Università di Pisa

Department of INFORMATICA

Master of Data Science and Business Informatics

PROJECT REPORT

SPOTIFY TRACKS DATASET

Submitted by:

Alessandro Spinello (550351)

Enrico Cilia (563234)

Academic Year 2023/2024

TABLE OF CONTENTS

CHAPTER 1

DATA UNDERSTANDING AND PREPARATION

The Spotify Tracks dataset contains data on audio tracks accessible through the Spotify catalog. These tracks span 20 distinct genres. Each track has key attributes: track name, artist, album name, and popularity within the catalog. In addition, audioderived characteristics are included, including aspects such as danceability, energy, pitch, and volume.

1.1 DATA SEMANTICS

The Spotify dataset is a dataset consisting of 15000 music tracks and 24 attributes that possess information on which further analysis can be done. Table 1.1 lists for each attribute the name, description and type associated with it.

TABLE 1.1: Description of dataset attributes

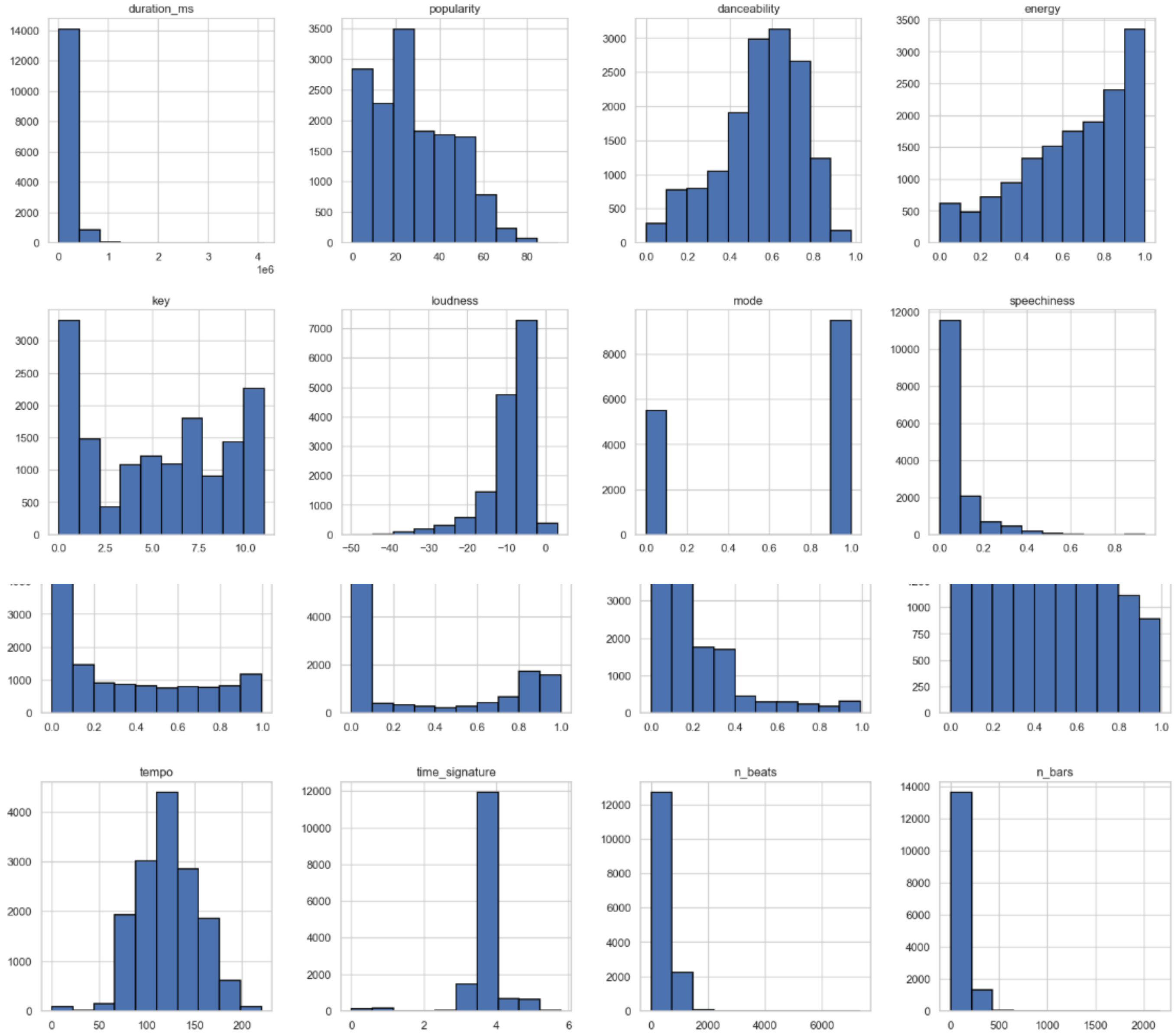
NAME	TYPE	DESCRIPTION
Name	Categorical	Name of the piece of music
Duration_ms	Numeric (integer)	Track length in milliseconds
Explicit	Boolean	The attribute suggests whether or not the song has explicit text. (true = yes, false = no, or unknown)
Popularity	Numeric (integer)	The attribute indicates in numerical terms the degree of popularity. It is a value between 0 and 100, where 100 is the most popular.
Artists	Categorical	The names of the artists who performed the song. If there is more than one artist, they are separated by a ;
Album_name	Categorical	The name of the album in which the song appears.
Danceability	Numeric (float)	Danceability describes how danceable a song is. A value of 0.0 is the least danceable and 1.0 is the most danceable
Energy	Numeric (float)	The attribute represents a perceptual measure ranging from 0.0 to 1.0 of intensity and activity
key	Numeric (integer)	The pitch in which the song is located. Integers correspond to pitches using standard Pitch Class notation.
Loudness	Numeric (float)	The overall volume of a song in decibels (dB).
Mode	Binary	This attribute indicates the major or minor mode of a song. Major is represented by 1 and minor by 0
Speechiness	Numeric (float)	Vocality detects the presence of spoken words in a song.
Acousticness	Numeric (float)	The attribute indicates a confidence measure expressing whether the song is acoustic. 1.0 → represents high confidence that the song is acoustic
Instrumentalness	Numeric (float)	It predicts whether a song contains no voices. The closer the instrumentality value is to 1.0, the higher the instrumentality value is.
Liveness	Numeric (float)	Detects the presence of an audience in the recording.
Valence	Numeric (float)	Valence describes the musical positivity conveyed by a song. A measure ranging from 0.0 to 1.0.

Tempo	Numeric (float)	The attribute indicates the estimated total time of a track in beats per minute (BPM).
Feature_duration_ms	Numeric (integer)	The duration of the song in milliseconds
Time_signature	Numeric (integer)	The attribute indicates an estimated time signature
N_beats	Numeric (integer)	The total number of beat intervals in the track.
N_bars	Numeric (integer)	The total number of beat intervals in the whole song.
Popularity_confidence	Numeric (float)	The attribute expresses the confidence from 0.0 to 1.0 of the popularity of the song
Genre	Categorical	The attribute indicates the genre to which the song belongs
Processing	Numeric	No information

Regarding the processing attribute, we decided to remove it at the beginning of our analysis because we had no relevant information to deal with it.

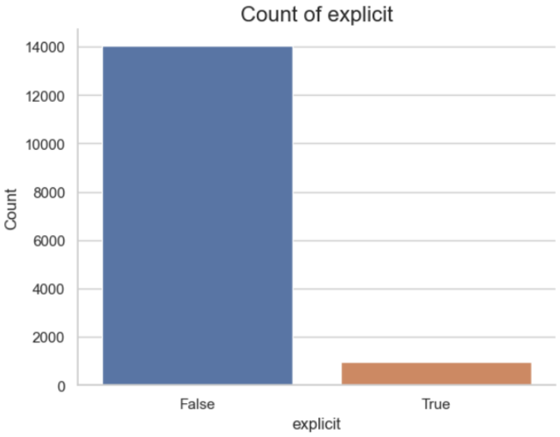
1.2 DISTRIBUTION OF THE VARIABLES AND STATISTICS

The data we find in the table above can be depicted through the use of certain graphs. Let us start by visualizing the distribution of numerical data using histograms:



These histograms show the frequency distribution of the numerical attributes of the dataset, where the attribute intervals are discretized into a fixed number of intervals (BINS). For each interval, the (absolute) frequency of the values within it is indicated by the height of the single bar.

Next, we plot the categorical variables using Bar Charts:

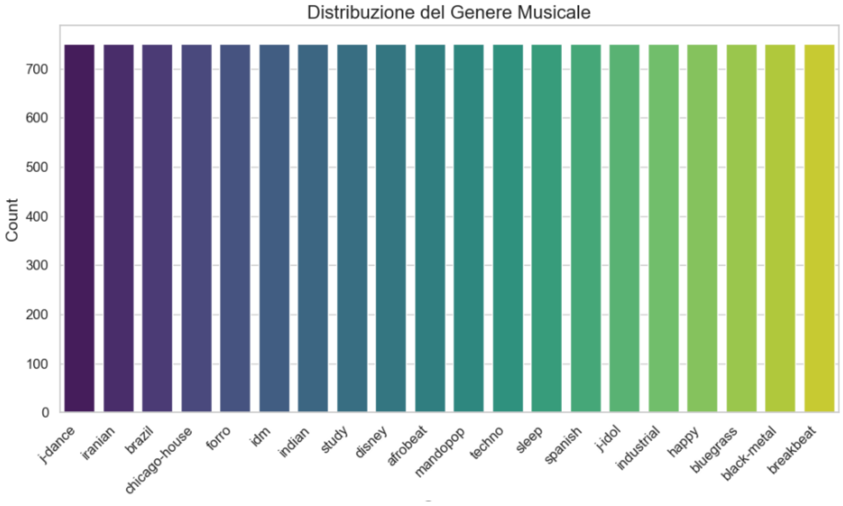


In the graph on the left, we can see the frequency distribution of the variable explicit. The attribute presents:

- 14034 occurrences False

- 966 occurrences True

In the graph on the right, we depict the distribution of the music genre. The attribute presents us with 12 musical genres that have 750 occurrences ciascuno



Bar Charts are better suited to depicting the frequencies of categorical attribute values because they are data measured on a scale with specific potential values.

1.3 EVALUATING DATA QUALITY

This section deals with understanding some general information concerning the data (missing values, outliers, correlation). In the next subsections we are going to implement the Data Preparation phase in which we try to solve problems related to the data we have. The goal is to improve the quality of the data by also going to reduce the size of the dataset, if necessary, and choosing the attributes we are most interested in.

1.3.1 MISSING VALUES

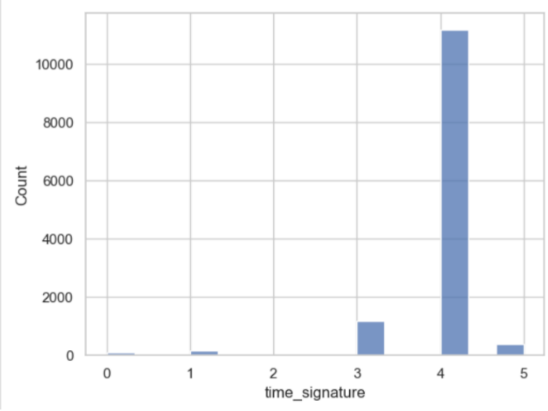
After careful analysis of the dataset, we realized that missing values are present distributed as follows:

- Mode: 4450
- Time_signature: 2062
- Popularity_confidence: 12783

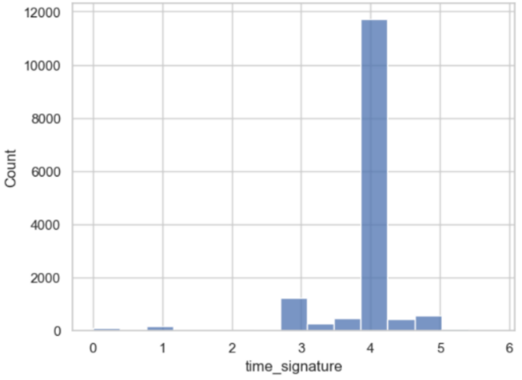
Regarding the Mode attribute, we know that it has two modes: 1 (Major) which has 6661 observations and 0 (Minor) which has 3889 observations. We replaced 63% of the missing values with 1 and the remaining 37% with 0.

With reference to the Time_signature attribute, we have replaced the missing values with the median

BEFORE REPLACING MISSING VALUES



AFTER REPLACING MISSING VALUES



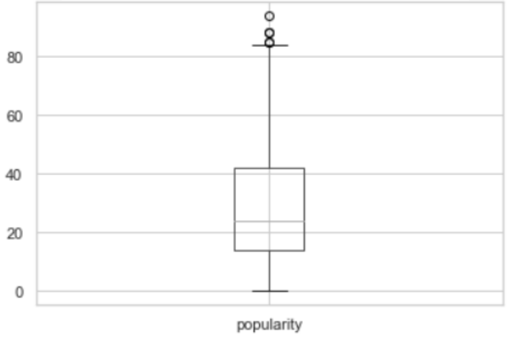
The Popularity_confidence attribute, on the other hand, has very many missing values i.e., 85 % of the total values, and this does not allow us to replace the missing values truthfully.

1.3.2 OUTLIERS

An Outlier is a value that belongs to a certain attribute but is very different from all other values. In most cases, the removal of Outliers is strongly recommended. In fact, ignoring these values could have very serious repercussions on our models and the performance of our results. Identifying Outliers is one of the fundamental pre-processing procedures. The identification of such values is called OUTLIER DETECTION.

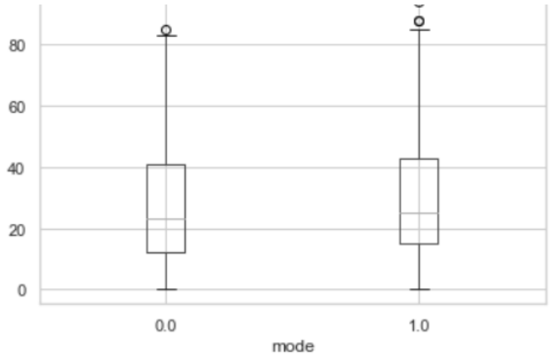
There is no precise and rigorous mathematical definition of Outliers but there are empirical ways to be able to understand how to identify them. Also visualizations can help to identify them, such as: boxplot, scatterplot

BOXPLOT POPULARITY



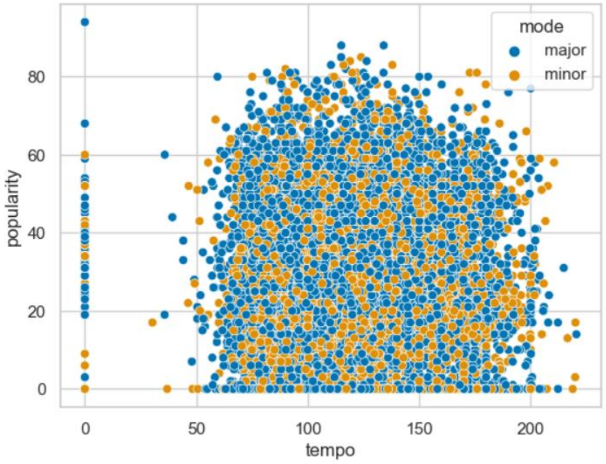
As far as popularity is concerned, the maximum observable value is 94. From the graph, one can see the presence of this outlier. In order to better understand what it is, we did further analysis to get detailed information on this point

BOXPLOT POPULARITY GROUPED BY MODE



Grouping the popularity by mode, we note the presence of a few outliers with both the minor and major value of the mode. In particular, the most popular song (=94) in the dataset has mode=1

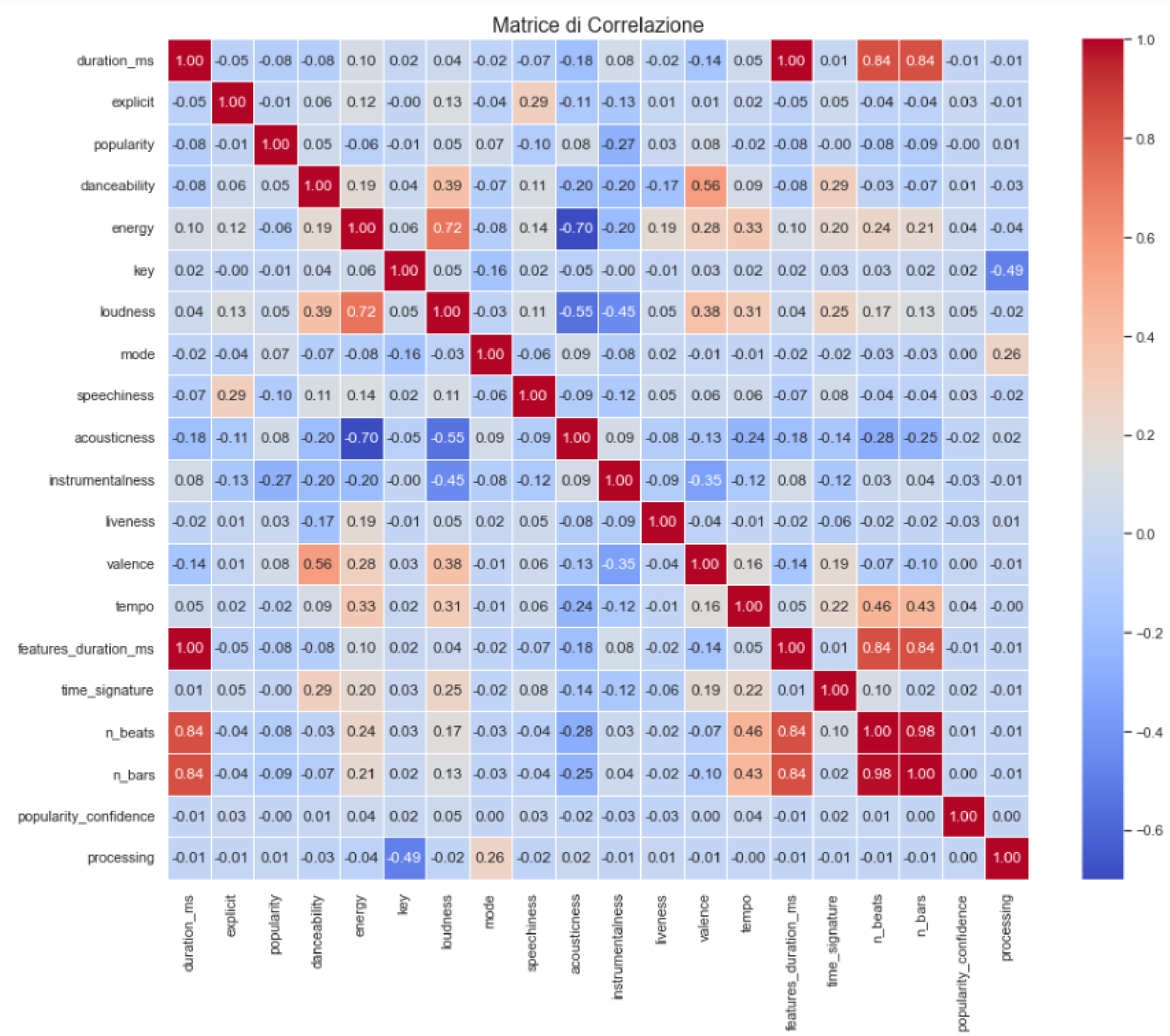
SCATTER PLOT POPULARITY AND TEMPO GROUPED BY MODE



In this scatterplot, it can be seen that the most popular song (Clean White Noise - Loopable with no fade) has tempo = 0. In fact, as one can see, these ‘songs’ with zero bpm (zero danceability, zero speechiness) are not real songs but noises or sounds

Usually in statistics the standard deviation is used as a criterion to remove outlier. A value that deviates three times the standard deviation from the mean usually is considered an Outlier. This is the method we used in our analysis: Taking only quantitative variables into account, we translate all observations that have values that are beyond the mean ± 3 times the standard deviation and replace them with the mean ± 3 times the standard deviation.

1.3.3 PAIRWISE CORRELATIONS AND ELIMINATION OF REDUNDANT VARIABLES



	duration_ms	explicit	popularity	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness	valence	tempo	features_duration_ms	time_signature	n_beats	n_bars	popularity_confidence	processing
duration_ms	1.00	-0.05	-0.08	-0.08	0.10	0.02	0.04	-0.02	-0.07	-0.18	0.08	-0.02	-0.14	0.05	1.00	0.01	0.84	0.84	-0.01	-0.01
explicit	-0.05	1.00	-0.01	0.06	0.12	-0.00	0.13	-0.04	0.29	-0.11	-0.13	0.01	0.01	0.02	-0.05	0.05	-0.04	-0.04	0.03	-0.01
popularity	-0.08	-0.01	1.00	0.05	-0.06	-0.01	0.05	0.07	-0.10	0.08	-0.27	0.03	0.08	-0.02	-0.08	-0.00	-0.08	-0.09	-0.00	0.01
danceability	-0.08	0.06	0.05	1.00	0.19	0.04	0.39	-0.07	0.11	-0.20	-0.20	-0.17	0.56	0.09	-0.08	0.29	-0.03	-0.07	0.01	-0.03
energy	0.10	0.12	-0.06	0.19	1.00	0.06	0.72	-0.08	0.14	-0.70	-0.20	0.19	0.28	0.33	0.10	0.20	0.24	0.21	0.04	-0.04
key	0.02	-0.00	-0.01	0.04	0.06	1.00	0.05	-0.16	0.02	-0.05	-0.00	-0.01	0.03	0.02	0.02	0.03	0.03	0.02	0.02	-0.49
loudness	0.04	0.13	0.05	0.39	0.72	0.05	1.00	-0.03	0.11	-0.55	-0.45	0.05	0.38	0.31	0.04	0.25	0.17	0.13	0.05	-0.02
mode	-0.02	-0.04	0.07	-0.07	-0.08	-0.16	-0.03	1.00	-0.06	0.09	-0.08	0.02	-0.01	-0.01	-0.02	-0.02	-0.03	-0.03	0.00	0.26
speechiness	-0.07	0.29	-0.10	0.11	0.14	0.02	0.11	-0.06	1.00	-0.09	-0.12	0.05	0.06	0.06	-0.07	0.08	-0.04	-0.04	0.03	-0.02
acousticness	-0.18	-0.11	0.08	-0.20	-0.70	-0.05	-0.55	0.09	-0.09	1.00	0.09	-0.08	-0.13	-0.24	-0.18	-0.14	-0.28	-0.25	-0.02	0.02
instrumentalness	0.08	-0.13	-0.27	-0.20	-0.20	-0.00	-0.45	-0.08	-0.12	0.09	1.00	-0.09	-0.35	-0.12	0.08	-0.12	0.03	0.04	-0.03	-0.01
liveness	-0.02	0.01	0.03	-0.17	0.19	-0.01	0.05	0.02	0.05	-0.08	-0.09	1.00	-0.04	-0.01	-0.02	-0.06	-0.02	-0.02	-0.03	0.01
valence	-0.14	0.01	0.08	0.56	0.28	0.03	0.38	-0.01	0.06	-0.13	-0.35	-0.04	1.00	0.16	-0.14	0.19	-0.07	-0.10	0.00	-0.01
tempo	0.05	0.02	-0.02	0.09	0.33	0.02	0.31	-0.01	0.06	-0.24	-0.12	-0.01	0.16	1.00	0.05	0.22	0.46	0.43	0.04	-0.00
features_duration_ms	1.00	-0.05	-0.08	-0.08	0.10	0.02	0.04	-0.02	-0.07	-0.18	0.08	-0.02	-0.14	0.05	1.00	0.01	0.84	0.84	-0.01	-0.01
time_signature	0.01	0.05	-0.00	0.29	0.20	0.03	0.25	-0.02	0.08	-0.14	-0.12	-0.06	0.19	0.22	0.01	1.00	0.10	0.02	0.02	-0.01
n_beats	0.84	-0.04	-0.08	-0.03	0.24	0.03	0.17	-0.03	-0.04	-0.28	0.03	-0.02	-0.07	0.46	0.84	0.10	1.00	0.98	0.01	-0.01
n_bars	0.84	-0.04	-0.09	-0.07	0.21	0.02	0.13	-0.03	-0.04	-0.25	0.04	-0.02	-0.10	0.43	0.84	0.02	0.98	1.00	0.00	-0.01
popularity_confidence	-0.01	0.03	-0.00	0.01	0.04	0.02	0.05	0.00	0.03	-0.02	-0.03	-0.03	0.00	0.04	-0.01	0.02	0.01	0.00	1.00	0.00
processing	-0.01	-0.01	0.01	-0.03	-0.04	-0.49	-0.02	0.26	-0.02	0.02	-0.01	0.01	-0.01	-0.00	-0.01	-0.01	-0.01	-0.01	0.00	1.00

The heat map (Figure 3) shows the correlation matrix, where the blue square indicates low correlation, while the red square indicates high correlation. Looking closely at the matrix we notice a higher correlation (1) between the variables DURATION_MS and FEATURES_DURATION_MS. Since these attributes are highly correlated, to simplify the model, the variable FEATURES_DURATION_MS was eliminated.

1.3.4 DATA REDUCTION

In this section we are going to define which attributes are unnecessary or superfluous for our data analysis. In section 1.3.1 we had noticed that the **Popularity_confidence** attribute had many missing values, and we were unable to replace them; this is a good reason to eliminate that variable from the model. Whereas, in section 1.3.4 we eliminated the variable FEATURES_DURATION_MS as it was highly correlated with DURATION_MS

Other variable to be eliminated are

- **EXPLICIT**→ l'attributo possiede 14034 occorrenze false e 966 occorrenze true.
- **N_BEATS** → N_bars and N_beats between them have 0.98 correlation and in turn have a correlation of 0.84 with duration_ms. We left n_bars.
- **NAME, ALBUM_NAME, ARTISTS**, → delete these attributes as they do not provide us with useful information for our analysis

CHAPTER 2

DATA CLUSTERING

The first basic step was to standardize numerical variables with zero mean and unit variance. In the following section, several methods are used to perform clustering. The algorithms applied are K-means, density-based (DBSCAN) and hierarchical clustering.

2.1 K-MEANS

As a first attempt at clustering, we used K-Means, an unsupervised learning method. The goal of this algorithm is to find groups of objects that have common features within them, minimizing their distance (intra-cluster distance), and at the same time, maximizing the distance between different groups (inter-cluster distance). To use the K-Means algorithm, we had to choose the number of K clusters into which to divide the dataset. The study of the value of K to be used was done by taking into consideration the trend of SSE and silhouette as K changes.

FIGURE 4: SSE GRAPH

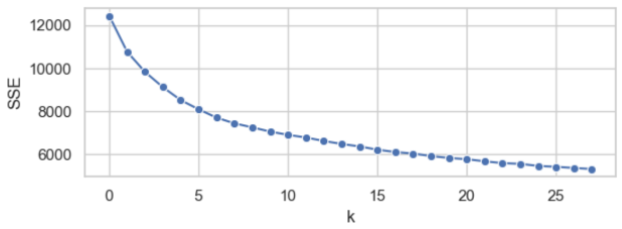
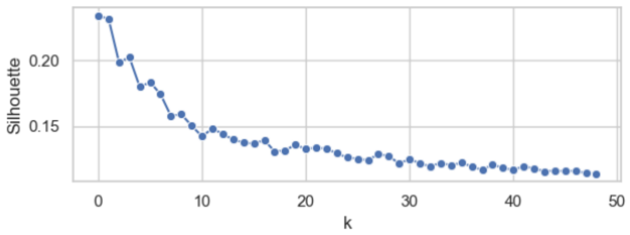


FIGURE 5: SILHOUETTE GRAPH



To evaluate the optimal number for K, the SSE (Sum of Squarred Error) and the Silhouette Score were calculated:

Running the algorithm for a range of K values between 2 and 51, as can be seen from figure 4, the best results are between 3 and 4, and thanks to figure 5, it can be seen that the optimal value of the coefficient is for K=3. A summary of the results is given in the following table:

N° of cluster	SSE	Silhouette
K = 3	10775.39	0.23
K = 4	9837.89	0.19

Based on the results obtained, it was decided to choose K=3,
which represents the best compromise between the number of clusters and the value of the two coefficients.

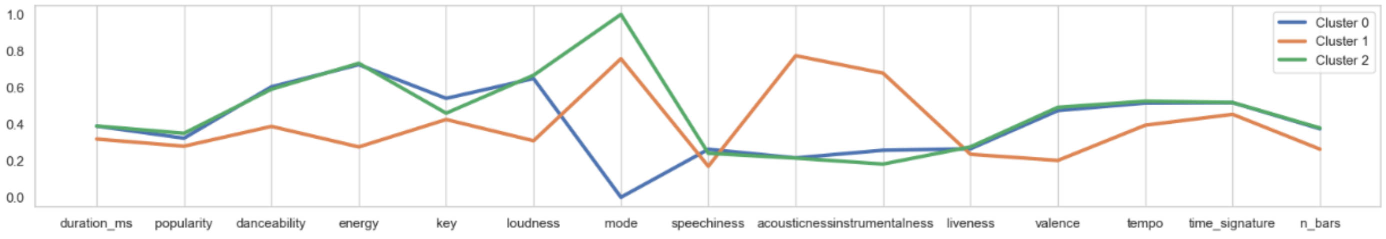
The size for each cluster is listed below and the values of the corresponding centroids within the parallel co-ordinate graph are shown:

1. Cluster 0: 2427

2. Cluster 1: 4989

3. Cluster 2: 7584

FIGURE 6 : PARALLEL COORDINATES



It can be seen from the above figure that:

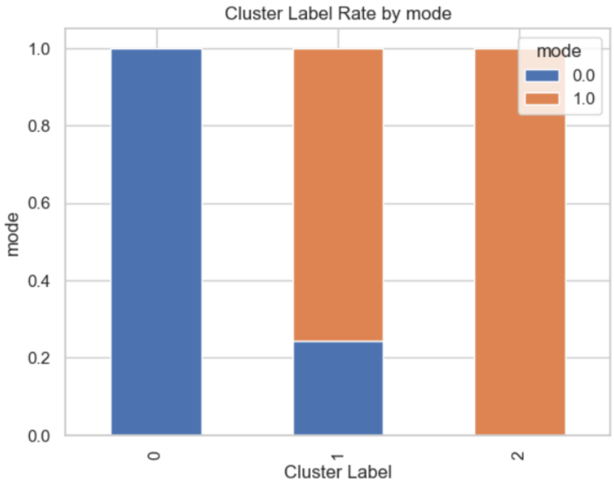
- Cluster 0 shows a discontinuous trend: in particular, very high values are observed for the energy variable and the loudness variable, as opposed to the mode variable, which shows minimum values

- Cluster 1 has a homogeneous trend for the first 6 variables and then undergoes abrupt trends with high peaks for the mode variable and the acousticness variable, while the speechiness and valence variables show low values

- Cluster 2 shows a discontinuous trend: very high values are observed for mode and very low values for the variable instrumentalness

In particular, the best differentiation of the three clusters is observed for the variable mode.

FIGURE 7: DISTRIBUZIONE MODE



Summarising the results obtained, in the distribution of the mode variable (figure 7) within the clusters, a distribution greater than mode=1 (major) can be observed. As can be seen from the figure, only in cluster 1 is the presence of both mode values with a percentage of 76% for mode = 1 and 24% for mode = 0

2.2 DB-SCAN

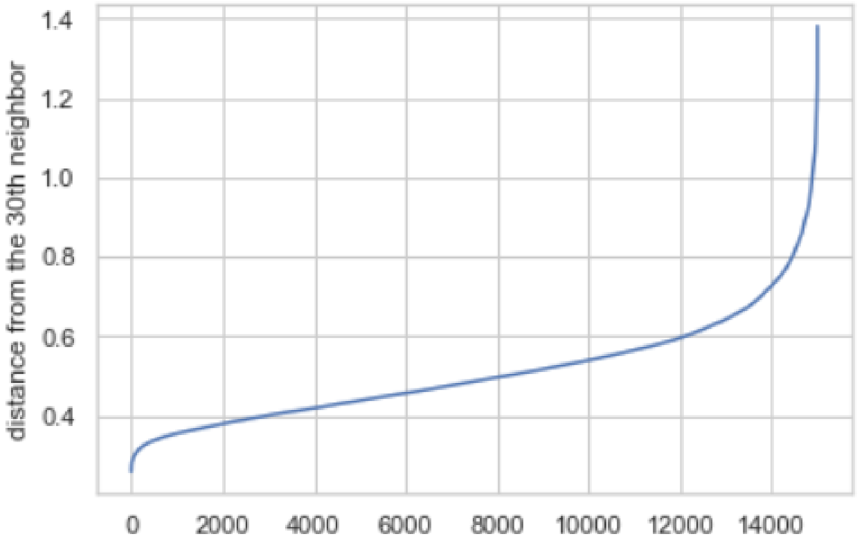
For the density-based clustering part, we decided to use DBSCAN. DBSCAN is an unsupervised learning technique used to identify clusters of different sizes and shapes.

It works with two parameters:

- The radius ε (eps), which specifies the distance within which two points are considered to be neighbors.
- Min_Samples, which is the minimum number of points needed to form a cluster.

To obtain an interesting result from running DBSCAN it is necessary to estimate the value of the MinPoints and Epsilon parameters. The choice of Epsilon is not trivial, as selecting too small a value would have too many points shown as outliers while too high a value would result in a single cluster. To estimate the value of Epsilon we used the K-Nearest neighbor method. In the graph in Figure 8 we show the curve obtained by the K-Nearest neighbor method.

FIGURA 8: Average of the distances between each point and the nearest 30 nodes



The minsample chosen by us is set at 30. This value was obtained by multiplying the number of features in the dataset (in this case 15) by 2. To estimate the value of epsilon, we chose a point lower than the elbow point so that we would have more meaningful clusters. The value we chose for epsilon is 0.7 and for Min_Sample is 30. The result obtained from DBSCAN is the formation of 265 noise points with 2 cluster sizes: **Cluster 0**:9294; **Cluster 1**: 5441;

The silhouette score of this DBSCAN model is 0.23

2.3 HIERARCHICAL CLUSTERING

We performed Hierarchical Clustering using three different methods, choosing those that are least susceptible to noise and outliers namely COMPLETE, AVERAGE and WARD.

2.3.1 ANALYSIS OF THE DENDOGRAMS OBTAINED

As a first step, we used the ‘connectivity constraint’, which is a technique used in some clustering algorithms, including those based on hierarchy such as agglomerative clustering. In practice, the ‘connectivity constraint’ limits which points can be connected during the process of joining clusters, based on a predefined proximity measure. The connectivity matrix can be constructed according to different logics, e.g. using the Euclidean distance between points or considering only those points that are the k-nearest neighbours of each other. We worked using the second method. In this case, we considered the 100 nearest neighbours for each point and, in addition, the point itself was not included as the nearest neighbour.

In addition, we set the distance threshold to 0, which means that the complete tree will be calculated during clustering. Clusters will be joined until all linkage distances are greater than the distance threshold.

For choosing the best number of clusters, we calculated the silhouette score for each of these methods. The results are:

COMPLETE METHOD:

```
Number Clusters:  3
Cluster {0:14993, 1:2, 2:5}
Silhouette Score 0.2506780422173214
```

AVERAGE METHOD:

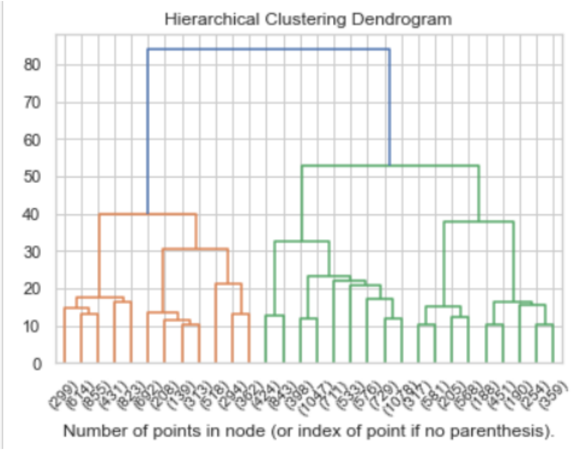
```
Number Clusters:  3
Cluster {0:14997, 1:2, 2:1}
Silhouette Score 0.27859383603438015
```

WARD METHOD:

```
Number Clusters:  3
Cluster {0:5548, 1:3113, 2:6339}
Silhouette Score 0.20199242103655426
```

If we had to choose the best clustering based on the silhouette value, we would choose the average method. I nstead, the final choice falls on the ward method because it has a better distribution of the data in the different clusters.

FIGURA 9: DENDROGRAM BY WARD METHOD



CHAPTER 3

PRE-PROCESSING PHASE

Before starting a classification model, we cleaned up the dataset by eliminating missing values and outliers. The next step was to transform the gender variable from categorical to numerical because classification models work best with variables of this type. The goal of our analysis is to predict popularity based on the attributes of our cleaned dataset. Since popularity is a continuous variable, it was necessary to implement a discretization technique to transform it into categorical. We chose five categories or classes for popularity: **-**

- “**Very low**” if less than or equal to 10

- “**Low**” if between 11 and 20

- “**Medium**” if between 21 and 30

- “**High**” if between 31 and 45

- “**Very high**” if greater than 45

DATA CLASSIFICATION

Regarding the classification procedure, we considered as dependent variable the attribute popularity which has 5 class, while as independent variables we used 15 numeric attributes. Then divide our dataset into training set (70%) and test set (30%).

Before proceeding to the application of the different classification algorithms, we performed the feature standardization operations.

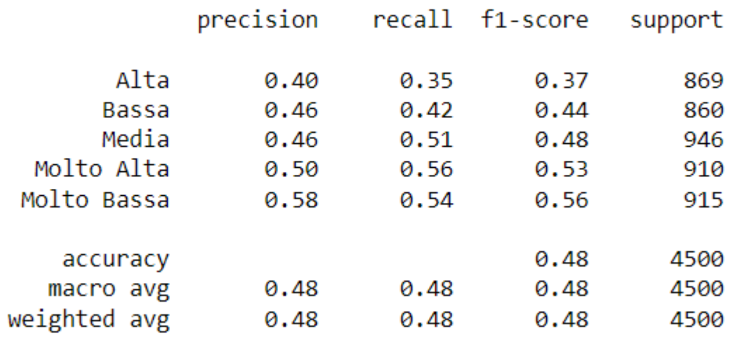
The following paragraphs will explain the three classifiers we used in our analysis, and then conclude the chapter by choosing the classifier that best evaluates our model.

3.1 DECISION TREE

The creation of the decision tree was done considering three metrics:

- Max_depth (number of children) → 8
- Min_samples_leaf (minimum number of samples that must be present in a tree leaf) → 10
- Min_sample_split (minimum number of samples required to subdivide an internal tree node.)→ 30

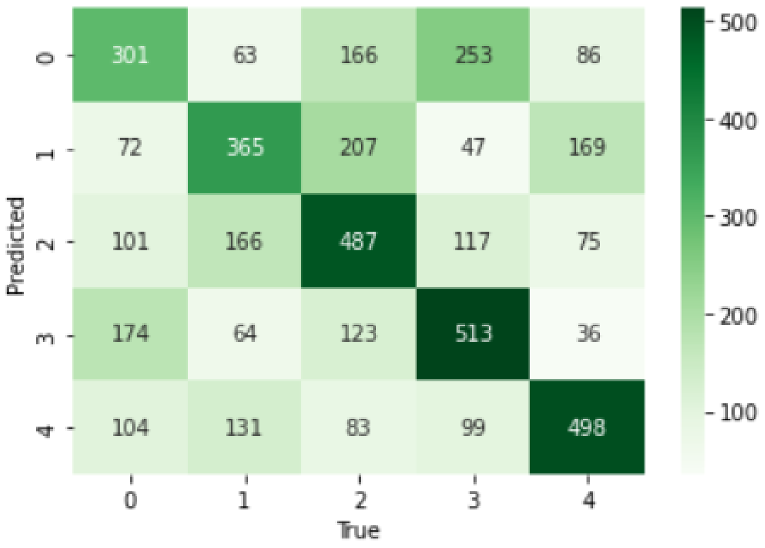
To evaluate the results of the decision tree we show the report with the model performance measures



```
              precision    recall  f1-score   support

        Alta       0.40      0.35      0.37       869
       Bassa       0.46      0.42      0.44       860
       Media       0.46      0.51      0.48       946
  Molto Alta       0.50      0.56      0.53       910
 Molto Bassa       0.58      0.54      0.56       915

    accuracy                           0.48      4500
   macro avg       0.48      0.48      0.48      4500
weighted avg       0.48      0.48      0.48      4500
```

The most important results to display are the **accuracy** which is equal to **0.48**

In the table below, the confusion matrix with the actual values in the x-axis and the predicted values in the y-axis is refined.

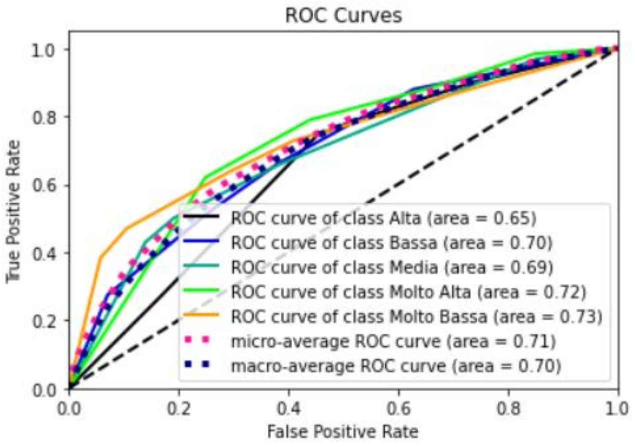


	0	1	2	3	4
0	301	63	166	253	86
1	72	365	207	47	169
2	101	166	487	117	75
3	174	64	123	513	36
4	104	131	83	99	498

The numbers on the main diagonal (301, 365, 487, 513, 498) represent cases where the model made correct predictions. The higher the number, the better the model predicted that class.

- **Class 0**: 301 correct predictions
- **Class 1**: 365 correct predictions
- **Class 2**: 487 correct predictions
- **Class 3**: 513 correct predictions
- **Class 4**: 498 correct predictions

Next, the roc curve of the model, which is a graphical scheme for binary-type Z classification problems, is refined. They are related with the ratio of true positives (TPR) = sensitivity in the y-axis and the ratio of false positives (FPR)=1-specificity in the x-axis.

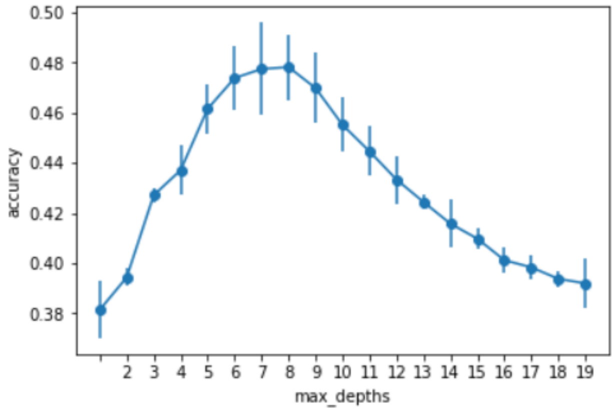


- **Class Alta (AUC = 0.65)**: Moderate performance.
- **Class Bassa (AUC = 0.70)**: Acceptable performance.
- **Class Media (AUC = 0.69)**: Moderate performance.
- **Class Molto Alta (AUC = 0.72)**: Acceptable performance.
- **Class Molto Bassa (AUC = 0.73)**: Acceptable performance.

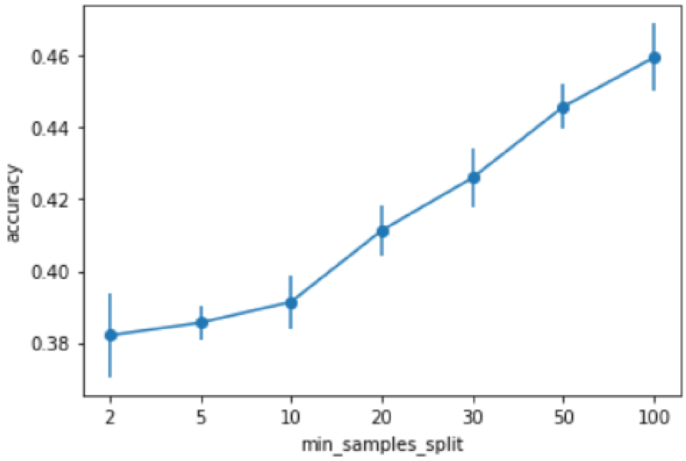
Micro-average ROC curve (AUC = 0.71): It indicates an acceptable average performance of the model considering all class instances as binary classification problems.

Macro-average ROC curve (AUC = 0.70): It indicates an acceptable average performance of the model by taking the average AUC of all classes.

Next, we tested the decision tree model by changing the number of max_depth (max depth) from 2 to 19. We see the accuracy results and plotted the results and saw that as max depth increases, accuracy decreases never going beyond a certain threshold (in this case 8)



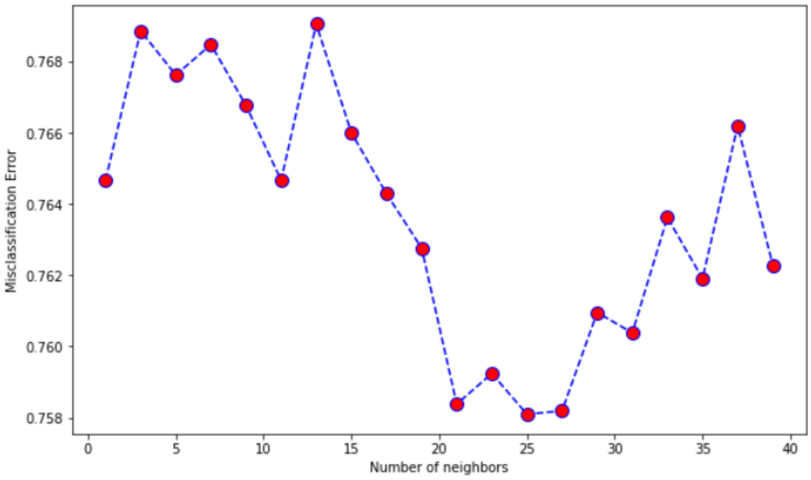
We also tested the decision tree model by gradually changing the number of min_samples_split by putting them with the values of [2, 5, 10, 20, 30, 50, 100] and noticed that as the value of min_samples_split increases the accuracy.



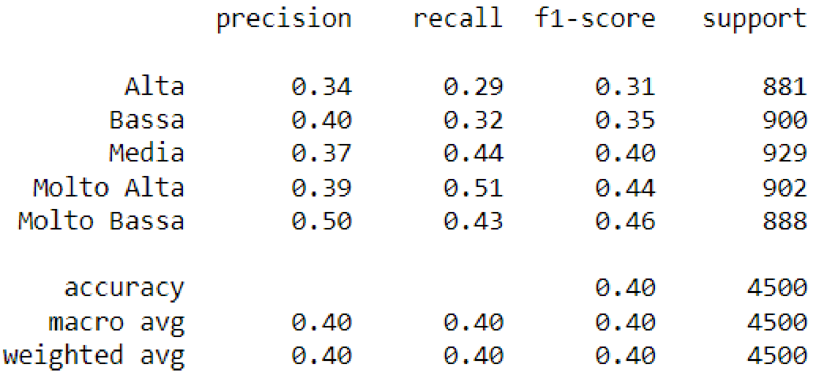
As can be seen from the graph, the best accuracy value is obtained with min_sample_split =100. We have chosen 30 as the value in order to have a model that is not too complex.

3.2 KNN

The knn is an algorithm to classify test instances according to its majority class of nearest neighbors. The best number of neighbors was found with an algorithm where for each k (neighbors) the misclassified error is calculated and, the k with the lowest misclassification error value is chosen. In this case the best k is 25 as we see from the following graph:



We then fit the knn model by putting in our variables, using uniform weights for all attributes, and derive per
formance values,
which will be as follows:



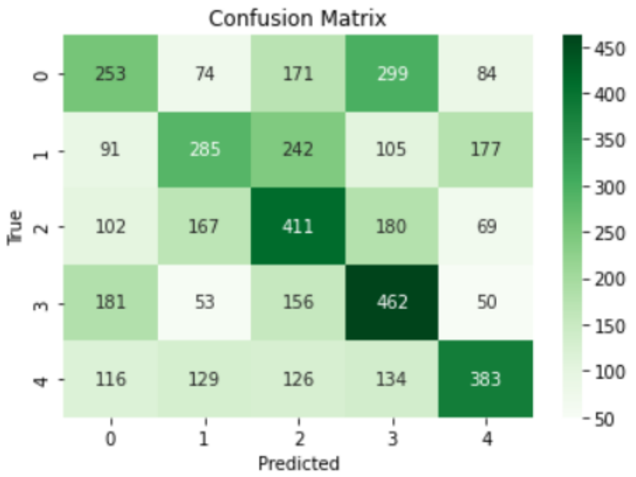
```
              precision    recall  f1-score   support

        Alta       0.34      0.29      0.31       881
       Bassa       0.40      0.32      0.35       900
       Media       0.37      0.44      0.40       929
  Molto Alta       0.39      0.51      0.44       902
 Molto Bassa       0.50      0.43      0.46       888

    accuracy                           0.40      4500
   macro avg       0.40      0.40      0.40      4500
weighted avg       0.40      0.40      0.40      4500
```

The most important results to display are the **accuracy** which is equal to **0.40**

In the table below, the confusion matrix with the actual values in the x-axis and the predicted values in the y-axis is refined.

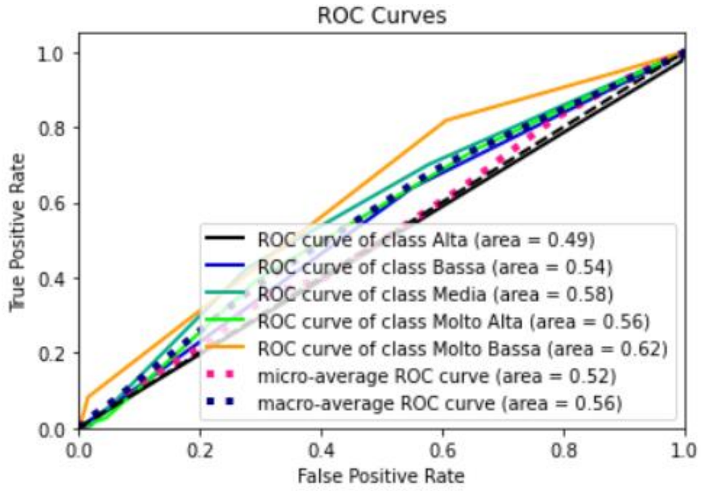
The numbers on the main diagonal (253,285, 411, 462, 383)
represent the cases in which the model made
correct predictions. The higher the number, the better the
model predicted that class.

- **Class 0**: 253 correct predictions
- **Class 1**: 285 correct predictions
- **Class 2**: 411 correct predictions
- **Class 3**: 462 correct predictions
- **Class 4**: 383 correct predictions



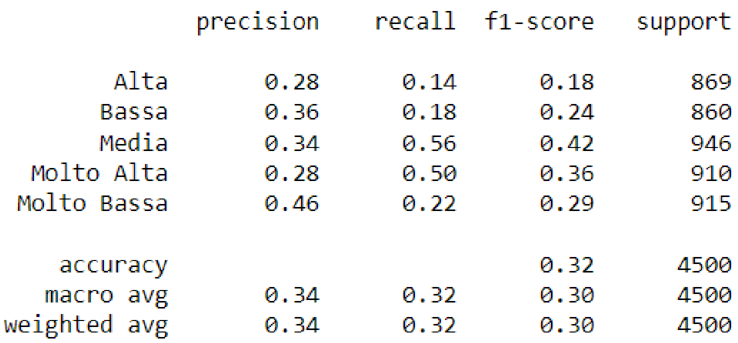
	0	1	2	3	4
0	253	74	171	299	84
1	91	285	242	105	177
2	102	167	411	180	69
3	181	53	156	462	50
4	116	129	126	134	383

Regarding the roc curve: The Auc values of the classes considered are relatively low



3.3 NAÏVE BAYES

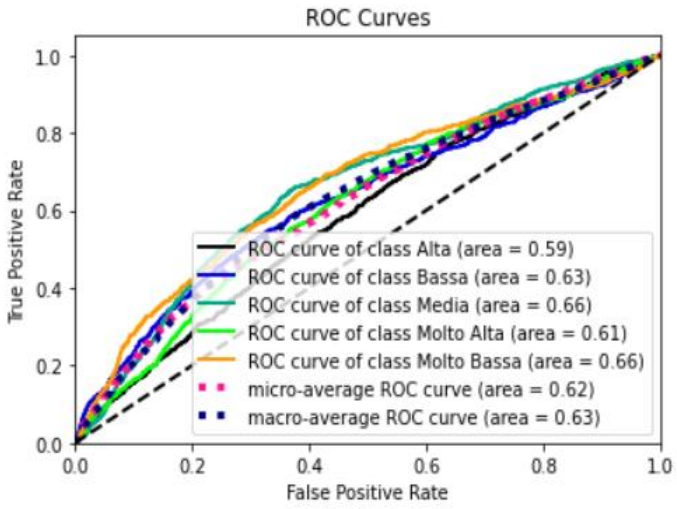
The third proposed model is a naïve bayes classifier model that is a probabilistic classifier model based on the application of Bayes' theorem with assumptions of independence.The response variables and explanatory variables are the same, and by going to implement the naive Gaussian Bayes model we obtain the following results:



```
              precision    recall  f1-score   support

        Alta       0.28      0.14      0.18       869
       Bassa       0.36      0.18      0.24       860
       Media       0.34      0.56      0.42       946
  Molto Alta       0.28      0.50      0.36       910
 Molto Bassa       0.46      0.22      0.29       915

    accuracy                           0.32      4500
   macro avg       0.34      0.32      0.30      4500
weighted avg       0.34      0.32      0.30      4500
```

As we see, the **accuracy** is **0.32**

Regarding the roc curve: The Auc values of the classes considered are relatively low



3.4 FINAL CONSIDERATIONS

Among the three models analyzed, the best one is the Decision Tree because it has higher values for accuracy. which means that the values predicted by the model (compared to the other two models) are closer to the actual values.

The worst model among the three is the KNN because in addition to having lower accuracy, it also has the worst Roc Curves

CHAPTER 4

PRE-PROCESSING PHASE

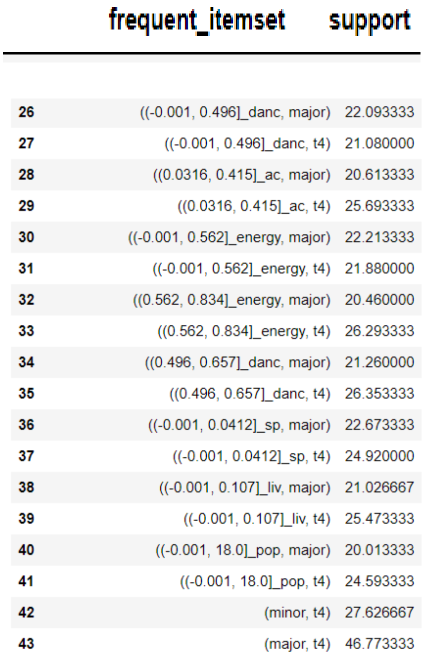
After cleaning the dataset, we used 10 attributes: duration_ms, popularity, danceability, energy, mode, speechiness, acousticness, liveness, time_signature, genre. We then discretized the numeric attributes into 3 intervals for each attribute

PATTERN MINING

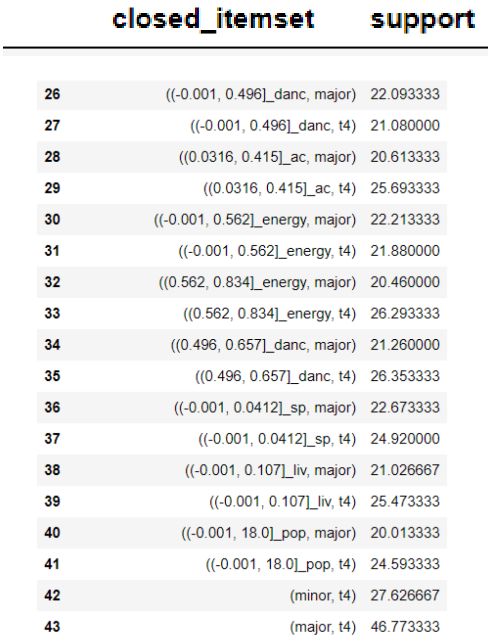
In this section we describe the process of association rules analysis. In the first section we discuss the preliminary operations performed on the dataset to prepare the data for subsequent operations. In subsequent sections we extract frequent itemsets and then, from these itemsets, association rules. We used as our algorithm for pattern extraction the **A PRIORI ALGORITHM**

4.1 ITEMSET

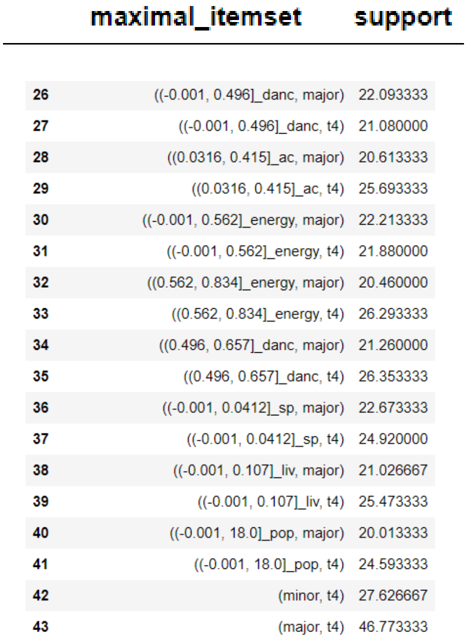
For Frequent ItemSets we used a MinSupp equal to 20% as a parameter by going to select only sets with 2 items. In the table we give the list of the most frequent ItemSets. The list of coincides with the list of "Closed" ItemSets and “Maximal” Itemsets, this is because the Frequent ItemSets are an over set of the "closed" ones listed above.



	frequent_itemset	support
26	((-0.001, 0.496]_danc, major)	22.093333
27	((-0.001, 0.496]_danc, t4)	21.080000
28	((0.0316, 0.415]_ac, major)	20.613333
29	((0.0316, 0.415]_ac, t4)	25.693333
30	((-0.001, 0.562]_energy, major)	22.213333
31	((-0.001, 0.562]_energy, t4)	21.880000
32	((0.562, 0.834]_energy, major)	20.460000
33	((0.562, 0.834]_energy, t4)	26.293333
34	((0.496, 0.657]_danc, major)	21.260000
35	((0.496, 0.657]_danc, t4)	26.353333
36	((-0.001, 0.0412]_sp, major)	22.673333
37	((-0.001, 0.0412]_sp, t4)	24.920000
38	((-0.001, 0.107]_liv, major)	21.026667
39	((-0.001, 0.107]_liv, t4)	25.473333
40	((-0.001, 18.0]_pop, major)	20.013333
41	((-0.001, 18.0]_pop, t4)	24.593333
42	(minor, t4)	27.626667
43	(major, t4)	46.773333

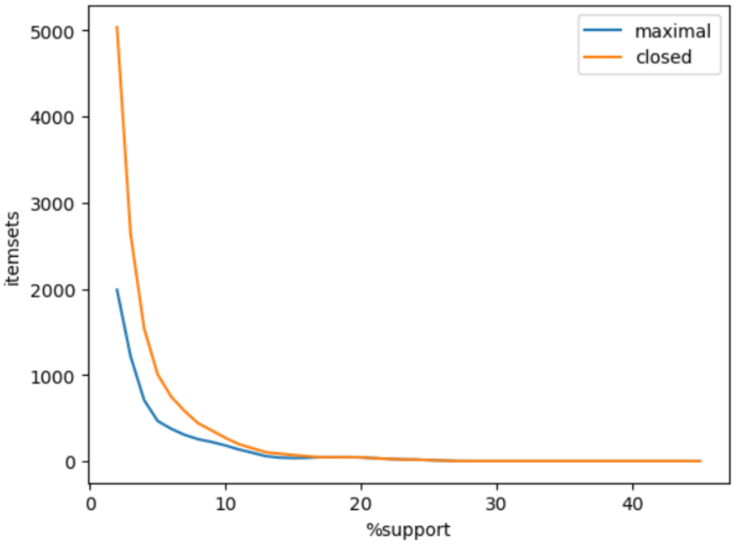


	closed_itemset	support
26	((-0.001, 0.496]_danc, major)	22.093333
27	((-0.001, 0.496]_danc, t4)	21.080000
28	((0.0316, 0.415]_ac, major)	20.613333
29	((0.0316, 0.415]_ac, t4)	25.693333
30	((-0.001, 0.562]_energy, major)	22.213333
31	((-0.001, 0.562]_energy, t4)	21.880000
32	((0.562, 0.834]_energy, major)	20.460000
33	((0.562, 0.834]_energy, t4)	26.293333
34	((0.496, 0.657]_danc, major)	21.260000
35	((0.496, 0.657]_danc, t4)	26.353333
36	((-0.001, 0.0412]_sp, major)	22.673333
37	((-0.001, 0.0412]_sp, t4)	24.920000
38	((-0.001, 0.107]_liv, major)	21.026667
39	((-0.001, 0.107]_liv, t4)	25.473333
40	((-0.001, 18.0]_pop, major)	20.013333
41	((-0.001, 18.0]_pop, t4)	24.593333
42	(minor, t4)	27.626667
43	(major, t4)	46.773333



	maximal_itemset	support
26	((-0.001, 0.496]_danc, major)	22.093333
27	((-0.001, 0.496]_danc, t4)	21.080000
28	((0.0316, 0.415]_ac, major)	20.613333
29	((0.0316, 0.415]_ac, t4)	25.693333
30	((-0.001, 0.562]_energy, major)	22.213333
31	((-0.001, 0.562]_energy, t4)	21.880000
32	((0.562, 0.834]_energy, major)	20.460000
33	((0.562, 0.834]_energy, t4)	26.293333
34	((0.496, 0.657]_danc, major)	21.260000
35	((0.496, 0.657]_danc, t4)	26.353333
36	((-0.001, 0.0412]_sp, major)	22.673333
37	((-0.001, 0.0412]_sp, t4)	24.920000
38	((-0.001, 0.107]_liv, major)	21.026667
39	((-0.001, 0.107]_liv, t4)	25.473333
40	((-0.001, 18.0]_pop, major)	20.013333
41	((-0.001, 18.0]_pop, t4)	24.593333
42	(minor, t4)	27.626667
43	(major, t4)	46.773333

The graph below compares the "maximum" feature set with the "closed" feature set considering the percentage of support with the number of features.



It can be seen that with equal support the number of items in the closed itemsets is always greater than the maximum number of itemsets.

We leave the number of items to be selected equal to 2, by increasing the number of items (e.g. equal to 3) there was no itemset with support >=20%, but the max support percentage reached 46%.

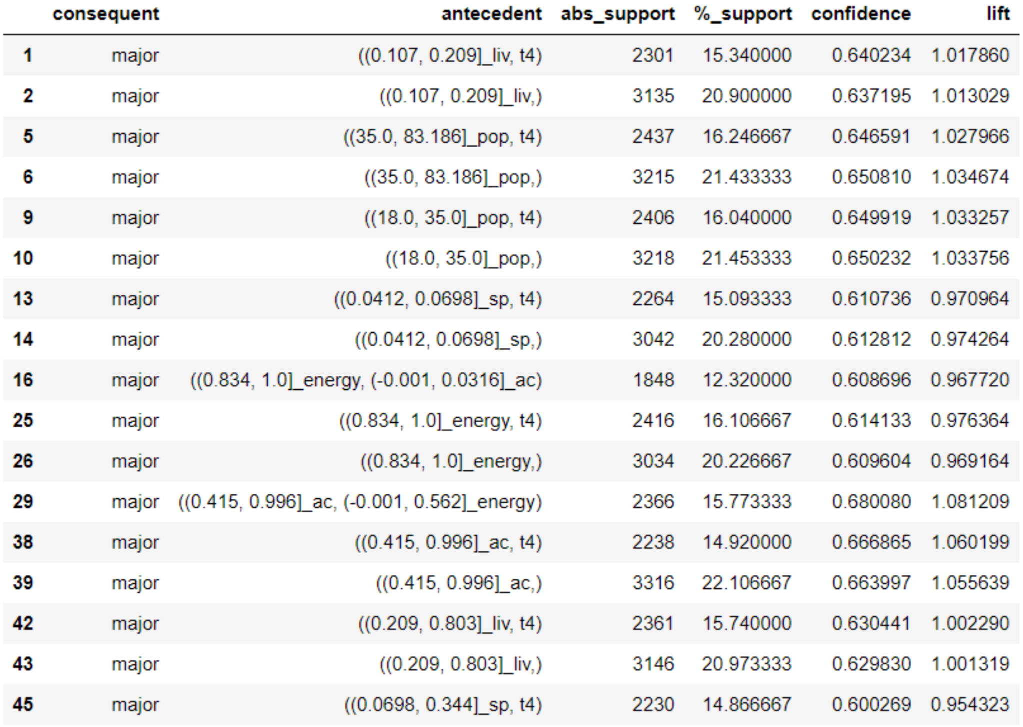
Regarding the support percentage we chose the threshold of 20%because raising it further (example 30%) resulted in very few records

4.2 ASSOCIATION RULES

In the table below we have given the list of association rules, deciding to extract these rules by considering only frequent itemsets that had a length greater than or equal to 2.

As a confidence value we set the threshold of 60% as a parameter. In these rules we put the values of attribute mode ("major" and "minor") arranged in descending order according to the weight given as postcondition values.

For example, the first record in these association rules stands for the song that has **(0.107,0.209) liv** and the time signature equal to **4** then it will simply be “major” mode (and not minor) [the mode of the attribute mode variable will be major].



	consequent	antecedent	abs_support	%_support	confidence	lift
1	major	((0.107, 0.209]_liv, t4)	2301	15.340000	0.640234	1.017860
2	major	((0.107, 0.209]_liv,)	3135	20.900000	0.637195	1.013029
5	major	((35.0, 83.186]_pop, t4)	2437	16.246667	0.646591	1.027966
6	major	((35.0, 83.186]_pop,)	3215	21.433333	0.650810	1.034674
9	major	((18.0, 35.0]_pop, t4)	2406	16.040000	0.649919	1.033257
10	major	((18.0, 35.0]_pop,)	3218	21.453333	0.650232	1.033756
13	major	((0.0412, 0.0698]_sp, t4)	2264	15.093333	0.610736	0.970964
14	major	((0.0412, 0.0698]_sp,)	3042	20.280000	0.612812	0.974264
16	major	((0.834, 1.0]_energy, (-0.001, 0.0316]_ac)	1848	12.320000	0.608696	0.967720
25	major	((0.834, 1.0]_energy, t4)	2416	16.106667	0.614133	0.976364
26	major	((0.834, 1.0]_energy,)	3034	20.226667	0.609604	0.969164
29	major	((0.415, 0.996]_ac, (-0.001, 0.562]_energy)	2366	15.773333	0.680080	1.081209
38	major	((0.415, 0.996]_ac, t4)	2238	14.920000	0.666865	1.060199
39	major	((0.415, 0.996]_ac,)	3316	22.106667	0.663997	1.055639
42	major	((0.209, 0.803]_liv, t4)	2361	15.740000	0.630441	1.002290
43	major	((0.209, 0.803]_liv,)	3146	20.973333	0.629830	1.001319
45	major	((0.0698, 0.344]_sp, t4)	2230	14.866667	0.600269	0.954323