MUSICAL INSTRUMENT RECOGNITION WITH MACHINE LEARNING

André Luis Ferreira Marques

Dept. Engª Elétrica -- Escola Politécnica – Universidade de São Paulo

ABSTRACT

This work deals with musical instrument identification (audio signal treatment), with machine learning, handling open datasets. The main purposes are to understand the whole digital process for music instrument identification (e.g., violin, guitar, etc.); to build up a machine learning (ML) classifier to work with more than 3 groups; to measure the efficiency of this joint tool set with a mathematical metrics (e.g. recall, accuracy etc.) The parameters of the ML classifier, here the K nearest neighbor (KNN), will be evaluated along the variation of some data pre-processing techniques.

***Index Terms* — Musical instrument, machine learning classifier, KNN.**

1. INTRODUCTION

The identification of musical instruments has developed with the use of neural networks, within the application known as 'music retrieval', also using some Data Science (DS) features, such as classifier algorithms and their metrics [1]. Polyphonic music is made of several instruments, by different principles and energy management (hand touch, air blowing, string vibration, sound waves, materials vibration etc.), making their 'sound' specific like a proper signature (timbre, loudness, duration, and pitch). Thus, instruments can be classified in categories, according to the method to sound generation method: aerophones, electrophones, membranophones, idiophones etc.

However, while hearing a symphony orchestra performance, it seems hard to identify the individual contribution of each instrument, unless with a special training or gift. This happens due to the high combination one may find it. As a result, the use of digital technology has aided the deep study of past music works, providing data/information on how each instrument, or even a set of instruments, have been planned to work [2].

This research work focuses the development of a digital setup, based on a machine learning technique, to identify whether a specific instrument has been used in a music play/dataset. This identification will be expressed as a category assigned to a specific instrument trace. Thus, one summarizes the main objectives as follows:

a) Comprehension of the digital techniques aimed for the music instrument identification, including tools like Fast Fourier Transform (FFT) and others.
b) Dataset selection & analysis, here based on open sources.
c) Build-up of a pre-treatment digital tool, using features like the MEL Frequency Cepstrum Coefficients (MFCC) and spectrograms.
d) Set-up of a machine learning (ML) classifier algorithm, for instance the K Nearest Neighbor (KNN), to indicate what music instrument had been used.
e) Evaluate the scheme above by mathematical metrics and confusion matrix with many categories.

3. METHODS

In this research work, one handles digital data from open sources, checked throughout an explanatory data analysis or similar [3]. Once the dataset has been selected, the main features are extracted, and data pre-treatment carries on.

The spectrograms/output is then loaded into a ML algorithm (here KNN) to identify the groups of instruments identified, using Python as the programming language. The results are quantified by evaluation metrics, such as 'recall', 'precision' and 'f1-score', because one deals with more than two groups of instruments [4]. The data are separated into 'train' and 'test' sets, according to the supervised scheme. The better the data input tunning, the higher the metrics above. In the end, the prediction is carried out.

3.1 The KNN Method

In a first view, the KNN method is a classifier based on the Bayesian approach and its characteristics come from the development of the Fisher discriminant function [5], which aims to identify the classes or groups among a data set. Under the Fisher framework, one intends to maximize the group separation by using a linear method, using weights for this aim, as seen in other applications. KNN belongs to the simplest examples of supervised ML techniques.

The learning criteria in the KNN framework focuses to discover how similar a specific datum (or vector) is to the others. The datum/vector may have manifold dimensions. The key scope of the function deals with the identification of a linear combination of features able to

pinpoint two or more classes of data. It is linked to the analysis of variance (ANOVA) [6].

One basic assumption of the KNN method relies on the items to be checked stay close to the similar one among themselves. The distance or closeness between the items is the parameter to represent how far the similar items are, considering different arrangements: Euclidian, Manhattan, and others [7]. The most common used is the Euclidian. Figure 1 shows a simple example of the method background. Consider the legend 'classe' as class.

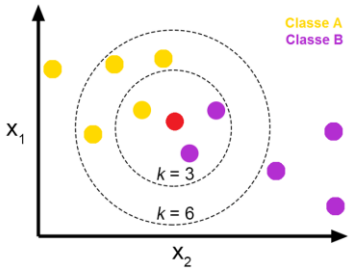

Figure 1: simple KNN method background [7]

Make note that 'K' is the number of neighbors of a specific datum (red dot). In short words, the algorithm associated to the KNN follows [8]:

a) Load the dataset and split into 'train' and 'test' groups.
b) Pick up a value of 'K', within [1,∞].
c) For an indicated datum, calculate the distance for the 'K' neighbors, considering their known indexes/parameters.
d) Add the indexes and distance to an ordered collection.
e) Sort the ordered collection from the shortest to the longest distance, with the related indexes.
f) Check the shortest distances/indexes and frequency of occurrence.
g) Relate the indicated datum to the group with the highest frequency and shortest distance.

The development of the classification method takes the train data branch and once defined, it will classify the test data group, when the metrics will be calculated. The key task is the chose of the 'K' value: when working with small K values (e.g., K=1), the chance of 'overfitting' becomes high; on the other hand, with large K values, the chance of 'underfitting' grows, although the classification becomes more consistent. The associated metrics (e.g., recall and precision) will indicate both situations [9].

The Euclidian distance is computed as shown, between points (x,y) and (a,b). It is a classical definition considering a two-dimension problem:

$$\mathbf{dist((x,\ y),\ (a,\ b))} = \sqrt{(x - a)^2 + (y - b)^2} \qquad \mathbf{(1)}$$

Considering the distance calculated as indicated above, the assigned class to a specific datum or sample comes from the highest probability, as stated by (2) based on the Bayes Theorem, taking 'j' as the class linked to the 'test' sample Y, considering the trained predictor from sample X for the class 'x_o'.

$$\mathbf{P(Y=j/X=x_0)} \qquad \mathbf{(2)}$$

When dealing with two classes (e.g, 1 and 2), the threshold for the probability above is 0.5, as shown by Equations 3 and 4:

$$\mathbf{P(Y=1/X=x_0) > 0.5} \qquad \mathbf{(3)}$$

$$\mathbf{P(Y=2/X=x_0) < 0.5} \qquad \mathbf{(4)}$$

The reasoning above can be extended to a larger number of classes, dimming the threshold proportionally. The conditional probability is not known in advance and the KNN method helps to settle it. The probability concept to assign a class to a specific sample, taking the highest value as a criterion is shown below.

$$\mathbf{P(Y=j|X=x_0) = (1/K) * \Sigma\ I(y_i = j)} \qquad \mathbf{(5)}$$

The sum indicated on the RHS refers to a finite set of data with 'Y' belonging to a 'j' class. When making the quotient with the 'K' value chosen, one reaches the conditional probability. The higher this amount, the closer the assignment of the test data to the class 'x_o'. The error rate of this Bayes Theorem application is shown by (6), and 'E' means the highest value from a probability distribution function (pdf) or the 'expected value' [10]:

$$\mathbf{1 - E(max_j\ P(Y=j|X=x_0))} \qquad \mathbf{(6)}$$

3.2 Classification Metrics

The commonsense metrics applied when doing a classification are [11]: recall, precision, f1_score and accuracy, and one presents remarks about them when dealing with a multi class framework. Equally important, there are four possible aftermaths: 'true positive' (TP), 'false positive' (FP), 'true negative' (TN) and 'false negative' (FN), with 'true' as a correct result, 'false' as an incorrect outcome, 'positive' as an indicated option and 'negative' as a non-indicated selection. The classification above can be structured for different number of classes or categories. In addition, reference [12] presents other metrics to be used in classification tasks: the ROC AUC score, the log loss, the Cohen Kappa score, and the Mathew´s correlation coefficient.

3.2.1 Recall

The straight math definition for Recall is a ratio between TP and the sum (TP + FN), or:

$$\textbf{Recall} = (TP)/(TP + FN) \quad (7)$$

The metrics above aids to answer the question: 'How many actual positive cases are correctly classified?', which has a key importance when dealing with medical subjects mostly.

3.2.2 Precision

This metrics aids the answer the questioning: 'What is the proportion of the predicted positives are really positive?', considering the positives may be TP or FP. In a short, this metrics focuses the 'correctness' of the classification tool. Thus, the math definition of precision is shown by Equation 7:

$$\textbf{Precision} = (TP)/(TP + FP) \quad (8)$$

3.2.3 f1_score

This metrics combines the two ones above, and it can be seen as a weighted approximation with them, with 'R' as recall and 'P' as precision:

$$\textbf{f1\_score} = 2*R*P/(R + P) \quad (9)$$

3.2.4 Accuracy

The overall assessment on how flawless the classification tool can be seen by the accuracy. It is a ratio between all the 'true' cases (TP + TN) and the total cases, or (TP+TN+FP+FN). Equation 9 presents the math definition:

$$\textbf{Accuracy} = (TP + TN) / (TP+TN+FP+FN) \quad (10)$$

In this work, one will calculate all the metrics above and take the 'f1_score' as the main figure while making the evaluations, because it handles the two most significative metrics [11].

3.2.5 ROC AUC

The Area Under the Receiver Operating Characteristic Curve revels how the classification tools can detect the classes, based on probabilities, more often used in 2 classes problems. It considers two rates: the True Positive Rate (TPR) and the False Positive Rate (FPR). Due to the several classes of this work, this metrics will put aside.

3.2.6 Log loss

This metrics also works with probabilities and considers the lowest rank as the best one, because it handles an error function. It also considers the cross-entropy loss, and it is more used also for a 2-class problem. For multiple class case, it becomes more complex, not adding significant value to the metrics above.

3.2.7 Cohen Kappa score

This metric handles the possible imbalance between classes, and it can be taken as a supercharged version of 'accuracy', more applicable to 'categorical' or qualitative items [13]. It has the following definition:

$$\mathbf{k} = (p_o - p_e)/(1-p_e) \quad (11)$$

where:

- p_o = observed proportional agreement between actual and predicted values.
- p_e = probability that true values and false values agree by chance.

3.2.8 Mathew´s correlation coefficient (MCC)

It handles the true values and the predicted ones, ranging from -1 to +1. For a 2-class problem, it has the following definition:

$$\textbf{MCC} = (TP*TN + FP*FN)/[A*B*C*D]^{0,5} \quad (12)$$

Where:

- A = (TP + FP)
- B = (FP + FN)
- C = (TN + FP)
- D = (TN + FN)

Normally, MCC values above 0.7 indicate a good rate for the classification tool. MCC equals to 1.0 means the perfect classifier and, when equals to 0.0, it indicates the classifier is not better than random occurrence [14].

In this work, one surveys the f1_score as the main metric, also having a look into the recall, precision, accuracy, and the Cohen Kappa scores, as an ancillary option. The higher the metric, the better the model under evaluation, except the Cohen Kappa, which the closer to 1.0, the better the model respectively.

3.2.9 Dataset

References [15,16] present some open-source datasets regarding the musical instruments field, including sets for specific instruments or a whole list of them. In this work, one considers the datasets with the 'MP3', 'WAVE' or 'WAV', formats, allowing the trimming of the silence and other sound oriented tasks. One of the most complete is ISMIR [16] has a vast set of sound data, including sole instruments or arrangements, different types of 'tempi', music genres, vocal tracks and playing techniques.

In this work, one takes the set from Philharmonia [15], an orchestra found in 1945 in the UK, also providing a set of instrument recordings for digital processing, in a MP3 format, with six musical instruments: viola, sax, trumpet, oboe, flute and cello. Oboe, sax, flute, and trumpet belong to the 'blowing' case, viola, and cello to the 'string' group. Equally important, each instrument has a different key and mode data collection, say 'B & pianissimo', 'D & forte normal', 'mezzo-forte normal', and so on. The capital letters refer to the harmonic keys: A-La, B-Si, C-Do etc. In the end, there are 600 audio files to be treated and handled.

The dataset from Philharmonia has 100 samples for each of the six classes above, which means the data is balanced, skipping the need to complete or pre-treatment in this field.

3.2.10 Task Sequence

Following a task flow normally used in DS applications, the step sequence planned follows:

a) The computing tools, libraries, functions, and others are selected/incorporated, such as 'librosa', 'numpy', 'seaborn', 'matplotlib', among others, according to its application profile to be used in a 'Google Colab notebook'.
b) The dataset is then downloaded from the website "https://www.philharmonia.co.uk/resources/sound-samples/", using an API access procedure. The other option is downloading the dataset into a working area of the local computer.
c) Each instrument file receives a numeric label regarding the six cases above, ranging 0 to 5: 0-cello, 1-flute, 2-oboe, 3-sax, 4-trumpet, and 5-viola.
d) The data is now converted into vectors, each instrument file represented by its MFCC parameter set.
e) As a basic hypothesis, to apply the sound treatment techniques, one considers the following baseline parameters:
    a. Sampling frequency: 44100 Hz.
    b. FFT window: 2048 bits.
    c. Hop length: 512 frames.
    d. Mel bands: 128.
    e. MFCCs: 13.

    The Mel bands and MFCCs will remain fixed. The parameters to vary will be then:
    - Sampling frequency: [22050-44100-96000]
    - FFT window: [1024-2048-4096]
    - Hop_length: [256-512-1024]
f) For the KNN ML method, the data will be divided in a ¼ ratio for the test group.
g) The dataset is then set in an array format and the 'standard scaler' function is also used to reshape the data.
h) The number of neighbors or 'K' will range within: [1,2,3,4,5,6,7,10,20,30,50 and 100].
i) At each run, for a specific sampling frequency, FFT window and hop_length, the variation of 'K' will be carried out after a new data separation into 'train' and 'test', to maintain the necessary random approach.
j) The KNN algorithm is then applied, and the metrics are assessed. For the highest f1_score, one will present the Confusion Matrix.

4. RESULTS

4.1 Data exploratory analysis

In DS works, the data exploratory analysis (DEA) helps a better understanding of the dataset. For instance, the duration of the data seems a factor to be known, because one can trim the 'silent' sectors, easing the computer work. For this work, the maximum duration of a instrument signal is about 2.6s. Table 1 shows the main statistics of the dataset.

Table 1 – Phillarmonia_600_duration_statistic

Parameter	Value
Count	600
Mean (s)	1.003
Standard deviation	0.512
Minimum	0.222
25%	0.653
50%	0.922
75%	1.245
Maximum	3.256

Figure 1 shows the duration detected with the dataset. Just a remark, the distribution does not show a gaussian profile, considering a sixty-bin graph configuration.

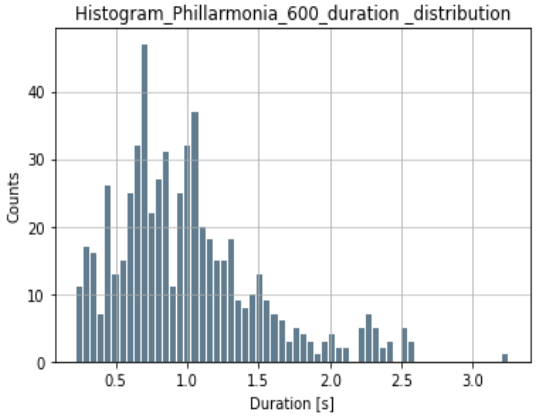

Figure 1 – histogram of Phillarmonia_600 duration

Spectograms were also checked, after the application of the Fast Fourier Transform (FFT) with the initial parameters. Figure 2 presents the spectrogram for the selected case: trumpet, C4_1, 'pianissimo_normal', where the lighter colors refer to the higher amplitude. The dark background of the picture means the 'silent' area or very low sound activity.

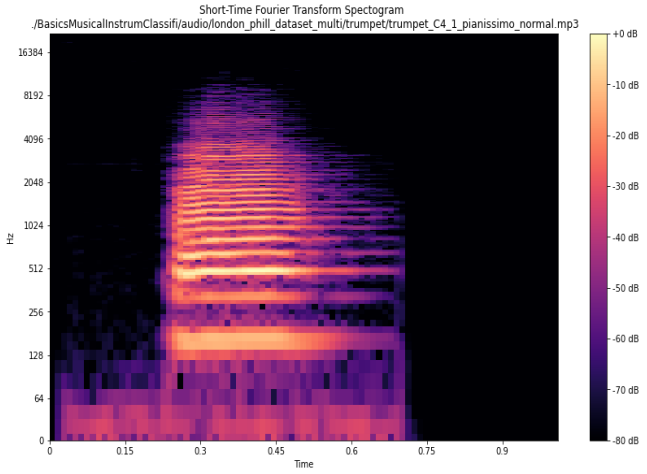

Figure 2 – spectrogram_trumpet_C4_1

4.2 Principal Component Analysis

As the whole dataset has 13 MFCCs, the evaluation of a dimension reduction was carried out, based on the Principal Component Analysis (PCA) method, useful for denoising and data compression, when one detects multicollinearity between variables. The method transforms the input data into a new input data, after a transformation of the correlated variables, now with smaller set of variables and uncorrelated. The dot product is used to operate the input variables. The resulting new input data holds the variance of the original input data, after the use of a linear or non-linear combination [17].

In this work, the least number of variables to be hold comes from the application of the Kaiser criterion, which leads to drop the components with eigenvalues lower than 1.0, considering the explained variance of the input data [18]. For the original input data above, the minimum number of features to be taken is 4, leaving then 9 off.

This fact can be seen in figure 3, where the vertical axis deals with the cumulative sum of the eigenvectors variance and the horizontal axis has the number of principal components associated to the eigenvectors above.

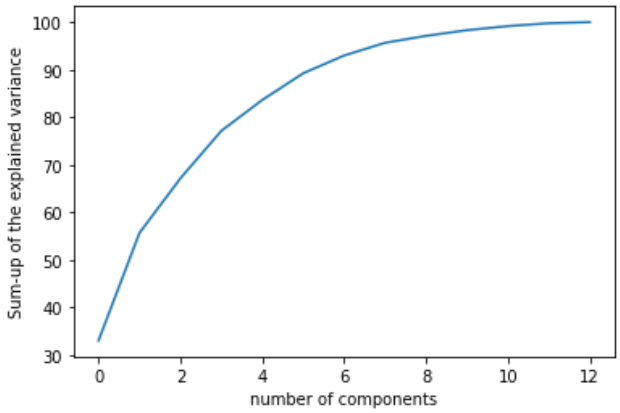

Figure 3 – KNN explained variance

The first four explain most of the variance, because their highest set of derivatives, taking the number of components as the independent variable. Thus, the work may be easier if one takes just these four main components, without any significant outcome loss.

4.3 KNN application

For the initial parameters, the KNN algorithm was trained with 450 samples and tested with the other 150 samples, with the average accuracy around 0.96. Table 2 presents the metrics of 'recall', 'precision' and 'f1_score' for each of the six instrument groups, with one neighbor (K=1).

Table 2 – KNN_baseline_metrics

	0-cello	1-flute	2-oboe	3-sax	4-trumpet	5-viola
Recall	1	0.96	0.96	0.96	0.96	0.92
Precision	0.93	1	1	0.96	0.96	0.92
F1_score	0.96	0.98	0.98	0.96	0.96	0.92

In a first assessment, the Recall values around 1.0 indicates the occurrence of 'overfitting', which must be avoided increasing the number of 'K'. On the other hand, low values of Recall indicate the 'underfitting', which also should be avoided, when increasing the 'K' too much.

To check whether the how strong is the presence of outliers, figures 4, 5 and 6 present the boxplot of the f1_score, Recall and Precision results.

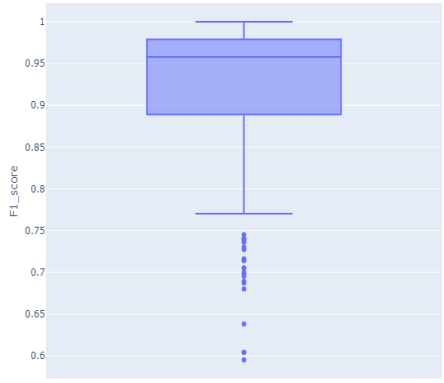

Figure 4 – Boxplot for the f1_score

As one can see, the maximum value for the f1_score was 1.0 and the lowest figure was lesser than 0.6. The mean value was close to 0.95, with some points classified as 'outliers', which were not taken out of the processes, although there are plenty of data, for the sake to complete the work in a first idea.

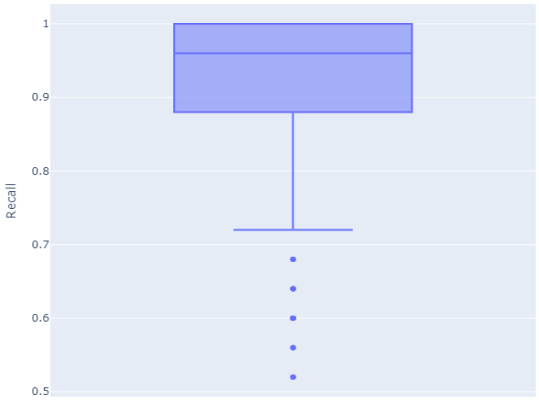

Figure 5 – Boxplot for Recall

For the Recall metric, the number of outliers is smaller than for the F1_score, and then they were not taken away from the dataset to complete the work with the KNN method. Nevertheless, for the precision metric, the same behavior of the F1_score was detected for the outliers, and again they were maintained within the input dataset. The maximum value of precision was smaller than 1.0 though.

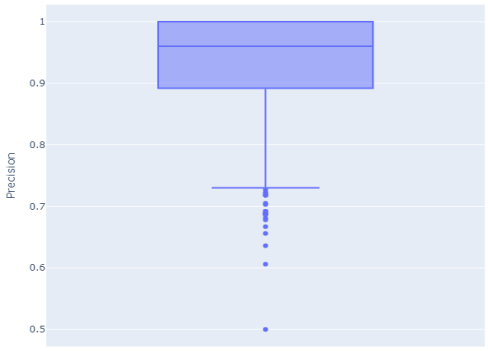

Figure 6 – Boxplot for Precision

Nonetheless, the mean value for the Recall index is higher than for the f1_score, or 0.975 against 0.96. The lowest value for Recall is smaller than the same for the f1_score (0.72 against 0.77).

Figure 7 shows the boxplot for the accuracy index, with the maximum value lower than 1.0, the slowest closer to 0.75, and the mean value equals to 0.95.

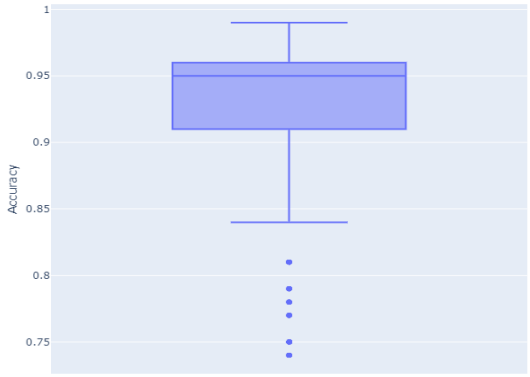

Figure 7 – Boxplot for Accuracy

Figure 8 looks differently than the previous, where the horizontal axis has the values of 'K' and the vertical axis has the F1_score values, but it also has side representations about the distribution for both variables. Its color scale has link with the sampling frequency Fs. There are 3 major regions for 'K': between 1 and 10, between 20 and 50 and 100.

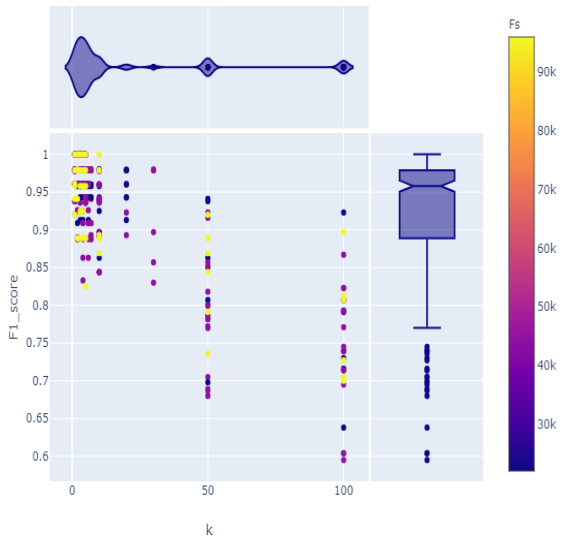

Figure 8 – F1_score versus 'K', as a function of Fs

In a similar way, Figure 9 shows the influence of the hop_length, as a function of the number of neighbors.

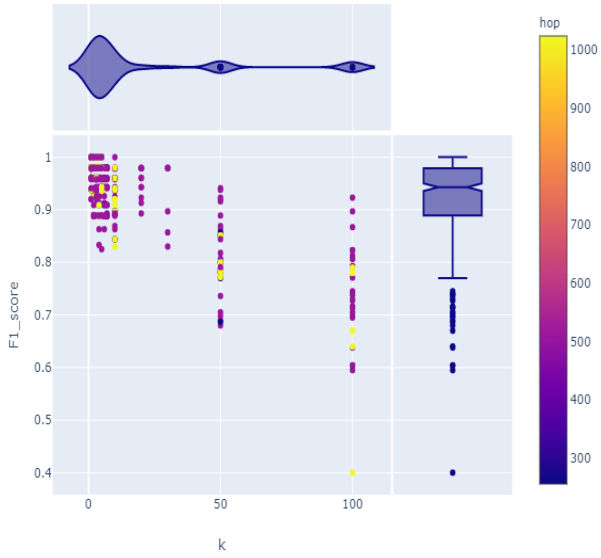

Figure 9 – F1_score versus 'K', as a function of hop

Figure 10 covers the scattering points related to the F1_score and the Recall, under the description of the variation of 'K'.

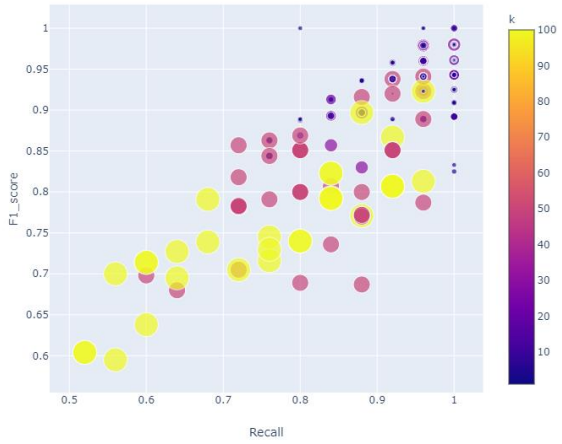

Figure 10 – scatter plot related to F1_score and Recall

Likewise, Figure 11 shows the scatter plot linking the F1_score and the Precision outcome.

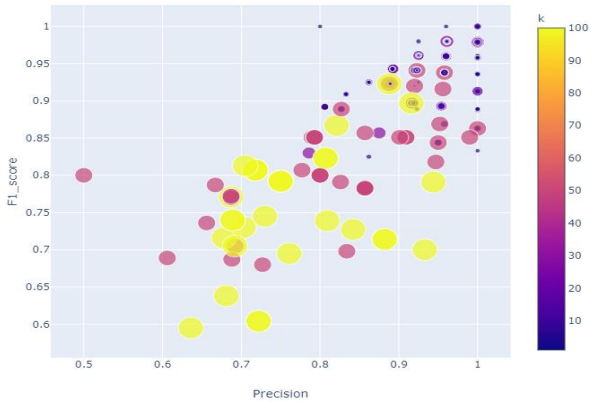

Figure 11 – scatter plot related to F1_score and Precision

Taking these two last figures, one can see the general coherence of the data processing with the KNN method, by the indication of a positive linear arrangement. For instance, figure 10 shows how precision weights in relation to the sum of recall and precision, which is like figure 11, here the relation between the recall and the same summation.

Getting into the specifics, Figure 12 shows the relation between the main metric F1_score and the three values of the sampling frequency (Fs), which is represented in the horizontal axis, according to the values of 'K' (see the color scale on the RHS).

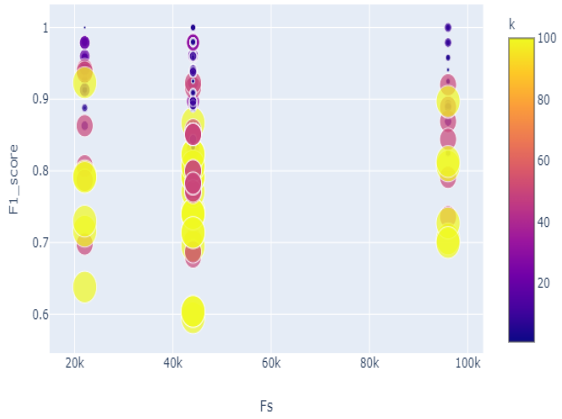

Figure 12 – F1_score as a function of Fs and 'K'.

The scatter plot points out the higher the value of K, the worse the F1_score, for the three sampling frequencies.

Figure 13 covers the relation between the three values of the 'n_fft' and the F1_score, again considering the values of 'K'.

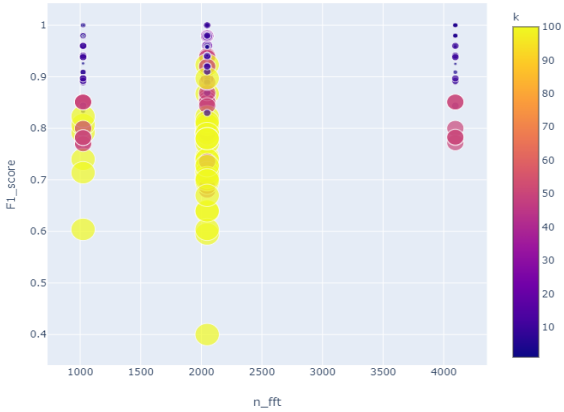

Figure 13 – F1_score as a function of n_fft and 'K'.

The same conclusion can be made. As a preview, the values of K must be smaller than 10 to achieve the best performance in classification. Figure 14 shows the scatter plot for the hop length in the same reasoning.

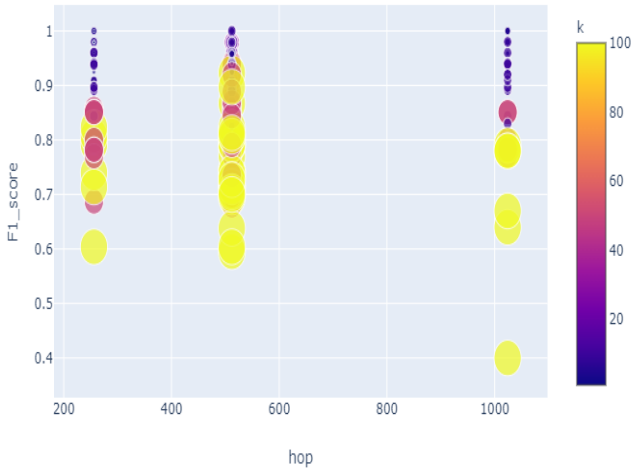

Figure 14 – F1_score as a function of hop lenght and 'K'

4.4 KNN and PCA

For this item, one takes k = 10 as the upper bound, when checking the 'outfitting' and 'underfitting' conditions. After applying PCA = 4 classes, as said above, for the baseline conditions, the KNN algorithm metrics were:

Table 3 – KNN_PCA_4_K_10

	0-cello	1-flute	2-oboe	3-sax	4-trumpet	5-viola
Recall	0.28	0.2	0.44	0.84	0.0	0.36
Precision	0.33	0.18	0.42	0.81	0.0	0.25
F1_score	0.30	0.19	0.43	0.82	0.0	0.30

For the same number of 10 neighbors, without applying the PCA procedure (baseline), the results were:

Table 4 – KNN_PCA_0_K_10

	0-cello	1-flute	2-oboe	3-sax	4-trumpet	5-viola
Recall	1	0.96	0.88	1	0.96	0.88
Precision	0.93	1	1	0.93	1	1
F1_score	0.96	0.98	0.94	0.96	0.98	0.94

The difference between the outcomes above induces to have a deeper look into the case of a higher sampling frequency, say 96 kHz, and the fft window of 4096 and the hop_lenght of 1024, with the PCA equals to 4. Table 5 presents the results:

Table 5 – KNN_PCA_4_K_10

	0-cello	1-flute	2-oboe	3-sax	4-trumpet	5-viola
Recall	0.36	0.24	0.16	0.52	0.04	0.24
Precision	0.28	0.21	0.13	1	0.04	0.23
F1_score	0.31	0.23	0.15	0.68	0.04	0.23

Without using the PCA method, the results for 10 neighbors, under the same conditions above, were as seen in Table 6:

Table 6 – KNN_PCA_0_K_10

	0-cello	1-flute	2-oboe	3-sax	4-trumpet	5-viola
Recall	0.96	0.96	0.96	0.84	1	0.8
Precision	0.89	0.92	1	1	0.87	0.87
F1_score	0.92	0.94	0.98	0.91	0.92	0.83

Checking the results above, one can see a significant difference, which can be evaluated with an additional technique as follows.

4.5 KNN and Feature Importance

When dealing with manyfold features, the present case in fact, one considers evaluating the importance of each feature, to explain their weight in the KNN algorithm. The concept here is to determine how key a specific feature (the 13 MFCCs) works within the prediction phase, and this approach aids to pick up the most important ones, following a specific criterion. Figure 15 presents the individual importance of each MFCC, or its distribution, indicating the last ones have more weight in the prediction. The horizontal axis receives the MFCC identification number, and the vertical axle has the relative importance.

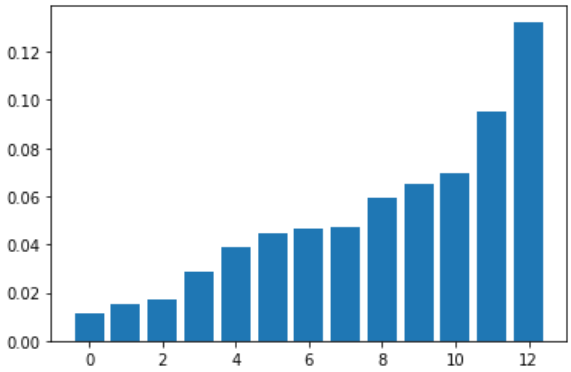

Figure 15 – MFCC & importance distribution

Thus, if we take the importance above 5%, for instance, the last five MFCCs (from #8 to #12) will be enough to explain the KNN prediction, which may help the assessment done with the PCA above, but not changing the core of each MFCC. Proceeding with this plan, Table 7 shows the overall metrics.

Table 7 – KNN and the 5 most important MFCCs

	0-cello	1-flute	2-oboe	3-sax	4-trumpet	5-viola
Recall	0.95	0.88	0.95	0.69	0.55	0.57
Precision	0.84	0.62	0.70	0.86	0.85	0.70
F1_score	0.89	0.73	0.81	0.77	0.67	0.70

For the baseline case, varying the value of 'K' from 1 to 100, with these 5 most relevant MFCCs, one gets Figure 16, where the highest weighted value of the F1 score was obtained with K = 2. In this case, F1 score considers the overall performance of the KNN algorithm, without focusing any instrument.

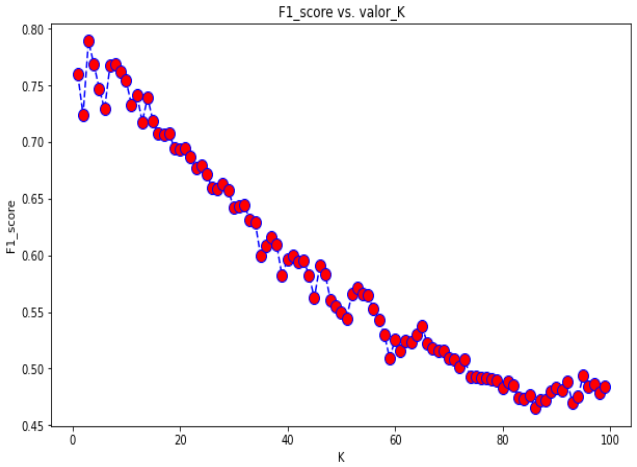

Figure 16 – F1_score (weighted) x K (fs=44.1 kHz)

Thinking on the highest sampling frequency of 96 kHz, with all other parameters as in the baseline case and following the same reasoning above, one winds up with Figure 17.

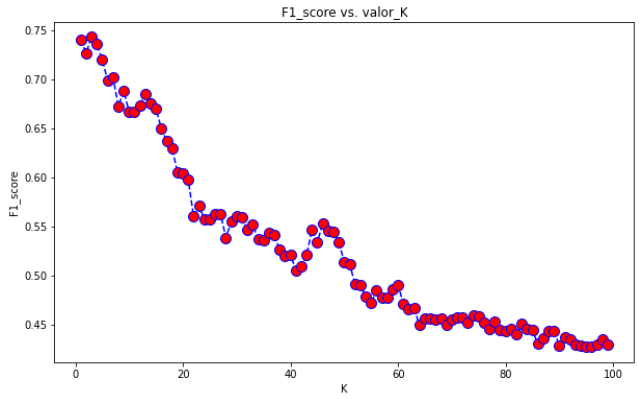

Figure 17 – F1_score (weighted) x K (fs=96 kHz)

The highest value of F1 score was obtained again with K = 2, around 10% smaller than the baseline case.

5. ANALYSIS OF THE RESULTS

The results above took the variation of the input feature as presented above, leading to the following evaluations:

a) For low values of the nearest neighbors, between 1 and 4, the KNN metrics show the presence of 'outfitting', due to the high values found (above of 0.97), although the input data is balanced [6].
b) For nearest neighbor number between 5 and 10, the algorithm metrics relay outside the phenomenon detected above.
c) If the number of neighbors is greater than 10, the F1_score fall deeply, in a ratio close to 0.3 points for every unity increase of 'K', based on Figure 15. Therefore, the maximum value for 'K' shall be 10 for the current application.
d) The higher the sampling frequency, the better the algorithm metrics, according to Figure 12, no matter the 'K' number.
e) Based on the scatter plot from Figure 14, the higher the hop length, the lower the F1_score, as a function of the increase of 'K'. However, there is no significant variation in the end, and the hop length may be around 512.
f) For the influence of the FFT window, the best results were taken for the 2048 option, which is the baseline. Nonetheless, the metric variation was not so remarkable.
g) To address the possible occurrence of overfitting, this work investigated the use of the Principal Component Analysis (PCA) to dim the complexity of the model, considering the MFCCs as the main features. From item 4.2, one has seen 4 main components (eigenvectors) can explain mostly of the math process. This evidence comes from figure 3 and the derivatives present.
h) Nonetheless, the KNN metrics fall in a large amount, roughly 4 or 5 times, in comparison to the baseline case or the algorithm runs without the PCA. Equally important, it also depends on the instrument, the number of neighbors and sampling frequency. For instance, for the trumpet case, one has found metrics equal to 0, which indicates the need for further investigation.
i) The alternative way to access the discrepancy above, throughout the feature importance analysis, shows the KNN prediction scores more reasonable, outside the boundaries of 'over' or 'under' fitting. Thus, it sounds more coherent to use this approach, laying between the upper bound (KNN simple) and the lower bound (KNN_PCA).
j) In terms of instrument performance, the ones with cords (viola and cello) were the most coherent in terms of KNN metrics. The "blow" instruments presented a wide variation of metrics, where 'trumpet' was the worse metric scores set.

The Jupyter Notebooks for this work are available in: https://github.com/65-1157/Aplica-o_NLP_instrumentos_musicais.

6. FUTURE WORK

As explained above, there are issues which require a deeper analysis, as the number of features to be used in the KNN algorithm.

Another subject for further investigation deals with other types of instruments, working with a broader list such as piano and other "blow" instruments. The trumpet case took our attention, and it also deserves a detailed analysis, using comparison of 'Spetrograms' with other keys than the ones used here.

An assessment of other metrics, as the Cohen Kappa metric and the Mathew´s correlation coefficient, is also advisable to explore other figures and to compare with these displayed here.

In a final recommendation, the KNN can be compared to the 'K means' or 'Support Vector Machine' (SVM), unsupervised ML methods, with the same metrics here used and with priority to the F1 score.

7. CONCLUSIONS

This work aided to the complete understanding of how to treat sound, digitally, to identify the presence of different music instrument. The pre-treatment of the dataset to generate spectrograms. The KNN ML algorithm was selected due to its simplicity.

The number of neighbors shall be smaller than 10. The overall performance of the method needs a deeper analysis, because there was a wide metric variation, based on the number of the components or feature importance taken.

The sampling frequency can be selected between 44.1 and 96 kHz for the highest KNN metrics. The hop length shall be around 512, no high variation on the metrics was detected. On the other hand, the best results were seen for the fft window of 2048, which is the baseline.

N. REFERENCES

[1] Rathikarani, V & Dhanalakshmi, P & S., Prabavathy. *Musical Instruments Classification using Pre-Trained Model*. Asian Journal of Electrical Sciences. ISSN: 2249-6297. Vol.9, No.1, 2020, pp.45-48.

[2] Kothe, R.S & Bhalke, D.G & Bhosale, S. *Music Instrument recognition using LPC and K-nearest neighbor classifier*. International Journal of Research and Analytical Reviews. ISSN 2348-1269. Vol.6, Issue 01, Apr-Jun 2019.

[3] Kulkarni, K & Naik, S. *A Review of Music Analysis Techniques*. International Research Journal of Engineering and Technology. ISSN 2395-0056. Vol.5, Issue 03, Mar-2018.

4] Gururani, Siddharth, Mohit K. Sharma and Alexander Lerch. "An Attention Mechanism for Musical Instrument Recognition." *ArXiv* abs/1907.04294 (2019): n. pag.

[5] Bishop, Christopher. (2007). Pattern Recognition and Machine Learning (Information Science and Statistics). Springer. ISBN: 0387310738. pp 186-187.

[6] https://machinelearningmastery.com/feature-selection-with-numerical-input-data/ , accessed on Aug, 25th, 2021.

[7] https://medium.com/@luigi.fiori.lf0303/distance-metrics-and-k-nearest-neighbor-knn-1b840969c0f4, accessed on Aug, 20th, 2021.

[8] https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_knn_algorithm_finding_nearest_neighbors.htm, accessed on Aug, 20th, 2021.

[9] https://towardsdatascience.com/how-to-find-the-optimal-value-of-k-in-knn-35d936e554eb, accessed on Aug, 30th, 2021.

[10] Zhang, Shichao & Li, Xuelong & Zong, Ming & Zhu, Xiaofeng & Cheng, Debo. (2017). Learning k for kNN Classification. ACM Transactions on Intelligent Systems and Technology. 8. 1-19. 10.1145/2990508.

[11]https://towardsdatascience.com/the-5-classification-evaluation-metrics-you-must-know-aa97784ff226 , accessed on July, 27th,2021.

[12] Ben-David, Arie. (2008). About the relationship between ROC curves and Cohen's kappa. Eng. Appl. of AI. 21. 874-882. 10.1016/j.engappai.2007.09.009.

[13] https://doi.org/10.1016/j.engappai.2007.09.009, accessed on July, 30th ,2021.

[14] https://towardsdatascience.com/the-best-classification-metric-youve-never-heard-of-the-matthews-correlation-coefficient-3bf50a2f3e9a, accessed on Sept, 4th, 2021.

[15] https://www.philharmonia.co.uk/resources/sound-samples/, accessed on July, 15th, 2021.

[16] https://ismir.net/

[17] https://www.numerical-tours.com/matlab/ml_1_pca_nn/, accessed on Sept, 4th, 2021.

[18] https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html, accessed on Setp, 4th, 2021.

[19]https://towardsdatascience.com/overfitting-more-than-an-issue-fac2d8b1fb5d, accessed on Setp. 1st, 2021.