Can "physical" characteristics of music explain motor and emotional states of the human body? A survey on the relationships between music and our body.

Corrado Baccheschi, Noemi Boni e Isabel Santucci

1 Introduction

Music plays a crucial role in our emotional and physical well-being, and can be utilized in various situations such as for entertainment, relaxation, and motivation during exercise. This research aims to explore the potential connections between different technical aspects of music (such as rhythm, tone, and duration) and their impact on psychological and physical responses (such as valence, energy, and danceability). To achieve this, we will analyze a dataset from Spotify containing various song tracks and specific metrics.

2 Data understanding and preparation

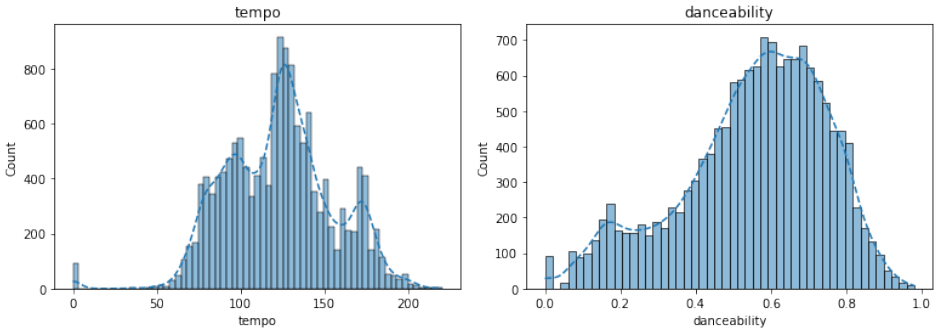
2.1 Data Semantics

The dataset contains Spotify tracks with different kind of information. The general ones can be detected by the *name* of the track, the performer *artist* or *artists* parted by semicolon, the *album name* from which the track is extracted, the *genre* which the track belongs. Moreover explicit lyrics is identified by *explicit* attribute, whose value is True if there is, otherwise is False or Unknown (if it's not indicated). There are several information about the duration of the track such as *duration_ms*, which measure it in milliseconds, the *tempo*, that describes the track in beats per minute (measured in BPM), and even an estimated *time_signature*. Moreover, specific technical features can be found, related to the kind of music arranged. The *danceability* value expresses how suitable a track is for dancing, a value from 0.00 to 1.0 and the *energy* measures the percepted intensity of the track, instead *valence* states for the positivity transmitted, with values from 0.0 to 1.0. *Key* is the track tonality, described by integers map to pitches, using standard Pitch Class notation and *acousticness* specifies if the track is acoustic or not, with measures from 0.0 to 1.0. *Loudness* reveals the general loudness of the track measured in decibels (dB) and the *mode* indicates the modality (major or minor) of a track, Major is represented by 1 and minor is 0. *Speechiness* defines the presence of spoken words in a track, whereas *Instrumentalness* detects the presence of vocal and the closer value is 1.0. The *liveness* recognizes the presence of the audience in the recording and the *n_bars* refers to the number of intervals of beats while *n_beats* is the total number of time intervals of beats. Finally, there are two attributes related to the success of the track with different units of measurement: *popularity* is a value between 0 and 100 whereas *popularity_confidence* values go from 0.0 to 1.0. There is also an unknown attribute called *processing*, it's not clear what it properly represents and its relevance, but we'd try to reach information about it studying its distribution.

Considering the purpose of our research, it was deemed necessary to disregard popularity-related attributes such as *popularity* and *popularity_confidence*, since they don't provide useful information to understand physical and emotional influences, but they are instead metrics related to marketing reports.

2.2 Distribution of the variables and statistics

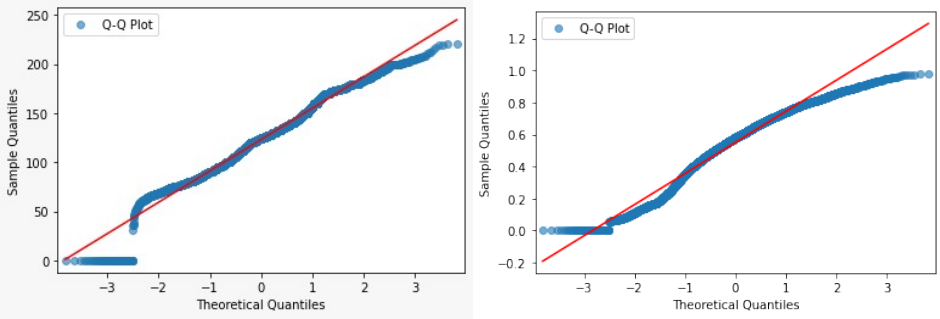
We first selected the most relevant variables for our purpose from the different kinds available. We also tested for the presence of normal distributions[1] in the continuous variables. While we didn't find any, we discovered some attributes that are close to meeting this criterion. In the next figure, you can see the histograms for the *tempo* and *danceability* attributes, which are the closest to a normal distribution.



(a) *Histogram for tempo attribute.* (b) *Histogram for danceability attribute.*

Figure 1: Distribution of *tempo* and *danceability* showed by histograms. Note the blue dashed line that limits and draws the curves.

The distributions shown in the plots are *not* normal, as indicated by the blue dashed line. The *tempo* has a kurtosis[2] of 0.42, while the *danceability* has a kurtosis of -0.13. In order to test the hypothesis of normality, the Shapiro test was not used due to potential inaccuracies with large data sets (N >5000). Instead, the Kolmogorov-Smirnov test[3] was applied, which is not affected by this issue. The resulting p values[4] were 0.0, using a *threshold* of 0.05, leading to the rejection of the normality hypothesis. Additionally, the QQPlot was used to identify the variable closest to a normal distribution, following the rule that if the blue line closely follows the red line, the data is likely to be normally distributed.



(a) *QQPlot for tempo attribute.* (b) *QQPlot for danceability attribute.*

Figure 2: *tempo* and *danceability* on a QQPlot. It's easy to see that the blue line of *tempo* (on the left) is one that follows better the red line.

At this level of analysis, outliers are still present, as can be seen in Figure 2 (the separate lines at the

[1]In statistics, a normal distribution or *Gaussian* distribution is a type of continuous probability distribution for a real-valued random variable.

[2]The Kurtosis value measures how many values lies around the mean. A curve with a high value of Kurtosis shows a peak around the mean and a steep nearby the tails, while with a low value the shape of the peak is more flat and the decrease around the tails is slower. Like the skewness, the Kurtosis value is normalized on 0.

[3]The Kolmogorov-Smirnov is a statistical test, which was developed by Andrey Kolmogorov and Nikolai Smirnov. It is used to determine whether a sample comes from a particular probability distribution.

[4]In statistic, the p-value provides a way to assess whether the observed data supports the null hypothesis ($p > \alpha$) or contradicts it in favor of the alternative hypothesis ($p \leq \alpha$)

bottom). The distribution may become closer to a normal distribution after removing these outliers[5][6].

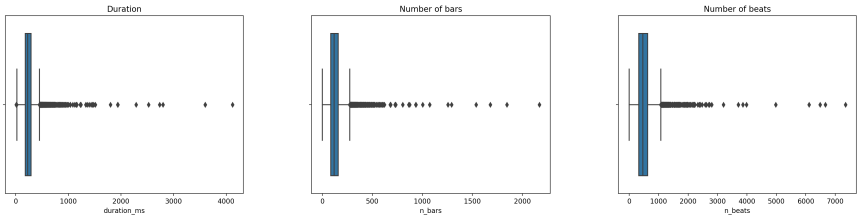
2.3 Variable transformations

When conducting analyses, it's important to consider the inconsistency of data formats. Since data are stored in different formats or follow different conventions, it can be difficult to integrate and compare them effectively. Inconsistency of data formats can lead to misinterpretation and incomplete or biased analysis. Firstly, the similarity between the two columns *duration_ms* and *features_duration_ms* was compared, since on first viewing the records looked similar. Thus, to determine whether to possibly consider one of the two columns, the cosine similarity was applied, obtaining 0.99 as a result, concluding the data are almost 100% equal. Hence, this value allowed to determine the elimination of that column. The cosine was also calculated between the attributes *n_beats* and *n_bars*, obtaining a value of 0.99. Despite their similarity, it was decided to keep both since they present two different types of information: *n_beats* measures the total number of time intervals of beats, while the *n_bars* concerns the total number of time intervals of the bars throughout the track. Moreover, data transformation and harmonization are other processes which may be necessary to unify formats and ensure consistency in data analysis: in fact, in the dataset there are objects as categorical attributes (name, artist, name of the album, genres), booleans as *explicit*, integers as *key* and *mode*, and some floats values. Since most of the attributes are in float format, it was applied a conversion from the integers to float obtaining a better standardization. Another type of standardization carried out concerns the numbers after the comma, rounded by 2.

2.4 Assessing data quality

When analyzing the initial data, it is important to identify outliers, which are values that are significantly different from the majority of the data. Outliers can affect the analysis and cause data quality issues. To identify outliers, box plots can be used to visually display the distribution of data for continuous variables. The middle line in the box represents the median of the data, and the bottom and top of the box show the 25^{th} and 75^{th} percentile. The length of the box is the interquartile range (IQR), and the lines extending from the box are called whiskers. These whiskers represent the expected data variation and extend 1.5 times the IQR from the top and bottom of the box. Any data points beyond the whiskers are considered outliers.

Figure 3 shows the graphs depicting the distribution of the variables *duration_ms*, *n_bars* and *n_beats*. These graphs help us easily understand the location of numerous outliers in relation to the majority of data.



(a) *Box plot for duration_ms* (b) *Box plot for the n_bars* (c) *Box plot for the n_beats*

Figure 3: Box plots made with seaborn for the attributes *duration_ms*, *n_bars* and *n_beats*.

Different kind of techniques can be used to catch and detect outliers such as:

- Distance Methods (z-score)
- Range Methods (interquartile range IQR)

[5]We didn't find changing in p value which in fact remains of 0.0, even after the elimination.
[6]See section 2.4 about outliers treatment and their detection.

We decided to use the z-score[7] distance in order to compute the distance of all the values from the mean of own distribution. Hence, for each numeric column, that is relevant for our analysis, an algorithm is run to find the outliers, applying a z-score threshold of 3. Below, the changing of *n_beats* is depicted by an histogram, to show the variation of distribution caused by the presence or absence of outliers.

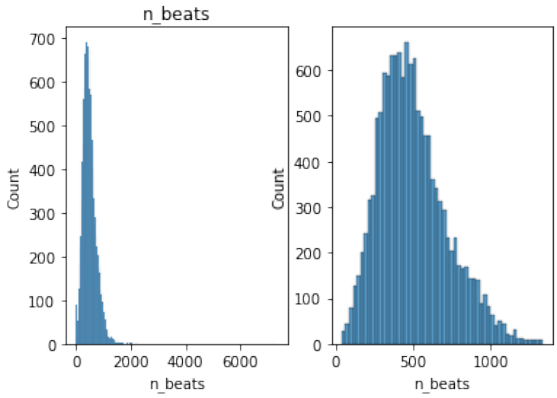


Figure 4: Histograms for *n_beats* attribute, observed before and after the elimination of the outliers.

Results in the Figure 4 for *n_beats* attribute show on the left the initial distribution containing outliers, with skewness[8] index of 5.29 and Kurtosis of 91.30. On the right, the final distribution is depicted showing an improvement both of the distribution toward a normal curve, both for the indexes like skewness and Kurtosis: reaching value of 0.73 for the first one, and 0.41 for the second one. Let's now retake the study on *tempo* attribute in Section 2.2. It was the nearest to a normal distribution. The new QQPlot is built below after the elimination of the outliers.

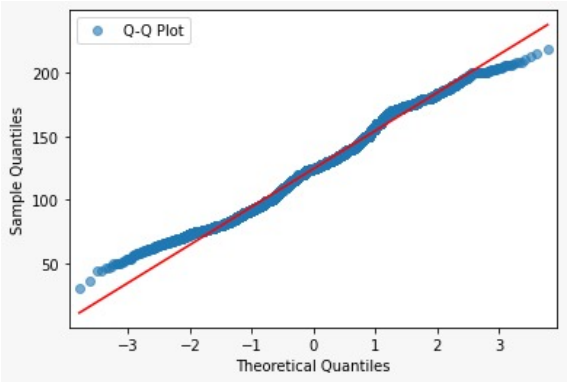


Figure 5: The *tempo* attribute after the elimination of outliers. It's simple to note that the blue line fits now better the red line.

Before conducting further analysis on the data, it is important also to take into account the presence of missing values. Missing values are easily visualized through the matrix shown in Figure 6, which displays the missing values for the attributes *mode* and *time_signature*. The null matrix plot allows us to understand the distribution of data within the entire dataset in all columns at once. It also displays a spark line that emphasizes the rows with the highest and lowest nullity in a data set.

[7]The z-score value represents how many standard deviations are between a value and the mean. We can calculate it dividing the difference between a value and the mean with the standard deviation. In a normal curve the 99.7% of values is between +3 and -3, while the 95% is between +2 and -2, so it can be a valuable ally to detect outlier values.

[8]The skewness is a value that defines how much a distribution lacks symmetry. It can have both positive and negative values, and we can say that a curve is mostly symmetrical if its value lies between +0.5 and -0.5. If the value is less than -1 or over +1 we can define the the curve is skewed.

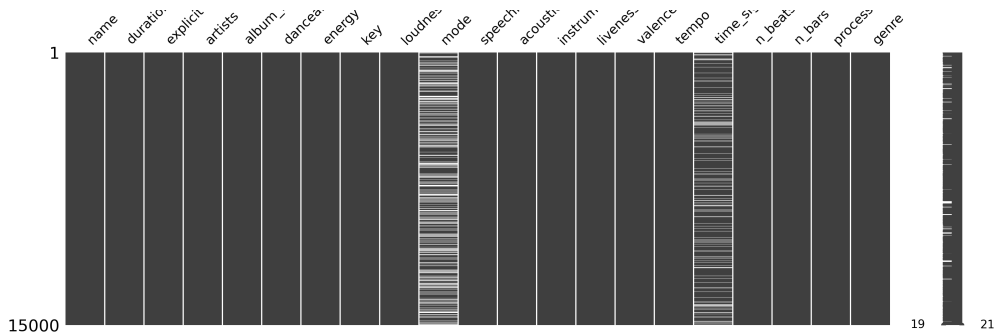


Figure 6: The nullity matrix, made with the library missingno, that shows the amount of missing values for each column.

The missing values in the dataset are handled differently for the two attributes, *mode* and *time_signature*, as they contain integer and continuous values, respectively. The *mode* attribute required an additional step, as discussed in the data semantics section, it is an integer with dichotomy values (e.g., 0 or 1), which can be treated as a categorical attribute due to its binary nature. To fill in the missing data for the *mode* attribute, we used logistic regression to predict the *mode* values, where the response variable has only two possible outcomes. The accuracy of the prediction was 0.63, which, although low, was found to be more effective than simply replacing values with the mode. For the missing values in the *time_signature* attribute, we implemented Ridge Regression[9], a linear regression technique that adds an L2 regularization term to the cost function to prevent overfitting[10], especially when dealing with many features or strongly correlated features. We considered the Mean Squared Error and R-squared, which both resulted in a value close to 0.23. Despite not being very high, we found it more useful to replace the missing values with those predicted by the model, rather than using the mean or median.

2.5 Pairwise correlations

Another important analysis to consider is the potential correlation between variables. It measures the linear relationship between two random variables and it is always between -1 and 1. A value of -1 indicates an inverse linear relationship, 1 indicates a direct linear relationship, and 0 indicates no linear trend between the two variables. The correlation can be easily visualized through a correlation matrix, which is a square and symmetrical matrix with all elements on the main diagonal equal to 1.

[9]The Ridge regression is a method of model tuning adopted when facing data with multicollinearity problems. Those are mostly present in models with a large number of parameters, in those case linear regression models that use least-squares can generate outputs consistently different from the actual targets.

[10]Overfitting is a problem that occurs when the train error of a model reaches a significantly low value, while the test error keeps increasing its own. Usually this problem is a consequence of having too much train epochs, leading to a situation where the model is overtrained on the example it saw but it's unable to generalize when presented to new data.

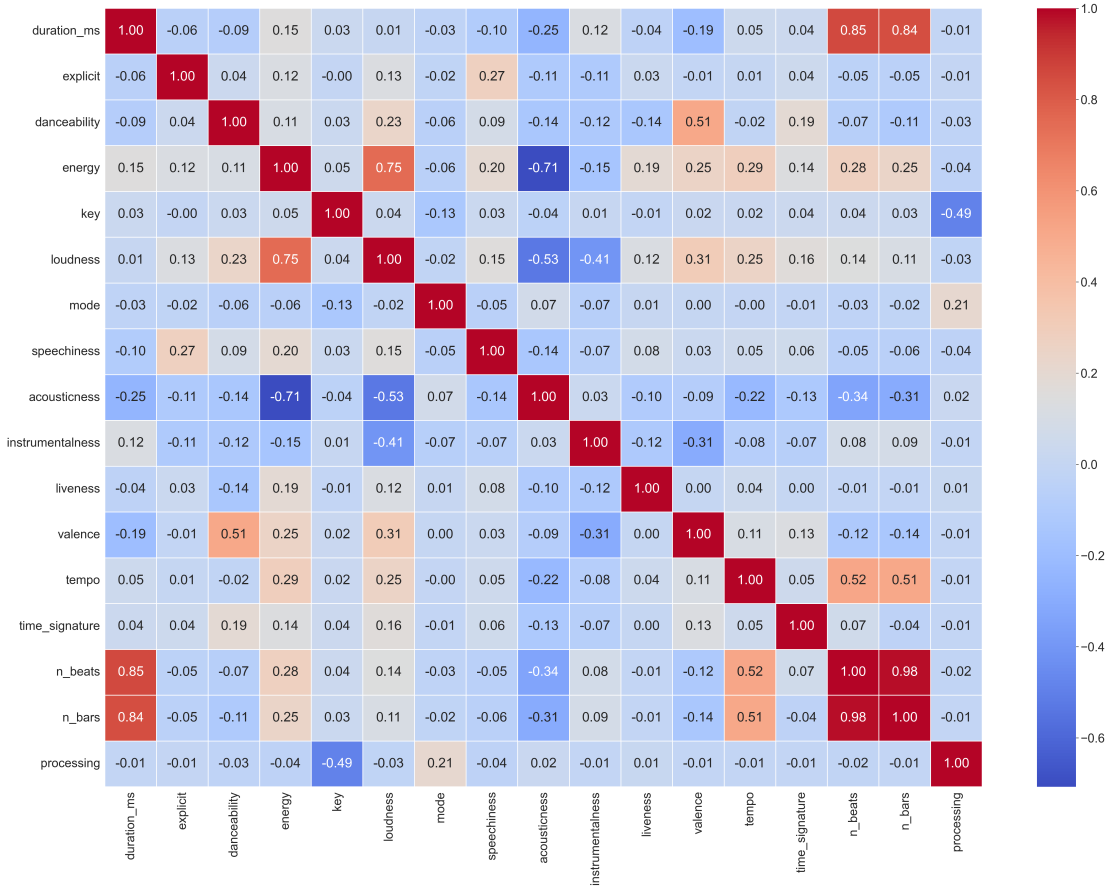


Figure 7: Heatmap made with the seaborn library. In this plot are matched all numerical attributes of the dataset between each other, we can see them both on the rows and the columns. In each square it's shown the numerical value of a correlation, and a colour that displays in a more rapid way how much the attributes are correlated: the more warm the more closer to 1, the more cool the more closer to -1.

Figure 7 shows a strong positive correlation between *duration_ms* and the *n_beats* and *n_bars*. There is also a positive correlation between *valence* and *danceability*, meaning that more positive songs are also more danceable. On the other hand, there are strong negative correlations between *energy* and *acousticness*, indicating that more acoustic songs are less energetic. Additionally, there is a negative correlation between *loudness* and *acousticness*, meaning that acoustic songs tend to have lower loudness.
There is also a soft negative correlation between *key* and *processing* that maybe could help us in our search to find the meaning of *processing*. It may seems that to lower pitch values correspond the highest values of processing of a track. All values in the squares are calculated using the Pearson coefficient[11]. Furthermore, we analyze the relationship between *energy* and *loudness*, which are related to body response and physical characteristics of a track, respectively. These are compared and plotted using the seaborn library, as shown in Figure 8.

[11]In statistic, a coefficient that evaluates the amount of the (linear) correlation. In particular, closer -1 means negative correlation whereas nearer to +1 means positive correlation. 0 indicates absence of linear relationship (but there may be a *non*-linear one)

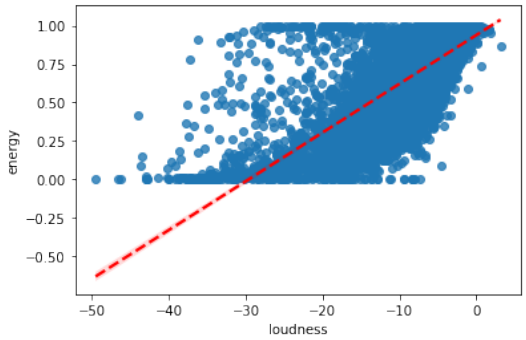


Figure 8: The obtained scatterplot for *energy* and *loudness* attribute. Note the red line that it's fitted with respect the positive correlation between both such variables.

From this plot, it's possible to conclude that more loudness a track contains, consequently more would be the energy percepted. The plot suggests that tracks with higher loudness also have higher perceived energy. These correlations provide a starting point for more in-depth analyses.

3 Clustering

The first type of analysis performed on the data is clustering, which is an unsupervised learning technique that explores data for possible relationships and groupings called clusters, defined as a collection of observations that share similar characteristics.

3.1 K-Means

The first type of clustering algorithm implemented is K-means, a partitioning algorithm. K-means divides the dataset into k numbers of nonoverlapping independent clusters with no internal structures or labels, such that the observations in one cluster are similar to each other and dissimilar to those in the remaining sets. The k-means algorithm attempts to maximize the inter-cluster distance between samples and minimize the intra-cluster distance. The algorithm is based on the concept of a centroid, the center of each cluster, and requires that the number k of clusters is specified a priori. To identify the best number of centroids to use there are two techniques: Elbow method and Silhouette method. The first one consists of K-means iteration for different values of K and each time is calculated a value as the sum of squared distances (SSE) between each centroid and the points in the cluster. Often the identification of the elbow value may not be easy to visualize, so we use KElbowVisualizer library, which selects the optimal number of clusters by fitting the model on a range of values for K. If the line graph resembles an arm, then the "elbow" (the inflection point on the curve) is a good indication that the underlying model fits best at that point. In the viewer "elbow" is noted with a dotted line. In the Figure 9, the optimal number for k is 5, with an SSE of 5770.11.

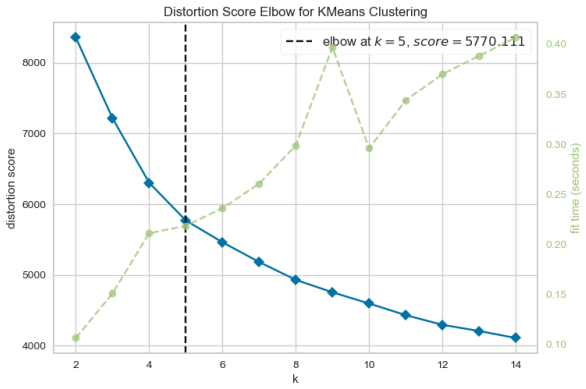


Figure 9: Elbow score for K-Means, computed using the module KElbowVisualizer from the library yellowbrick.cluster

Instead Silhouette Coefficient or Silhouette score is a metric used to calculate the goodness of a clustering technique. Its value ranges from -1 to 1. The positive score indicates that the sample is far away from the neighboring clusters, whereas a negative score identify that the samples might have got assigned to the wrong clusters. To draw the silhouette plots and perform comparative analysis is used YellowBrick, a machine learning visualization library. The optimal number of clusters is chosen by:

- The average of Silhouette scores that for each cluster must be exceeded
- The thickness of the Silhouette plot, representing each cluster, should be more uniform

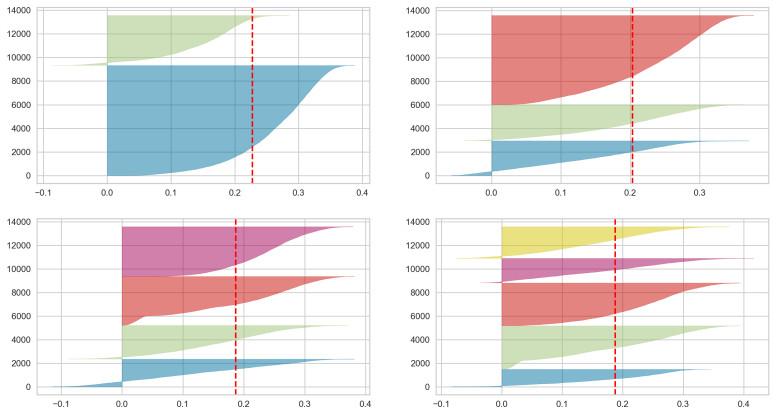
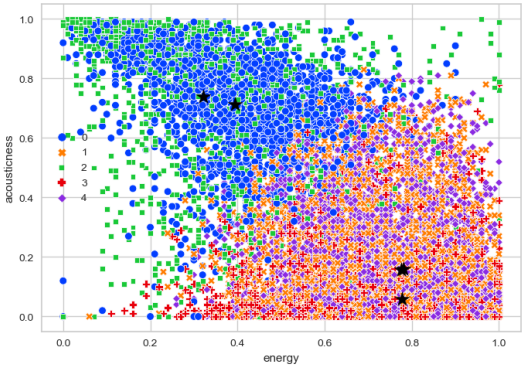


Figure 10: Silhouette plot for K-Means, computed using the module SilhouetteVisualizer of the yellowbrick.cluster library

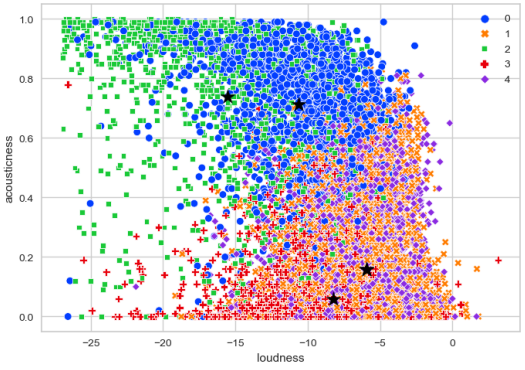
In the Figure 10 are represented Silhouette scores for values of k ranging from 2 to 5, and it can be seen that the optimal number is between 4 and 5, since for k equal to 2 and 3 there is an overshoot of the mean value but the clusters are not uniform. Therefore, the choice falls on the values 4 or 5, but for k=5 more uniformity is seen, moreover with the Elbow technique it was shown as the best score.

Next, the K means algorithm was implemented for the variables that had the positive or negative correlation values in the heatmap in a pronounced manner. The variables that had the most negative correlation are: *energy acousticness* (-0.71), *loudness acousticness* (-0.53) and *instrumentalness loudness* (-0.41).
Figures 11 show the relative clustering of negatively correlated variables. It can be seen that in all the figures although 5 centroids were chosen, only 4 actually appear, except in the Figure 11a, where

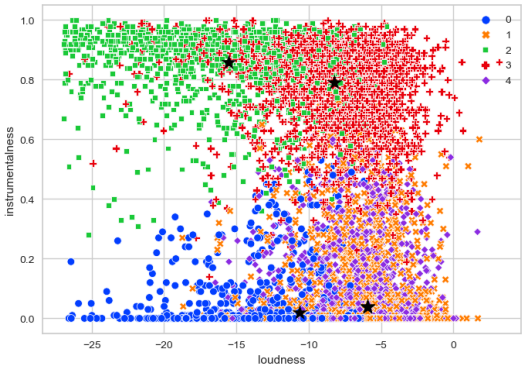
a slight overlap between 2 centroids can be seen. It occurs because k-means is very sensitive to noise, and although the outliers have been removed, a tiny fraction is still present, creating difficulties for the algorithm in identifying centroids. Nevertheless, at least 4 clusters can be identified in all figures.



(a) *Energy and Acousticness*



(b) *Loudness and Acousticness*



(c) *Loudness and Instrumentalness*

Figure 11: Scatterplots that show the distribution of the values belonging to the different clusters, each highlited by a different color. The star symbols represent the centroids.

In Figure 11b related to *acousticness* and *loudness*, a majority of songs can be seen to have very high acousticness and relatively low loudness between -5 and -10, identified by the blue cluster, while a very small fraction of songs and a very small cluster comprising *acousticness* very close to 0 and *loudness* values between -10 and -20 (red cluster). While most of the points are represented by noise.

In Figure 11c with *loudness* and *instrumentalness* there are 2 distinct areas. One (green) cluster containing songs in which there are vocals, indicative of a very high instrumentalness score, that have very low loudness. Another (red) cluster again with tracks in which there are voices but with higher loudness. While the other 2 areas consist mainly of noise.

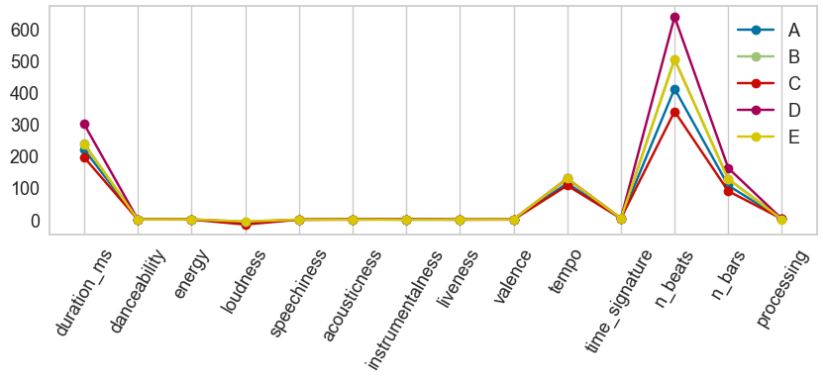


Figure 12: This graph shows the distance between centroids, each represented by a color, for all attributes.

We also researched other variable combinations for scatter plots but they revealed themselves as mostly noisy, so they were not considered useful in order to report the results.
Our choice of number of centroids, as said before, was leaded by the result of Elbow score, and it was five, while the Silhouette test result suggested also four as a valid number. Trying them out, the results were clearer with five centroids, obtaining mostly messy and not very informative graphs in the other case. Figure 12 highlights the fact that centroids are extremely close to each other for most attributes, so they do not allow specific clusters to be identified.
Although the choosing a K equal to five gave us the best result, two of the centroids were often find out very close to each other, generating some inconsistency in the visualization.

3.2 Hierarchical clustering

Hierarchical clustering, unlike KMeans which creates independent clusters, is a method that organizes data in a hierarchical manner, creating nested clusters. There are two main types of hierarchical clustering: agglomerative and divisive. They differ in their starting point (each point as a single cluster or a big cluster containing every point) and the basic step of the algorithm (merging or dividing the clusters). In the agglomerative method the algorithm decides to merge the clusters comparing the inter-cluster distance[12], which can be defined in many ways, such as:

- setting a centroid for the cluster and using it to measure the distance between it and the other centroids
- taking the minimum, the maximum or the average distance in the proximity matrix[13]
- using specific functions

Each method has its own strengths and weaknesses, so it is important to consider these factors when choosing the most appropriate method for analyzing the data. One of the key advantages of Hierarchical clustering is that it does not require an initial selection of the number of clusters, unlike KMeans which requires the selection of centroids. Instead, the number of clusters can be chosen after running the algorithm and analyzing the dendrogram. In our dataset, as shown in Figure 13, the dendrogram[14] graph clearly indicates the presence of two main groups, with one being significantly larger than the other. These groups are clearly separated, as evidenced by the distance between them before merging into the final cluster.

[12]Distance computed between two clusters, which can be obtained with various methods and type of distances, for example euclidean or manhattan distance.

[13]Matrix built assigning all points of one cluster to rows and all points of the other in columns, then filling all cells with the distance between the two points that correspond to row and column of the cell. It is usually used to search the designated distance (like minimum, maximum or average) in order to compute the inter-cluster distance.

[14]Graph that represents the merging or splitting of data during the hierarchical algorithm. It can be used to choose how many clusters are more suitable to investigate the dataset in a graphic way.

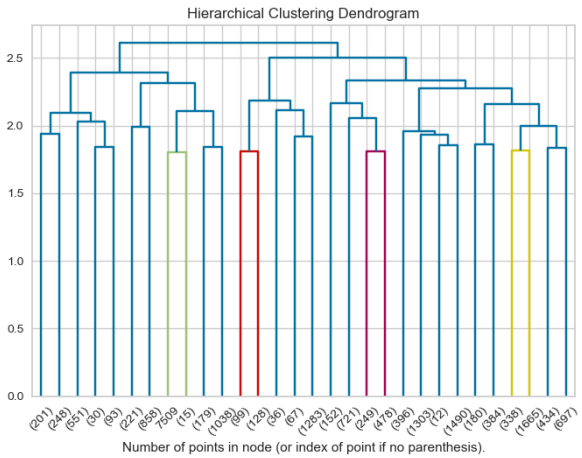
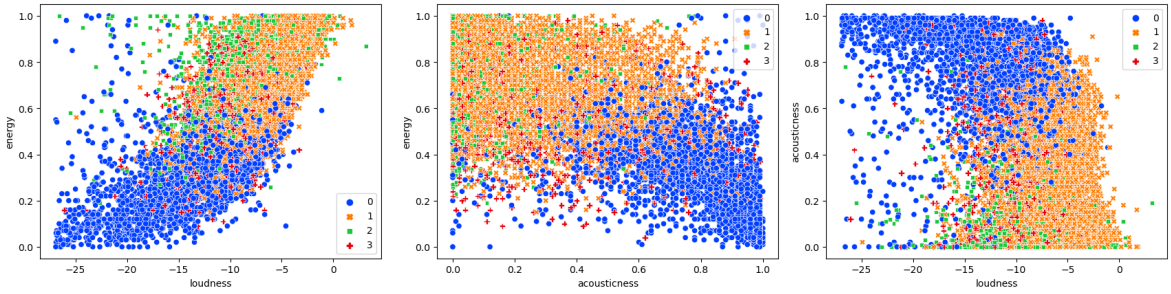


Figure 13: Dendrogram

After examining both the dendrogram and the empirical trials, we decided to categorize the data into four clusters. Similarly to previous observations, Figure 14 displays scatter plots of highly correlated variables, such as *energy* and *loudness* or *acousticness*. Upon inspection, it is evident that the majority of the data falls into one of two main clusters, while the other two contain significantly fewer samples. Despite this, there is a noticeable distinction between the clusters labeled as 0 and 1 in the plots, although it is still somewhat scattered.



(a) *Energy and Loudness* (b) *Energy and Acousticness* (c) *Acousticness and Loudness*

Figure 14: Scatterplots that show the distribution of values belonging to the different clusters, each one highlighted by a different color.

3.3 Density-based clustering

DBSCAN was designed to identify density-based clusters by grouping together points that are close to each other, rather than the entire group. The algorithm produces non-overlapping groups with shapes determined by their internal structure. The main parameters of the algorithm include epsilon[15], the minimum number of samples required for a cluster, and the type of metric used to measure the distance between samples (default is the Euclidean metric). These parameters are used to calculate the neighborhood around each point. If a sample's neighborhood contains equal to or more points than the minimum, it is labeled as a cluster; otherwise, the point is considered noise until it is assigned to another point's cluster. If a point is already part of a group and is recognized as part of another cluster, the two clusters will merge into a single cluster, and this process continues until every point is labeled.

One of the main strengths of this method is its ability to handle noise better than previous algorithms, and it does not require preselecting the number of groups to search for. However, it may struggle with high-dimensional data, such as in our survey. In order to find the best epsilon, we calculated

[15]Epsilon represents the maximum distance between two points in order that the second point is considered part of the cluster of the first point.

the average distance between each point, taking into account its 3^{th} nearest neighbor. The point of maximum slope, known as the "knee", indicates the best epsilon to use. As depicted in Figure 15, the best epsilon is 0.5.

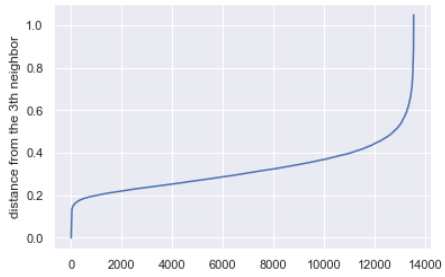
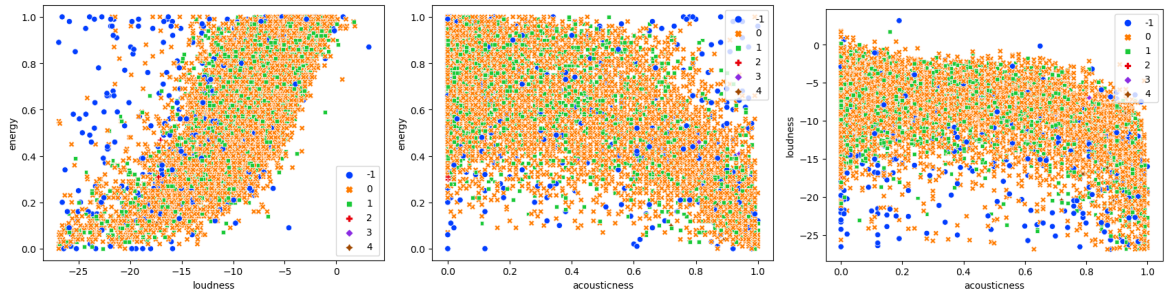


Figure 15: 3^{th} distance plot

When examining the visual representations of the results of density-based clustering, it is challenging to identify informative features. In Figure 16 the points are not clearly separated, with most of them belonging to the orange cluster, while the remaining samples from different clusters are scattered in a disorganized manner. It is possible that the high dimensionality of our dataset hindered the algorithm's ability to accurately label the data, despite its strength in handling noisy data.



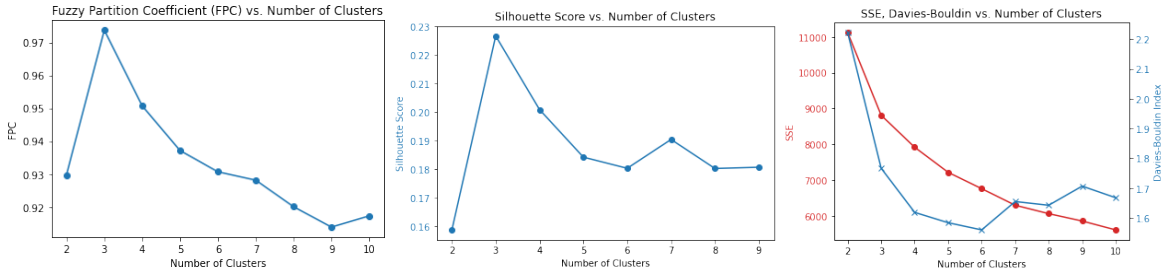
(a) *Energy and Loudness* (b) *Energy and Acousticness* (c) *Loudness and Acousticness*

Figure 16: Scatterplots for the DBSCAN algorithm.

3.4 Fuzzy C-Means

Classical clustering methods typically have well-defined boundaries between clusters. However, in many real-world scenarios, cluster boundaries are not clearly defined, leading to patterns that may belong to more than one cluster. In these cases, Fuzzy clustering methods offer a more effective way to classify these patterns. To determine the number of clusters, the Elbow method is used, with the Fuzzy Partition Coefficient (FPC)[16] being the calculated coefficient to assess the quality of Fuzzy clustering. Additionally, both SSE and Davies-Bouldin index are calculated to provide a more comprehensive evaluation.

[16]Fuzzy Partition Coefficient (FPC) provides an indication of how well-defined the clusters are. A higher FPC generally implies better separation between clusters.



(a) *FPC vs number of clusters* (b) *Silhouette score vs number of clusters* (c) *SSE and Davies-Bouldin index visualized together*

Figure 17: Series of studies on Fuzzy C-Means algorithm with respect of the different measures of assessment

Based on the Silhouette method, the optimal number of clusters is 3 and it results in a lower SSE. Additionally, when considering the Davies-Bouldin and FPC coefficients, 3 clusters is again the optimal choice. The results of Fuzzy C-Means are obtained using a Fuzziness coefficient[17] of 1.1[18] and a value of tolerance[19] of 1^{-e}.

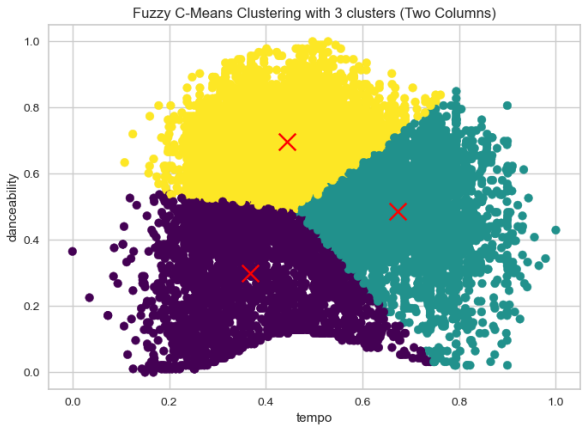


Figure 18: Results for *tempo* and *danceability*.

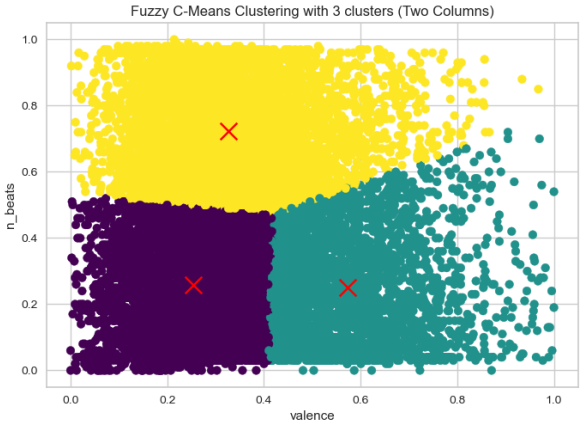


Figure 19: Clustering between the *n_beats* and *valence*.

In Figure 18 the yellow cluster represents the most danceable songs, with tempo values between 0.2 and 0.7. There are also songs with a similar tempo range, but they are less danceable. Additionally, this cluster includes songs with tempo values between 0.0 and 0.2. Another group of songs has intermediate danceability, but with a tempo higher than 0.5.

In Figure 19 it is evident that there are numerous tracks with a valence between 0.0 and 1, so more positive, and with high number of beats. Then there are two clusters with number of beats between 0.0 and 0.5, with the difference on the valence values, since the purple cluster have values between 0.0 and 0.4, instead the green cluster has values higher than 0.4.

From these graphs, it can be observed that the most danceable songs tend to have intermediate tempo values, and that the tracks that have high number of beats tend to be more positive.

[17]The fuzziness coefficient m controls how much each data point is allowed to belong to multiple clusters. A higher m leads to a softer partitioning of the data, where points can belong to more than one cluster with varying degrees of membership.

[18]We also conducted empirical test with m=2 obtaining a lower silhouette and higher SSE.

[19]The tolerance parameter is used to define a threshold for convergence. The algorithm will continue iterating until the change in the objective function between consecutive iterations is below this threshold.

After evaluating various clustering algorithms, we found that Fuzzy C-Means was the most effective in creating visually distinct clusters for the relevant attributes in our research.

3.5 Clustering evaluation

After understanding the different clustering algorithms, it is important to identify the best one. It is possible to compare the Silhouette coefficient, which measures how well the clusters are separated and the best value is one that approaches 1.

Clustering algorithm	Silhouette Coefficient
(Fuzzy) C-Means	0.22
K-Means	0.19
Hierarchical	0.16
DBSCAN	0.15

Table 1: Comparison of clustering algorithms using the Silhouette Coefficient

In Table 1, the varying values of the Silhouette coefficient are presented in relation to the various clustering algorithms implemented. Based on the values, it can be concluded that, for our data, Fuzzy C-means is the best algorithm for data division, as it has the highest value among the 3, while DBSCAN algorithm is the worst one.

We also used the Davies-Bouldin index as an internal evaluation measure for the K-Means, Hierarchical and Fuzzy C-Means algorithms. This index assesses the cohesion and separation of clusters, with a lower value indicating more compact and better separated clusters. As shown in the Table 2, for K-Means the value is 1.60, while for Hierarchical clustering it is 2.34. Overall, the two algorithms show little difference in their ability to create compact internal clusters. Instead Fuzzy C-means is the algorithm that provides visually a more effective way to divide the attributes of the dataset. It's important to note that the dataset contains a significant amount of noise, even after removing outliers, which particularly affects the results in the case of K-Means.

Clustering algorithm	Davies-Bouldin index
K-Means	1.60
(Fuzzy) C-Means	1.77
Hierarchical	2.34

Table 2: Comparison of the clustering algorithm through Davies-Bouldin index

We used K-Means to analyze variables that are negatively correlated, discovering that many songs have high energy but are also quite noisy. Additionally, we identified two clusters, both containing tracks with vocals, but one cluster consists of tracks with high volume while the other consists of tracks with lower volume. DBSCAN showed that there are a significant number of songs with moderate energy levels, ranging from non-acoustic to highly acoustic. Hierarchical clustering helped us to more effectively visualize the division of our dataset, and with Fuzzy C-Means it was possible to deepen the distinction between different clusters, seeing them more compactly and clearly.

4 Classification

In order to investigate the connection between the technical aspects of music and the body responses generated (psychological, emotive and motor), we designed a classification task focusing on the related feature *energy*, having as our objective find to how much the physical related data alone are enough to predict if a song conveys energy or not, comparing that with the same predictions on all the features suitable for classification models. In order to classify *energy*, which has continuous values, we binned the feature in two labels, which we called "High energy", for very energetic songs, and "Low energy", for the others, taking the value 0.5 as a threshold.
The datasets used were stripped of their outliers and had their NaN values, replaced with a prediction

of them, using classification and regression models (see chapter 5 for better information). We also dropped categorical values as *name*, *album_name* or *artists*, while we assigned integer values to *genres*, creating a new column called *genres_val*, in order to use those information for the classification task. Also *popularity confidence* column was dropped due to the inability to find a sufficient regression to describe its values. To be able to compare the tests made with the full dataset with the tries made without the body responses related features, a new copy of the dataset was created, and from that there were dropped the columns: *danceability*, *acousticness*, *valence*. We also dropped *popularity* and *genres_val* because not relevant to our task. We will refer to the first dataset as "Full Dataset" and to the second one as "Task Dataset".
The predictions were made with different classification models, such as KNN, Naïve Bayes and Decision Trees; in order to maximize their performances we used techniques of hyperparameter selection, preferring the ones that use methods like cross-validation or random searches to achieve better results.

4.1 K-nearest neighbors (KNN)

The k-nearest neighbors algorithm, also known as KNN or k-NN, is a supervised model, used both for classification and regression, that uses the notion of proximity to predict the correct classification label or value to associate to the given features.
For our task we built two models, one for the complete dataset and the other for the second one, related to the task; and for each of them we performed a hyperparameter selection in order to find the best number of neighbors, having as a result 15 for the first and 13 for the second one.

Dataset	Class	Precision	Recall	F1-score	Support
Full Dataset	High energy	0.91	0.96	0.93	3318
	Low energy	0.87	0.72	0.79	1189
	Accuracy			0.90	4507
	Macro avg	0.89	0.84	0.86	4507
	Weighted avg	0.90	0.90	0.90	4507
Task Dataset	High energy	0.88	0.94	0.91	3318
	Low energy	0.80	0.63	0.71	1189
	Accuracy			0.86	4507
	Macro avg	0.84	0.79	0.81	4507
	Weighted avg	0.86	0.86	0.86	4507

Table 3: KNN Confusion Matrix

The principal way to assess the goodness of the performance of a model is to calculate metrics, like accuracy. Having to assess the quality and the difference between our models, the principal path to follow was to calculate and analyze the classification report, composed of metrics like Accuracy[20], Precision[21], Recall[22], and F1-score[23].
Looking at Table 3, results for both models, with and without emotional and social related features, are shown in relation of the two labels "High energy" and "Low energy"; also for the Macro Average[24] and the Weighted Average[25] of the model. The analysis for the first dataset reveals generally good

[20]Accuracy is a metric that shows the goodness of the classification skills of a model, in particular how often it classify correctly the items of the Dataset compared to the total classification trials

[21]Precision is a metric which indicates how often the classifications made by a model are appropriate while classifying a single class. It takes in account all the items of that selected class that the model has classified correctly, dividing them by all the items it classified with that same label, whether they are correct or not.

[22]Recall is a metric which measures the model's ability to correctly identify all examples of a given class, comparing the items correctly classified with that label to the total number of items that actually belong to the class, even if they haven't been identified, in the population

[23]F1-score is a metric given by the harmonic mean of precision and recall, two of the metrics named previously. An increase in F1-score indicates improved performance for one of them or both, it's usually used to check at the balance between them

[24]The macro average is a method to aggregate metrics without weighting them regarding the actual dimensionality of the class the metrics was calculated from.

[25]The weighted average is a method used to aggregate metrics taking account of the dimensionality of the different classes and assigning bigger weights to bigger classes in order to reflect their importance on the general performance on the dataset.

performances, having a little difficulties dealing with songs with low energy, but overall achieving a optimum 90% of Accuracy and also a 90% on Recall and Precision looking at the weighted average, showing a general balance displayed also looking at the weighted F1-score. However, looking at both the "Low energy" class result and at the macro average, the drop in the recall value shows that this model tends to assign more "High energy" labels that it should. We'll also see this imbalance in all the following models. This may be caused by the big imbalance between the two classes, since the number of song labelled "high energy" is almost three times bigger than the "low energy" occurrences.
The Task Dataset, deprived of five columns of its, shows this lack in a slight loss of all metrics, however keeping a good performance with a general accuracy of 86%, with only a 4% drop. The prediction gap between "High energy" and "Low energy" shows a significant increase, having a recall for the second label equal to 63% and showing a not little difficulty in detecting the second class. Despite all that, we can say we're pretty satisfied by the general performances, that maintains themselves, in the weighted average, equal to 86% for Precision, Recall and F1-score, proving that, with the k-NN model, it's possible to achieve satisfying results predicting the level of energy using only physical related attributes.

4.2 Naïve Bayes

Naïve Bayes models are probabilistic classifiers, which work using statistical techniques in order to determine the probability of an item to belong to a specific class. These classifiers are based on the Bayes theorem[26] and they're called naïve because their algorithms assume the conditional class independence[27] in order to simplify calculations.

Dataset	Class	Precision	Recall	F1-score	Support
Full dataset	High energy	0.92	0.92	0.92	3318
	Low energy	0.78	0.79	0.78	1189
	Accuracy			0.88	4507
	Macro avg	0.85	0.85	0.85	4507
	Weighted avg	0.88	0.88	0.88	4507
Task dataset	High energy	0.89	0.89	0.89	3318
	Low energy	0.69	0.69	0.69	1189
	Accuracy			0.84	4507
	Macro avg	0.79	0.79	0.79	4507
	Weighted avg	0.84	0.84	0.84	4507

Table 4: Naïve Bayes Confusion Matrix

In Table 4 are shown results for both the models for the complete dataset and the stripped dataset, as was presented before. The Accuracy of the model for the complete dataset is slightly below the one achieved with K-nn, with a value of 88% against its 90%, although maintaining more balanced values on the prediction metrics on the "Low energy" label, dropping the precision to 78% but enhancing the recall to 79%. The same trend is maintained with the macro and Weighed Average of the model, with respectively the repetition of 85% and 88% for Precision, Recall and F1-score, not only, we can also find it looking at the task dataset for both classes and average types.
Reflecting on these results and comparing them to the previous ones, it's to possible to see how the Naïve Bayes model, despite having a subtle decline on accuracy and the general mean of metric values, shows a more balanced behaviour toward our datasets, not leaning toward nor Precision neither Recall for both classes.

[26]The Bayes theorem is a probabilistic theorem that allows to calculate the conditional probability of an event given another correlated event.

[27]The conditional class independence is a concept that implies that the effect of an attribute on a given class is independent from the values of the other attributes. This assumption may simplify computation, but not always it reflects the ground truth.

4.3 Decision Tree

Decision trees are both classification and regression models that work with a hierarchical structure in order to predict one or more, in case of regression, target values, given a certain input. In order to find the best parameters for the models, we decided to rely on a Randomized Search[28], obtaining the following results: for the Full Dataset, 10 minimum samples in each split, 50 minimum samples in each leaf, a maximum depth equal to 9 and Entropy as the best criterion for impurity measurement; while, for the dataset related to our task, the minimum samples per split counted as 30, while 50 was the number per leaf, the maximum depth was set as 8 and the criterion as Gini.

Dataset	Class	Precision	Recall	F1-score	Support
Full Dataset	High energy	0.92	0.93	0.93	3318
	Low energy	0.80	0.78	0.79	1189
	Accuracy			0.89	4507
	Macro avg	0.86	0.86	0.86	4507
	Weighted avg	0.89	0.89	0.89	4507
Task Dataset	High energy	0.88	0.94	0.91	3318
	Low energy	0.80	0.63	0.71	1189
	Accuracy			0.86	4507
	Macro avg	0.84	0.79	0.81	4507
	Weighted avg	0.86	0.86	0.86	4507

Table 5: Decision Tree confusion matrix

The Table 5 shows good results for both models, with Accuracy values of 89% for the Full Dataset and of 86% for the Task Dataset. Both of them are suffering from a difficulty to identify "low energy" labelled song, such as all the previous models, which is a reflection on an unbalance between Precision and Recall metrics; although, for the Full Dataset trained model, this disproportion it's not as significant as the one in K-NN models, as we can see by the equal score of the Macro Average. The Task Dataset model, tracing the K-NN results more than it's counterpart, it's significantly more affected by the misclassification error on songs with low energy values, possibly because the model, lacking five of it's attributes and facing a low frequency class, doesn't have enough data to distinguish between the two of them.

A powerful tool of Decision Trees for data analysis is the possibility to access to feature importance[29] data, which indicate the relative importance of each feature when making a prediction. For the sake of our analysis, here are reported the first six most important features from the two datasets.

Feature	Value
loudness	0.5728
acousticness	0.2672
valence	0.0568
speechiness	0.0258
danceability	0.0188
instrumentalness	0.0128

Table 6: Full Dataset

Feature	Value
loudness	0.7495
speechiness	0.0738
instrumentalness	0.0683
n_beats	0.0608
liveness	0.0142
duration_ms	0.0140

Table 7: Feature Removal Dataset

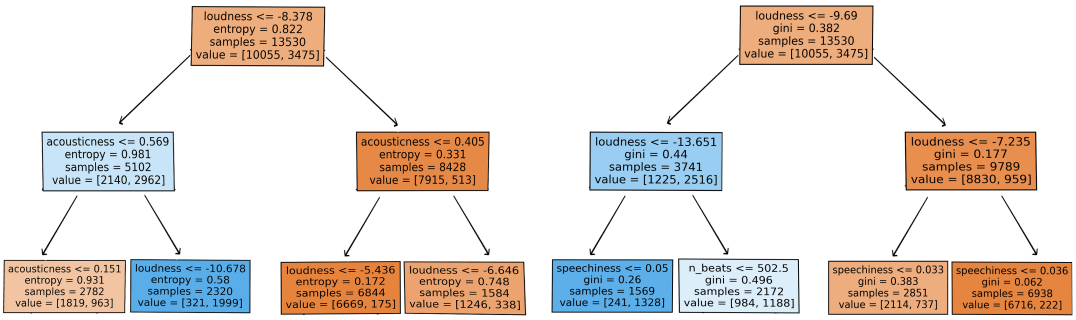
Table 8: Feature importance values comparison

Having *loudness* as most important in both cases is not a surprise, being it correlated to *energy* with a Pearson value of 0.75. The same goes for *acousticness*, in the first model where it's present in the

[28]Randomized Search set up a grid of hyperparameter values and select random combinations to train the model and score. The number of search iterations is set based on time/resources

[29]Feature importance is calculated by measuring the decrease in node impurity, weighted by the probability of reaching that node. This probability can be calculated by dividing the number of samples reaching the node by the total number of samples.

Training Set, which is correlated negatively to the target value with -0.71. In both of the models, the other features than the correlated ones share an importance below 10%. In particular, the unexpected placement of danceability, with a value near 2%, suggests that, for a song to be danceable, being energetic is not a prerogative. This could be because high *danceability* values are present also for songs suited for slow dances, which are usually combined with a low energy.
In the second model the *loudness* importance increases by over 15%, showing that, in absence of some of the most important features of the previous model, it's more needed to determine the label of the target; being also less able to rely on the other features. However, looking at the general performances, it's possible to conclude that the physical related features are pretty able to overcome the lack of the other ones, being able to predict a body response like *energy*, either giving importance to the most correlated attribute or spreading more importance over the remained features.

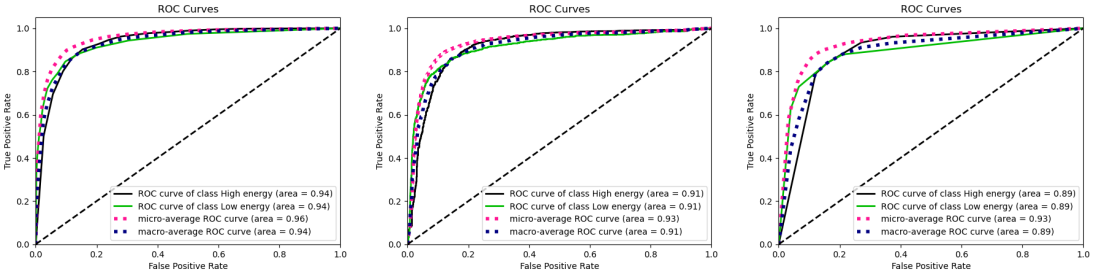


(a) *Full Dataset* (b) *Task Dataset*

Figure 20: Full Dataset and Task Dataset Decision Tree.
The images were extracted by the two tree graphs, with maximum depth of 2 (modified images for better visualization)

The comparison of the trees from the two datasets provides a clearer visual understanding of the most important features. In both cases, *loudness* is seen at the root, indicating its significant role in feature importance. In the model related to the Full Dataset we can see *acousticness*, the second feature for importance, appear after the first split and also in a branch of the third, while, in the model related to the Task Dataset, other features than *loudness* appear only at the third level of depth of tree, being those *speechiness* and the *number of beats*, respectively the second and fourth for importance.

4.4 Performance evaluation



(a) *KNN ROC curve* (b) *Naive Bayes ROC curve* (c) *Decision Tree ROC curve*

Figure 21: ROC curves of classification models on the Full Dataset

ROC curve illustrates the model's performance on the two classes. Its graph also includes the Micro-Average[30] and Macro-Average[31] of the ROC Curve itself. When comparing the relative Roc curves of different classification models, it is evident that the KNN has the best performance, consistent with the previously observed Accuracy results. All the values under the areas for KNN are the same, indicating that the model performs very well on all classes. In comparison, the values under the areas for Naïve Bayes and Decision Tree drop slightly, but the area under the Micro-Average has slightly higher values than the other areas, suggesting that the model performs well in overall classification, considering all samples and classes.

Our analysis of classification revealed that the KNN model was the best in performing with both datasets, as highlighted by the high Accuracy shown in the ROC curves. However, the highest accuracy results were achieved with the Full Dataset. Removing body responses related features from the model resulted in more difficulties in classification, but overall performance only slightly decreased. The majority class in both datasets is High Energy, leading to more optimal results for this class compared to the Low Energy class.
Although Naïve Bayes had less impressive results, it showed more balance between Precision and Recall. Leaving aside the accuracy values, the analysis with Decision Trees was important in identifying the most significant features and how they behaved in predicting results. Features such as *acousticness*, *valence*, and *danceability* were found to be crucial for classification, as they were located close to the root of the tree and with high values of feature importance. This insight helped to explain why models had more difficulty in achieving classification in the second dataset when these features were removed.

5 Regression

Regression models are probabilistic models meant to predict continuous variables, called dependent, given one or more independent features. Between the different models, having a linear or non-linear algorithm splits them in two main groups, differing from the way they build the relationship between dependent and independent variables. In the analysis on this essay, K-NN and Decision Tree algorithms were used as non-linear models, while a simple Linear model[32], a Ridge[33] and a Lasso regression[34] were used as the counterpart.
The same dataset as for the Full classification was used, while *popularity* was chose as target for the univariate regression and *popularity* and *energy* were chose for the multivariate one.

	Univariate task			Multivariate task		
Modello	**R^2**	**MSE**	**MAE**	**R^2**	**MSE**	**MAE**
Linear Regression	0.011	3423.624	18.198	0.101	3388.310	18.616
Ridge	0.011	3424.626	18.198	0.101	3388.310	18.616
Lasso	0.011	3423.620	18.198	0.101	3388.190	18.616
Decision Tree Regressor	0.027	3347.215	17.947	0.121	3313.941	18.380
Knn Regressor	-0.019	3538.708	18.329	0.098	3410.307	18.607

Table 9: Univariate and Multivariate metrics Regression

The first things that stands out in Table 9 are the values of Mean Squared Error[35], which are bigger than three thousands in each model. This occurrence may seem odd, especially having Mean Absolute Errors[36] with values between 17 and 19 for each instance, but can be explained with the definition

[30]The ROC micro-average curve considers all instances and classes as a single class, reflecting the overall prediction performance across all classes and aiming to balance performance across them

[31]The macro-average ROC curve calculates and averages the ROC curves for each class independently. Its position is determined by averaging performance over all classes, regardless of any imbalance in class size

[32]Linear regression is a data analysis technique that predicts the value of unknown data using another related, known data value.

[33]Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where the independent variables are highly correlated

[34]Lasso regression is an analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model.

[35]Mean Squared Error indicates the mean square discrepancy between observed data values and estimated data values

[36]Mean Absolute Error is the average of the absolute differences between forecast and target.

of the two metrics: MAE, using absolute values, treats every instance with the same weight, while MSE, raising to square each value, tends to penalize bigger errors. In our case, this can be interpreted as having models that don't commit a big number of errors, but that, in some cases, predict values significantly far from the original ones.
Looking at the R^2 [37] values, univariate models show results near zero, indicating the poor capability of models of explaining the variance in the dependent variable, up to having negative results for K-NN model, meaning that its performances are worse than a model that just applies the mean as a prediction. In the multivariate case each R-squared value shows an increase between 9% and 10%, maintaining though similar values of MSE and MAE, if not having a slight worsening in MAE values. Those fluctuations can be easily explained by reminding the scaling of the two variables *energy* and *popularity*: respectively from 0 to 1 and from 0 to 100. Absolute measures such as MSE and MAE are little affected by *energy*, whereas the more relative measure R^2 effectively shows the improvement when the second dependent variable is added. The increase in R^2 values suggests that the higher correlation between *energy* and the input attributes, compared to *popularity*, leads to better predictions for energy values and overall model performance. The weak results for *popularity* in univariate models may be due to its low correlation with the independent variables, where the strongest is with *instrumentalness* with a correlation of -0.3.
We used this models at the beginning of the classification task in order to substitute the NaN values in *time signature* and *popularity confidence*. After various try, we found out that the best model for *time signature* was K-NN, with values equal to 0.48 for R^2, 0.163 for MSE and 0.141 for MAE. Unfortunately, we couldn't find a sufficient model to replace *popularity confidence*, probably due to the majority of missing values, having as the best result an R^2 of -0.006 with Lasso regression, so we decided to drop it.

6 Pattern mining

Pattern mining is the process of identifying rules that describe specific patterns within the data. The goal is to find patterns or trends in the data that can be used to make predictions or identify interesting behaviors. Our analysis aims to uncover any interesting patterns among the various variables in the musical dataset.

6.1 Preprocessing

To prepare the data, we categorized the variables into specific ranges. Those between 0 and 1 were grouped according to three ranges (low, medium, high), while those without a specific range were divided according to four ranges (low, medium low, medium high, high). Each variable's resulting interval was then combined with its name (e.g. (94.0, 180.0]_nbars). *Mode* and *genres*, being them already in a categorical format, where treated differently: *genres* column was left uncharged, while *mode*, being in a boolean format with 0 and 1, saw its values renamed as "Minor mode" for 0 and "Major mode" for 1, making explicit their meaning.

6.2 Frequent Itemset

In the initial analysis, we used the Apriori[38] algorithm to extract frequent patterns with a minimum itemset number of 2 and a minimum support of 20%. However, due to our relatively small dataset, the use of Apriori and FP-growth[39] algorithms may not be very significant. We observed that adjusting the support level did not significantly impact the results. Increasing the support led to a decrease in the number of resulting itemsets, as more extremely frequent ones emerged. The results are presented in descending order of support.

[37] R^2 is the index that measures the link between the variability of the data and the correctness of the statistical model used

[38] Apriori algorithm proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database

[39] FP-growth algorithm works by constructing a frequent pattern tree (FP-tree) from the input dataset. The FP-tree is a compressed representation of the dataset that captures the frequency and association information of the items in the data

Itemset	%support
(0.001, 0.21]_liveness, (0.02, 0.1]_speechiness	57.47
(-0.001, 0.33]_instrumentalness, (0.020, 0.1]_speechiness	53.81

Table 10: The most relevant frequent Itemsets

We can see that highly common itemsets consist of *liveness* and *speechiness*, specifically very low *liveness* (indicating songs that have not been performed live) and consistently low speechiness (indicating songs without vocals), with a support of 57%. Additionally, there are songs with minimal instrumental presence and medium-low speechiness, indicating poorly spoken songs with few instruments. The other resulting itemsets showed similar values, which are not relevant to our analysis.

6.3 Maximal Itemset

We found some very interesting results when looking at the maximal itemset, which is again ordered in decreasing order based on support (with a minimum support of 20%).

Itemset	%support
(0.36, 0.67]_dan, (173.28, 325.33]_dur, (-0.001, 0.33]_inst, (0.022, 0.10]_spe	26.20
(0.667, 1.0]_en, '(173.285, 325.33]_dur, '(-0.001, 0.33]_ac, '(0.022, 0.10]_spe)	25.40
(-6.87, 3.156]_lou, '(0.68, 1.0]_en, '(-0.01, 0.33]_ac, '(0.02, 0.10]_spe)	25.14

Table 11: The most relevant Maximal Itemsets

We obtained the most significant results with 26% minimum support for itemsets that include medium danceability, medium-low duration, instrumentalness, and low speechiness. This means that medium danceable songs are those without instruments and with few voices. At 25% support, there are energetic songs with low-medium duration, non-acoustic, and no vocals, which could be suitable for physical activity. Additionally, there are other energetic songs with medium-high volume, non-acoustic, and no vocals, also with approximately the same support. This shows that volume can also be relevant in the energy of a song.

6.4 Association Rules

We experimented with various values for the support variable, revealing both less and more intriguing outcomes. Specifically, certain association rules have surfaced, proving to be potentially discriminative in identifying distinct genres. Furthermore, we observed that the most interesting results are achieved with a higher lift, having conducted our surveys only on rules with a lift over 1, but lower confidence (<25). Searching on items with lower confidence may be needed to investigate on genres, which, being balanced, are present each in only 1/16 of the dataset. The table below illustrates the relevant findings.

consequent	**antecedent**	**%support**	**confidence**	**lift**
study	$(20.63, 173.28]_{\text{dur}}$	4.42	0.21	4.06
chicago-house	$(0.67, 0.98]_{\text{dan}}$, $(-0.0001, 0.33]_{\text{ac}}$	4.29	0.19	3.59
black-metal	$(-0.0001, 0.332]_{\text{val}}, (-0.0001, 0.33]_{\text{ac}})$	4.11	0.17	3.43
forro	$(0.66, 0.99]_{\text{val}}, (-0.001, 0.33]_{\text{inst}})$	3.61	0.17	3.34
j-idol	$(-6.86, 3.15]_{\text{lou}}, (0.66, 1.0]_{\text{en}}, (-0.0001, 0.33]_{\text{ac}}$	3.82	0.18	3.31

Table 12: Association rules relating to the genres.

From this table, we can infer that a song listened to for studying purposes (study) typically has a shorter duration. This suggests the possibility that this type of music is played in a 'loop,' and we could speculate that is caused to its aim to enhance concentration. The genre 'chicago-house' as expected, is characterized by very high danceability (closer to 1) and significantly lower acousticness. This aligns with our anticipation, given that this genre is often produced using electronic devices and aims to provide a lively and entertaining experience. In the third row of the table, it's intriguing to highlight the rules associated with 'black-metal,' revealing a notably lower valence. This is inherently

linked to the nature of this music genre, which often conveys darker emotions and intensity through aggressive vocals. In contrast, when it comes to 'forro' songs, there is an opposite situation, with a higher valence (very close to 1). This highlights the genre's ability to convey positive feelings and emotions. The last genre, 'j-idol,' is associated with higher energy (most elevated in the bin), increased loudness, and very low acousticness. This genre is evidently inspired by 'j-pop,' representing a distinct japanese subcategory of pop music.
We have also uncovered additional clues into the nature of the *processing* attribute. Specifically, our findings corroborate the assertions made in the initial correlation studies, revealing a connection between *processing* and the *key* attribute. Additionally, a relationship with the medium high values of *key* is uncovered, along with a new association with lower values of *speechiness*. This suggests that medium high *processing* is likely linked to a higher absence of vocals, suggesting that its meaning is related to the electronic treatment of house and techno songs.

consequent	**antecedent**	**%support**	**confidence**	**lift**
processing	$(2.408, 3.237]_{\text{pro}}, (5.5, 8.25]_{\text{k}}, (0.0225, 0.103]_{\text{spe}})$	4.81	0.23	3.92

Table 13: Association rules relating to processing.

7 Conclusions

In our essay we performed all the necessary steps in order to extract information from a dataset following KDD process, starting from data understanding, preparation (chapter 2) and preprocess its data, replacing missing values and removing outliers. Our research sought to uncover potential robust correlations between the physical characteristics of a song and the diverse human responses it elicits, encompassing motor, psychological, and emotional reactions. The preliminary statistical analysis disclosed a notable insight: songs with higher loudness levels are perceived as having greater energy. In chapter 3 we tried various clustering algorithms, after which we moved on classification on *energy* attribute with three models (chapter 4). As our next step we observed several regression tasks (chapter 5) and we repeated the analysis on missing values addressing the new insights we learned experimenting with new models. Finally, with pattern mining (chapter 6), we identified some possible association rules between genres and physical and psychological features of songs.
During clustering we witnessed the creation of clustering between energy, acousticness and loudness; associations we lately found as recurrent in classification and pattern mining. Intriguing clusters are identified using C-Means, with the observation that the intermediate *tempo* is associated with the songs that most make us dance (*danceability*), and that the tracks with high number of beats tend to convey more positiveness (*valence*). The classification done on a dataset without emotionally and psychologically related features displayed us that physical related features of a track, altogether, were able to overcome their absence. In conclusion, on the basis of our results, it's possible to answer our initial question, suggesting that it is conceivable for physical features of a song to influence the body responses while listening it, even though it may not be the primary factor.

8 References

1. Andrea Minini, "Introduzione al Clustering con l'Algoritmo K-Means", `https://www.andreaminini.com/ai/machine-learning/clustering`
2. Andrea Provino, "K-Means Clustering - Andrea Provino", `https://www.andreaprovino.it/k-means-clustering#`
3. Yellowbrick Documentation, "Elbow Method", `https://www.scikit-yb.org/en/latest/api/cluster/elbow.html`
4. Scikit-Learn Documentation, "K-means Silhouette Analysis", `https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html`

5. MathWorks Documentation, "Davies-Bouldin Index",
https://www.mathworks.com/help/stats/clustering.evaluation.daviesbouldinevaluation.html

6. Scikit-Learn Documentation, "sklearn.metrics.davies_bouldin_score",
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.davies_bouldin_score.html

7. Andrea Minini, "Metriche dei Regressori",
https://www.andreaminini.com/ai/machine-learning/metriche-dei-regressori

8. Laura Ferreri, et al., "How Music Can Influence the Body: Perspectives From Current Research",
https://www.researchgate.net/publication/304711031_How_Music_Can_Influence_the_Body_Perspectives_From_Current_Research