Analysis of the RAVDESS dataset using data mining techniques

Andrea Napolitano,* Claudio De Martino,† and Steffania Sierra‡

Introduction

This documents contains the first two stages in the analysis of the RAVDESS data set. Firstly, the data set is cleaned and prepared for the second part that corresponds to the performance of clustering algorithms. The data cleaning is elaborated in several subtasks like variable transformation, correlation analysis and visualization. Finally, the algorithms used in the second part are K-Means, Bisecting K-means, DBScan, Optics and hierarchical clustering with different parameters.

1 Data Understanding and Data Preparation

1.1 Attribute understanding

The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) used in this report has 38 variables and 2452 data objects. The data set collect the information of 24 actors identified from 1 to 24, being 12 men and 12 women, represented in the variable sex with the letters M and F. Each actor, identified in the dataset from 01 to 24, repeats two times one of the *statements*, *Kids are talking by the door* or *Dogs are sitting by the door*.

The *modality* correspond to the media file types, full-AV, video-only, audio-only. The *vocal_channel* has two options speech or song. The domain of *emotion* is the set of values neutral, calm, happy, sad, angry, fearful, disgust and surprised. The *emotional_intensity* can be normal or strong, with no strong intensity for the neutral emotion.

An *audio_channel* is the path via a signal or data is delivered, i.e, it is where a sound signal is conveyed from the player source to the speaker. For one channel we talk about *mono*, and several channels we refer to *stereo*. A sample of an audio is a portion of an audio's wave in certain interval, but we will also refer to an audio file as a sample. The length of each sample in milliseconds is recorded in the variable *length_ms*. The *sample_width* is the size of an individual sample and indicates how many bits of information a sample contains. This is one important factor in the quality/resolution of the audio. We write 1 or 2 if the size is 8-bits or 16-bits, respectively.

An audio frame is a data record that contains the samples of all the channels available in an audio signal to the same point in time. The *frame_rate* is the number of frames per second expressed in Hertz, and the *frame_width* is the number of bytes for each frame. Since one frame contains a sample for each channel, it can be computed as *channels* $\times$ *sample_width*. The number of frames of the sample is the *frame_count*, and can be computed as *frame_rate* $\times$ *length_ms*$\div$1000 . The *intensity* is the loudness in dBFS.

The zero-crossing rate is the frequency at which a signal changes from positive to zero or negative, and vice versa. It is a measure of the smoothness of the signal. The *zero_crossing_sum* is calculated as *zero_crossing rate* $\times$ *length* $\div$1000.

The original audio signal is represented as a wave function, therefore we can obtain the mean (*mean*),

*a.napolitano12@studenti.unipi.it
†c.demartino1@studenti.unipi.it
‡s.sierragalvis@studenti.unipi.it

standard deviation (*std*), minimum (*min*) and maximum (*max*) values, kurtosis (*kur*) and skewness (*skew*) of the function.

The Mel-Frequency Cepstral Coefficients is a set of features which represent the shape of a spectral envelope. From this set we can obtain the mean and standard deviation of the values, minimum and maximum value. The names of these variables are *mfcc_mean, mfcc_std, mfcc_min, mfcc_max* respectively.

Spectral Centroid is a good predictor of the brigthness of a sound. It is computed for every frame of the sample, so each audio sample has a set of values of spectral centroids. So, the *sc_mean, sc_std, sc_min, sc_max, sc_kur, sc_skew* represent the statistics associate to the spectral centroid.

Finally, the short-time Fourier Transform chromagram (stft) value of an audio represents the intensity of the twelve distinctive pitch classes that are used to study music, and the *sc_mean, sc_std, sc_min, sc_max, sc_kur, sc_skew* represent the corresponding statistics of the stft.

After understand the meaning of each variable we are able to find the characteristics of each feature. The data set has both types of variables, categorical and numerical, and all the numerical variables are continuous. Table 1 shows in detail the classification of the variables.

Attribute type	Feature	Classification (domain)
Nominal	sex, statement, repetition, channel, vocal channel, emotional intensity sample width, frame width	Binary
	actor, emotion, modality	Discrete
Ratio	mean, std, min, max, kur, skew mfcc_ mean, std, min, max sc_ mean, std, min, max, kur, skew stft_ mean, std, min, max, kur, skew	Continuous

Table 1: Classification of the dataset's features.

Using Table 1 and the *nunique* function in Python, we notice the existence of attributes with one only distinct value. So, we decide to delete them: modality, sample_width, frame_rate, stft_max.

1.2 Distribution of the variables and statistics

With the attribute classification done, we can continue exploring the data set. In this case we try to obtain some insights from the data through visualizations and statistics. We start analyzing variables individually. The two values that *repetition* and *statement* take are equally distributed. In figure 1 we can see that *actor's* distribution is unimodal skewed as *emotional_intensity* and *vocal_channel*.

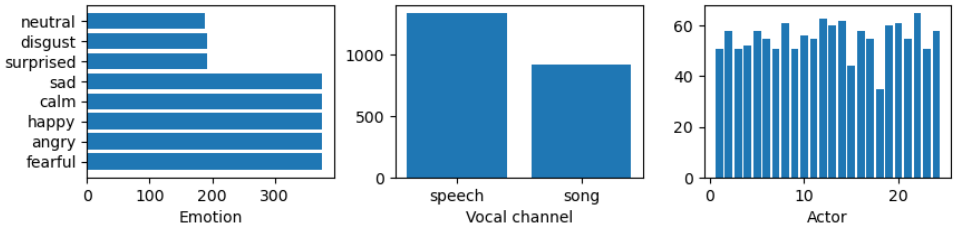


Figure 1: Distribution of the nominal variables

To observe the distribution of the numerical variables, we use density plots, and skewness and kurtosis measures. The features *intensity, sc_mean, sc_max, sc_skew, stft_mean, stft_skew, stft_max, skew* and all the *mfcc_* variables have a fairly symmetric distribution, what means that the absolute value of their skewness values is less than 0.5, and the absolute value of the kurtosis is close to zero indicating thiner tails and lower and broader peaks as shown in Figure 2. On the other hand, *length_ms, zero_crossing_sum, sc_std and stft_std* are moderately skewed with a similar shape in terms of the peaks and the tails than

the variables mentioned before. Finally, the variables with highly skew distribution are *frame_width, frame_count, sc_kur, stft_min, stft_kur, mean, std, min, max* and *kur*, with skewness value greater than 1, and really high values of kurtosis, which explain the sharper peaks and long tails.

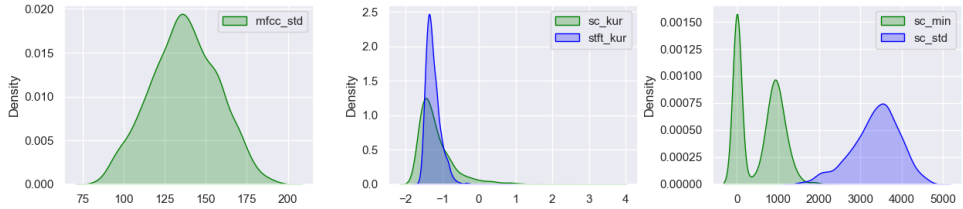


Figure 2: Distribution of the numerical variables

After exploring single variables we would like to have some notion of the distribution of pairs of variables. Some of the distributions that are intuitive because of the meaning of the variables are the relation between emotional intensity and intensity, where we can notice that the mean intensity of strong emotional intensity is higher than the mean of normal intensity, also, because we can compute the frame count from using the length, we can see a linear relation between these two features. Finally, Figure 3, shows that the mean of the intensity of fearful and happy emotions is almost the same, while angry and calm emotions have the highest and lowest mean intensity, respectively. Finally, in the same figure we can see that the mean of the stft coefficients is higher for men than for women.

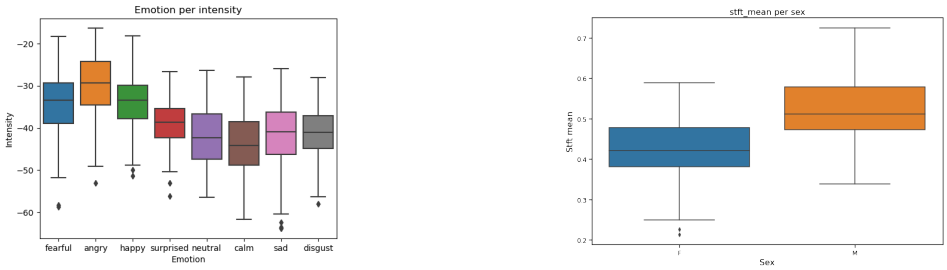


Figure 3: Distribution of pair of variables

To conclude this subsection we look at the statistics of the variables. Figure 4 shows the statistics of the categorical features, and Figure 5 shows the statistics of the numerical variables.

	modality	vocal_channel	emotion	emotional_intensity	statement	repetition	sex
count	2452	2256	2452	2452	2452	2452	2452
unique	1	2	8	2	2	2	2
top	audio-only	speech	fearful	normal	Dogs are sitting by the door	2nd	M
freq	2452	1335	376	1320	1226	1226	1248

Figure 4: Statistics of the categorical features.

1.3 Data quality

To check the data quality of the data set we start checking the semantic inconsistencies with the *value_counts()* method. As result, we find that all nominal variables values are syntactically accurate.

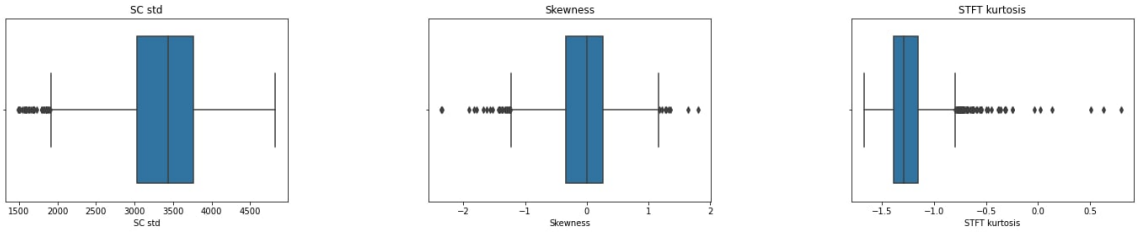
The *isna()* method displays that there are three variables with missing values. Being *actor* the variable with the most quantity on NaN values, 1126, followed by *intensity* with 816 and 196 of *vocal_channel*. In a first try the missing values of intensity were replaced with the mean of the intensity corresponding

	count	mean	std	min	25%	50%	75%	max
intensity	1636.0	-37.625332	8.451982	-63.864613	-43.539869	-37.072745	-31.591309	-16.353953
mfcc_mean	2452.0	-28.769180	4.461886	-43.812923	-31.828597	-28.681109	-25.550238	-15.491450
mfcc_max	2452.0	199.182514	26.002107	126.250810	180.081417	201.697175	218.185288	280.173700
sc_mean	2452.0	5170.101398	875.185444	2360.880942	4563.684781	5122.712259	5775.959809	7655.335726
sc_std	2452.0	3365.453393	580.479034	1491.341071	3025.431971	3433.835368	3768.503344	4819.783069
sc_min	2452.0	551.834124	508.025890	0.000000	0.000000	707.319256	977.693852	2121.417965
sc_max	2452.0	11830.461864	1004.955976	7657.495158	11516.034429	12000.292653	12091.886054	17477.540047
sc_kur	2452.0	-1.142642	0.572654	-1.795576	-1.496187	-1.308938	-0.982944	3.657953
sc_skew	2452.0	0.348442	0.353005	-0.510390	0.098549	0.347621	0.557427	1.825436
stft_mean	2452.0	0.475846	0.082551	0.214089	0.415260	0.475740	0.530571	0.724077
stft_std	2452.0	0.331371	0.023773	0.210126	0.317780	0.334224	0.349272	0.391928
stft_min	2452.0	0.002272	0.004830	0.000000	0.000000	0.000190	0.001999	0.039378
stft_kur	2452.0	-1.247929	0.211781	-1.669603	-1.390612	-1.292105	-1.152519	0.794669
kur	2452.0	11.203002	6.614859	1.757794	6.519988	9.828686	14.085435	59.085695
skew	2452.0	-0.048245	0.454925	-2.356526	-0.336624	0.004256	0.262978	1.799676
emotion_positivity	2452.0	-0.306688	0.822315	-1.000000	-1.000000	-1.000000	0.000000	1.000000
length	2452.0	4.092151	0.598322	2.936000	3.604000	4.004000	4.538000	6.373000
zero_crossings_rate	2452.0	3167.666760	851.941008	1159.781288	2592.111116	3040.051311	3646.926511	8130.602145

Figure 5: Statistics of the numerical variables.

to normal and strong emotional intensity. However, when some of clustering algorithms were applies, we discover that this was not the best decision, and therefore, we decided to undo the replacement. Approximately 20 NaN values of vocal channel were replaced using the information provided by the data set. Speech channel includes surprised and disgust expressions, while song does not. Finally, almost half of the values of actor are missing, so to first glance it seems better to eliminate the variable, but this will be done just when we start to use the data on some algorithms.

To find outliers we use the interquartile range (IQR) method, that classifies as outliers the variables that are above and below the first and last quartile minus 1.5 times the IQR, rspectively. Graphically this can be seen using box plots as in Figure 6. In this exploration of outliers we found that almost half of the values *sc_min* and *stft_min* are zero which makes us to think that these zeros values can be considered as a type of outlier. Therefore when we will be using the clustering algorithms for pairs of variables when at least one of them is *sc_min* or *stft_min*, we will remove the zeros, because they don't provide additional information to the plots.



(a) (b) (c)

Figure 6: Boxplots showing the outliers of the sc_std, skewness and stft_kur variables.

1.4 Variable transformation

In order to understand what variables need to be normalized, we looked for the first and third quartile for each variables. In this way we understood what variables are too big or too small compared to other ones. However, in this step of the data cleaning we don't do any normalization. This is done just before use any of the clustering algorithms. Finally, in this phase we modified some of the variables. First, we change the measure unit of length to seconds, instead of milliseconds, this is because the standard measures related with time are given in seconds, and finally, since it is easier to understand the zero

crossing rate instead of the zero crossing sum, we remove this column and replace it for *zero_crossing_rate* that is equals to *zero_crossing_sum / length*. Additionally, we decided to add a new variable based on the *emotional_intensity*. This variable takes only three values: -1, 0 and 1. With -1 for fearful, angry, sad and disgust values, 0 for the neutral and calm, and 1 for the remain ones.

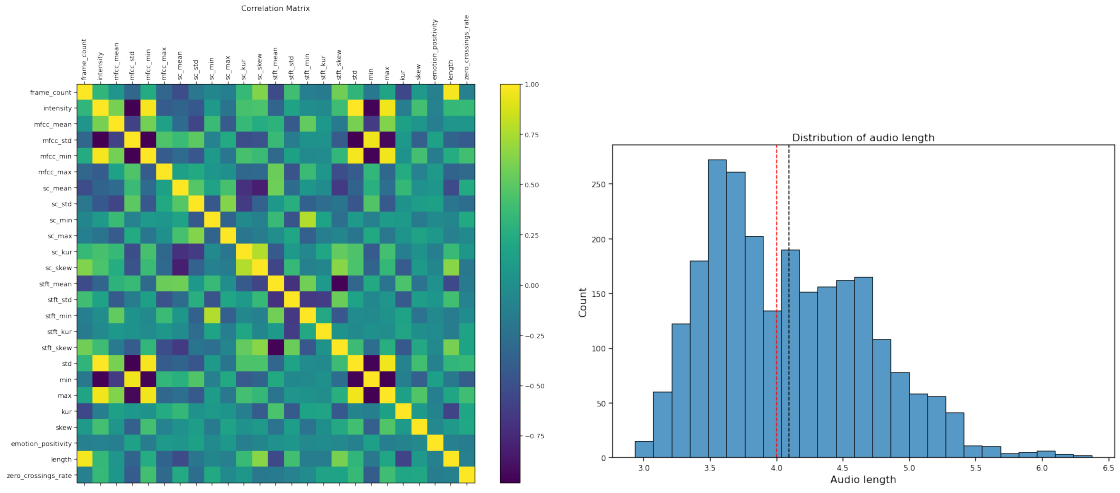
1.5 Correlation and elimination of variables

The idea of clustering is to find clusters that cannot be deduced at first glance from the relation between variables. This means that is necessary to remove from our data set the high correlated variables. From the data exploration of pairs of variables done in the subsection 1.2 and using the correlation matrix with the Spearman coefficient 7a we found that the following pairs of variables with an absolute value of the correlation greater than 0.9.

CORRELATION > 0.9 :
frame_count & length 0.97
intensity & mfcc_min 0.97
intensity & std 0.99
intensity & max 0.96
mfcc_std & min 0.95
mfcc_min & std 0.97
mfcc_min & max 0.96
max & std 0.95

CORRELATION < - 0.9 :
mfcc_std & intensity -0.98
mfcc_std & mfcc_min -0.98
mfcc_std & std -0.98
mfcc_std & max -0.94
mfcc_min & min -0.97
stft_mean & stft_skew -0.97
std & min -0.96
min & intensity -0.96
min & max -0.98

With these results we decide to remove the columns frame_count, frame_width, mfcc_min, mfcc_std, stft_skew, min, max, std.



(a) (b)

Figure 7: (a) Correlation Heat Map. (b) Length distribution, we can notice here that there are two spikes that are divided by the value 4

2 Clustering Analysis

In this section we analyze the dataset using the three types of clustering algorithms, centroid, density and hierarchical-based methods. As mentioned in subsection 1.4, it is in this part of the analysis were the transformation of variables is done. The first time we tried to apply the clustering methods we did it without performing any division of the database producing useless results, i.e., trivial clusters. After

reconsidering the analysis strategy, the team considered that the best approach was to proceed with different partitions. Therefore, in each subsection a different partition is considered for each algorithm and for the common one that is emotional intensity, we will perform each of the algorithms to analyze length and intensity, this is based on the fact that we found, during some test with hierarchical clustering, two well separated clusters.

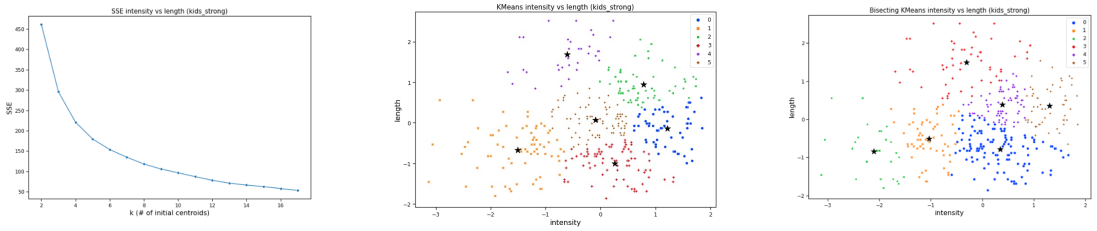
2.1 Centroid-based methods

2.2 Pre-processing and partitioning

For centroid-based methods we define a function that draw three plots: the SSE/K plot, which can help us to estimate the number of clusters and to better understanding the choices made by us, the KMeans plot and the Bisecting KMeans plot. Then, the function can be invoked to draw these plots between two attributes. We could have divided the choice of the number of clusters into two variables, but the goal is to compare the graphs produced using both methods, so we decided to assume only one number of clusters for both. After that, we split the dataset twice: first time with statement Dogs are sitting by the door and Kids are talking by the door, represented by 0 and 1 respectively, and second time with *emotional_intensity* values, which are normal and strong. In this way we get four partitions. In the function that divide the dataset we also normalize the data and eliminate the missing values. Also, before performing KMeans, we remove the outliers to which KMeans is sensitive, by calling a dedicated function, named *clean_outliers()*. After an exploration of dataset we decide to take in analysis the couple of variables composed by intensity and length.

2.3 KMeans and Bisecting KMeans

We skip the first and second partition and focus on the last two partitions. The third partition is with emotional_intensity = "strong" and the statement Kids are talking by the door.

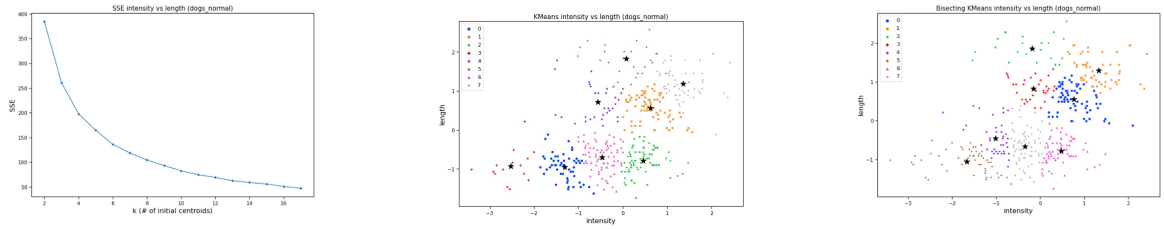


(a) (b) (c)

Figure 8: Clusters obtained after applying Kmeans and Bisecting Kmean algorithms.

From the SSE graph in Figure 8a this time we see that it is from $k \geq 6$ that the trend becomes more linear. Partitioning for Kids are talking by the door and emotional_intensity = "strong" we see that both the distribution of values on the axis x and the absence of correlation between the two attributes with respect to the previous partition have remained unchanged. The values of these graphs are slightly more scattered than the previous ones, with most of the values distributed on the X axis between -2.5 and 1.5 on the Y axis between -1.5 and 2. There are also notable differences in clustering between standard KMeans and Bisecting KMeans, as almost all clusters are different the blue clusters of standard KMeans in Figure 8b is absorbed by the clusters around in Bisecting KMeans in Figure 8c. The fourth partition is with emotional_intensity = "normal" and the statement Dogs are sitting by the door.

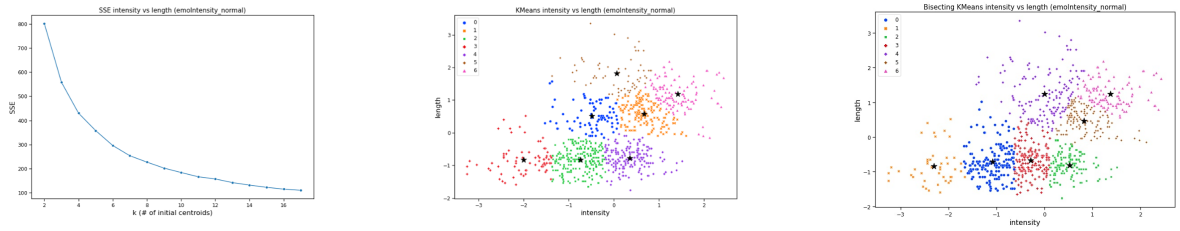
For the number of clusters, eight was chosen because from the SSE graph in Figure 9a we see that the curve begins to flatten with $6 \leq k \geq 10$ and $k = 8$ is the median of this range of values. We can see that the weak correlation between the two attributes present in the first partition returns and makes us assume that this dependence is given by the partition emotional_intensity = "normal" and therefore it is independent of the type of statement. Note that Bisecting KMeans distinguish very well low clusters from



(a) (b) (c)

Figure 9: Clusters obtained after applying Kmeans and Bisecting Kmean algorithms.

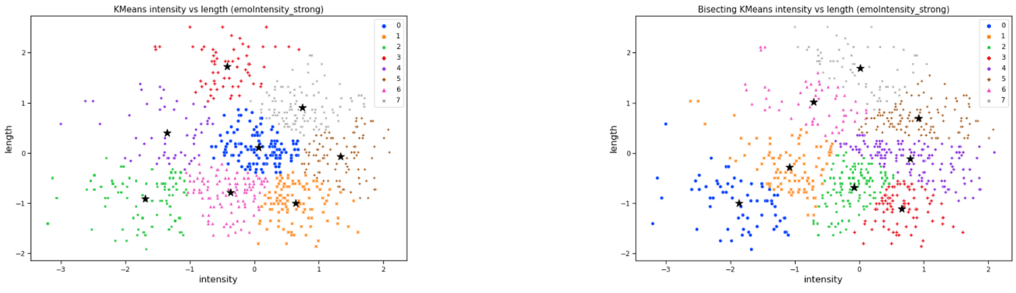
high clusters, you can see four clusters for low intensity and length and other four clusters for positive intensity and length in Figure 9c, while in standard KMeans there is a brown cluster in Figure 9b.



(a) (b) (c)

Figure 10: Clusters obtained after applying Kmeans and Bisecting Kmeans algorithms

Partitioning the data set by by emotional intensity, seven clusters were chosen according to the SSE graph, since for $k \geq 7$ the trend of the curve becomes more linear as in Figure 10a. It can be seen that Bisecting KMeans clearly distinguishes the lower clusters in the graph, that is those with negative intensity and length values, from the higher ones, that is those with positive values of both attributes as in Figure 10c. While the KMeans standard performs a more "gradual" clustering, with two "intermediate" clusters in the center of graph. The shape of the scatter plot is similar to those seen in partitions with emotional_intensity = "normal" as in Figure 10b.



(a) (b)

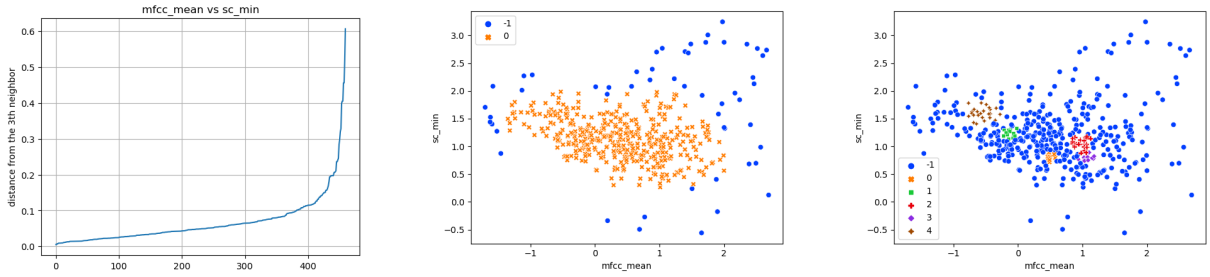
Figure 11: Clusters obtained after applying Kmeans and Bisecting Kmeans algorithms

The same goes for the previous partition, in emotional_intensity = "strong" we find more or less the shape we observed in the corresponding partitions with the statements. From here we have the proof that the shape and, consequently, the presence of correlations between the attributes doesn't depend on the statement but on the value of emotional_intensity. Note that also here Bisecting KMeans tends to distinguish more clearly clusters between negative and positive values for both attributes (Figure 11b), while the KMeans standard doesn't make this distinction very well and it makes a more "gradual" clustering for both attributes (Figure 11a).

2.4 Density-based clustering

In this subsection we will use the DBScan and OPTICS algorithms. The database is divided in three different ways: by sex, by sex and emotion positivity, and by emotional intensity. DBScan algorithms requires two parameters, one is the ratio (eps) of the neighborhood around each point, and the second one is the minimum quantity of points (MinP) that should contain every neighborhood. These parameters are chose using the $k-$ neighbor distance curve. For every partition we plot first the kth neighbor distance using $k = 3$ because we will analyze only pairs of variables. After this, with the information from the curve we use a function that computes the optimal -based on the highest silhouette score- eps and MinP iterating between 3 and 20 as range for MinP with a step of 4 points and the interval of distance containing the point were there is a drastic change of the distance.

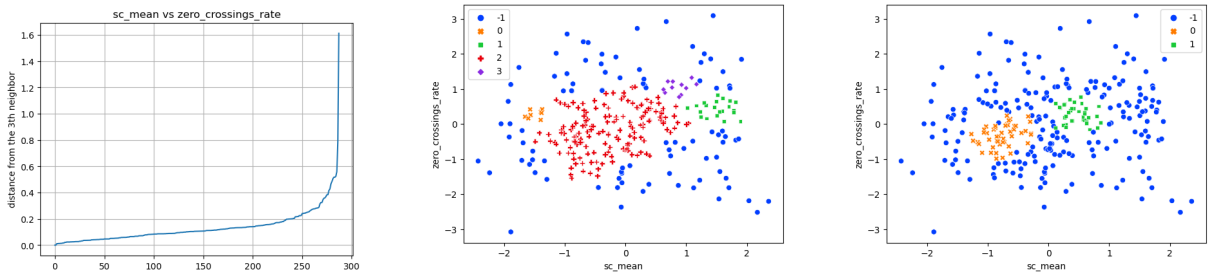
For instance, when the dataset is partitioned by sex, most of the clusters obtained using DBScan algorithm were similar to Figure 12b, where only two clusters are distinguished, with the blue points considered as noise points or outliers. In the particular case of 12, we are working with the dataset that contain only the information of the female actresses. Figure 12c shows the clusters obtained using OPTICS algorithm with MinP=15. We can see that the clusters obtained using both algorithms is completely different, and moreover, applying OPTICS doesn't gives better result.



(a) (b) (c)

Figure 12: Clusters obtained after applying DBScan and OPTICS algorithms splitting the dataset by sex.

Secondly, we consider the partition by sex and emotional positivity. In Figure 13, we analyze the variables *sc_mean* and *zero_crossing_rate* of the dataset containing the values corresponding to women, and 1 for emotional positivity. Using the k- neighbor distance curve we found that the optimal parameters are eps=0.94 and MinP=7. With these parameters, the DBScan algorithm yielded four clusters, with three of them more or less of the same size, while the OPTICS algorithm produced only two clusters and the noise points dominating the classification.

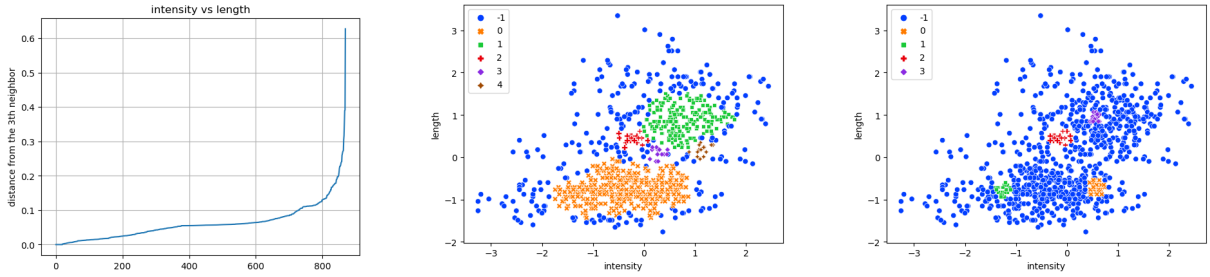


(a) (b) (c)

Figure 13: Clusters obtained after applying DBScan and OPTICS algorithms spliting the dataset by sex and emotional positivity

Finally, with the partition by emotional intensity analyzing the length and *intensity* in the dataset containing the objects of strong intensity we obtain better clusters using DBScan as shows Figure 14b,

where the clusters are well differentiate. However, OPTICS still does not give better results than DBScan, Figure 14c, because the clusters are not well defined, and most points are labeled as noise points.



(a) (b) (c)

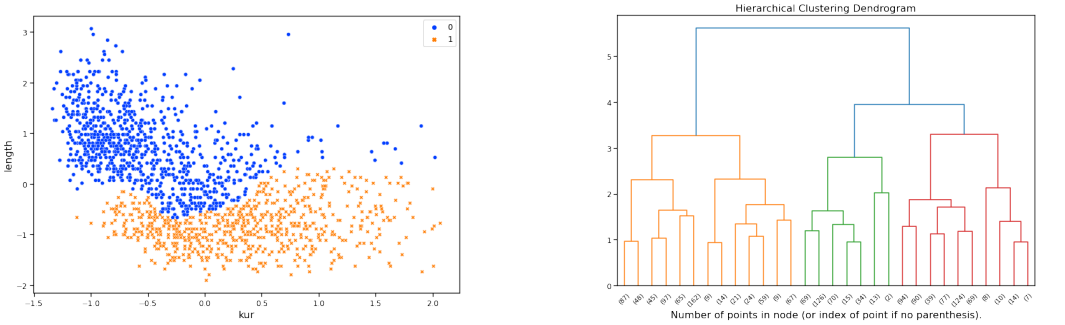
Figure 14: Clusters obtained after applying DBScan and OPTICS algorithms splitting the dataset by emotional intensity

2.5 Hierarchical clustering

As a criterion for choosing the interesting clusters[1] produced by this algorithm, we have looked through all the clusters produced for each pair of attributes, varying the types of linkages that can be achieved and keeping the threshold at a value that generates an average of two or three clusters. In this way we have observed, by looking at the scatter plot visualizations[2], that the "single" type linkage does not produce any kind of interesting result regardless of the threshold set or the number of clusters chosen. A similar result was obtained for the "ward" type linkage, which returned some clusters but nothing of interest. It happened differently using the "complete" and "average" type linkages, which provided us with many noteworthy clusters.

2.5.1 No Partition

Using the entire unpartitioned dataset, we obtained a single attribute pair that gave us two clusters with a particular shape, in fact even from the dendrogram it can be deduced that these two clusters are quite distinct.



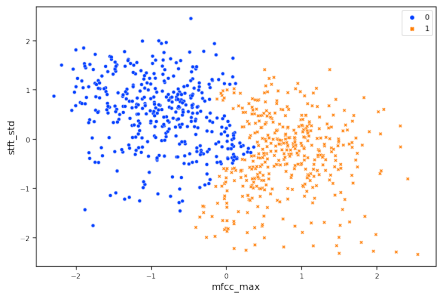
(a) Linkage: Complete, Threshold: 4.5 (b) Dendrogram of the clusters in picture (a)

Figure 15

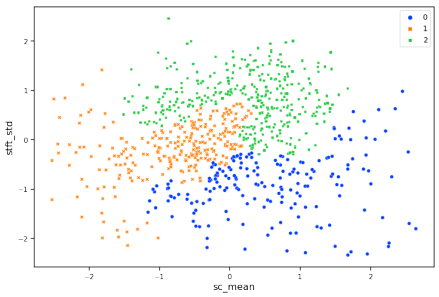
[1] We define "interesting" clusters that have a particular pattern or that are well separated from each other
[2] We didn't look at the dendrograms because they didn't provide us useful information by only looking at them

2.5.2 Length Partition

Fig. 7b shows a clear division in two parts of the distribution of the attribute "length" by the value 4, so we decide to analyze a partition using them. In graph 16a we can see that a first cluster is dominantly in the negative part of the horizontal axis, while the second one is in the positive side of the horizontal axis, so they form a pseudo-symmetry. Graph 16b, on the other hand, has a higher density of points in the center with two other clusters around it: one more dense than the other two.



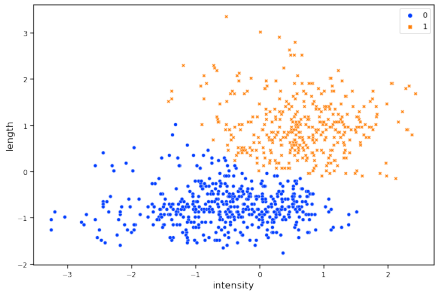
(a) Partition: Length $\geq$ 4, Linkage: Complete, Threshold: 4.5



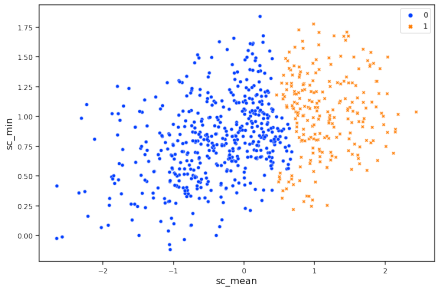
(b) Partition: Length $\geq$ 4, Linkage: Complete, Threshold: 4.5

Figure 16

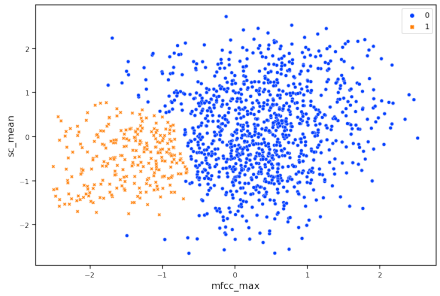
2.5.3 Emotional Intensity Partition



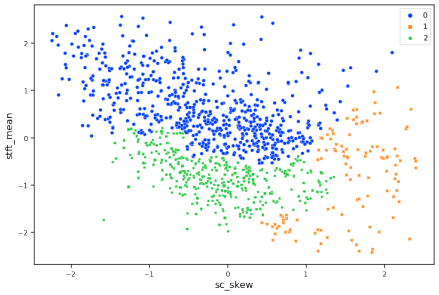
(a) Partition: Emotional Intensity = "normal", Linkage: Complete, Threshold: 4.5



(b) Partition: Emotional Intensity = "normal", Linkage: Complete, Threshold: 4.5



(c) Partition: Emotional Intensity = "normal", Linkage: Average, Threshold: 2



(d) Partition: Emotional Intensity = "strong", Linkage: Complete, Threshold: 4.5

Figure 17

This is the partition that gave us the most reflection insights, in fact from Figure 17a, the clear division between the two clusters found is clear. In contrast, for Figure 17b it is possible to see two clusters separated by the value of *sc_mean* ≈ 0.8. This algorithm, with linkage = "average," also identified two clusters with globular shape and different densities 17c. Finally, in Figure 17d, it is interesting to note

the division between clusters 0 and 2 while the points in cluster 1 can be defined as noise compared to the rest.

2.6 Discussion

Looking at the clusters obtained using the different algorithms to the same data set [3], we noticed that:

1. the density-based algorithms do not provide significant clusters because the data objects are crowded close to each other, and the data set do not contain substantial changes in density. The opposite was the case for the other two types, which provided us with several affected clusters as output.
2. centroid-based algorithms, according to the SSE/k graph, require on average a larger number of clusters to find the ideal SSE value. Nevertheless, bisecting k-means solves the problem of standard k-means: it is not susceptible to centroid initialisation problems.
3. The Hierarchical Clustering algorithm, thanks to the "complete" and "average" linkage, allowed us to find fewer clusters, compared to the centroid-based ones, but with a clear separation between them and identified several clusters with non-globular shapes.

For the aforementioned, we consider the Hierarchical Clustering algorithm to be the one that provided us with the best results in terms of meaningful results and quantity of variables that were able to be analyzed using the algorithm.

3 Classification

In this section we present the results of applying three classification algorithms: decision trees, k-NN and Naive Bayes. These algorithms are applied to three variables that the group consider the more interesting to analyse: *sex*, *emotion*, and *emotion positivity*.

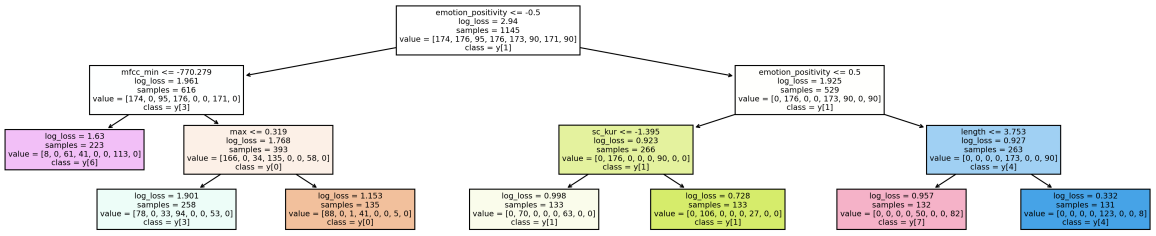
3.1 Decision Trees

We used the decision trees classifier with the default values as first approach. Those values usually gave us complex decision trees, and models with low performance. Then, we decided to search for the best parameters using a grid search.

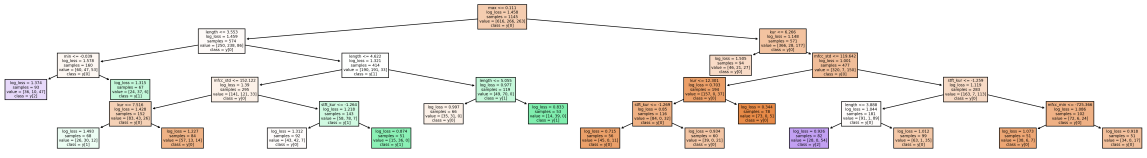
First we tried to predict *emotion*. The default set-up generated a model that gives a decision tree with depth greater than 10, accuracy and f1-score of 0.61, and roc score of 0.78. Curious to know if we were able to find a less complex tree with higher or approximately the same performance, we used crossvalidation to divide the dataset and then we conducted a grid search that gave us the following optimal parameters: *max_depth*: 25, *min_samples_leaf*: 0.002, and *min_samples_ split*: 0.01. With these parameters we are able to obtain a model with accuracy score of 0.6 and roc score of 0.90. The new decision tree retrieved in Figure 18a shows a less complex tree with *depth*=4. The roc curves for each class of emotion in Figure 19a shows that happy and surprised are the best predicted classes, followed of calm and neutral classes.

After, we trained a model to predict *sex*. The first model obtained with the default parameters of the classifier gave us a tree with depth four, accuracy score 0.83 and roc score of 0.90. Using a grid search we obtained a new tree with higher accuracy of the prediction and roc scores, respectively, 95% and 94%, but in this case with higher depth than the first model as shown in Figure 18c. The optimal parameters given by the grid search were *criterion*=entropy, *max_depth*= 117, *min_samples_ leaf*= 0.026, and *min_samples_split*= 0.00873. The roc curve in Figure 19c shows that the area under the curve for both classes, F and M is the same, 0.94 which shows that the model has a good performance when predicting both classes.

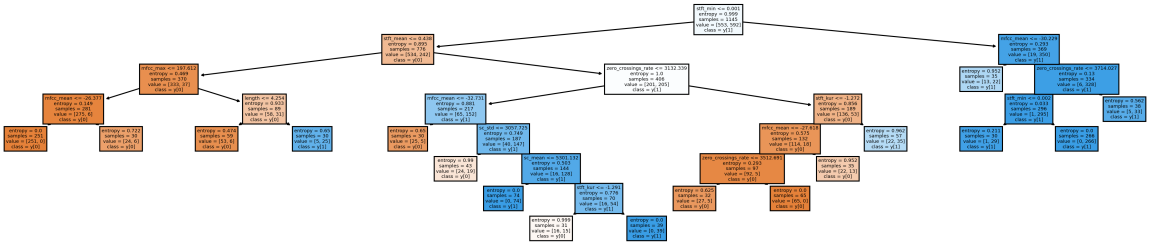
[3] Partition: Emotional Intensity, Value: Normal, Attributes: Intensity and Length



(a)



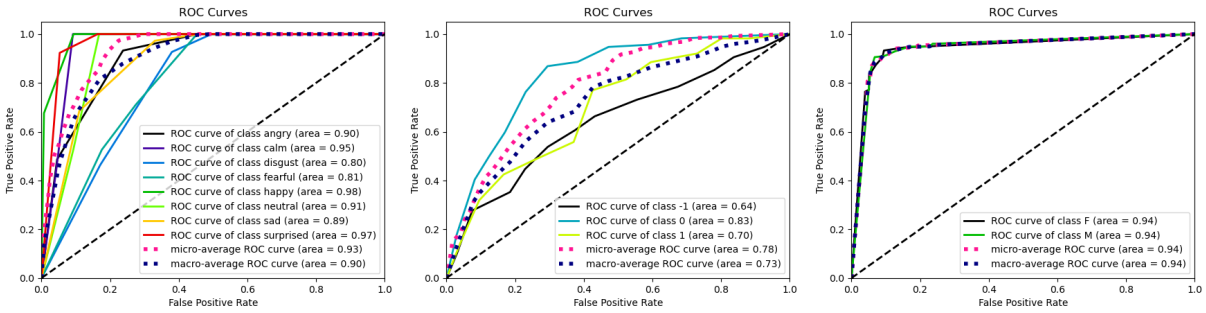
(b)



(c)

Figure 18: Decision trees to predict a) *emotion*, b) *emotion positivity*, and c) *sex*.

Finally, we trained a model to predict *emotion positivity*. The parameters obtained after performing a grid search generated a model with accuracy score of 58%, f1-score 0.54 and roc score 0.72. The optimal parameters *criterion*=log_loss, *max_depth*= 127, *min_samples_leaf*= 0.0554, and *min_samples_split*= 0.043 generated the tree in Figure 18b. Additionally, from the roc curves in 19b we can see that the class better predicted is class 0 with area under the curve of 0.83 that is higher than the average roc score.



(a) (b) (c)

Figure 19: Roc curves of the trained models to predict a) *emotion*, b)*emotion positivity*, and c) *sex*.

3.2 Naive Bayes

The dataset used for this part is the same used in the data preparation and understanding part, also used for clustering. After applying the algorithm for all these attributes, the one that came out to be the most interesting is *emotion*.

Poor model reliability for the emotion prediction because the accuracy is of 60%. The highest precision is for the class sad with score 0.76, which has the best TP/FP ratio and the lowest FP support. The highest recall has it the class calm with score 0.75, which has the best TP/FN ratio, but not the lowest FN support. Also the calm class has precision of 0.67 and it is the most balanced class, for this reason the highest f1-score is of calm class, since the latter is both sensitive to FP and FN.

Using the confusion matrix in Figure 20a we have another proof that the class calm has the highest TP support. We can also see that the fearful class has the highest FN support, and in fact, it has also the lowest recall (0.43), since the latter is sensitive to FN. In ROC curve, Figure 20d, we see what we can expect from everything we have said so far: the calm class has a high area value, lower only than surprised and neutral classes but they have a much lower support than calm one.

When trying to predict *sex*: the accuracy obtained, 93%, is higher than the emotion one, it is also the highest accuracy among all the chosen attributes, but we have also to take into account that *sex* is a binary class problem. The female class has a higher precision (0.96) than male (see Figure 20c), and in fact it has only 10 FP out of 491, but the highest recall is of male class (0.96), that means female has an higher FN value, 23 out of 491. However, these numbers are small compared to the total support and the differences are not so significant, in fact, the f1-score and roc score are similar (see Figure 20f), 0.93 for female and 0.94 for male, and 0.94 for both, respectively.

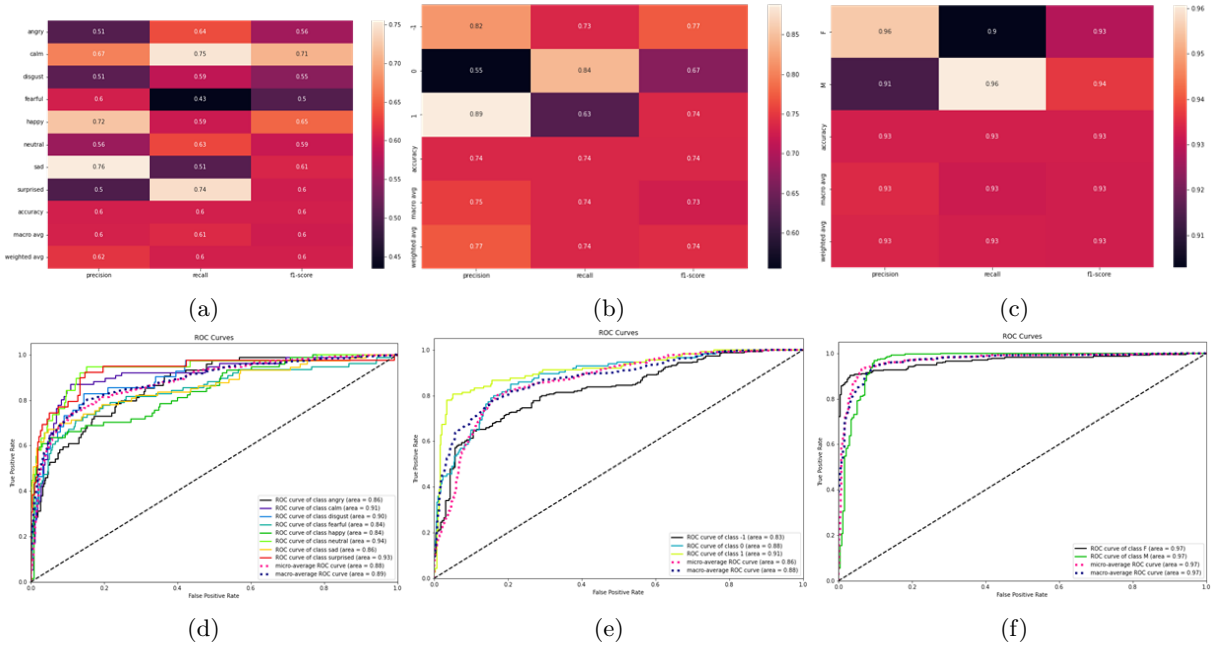
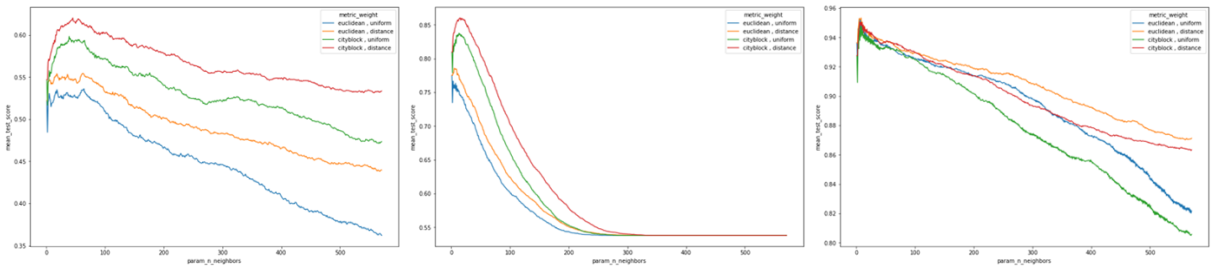


Figure 20: First row: matrices of performance of the trained models. Second row: ROC curves of the trained models for a) *emotion*, b) *emotion positivity*, and c) *sex*.

Finally, the last attribute is *emotion_positivity*, which is a three-class problem with accuracy of 74%, higher than emotion but lower than sex. The class 1 has the highest precision, 0.89, with only 9 out of 113 FP. On the other side, class 1 has the lowest recall, and consequently the highest FN value, 42 (Figure 20b and 20e). The highest recall is of neutral emotions, the class 0, which is 0.84, but with much higher support and the lowest number of FN, 18.

3.3 *k*-nearest neighbours (*k*-NN)

In this subsection the dataset used is the same than the one obtained after the data preparation phase. To choose the best parameters was used a grid search changing: the number of neighbours to use from 1 to $\sqrt{N}$ with N the number of rows, the weight between uniform and distance, and the metrics between euclidean and cityblock. Figure 21 shows the accuracy values as a function of the value of k. In this figure



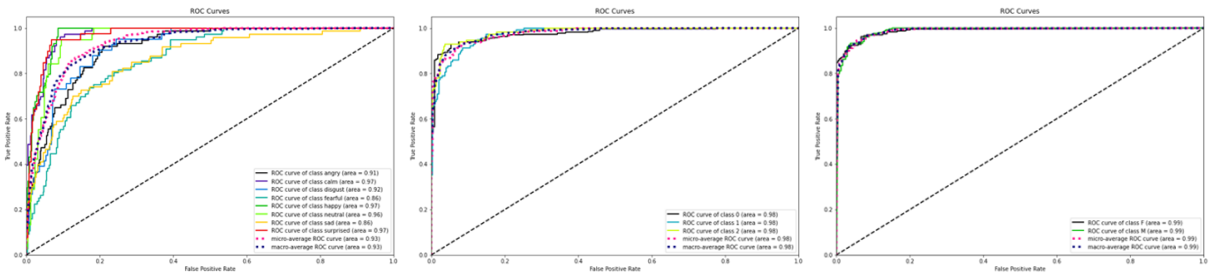
(a) (b) (c)

Figure 21: Mean test score (accuracy) as a function of the neighbour number for the models of a) *emotion*, b) *emotion positivity*, and c) *sex*.

is possible to see that models that use cityblock as metric and distance as weight perform better regardless of the value of k, in contrast to the models inherent to the *sex* variable, which alternate performance according to the combination of metric, weight and k.

The optimal number of neighbours obtained is relatively small for all three variables, in fact in the case of *emotion* it is 45, while it is equal to 15 for *emotion positivity* and 8 for *sex*. The area under the ROC curve (Figure 22) appears excellent, being 0.98 for each class of *emotion positivity* and 0.99 for each class of *sex*. Good results are instead obtained for the model that classifies the *emotion* variable, which has AUC values between 0.86 and 0.97. It is curious to note that the model relating to the variable *emotion positivity* is much better than the one relating to *emotion*, despite the fact that the first one is an attribute derived from the second. Finally, if we choose to keep as classification models those with higher accuracy, we will have as results:

- Accuracy *emotion* model: 0.641
- Accuracy *emotion positivity* model: 0.896
- Accuracy *sex* model: 0.945



(a) (b) (c)

Figure 22: Roc curves of the trained models with k-NN to predict a) *emotion*, b) *emotion positivity*, and c) *sex*.

3.4 Discussion

The following table summarizes the performance results of the trained models for each variable, using the three classification algorithms: decision trees (DT), Naive Bayes (NB), and k-NN.

	Emotion			Emotion positivity			Sex		
Regressor	DT	NB	k-NN	DT	NB	k-NN	DT	NB	k-NN
Accuracy	0.6	0.6	0.64	0.58	0.54	0.72	0.94	0.93	0.95
F1-score	0.61	0.6	0.61	0.54	0.73	0.78	0.9	0.93	0.94
Roc	0.9	0.89	0.93	0.72	0.78	0.98	0.95	0.97	0.99

Observing the table, we can appreciate that for all the variables analysed, the values of the performance metrics are always higher for the algorithm k-NN. All the algorithms had good performance when trying to predict *sex*. The *sex* variable appears to be an easy variable to predict compared to the others, as the efficiency of the different algorithms differs by at most 0.04. Instead, when we observe the values of *emotion positivity*, we realize that it was a variable difficult to predict. In fact, we can observe that the efficacy of DT and NB is really similar, and there is a big difference with the performance of k-NN. On the other hand, k-NN has the lower performance when trying to predict emotion, which is a bit curious since emotion positivity is a variable created in base of emotion. Therefore, it is expected that the performance values of the algorithms be similar for these two variables. Additionally, when thinking about the importance of the measures precision and the recall, we noticed that they do not need be considered in this analysis, because we did not choose as target attributes cases that required the evaluation of these parameters where depending on the variables to have one of them higher than the other is better. In general, it does not seem to us that the RAVDESS dataset has attributes that require this type of assessment, on the contrary, a case in which Precision and Recall would have made sense in datasets similar to the ones of virus infections, such as COVID-19 test results.

Finally, from the decision trees plot we can say that one way to predict *emotion* is checking only to the values of the variables *max, intensity, kur, stft_skew* and *sc_std*. To predict *emotion positivity* are needed it only *max, min, length, kur, mfcc_std, stft_kur*, and *mfcc_min*; and to predict *sex* we require to use only *zero crossing rate, stft_mean, stft_min, length, sc_mean, sc_std, stft_kur, mfcc_max*, and *mfcc_mean*.

4 Pattern mining

4.1 Frequent pattern extraction & Discussion

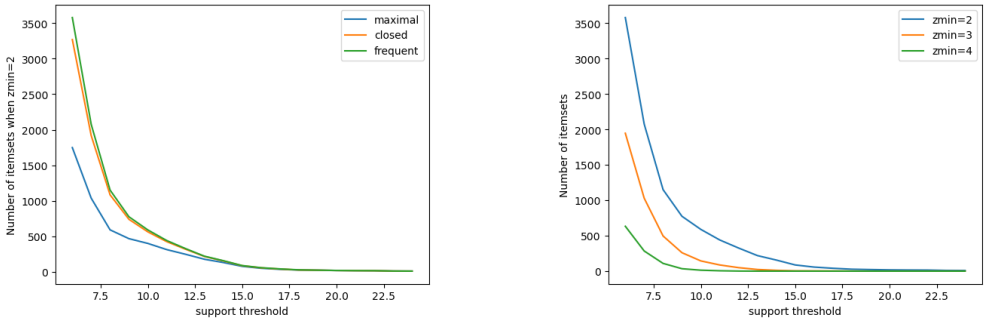
In this section we used the *apriori* algorithm to find the frequent itemsets. We considered different minimum support values, *minsupp*, and minimum number of elements, *zmin*, in the itemsets. Firstly, when *zmin*=2 and varying *minsupp* starting from 5, we noticed that initially there are more than four thousand frequent itemsets. This number continues decreasing when the *minsupp* value increase until we arrive to *minsupp*=27, for which there are only three frequent itemsets which is a really small number to obtain useful association rules. Figure 23a shows how changes the quantity of frequent itemsets depending on the *minsupp* value when *zmin*=2. Additionally, something interesting we can notice in this case is that from *min_sup*=20, the closed itemsets are the same maximal itemsets.

When *zmin*=3, we notice that the maximum value that can take *minsupp* is 18 for which is obtained only one frequent itemset with support of 18.58%. We perceived a similar behaviour to the first case when varying the values of the *minsupp*. For *minsupp* $\geq$ 13 the maximal itemsets are the same closed itemsets. Finally, when *zmin*=4, it is expected that the number of itemset decrease, as well as the support threshold. In fact, in this case we have that the maximum value that can take the *minsupp* is 12% for which is found only one itemset, and if *minsupp*=10%, we get 13 itemsets. In Figure 23b, we can appreciate how it changes the number of itemsets for different values of *zmin* and *minsupp*.

4.2 Association rules extraction & discussion

Different values of confidence were considered to find different association rules when *zmin*=2,3,4. According to the analysis done during the frequent itemsets extraction, we decided to choose for each value of *zmin* the values 20%, 14%, and 11% of *minsupp* respectively. This values allows us to have an enough quantity of interesting rules. In Figure 24a we observe that he lowest the percentage of confidence, the higher the number of rules obtained. Moreover, when the confidence is between 80% and 90%, the lower the support threshold is, the higher the number of rules.

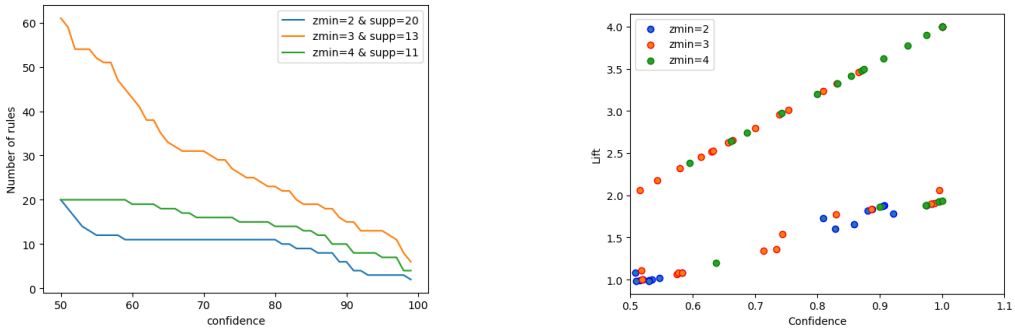
Since we would like to use the association rules obtained in this step for a possible replacement of missing values or prediction, we would like to use only the ones with a high precision value, in this case measured in terms of confidence and lift score. Some of the rules with higher confidence value for *zmin*=2 are presented in Figure 25. The rules with confidence value equal 1 should be treated carefully because their



(a) (b)

Figure 23: Variation of the number of itemsets for different values of support when a) *zmin*=2, and b) *zmin*=2,3,4.

atypical value can be due to a wrong preprocessing manage of the data or bias already presented in the data. Using the rules obtained when the confidence threshold is 50, we obtained the scatter plot 24b linking the confidence and lift values. This graph shows us a clear linear relationship between confidence and lift when *zmin* is 3 and 4. Instead, when *zmin*=2, we obtain two groups. The lift of the rules with confidence less than 60% is around 1, and the rules with confidence between 0.8 and 0.95 is between 1.5 and 2.



(a) (b)

Figure 24: a) Quantity of rules obtained for different values of *zmin*. b) Comparison between confidence and lift values of the rules obtained for different values of *zmin*.

	consequent	antecedent	abs_support	%_support	confidence	lift
4	(0.999, 409.75]_stft_min	((0.999, 409.75]_sc_min,)	409	25.000000	1.000000	4.000000
5	(0.999, 409.75]_sc_min	((0.999, 409.75]_stft_min,)	409	25.000000	1.000000	4.000000
0	M	((1227.25, 1636.0]_stft_min,)	402	24.572127	0.982885	1.900709
2	F	((-3.181, -0.727]_stft_mean,)	371	22.677262	0.907090	1.878481
3	F	((-2.8489999999999998, -0.713]_mfcc_max,)	370	22.616137	0.904645	1.873418
1	M	((-3.78, -0.595]_stft_std,)	377	23.044010	0.921760	1.782506

Figure 25: List of the rules with confidence value higher than 0.9 when *zmin*=2 and *minsupp*=20.

4.3 Using the extracted rules

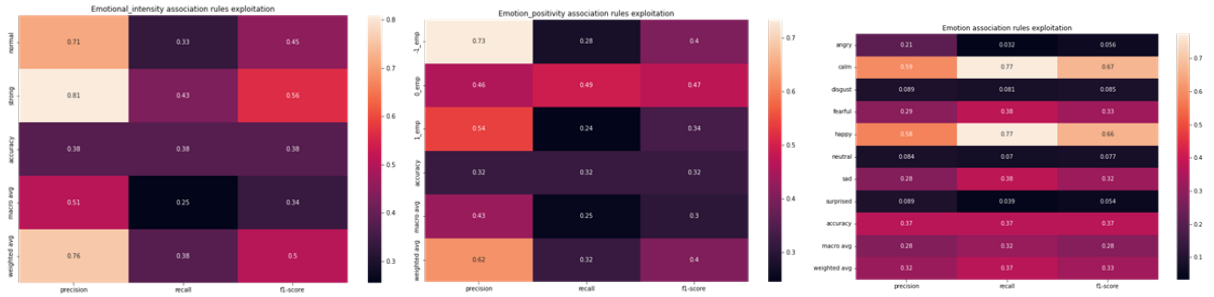
The variables with missing values are *actor*, *vocal channel*, and *intensity*. With the associated rules found we noticed that is not possible to replace missing values of the continuous variables, like *intensity*, because the rules gave us an interval, and not an exact number. On the other hand, looking at the rules generated with minimum confidence equal to 80% we found that is possible to predict the attribute *sex*.

First, we started using the rule with the highest confidence value having as consequent *M*. With this rule the performance of the prediction is of 70%, predicting better M than F. We continue using more rules to predict M, but the prediction performance didn't have dramatic changes, specifically when predicting F, the score continue being low. So, we thought that the best way to obtain better performance was using together rules to predict F and M. Using the rules with the three highest values of confidence for both F and M we obtained an accuracy of 90% predicting F, 84% predicting M, and f1-scores of 73% and 65% respectively.

Additionally, we tried to predict *emotional intensity*. With a minimum confidence threshold of 70% and taking the rule with the highest confidence for each *emotional_intensity* class we obtain a prediction model with 38% accuracy, which is a much lower value. This is due to the lower confidence of the strong class, which has only one rule with greater than 70% confidence, compared to the normal class which has four rules with greater than 70% confidence. Despite this, the values of precision, recall and f1-score are higher in the strong class than in the normal class and this could be caused by an higher support and confidence values in strong, which has in its only rule taken into account a percentage respectively of the 20.23% and the 81%, compared to the normal rules in which the most common reaches 17.85% with a confidence of 71%.

To predict *emotion positivity*, we used one rule for each class of *emotion positivity*. From this we obtained a model with an accuracy of 32%, a very low value but influenced by the minimum confidence threshold of 30% because only the −1_emp (negative emotions) class has at least one rule that reaches the 70% of confidence. The other two classes have a lower confidence rule, which at most reach 36% and 46%.

Finally, during the prediction of *emotion*, the model turned out to be problematic to build, as some classes occur much less times than others, as can also be understood from the results. However we still wanted to try to make a prediction and, although the minimum confidence value is very low, 10%, the total accuracy is much higher, 37%. Net of a couple of interesting values, such as the recall of the calm and happy classes, it is obvious that the above model is neither reliable nor informative.



(a) (b) (c)

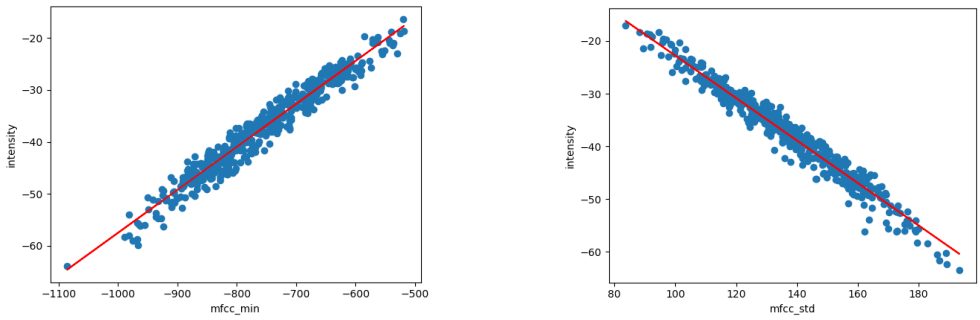
Figure 26: Classification report a) *emotion intensity*, b) *emotion positivity*, and c) *emotion* by exploiting the association rules obtained in subsection 4.2.

5 Regression

Having into account the insights we obtained from the visualizations at the beginning of this report, we noticed that *intensity* shows a clear relation, linear or not with different variables like *mfcc_std, mfcc_min, std, min* and *max*. Therefore, we will use several regressors to find such relations. First, we used the *simple linear* regressor to find the relationship between *intensity* and *mfcc_min*. We obtained the line shown in Figure 27a, with slope=0.0827, intercept with the axis y, b= 25.1612. The metrics of performance corresponding to this regression are: coefficient of determination = 0.95, Mean Squared Error (MSE) of 3.61, and Mean absolute Error(MAE) of 1.56.

After, we used *Ridge* regressor to find the relationship between *intensity* and *mfcc_std*. This regressor gave us the linear model $y = -0.4x + 17.52$ shown in Figure 27b, with a coefficient of determination of 96% better than the linear model applied above. Moreover, the MSE and MAE improved, being their

valuer 2.93, and 1.33 respectively.



(a) (b)

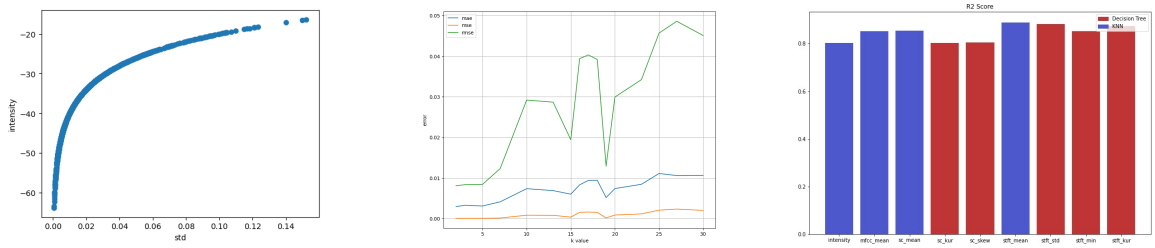
Figure 27: a) Linear regression model of the pair *mfcc_min* and *intensity*. b) Regression model got with Ridge regressor for the pair *mfcc_std*, and *intensity*.

Staying on the insights got in the data exploration phase, we remarked a non linear relation between *intensity* and *std*, as shown in Figure 28a. So we decided to use k-NN regression to predict *intensity* using the values of *std*. For this we divided our dataset in train and test set. We compute the Manhattan distance from the test set to the train set, and we tested different values of k in order to choose the one that generates the lowest error. Using the comparison of error with respect the values of k in Figure 28b, where we can see the MSE and MAE errors for the different values of k, we decided to choose k=5 as parameter to perform the prediction.

Finally, to continue exploring the non-linear regression comparing the k-NN regression model with the one that uses the Decision Tree algorithm, we used a randomized search varying the criteria, and using the root mean squared error as the comparison metric. We decided to use this scoring method because it is very valid for comparing models relating to the same variable and, in addition to this, it is an easily interpretable metric.

Since we have many numerical variables, and we didn't know a priori which ones might have more interesting models, we decided to analyse each variable. Two models were created for each variable using the k-NN and Decision Tree algorithm, in order to keep only those with an R^2 value greater than 0.8. In the case where both respected this condition, we kept the one with the greater value. As a result, the models with better R^2, Figure 28c, value were:

- Decision tree regressor: *sc_kur*= 0.803, *sc_skew*= 0.804, *stft_min*= 0.851, *stft_kur*= 0.874,
- k-NN regressor: *intensity*=0.803, *mfcc_mean*= 0.851, *sc_mean*= 0.853, and *stft_mean*= 0.889.



(a) (b) (c)

Figure 28: a) Non-linear relation between *std* and *intensity*. b) Error measurements when varying the k parameter for the prediction of *intensity*, when using k-NN regressor. c) R^2 score of the non-linear models.