FINAL PROJECT

Clustering Now with Spotify

Aakanksha Gurjar

Theodore Wiebold

Kirthana Sukumaran

Jeetesh Garsund

Applied Cognition and Neuroscience Program

University of Texas at Dallas

Abstract

The project is to display the features of top 20 billboard Spotify songs from 2003 to 2016 through the computational model of k-means in order to create a possible foundation for generating what moods the top songs create. The aim is to see if a simplified version of musical data is sufficient for genre or mood categorization. After the dataset has been clustered through k-means, the two outcome will be tested subjectively to see if it replicated how a human would classify the music. The results will show what features a song needs to be most likely to get onto the top 20 Spotify billboards. The results give clear definitions of what a song needs in order to be on the top 20 billboard hits, so the next exploratory step would be classifying the songs into genres moods based on their features. The outcome of this study created a simplified method of classifying music without the requirement of audio files.

Keywords: Kmeans, clustering, music, features, Spotify

Clustering Now with Spotify

Today's world is all about efficiency and simplicity. Many of the problems faced with music categorization is the huge size of the data consisting of audio files. A second problem is finding the most efficient algorithms to evaluate those data sets. Finally, a third problem is determining what type of emotions listeners experience when listening to some of the music industry's most popular songs.

Abubaker and Ashour (2013) identified the drawbacks of the Kmeans algorithm and suggested some improvements. Their improvements consisted of modifying the basic Kmeans algorithm so that it was affected less by noise and outliers while automatically detecting the number of clusters in the data set. Their other improvements involved creating the best initial prototype despite the clusters having different shapes and densities. The extreme simplicity and efficiency of the Kmeans algorithm may be too simplistic for accurate data representation for complex data.

Bahamani, Moseley, Vattani, Kumar, and Vassilvitskii (2012) created an improved version of the initialization algorithm for Kmeans. Compared to the Kmeans++ algorithm, that is already an improvement over the basic Kmeans algorithm, the proposed Kmeans|| algorthim Bahamani et al. (2012) created needed less rounds and $O(\log n)$ was not necessary. They found that their algorithm was much faster than other existing Kmeans algorithms and the number of iterations is smallest when using Kmeans|| as the seed during Lloyd's algorithm convergence.

The initialization of the Kmeans algorithm is crucial in order to obtain representative clusters, especially with a large and complicated dataset.

Na, Xumin, and Yong (2010) focused on improving the efficiency of the Kmeans algorithm. Na et al. (2010) change the algorithm from calculating the distance between each data objective and all clusters in each iteration to creating a simple data structure that stores information in every iteration to be used in the next iteration. They found that with smaller data sets the improved version reduced run-time and improved accuracy. They also found that the larger the data set the more efficient their algorithm became. With a large and complicated dataset, the highest efficiency and accuracy is desired in order to create the most representative clusters.

In the proposed study, the data set will be represented by Kmeans to see how well it represents the simplistic data set. Once the features are clustered together we will have established a foundation for future classification of the data set. Based on the results from Abubaker and Ashour (2013), Bahamani et al. (2012), and Na et al. (2010) we hypothesize that Kmeans will sufficiently represent the data in a way in which a human would group the songs together.

Method

Data Set

The data set contains 18 audio features of the 1193 songs convened from the Spotify Top 20 Billboard Charts from 2003 to 2016. The features included are as follows: speechiness, key, time signature, liveliness, loudness, duration in milliseconds, danceability, duration, valence,

acousticness, Spotify ID, volume number, energy, tempo, instrumentalness, mode, number, and artist (see Appendix 1).

Pre- Processing the Data

- Removed blank or incomplete data.

  The data set comprised of a few blank rows and incomplete information. This was taken care of using the complete.cases function which returns a logical vector indicating the cases that are complete, i.e., have no missing values.

- Standardization

  The Scale function in R whose default method centers and/or scales the columns of a numeric matrix was used on the data set. Standardization is an important tool while performing cluster analysis because the similarity within data points are measured in terms of distance and if dimensions are measured in large scale, they have a much larger effect.

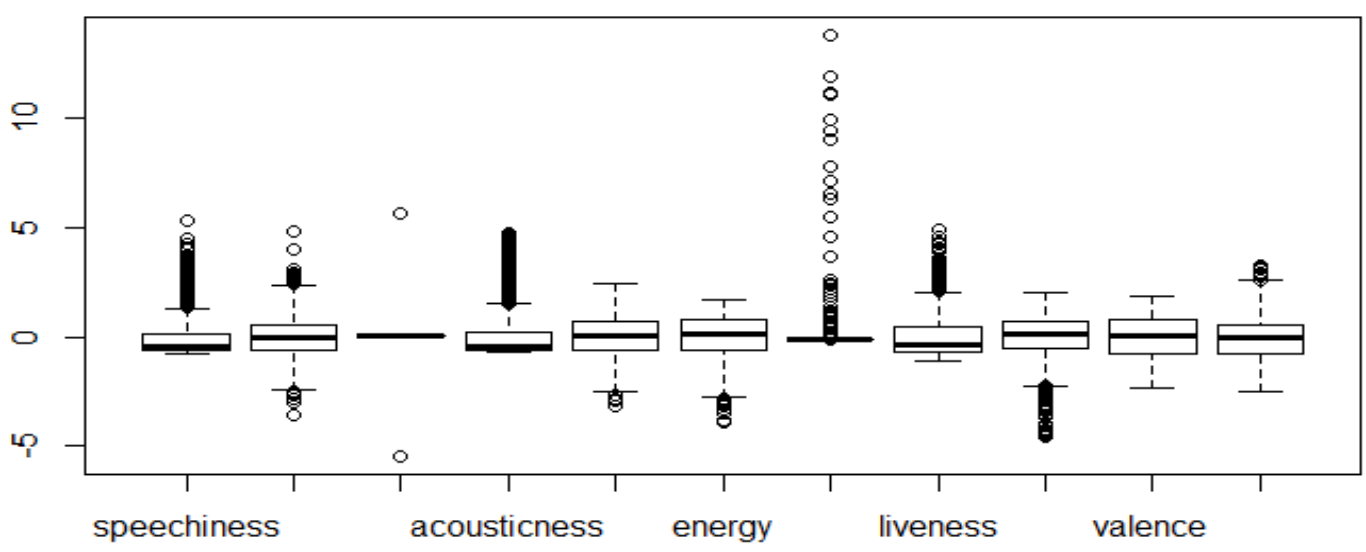

Figure 1. Box plot after normalisation of the numerical feature variables

- Separated out the variables into categorical and numerical data.

The standard k-means algorithm isn't directly applicable to categorical data, for various reasons. The sample space for categorical data is discrete, and doesn't have a natural origin. A Euclidean distance function on such a space isn't really meaningful. So the variables in the dataset were separated into numerical and categorical variables.

Numerical Data	Categorical Data
- Danceability - Tempo - Speechiness - Loudness - Acousticness - Liveliness - Instrumentalness - Valence - Energy - Duration_ms	- Spotify url - Time_ signature - sotify_id - Track_href(Link to song API for full details of the song) - Key - Analysis_url - Mode - Type (Audio feature)

Clustering Data

- Clustering is an unsupervised machine learning task that automatically divides the data into clusters**,** or groups of similar items. It does this without having been told how the groups should look ahead of time.

- As we may not even know what we’re looking for, clustering is used for knowledge discovery rather than prediction. It provides an insight into the natural groupings found

within data.

- Meaning of "good clustering" is subjective and its interpretation varies with applications, time and even the analyst.

- To figure out the optimum number of clusters for our data we used NBClust function. NBClust package provides 30 indices for determining the number of clusters and proposes to user the best clustering scheme from the different results obtained by varying all combinations of number of clusters, distance measures, and clustering methods.

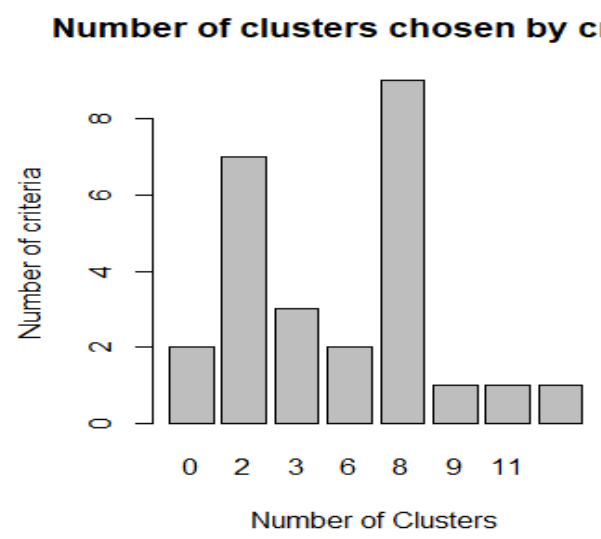

Figure 2. Bar graph describing optimum number of clusters

K-means

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups).

The *K*-means clustering algorithm uses iterative refinement to produce a final result. The algorithm inputs are the number of clusters *K*and the data set. The data set is a collection of

features for each data point. The algorithms starts with initial estimates for the *K* centroids, which can either be randomly generated or randomly selected from the data set. The algorithm then iterates between two steps:

1. Data assignment step:

Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance. More formally, if *ci* is the collection of centroids in set *C*, then each data point *x* is assigned to a cluster based on

$$\underset{c_i \in C}{\arg\min}\ dist(c_i, x)^2$$

where $dist(\cdot)$ is the standard ($L2$) Euclidean distance. Let the set of data point assignments for each *ith* cluster centroid be *Si*.

2. Centroid update step:

In this step, the centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster. More formally, if *ci* is the collection of centroids in set *C*, then each data point *x* is assigned to a cluster based on

$$c_i = \frac{1}{|S_i|} \sum_{x_i \in S_i} x_i$$

The algorithm iterates between steps one and two until a stopping criteria is met (i.e., no data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached).

This algorithm is guaranteed to converge to a result. The result may be a local optimum (i.e. not necessarily the best possible outcome), meaning that assessing more than one run of the algorithm with randomized starting centroids may give a better outcome.

The algorithm described above finds the clusters and data set labels for a particular pre-chosen *K*. To find the number of clusters in the data, the user needs to run the *K*-means clustering algorithm for a range of *K* values and compare the results. In general, there is no method for determining exact value of *K*, but an estimate was made using Nbclust function which estimated the optimum number of clusters for our data to be N=8.

The procedure for evaluating the data is as follows: clean and transform the data per the selected number of features, choose K and run the algorithm, review results, iterate over several values of K.

Results

The 2D visualisation of cluster plot revealed the presence of a prominent feature or combination of features within each cluster. This observation can also be deduced from *Means of features within the Cluster (Table 1)*. The higher the mean of a feature in a cluster, the higher its influence on the cluster. For example cluster 4 showed a high mean value for speachiness which detects the presence of spoken words in the song. This observation was verified by listening to the songs in the cluster 4 which consists of songs like “My Humps” by black eyed peas. Similar relationship was observed within other cluster as well as shown in table 2 representing the analysis of clusters.

Discussion

The results from this study did support the hypothesis that the basic Kmeans will sufficiently represent the data in a way in which a human would group the songs together. Based on the subjective perception of the representativeness of the clusters (see Figure 4), the clusters seemed to group the songs together like a human would. For example, cluster 4 representing high speechiness was mainly composed of hip-hop/rap songs. Despite the improvements created by Abubaker and Ashour (2013), Bahamani et al. (2012), and Na et al. (2010) the basic Kmeans was sufficient in clustering the songs in a way in which a human would.

The overarching confound is that dataset is based on a single person’s perception of the music rated on objective scales they created. For example, the danceability feature of a song or rather its value was based on a person's perception of danceability. So basically it’s rather a subjective analysis of the features.

The withiness between the cluster which measures the sum of squares between the clusters was low which implied that the clusters overlapped. As the clusters were overlapping no concrete information about the clusters could be interpreted. Moreover, the distance between the cluster should be more and within it should be less, but we are getting different results.

The clusters were made using k-mean , which clustered only numerical data . Trying other clustering techniques like k-node might have helped us cluster nominal data as well , which might have given us separate cluster (non-overlapping) and further helped us in classifying songs into different genres.

Future research could focus on correlating the complexity of the data set with the necessary complexity of K-means in order to efficiently and accurately represent the data. This

study showed that the basic K-means was sufficient in order to cluster the features of songs similar to how a human would. Identifying the desired complexity of the K-means algorithm and in relation to a dataset would save time and potentially wasted effort by future researchers and data analysts.

References

Abubaker, M., & Ashour, W. (2013). Efficient data clustering algorithms: Improvements over kmeans. *International Journal of Intelligent Systems and Applications, 5*(3), 37-49. doi:http://dx.doi.org.libproxy.utdallas.edu/10.5815/ijisa.2013.03.04

Bahamani, B., Moseley, B., Vattani, A., Kumar, R., & Vassilvitskii, S. (2012). Scalable k-means++. *Proceedings of the VLDB Endowment, 5*(7), 622-633. doi:https://dx.doi.org/10.14778/2180912.2180915

Na, S., Xumin, L., & Yong, G. (2010). Research on k-means clustering algorithm: An improved k-means clustering alorithm. *Third International Symposium on Intellignet Information Technology and Security Inofrmaticns,* 63-67. doi:https://doi.org/10.1109/IITSI.2010.74

Appendix

Audio Features Object

KEY	VALUE TYPE	VALUE DESCRIPTION
acousticness	float	A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic.
analysis_url	string	An HTTP URL to access the full audio analysis of this track. An access token is required to access this data.
danceability	float	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.
duration_ms	int	The duration of the track in milliseconds.
energy	float	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. For example, death metal has high energy, while a Bach prelude scores low on the scale. Perceptual features contributing to this attribute include dynamic range, perceived loudness, timbre, onset rate, and general entropy.
id	string	The Spotify ID for the track.
instrumentalness	float	Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
key	int	The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C♯/D♭, 2 = D, and so on.
liveness	float	Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
loudness	float	The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). Values typical range between -60 and 0 db.
mode	int	Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
speechiness	float	Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value. Values above 0.66 describe tracks that are probably made entirely of spoken words. Values between 0.33 and 0.66 describe tracks that may contain both music and speech, either in sections or layered, including such cases as rap music. Values below 0.33 most likely represent music and other non-speech-like tracks.
tempo	float	The overall estimated tempo of a track in beats per minute (BPM). In musical terminology, tempo is the speed or pace of a given piece and derives directly from the average beat duration.
time_signature	int	An estimated overall time signature of a track. The time signature (meter) is a notational convention to specify how many beats are in each bar (or measure).
track_href	string	A link to the Web API endpoint providing full details of the track.
type	string	The object type: "audio_features"
uri	string	The Spotify URI for the track.
valence	float	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).

Table 1

Means of features within the Cluster

	Speech	Duration	Time_Sig	Acoustic ness	Daceability	Energy	Instrumentalness	Livelin ess	Valence	Tempo
Cluster 1	-0.24	-0.38	0.07	-0.17	0.29	-0.16	9.16	0.14	0.34	0.15
Cluster 2	-0.49	0.28	0.11	-0.22	-0.48	-0.02	-0.07	-0.22	-1	-0.23
Cluster 3	0.03	-0.21	-0.5	-0.33	-1.13	0.58	-0.08	-0.07	-0.22	1.49
Cluster 4	2.3	0.35	0.37	0.12	0.55	-0.11	-0.14	-0.06	0.45	-0.35
Cluster 5	-0.29	-0.45	0.09	-0.23	0.48	0.50	-0.09	-0.27	2.06	-0.29
Cluster 6	0.05	-0.15	0.07	-0.38	0.07	0.56	-0.08	2.42	0.32	0.1
Cluster 7	-0.33	0.26	-0.99	2.74	-0.59	-2.06	-0.07	-0.35	-1.06	-0.16
Cluster 8	-0.17	0.52	0.03	1.78	0.82	-1.08	-0.12	-0.31	0.08	-0.43

Table 2

Analysis of Clusters

	Size	Features	Song picked from clusters
Cluster 1	12	High Instrumentalness , low danceability , low loudness (Black)	Lenny “Lady”
Cluster 2	245	Low Valence	BB “More than that”
Cluster 3	174	Low Danceability , low Tempo (Light green)	Nickelback-”Photograph”
Cluster 4	112	High Speechiness (dark blue/ purple)	BEB-”My Humps”
Cluster 5	334	High Valence	Taio Cruz- “Dynamite”
Cluster 6	97	High Liveliness (Pink)	U2 -”Beautiful Day”
Cluster 7	68	High Acousticness , low energy , slightly low valence & loudness(yellow)	ZZ Ward- “Last Love”
Cluster 8	154	Slightly low energy , low loudness	Joe- “I wanna Know”

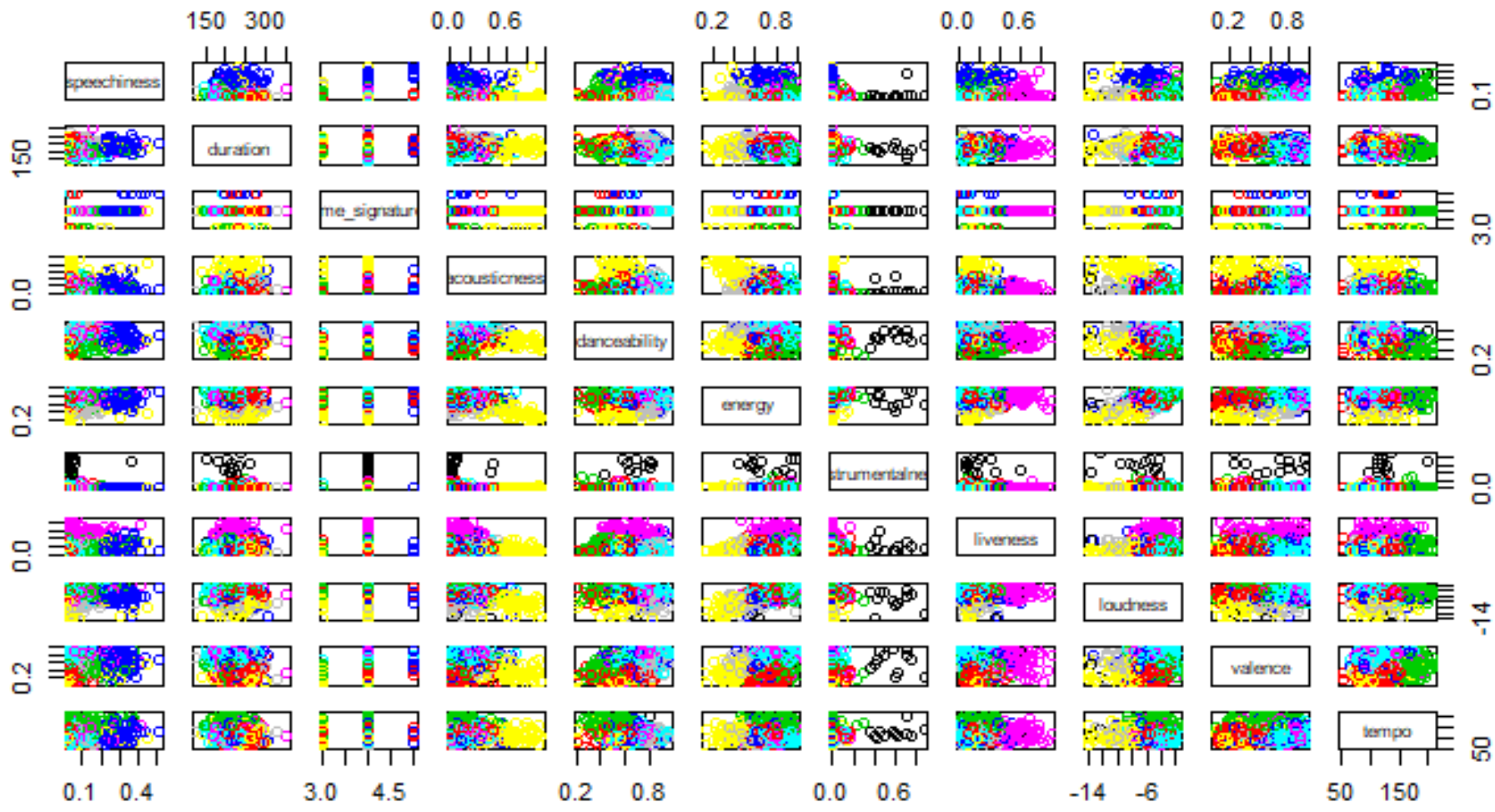

Figure 3. 2-dimensional representation of combinations of the different clusters.

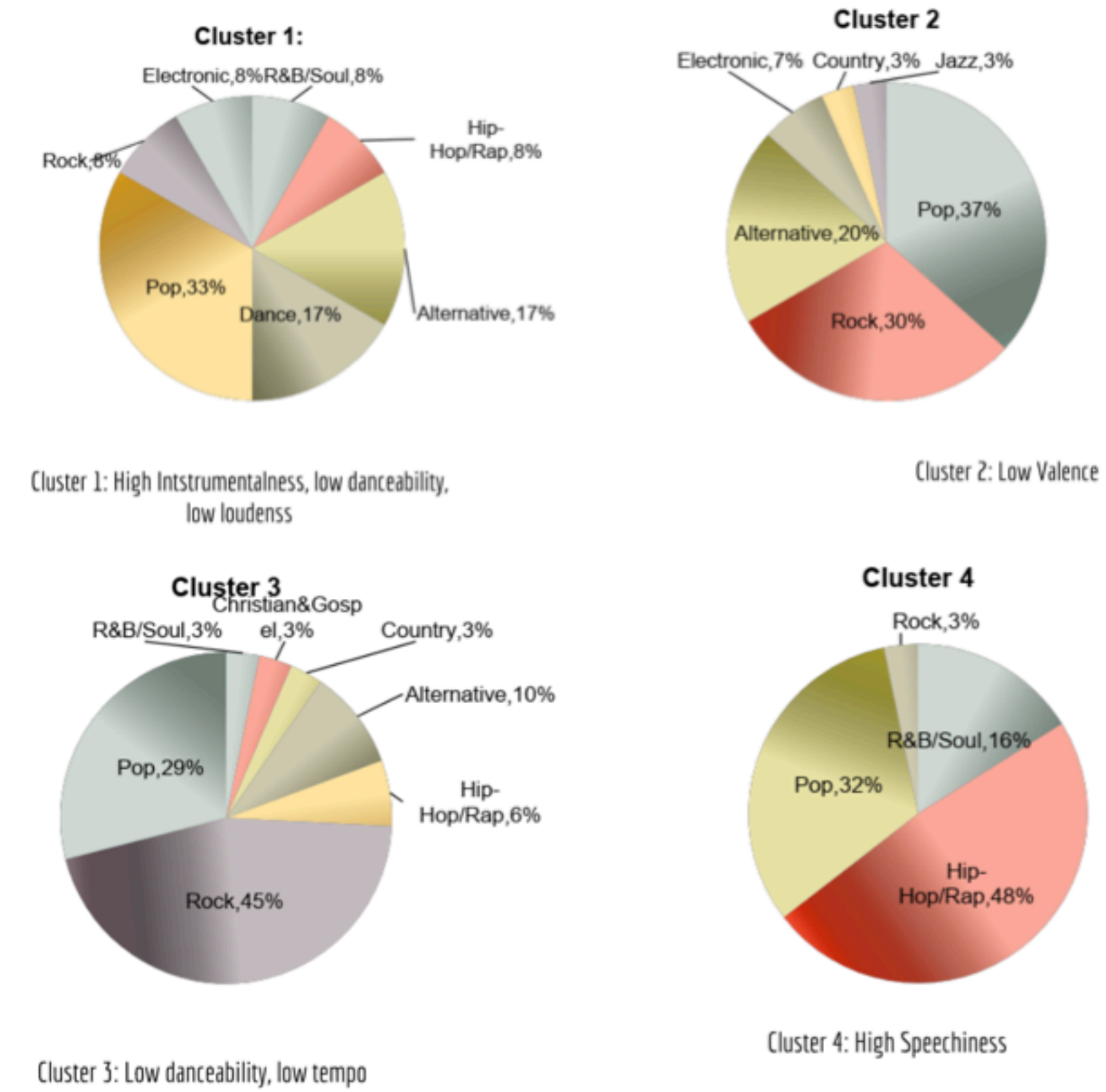

Figure 4. Composition of each of the first 4 clusters based on genre distribution.