Assignment 1- Audio Analysis

Task 1 – Pick 2 descriptors by group, depart from the formula and explain the theoretical expected values for a sinusoid and white noise.

We decided to use the RMS/Energy and Spectral flatness descriptors.

- RMS/Energy: The RMS/Energy descriptor is the energy (loudness) of a sound in a given moment, that energy is directly related to the amplitude of the corresponding sine wave.
    - Sinusoid: assuming the sound wave keeps the same amplitude over time, the RMS graph would look like a horizontal line. If, on the other hand, the sound wave amplitude decreases steadily, aka the sound becomes quieter, the RMS graph will be a straight line with a negative incline.
    - White noise: The RMS graph is a near horizontal line at y=(energy of the sound).

- Spectral flatness: The spectral flatness is a value between 0 and 1 that quantifies how close a sound is to being a tone or a noise, respectively.
    - Sinusoid: Assuming the sound wave keeps the same amplitude over time, the Spectral flatness graph would look like a horizontal line at y~0. If, on the other hand, the sound wave amplitude decreases steadily, aka the sound becomes quieter, the RMS graph will be a curve from ~0 to ~1, it reaches ~1 when the sound wave amplitude is near 0.
    - White noise: The Spectral flatness graph is a horizontal line near y~1.

Task 2 – Visualize and explore different descriptors values for different audio files. Do you think these descriptors are enough to accomplish both classifications?

All of the visualization graphs are visible in the Jupiter notebook.

The most interesting graphs are:

- rmsMax vs rmsMean
- zcrMean vs rmsMean
- zcrStd vs rmsMean
- lat vs anything
- specCentMean vs rmsMean
- specCentMax vs rmsMean
- specFluxStd vs rmsMean/rmsMax
- specFluxMax vs rmsMean
- specFluxStd vs zcrMax

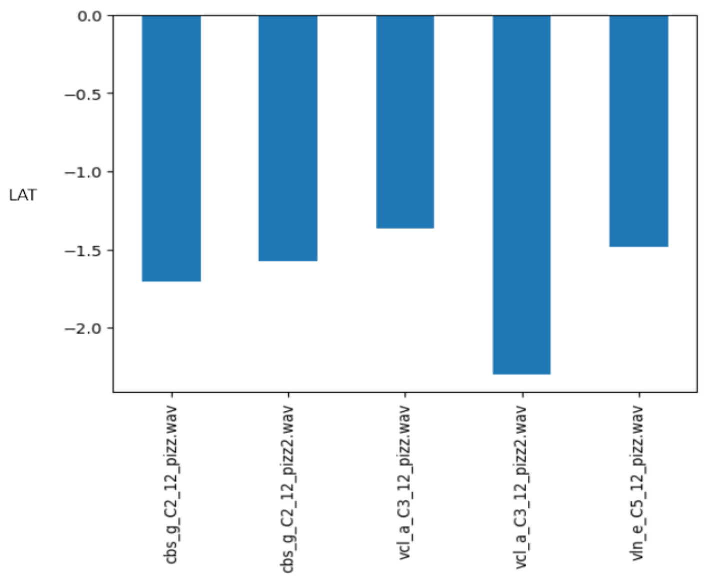
A few conclusions we found are that *LAT* is usually a lot lower in percussive sounds; *ED* is also lower in percussive sounds; *ZCR average* is lower in the percussive sounds.

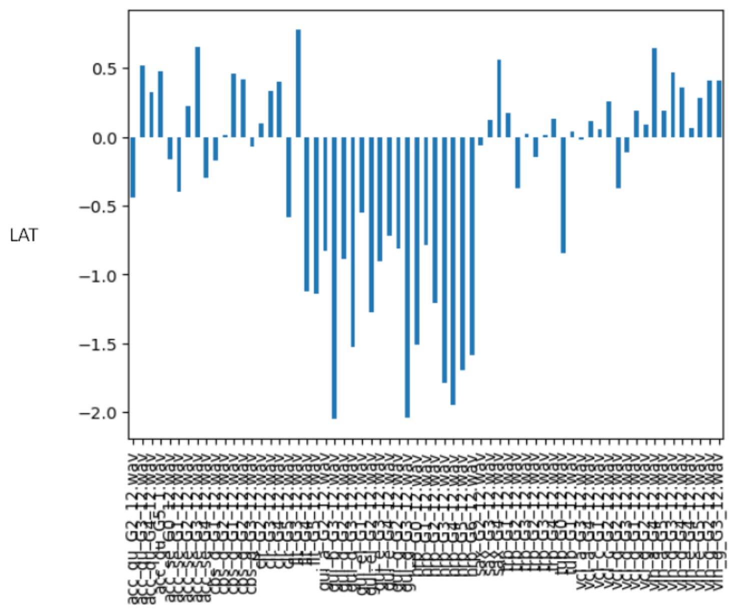
Overall, although it looks like we could identify whether a sound is percussive, we don't believe that with only these descriptors we're able to answer the classification problem with a high precision value. More descriptors would be needed, possibly adding more data might also help solve the problem.

Task 3- Describe in a short paragraph (max 4/5 lines) a sound-based multimedia application that could make use of this set of sound descriptors.

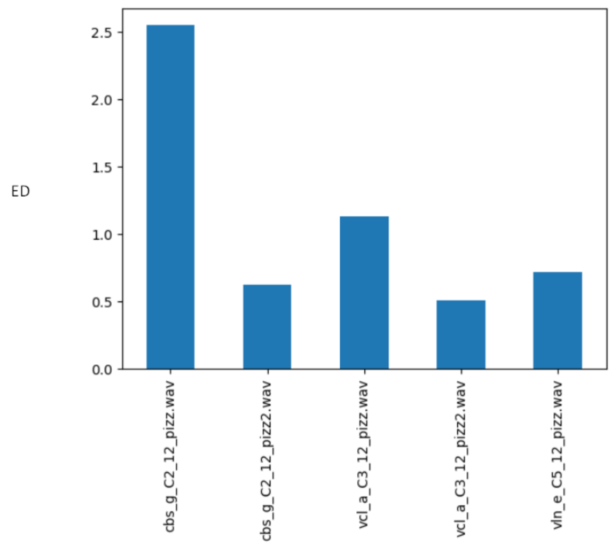
We can use these descriptors in a multitude of different applications, but we would argue that the general public won't give a lot of importance to then, usually people want the simplest and least complicated option. With that in mind, we can see two different applications, the first would be a simple application for the general population where they could upload a sound and the app would give it a score according to what the supposed values are, on the other hand, a second application could be for the music savants, in which they could detect the quality and purity of the sound.

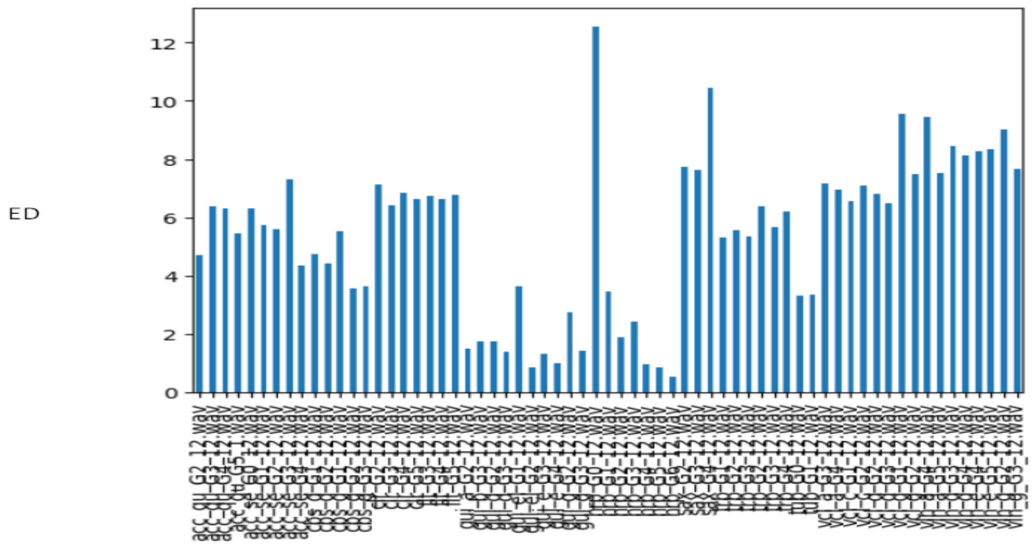
Task 4 - binary classification.





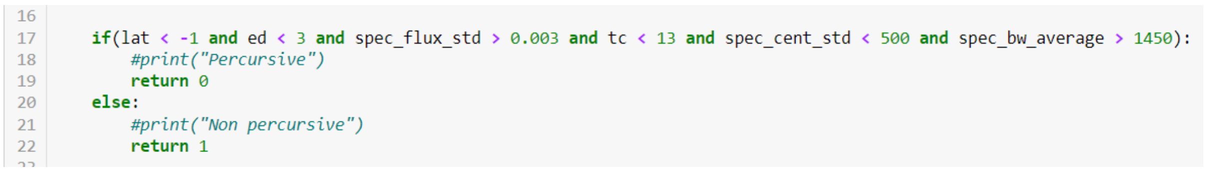
This LAT graphs tell us that a LAT value under -1 is a necessary condition for the sound to be classified as percussive, but it’s not enough.





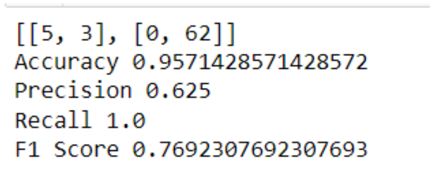
This ED graphs tells us that the percussive sounds have a much lower ED than the not percussive, but then again it’s not enough.

After analysing these graphs (and a few more), we formulated the following function to try and assert the binary classification of a given sound:



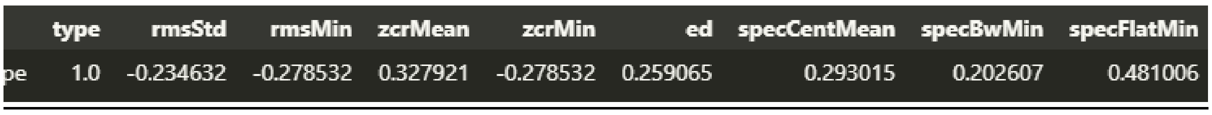
```
    if(lat < -1 and ed < 3 and spec_flux_std > 0.003 and tc < 13 and spec_cent_std < 500 and spec_bw_average > 1450):
        #print("Percursive")
        return 0
    else:
        #print("Non percursive")
        return 1
```

This function gave us the following classification results:



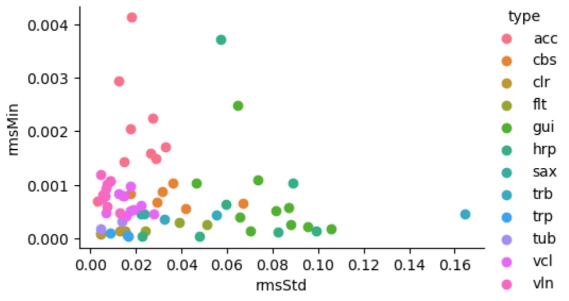
```
[[5, 3], [0, 62]]
Accuracy 0.9571428571428572
Precision 0.625
Recall 1.0
F1 Score 0.7692307692307693
```

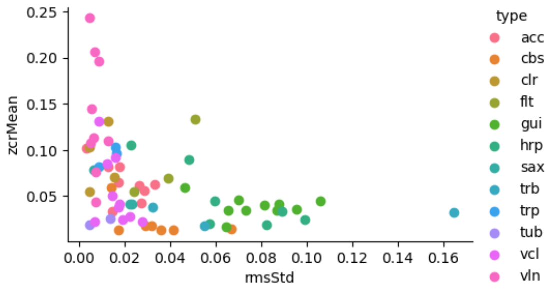
Task 5 -

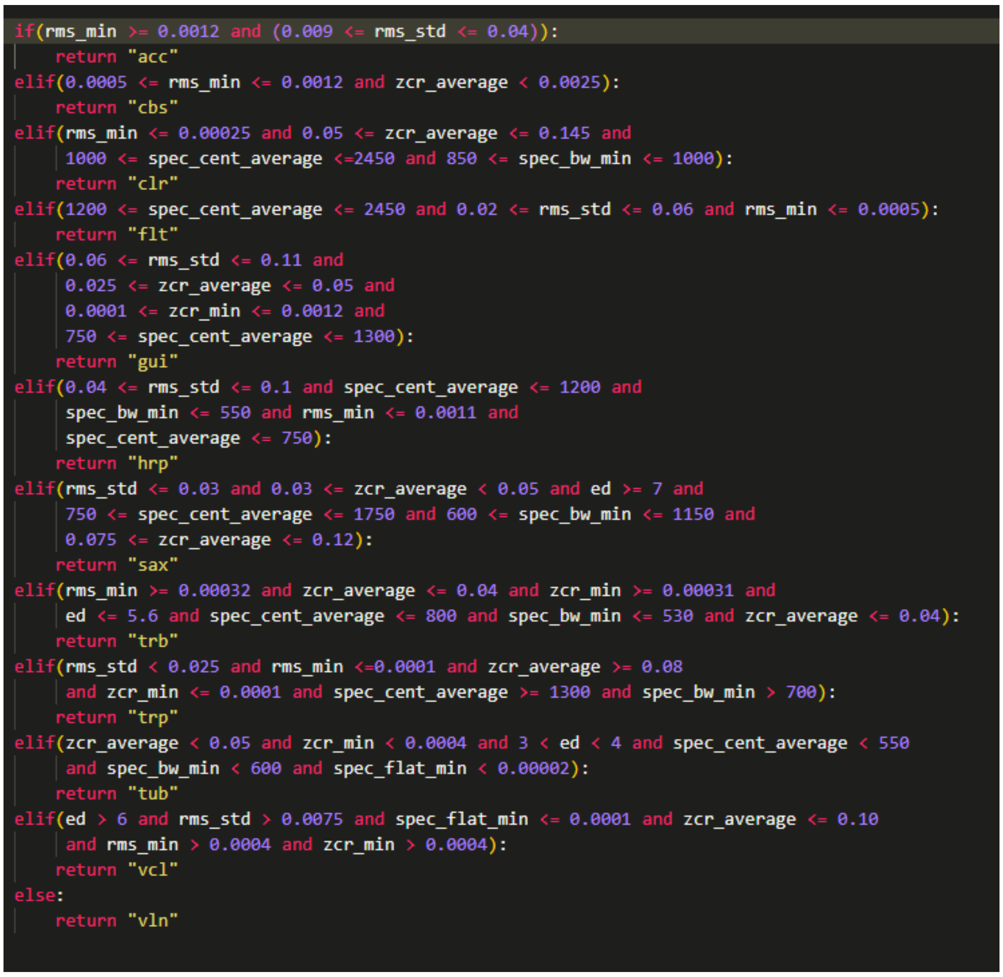


	type	rmsStd	rmsMin	zcrMean	zcrMin	ed	specCentMean	specBwMin	specFlatMin
pe	1.0	-0.234632	-0.278532	0.327921	-0.278532	0.259065	0.293015	0.202607	0.481006

These represent the most impactful attributes when it comes to deciding the type of an instrument (acc, flt, vln…)



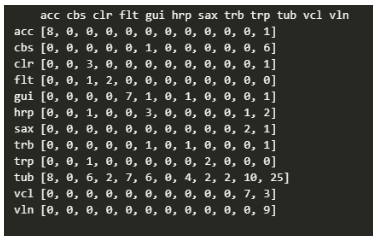




```
if(rms_min >= 0.0012 and (0.009 <= rms_std <= 0.04)):
    return "acc"
elif(0.0005 <= rms_min <= 0.0012 and zcr_average < 0.0025):
    return "cbs"
elif(rms_min <= 0.00025 and 0.05 <= zcr_average <= 0.145 and
    1000 <= spec_cent_average <=2450 and 850 <= spec_bw_min <= 1000):
    return "clr"
elif(1200 <= spec_cent_average <= 2450 and 0.02 <= rms_std <= 0.06 and rms_min <= 0.0005):
    return "flt"
elif(0.06 <= rms_std <= 0.11 and
    0.025 <= zcr_average <= 0.05 and
    0.0001 <= zcr_min <= 0.0012 and
    750 <= spec_cent_average <= 1300):
    return "gui"
elif(0.04 <= rms_std <= 0.1 and spec_cent_average <= 1200 and
    spec_bw_min <= 550 and rms_min <= 0.0011 and
    spec_cent_average <= 750):
    return "hrp"
elif(rms_std <= 0.03 and 0.03 <= zcr_average < 0.05 and ed >= 7 and
    750 <= spec_cent_average <= 1750 and 600 <= spec_bw_min <= 1150 and
    0.075 <= zcr_average <= 0.12):
    return "sax"
elif(rms_min >= 0.00032 and zcr_average <= 0.04 and zcr_min >= 0.00031 and
    ed <= 5.6 and spec_cent_average <= 800 and spec_bw_min <= 530 and zcr_average <= 0.04):
    return "trb"
elif(rms_std < 0.025 and rms_min <=0.0001 and zcr_average >= 0.08
    and zcr_min <= 0.0001 and spec_cent_average >= 1300 and spec_bw_min > 700):
    return "trp"
elif(zcr_average < 0.05 and zcr_min < 0.0004 and 3 < ed < 4 and spec_cent_average < 550
    and spec_bw_min < 600 and spec_flat_min < 0.00002):
    return "tub"
elif(ed > 6 and rms_std > 0.0075 and spec_flat_min <= 0.0001 and zcr_average <= 0.10
    and rms_min > 0.0004 and zcr_min > 0.0004):
    return "vcl"
else:
    return "vln"
```

The plots above represent some of the ones used to decide how to build our instrument classifier (more plots in the Jupiter notebook). The picture on the left clarifies the values used to distinguish and select between instruments.

The figure down below represents the confusion matrix using the previously built classifier.



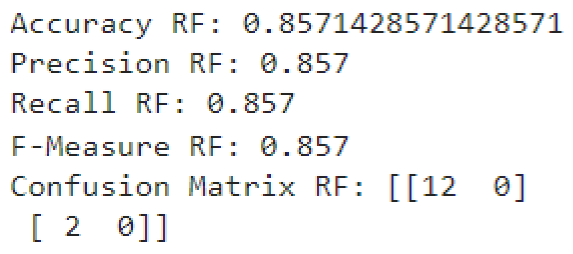
```
     acc cbs clr flt gui hrp sax trb trp tub vcl vln
acc [8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
cbs [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 6]
clr [0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 1]
flt [0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0]
gui [0, 0, 0, 0, 7, 1, 0, 1, 0, 0, 0, 1]
hrp [0, 0, 1, 0, 0, 3, 0, 0, 0, 0, 1, 2]
sax [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 1]
trb [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1]
trp [0, 0, 1, 0, 0, 0, 0, 0, 2, 0, 0, 0]
tub [8, 0, 6, 2, 7, 6, 0, 4, 2, 2, 10, 25]
vcl [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 3]
vln [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9]
```

In total, our classifier decided correctly on 43 out of 70 occasions.

Task 6 - Can you solve tasks 4-5 using Machine Learning?

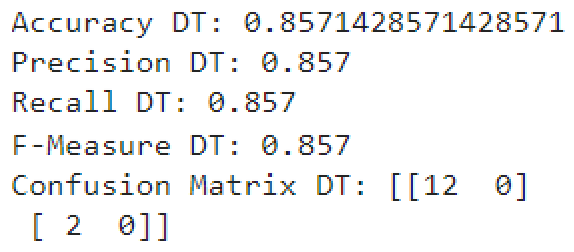
Yes, we can, we implemented 2 different machine learning algorithms!

- Exercise 4:
  1. Random Forest:



```
Accuracy RF: 0.8571428571428571
Precision RF: 0.857
Recall RF: 0.857
F-Measure RF: 0.857
Confusion Matrix RF: [[12  0]
 [ 2  0]]
```

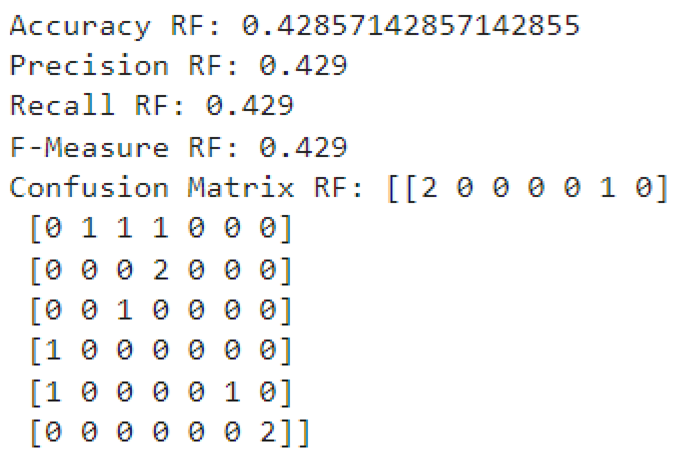
  2. Decision Tree:



```
Accuracy DT: 0.8571428571428571
Precision DT: 0.857
Recall DT: 0.857
F-Measure DT: 0.857
Confusion Matrix DT: [[12  0]
 [ 2  0]]
```

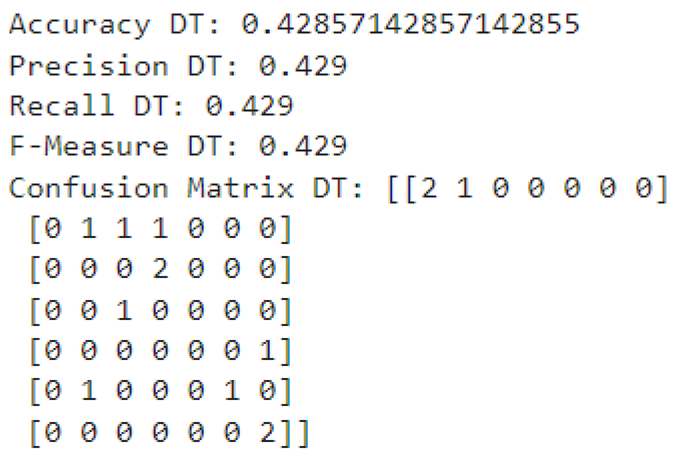
For Exercise 4 the results were equal, that is to be expected when accounting the binary nature of the problem. The quality of the results is quite high for a such a small data set.

- Exercise 5:
    1. Random Forest:



```
Accuracy RF: 0.42857142857142855
Precision RF: 0.429
Recall RF: 0.429
F-Measure RF: 0.429
Confusion Matrix RF: [[2 0 0 0 0 1 0]
 [0 1 1 1 0 0 0]
 [0 0 0 2 0 0 0]
 [0 0 1 0 0 0 0]
 [1 0 0 0 0 0 0]
 [1 0 0 0 0 1 0]
 [0 0 0 0 0 0 2]]
```

    2. Decision Tree:



```
Accuracy DT: 0.42857142857142855
Precision DT: 0.429
Recall DT: 0.429
F-Measure DT: 0.429
Confusion Matrix DT: [[2 1 0 0 0 0 0]
 [0 1 1 1 0 0 0]
 [0 0 0 2 0 0 0]
 [0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1]
 [0 1 0 0 0 1 0]
 [0 0 0 0 0 0 2]]
```

For exercise 5 the results were a lot worse, machine learning algorithms work better with a larger data set and a more restricted classification problem. Still, the algorithms gave similar answers to the problem, but, because the results are so imprecise, there's no specific reason as to why that happens.