2018 IEEE International Conference on Information Reuse and Integration for Data Science

Automated Playlist Generation from Personal Music Libraries

Diana Lin and Sampath Jayarathna
Department of Computer Science
California State Polytechnic University
Pomona, CA 91768, USA
(delin, ukjayarathna)@cpp.edu

***Abstract*—Manual creation of music playlists is a time consuming task that is typically characterized by an individual listening for similar audio features among a number of songs. The goal of this project is to reduce the time spent in performing this task by achieving the following: given an arbitrary collection of digital music recordings, automatically sort songs with similar musical qualities into playlists. This problem statement can essentially be reduced to a clustering problem, and this is the approach that we take. In this project, we accomplish the automatic generation of playlists by combining the use of music analysis tools and clustering algorithms from the field of machine learning. Additionally, we incorporate various visualization tools into the application user interface in order to give users an intuitive representation of the resultant playlists.**

Keywords-Music Clustering; K-Means; Affinity Propagation; DBSCAN

I. INTRODUCTION

Since the early 2000s, digital music libraries and music streaming services has seen a dramatic rise in popularity. The increase in available platforms for downloading and streaming music led digital music to generate forty-five percent of the music industry's total revenue in 2016 and overtake physical formats for the very first time [8]. This rapid growth in digital distribution of music has had both positive and negative implications for the music listening community. On one hand, the growth in distribution has made songs much more widely accessible than ever before. Emerging artists now have the opportunity to expose their recordings to the world at the click of a button. On the other hand, the sudden widespread availability has led to an overabundance in musical selection which has made it more difficult for some users to decide what to listen to [15]. Because of this, many music listeners have turned to platforms such as Spotify, Pandora, and iTunes to more easily manage and sort through large collections of digital recordings.

As individuals obtain rapidly growing collections of music, many music services are now delving into research to meet the increasing need for accurate retrieval of songs that are relevant to a user's personal preferences. This has prompted several recent developments in the emerging field of Music Information Retrieval (MIR) technology, from which two major approaches have risen [12]. The first approach primarily makes use of meta-data of digital recordings along with tagging information. The second approach considers user-related data and incorporates recommendation techniques such as collaborative filtering to generate relevant responses to user queries. Many platforms make combined use of these methods in order to offer a variety of features that allow users to easily curate and stream their favorite songs. Such features include, and are by no means limited to, the creation of personal playlists, access to precompiled playlists, and even personalized music recommendations.

Manually compiled playlists and auto-generated radio stations are oftentimes comprised of tracks that share similar musical qualities or that may be classified into the same genre. In Spotify, playlists are much like mixtapes which are compiled by individual users and may be shared with the public for universal consumption and enjoyment. On the other hand, radio stations offered by services such as Spotify and Pandora are constructed automatically through the use of machine learning algorithms applied towards musical analysis. Radio stations from these platforms take a single input parameter from the user in the form of a music track, artist, album, or music genre, and use it to generate a collection of songs that have similar characteristics. For instance, a Classical Music Radio Station may contain works from composers such as Beethoven and Mozart and a Classic Rock Radio Station may contain recordings from The Rolling Stones and Led Zepplin. An additional factor that distinguishes playlists from radio stations is that the music in a playlist is drawn from a user's personal music library, while the music included in a radio station is drawn from the service provider's music catalogue.

This project aims to incorporate the best features of personalized playlists and auto-generated radio stations. Personalized playlists have the benefit of containing only the music which a user enjoys. In particular, since a playlist is comprised of songs from a user's personal music library, each song has already been screened and approved by the user. This is extremely valuable due to the fact that it is often difficult to find music that fits the unique taste of any given individual. On the other hand, auto-generated radio stations have the convenience of saving users a substantial amount of time through the use of machine learning

978-1-5386-2659-7/18/$31.00 ©2018 IEEE
DOI 10.1109/IRI.2018.00039

algorithms. The algorithms that are used to make recommendations are designed to be scalable and are thus readily applicable to large collections of music.

II. MOTIVATION

Today's digital music service providers have made extraordinary advances in a short amount of time. The ability to make recommendations for music, which is a highly subjective task, is a great feat in itself. However, there is still room for improvement in the midst of these great developments. Auto-generated radio stations are great for saving listeners time and effort, but oftentimes these stations may still contain songs which do not suit a listener's taste. Although it is possible for individual users to compile their own playlists filled with their favorite songs, doing so is no trivial task. Organizing music into playlists requires consideration of several aspects of a song such as tempo, acousticness, and loudness and music with similar features and styles are usually grouped into the same playlist. Since today's technology has made it easy for people to store hundreds or even thousands of audio recordings, combing through large collections to create playlists consumes a considerable amount of time and effort. A probable solution to this is a service that automatically compiles playlists from a user's personal music library. Creating playlists from a user's existing library will guarantee a higher level of user satisfaction since each individual's library is known to be comprised of music which he or she enjoys.

III. CHALLENGES

A. Objective Analysis of Subjective Items

The classification and analysis of music is a relatively subjective task due to its dynamic nature and the emotion that it expresses – a song may start out quiet and somber but later build up to a dramatic climax. Although songs contain some measurable features such as tempo or the key that it is in, there are several other features, such as liveliness, that are not easily extracted. These subtle nuances that are inherent in musical pieces are more easily detected by a human than a machine. Because of this, the initial method of classification was primarily manual inspection by human experts, with the most notable and comprehensive effort being the Music Genome Project spearheaded by Pandora [3]. Ascribing measurements to subjective features remains a difficult task for the field of music information retrieval today. For instance, an adequate solution to the longstanding challenge of automated genre classification remains elusive to researchers even after years of advancement in the field. The quantization of many musical features face this similar challenge because they are not measurable nor do they fit into binary classifications. As such, the challenge at hand is to extract various musical features which are traditionally subjective and to quantify them via track analysis.

B. Large Collections of Music Recordings

Undeniable advancements in technology have led many of the world's industries to transition into the digital age, and the music industry is no exception. The growth of the Internet has made music recordings more accessible than ever before and users now have numerous options to download or purchase their favorite songs for their personal collection. The revenue from digital music has steadily overtaken physical formats (e.g. CDs, cassette tapes, vinyl records) on a global level [8]. Knowing this, it is reasonable to suggest that such an increase in revenue is linked with an overall increase in the size of music collections. Since music libraries can easily contain up to hundreds or thousands of songs, the methods used in this project face the challenge of being effective and scalable.

C. Usability for Non-technical and Advanced Users

Classification of music requires in-depth analysis of several aspects of a musical piece and the algorithms that are used in this process are complex and involve much mathematical, statistical, and computational theory. Most users are casual music listeners and do not have much technical knowledge concerning music or the inner workings of clustering algorithms. Therefore, special attention must be paid to the interface design. Studies have shown that there is a negative correlation between visual complexity and overall perception [13]. Therefore it is important to abstract away unnecessary complexities to give users a more pleasant experience. Although some advanced users may appreciate having the ability to fine tune the parameters of an algorithm, exposing too many options may disorient the average user. Thus, we are faced with the challenge of achieving a balance between a simple and intuitive interface while providing more advanced users options to control finer details.

IV. SOLUTION

In response to the challenges involved in creating playlists from large music libraries, this project incorporates clustering algorithms and musical analysis tools into an intuitive web application. The features of this project are built upon the open-source tool Organize Your Music which was designed by Paul Lamere, the Director of Developer Platform for The Echo Nest at Spotify.

A. Analysis of Quantifiable Features

The Echo Nest is a "music intelligence platform [that] synthesizes billions of data points and transforms it into musical understanding to power smarter music applications" [19]. Together with Spotify, they have worked to analyze music and make the information available to the public via the Spotify Developer Web API. One great contribution is the large scale analysis and quantization of subjective audio features into integer or float values for tracks in the Spotify catalogue. Table 1 displays a few of the available audio features along with a brief description of each.

The mapping of these features into numerical values is extremely valuable because it allows for clustering analysis based on Euclidean distance. Fig. 1 demonstrates the visualization of audio features in three-dimensional Euclidean space. When using Euclidean space as the distance measure in clustering algorithms, a collection of points may be characterized by its average. The average

TABLE I. QUANTIFIED AUDIO FEATURES FROM THE SPOTIFY WEB DEVELOPER API [20].

Feature	Description
Danceability	Danceability describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable.
Energy	Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity. Typically, energetic tracks feel fast, loud, and noisy.
Valence	A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).

represents what is known as the centroid [11]. Without the ability to map data into such a space, one must find an alternative means of summarizing clusters.

Many of these features are subjective and difficult to incorporate into analysis since there had previously been no standard way to quantify them. Excluding these features meant that a big part of important information was missing from analysis. Now that they are available, this project incorporates these features as the measurement for clustering algorithms. For example, a user may select which of these audio features that the creation of playlists is based on. So if "danceability" and "energy" are chosen, then the clusters will be formed based on songs that have similar measures of "danceability" and "energy".

Figure 1. Three-dimensional visualization of musical tracks with various audio features from the Organize Your Music 2.0 application.

B. *Application of Machine Learning and Clustering Algorithms*

To accommodate the increasing size of music collections, we chose to use machine learning algorithms due to their efficiency and scalability. For this project we use K-Means Clustering, Affinity Propagation, and DBSCAN Clustering. A sample dataset of approximately seventy songs from four

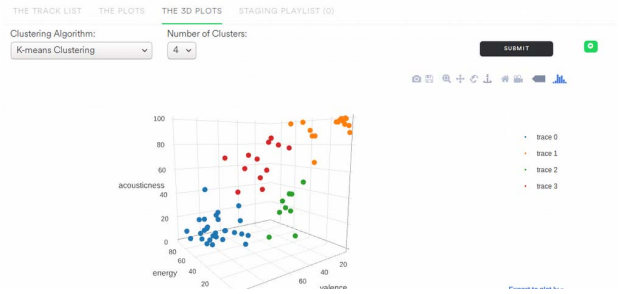

Figure 2. Example of resultant clusters from the K-Means Clustering Algorithm.

different genres are used to show the resultant clusters of these three algorithms – they are shown in Fig. 2, Fig. 3, and Fig. 4.

K-Means clustering is one of the most widely used clustering methods in data mining due to its simplicity and scalability [21]. It is an unsupervised method of machine learning that can be broken down into two major steps. The algorithm begins with a random set of initial centers where each data point is assigned to its closest center. Secondly, each center is repeatedly recomputed until each data point converges to a fixed cluster [2]. K-Means clustering has been applied to several different fields such as segmentation and information retrieval, and this project brings this widely used algorithm into music clustering to test its effectiveness when applied to personal music libraries. Fig. 2 shows the output of K-Means clustering when applied to the sample dataset. Clustering is based on the audio features provided by the Spotify Developer Web API and the user is allowed to specify the number of clusters that he or she would like the algorithm to produce.

Although the K-Means algorithm is intuitive and widely used, it is sometimes seen more as of a partitioning method than a clustering method. In other words, it partitions the data set into *k* groups rather than identifying naturally occurring clusters and placing less emphasis on outliers. Additionally, the need to specify the initial number of clusters may not always be desirable. A well-known clustering algorithm that addresses some of the shortcomings of K-Means clustering is Affinity Propagation, which was developed and published by Frey and Duek in 2007. This is one of the favored algorithms among the various methods

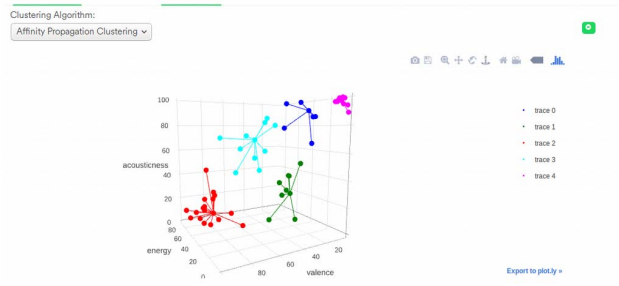

Figure 3. Example of resultant clusters from the Affinity Propagation Algorithm

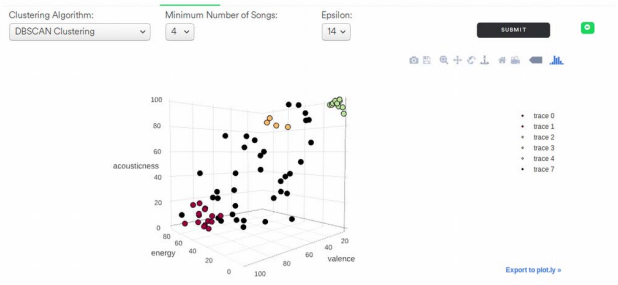

Figure 4. Example of resultant clusters from the DBSCAN algorithm.

because it maintains high performance and, unlike K-Means clustering, it does not require the specification of number of clusters. Affinity Propagation uses a message passing network and recursively transmits messages between all data points. This is done until a promising set of centroids begin to naturally appear with their corresponding clusters [7]. The output of Affinity Propagation for the sample dataset is shown in Fig. 3. The centroids for each of the resultant clusters are also shown in the plot and can be identified as the node which shares a connection with all other nodes in that cluster.

Both K-Means clustering and Affinity Propagation have better performance when clusters within the data are globular. However, globular clusters may not always be naturally occurring. Thus we turn to density-based spacial clustering of applications with noise, otherwise known as DBSCAN. This algorithm is highly scalable and excels in discovering clusters with arbitrary shape due to its ability to classify sparse areas as background noise and hone in on denser regions [5]. To determine whether a region is dense, "each point of a cluster in the neighborhood of a given radius has to contain at least a minimum number of points, i.e. the density in the neighborhood has to exceed some threshold" [5]. The radius used to determine the neighborhood is a user-specified parameter, epsilon. Intuitively, the larger the value of epsilon, the more data points a cluster will encompass. An additional input parameter that can be tuned in this project is the minimum number of points that is in each set. This can help users enforce the generation of playlists of at least a certain size. Although it has been shown that DBSCAN has the potential to generate better results than K-Means clustering and Affinity Propagation, it comes with a few shortcomings. Among them is that the tuning of the epsilon and minimum samples parameter may prove to be difficult for less technical users, since the algorithm is quite sensitive to them. A sample output using DBSCAN is shown in Fig. 4 with the colored clusters representing playlists and the black data points representing regions of data that are too sparse to be included in a cluster.

C. User-Friendly Interface

The Organize Your Music 2.0 application faces a challenge of bringing together technical tools for the use of non-technical users. This issue is important to address because studies have shown that a user's overall impression of a site is very influential in determining whether he or she will continue interacting with the it or move on to another site [18]. The main obstacle is to abstract away the complexities of clustering algorithms for the average user and simultaneously give advanced users the control and the ability to fine tune various parameters for each algorithm. This is handled through specific design choices and the use of web tools.

To reduce some complexities, we remove the need for users to make many initial decisions by setting a few default choices. In particular, we set the default algorithm as the K-means clustering algorithm. This choice was made because the concept behind K-means is more simple to grasp than Affinity Propagation and DBSCAN. Additionally, we change the input parameter to be more relevant to the needs of the application's target audience. Traditionally, the input parameter is a value *k* which determines the number of clusters to produce from the dataset. However, this application takes as input the approximate number of songs desired in each resultant playlist. The reasoning behind this design choice is that it is more natural to think of the average number of songs that one desires in a playlist than to think of the number of playlists one wishes to have.

Development tools were also used to help accustom users to the unique features of this application. One such tool is Intro.js, a product tour creation library. This library is used to give a simple introduction to the basic features that the application provides and integrate helpful tips throughout the webpage. A welcome tour automatically starts when a user first visits the site and the user can opt out of the tour at any time. To resume the tour from any particular feature page, the user simply selects the 'Help' icon. The 'Help' icon is strategically placed in the top-right corner of every page so that it is unobtrusive and easily accessible to the user. Fig. 5 depicts the welcome tour that the user is greeted with and the 'Help' icon is highlighted in the top right corner.

Lastly we make use of an 'Info' icon that produces a modal dialogue when clicked. The modal dialogue is shown in Fig. 6 and displays basic information regarding the current selected clustering algorithm including details about its input parameters. It also has an additional link to an online reference that contains more information regarding the specific algorithm. This feature helps non-technical users gain a better understanding of each algorithm and also provides insight into the meaning of each parameter.

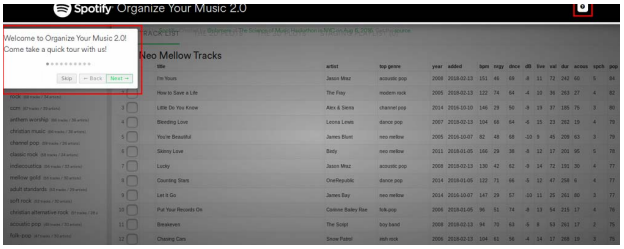

Figure 5. Application welcome tour and 'Help' icon.

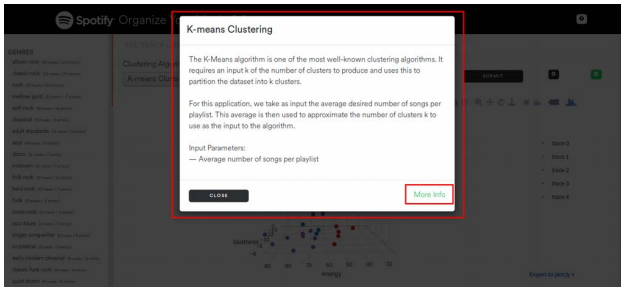

Figure 6. Modal dialogue displaying basic information regarding the K-means Clustering algorithm with a link t o an online resource for further information.

V. Development Techniques

The Organize Your Music 2.0 application is designed to be a web service that is available to all Spotify users. Its foundation is based on the work of Paul Lamere, the Director of Developer Platform for the Echo Nest at Spotify, who developed the original Organize Your Music application and graciously made it open-source on GitHub. This project adds several components to the existing code base that allow users to automate the sorting of their music library into playlists and simultaneously visualize the results. Organize Your Music 2.0 is composed of a client-side and server-side framework that can be hosted and modified on a local machine or actively running on a remote server such as an Amazon Elastic Cloud Compute (EC2) instance. The client side is primarily comprised of HTML and Javascript functions that are able to perform some basic manipulations to music data. The client side also communicates with the server side through routing methods to access the Spotify Web Developer API and carry out complex computations. The server makes extensive use of Node.js in order to incorporate the features of this project that are more computationally heavy. Most notably, it runs Python scripts which make calls to Python libraries designed for machine learning, clustering, and data visualization.

A. Authentication and Information Gathering

To access user-specific data, the Organize Your Music 2.0 application must first pass two forms of authorization: application authorization and user authorization. Application authorization allows the application to access the tools that are made available by Spotify, such as APIs, SDKs, and Widgets, and it is granted by Spotify once the application is registered [1]. User authorization is granted by the Spotify user and this allows Organize Your Music 2.0 to access a user's data, such as personal playlists.

After a user grants authorization to the application, it can begin to retrieve the information that is needed to perform clustering. The metadata for each music track is obtained by sending a GET request to the RESTful Spotify Web API which returns the data in JSON format. To analyze and cluster songs, the application requests the audio features of each track. Some of these features include tempo, danceability, energy, acousticness, and liveness. After analyzing this data either manually or with one of the available clustering algorithms, a user has the option to create a new playlist with the selected songs. This last step is accomplished through a POST request to the API.

B. Data Visualization

The visualization of music data focuses on the audio features that have gone through analysis via The Echo Nest's music audio analysis tool called "Analyze". This tool "uses proprietary machine listening techniques to simulate how people perceive music" and in turn produces a breakdown of musical attributes of a given recording [9]. The metadata for each track is queried from the Spotify Developer Web API in the manner described in the previous section.

After the data is acquired, the user-specified features are extracted and used to form a visual representation via Plotly. Plotly is an open-source tool that is designed to aid in interactive data visualization and is available for use with both Javascript and Python. This library is designed to be easily incorporated into web applications and it makes heavy use of the JSON objects for the customization of charts as the primary input format for data. The Plotly toolbox has numerous charting options available that range from basic charts such as scatter plots and bar charts, to statistical charts such as histograms. For this project, we choose to display the music data with two-dimensional and three-dimensional scatter plots. Scatter plots are a suitable choice for this application due to their simplicity and the ease with which they can intuitively represent multiple dimensions.

C. Machine Learning and Clustering Tools

The computational tools used on the server side of Organize Your Music 2.0 primarily consist of resources from Python libraries. The clustering of music data was accomplished using the Scikit-learn machine learning library which is built on top of Scipy, another Python library that has many applications in mathematics, science, and engineering. Scikit-learn offers tools for classification, regression, clustering, and preprocessing and this project makes use of the K-Means, Affinity Propagation, and DBSCAN algorithms which were discussed in the Section IV.

For each of the three algorithms, a Python script takes the music metadata as input along with any tuning parameters and produces clusters using the corresponding cluster method. These inputs are passed from the client-side to the server-side using routing methods with the Node.js Express framework. Express provides a simple framework that allows applications to respond to POST and GET requests from the client-side [6]. To retrieve music metadata and other input parameters for the clustering algorithms, Organize Your Music 2.0 sends a POST request with the data in a JSON format, allowing the Python script to easily extract the necessary fields.

VI. Empirical Results

Since it is difficult to give a direct measure of how similar auto-generated playlists are to playlists compiled by individual users, the results of this application were evaluated based on user sentiment. This was accomplished through a questionnaire which users completed a prior to and following the use of the application. There were six respondents to these surveys, three male and three female, that ranged from twenty-three years of age to thirty-six years of age. All of the participants have been Spotify users for at least one year.

A. Pre-Questionnaire: Determining User Sentiment

The questionnaire that was given prior to using the Organize Your Music 2.0 application is designed to help determine the general sentiment about manual playlist creation and whether there is a need for the application. The survey questions and responses are shown in Table 2 and the results show that the average music listener does not spend much time creating their own playlists nor is it a highly enjoyable task.

The majority of respondents expressed that time was the primary reason they do not create their own playlists very often. The second most common reasons are that they have too many songs to sift through and that it requires too much effort. These results support the motivations behind creating the application.

TABLE II. Questionnaire and results gathered prior to using the Organize Your Music 2.0 application.

Questions	Responses	
Do you enjoy making your own playlists?	Yes: No: Sometimes:	16.67% 50.00% 33.33%
On average, how often do you create your own playlists?	Once every few months: Twice a year: Once a year: Never:	0.00% 33.33% 50.00% 16.67%
Briefly give some reasons why you don't make playlists very often (e.g. too time consuming, too much effort, too many songs to go through, etc). You may give more than one reason.	Too much time: Too many songs to go through: Takes too much effort: Other playlists are provided:	83.33% 33.33% 33.33% 16.67%
Would you rather create your own playlist or have songs automatically sorted into playlists for you?	Rather create their own: Rather automatically sort:	33.33% 66.67%

B. Post-Questionnaire: Evaluating Performance

The questionnaire that was completed by users after using the Organize Your Music 2.0 application is designed to determine the quality of the resultant playlists and how well they represented playlists that they might have generated themselves. Table 3 demonstrates that the overall performance of the application was positive although there are still areas that can be improved.

TABLE III. Questionnaire and results gathered after using the Organize Your Music 2.0 application.

Questions	Responses	
Did the auto-generated playlists resemble playlists that you would create yourself?	(Not at all) 1: 2: 3: 4: (Yes, it was spot on) 5:	0.00% 16.67% 50.00% 33.33% 0.00%
In the auto-generated playlists, roughly what percentage of the songs were appropriately grouped?	50%: 60%: 70%: 80%:	16.67% 16.67% 50.00% 16.67%
Would you use this site to create playlists from your music library in the future?	Yes: No, I would rather make my own: No, I would rather listen to Spotify's playlists: Maybe once in a while:	16.67% 0.00% 50.00% 33.33%

When asked to rate the generated playlists on a scale from 1 to 5, with 1 being least similar and 5 being most similar to playlists that they would compile, half of the users responded with a 3 and one-third responded with a 4. The survey respondents were also asked to give the percentage of songs that were appropriately grouped into a playlist, and all participants indicated that fifty percent or more of the songs belonged. However, two-thirds of these responses were in the fifty percent to sixty percent range which suggests that alternative clustering algorithms, or perhaps even alternative methods altogether, may need to be considered in the future. Lastly, users were asked whether or not they would use the application again to organize their music library. There were mixed sentiments with half of the users indicating that they would use it again (either frequently or infrequently) and the remaining half indicating that they would rather use playlists from within Spotify's collection. The fifty percent of users that indicated they would use the application again support the concept of this project. Although the remaining fifty percent of the respondents indicated that they would rather use playlists generated by Spotify, this result shows that there is still a need for automated playlist generation. The combined results of this last question encourage further investigation into the general field of automatic music playlist generation. In regards to this project, it suggests that alternative algorithms may need to be considered to increase effectiveness.

VII. Related Work

Several additional methods of music clustering exist and two popular approaches are collaborative filtering and genre-

based clustering, both of which are incorporated in the recommendation systems of Spotify and Pandora. Collaborative filtering can be further classified into two main types: user-based and item-based. As the names suggest, user-based collaborative filtering makes recommendations on the basis of user behaviour and takes into consideration the likes and dislikes of other users that have similar behaviour patterns [22]. On the other hand, item-based collaborative filtering makes recommendations based on shared characteristics between items that users have expressed either positive or negative sentiment towards [14].

For this particular application, user-based collaborative filtering may not be as appropriate since we consider personal music libraries independent of other users. However, there may be some aspects of item-based collaborative filtering which we may draw from to improve the current implementation of the clustering algorithms.

Genre-based clustering seeks to separate songs into groups that are each comprised of a single genre [17]. This form of clustering is an ideal candidate for this application since music within the same genre share many similar musical characteristics. However this approach relies heavily on the accuracy of genre classification, which is an area that is still undergoing development. Once genre classification is more developed and reliable, this may be a good alternative to the current method of clustering based on audio features.

VIII. Conclusion and Future Work

Our preliminary results confirm that there is a need for automated playlist generation in the music listening community. After conducting a general survey, it is clear that listeners find the manual creation of playlists to be too time consuming and requiring too much effort. This application relieves some of these needs by making use of machine learning algorithms to analyze large collections of music and to sort them into playlists. Additionally, it serves as visual analysis tool for audio features of music tracks. After testing the application, user responses indicated that our solution is useful but requires some modifications to remain competitive with existing alternatives.

To improve upon the current implementation, additional information, such as a song's album and artist, can be incorporated to increase the likelihood that songs with these shared fields will be in the same cluster. Future work may also include the use of alternative machine learning algorithms that work more as clustering algorithms rather than partitioning algorithms. Ideally, we would like to add a one-click feature that generates playlists without the need for user input and the ability to make the visualization aspect optional. As development in this application continues, we also seek to incorporate continuous feedback regarding user interface design in order to maintain ease-of-use for the average music listener.

References

[1] Authorization Guide (n.d.). Retrieved March 03, 2018, from https://beta.developer.spotify.com/documentation/general/guides/authorization-guide/

[2] Bahmani, B., Moseley, B., Vattani, A., Kumar, R., & Vassilvitskii, S. (2012). Scalable k-means++. *Proceedings of the VLDB Endowment*, *5*(7), 622-633.

[3] Castelluccio, M. (2006). The music genome project. *Strategic Finance*, *88*(6), 57.

[4] Cunha, R. L., Caldeira, E., & Fujii, L. (2017). Determining Song Similarity via Machine Learning Techniques and Tagging Information. *arXiv preprint arXiv:1704.03844*.

[5] Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd* (Vol. 96, No. 34, pp. 226-231).

[6] Express – 4.16.1 (n.d.). Retrieved March 03, 2018, from https://expressjs.com/

[7] Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. *Science*, *315*(5814), 972-976.

[8] IFPI, I. (2016). *IFPI Global Music Report 2016*. [Online] Ifpi.org. Available at: http://www.ifpi.org/news/IFPI-GLOBAL-MUSIC-REPORT-2016 [Accessed 26 February 2018].

[9] Jehan, T., DesRoches, D. (2014, January 7). *Analyzer documentation (analyzer version 3.2)*. Retrieved March 3, 2018, from http://docs.echonest.com.s3-website-us-east-1.amazonaws.com/_static/AnalyzeDocumentation.pdf

[10] Lamere, P. (2013, September 22). Music Popcorn – A visualization of the music genre space [Web log post]. Retrieved March 2, 2018, from https://musicmachinery.com/2013/09/22/5025/

[11] Leskovec, J., Rajaraman, A., & Ullman, J. D. (2014). *Mining of massive datasets*. Cambridge university press.

[12] McFee, B. (2012). *More like this: machine learning approaches to music similarity*. Ph.D. Dissertation, University of California, San Diego.

[13] Pandir, M., & Knight, J. (2006). Homepage aesthetics: The search for preference factors and the challenges of subjectivity. *Interacting with Computers*, *18*(6), 1351-1370.

[14] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (pp. 285-295). ACM.

[15] Schwartz, B. (2004) *The Paradox of Choice: Why More Is Less*. New York, NY: HarperCollins.

[16] Slaney, M., Weinberger, K., & White, W. (2008). Learning a metric for music similarity. *International Symposium on Music Information Retrieval (ISMIR)*.

[17] Tsai, W. H., & Bao, D. F. (2010, April). Clustering music recordings based on genres. In *Information Science and Applications (ICISA), 2010 International Conference on* (pp. 1-5). IEEE.

[18] Tuch, A. N., Presslaber, E. E., StöCklin, M., Opwis, K., & Bargas-Avila, J. A. (2012). The role of visual complexity and prototypicality regarding first impression of websites: Working towards understanding aesthetic judgments. *International Journal of Human-Computer Studies*, *70*(11), 794-811.

[19] We Know Music... (n.d.). Retrieved March 02, 2018, from http://the.echonest.com/

[20] Web API Object Model (n.d.). Retrieved March 02, 2018, from https://developer.spotify.com/web-api/object-model/

[21] Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., ... & Zhou, Z. H. (2008). Top 10 algorithms in data mining. *Knowledge and information systems*, *14*(1), 1-37.

[22] Zhao, Z. D., & Shang, M. S. (2010, January). User-based collaborative-filtering recommendation algorithms on hadoop. In *Knowledge Discovery and Data Mining, 2010. WKDD'10. Third International Conference on* (pp. 478-481). IEEE.