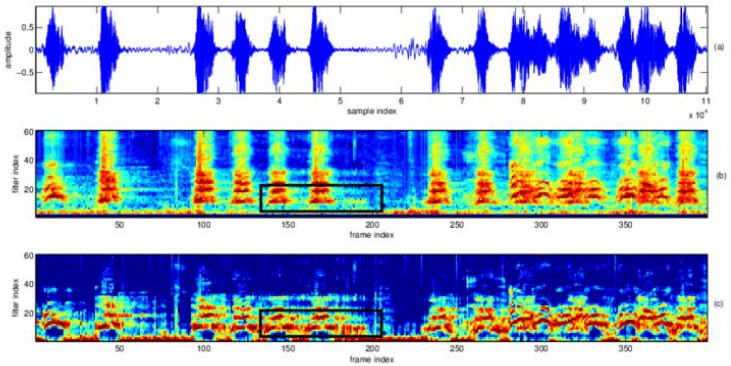
Music Instrument Classification using Machine Learning Techniques

Machine Learning and Artificial Intelligence Project

Denis Tognolo *VR480314*

Audio Analysis Problems

Audio analysis and classification have become very popular researches in the digital world, for both entertainment and business purposes.



Music Classification Task:

- Genre classification
- Mood classification
- Instrument identification
- Music tagging

Sound Classification Task:

- Speech recognition
- Speech filtering
- Failure detection

DATASET

Philharmonia Sound Samples

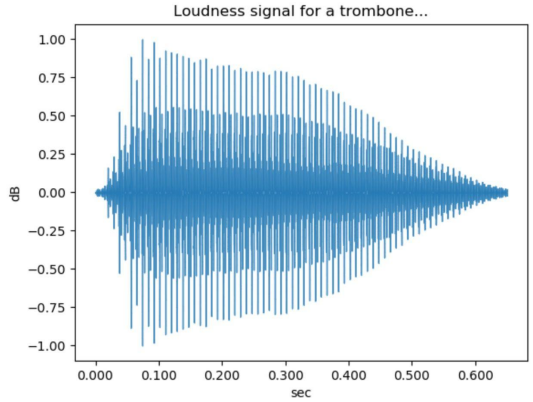
Thousands of free sound samples recorded by Philharmonia musicians.

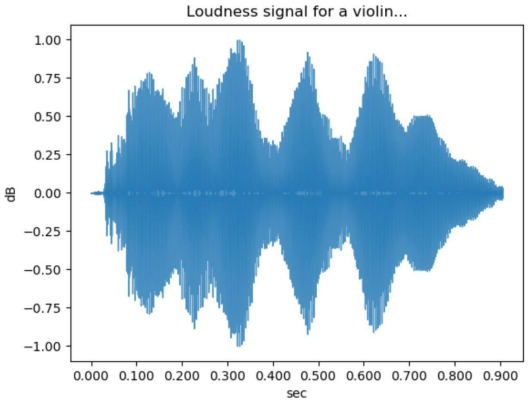
Philarmonia orchestra provides a dataset of over 13000 samples from 15 different instruments.

I actually consider a subset of it, made of 12 classes, with 300 samples for each of them:

- bass clarinet
- contrabassoon
- cor anglais
- double bass
- french horn
- oboe
- saxophone
- trombone
- tuba
- viola
- violin

RAW AUDIO

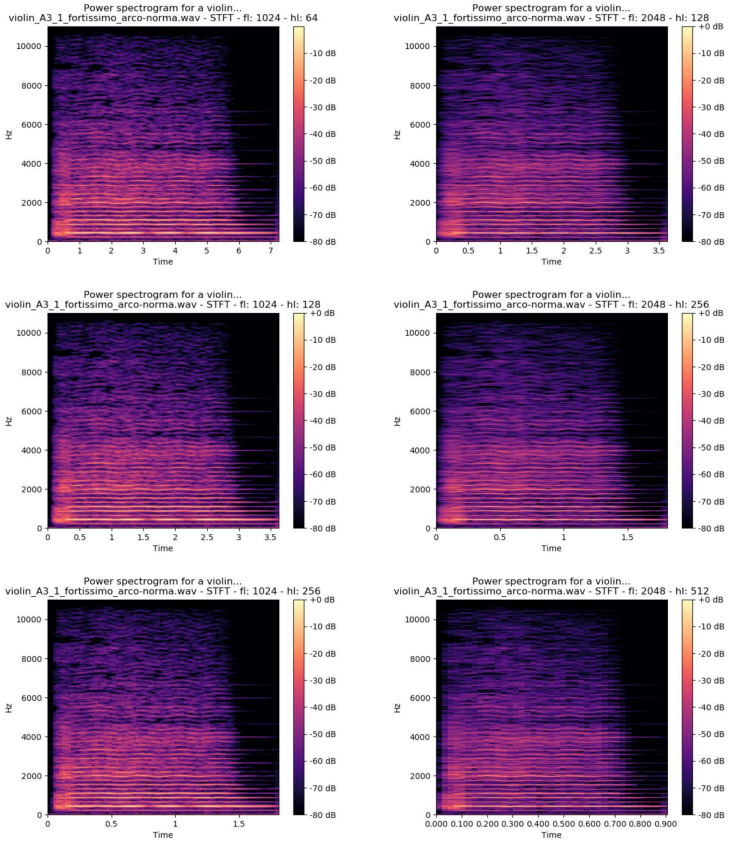




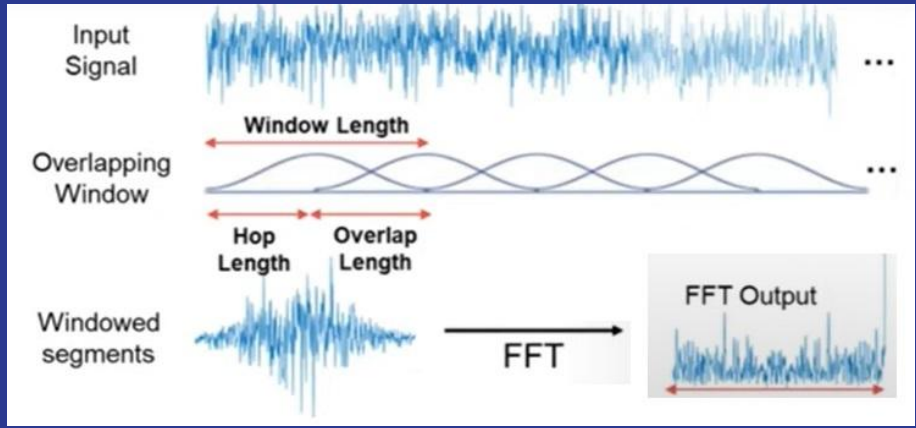
A raw audio signal consist in a measure of the loudness, usually expressed in a log scale (dB), for each time sample.

This signal represents more the way an instrument is played (legato, pizzicato, tremolo...)
than which one is playing.

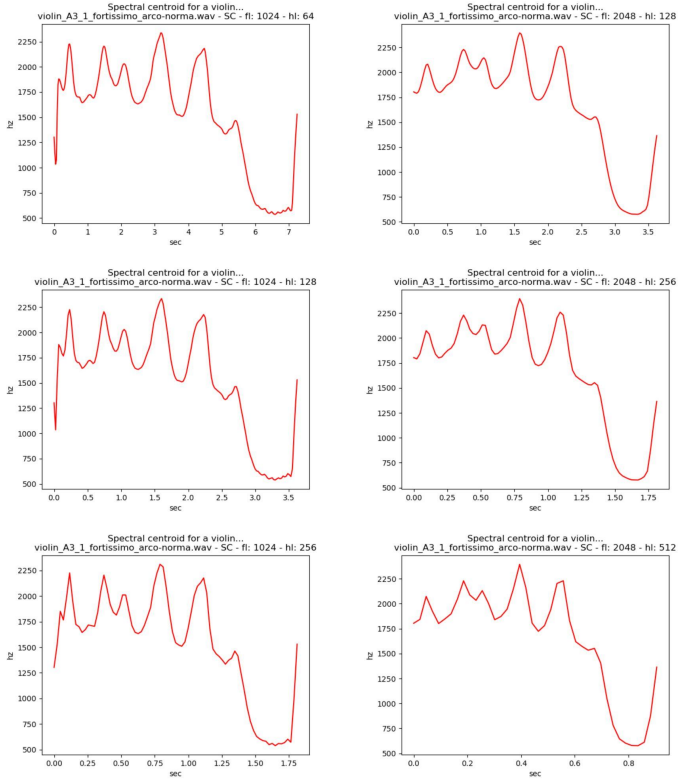
STFT



Short-Time Fourier Transform consist in multiple discrete fast Fourier transform applied to some partially overlapped frames of the original audio sample.



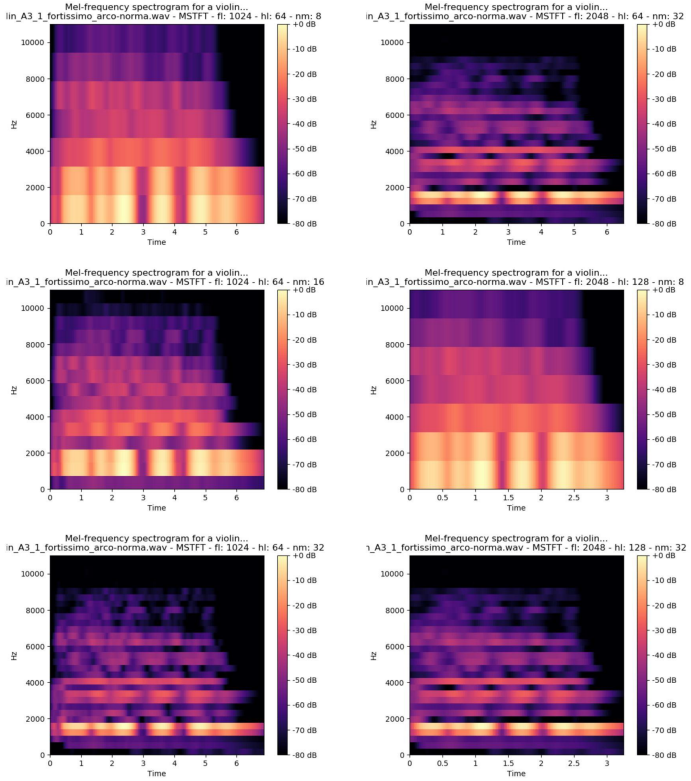
Spectral Centroid



Spectral centroid indicates where the ”center of mass” for a frame is located by calculating the weighted mean of the frequencies present in it.

$$\text{Spectral centroid} = \frac{\sum_{n=0}^{N-1} f(n)x(n)}{\sum_{n-0}^{N-1} x(n)}$$

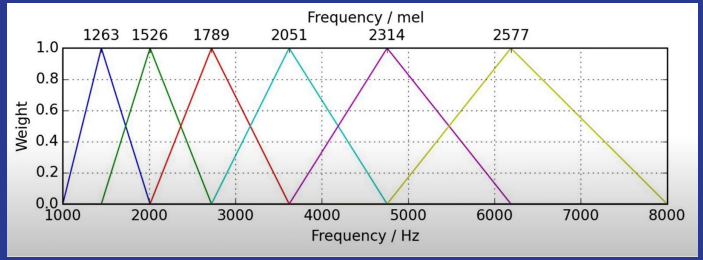
MSTFT



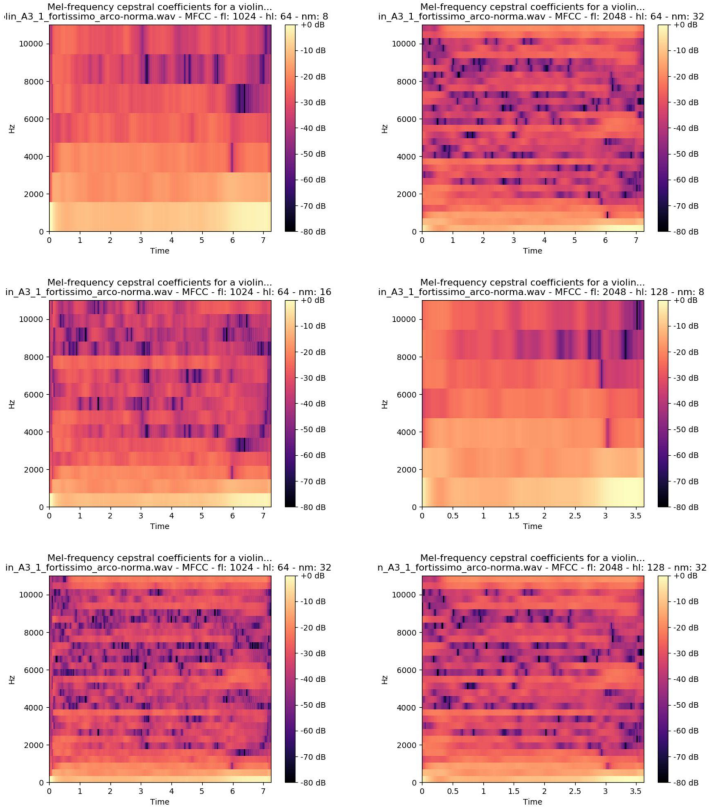
The mel scale is a perceptual scale of pitches judged by listeners to be equal in distance from one another.

$$m = 1127 \cdot \log\left(1 + \frac{f}{700}\right)$$

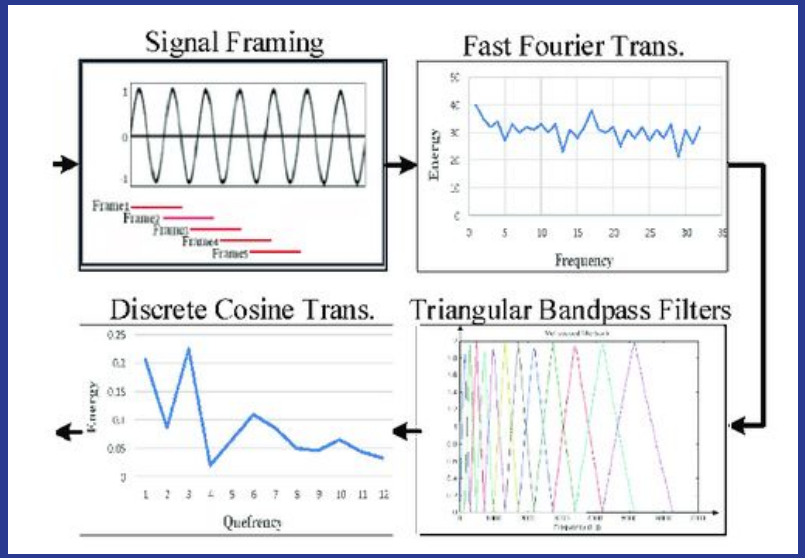
After the scaling the signal is also divided into some bands.



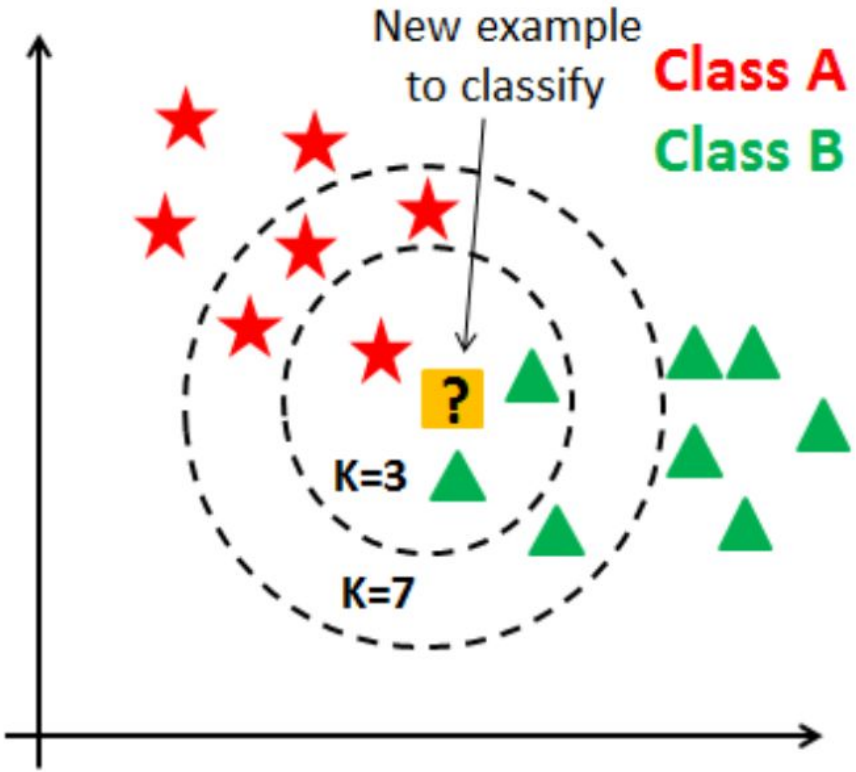
MFCC



Mel-Frequency Cepstral Coefficients extracts for all frames, a small set of features that concisely describe the overall shape of a curve in the frequency-amplitude plane.



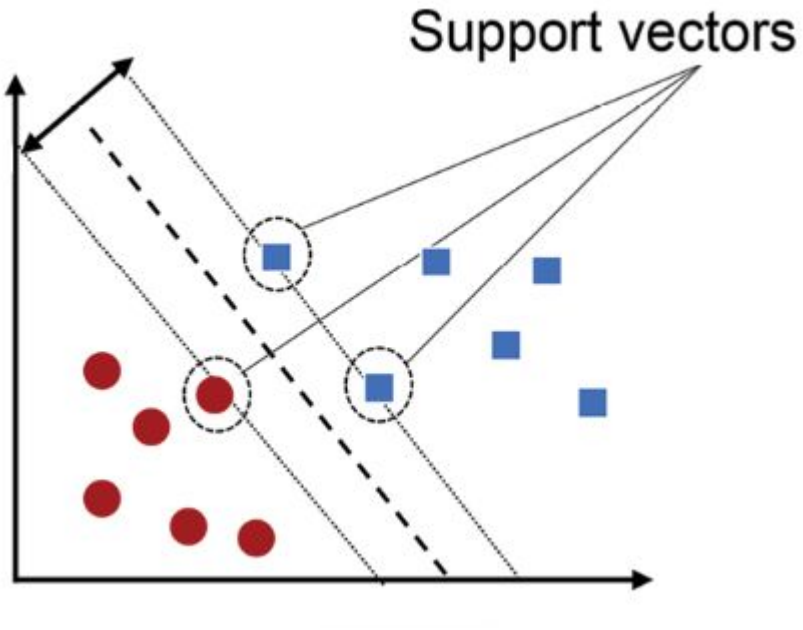
KNN



The k-nearest neighbors algorithm is a supervised learning technique, which uses proximity to make classifications working off the assumption that similar points can be found near one another.

The k value in the k-NN algorithm defines how many neighbors will be checked to determine the classification of a specific point.

SVM



Support vector machines are supervised learning models which main idea is to divide data into two categories, by placing an hyperplane between them. In particular it try to maximize the distance from the nearest data from both groups (the support vectors).

The hyperplane can be of different types depending on the data distribution.

Implementation and Development

The main idea behind the software architecture is to have an *instrument_classificator.py* that collect all the functions and then some other scripts (*main_KNN.py* and *main_SVM.py*) to build a particular model using them.
Then I also set up some bash files (*run_KNN.sh* and *run_SVM.sh*) to run them several times changing some parameters .
At the end of each test all the metrics (accuracy, precision, etc..), are automatically saved in a text file including the considered parameters. Also the figure representing the confusion matrix is saved at the end of each test.

I crafted and run a total of more than 60 test with different configurations.

```
instrument_classification
├── philharmonia-samples
│   ├── bass_clarinet
│   ├── cello
│   └── ...
├── images
│   ├── trombone_A2_1_mezzo-forte_norma.wav ... 16.png
│   ├── violin_A3_1_fortissimo_arco-norma.wav .. 16.png
│   └── ...
├── results
│   ├── KNN - SC - fl: 1024 - ...  5.txt
│   ├── conf_mat_KNN - SC - fl: 1024 - ... 5.png
│   └── ...
├── instrument_classificator.py
├── main_KNN.py
├── main_SVM.py
├── visualize_features.py
├── run_KNN.sh
├── run_SVM.sh
└── run_visualization.sh
```

Test and Results

	KNN		SVM	
	Accuracy	F-Score	Accuracy	F-Score
Raw Signal	0.88	0.23	0.86	0.1
STFT	0.96	0.77	1.0*	0.99*
Spectral Centroids	0.92	0.52	0.90	0.39
MSTFT	0.96	0.76	0.98	0.87
MFTCC	0.94	0.62	0.97	0.83

- All the presented results consider the best parameters for that particular setup.
- * marks a really slow training (~15 mins)

Conclusion

KNN

- Good accuracy in general
- Fast training even for large features
- Best setup considering k=4 neighbours

SVM

- Excellent accuracy in general
- Slow training for large features
- Best setup considering a 5th order polynomial kernel

Regarding **features**, the worst one for this task results to be the spectral centroid, and the best one the MSTFT.
That's because MSTFT achieve high performances with small features, that makes the difference while using SVM.