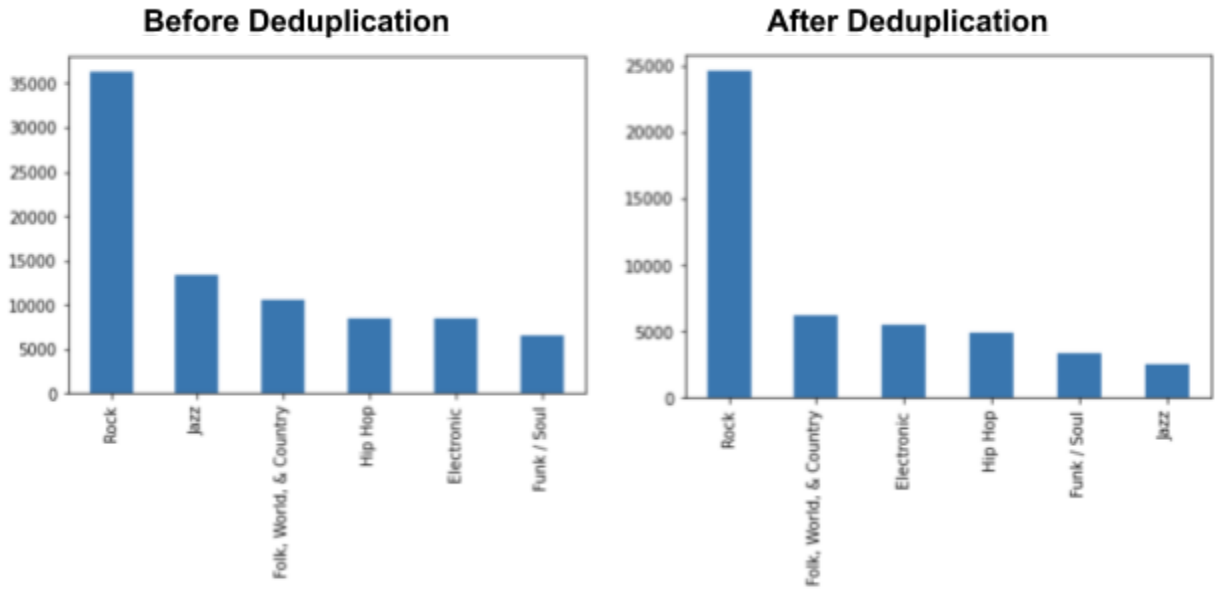

Figure: Genre counts before and after deduplication. Jazz decreases the greatest amount, while Electronic has the lowest ratio of duplicate tracks.

Investigating Cleaned Data

We can interrogate the data now looking for interesting trends between the four genres of interest.

When considering the length of lyrics in each song, Rap has by far the longest average song length at 2800 characters, while Jazz is the shortest at roughly 750.

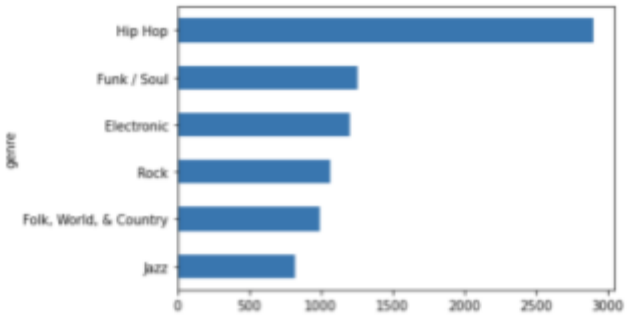

Figure: Hip Hop has the longest lyric length by character count, while Jazz has the shortest. The majority of songs range between 1000-1500 characters.

The number of unique words in each genre roughly approximates this distribution. In order to compensate for the Rock category's much larger data set, ten bootstrap samples of 2000 songs were taken from each genre and the number of unique words were computed. Rap again predominates with an average of roughly 70,000 unique words per 2000 song subset, while Rap and Electronic come in at an average of 22,000 unique words per song analyzed, and Folk comes in last with 18,000.

We can also look within each genre to see which artist has the largest number of unique songs associated with their name in the Lyrics.com dataset. Elvis leads the pack as the biggest contributor to the Rock genre with 149 distinct songs; Johnny Cash leads Folk with 103; Depeche Mode leads Electronic with 62; and Jay-Z is the top Rap contributor with 52 songs.

Word Clouds

Finally, to provide an overall visualization of the differences between the various genres, we can use the wordcloud python library to create word cloud representations of each genre. From these we can see some intuitive words that are associated with each, including Rap's favorite pronoun.

Figure: A subset of genre word clouds. "Love" and "know" feature prominantly, as do words like "got" and "want". Hip Hop features more slang terms and vernacular phrasing.

The Initial investigation using observational visualizations and EDA demonstrates that the dataset has limitations associated with genre tagging and duplicated tracks. Despite this, we can infer a number of interesting things about the data.

The discrepancy in Rap's character length relative to other songs is not unexpected, but is striking. Jazz's low complexity may indicate this class will cause difficulties when attempting to classify. Regardless, this discrepancy is a useful meta-feature to consider.

genre	artist	
Electronic	Depeche Mode	67
	Madonna	55
	Pet Shop Boys	51
Folk, World, & Country	George Jones	103
	Johnny Cash	92
	Marty Robbins	71
Funk / Soul	James Brown	54
	Aretha Franklin	51
	Earth, Wind & Fire	47
Hip Hop	Jay-Z	53
	Nas	50
	Eminem	40
Jazz	Frank Sinatra	111
	Nat King Cole	66
	Ella Fitzgerald3	62
Rock	Elvis Presley	145
	The Rolling Stones	104
	Elton John	81

Figure: Top contributing artists by genre. By observing the three top artists in number of unique songs within each genre label, we can form intuition about what constitutes a song in each classification group.

We can also observe differences in the word clouds generated from each artist. Core vocabulary differences can be observed, but the similarities are again the most striking thing about these visualizations. Hip Hop again seems to assert its individuality from the older music styles.

Application of Machine Learning Techniques

Now that we have performed data cleaning and performed an exploratory analysis of the data, we can turn to the task of creating a classifier. The remaining data set comprises roughly 40,000 unique tracks with annotated genre information. To create our classifier, we will be considering three of the remaining genres: Folk, Hip Hop, and Electronic. We will omit the Rock and Pop genres (we will return to this decision in more detail later in the analysis).

In order to create a language model to classify the genre data, we must first make several decisions related to how we will preprocess the text information. Tokenization refers to the processes of creating representative vectors of information that can be consumed and applied by machine learning models, and lemmatization refers to the process of ascertaining and curating morphological features of words that may be of use to a classifier.

First-pass investigations into the data set showed that classifiers performed significantly worse when stop words like “the” or “and” were removed from the dataset, implying that these may encode genre-specific information that is valuable to determining the genre of interest. Because of this, stop words were retained in the corpus.

Additionally, the poetic nature of lyric data leads to grammatical structures that may not be observed in standard text, and because of this lemmatization seemed to increase the complexity of data while providing minimal benefit. For this reason, we do not make use of stemming or part-of-speech tagging in the data.

As such, the classification task becomes much more straightforward. We can begin by analyzing the word complexity of the data by seeing the relative representation of various ngrams in the dataset. An ngram is a grouped set of words, meant to form a composite meaning that may be distinct from the individual parts. For example a “pancake house” has a very different meaning than a “pancake” or a “house”, and therefore will be useful in predicting different contexts than the constituent words.

We can start by looking at various ngram lengths and characterize the amount of coverage each set represents. Due to computational constraints using consumer hardware, we will have to limit our dataset to a relatively small number of tokens. As such, we wish to know what range of ngrams we should compute in order to get the best tradeoff of computational time for our use case.

The provided figure shows the plot of the number of ngrams computed, and the relative frequency of each of those levels of complexity. Each line represents the number of ngrams that

are present at each threshold (counts of 25, 50, and 100). As we can see, the highest coverage represented is from the ngrams in the 2-3 range. Comparatively fewer ngrams are present at a level of one or four ngrams, and virtually no ngrams of 5-6 words are represented in the larger set.

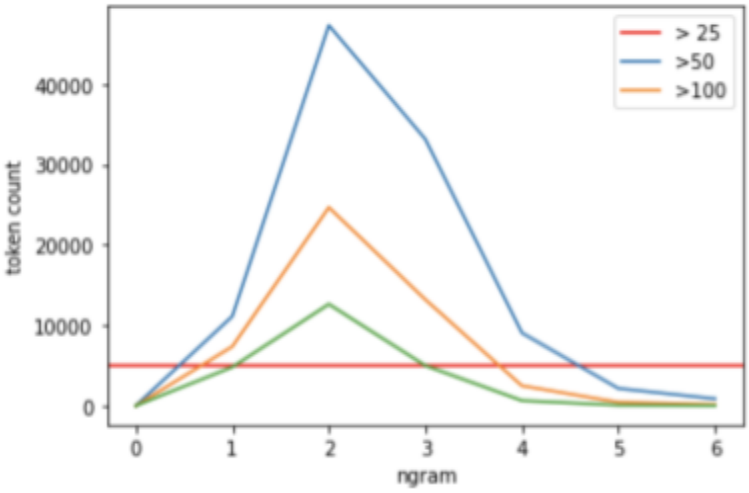

Figure. Here we plot the number of ngrams at each complexity level that reach various count thresholds. As we can see, tokens of length 2-3 have the greatest representation in the data, and are likely to predominate the classifier. Tokens of length greater than four are less frequently repeated in the data.

We can also reasonably conclude that since we did very little removal of stopwords, that some subset of overrepresented ngrams likely reflect phrases like “in the” or “on the” or similar short phrases that provide fairly little information. However, since the context and construction of lyrical information does not necessarily reflect standard English usage, there may still be useful information conveyed in these structural tokens, and the accuracy gains from inclusion of stopwords may reflect that.

We next use the TF-IDF transformer provided with scikit-learn to create a vector representation of our corpus. TF-IDF adaptively weights each term based on its representation in each individual document relative to its representation in the full corpus. The resulting sparse matrix creates a linear algebra representation of our data where each vector is a high-dimensional location that can be compared with other vectors that may have similar features.

The TF-IDF transformer is built using a range of ngrams of length 1-3, reflecting our earlier analysis. We omit stopwords and take a relatively small subset of terms - the top 1500 tokens by representation in the corpus.

Clustering Analysis

In order to gauge the degree to which the data falls into distinct spatial groupings, we can apply a cluster analysis. We use the KMeans algorithm to attempt to spatially partition the data, evaluating a range of various cluster sizes for best fit to the data.

The results imply that the data is not easily separable in Euclidean geometry. A high silhouette score represents good separation of classes, and a low inertia value represents the

same. In constructing these plots, it's generally hoped that one can observe an "elbow" or "notch" at which one value is maximized. However, we instead see a slow, steady decline in inertia as clusters increase, and our silhouette score is maximized at two clusters, which may indicate that the data represents many smaller clusters or disjoint subspaces, if it is separable at all.

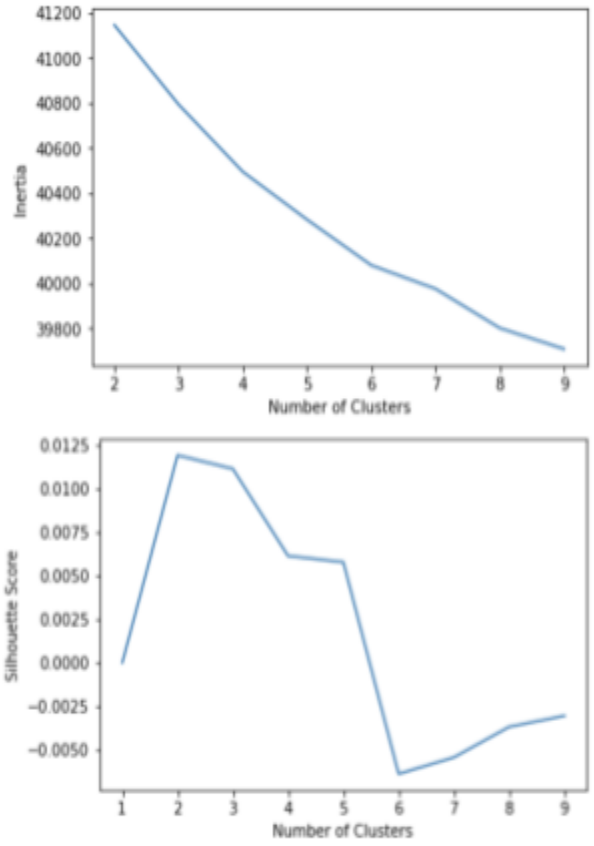

Figure. Clustering analysis is inconclusive. An inflection point is not detectable in our inertia values, and the silhouette score does not give a clear indication that there is an ideal number of clusters into which to divide the data.

Classifier Construction

Given the data that we have observed to this point, we turn to classification methods that thrive in high dimensional space and highly nonlinear surfaces. If the data's classes are able to be separated, the best bet is to consider classifiers that excel at separating strongly intermingled data.

We apply four different models to the data in order to judge which represents the best tradeoff of performance to computational cost: Naive Bayes, SGD, XGBoost, and a Neural Network classifier.

We apply each model and observe the range of classification accuracies. All three models are able to distinguish Hip Hop from the other two, but they face varying degrees of difficulty in discerning the difference between Electronic and Folk. Intuitively this is easy to understand. Rap has a very different vocabulary than other modern genres, and we have already catalogued a distinct difference in the number of words in the average rap song when compared to other genres. Both of these tendencies likely contribute to the classifiers' success singling out Hip Hop.

All four classifiers also achieve relatively similar results. Each achieves roughly 75-80% accuracy. XGBoost performs the best, with a peak accuracy of 80%, and similar metrics for precision, recall, and F1 score. The linear SVM SGD classifier also performs surprisingly well, with around 78% accuracy. The Neural Network and Naive Bayes methods lag behind at 77% and 75% accuracy, respectively.

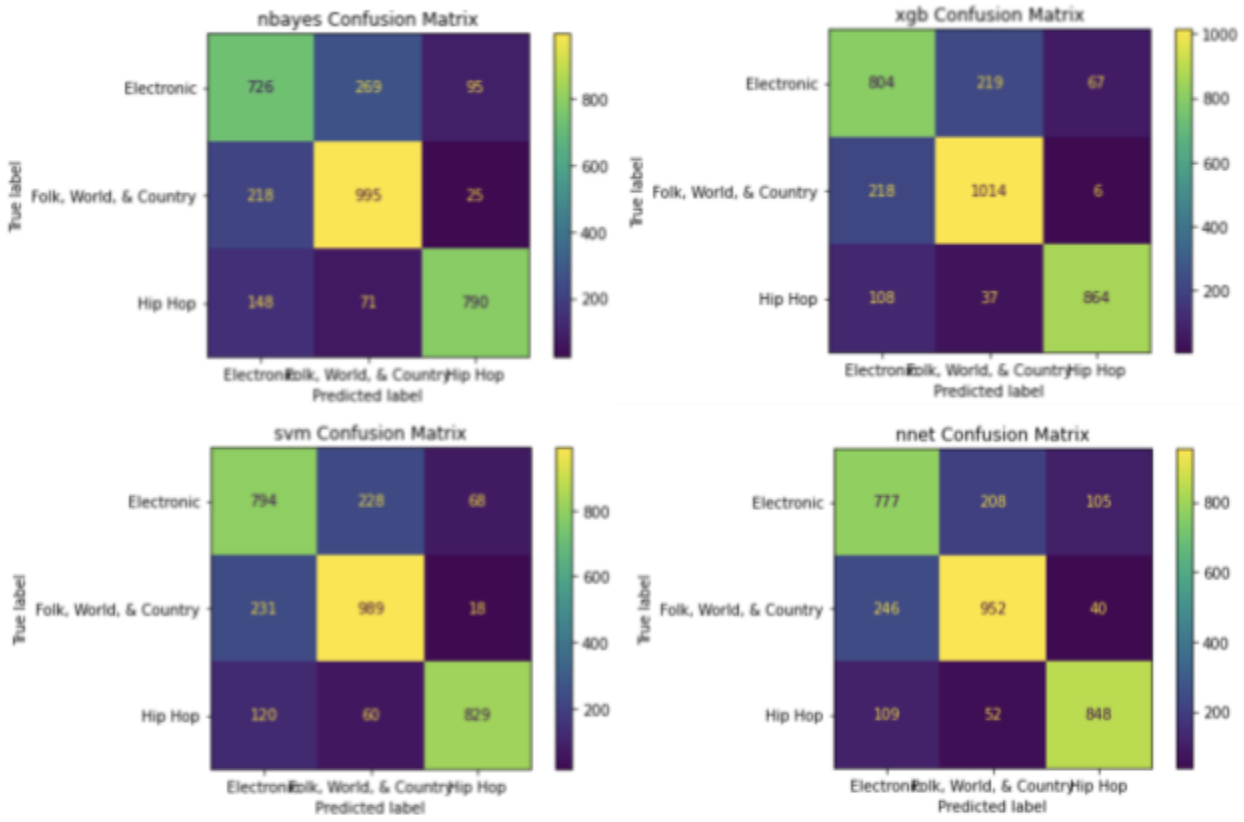

Figure. Confusion matrices for each classifier give an intuitive measure of the level of difficulty faced in separating each class. We can see that Hip Hop is the easiest to classify, while Electronic seems to be the most difficult, and Folk and Electronic are the most similar to one another.

When we consider the computational burden of training each classifier, the tradeoffs between the XGBoost and SGD model become more interesting. While both obtain a relatively similar result, the SVM trains in 1/30th of the time that is required by XGBoost. This computational performance boost may be of a great deal of interest if we are dealing with data at scale, where we may have millions or billions of examples. If it is possible to increase the performance of the model only a couple of percentage points, we may be able to make this performance discrepancy disappear.

Model Optimization

In order to evaluate whether boosting the performance of the SVM is possible, we can perform a detailed grid search of potential parameters. The short time the classifier takes to train plays to this advantage, as we can attempt hundreds of parameter combinations in the time it would take to train just a handful of XGBoost models.

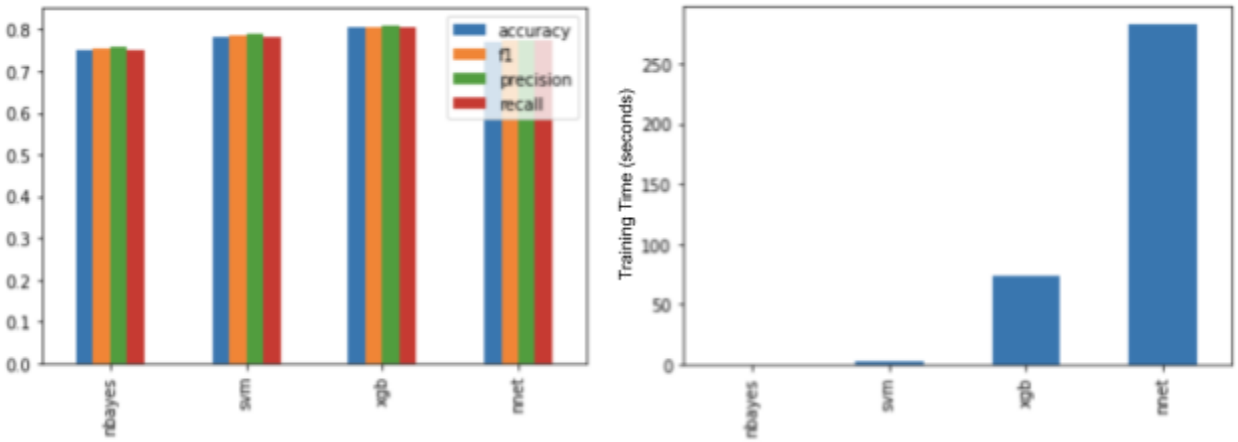

Figure. We can see that while the performance of the classifiers is relatively similar, the difference in training time is stark. The production environment in which our hypothetical classifier is deployed is likely to be a deciding factor in which classifier is best fit to our use case.

We focus on a subset of the available parameters - loss (the type of optimization strategy), alpha (a regularization parameter), penalty (the manner in which regularization is applied), and maximum training iterations (the number of epochs to train the classifier using). We then compute the exhaustive grid of all parameter combinations, representing 108 potential models. These models are trained using cross validation in order to minimize the impact of the division of data into training and test data on model fit.

After training, we arrive at the ideal combination of parameters from the grid search. A logistic regression approach with an elasticnet regularization approach performs the most effectively on the data. The resulting classifier accuracy is 78.5%, which represents an improvement over the initial model, but likely not one that is statistically significant or practically meaningful.

A Note on the Rock Genre

I mentioned that I would return to the decision to exclude Rock as a genre from our classification task. It seems odd to exclude this subset of the data as we are already limited in the amount of training data available to us and Rock is the most plentiful of the genres in our training set.

The reason for this is that Rock appears to be an actively uninformative. The addition of the Rock genre to the training set decreases prediction accuracy across the Electronic and Folk genres. Hip Hop still appears distinct, but Rock, Folk, and Electronic become intermingled. This discrepancy seems to indicate that Rock actually represents a catch-all classification, consisting

of lyrical features that are present in many other genres. As such, it may be worthwhile to consider dividing songs labeled as Rock into subgenres, or to characterize the subspaces in which the most overlap between genres classified as Folk and Electronic flip classification when Rock is introduced.

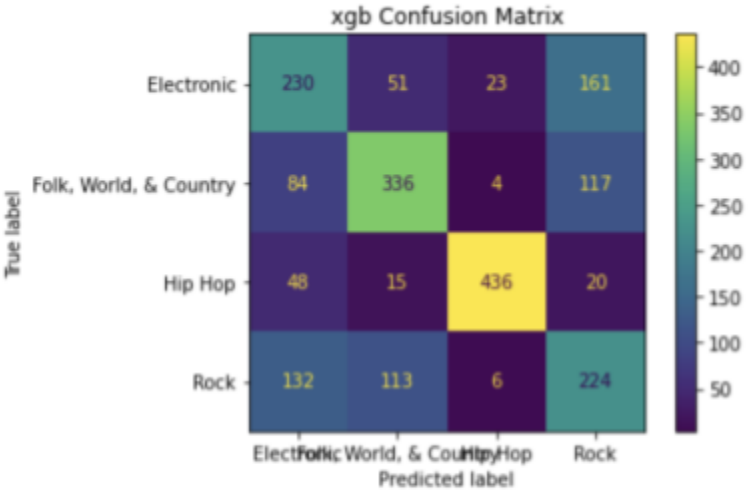

Figure. The confusion matrix for the dataset with the Rock genre included demonstrates that this label is at best somewhat uninformative, and at worst may actively contribute to an overall decrease in classification effectiveness.

Conclusions

This analysis is a small sample of what's possible in textual genre classification. It's an even more stark demonstration of how extreme the reduction of data from a scraping set can be. We started with 1.3 million tracks, but ended up with around 20,000 tracks from which to build our classifier. We eliminated roughly 98% of our dataset.

That said, the performance of the SVM and Naive Bayes classifiers on the final dataset is impressive (and fast!). A complicated classifier is not always the most effective - a point that's made even more clear by the fact that the neural network performs almost exactly the same as the SVM even though it takes 300x as long to train.

The failure of clustering coupled with the success of methods that focus on individual word combinations seems to indicate that the decision surface for the genre classes may be highly non-linear or exist in a euclidean space that is different from what can be detected using spatial clustering approaches.

It is also interesting to observe the negative effect that including Rock tracks has on the classification accuracy. Removing Rock increases the ability of the class to classify both Electronic and Folk, suggesting that the two have features independent of one another, both of which are present in Rock. This may indicate that "Rock" is an ineffective classifier, requiring greater granularity in genre labeling.

For additional investigation, it's likely that a larger training set will be required. Approaches using neural-network-based approaches like word-to-vec and LSTM can be extremely informative, but require datasets on the order of 10 to 100 times the size of the annotated portion of this dataset.

All together, this data scarcity reflects why the problem of genre classification is so important for current day efforts in music recommendation and marketing.