# Creating a Feature Vector to Identify Similarity between MIDI Files

Joseph Stroud
2017 Honors Thesis
Advised by Sergio Alvarez
Computer Science Department, Boston College

## Abstract

Today there are many large databases of music, whether used for online streaming services or music stores. A similarity measure between pieces of music is extremely useful in tasks such as sorting a database or suggesting music that is similar to a selected piece. The goal of my project is to create a feature vector using only the actual musical content of a piece, ignoring metadata like artist or composer names. This feature vector can be used to measure distance between pieces in a feature space. To analyze the information contained in the feature vector, clustering and classification machine learning tasks were run with the goal finding pieces that share musical characteristics, whether they are grouped together into a cluster or classified as the correct genre.

## Contents

# 1. Introduction

## 1.1 Overview of Feature Vector Approaches to Music

### 1.1.1 Musical Similarity

What does musical similarity mean? People use genres to group similar pieces of music. Unfortunately for computation, genres are very poorly defined. There is no common factor determining what determines a genre--it can be anything from geographic location to time period to technical musical requirements. In addition, a given piece of music usually fits into many different genres and can be classified into a different genre based on the cultural context [10]. This means that using genre as a descriptor of musical similarity requires interpretive decisions on the part of the person assigning genres to pieces. Nonetheless, because genre is such a widely known concept, in this project I use genre as one route for evaluating whether similar pieces are correctly being represented as similar.

### 1.1.2 Feature Vectors

To computationally analyze a piece of music, it must be placed in a format easy for a machine to work with. In order to analyze a dataset using machine learning techniques, each instance in the dataset must be described by assigning it a set of values that represent certain features, often known as a feature vector. These features should have some relevance to the knowledge that the machine learning algorithm is trying to uncover [2].

Research has already been done on feature vectors that describe pieces of music, and in general the features fall into several categories. Some categories that are used are timbre features, melodic and harmonic features, and rhythmic features. Timbre is the difference in sound between two instruments playing at the same volume on the same pitch. Melody is a series of different pitches perceived together, while harmony is the use of pitches at the same time. Rhythm has to do with the timing of notes [12]. Other attributes that can be used are artists, composers, or albums associated with a piece [10]. Most attempts at creating features, including features describing rhythm, melody, and harmony, are mainly mathematical functions of the sound waves created by musical performance [12]. These include features describing the energy and spectral shape of the sound wave, features analyzing the distribution of pitch, and features attempting to measure periodicity [3][4][5][12].

In this project my goal was to use features derived exclusively from the musical content of the piece, ignoring data such as artist, composer, or album names. I used features describing aspects of the harmony, pitch, rhythm, and timbre. These included both simple statistical measures and music theory analysis.

### 1.1.3 MIDI Files

In order to make it simpler to examine these features, I used MIDI files as my data set. Unlike most music files, MIDI files do not store sound waves but instead store a sequence of events [8][9]. Most of the information is stored in note on and note off events, which are associated with a specific instrument, a pitch, a volume, and a time stamp. This means that no algorithms are needed to extract pitch, timing, volume, or instrumental part information from the music file. Instead, I could focus on rudimentary musical analysis when creating features.

## 1.2 Clustering, Classification, and Feature Selection

In order to evaluate feature vectors, I used clustering and classification algorithms, which take a dataset of instances, each represented by a feature vector, as input and provide a model of the dataset based on information in the feature vectors as output. Clustering and classification algorithms have distinct aims.

### 1.2.1 Clustering

Clustering algorithms attempt to group the instances in order to accomplish three goals: each data instance should be close to instances in the same cluster, each instance should be far from instances in different clusters, and there should be a relatively small number of clusters [1][2]. In this project I used the k-means clustering algorithm to cluster my data. The k means algorithm takes a number k as input and produces k clusters as output, where each cluster is defined by its center, which is a vector in the feature space. The algorithm assigns each data instance to the cluster with the closest center and then recomputes the center for each cluster after all instances are assigned. It repeats these two steps until no data instances change clusters after the centers are recomputed [1][2].

### 1.2.2 Classification

Classification algorithms attempt to predict the predetermined class of an instance (e.g., the genre of a musical piece) using its feature vector. This is accomplished by training the algorithm using a training set, so that the algorithm infers relationships between the attributes of the instance and its class. The accuracy of this uncovered knowledge can then be tested by classifying a separate test data set [1][2]. In this project I used the logistic regression classifier, which uses the logistic function to estimate the probability that a given instance belongs to each class.

### 1.2.3 Feature Selection

In order to improve the classification and clustering output, I used some feature preparation methods in order to select the features that contained the most musical information. Feature preparation is useful for two reasons: it can cut down on the number of features, which reduces runtime, and it can eliminate noise in the data, leading to more accurate outcomes. One method I used was to select features that were highly correlated with genre. In order to measure correlation, I used the Pearson correlation coefficient, which computes a score between -1 and 1 describing how positively or negatively correlated two variables are, with no correlation getting a score of 0 [1][6]. I also used principal component analysis to prepare my features. Principal component analysis transforms a set of vectors to produce another set of vectors (principal components) which are linearly independent and linear combinations of the original set. These vectors can be used as features in a new feature vector. The set of all of the principal components captures the underlying data exactly. A given number of principal components will capture a greater portion of the variance in the data than any other set that contains that same number of vectors. The number of principal components to be retained can be selected by deciding what fraction of the total variance is to be captured [1][6].

## 2. Methods

### 2.1 Data Set

I tested my features on a dataset made of 165 MIDI files representing unique pieces of popular music. I assigned five different genres to these pieces: Country, Rock, Folk Rock, Pop, and Soul. There were nineteen different artists represented in my dataset, each one with all of their songs placed in the same genre. Of particular note is that one genre, Pop, was made up of songs from only one artist. Every other genre was made up of songs from either four or five artists. The dataset contained between six and twelve songs by most of the artists. Kansas, with 17 songs, and John Mayer, with 25 songs had many more songs than the other artists in the dataset. Crosby, Stills, Nash, & Young had fewer songs in the dataset than all other artists, with only four songs by them present.

| | | | | | |
|---|---|---|---|---|---|
| **Country** | Brad Paisley | Carrie Underwood | Dolly Parton | Lady Antebellum | Luke Bryan |
| **Folk Rock** | Crosby, Stills, Nash, & Young | Elton John | James Taylor | Simon & Garfunkel | |
| **Rock** | Guns N Roses | Journey | Kansas | Styx | The Cars |
| **Pop** | John Mayer | | | | |
| **Soul** | Marvin Gaye | Stevie Wonder | The Supremes | The Temptations | |

*Table 1: Artists present in the dataset of MIDI files, arranged by genre.*

## 2.2 Feature Vector Description

| **Features Used** | **Features Used** |
|---|---|
| Tonality | Augmented Triad Prevalence |
| Chromaticism | Major Seventh Prevalence |
| Time | Minor Seventh Prevalence |
| Pitch Mean | Diminished Seventh Prevalence |
| Pitch Range | Dominant Seventh Prevalence |
| Volume Mean | Other Chord Prevalence |
| Volume Range | $\text{Timbre}_0$ |
| Note Density | $\text{Timbre}_1$ |
| Average Note Duration | Electronic vs Acoustic |
| Major Triad Prevalence | Number of Instruments |
| Minor Triad Prevalence | 1st Quartile Note Length CDF by Instrument |
| Diminished Triad Prevalence | 3rd Quartile Note Length CDF by Instrument |

*Table 2: Features created for this project. There are twenty one individual features, as well was another two features that were replicated for each of the 128 MIDI instrument types for a total of 278 features. These features were automatically calculated from a MIDI file using a Java program.*

### 2.2.1 Tonality and Chromaticism

I used two features relating to the key of a piece: tonality and chromaticism. Tonality refers to the key of the piece. A key is a specific set of pitches used to construct a song, and keys are split into the two major categories of major and minor. I found the key of a song by finding the number of notes in a given song that are in each key, and selected the key with the most notes. Chromaticism is a measure of how much a song stays within its given key. I measured chromaticism by finding the number of notes outside the dominant key and dividing that by the total number of notes in the song.

### 2.2.2 Time

For the time feature, I found the time stamp of the end of the final note, which was measure in microseconds. Because MIDI is designed to be used for real time synthesis of sound from multiple electronic instruments, it has a very high resolution of time. In order to make this feature fit between zero and one, I divided it by the constant $3*10^9$.

### 2.2.3 Pitch Mean, Pitch Range, Volume Mean, and Volume Range

MIDI represents both the pitch and volume of a note as integers between 0 and 127, so I found the mean pitch and volume by averaging the pitch and volumes for every note. Because I used the mean value as a feature, I represented the range as a single number, calculated by subtracting the maximum value from the minimum value. I divided these values by 127 in order to keep them between zero and one.

### 2.2.4 Note Density and Average Note Duration

I also used note density and average note duration as features. I define note density as the total number of notes in the piece divided by the time. I multiplied the note density by 30,000 to keep its value between zero and one. This is such a large number because of MIDI's high time precision. I also found the mean note duration, which was the total duration of every note in microseconds divided by the number of notes. This was multiplied by $10^{10}$ in order to keep most values between zero and one.

### 2.2.5 Chord Percentages

I also included eight features derived from chordal analysis of a given piece. A chord describes a specific set of pitches being played at the same time. The sets of pitches defining chords are not defined as absolutely but rather as specific intervals, or distances between pitches. These eight features kept track of the percentage of time in a piece that the pitches in a given chord were the only pitches currently being played. I used seven chords: major, minor, diminished, and augmented triads and major, minor, dominant, and diminished sevenths. I also used the percentage of time no defined chord was being heard as a feature.

### 2.2.6 Number of Instruments and Timbre

Four other features had to do with the instrument selection during a piece of music. The first feature was simply the number of instruments. Because there can only be a maximum of sixteen instruments in a piece stored in the MIDI format, I divided this value by sixteen in order to keep it between zero and one. I also used a representation of the timbre. The timbre of a sound refers to its auditory characteristics that allow differentiating between two instruments playing the same pitch and volume. Representing timbre presented challenges because MIDI files do not store sound. They store an instrument name and then rely on a soundbank to create the actual sound for a given note from a pitch, instrument name, and volume. In order to represent timbre in my feature vector, I used a timbre space, which represents the sound of an instrument as a point or region in some n-dimensional space [11]. I used a two dimensional timbre space defined by Paolo Prandoni which contains 27 classical instruments [11]. MIDI specifications include 128 possible instruments, so I assigned the other 101 instruments coordinates in the timbre space based on my intuition of their sounds' relation to the sounds defined by Prandoni. Because this process relied on my subjective intuition, it is probable that it was subject to significant error. Additionally, because Prandoni's timbre space only contained classical instruments, I did not trust my intuition to define the timbre of electronic instruments because they have a very different sound quality, so I added a third binary feature describing whether an instrument was electronic or acoustic. To find these three timbre features for a given piece, I averaged the timbre features of all the instruments in a piece.

#### 2.2.7 Note Length Quartiles

I also added features that describe the rhythm of each instrument in the piece. To do this, I used the cumulative distribution function of the note lengths in a piece. A cumulative distribution function gives the probability that a random variable will have a value less than the input to the function. To calculate the cumulative distribution function I assumed that the probability that a random note length would be less than a given value was equal to the percentage of measured note lengths less than that value. For features, I used the first and third quartile of the cumulative distribution function of the note lengths for each instrument type. Because there are 128 instrument types in MIDI, this adds 256 features. However, all instrument types not present in a given piece while have first and third quartile values of zero, meaning that most of these 256 features for any given piece will be zero.

## 2.3 Feature Vector Evaluation

In order to test the features described in section 2.2, I used the machine learning software Weka [7] to do clustering and classification on the dataset described in section 2.1. This contains both a user interface and a Java API, both of which I used in my project. I did clustering tasks using the k-means algorithm and classification tasks using the logistic regression classifier, described in sections 1.2.1 and 1.2.2 respectively. I did clustering on four different feature vectors: one with all features I developed, one with all the features except for the note length quartiles, one with features created by doing a principal component analysis, and one with features selected by the Pearson correlation coefficient to have a correlation with genre. I did classification only on the features selected to have a correlation with genre.

## 3. Results and Discussion

### 3.1 K-Means Clustering with All Features

For k-means clustering with an input vector containing all of the features described in section 2.2, most artists had the majority of their songs in the same cluster (see Figure 1), indicating that some useful information is present. However, there were two clusters which only contain one song and a third cluster which contains a little over half of the songs (see Figure 2). The fact that there are two tiny clusters and one very large cluster which do not seem to match any obvious musical characteristics means that this set of features seems to have as much noise as signal—implying that there is room for improvement.

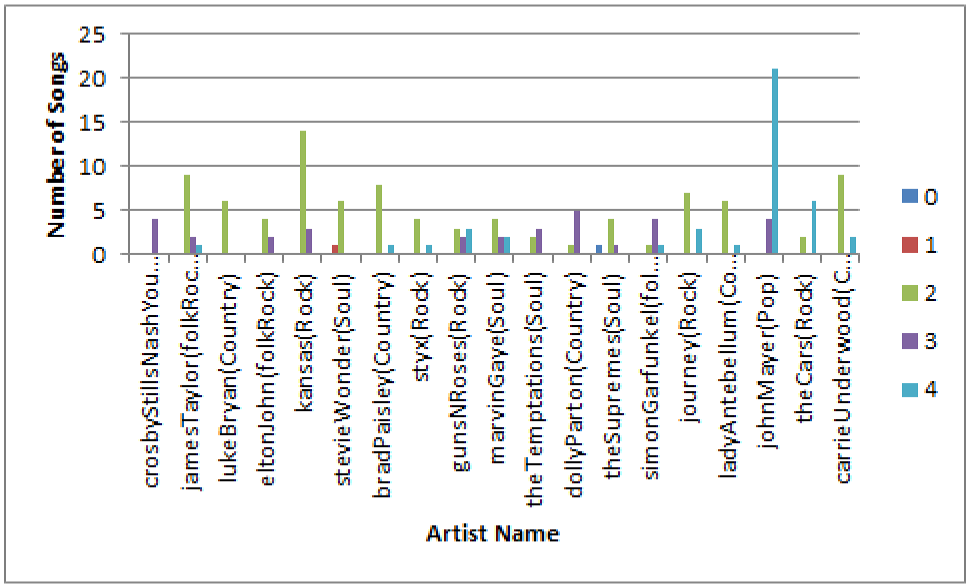

*Figure 1: Number of songs by each artist in each cluster for k-means clustering where k = 5 with all features present*

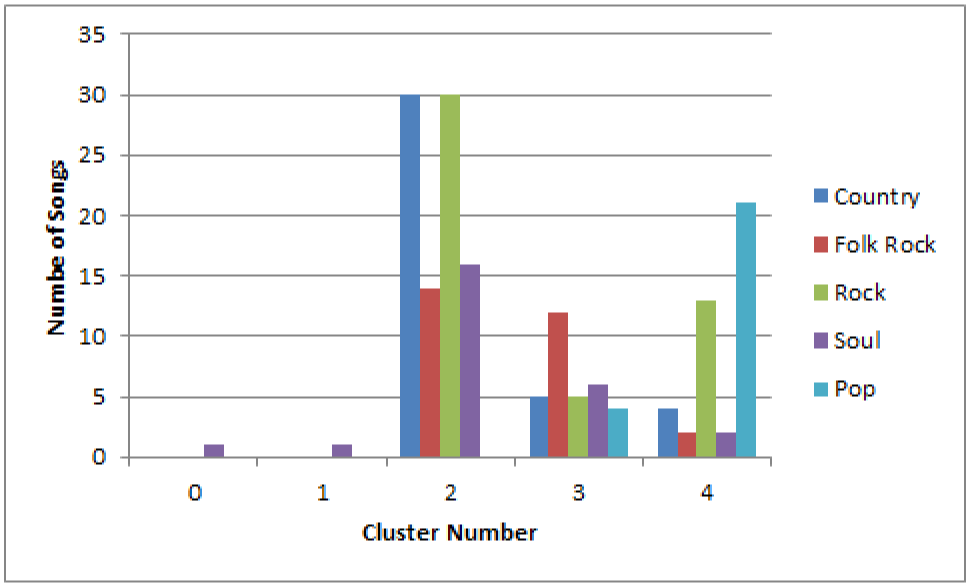

*Figure 2: Number of songs from each genre in each cluster for k-means clustering where k = 5 with all features present*

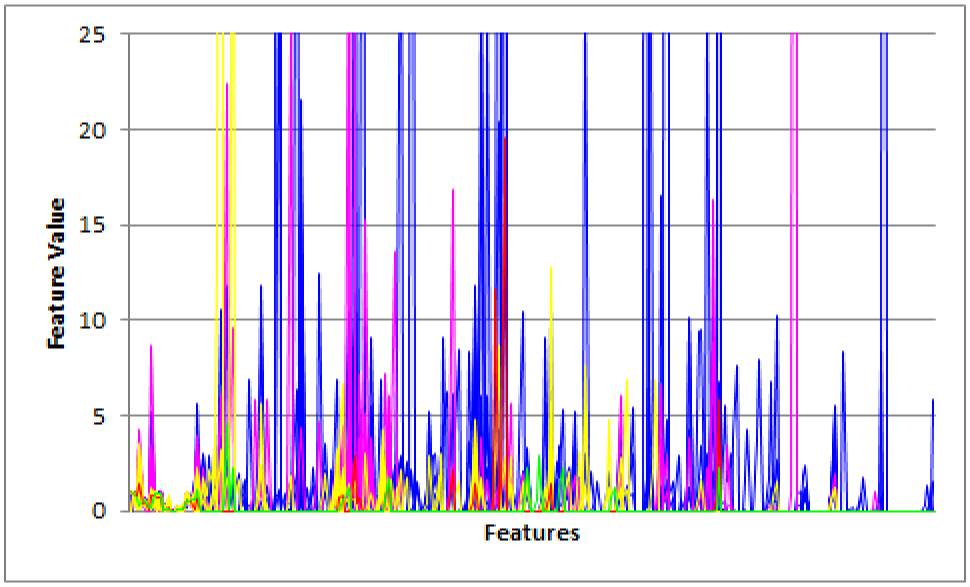

*Figure 3: Parallel coordinate visualization of all 278 features colored by cluster*

## 3.2 K-Means Clustering without Note Length Quartile Features

For k-means clustering without the note length quartile features described in subsection 2.2.7, the results do seem to indicate musically relevant information is present. Three clusters are each dominated by a single genre (see Figure 5), respectively rock, country, and pop. Two of these clusters have specific feature values associated with them: in the cluster with mostly country songs, every song has a large number of instruments. In the cluster with mainly pop songs, every song has a high proportion of electronic instruments. In additional cluster, cluster three is almost entirely composed of slower songs. One example of a song in this cluster is *Scarborough Fair*, by Simon and Garfunkel. The other cluster, Cluster 0, only contains songs in a minor key, while also including every available song in a minor key. Although a musical characteristic, this is not immediately apparent to a listener and so is not a very useful result. However, all five clusters are associated with some musical characteristic, and four of those five clusters contain songs that have shared characteristics that are easily audible. This means that the pieces are arranged in the feature space in a meaningful way.

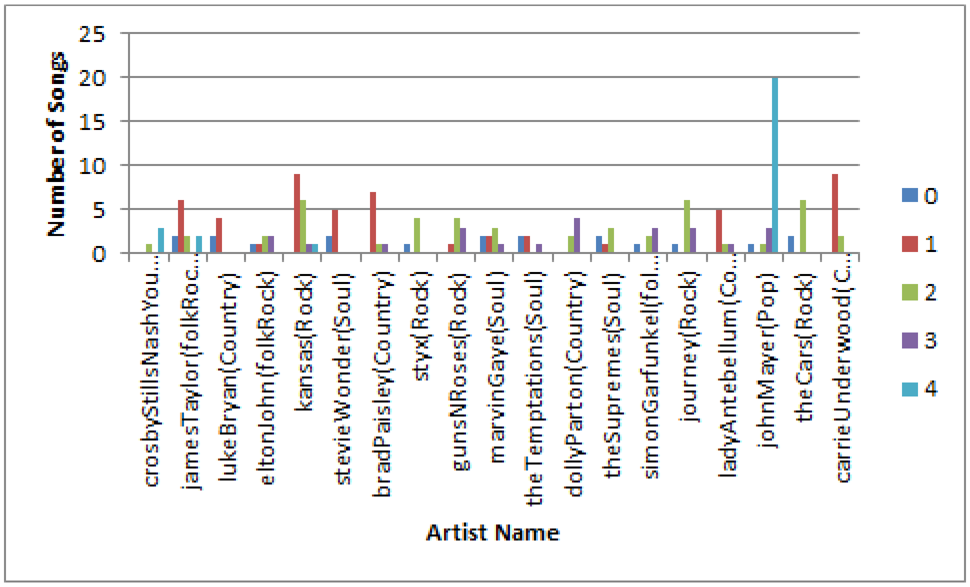

*Figure 4: Number of songs by each artist in each cluster for k-means clustering where k = 5 with no note length quartile features*

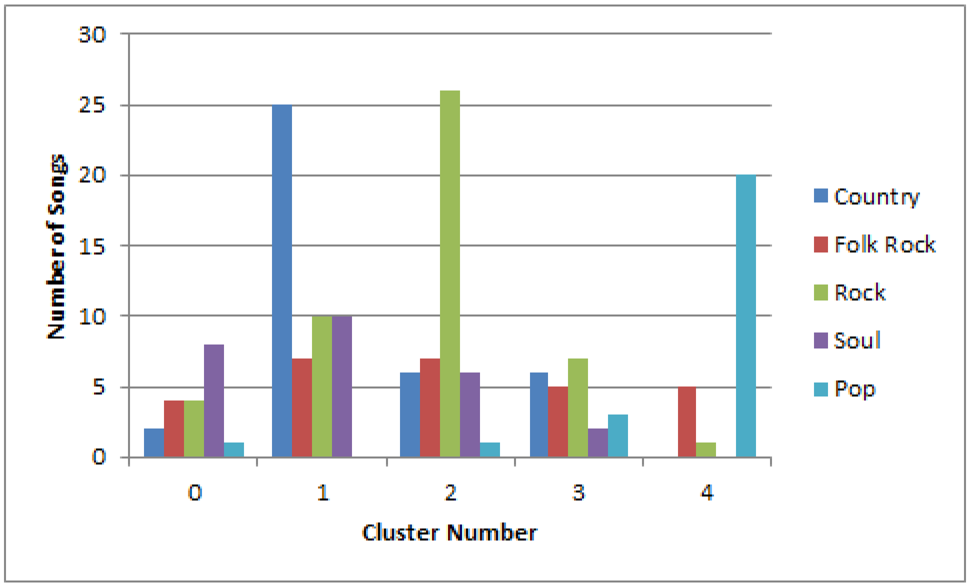

*Figure 5: Number of songs from each genre in each cluster for k-means clustering where k = 5 with no note length quartile features*

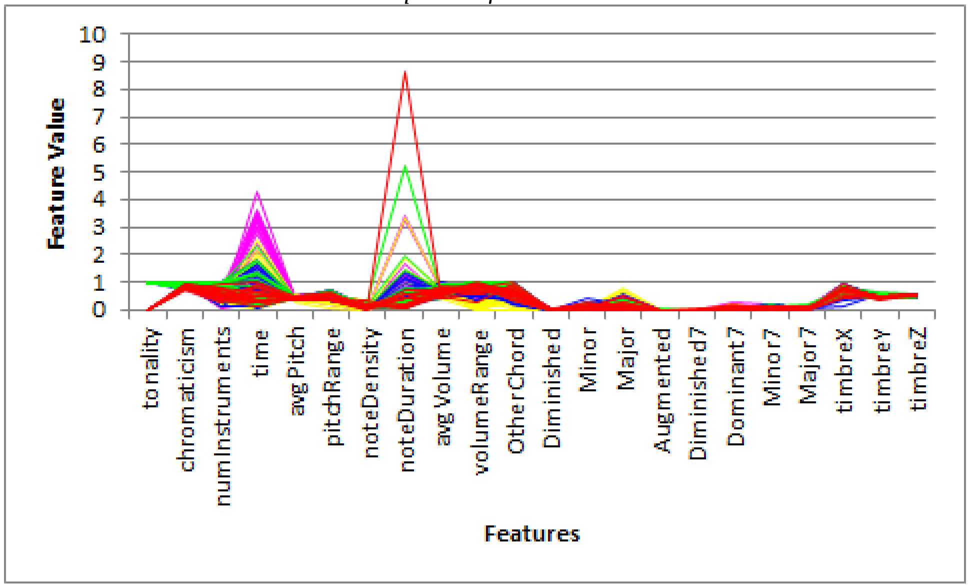

*Figure 6: Parallel coordinate visualization of all non-note length quartile features colored by cluster*

## 3.3 K-Means Clustering with Principal Component Analysis Features

In the k-means clustering using features created by principal components analysis (see Figure 7), described in subsection 1.2.3, the results tended toward placing most pieces in a single cluster. This is not a successful use of feature reduction and does not contain very much musical information.

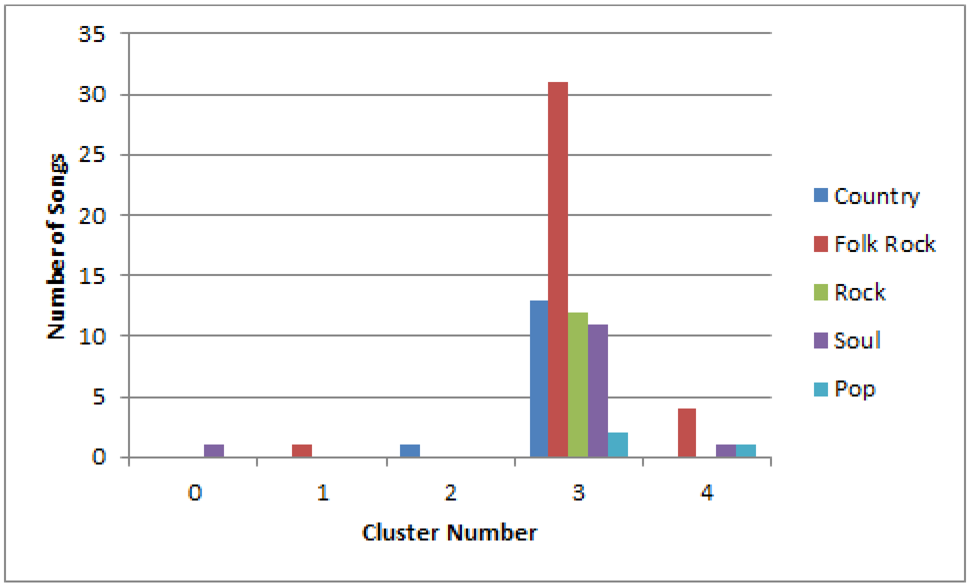

*Figure 7: Number of songs from each genre in each cluster for k-means clustering where k = 5 with features created by principal component analysis*

## 3.4 K-Means Clustering with Features Correlated with Genre

In the k-means clustering using features highly correlated with genre (see Table 2), discussed in section 1.2.3, only two clusters were associated with a genre. There was one cluster made up of almost entirely pop music, and another genre that was almost entirely country music (see Figure 9). Both of these genres also dominated clusters in the clustering task with the 22 non-note length quartiles features, discussed in section 3.2. However, with the feature vector in this task there are even fewer songs from the non-dominant genres. The feature values that these clusters respectively shared were many instruments and a long running time for the country cluster, and electronic instruments and a high note density (notes per time) for the pop cluster. In addition, another cluster, Cluster 3, was composed of songs with a higher proportion of major chords. However, similarly to the cluster of songs in a minor key, this feature by itself did not translate to a perceptual similarity and so is not as useful.

| **Feature** | **Pearson Correlation Coefficient** |
|---|---|
| Time | 0.1814 |
| Pitch Range | 0.1674 |
| Number of Instruments | 0.1638 |
| Volume Range | 0.1453 |
| Note Density | 0.1431 |
| $\text{Timbre}_0$ | 0.1417 |
| Other Chord | 0.1309 |
| Electric Bass(finger) Quartile 3 | 0.1274 |
| Electronic/Acoustic | 0.1237 |
| Violin Quartile 1 | 0.1191 |
| Minor Triad | 0.1183 |
| Major Triad | 0.1178 |
| Minor Seventh | 0.1173 |
| Fiddle Quartile 1 | 0.1142 |
| Mean Note Duration | 0.1127 |
| Rock Organ Quartile 3 | 0.1114 |
| Acoustic Guitar (steel) Quartile 3 | 0.1087 |
| String Ensemble 1 Quartile 3 | 0.1086 |
| Violin Quartile 3 | 0.1084 |
| Fiddle Quartile 3 | 0.1079 |
| Electric Bass (finger) Quartile 3 | 0.1056 |
| Average Pitch | 0.1038 |
| Acoustic Guitar (nylon) Quartile 1 | 0.103 |
| Alto Sax Quartile 1 | 0.1012 |

*Table 3: Features highly correlated with genre*

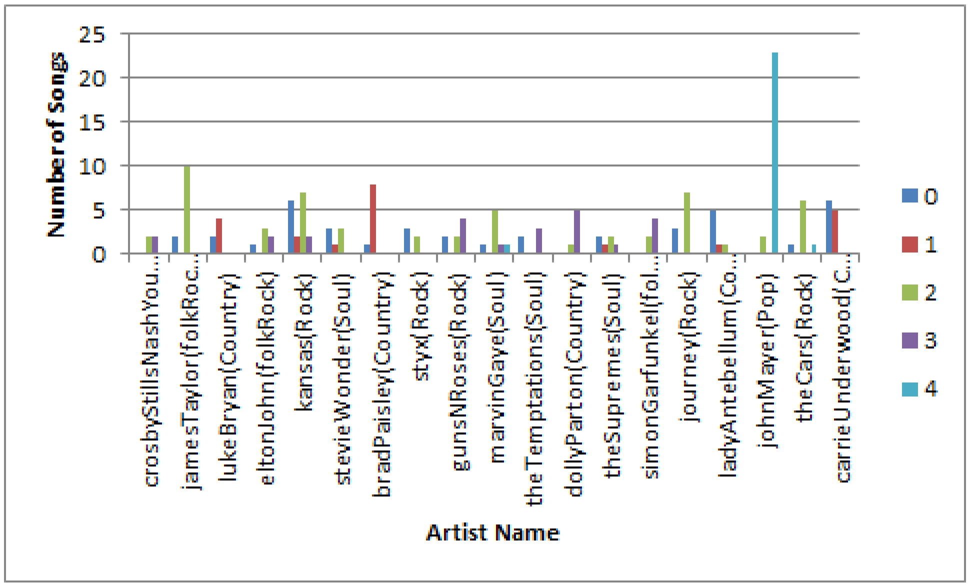

*Figure 8: Number of songs by each artist in each cluster for k-means clustering where k = 5 with features selected for correlation with genre*

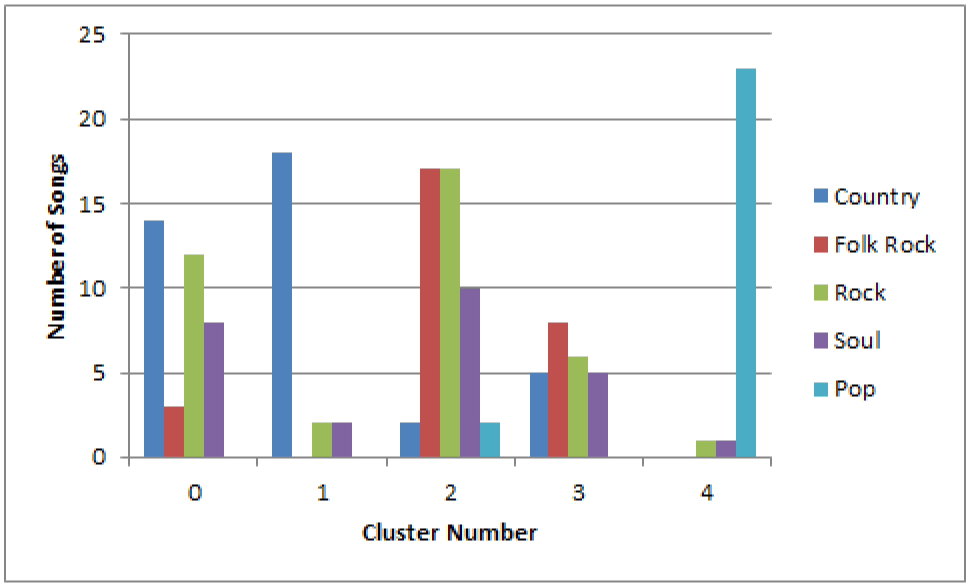

*Figure 9: Number of songs from each genre in each cluster for k-means clustering where k = 5 with features selected for correlation with genre*

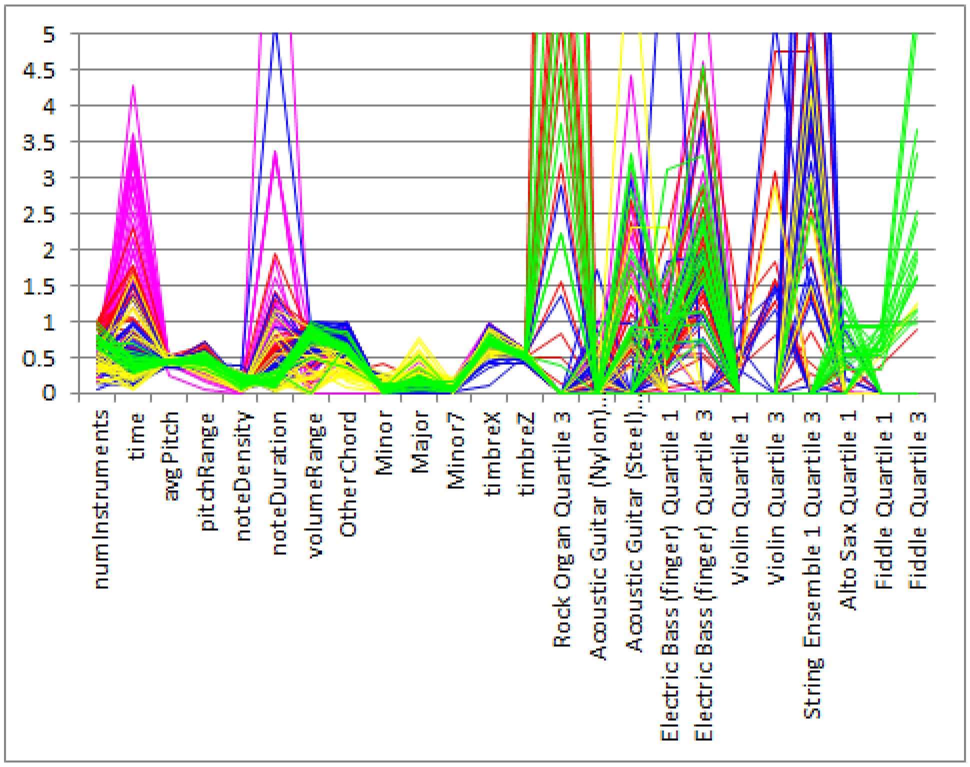

*Figure 10: Parallel coordinate visualization of features selected for correlation with genre colored by cluster*

## 3.5 Logistic Classification with Features Correlated with Genre

For classification, using a logistic classifier with 10 fold cross validation, the accuracy rate was about 56% (see Table 4). We can compare this with the expected accuracy if the classifier was placing songs into the five classes randomly (i.e., if the expected value of songs accurately classified per genre was one fifth of the total songs in that genre):

$$\mathbf{E}(\textit{Country songs correctly classified}) = 39/5 = 7.8$$

$$\mathbf{E}(\textit{Folk Rock songs correctly classified}) = 28/5 = 5.6$$

$$\mathbf{E}(\textit{Pop songs correctly classified}) = 25/5 = 5$$

$$\mathbf{E}(\textit{Rock songs correctly classified}) = 47/5 = 9.4$$

$$\mathbf{E}(\textit{Soul songs correctly classified}) = 26/5 = 5.2$$

$$\mathbf{E}(\textit{Percentage of total songs correctly classified}) = \frac{E(\textit{Total songs correctly classified})}{\textit{Total number of songs}} = \frac{33}{165} = 20\%$$

As the classifier performed much better than random chance, we can see that there is meaningful musical information encoded in the feature set made up of features highly correlated with genre.

| | **A** | **B** | **C** | **D** | **E** |
|---|---|---|---|---|---|
| **Country** | 25 | 5 | 2 | 6 | 1 |
| **Folk Rock** | 2 | 12 | 5 | 4 | 5 |
| **Pop** | 0 | 1 | 23 | 1 | 0 |
| **Rock** | 7 | 6 | 4 | 22 | 8 |
| **Soul** | 3 | 6 | 1 | 6 | 10 |

*Table 4: Confusion matrix for logistic regression classification with 10-fold cross validation with features selected for correlation by genre*

## 4. Conclusions

The goal of this project was to construct a feature vector that could be automatically computed and that leads to meaningful similarity measurement between pieces of MIDI music. Based on the classification and clustering results, both the feature vector made up of all features except note length quartiles (described in section 2.2.7) and the feature vector made up of features selected by correlation with genre (see Table 2) did contain useful information describing the musical content of the MIDI files. However, there is still room for improvement. In every cluster that was associated with a specific musical meaning, there were a small number of pieces that did not match that meaning, and in the clustering using both feature vectors, there were clusters that do not have an obvious description in musical terms. For the classification task, while the performance was far better than a random assignment of songs, there is still room for improvement from 55% accuracy. Hopefully, I can continue to fine tune this feature vector in the future.

## References

1. Clarke, B., Fokoué, E., Zhang, H.H. *Principles and Theory for Data Mining and Machine Learning*. New York, NY: Springer, 2009.

2. Freitas, Alex A. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Berlin, Germany: Springer, 2002.

3. Gomez, E., Klapuri, A., Meudic, B. "Melody description and extraction in the context of music content processing", *Journal of New Music Research* Vol. 32 Issue 1 (2003).

4. Gouyon, F., Pampalk, E., Widmer, G. "Evaluating rhythmic descriptors for musical genre classification." Paper presented at Audio Engineering Society 25th International Conference, London, United Kingdom, June 2004

5. Klapuri, A. "Multiple fundamental frequency estimation based on harmonicity and spectral smoothness", *IEEE Transactions on Speech Audio Processing* Vol. 11 Issue 6 (2003): 804-816.

6. Lomax, Richard G. *Statistical Concepts*. White Plains, NY: Longman Publishing Group, 1992.

7. Machine Learning Group at the University of Waikato. "Weka 3: Data Mining Software in Java" Viewed May 9, 2017. http://www.cs.waikato.ac.nz/ml/weka/

8. MIDI Manufacturers Association. "The Complete MIDI 1.0 Detailed Specification" Viewed May 12 2017. https://www.midi.org/specifications/item/the-midi-1-0-specification

9. MIDI Manufacturers Assocation. "Summary of MIDI Messages" Viewed May 12 2017. https://www.midi.org/specifications/item/table-1-summary-of-midi-message

10. Pachet, F., Aucouturier, JJ., La Burthe, A. et al. "The Cuidado music browser: an end-to-end electronic music distribution system." *Multimedia Tools and Applications* Vol. 30 Issue 3 (2006): 331-349.

11. Prandoni, Paolo. "An analysis-based timbre space." MS diss, University of Padua, 1993.

12. Scaringella, N., Zoia, G., and Mlynek, D. "Automatic genre classification of musical content: a survey." *IEEE Signal Processing Magazine* Vol. 23 Issue 2 (2006): 133-141.