Università di Pisa

Department of Computer Science

Master in Data Science & Business Informatics

DM1: Data Mining Foundation

Dataset: Spotify

Submitted To:

Prof. Dino Pedreschi

Prof. Riccardo Guidotti

Andrea Fedele

Submitted By:

Amir Hassan (683185)

Natthida Wiwatwicha (682482)

Academic Year 2023/24

CONTENT

TABLES

FIGURES

Chapter 1: DATA UNDERSTANDING AND PREPARATION

In this chapter we describe the characteristics of the dataset and our preprocessing decisions

1.1 Data Semantic

The Spotify Dataset is composed of a training test and a test set, each with 15000 instances equivalent to songs (also known as tracks or pieces of musical composition). The items have 24 attributes obtained from the Spotify catalog and also results of analysis of the audio files. The attributes include both categorical-qualitative (nominal type) and numerical-quantitative (discrete and continuous) data types. No categorical-quantitative ordinal data type is present, neither are Interval-scaled and ratio-scaled data types. We found that 2 attributes are redundant, so there are 23 unique attributes. **Table 1** "Nominal attributes", **Table 2** "Categorical attributes", and **Table 3** "Continuous numerical attributes" display the attribute names, their data format, descriptions, and interpretation or observations about their meaning. We observe that some numerical attributes (integer data format) should be considered categorical due to lack of logical order, namely *mode* and *key*, so we place them with other qualitative categorical attributes. In the Tables below, the text in quotation marks (i.e. "...") are the description taken from the readme file of the dataset.

Table 1.1 Characteristics of three nominal attributes

Attribute name	Data Format	Description	Interpretation/Observation
name	String	"The track length in milliseconds"	Notes about language: 1. Various languages are present, but also transliterations into roman alphabets. 2. Language is also not always indicative of *genre* as some non-english tracks may have english words. These observations can be important because some genres in this dataset are based on language/culture.
abum_name	String	"The album name in which the track appears"	
artists	String	"The artists' names who performed the track…"; multiple tracks share the same artists.	

Table 1.2 Characteristics of five categorical attributes

Attribute name	Data Format	Values	Description	Interpretation/Observation
explicit	Boolean	0 (False) or 1 (True)	"Whether or not the track has explicit lyrics (true = yes it does; false = no it does not OR unknown)"	All genres contain songs with explicit = True except for disney genre.
key	Integer	0 to 11	"The key the track is in. Integers map to pitches using standard Pitch Class notation."	1. There are disadvantages of integer notation to capture the musical quality of tracks. 2. Relationships between key and genre are complex due to diversity within genre. 3. Single key assignment does not capture modulation or key changes
mode	Integer	0 or 1	"Mode indicates the modality (major or minor) of a track. Major is represented by 1 and minor is 0"	The more frequent value in the data set is 1; therefore, it is reasonable to assume that the assignment of 1 to major modality and 0 to minor modality is correct.
time_signature	Integer	0-5	"An estimated time signature."	1. Describing time signatures using only 6 values is challenging as they are fractions with more than 6 possibilities, unlike integer-mapped keys. 2. "Estimating" time signatures does not make logical sense, as the quality represented by a time signature cannot be reduced to a numerical value (must maintain both numerator and denominator). Dividing these fractions (usually between 0 and 2) into 6 bins further makes the values less useful. 3. Songs often have multiple time signatures, and they cannot be combined or averaged. 4. If the description was a mistake and the integers represent qualitative categories of time signature, this attribute could be useful.
genre	String	20 categories	"The genre in which the track belongs"	1. Genre is fluid as one can have influences from another or more. 2. The concepts behind the categorization are multiple: some genres are based on the musical style while some are based on "culture", which could theoretically overlap with other genres that are based on audio or musical quality. This is important to note because

				some tasks may consider only musical qualities and could confuse genres for this reason 3. There exists at least 1 genre that theoretically could be considered a subset of or overlapped with another e.g. Forro vs. Brazil. This may be useful to note for classification tasks.

Table 1.3 Characteristics of sixteen numeric (continuous) attributes

Note: The value ranges are as found in the training data, and are only for reporting purposes or to get an idea whether a variable should be normalized.

Attribute name	Data Format	Value range of training set	Description	Interpretation/Observation
duration_ms	Integer	8587 to 4120258	"The track length in milliseconds."	Duplicates. Only *duriation_ms* will be used.
features_duration_ms	Integer	8587 to 4120258	"The duration of the track in milliseconds."	
popularity	Integer	0 to 94	"The popularity of a track is a value between 0 and 100, with 100 being the most popular."	-
popularity_confidence	Float	0 to 1	"The confidence, from 0.0 to 1.0, of the popularity of the song."	-
danceability	Float	0 to 0,98	"Danceability describes how suiTable a track is for dancing. A value of 0.0 is least danceable and 1.0 is most danceable."	-
energy	Float	0 to 1	"Energy is a measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity."	Could this be related to loudness?
loudness	Float	-49.531 to 3,156	"The overall loudness of a track in decibels (dB)"	Could this be related to energy?
speechiness	Float	0 to 0,939	"Speechiness detects the presence of spoken words in a track."	This should be inversely proportional to *instrumentalness*.
acousticness	Float	0 to 0,996	"A confidence measure from 0.0 to 1.0 of whether the track is acoustic. 1.0 represents high confidence the track is acoustic"	*Acousticness* refers to musical instruments that do not have "electric" qualities.
instrumentalness	Float	0 to 1	"Predicts whether a track contains no vocals. The closer the instrumentalness value is to 1.0"	This should be inversely proportional to *speechiness*..
liveness	Float	0 to 0,994	"Detects the presence of an audience in the recording."	-
valence	Float	0 to 0,995	"A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track."	Would high valence be related to mode = 1 (major key) while low valence to mode = 0 (minor key)?
tempo	Float	0 to 220,525	"The overall estimated tempo of a track in beats per minute (BPM)."	Tempo should be related to n_beats and n_bars
n_beats	integer	0 to 7348	"The total number of time intervals of beats throughout the track."	
n_bars	integer	0 to 2170	"The total number of time intervals of the bars throughout the track."	
processing	float	0,748 to 4,0467	Unknown	-

Both the training and test set have equal numbers of features from each genre (20 genres x 750 items = 15000 items per dataset), which suggests that the dataset would be **balanced** if in the future we choose **genre** as the output (target) class (Chapter 3). The implications of data type means that we have insights of operations that are

suiTable for these attributes (Cite: Data Understanding Lecture slides). For instance, nominal attributes may have operations such as mode, entropy contingency correlation, Chi-square test.

Another implication of our data type is the potential of binarization (Cite: Data Preparation slide). Because we have a mix of continuous and categorical attributes, binarization may be useful later if association analysis should be conducted. If this is the case, it will be in the Pattern Mining chapter (Chapter 4), and not in this chapter.

1.2 Variable Distribution

In studying the dataset's variable distribution, we choose to only look at the training set, to simulate real life situation. Plotting the histograms for each attribute as we have shown in Figure 1.1, we identify long-tail distribution *(speechiness, acousticness, instrumentalness, feature durations, liveness*). This means that they should be logtransformed if the methods of analysis that we will conduct perform better on data with normal distribution. Part 4 of this chapter will demonstrate the aforementioned attributes' distribution after log-transformation and square-root transformation. Figure 1.2 shows distributions of some example variables by genre which is a useful capability to start seeing different mean values and distributions across different genres.

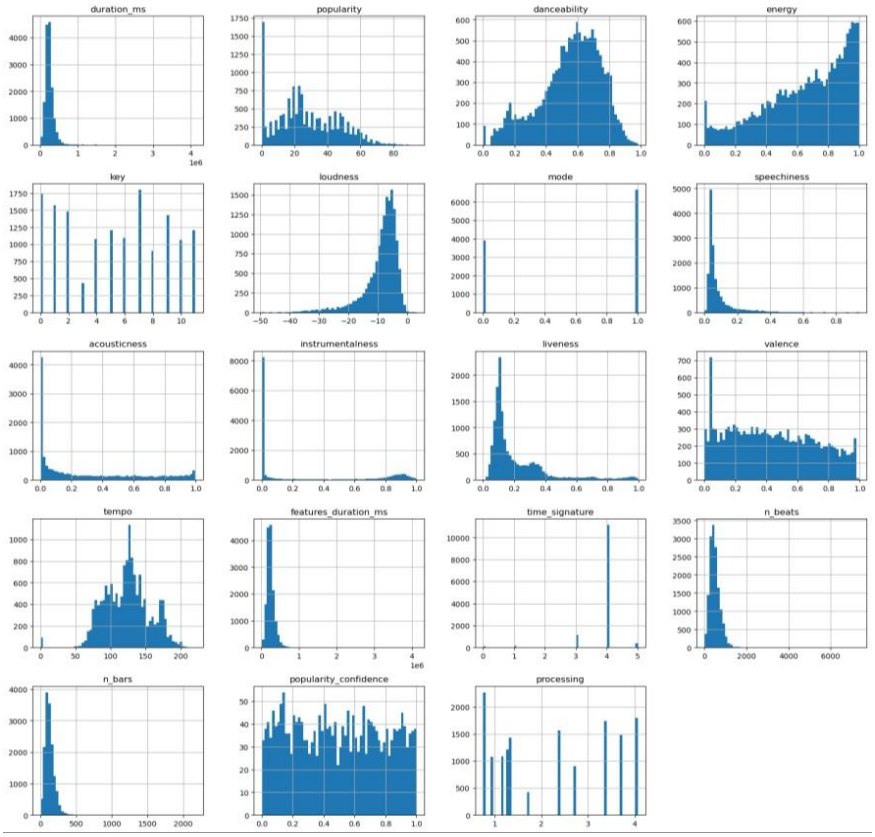


Figure 1.1 Data Distributions Graph of Numerical Columns

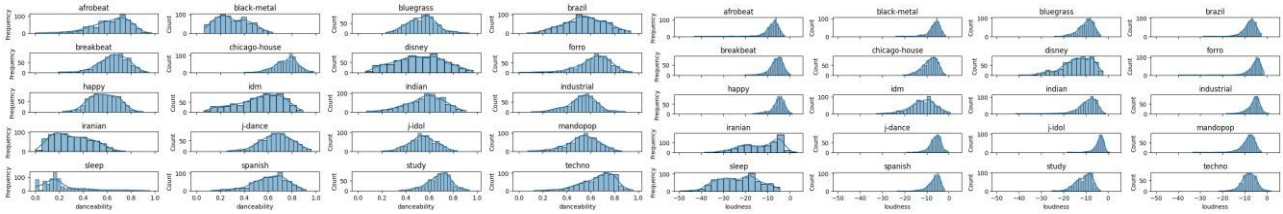


Figure 1.2 Examples of Data Distribution by Genre (left: Danceability; right: Loudness)

1.3 Assessing Data Quality

1.3.1 Errors

Delimiter settings in the common separated value file could cause parsing errors when previewed in various environments. However, the python data analysis library (pandas) bypasses these issues and is able to handle data without errors.

1.3.2 Missing values

Three attributes have missing value (Figure 1.2), namely, *mode* (4450 out of 15000 values are missing), *time_signature* (2062 out of 15000 values are missing), and *popularity_confidence* (12783 out of 15000 values are missing). Since the *mode* and *time signature* are qualitative categorical, it is difficult to fill in missing values. If necessary, will fill it with most frequent values per its *genre*, since we do not have missing value for *genre*. This means that we assume that there is a general tendency for a genre to have a prevalent modality (minor or major key) and also a prevalent time signature (characteristic of rhythmic profile). However, since the validity of *time_signature* attribute value is assumed to be questionable, the outlier treatment should not cause much (more) damage to the analysis. For the *popularity_confidence* attribute with continuous data type, we will impute the missing value with random values given its random distribution seen in Figure 1.1. In Chapter 4, Classification, we show that much of the missing values for these attributes are from the sleep *genre*, and their values are the same; naturally, imputing by mode (most frequent) for sleep genre improves classification performance for the sleep genre but not others.)

1.3.3 Outliers

Discarding 5%-25% of data from each variable may mean that a significant part of the dataset per each genre will be missing. We used IQR method to discard 12.5% upper and lower bound (Figure 1.3 and 1.4). However, in data mining tasks, we may omit outlier treatment to prioritize balanced training set for each genre.

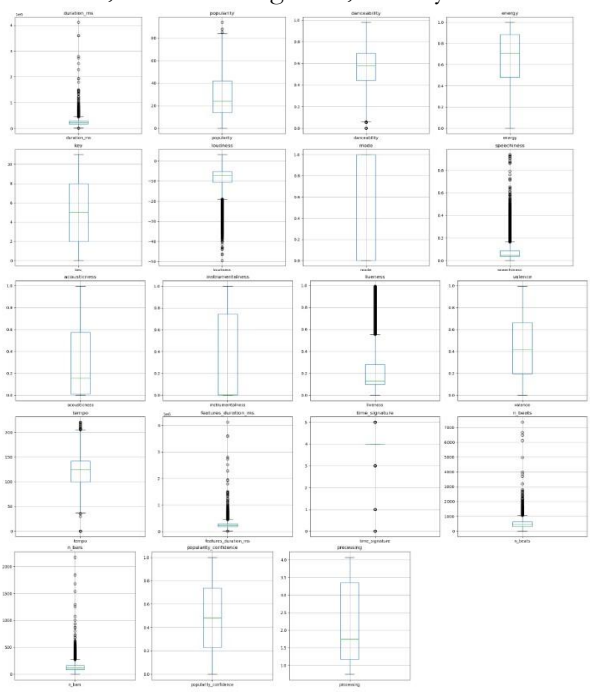


Figure 1.3: IQR Visualization of numerical variables before outlier treatment

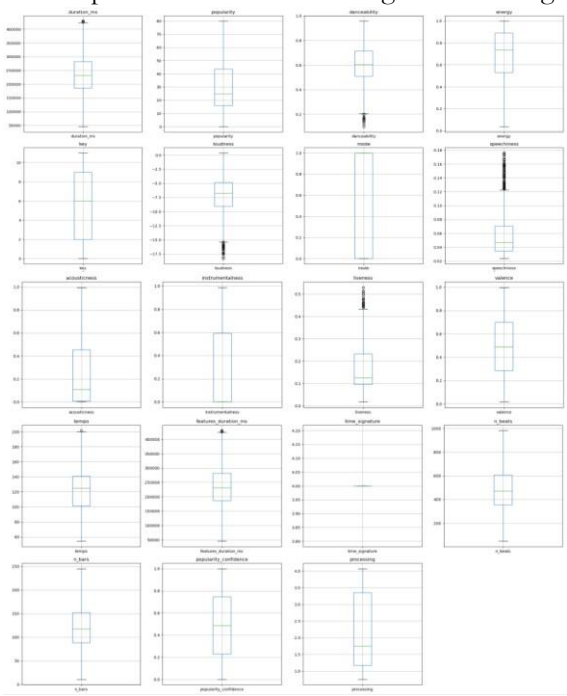


Figure 1.4: IQR Visualization after handling outliers

1.3.4 Semantic inconsistencies

For attributes with integer mapping, we found that **mode** should be mapped correctly (1 major vs. 0 minor) as the majority of music is in major modality based on the assumption that there are more major key songs than minor key songs on Spotify and that this dataset shares that assumption. This task is difficult because it require knowledge about the items in the dataset to be able to verify the semantic consistency.

1.4 Variable Transformations

Some attributes are shown to be long-tail distribution (highly positively or negatively skewed) namely, *speechiness, acousticness, instrumentalness,* and *duration*; therefore, log-transformation may improve data analysis performances for methods that are sensitive to high variance or expects normal distribution of data that could be seen from Figure 1.1. We find that log-transformation increases symmetry of the dataset and stabilizes variance. Additionally, square-root transformation performs better for moderately skewed variable distributions, while log transformation performs better for significantly skewed ones. Based on our visualization of the distribution of

speechiness, acousticness, and instrumentalness (Figure 1.5 left column from top to down); speechiness improves very much in its skewness after log-transformation, while acousticness and instrumentalness becomes slightly less flattened (see second column of Figure 1.5). The square-root transformation does not make the distribution less flat or skewed or more symmetric (see third column of Figure 1.5). However, Table 1.4 shows changes in variance, skewness, and kurtosis. We note that log-transformation worsen the variance because these variables are negatively skewed which increases noise and may effect performance of certain data mining tasks.

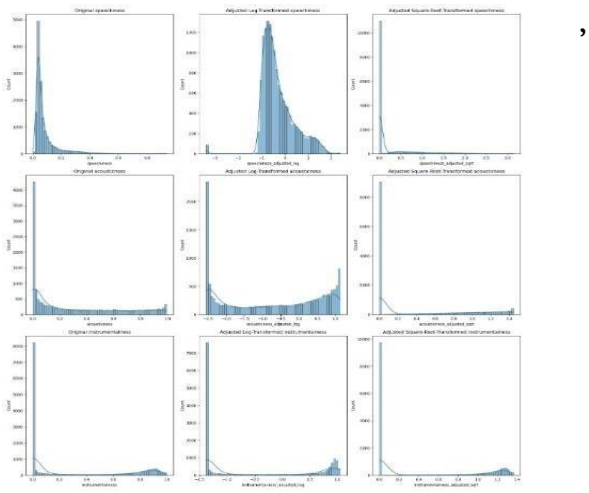


Figure 1.5 Distribution of variables (rows top to bottom): Speechiness, Acousticness, Instrumentalness, across transformations (columns left to right): Original, log-, and sqrt-transformed.

Table 1.4 Comparison of Variance, Skewness, Kurtosis of three variables before and after transformations

Original Data Statistics:

	speechiness	acousticness	instrumentalness
count	15000.000000	15000.000000	15000.000000
mean	0.083779	0.303896	0.286734
std	0.086709	0.329536	0.382930
min	0.000000	0.000000	0.000000
25%	0.037300	0.009745	0.000000
50%	0.051000	0.155000	0.003130
75%	0.088600	0.573000	0.744000
max	0.939000	0.996000	1.000000
variance	0.007518	0.108594	0.146636
skewness	3.120735	0.760625	0.776873
kurtosis	13.462732	-0.877141	-1.223643

Log-Transformed Data Summary Statistics:

	speechiness_log	acousticness_log	instrumentalness_log
count	15000.000000	15000.000000	15000.000000
mean	-0.289313	-0.735686	-0.539774
std	0.703760	1.350241	1.026917
min	-3.387548	-2.553529	-1.381462
25%	-0.767947	-2.231395	-1.381462
50%	-0.474873	-0.601179	-1.349443
75%	0.054107	0.596975	0.785783
max	2.385371	1.131479	1.051749
variance	0.495278	1.823151	1.054558
skewness	0.509143	-0.085663	0.563315
kurtosis	2.286825	-1.562648	-1.550660

Square-Root-Transformed Data Summary Statistics:

	speechiness_sqrt	acousticness_sqrt	instrumentalness_sqrt
count	15000.000000	15000.000000	15000.000000
mean	0.252778	0.392051	0.389368
std	0.507167	0.532456	0.552093
min	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000
75%	0.235791	0.903668	1.092759
max	3.140556	1.449220	1.364790
variance	0.257219	0.283509	0.304806
skewness	2.123448	0.837633	0.791548
kurtosis	3.873794	-1.000725	-1.271444

1.5 Pairwise correlations and elimination of variables

We analyze correlations between variables. Highly correlated variables can inform elimination choice, Figure 1.5. Compared to our expectations from observations in the Data Semantics part of this chapter, the Matrix Correlation result confirmed or denied. Our hypothesis that speechiness and instrumentalness are an inversion of one another. The Matrix Correlation does not confirm our expectation. The heatmap show the correlation matrix (Figure 1.6). There is a strong correlation between duration_ms and feature_duration_ms, showing that these columns are linearly dependent and effectively convey the same information. Meanwhile, n_beats and n_bars are showing second strongest correlation that we can see from heatmap.

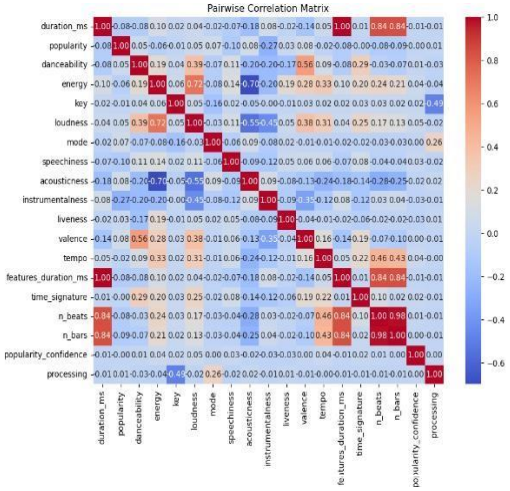


Figure 1.6: Correlation Heatmap

1.6 Data Preparation Conclusion

In conclusion, we have removed feature_duration_ms to eliminate redundancy. We also remove n_beats or n_bars, in many data mining processes due to their linear correlation, but keep both when it should not affect the result (such as decision tree classification). We normalized some attributes and also log-transform, but may omit these transformations in data mining methods that are not sensitive to the data characteristics previous to these transformations. The same logic applies to our treatment of missing data where we fill *mode* and *time signature* with the mode value per genre, and popularity confidence randomly informed by random distribution. We also may omit outlier elimination in favor for balanced dataset (equal data points per genre).

Chapter 2: CLUSTERING

In this chapter, we have applied different algorithms to perform clustering. We've applied K-Mean, DBscan and Hierarchical clustering to discover clusters from our data set. To perform clustering, we have chosen *popularity*, *danceability*, *energy* and *speechiness* columns.

2.1 Clustering with K-Means

For K-mean clustering algorithm, we have used elbow method and silhouette method to see the results on different k values. We got sse=1564.896 and silhouette=0.327 We observed that when K value increase and SSE decrease. We observed that K value increase and SSE decrease because more centroids means that each cluster could be more tightly packed. Meanwhile, silhouette score indicates how our points are well-matched to their own cluster and poorly matched to neighboring clusters. Figure 2.1 plots informed us that we want to explore k values smaller than 5.

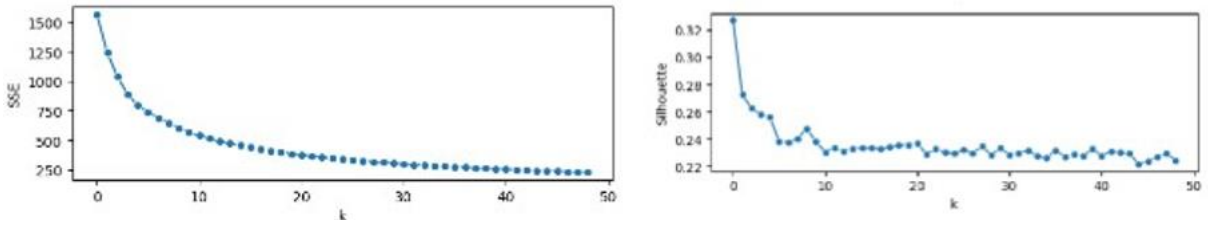


Figure 2.1 Relationship between k with SSE and Silhouette Score

By looking at the graphs in Figure 2.1, we were able to visually assess that SSE decrease significantly up to k=3 and then slowly decreasing, this indicates that k=3 capture most of the structure in data without any unnecessary complexity while silhouette reaches a reasonable peak at k=3, suggesting that clusters are well separated and coherent.

However, we also evaluate visually k=2, 3, and 5, and compare result in Table 2.1 which shows that k=3 gives us the best silhouette score, and better SSE (lower meaning more compact) than k=2 which has competing silhouette score. A lower SSE is generally desirable as it indicates better clustering in terms of compactness. Figure 2.2 and 2.3 shows line graph and scatter plots for clusters for k=3 and k=5 respectively.

K	SSE	Silhouette
2	2004.906	0.286
3	1642.566	0.289
5	1160.964	0.258

Table 2.1: K_Mean Clustering Result

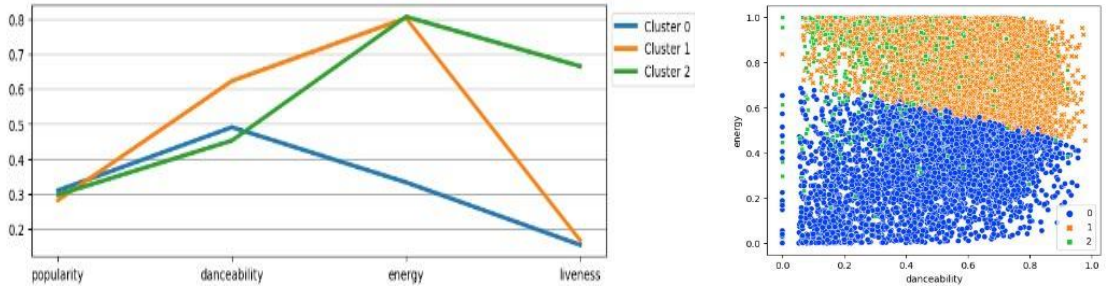


Figure 2.2 Line and all cluster Graph for Centroid positions based on the featured clustered for k = 3

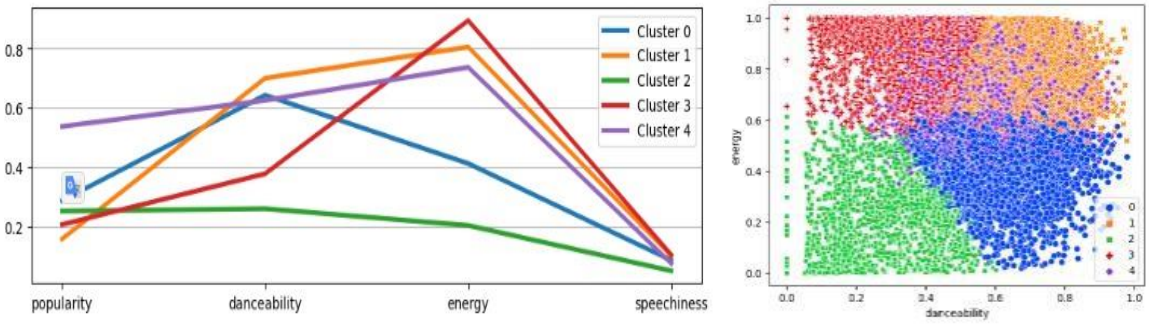


Figure2.3 Line and all cluster Graph for Centroid positions based on the featured clustered for k = 5

In multi-dimensional K-mean which cannot be visualized, we observed three clusters; they generally are similar in popularity dimension and more or less in the danceability dimension, but differ much in energy and liveness aspects:
Cluster 1 contains songs lower energy and low liveness compared to others. This cluster likely to represent studio-recorded mainstream music that is moderately appropriate for dance to but not too energetic.
Cluster 2 has much higher danceability than the other two clusters, high energy but low liveness. This cluster likely includes highly energetic, danceable studio recorded songs.
Cluster 3 is very similar to cluster 2 but distinguishes itself in high liveness, which means it can be live music with danceability and highly energetic qualities.

2.2. Clustering with Bisecting K-Mean

In the bisecting k-mean section, we have shown five clusters based on *danceability* and *speechiness*, to see the clustering differences between K-mean and Bisection K-mean. The silhouette score of BK-Mean is 0.220, which is lower than K-means. We tested a few k values and find that the data is very difficult to parse as cluster even visually (see Figure 2.4-2.5), and even with k=2 it is hard to discern clusters.

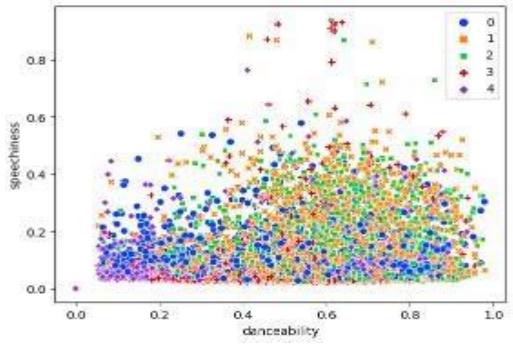


Figure 2.4 Clusters of speechiness and danceability in BK-means (k = 5)

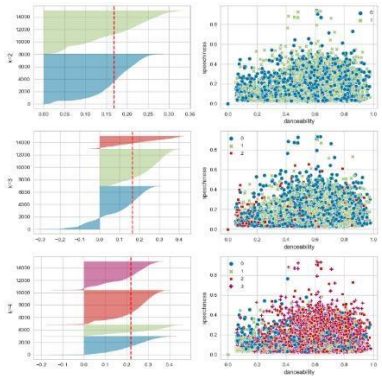


Figure 2.5 Silhouette Visualizer and Clusters with Clusters of speechiness and danceability in BK-means (k=2, 3 and 4)

2.3. DBSCAN

DBSCAN is an unsupervised density-based clustering algorithm. We compared several 2D and 3D clustering to test different sets of attributes (see Table 2.1). We used Silhouette Score to determine the relative best epsilon and minimum sample parameters for each set of attributes (Features column in Table 2.2). To compare across different attribute sets, we use SSE score. We do not study entropy in DBSCAN because we cannot control the number of clusters to match the genre label (true label), and neither did we attempt to group the true labels of genre into more generalized groups of genres. We find that for DBSCAN, when Silhouette Score improves for a clustering of a set of features, its SSE worsens, which may reflect that DBSCAN has issue with non-uniform densities of clusters. While dense regions might be clustering well (improving the Silhouette Score), more spread-out regions (which could still be valid clusters) might be increasing the overall SSE.

Table 2.2 DBSCAN Silhouette Scores and SSE

Features	Parameters (Epsilon, min sample)	Number of clusters (including -1 label)	Silhouette Score	SSE
Danceability and Energy	0.05, 50	3	0.276	1448
	0.0467, 11	3	0.367	1551
Danceability and Speechiness	0.05, 50	2	0.400	590.4
	0.082, 2	3	0.650	669.1

Danceability, Energy, Speechiness, Popularity	0.115, 10	101	0.267	806.2
	0.115, 20	67	-0.081	472.0
	0.2, 5	94	0.587	1530.2

Since DBSCAN should be able to detect dense regions of points, which are separated by low-density regions, it is difficult to parse dense data, while it is able to detect low density regions as noise. Therefore, in SSE analysis we use clusters with -1 label (noise) as another cluster for the purpose of scoring. Otherwise, we would have only 1 cluster in some cases. Furthermore, sometimes DBSCAN evaluates that the majority of the dataset is noise (with label -1). We also find that Silhouette score alone does not speak much of the quality of clustering in 2D when evaluated visually; therefore, clustering in higher dimensions should not rely on just silhouette score alone. In a future study, we could try sampling the data to be more sparse (such as taking only 10% of the sample from each genre) to see if DBSCAN performs better.

Figure 2.5 and 2.6 shows 2D clustering for *danceability* and *energy*, and *danceability* and *speechiness* respectively. We expect that DBSCAN would be able to capture would be able to capture the data points where danceability = 0. However, it does not as it has to compromise distances in both dimensions. In Figure 2.6 through the second and third clusters are very small, they refer to tracks that have slightly lower danceability vs slightly higher danceability where the speechiness is very high. Interpreting this, we can conclude that these clusters are not very useful.

Figure 2.5 Danceability x Energy (Left to right) A. colored by genre; B. Epsilon = 0.05, Min sample = 50 ; C. Epsilon = 0.0467, Min sample = 11; the latter has slightly better SS but slightly worse SSE.

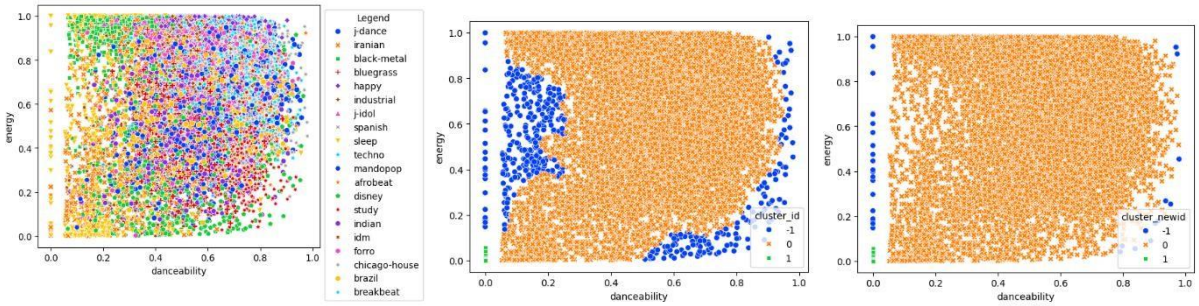
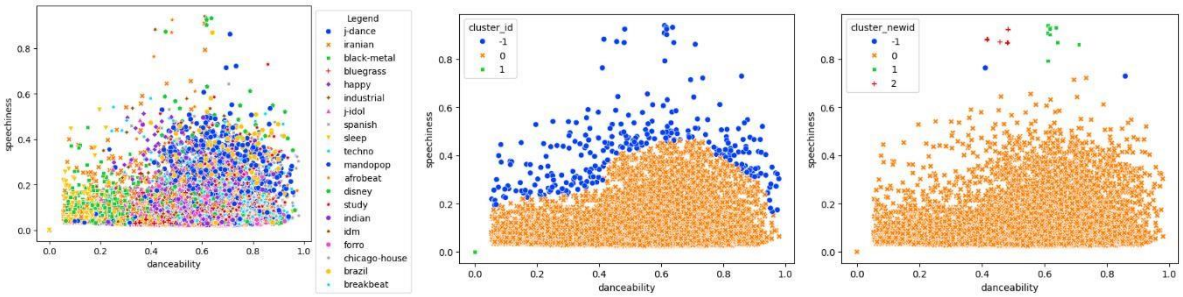


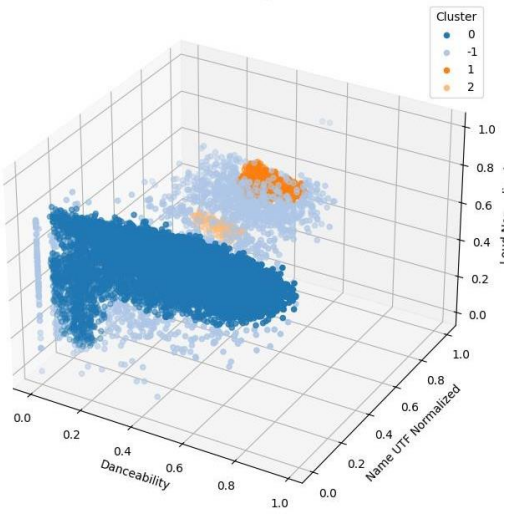
Figure 2.6 Danceability x Speechiness A. colored by genre; B. Epsilon = 0.05 , Min sample = 50; C. Epsilon = 0.082 , Min sample = 2; the latter has slightly better SS but slightly worse SSE.



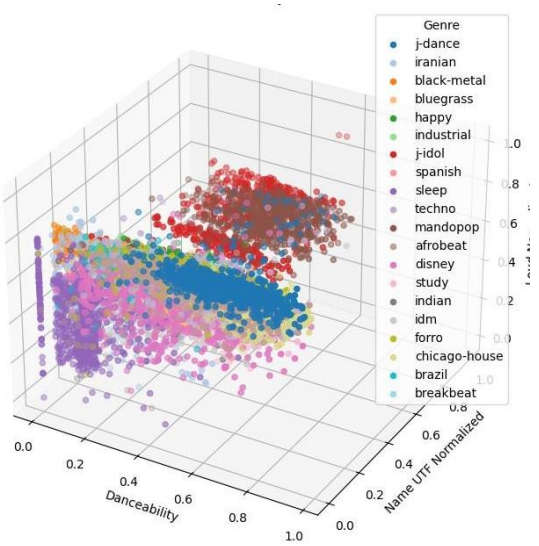
In DBSCAN exercise, we want to experiment with some data variable that would behave well or have some clear expectation in clustering, so we create a column that stores highest UTF value of the letters in the track name, hoping to capture whether the track name is in Chinese, Japanese, English, Arabic, or has accented latin alphabet. This dimension gives useful interpretability of clusters whether colored by genre or by DBSCAN (Figure 2.7 and 2.8). With the parameters Epsilon = 0.05, Min sample = 50, though the silhouette score is 0.07, we can see the 3 clusters separated by Name UTF variable more than other two variables, roughly dividing the data into 3 approximate character types in the track names (Figure 2.7). However, if colored by genre, we can see some meaningful clusters (Figure 2.8); for example, tracks with low danceability and loudness with low UTF max value in name are sleep tracks; we can also see two clusters in blue and red colors. The blue cluster is low, medium to

high UTF max value, high danceability and low loudness, corresponding to j-dance, which may have English track names or Japanese characters. While the red and dark red mixed cluster is a mixture between j-idol and mando-pop which share qualities being both pop music (high loudness and medium range of danceability) and sharing some alphabest as mandopop uses Chinese characters and some version of Japanese (Kanji) uses the same characters as Chinese, so we see a patch of red with medium to high UTF values. However, it is reasonable that clustering algorithms are not able to capture similar clusters due to unclear boundaries between datapoints.

2.7. 3D scatterplot of Dancebility, Loudness, and max-UTF-Name variables, colored by DBSCAN clusters; Epsilon = 0.05, min_sample 50.



2.8. 3D scatterplot of Dancebility, Loudness, and max-UTF-Name variables, colored by genre.



2.4. Hierarchical clustering

In the hierarchical clustering analysis, we have used AgglomerationClustering model with parameter set to enable the computation of full tree. By using complete linkage criterion and Euclidean distance metric, we have given the training set data to generate dendrogram to visualize the hierarchical structure of the clustered data points. The dendrogram gives us an insight into the clustering process, displaying the merging of clusters based on their proximity, with a color threshold set at 1.2 to help us in cluster distinction (Figure 2.9).This visualization facilitates the interpretation of the relationship between data points and the identification of potential clusters within the dataset.

Table: 2.3 Silhouette Score vs Euclidian Distance (3 clusters)

Euclidean Distance	Silhouette Score
Ward Linkage	0.166
Complete	0.433
Average	0.452

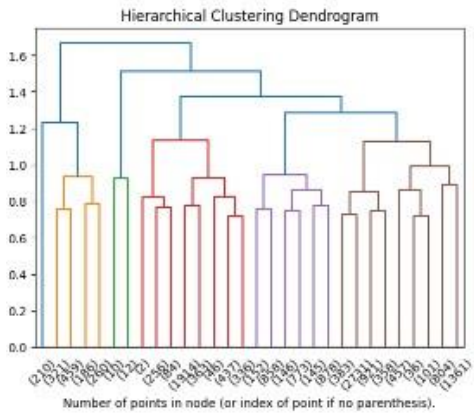


Figure: 2.9 Dendrogram clusters graph

Then we used Euclidean distance meric on agglomerative hierarchical clustering algorithm and compared the result with different clustering measures and find that the best linkage according to the silhouette value is the average linkage. We studied hierarchical clustering with a predefined cluster count and visualized the result via a scatter plot to illustrate cluster distribution on k=3 (Figure 2.10). We find that the third cluster is not discerned very well and mixed with the first two which are characterized just by lower danceability and lower energy vs. higher danceability and higher energy or having just high danceability and low energy and vice versa.) Though visually one can see a diagonal dividing the two clusters, the blue cluster covers too much characteristics that are even opposite of one another. This shows that sometimes the scores are not enough measure to guarantee meaningful clusters to human eye. Lastly, we have used connectivity constraints to refine

clustering by considering the neighboring samples' structure, enhancing cluster chart and silhouette score 0.3 as shown in the hierarchical clustering dendrogram in Figure 2.10. As before, the more the number of clusters, the lower silhouette score.

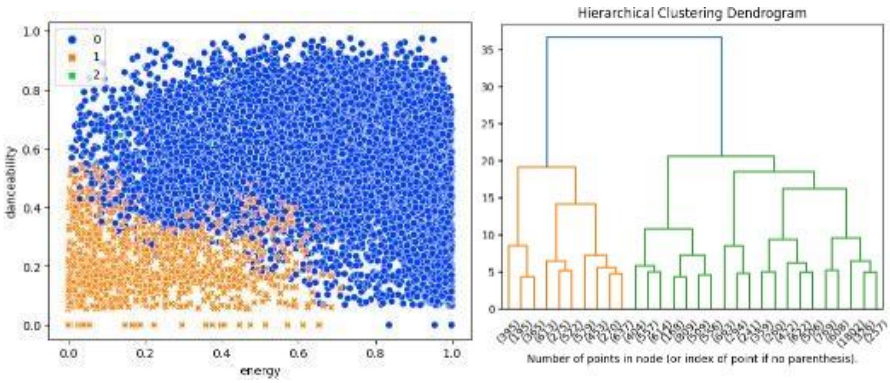


Figure: 2.10. K Clustering and Connectivity Constraint Visualization (3 clusters)

Agglomerative clustering generating seven clusters as depicted in Figure 2.9. Cluster 0 has high speechiness but low acousticness and instrumentalness. Cluster 1 features low speechiness but high acousticness and instrumentalness. Cluster 2 exhibits moderate levels of speechiness, acousticness, and instrumentalness. Cluster 3 is marked by high speechiness and acousticness but low instrumentalness. Cluster 4 is characterized by low speechiness, acousticness, and high instrumentalness, likely instrumental tracks with minimal speech and acoustic elements. Cluster 5 shows medium speechiness, low acousticness, and medium instrumentalness. Lastly, Cluster 6 displays low speechiness, medium acousticness, and low instrumentalness, likely indicating acoustic tracks with minimal speech and instrumentation. Overall, most clusters seem to have distinct characteristics that would allow for reasonable separation. However, there may be some overlap, particularly with Cluster 2 (balanced characteristics) and Cluster 5 (medium levels of speechiness and instrumentalness). The clusters with extreme values (e.g., very high or very low) in specific features are likely to separate well. Corresponding to Dendogram in Figure 2.10, Cluster 0 is marked by lower speechiness and higher acousticness, implying fewer spoken words and more acoustic tracks. It also shows a higher instrumentalness, indicating a likelihood of instrumental tracks, and moderate liveness, suggesting occasional live performances. Conversely, Cluster 1 exhibits higher speechiness but lower acousticness, indicating more spoken words and fewer acoustic tracks. It also displays lower instrumentalness and higher liveness, suggesting fewer instrumental tracks and more live performances. Cluster 2 strikes a balance with moderate speechiness and acousticness, suggesting a mix of spoken and non-spoken tracks, and acoustic and non-acoustic tracks, respectively. Its instrumentalness is moderate, indicating a mix of instrumental and vocal tracks, with lower liveness, suggesting fewer live performances.

2.5 Final Discussion

After observing K-mean, BK-Mean, DB-Scan, and Hierarchical clustering algorithms, we compare the result based on silhouette and SSE score metric, searching for higher silhouette score and lower SSE for the "best" result for choosing the clusters. We have shown the values in the below Table 2.4.

We find that K-means is effective for well-separated clusters, but struggles with clusters of varying shapes and densities. Bisecting K-means as an alternative approach to K-means, offering a hierarchical splitting method that can sometimes yield better results for specific datasets but not our spotify dataset. In our experiments with DBSCAN (Density-Based Spatial Clustering of Applications with Noise), we tested various parameter combinations and encountered challenges with nonuniform cluster densities and noise detection, highlighting its sensitivity to parameter choices and difficulties with varying cluster shapes. Hierarchical clustering using the AgglomerativeClustering algorithm provided competitive results with DBSCAN. Among the various methods, hierarchical clustering with complete linkage and the Euclidean distance metric emerged as the best clustering algorithm based on our evaluation criteria though it lacks SSE, and it has potential for more interpretability of clusters. However, the clustering results in this exercise are not very discernable, and we hope to study clustering with more sets of parameters to further improve the results.

Table: 2.4 Comparison of Best Silhouette Score and SSE of K-Mean, BK-Mean, DBScan and Hierarchical Clustering

Clustering	Silhouettes Score	SSE Score
K-Mean	0.327	1564.89
BK-Mean	0.220	N/A
DBSCAN	0.276/0.650	472.0/1551.0
Hierarchical	0.452	N/A

Chapter 3: CLASSIFICATION

We studied the performance of 3 classification algorithms to predict target variable "genre": Decision Tree, KNN, and Gaussian Naïve Bayes. This study did not turn all variables into categorical bins to run Categorical Naïve Bayes. Below we discuss tunings and best performing instance of each classifier type and the comparison of their best performances. In this study, we introduce new variable called UTF-name which is the max UTF value of the letters in a string of the track name. We found that it helped increase accuracy from below 0.2 to around 0.5 in Decision Tree and applied it across 3 algorithms. We also created Calculated_Duration which is (n_beats/tempo) to get the number of seconds; this is not used for Naïve Bayes because it is not independent.

3.1 Decision Tree

We first tested the algorithm with only numerical (continuous) attributes as a baseline to compare the result with KNN and Gaussian Naïve Bayes which require this input type. We found that the model is severely overfit (0.99 training accuracy) while test accuracy is 0.29.

With a balanced dataset, the splits are less dependent on class imbalance. Both Gini and entropy splitting criteria shows similar results, so we choose Entropy as the split criterion and tune max depths, min sample splits, and min sample leaf. With best result from grid search (min sample split = 80, min leaf split = 20, and max depth = 9), we were able to raise test accuracy to 0.35. Pruning greatly simplify the tree but reduced accuracy to 0.25.

Next, we trained the model with all attributes in the dataset including categorical attributes, and found a new parameter configuration: min sample split = 10, min leaf split = 10, and max depth = 8. The resulted tree is much more complex, but the test accuracy increases to 0.43. Random forest improves the accuracy to 0.45. Now we add the custom formula for UTF-name/artist/album, and calculated-duration, the accuracy reaches 0.511 and, with random forest, to 0.5022. CCP Alpha at 0.006 reduces accuracy to 0.45 though it achieved this in 104 nodes instead of 310 nodes from the random search forest. We noticed that sometimes random forest reduces accuracy, and this may be because some variables maybe strong predictors for certain classes but not the other, meaning some class has noisier data than others. Comparing nodes and feature importance, it shows that not necessarily the most important features (*popularity, danceability*, UTF-name) are at the top (*instrumentalness*) or near the top of the three trees (UTF-name, *acousticness, energy*). We found that features with zero importance are: key, mode, time signature, and processing.

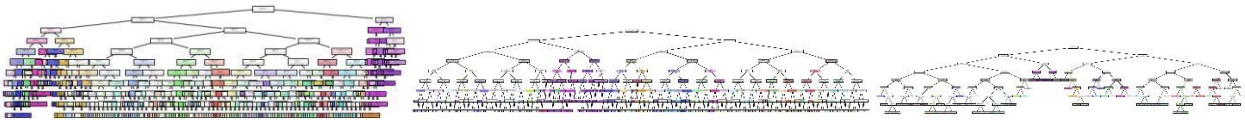


Figure 3.1 Left to Right, best decision tree, after random forest, and after CCP alpha.

Random forest yields more balanced tree and close accuracy to the full tree. More balanced tree helps prevent overfitting, improve computational efficiency, lessen bias and give more consistent prediction result.

3.2 KNN

K-nearest neighbor is a non-parametric, supervised learning classifier using the number of neighbors to classify data, and performs well especially in systems where decision boundaries are irregular. We use all continuous variables for KNN, except durational variables and popularity confidence (seems randomly distributed). Since KNN struggles with high dimensionality, we reduce the number of variables and found that it performs better with reduced number of variables but worse when it is reduced too much (to 4 variables). We conduct grid search for the best performing set of variables.

Best performing set of variables: danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, tempo, UTF-name, calculated-duration. Best Parameters from grid search: 'metric': 'cityblock', 'n_neighbors': 25, 'weights': 'distance' with clf score = 0.585. Figure 3.2 shows best k = 25.

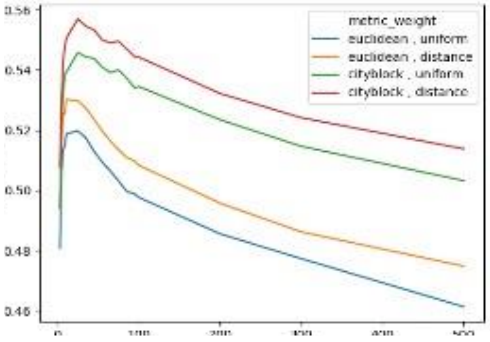


Figure 3.2 Comparing mean test score for k-values

3.3 Naïve Bayes

Naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features. It performs well with high dimensions, and works well for multi-class prediction. It performs well with categorical input variables, and text classification, but can handle numerical data by assuming a distribution (e.g., Gaussian distribution). However, we choose to test only Gaussian to be able to compare the same set of variables with KNN and DT. We found that with the same variables as best performing set in KNN, the accuracy is only around 0.22-0.3, not comparable to 0.5 of KNN. After eliminating some variables which we assume to be "dependent" on other variables (only based on semantics) such as danceability or liveness, we found that the best variable set for Naïve Bayes is: popularity, speechiness, acousticness, instrumentalness, valence, tempo, energy, loudness, UTF-name. We did not attempt variable smoothing for hyperparameter tuning since the range of performance does not seem comparable to KNN or DT.

3.4 Comparison and Conclusion

Table 3.1 Comparison of best performing model of each classifier DT, KNN, Bayes

		DT	**KNN**	**Bayes (Gaussian)**
Overall performance matrix	Accuracy (f-1)	0.51	0.49	0.38
	Precision (macro avg)	0.54	0.49	0.39
	recall	0.51	0.49	0.38
	F1	0.52	0.48	0.35
	support	5000	5000	5000
Detailed Performance Metrics	ROC micro-avg	0.90	0.90	0.87
	ROC macro-avg	0.89	0.89	0.87
	Precision-recall micro average	0.49	0.51	0.37

Decision Trees shows better general metrics like accuracy, precision, recall, and F1 score while having a slightly worse precision-recall micro average than KNN. This happens because precision-recall metrics focus more on the handling of positive instances, which can vary in subtle ways not captured by the overall metrics.

The ROC curve shows the trade-off between sensitivity (or TPR) and specificity, and shows the almost identical best top 7 performing classes (genre) though their rankings are slightly different. For DT, best ROC area values are class 18 [0.96], 15, 1, 5, 12, 16, 7; for KNN are 15 [0.96], 16, 7, 5, 12, 14, 18; while for Naïve Bayes are class 18 [0.98], 16, 7, 15, 1, 5, 12. They are 18-study, 15-mando pop, 16-sleep, 7-forro, 5-chicago house, 12-iranian, and 1-blackmetal, while 14 is j-idol (only shows up in top list of KNN). This may possibly mean that the language difference from UTF value column has more impact on classification in KNN than decision tree and Naïve Bayes. The absolute worst performing class across classifier is 0-afrobeat.

The three models offer the same value of best precision-recall of their best genre (0.792-0.793), though slightly different class: DT's best class is 15 (mandopop), KNN is 15, 16 (mandopop, sleep), Naïve Bayes is 18 (study). Generally, the worst performing classes from the perspective of precision-recall across the 3 classifiers are 0afrobeat, 4-breakbeat, and 11-industrial. ROC average for Naïve bayes is close to the better performing models while precision-recall micro average is worse, possibly due to chosen variables are not truly independent or have complex relationships.

Confusion matrix shows that there is most confusion in the Naïve Bayes classifier which corresponds to its lowest ROC/Precision-recall evaluations compared to DT and KNN. The confusion is distributed across the matrix, with most prominent confusion is mandopop is confused for Brazil, which is not clear why. On the other hand, the confusion matrix for DT is relatively very clean with most confusion occurring at true class 17 (Spanish) being confused for Brazil (65), Forro (50), Indian (50), and J-dance (50); it is might be explainable that Spanish, Brazil, and Forro share similar music influences as Spanish music, and this assumption aligns with the fact that true Brazil and Forro classes do get classified as Spanish and each one another as well. Surprising is that Indian class is also in this mix, which means that they might have similar characteristics despite not sharing the same influence. Meanwhile, these aforementioned 4 classes are not misclassified to be J-dance, so simi lar conclusions cannot be drawn though Spanish is equally misclassified to be J-dance as the other three genres. The

shared influence between Spanish, Forro and Brazil is supported again in the confusion matrix of KNN but only for true Forro and Brazil classes being misclassified as one another and also Spanish, but true Spanish class is not misclassified to be Brazil and Forro. KNN's confusion matrix also shows equally strong confusion for true bluegrass to be misclassified as afrobeat, Disney, and industrial, which is not intuitive given domain knowledge about these genres. Overall, DT confusion matrix gives more interpretable insights consistent with domain knowledge more than KNN or Naïve Bayes, but the lack of inaccuracy also misses out some insights from misclassification, such as the fact that KNN shows reciprocal confusion between mandopop and j-idol, which does not appear in Decision Tree (not enough significant confusion).

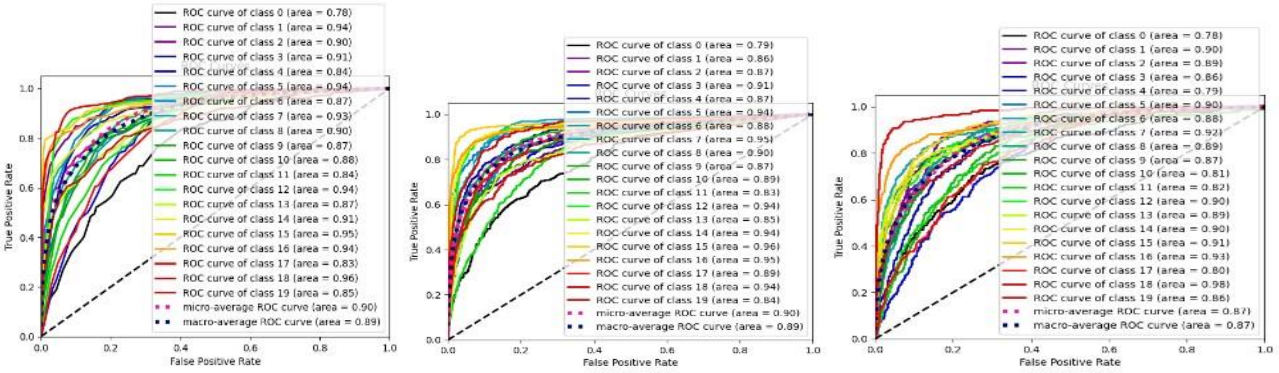


Figure 3.3 ROC Curve (DT, KNN, Naïve Bayes)

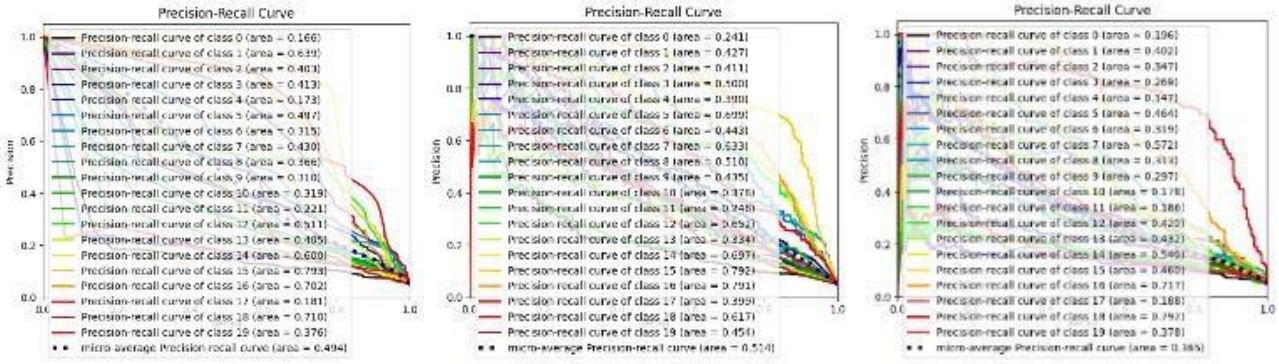


Figure 3.4 Precision-Recall Curve (DT, KNN, Naïve Bayes):

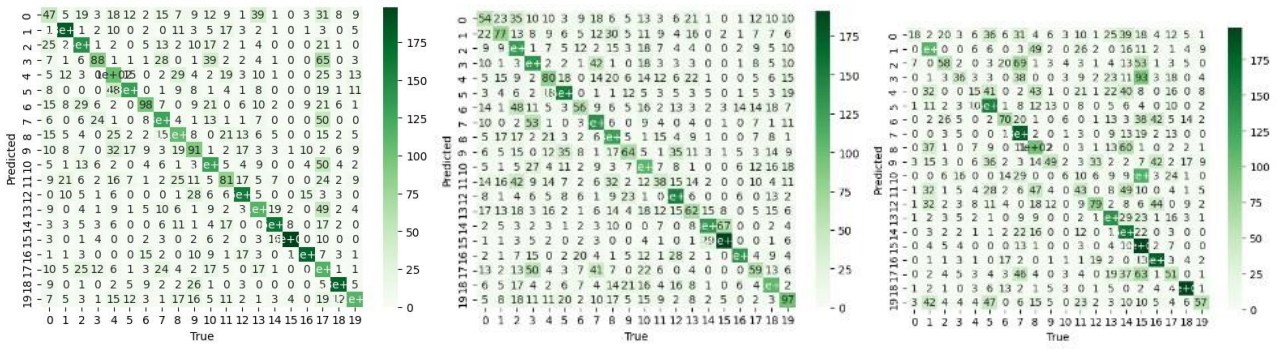


Figure 3.5 Confusion Matrix (DT, KNN, Naïve Bayes):

Note: the genre labels for Figure 3.3-3.5 are: afrobeat: 0, black-metal: 1, bluegrass: 2, brazil: 3, breakbeat: 4, chicago-house: 5, disney: 6, forro: 7, happy: 8, idm: 9, indian: 10, industrial: 11, iranian: 12, j-dance: 13, j-idol: 14, mandopop: 15, sleep: 16, spanish: 17, study: 18, techno: 19

To conclude, KNN performs very well considered it could take only continuous variables and performs well with only 11 variables, yielding performance close to best performance of Decision Tree, which uses almost all of the variables. However, confusion matrix of Decision Tree reveals more insights about the similarity between classes, but comparing it with KNN's also shows which observations are more supported (ie: more consistent confusion between Brazil, Forro, and Spanish classes.) We learned that better overall performance could have slightly worst ROC and Precision-recall, yet its confusion matrix can be more interpretable (Decision Tree), though two confusion matrices from two comparably good models can reveal more information than one.

Chapter 4: PATTERN MINING

To study apriori algorithm for frequent, closed, maximum item sets and association rules, we filled missing values with mode value (most frequent) for each genre except popularity confidence which is filled randomly based on its distribution and removed string-based columns. Transformations are not necessary because association rules can handle raw data with various types of distribution. Next, we discretized the remaining variables, assigning labels to the bins, generally Lo-, Mid-, Hi, to conceptualize values as items., while each 'genre' is its own bin. We also removed 'processing' and, later on, 'time_signature' because they are not interpretable or meaningful in the itemsets and rules. The discretization results are shown in Figure 4.1

	genre	duration_msBin	popularityBin	loudnessBin	tempoBin	modeBin	ExplicitBin	danceabilityBin	energyBin	speechinessBin	acousticnessBin	instrumentalnessBin	livenessBin	valenceBin
0	j-dance	MLen	HiPop	LoLoud	Fast	Major	PG	HiDan	LowE	HiSp	MAc	HiInst	HiLive	HiVal
1	iranian	Long	LoPop	LoLoud	Slow	Major	PG	NoDan	LowE	HiSp	MAc	HiInst	HiLive	LowVal
2	black-metal	Long	LoPop	MLoud	MTempo	Major	PG	NoDan	HighE	LoSp	NotAc	HiInst	HiLive	MVal
3	bluegrass	Long	MPop	HiLoud	Fast	Major	PG	MDan	MidE	HiSp	MAc	MInst	MLive	MVal
4	happy	Long	MPop	LoLoud	Fast	Major	PG	MDan	MidE	MSp	NotAc	MInst	HiLive	HiVal
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
14995	idm	Long	LoPop	LoLoud	MTempo	Major	PG	MDan	MidE	MSp	NotAc	HiInst	MLive	LowVal
14996	sleep	Short	HiPop	LoLoud	Slow	Major	PG	NoDan	HighE	MSp	HiAc	HiInst	MLive	LowVal
14997	techno	Long	HiPop	MLoud	MTempo	Major	PG	HiDan	LowE	MSp	NotAc	HiInst	Studio	LowVal
14998	breakbeat	MLen	HiPop	HiLoud	Fast	Major	PG	MDan	HighE	MSp	NotAc	MInst	HiLive	MVal
14999	indian	MLen	HiPop	MLoud	Slow	Major	PG	HiDan	LowE	LoSp	HiAc	LoInst	Studio	HiVal

15000 rows × 14 columns

Figure 4.1: Discretization result for apriori algorithm pattern mining

In the frequent item set initial tests, we found that many interesting rules with high support include high acousticness, low energy, low loudness and vice versa. Table 4.1 shows frequent item sets with top 28 support value, the item sets that semantically make sense based on intuition or musical knowledge are highlighted in colors. For example, high energy is associated with high loudness; low loudness with low energy; high acousticness with low energy and low loudness; high danceability with high valence etc.

Table 4.1: Excerpt from Frequent Item sets, supp = 10% , zmin = 5 which generates total 206 itemsets (before removal of time_signature column)

frequent_itemset					supp
HighE	**HighLoud**	**TSig3**	**PG**	**Major**	**18,9**
LowLoud	**LowE**	**TSig3**	**PG**	**Major**	**18,42**
HighAcous	**LowE**	**TSig3**	**PG**	**Major**	**17,84**
HighAcous	**LowLoud**	**LowE**	**PG**	**Major**	**17,646667**
HighE	**NotAcous**	**TSig3**	**PG**	**Major**	**17,16**
HighDance	**HighVal**	**TSig3**	**PG**	**Major**	**16,08**
HighDance	MedTempo	TSig3	PG	Major	14,913333
HighAcous	LowLoud	TSig3	PG	Major	14,826667
HighInstr	LowPop	TSig3	PG	Major	14,586667
HighInstr	LowLoud	TSig3	PG	Major	14,313333
NonDance	LowVal	TSig3	PG	Major	14,286667
MedLoud	MedE	TSig3	PG	Major	14,18
HighPop	LowInstr	TSig3	PG	Major	14,153333
NotAcous	Long	TSig3	PG	Major	13,966667
HighLoud	NotAcous	TSig3	PG	Major	13,546667
HighInstr	LowVal	TSig3	PG	Major	13,513333
HighDance	MedE	TSig3	PG	Major	13,406667
HighAcous	LowLoud	LowE	TSig3	Major	13,406667
HighVal	MedAcous	TSig3	PG	Major	13,38
HighVal	MedE	TSig3	PG	Major	13,36
HighE	HighLive	TSig3	PG	Major	13,32
LowE	LoSpeech	TSig3	PG	Major	13,306667
HighDance	Studio	TSig3	PG	Major	13,293333
HighAcous	LowLoud	LowE	TSig3	PG	13,26
HighAcous	LowLoud	LowE	TSig3	PG	13,26
HighAcous	LoSpeech	TSig3	PG	Major	13,226667
LowInstr	MedAcous	TSig3	PG	Major	13,193333
HighVal	LowInstr	TSig3	PG	Major	13,18

In the histogram (Figure 4.2) showing the support value for each frequent itemset, we can see a rule with particularly high support: High acousticness, Low loudness, Low energy, not Explicit, and Major mode. However, the rest of the item sets largely concern high acoustic (non-electronic sounding) and quieter tracks, suggesting more calm or relaxing musical qualities tend to group together (high acoustic, slow, low loudness, low energy, low valence, no danceability). Two itemsets out of twenty four suggests the opposite qualities group together: high energy, high loudness, fast tempo and not acoustic.

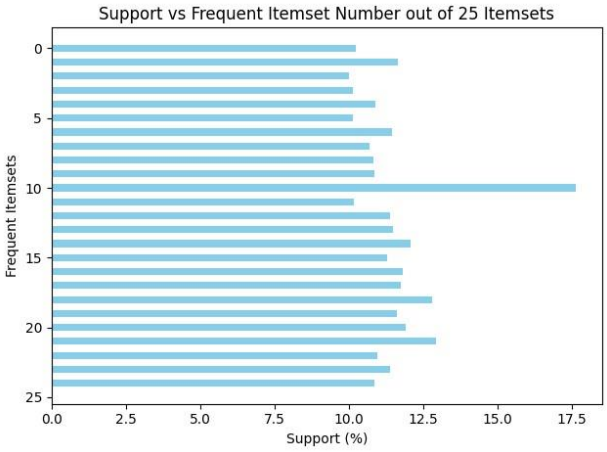


Figure 4.2 (above) Frequent itemset supp = 10% zmin = 5 (25 items) after removal of time_signaturre

Table 4.2 (right) Frequent itemsets zmin = 5 (25 items)

	frequent_itemset	support
0	(MLive, HiAc, LowE, PG, Major)	10.226667
1	(HighE, HiLoud, NotAc, PG, Major)	11.660000
2	(HighE, HiLoud, Fast, PG, Major)	10.013333
3	(HiAc, Short, LoLoud, PG, Major)	10.153333
4	(HiAc, Short, LowE, PG, Major)	10.880000
5	(HiAc, Slow, LoLoud, PG, Major)	10.153333
6	(HiAc, Slow, LowE, PG, Major)	11.446667
7	(HiAc, LoLoud, HiInst, PG, Major)	10.700000
8	(HiAc, LoLoud, NoDan, PG, Major)	10.820000
9	(HiAc, LoLoud, LowVal, PG, Major)	10.866667
10	(HiAc, LoLoud, LowE, PG, Major)	17.646667
11	(HiAc, HiInst, LowE, PG, Major)	10.180000
12	(HiAc, NoDan, LowE, PG, Major)	11.380000
13	(HiAc, LowVal, LowE, PG, Major)	11.480000
14	(HiAc, LowE, LoSp, PG, Major)	12.073333
15	(Short, LoLoud, LowE, PG, Major)	11.293333
16	(Slow, LoLoud, LowE, PG, Major)	11.820000
17	(LoLoud, HiInst, LowVal, PG, Major)	11.746667
18	(LoLoud, HiInst, LowE, PG, Major)	12.800000
19	(LoLoud, NoDan, LowVal, PG, Major)	11.620000
20	(LoLoud, NoDan, LowE, PG, Major)	11.913333
21	(LoLoud, LowVal, LowE, PG, Major)	12.933333
22	(LoLoud, LowE, LoSp, PG, Major)	10.973333
23	(HiInst, NoDan, LowVal, PG, Major)	11.393333
24	(NoDan, LowVal, LowE, PG, Major)	10.873333

4.1 Frequent Pattern Extraction: Closed vs Maximal itemsets

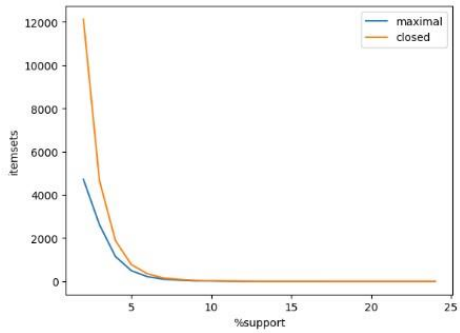


Figure 4.3: Plot of number of item sets based on support percentage at zmin = 5

A closed itemset is an itemset where none of its immediate supersets have the same frequency as the itemset itself. In other words, an itemset is closed if there is no larger itemset with the same support count. A maximal itemset is an itemset that is frequent and has no frequent supersets. In other words, an itemset is maximal if there is no larger itemset that is also frequent. Closed itemsets are useful for generating concise representations of frequent itemsets without losing any information about the frequency of combinations. Maximal itemsets provide a more compact representation by including only the largest frequent itemsets. They reduce the number of itemsets that need to be considered, which can be advantageous in terms of computational efficiency. We find that as we increase support, our dataset (train.csv) generates converging number of closed and maximal itemsets. (see Figure4.3). Table 4.3 also shows that the number of frequent, closed, and maximal item sets tend to be closer as the constraints of zmin and min support become more demanding (larger zmin and larger min support). Here we also begin looking for itemsets that include genre as one of the items and we find that for a given support and zmin, the same set genres appear across frequent, closed, and maximal itemsets. Next, we study the relationship between %support and itemsets by genre for a given zmin (Figure 4.4-4.6). We find that itemsets capture sleep and iranian genre more easily than other genres (at zmin = 10; Figure 4.6), and as we reduce zmin and support we can capture more itemsets to include other genres. However these Figures start at support = 1. We find that at zmin = 10, if we reduce support to 0.5, the itemsets also include black-metal, Chicago-house, forro, study, techno, in addition to the sleep and Iranian genre.

Table 4.3: Comparison of number of Frequent, Closed, Maximal itemsets given support and zmin values. The same set of genres are observed across the three types of itemsets for a given pair of zmin and support.

Number of Apriori itemsets	Support 10%; Zmin = 5	Support 5%; Zmin = 5	Support 1%; Zmin = 5	Support 1%; Zmin = 10	Support 0.5%; Zmin = 10
Frequent Itemsets	24	898	90973	65	1051
Closed Itemsets	24	771	53041	64	772

Maximal Itemsets	24	493	15218	53	226
Genres appearing in itemset	None	None	all	sleep, iranian	'black-metal', 'chicago-house', 'forro', 'iranian', 'sleep', 'study', 'techno'

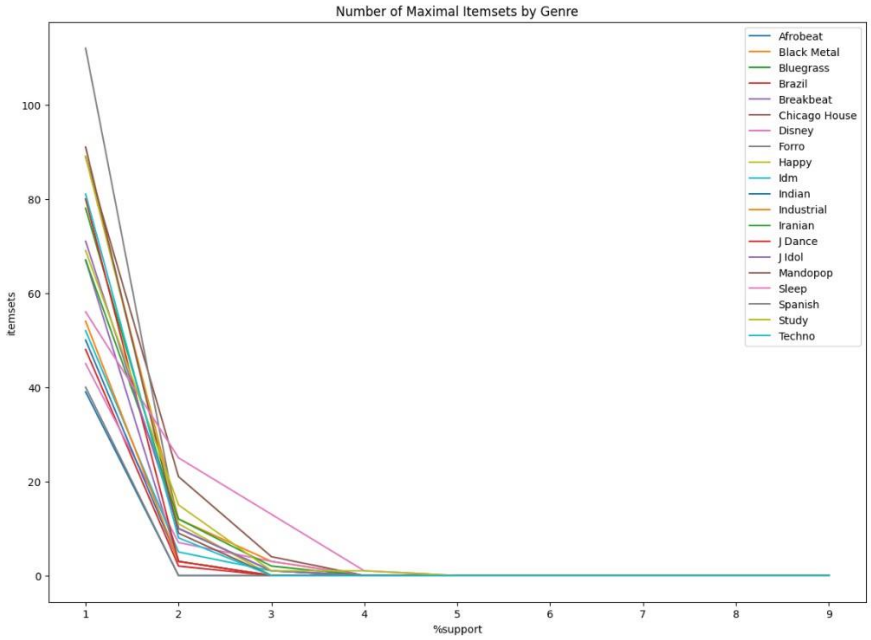


Figure 4.4 Number of maximal itemsets vs. %support for each genre (zmin = 5)

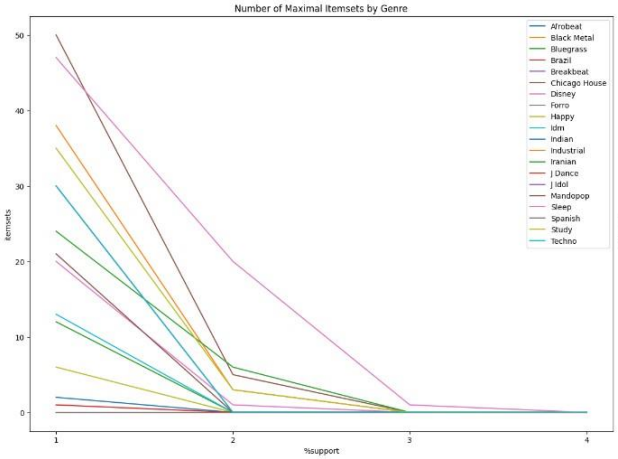


Figure 4.5 Number of maximal itemsets vs. %support for each genre (zmin = 7)

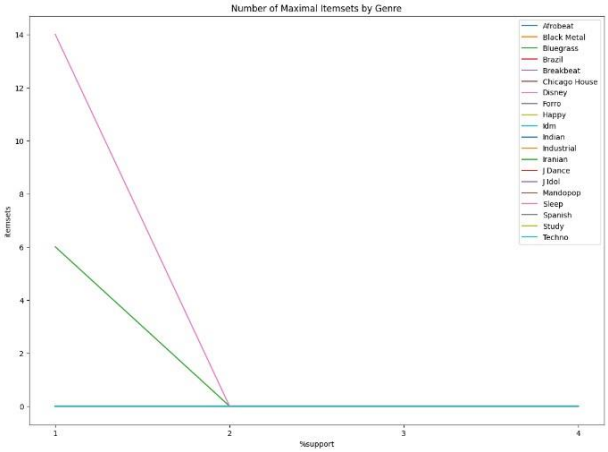


Figure 4.6 Number of maximal itemsets vs. %support for each genre (zmin = 10). Sleep- pink; Iranian- green.

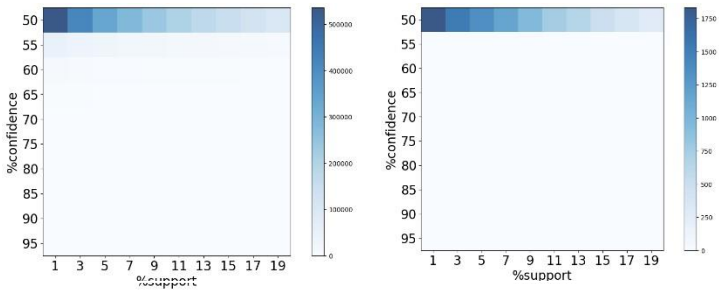


Figure 4.7-4.8: Heat Map of Number of Rules for percentage of Confidence and Support (left image zmin=5; right image zmin =10)

4.2 Association Rules Extraction

To use apriori algorithm to extract association rules, we need to identify accepTable confidence value. We find that confidence cutoff above 55% will give us too little number of rules. (Figure 4.7-4.8)

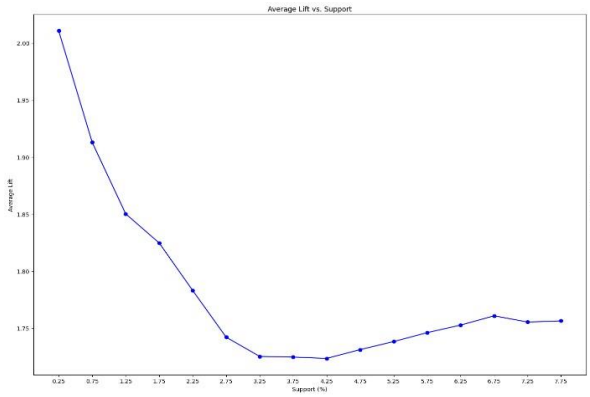


Figure 4.9 Average Lift of Rules vs. % Support, with fixed z min = 5

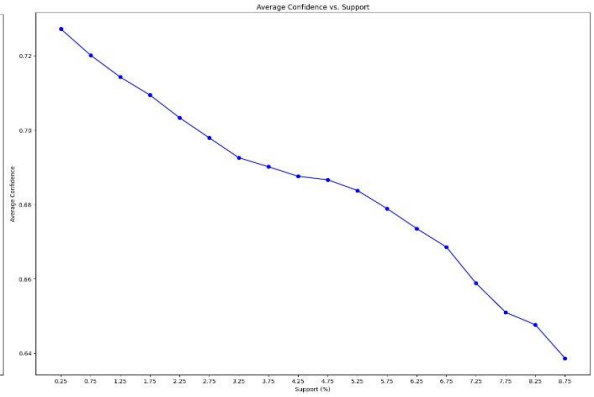


Figure 4.10 Average Confidence of Rules vs. % Support, with fixed z min = 5

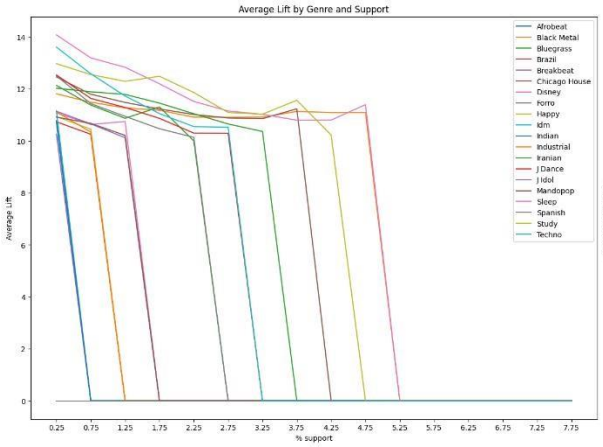


Figure 4.11 Average Lift of Rules vs. % Support by Genre, with fixed z min = 5 (Lift in plot: 0-14)

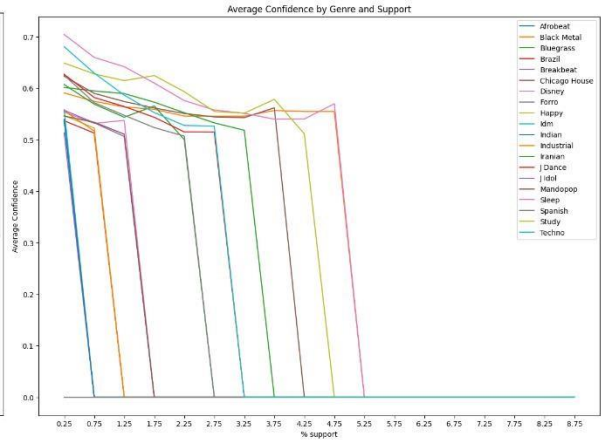


Figure 4.12 Average Confidence of Rules vs. % Support by Genre, with fixed z min = 5 (Confidence in plot: 0-70%)

We observe lift (Figure 4.9) and confidence (Figure 4.10) vs. %support (assuming zmin = 5) for our train.csv dataset. The average lift per rule sharply drop as it approaches support = 3, meanwhile, the average confidence decreases almost linearly as support. This means that the lower the support, the higher average lift and confidence. However, if we plot average lift and average confidence against %support considering only the rules that have a genre as the "consequent", by each genre, we find that each genre has a range of performing support where lift and confidence are non-zero, and that in its own range, each genre's the average lift is much higher (above 10). Moreover, and interestingly, average confidence vs. %support and lift vs. %support by genre have the same shapes of plot, which suggests possible proportional relationship considering formal definitions of lift and confidence. When we inspect the Table (Table 4.4) of association rules we find that rules with genre as consequent has much higher lift (11-14) compared to non-genre characteristic of the track (only 1-3).

Table. 4.4 Analysis of number of rules, average support, lift, and confidence for each genre (zmin = 10, minimum support 0.5%).

Genre	rules	total abs support	Avg abs support per rule	Total lift	Avg lift per rule	Avg conf per rule
sleep	340	29478	86,7	5802,26	17,0654798	0,853274
techno	13	891	68,538462	208,87	16,06676583	0,803338
study	14	1007	71,928571	224,78	16,05481331	0,802741
forro	1	64	64	15,42	15,42168675	0,771084
chicago-house	78	5464	70,051282	1196,82	15,34387703	0,767194
black-metal	10	706	70,6	149,08	14,90822275	0,745411
iranian	122	9744	79,868852	1761,31	14,4370226	0,721851

Table 4.5 Top three association rules (sorted by lift) for each consequent, showing only a sample of genreconsequents and 1 non-genre-consequent (studio = low liveness), at zmin = 9, minimum support 0.5%; we see that rules with genre consequent has much higher lift and confidence than rules with non-genre consequent.

consequent	antecedent	abs_support	%_support	confidence	lift
techno	('HiPop', 'HiDan', 'HiInst', 'MTempo', 'Long', 'NotAc', 'LowVal', 'PG', 'Major')	75	0,5	0,949367	18,98734
techno	('HiPop', 'MSp', 'HiDan', 'HiInst', 'MTempo', 'Long', 'NotAc', 'PG', 'Major')	71	0,473333	0,8875	17,75
techno	('HiPop', 'MSp', 'HiDan', 'HiInst', 'MTempo', 'Long', 'Studio', 'PG', 'Major')	67	0,446667	0,87013	17,4026
techno	...				
study	('HiDan', 'MVal', 'HiSp', 'Short', 'LoLoud', 'HiInst', 'LowE', 'PG', 'Major')	69	0,46	0,8625	17,25
study	('HiDan', 'MVal', 'Short', 'LoLoud', 'HiInst', 'Slow', 'LowE', 'PG', 'Major')	74	0,493333	0,860465	17,2093
study	('MLive', 'HiDan', 'Short', 'LoLoud', 'HiInst', 'Slow', 'LowE', 'PG', 'Major')	66	0,44	0,857143	17,14286
study	...				
Studio	('HiPop', 'MSp', 'HiDan', 'HiInst', 'MTempo', 'Long', 'NotAc', 'PG', 'Major')	63	0,42	0,7875	2,323009
Studio	('chicago-house', 'MSp', 'HiDan', 'HiVal', 'HiInst', 'MTempo', 'Long', 'PG', 'Major')	61	0,406667	0,782051	2,306936
Studio	('chicago-house', 'MSp', 'HiDan', 'HiVal', 'HiInst', 'Long', 'LoPop', 'PG', 'Major')	59	0,393333	0,776316	2,290017
Studio	...				
iranian	('HiInst', 'Slow', 'Long', 'NoDan', 'LowE', 'LowVal', 'LoPop', 'PG', 'Major')	118	0,786667	0,746835	14,93671
iranian	('LoLoud', 'HiInst', 'Slow', 'Long', 'NoDan', 'LowE', 'LowVal', 'LoPop', 'PG')	116	0,773333	0,74359	14,87179
iranian	('LoLoud', 'HiInst', 'Slow', 'Long', 'NoDan', 'LowE', 'LowVal', 'LoPop', 'Major')	116	0,773333	0,74359	14,87179
iranian	...				
chicago-house	('HiDan', 'HiVal', 'LoLoud', 'MTempo', 'Long', 'NotAc', 'LoPop', 'PG', 'Major')	68	0,453333	0,871795	17,4359
chicago-house	('MSp', 'HiDan', 'HiVal', 'MTempo', 'Long', 'NotAc', 'Studio', 'LoPop', 'Major')	68	0,453333	0,871795	17,4359
chicago-house	('MSp', 'HiDan', 'HiVal', 'MTempo', 'Long', 'NotAc', 'Studio', 'LoPop', 'PG')	67	0,446667	0,87013	17,4026
chicago-house	...				
black-metal	('HighE', 'HiSp', 'MLoud', 'Long', 'NotAc', 'NoDan', 'LowVal', 'PG', 'Major')	66	0,44	0,804878	16,09756
black-metal	('HighE', 'HiInst', 'MLoud', 'Long', 'NotAc', 'NoDan', 'LowVal', 'PG', 'Major')	77	0,513333	0,777778	15,55556
black-metal	('MPop', 'HighE', 'HiSp', 'HiInst', 'NotAc', 'NoDan', 'LowVal', 'PG', 'Major')	66	0,44	0,776471	15,52941
black-metal	...				

4.3 Target Variable Prediction

To use association rule to make target prediction, we need to simplify the model, so we choose zmin = 10 and minimum support 1, which means we will only have 'sleep' and 'iranian' as genre-consequents, and fewer number of rules to match up with the test set's data. To simplify the task for implementation, we will label our data with 'sleep' and 'not_sleep', conducting single-class classification (or 2-class, 1 vs. 0) at a time. We extract rules from the training dataset (receiving 27 rules for sleep consequent), and we try to capture the calssification performance of the training set and the test set.

First, we check that any identical antecedents will not lead to multiple genres. We find that if we only analyze the rules with genre-consequents, there is indeed no confusion of possible consequent genres within the same dataset and across training and test set (please note, we checked in many rounds of association rules zmins and support values up to about half the genres being captured but not all due to computational time). However, if we leave in all of the rules, one antecedent could lead to a genre consequent and up to 2 additional itemized characteristics as consequents. However, generally genre-consequent rules have higher lift so we keep this in mind for implementation. We suspect that accuracy even in the training set will not be perfect because of these multiple possible consequents.

After we create the new labels (sleep vs. not_sleep), we get the association rules from the training set, and convert the antecedents into an itemized array, ranked by lift value. Then we turn the data points in the testing set into itemized arrays as well, and we search through the array of training-set antecedents and take the consequent of the first antecedent that is a subset of the testing set's itemized array for which we want to predict the label. We actually did the same for the training set to see if there is over-fitting or not.

Table 4.6 Classification Performance on training set

	precision	recall	f1-score	support
not_sleep	0.99	0.97	0.98	14531
sleep	0.42	0.67	0.52	469
accuracy			0.96	15000
macro avg	0.70	0.82	0.75	15000
weighted avg	0.97	0.96	0.96	15000

Table 4.7 Classification Performance on training set

	precision	recall	f1-score	support
not_sleep	0.99	0.96	0.98	4901
sleep	0.29	0.73	0.41	99
accuracy			0.96	5000
macro avg	0.64	0.85	0.70	5000
weighted avg	0.98	0.96	0.97	5000

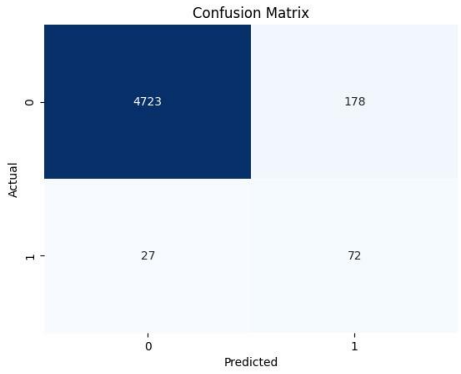


Figure: 4.13 Confusion Matrix to predict 'sleep' on test set by association rules from training set

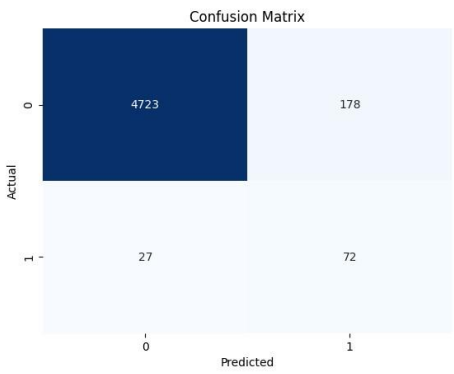


Figure: 4.14 Confusion Matrix to predict 'sleep' on training set using association rules extracted from training set

Using association rules extracted from training set, we use them to predict the sleep or not_sleep label. We achieve high accuracy score (0.96) because in our sleep or not_sleep classifcation there are a lot of negative labels (not_sleep), which the association rules are able to capture well. Though our precision of sleep label is not so high (0.42 in training set and 0.29 in test set), our recall and f-1 reflect that the performance is not poor considering imbalanced labels. Overall, association rules allow for better performance report than a random classifier that may guess 7500 sleep tracks and 7500 not_sleep tracks in training set and 2500 sleep tracks and 2500 not_sleep tracks in the test set. For example, even if the random classifier guesses correctly all true positives, its precision will still equate 0.1% because of a lot more false negative data points.

In more detailed view, the tracks that were predicted as "sleep" but were something else tend to be Indian and Disney. The extracted rules from training set have the following confusion count: Confusion when predicting label in training dataset: {'bluegrass': 3, 'afrobeat': 7, 'study': 30, 'forro': 4, 'indian': 23, 'brazil': 18, 'j-idol': 2, 'mandopop': 12, 'spanish': 5, 'disney': 31, 'idm': 10, 'techno': 5, 'industrial': 2, 'j-dance': 1, 'black-metal': 2}; when predicting label in testing dataset: {'mandopop': 2, 'indian': 6, 'idm': 2, 'spanish': 1, 'disney': 8, 'black-metal': 1, 'study': 4, 'brazil': 1, 'techno': 1, 'forro': 1}.

4.4 Conclusion of Pattern Mining

The relationships between itemsets and support by genre tell us that the higher z-min, the more difficult to find maximal itemsets with higher support. We know that the genres that have itemsets at higher zmin (more precise and elaborate itemset) as high as 10 are sleep and Iranian. From here we can lower the support to try to include more genres, or we can try lowering zmin. We found that lift in association rules that include 'genre' as consequent is higher than the rules that have some other characteristics as consequent, which suggests that genre-based rules have higher strength of relationships between items compared to what would be expected by chance, promising capacity of pattern recognition. Testing association rules' antecedents extracted from training set to predict genre the strongest genre that is "sleep" vs. not_sleep, we find that though precision is not particularly strong (0.29 in test set data, 0.42 in train set data), our recall is not too bad (0.73 in test set, 0.63 in train set). This gives the overall model extracted from association rules 0.96 accuracy when applied to both the training and testing sets. In future studies, we would like to try reducing the zmin and support to see the classification performance, and also compare the results for different genres and also do a 20-class classification with the rules when we Figure out how to implement it without computational inefficiency.

4.5. Regression

Based on insights from the previous section, we chose two continuous columns, namely "acousticness" and "loudness" against the target column of "energy" and apply different regressions to explore their relationship. In Table 4.6, we show the performance evaluations of different regression models, namely Mean Square Error, Mean Absolute Error, and R-Square.

Table 4.6 Regression performance result

Model Name	Mean Square Error (MSE)	Mean Absolute Error (MAE)	R-Square((R^2)
LINEAR REGRESSION			
Ridge	0.120	0.025	0.645
Lasso	0.061	0.208	0.129

NON-LINEAR REGRESSION			
Decision Tree (DT)	0.041	0.144	0.408
KNN Regressor	0.025	0.114	0.646
MULTIVARIANT LINEAR REG			
Linear Regression	0.025	0.199	0.647
Non-Linear: Decision Tree Regressor	0.038	0.137	0.458
2 Target Variables (Energy and Valence): Decision Tree Regressor	0.015	0.068	0.744

In Ridge Linear Regression, the model has a moderate R^2, indicating it explains around 64.5% of the variance. The low MAE suggests that the predictions are close to the actual values. In Lasso Linear Regression, the lower R^2 indicates that this model explains only 12.9% of the variance, which is relatively poor. The higher MAE suggests less accurate predictions. For Non-Linear Regression, in Decision Tree, the model explains 40.8% of the variance, with a moderate MAE. It performs better than Lasso in terms of R^2 but worse than Ridge. Meanwhile, KNN Regression has a high R^2 (64.6%), indicating good performance. It also has a low MSE and MAE, suggesting accurate predictions.

In Multivariate Regression, the linear model has a high R^2 (64.7%), similar to Ridge and KNN, indicating good performance. However, the MAE is slightly higher, suggesting less precise predictions. The non-linear model explains 45.8% of the variance, which is moderate. It performs better than the Decision Tree but not as well as the KNN or Ridge in terms of R^2. Finally, for 2+ target variables using Decision Tree regressor, the model has the highest R^2 (74.4%), indicating it explains the most variance among all the models. The lowest MSE and MAE suggest it provides the most accurate predictions. In summary, the "2+ Target Variables" Decision Tree regressor model is the best performer, with the highest R^2 (0.744), the lowest MSE (0.015), and the lowest MAE (0.068). The other good performers are The KNN Regressor and Linear Regression (Ridge) models, with high R^2 values (0.646 and 0.645, respectively) and low errors. The least effective model is the Lasso regression model, with the lowest R^2 (0.129) and relatively high MAE (0.208).

4.6. Conclusion of Regression

In our dataset, Linear regression performs reasonably well with moderate errors and a decent R-squared value. Ridge regression, with lower MSE, higher MAE, and a lower R-squared, indicates less predictive power. Lasso regression has lower MSE and MAE with a good R-squared value. For non-linear regression, the Decision Tree (DT) performs similarly to Lasso regression but is a better fit for the data compared to Ridge regression. The KNN regressor, with low errors and a high R-squared value, indicates good predictive performance. Multivariate linear regression performs similarly to the KNN regressor but with a slightly higher MAE. In contrast, the nonlinear regression model has a lower R-squared compared to both linear regression and the KNN regressor. Multivariate linear regression with additional target variables has the lowest MSE and MAE and the highest R-squared compared to all other models, indicating the most superior predictive power.

Chapter 5: OVERALL CONCLUSION

The spotify dataset has many attributes whose relationships are unclear and we eliminate several of them in real data mining tasks. In the beginning we removed redundant or highly correlated variables and questionable variables such as time signature or processing. Later we discover through various algorithms variables that do not impact the data mining tasks such as through feature importance in decision tree or discretized values that appears in almost all apriori itemsets such as time_signature, but this depends on the algorithms. While log transformation helps improve distribution for some variables, not all data mining tasks require normal or unskewed distributions so we do not always pre-process our data the same way for all datamining tasks. In clustering analysis, our datapoints do not have shapes or clear density differences to be captured well with clustering algorithms, especially in higher dimensions. Hierarchical Clustering with agglomerative clustering promise interpretability of the clusters with competitive silhouette score with DBSCAN. In classification tasks, KNN and Decision Trees perform quite well on the 20-genre label prediction tasks with around 50% accuracy and precision-recall. Using multiple classification tools and studying their confusion matrices give more insight about relationships between genres than studying confusion matrix from one best model. In regression analysis, multivariate linear regression through decision tree regressor emerged as the most predictive model. Pattern Mining reveals the strength of relationships among musical characteristics with genre labels, as genre-based association rules have higher lift than rules without genre consequents as we approach higher number of required minimum itemsets. Despite not being able to fit perfectly to the dataset from which they are extracted from, association rules can be effectively used for classification task to unseen data, comparable to decision tree, while promising simpler interpretability in higher dimension analysis.