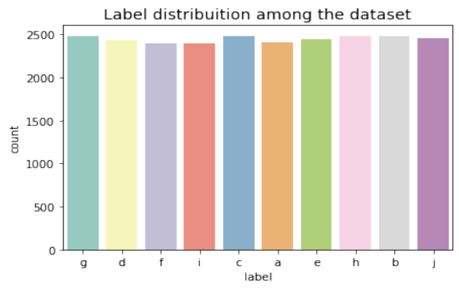
Data Science Lab: Process and methods
Politecnico di Torino

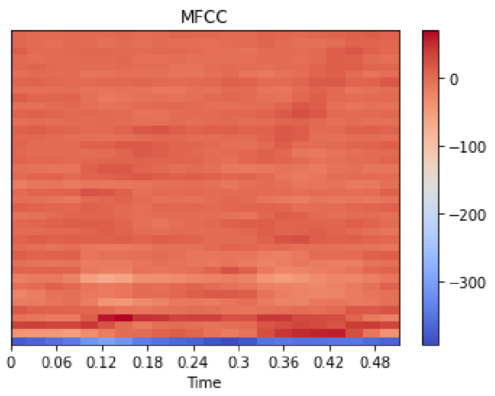
Project report
Student ID: s265348

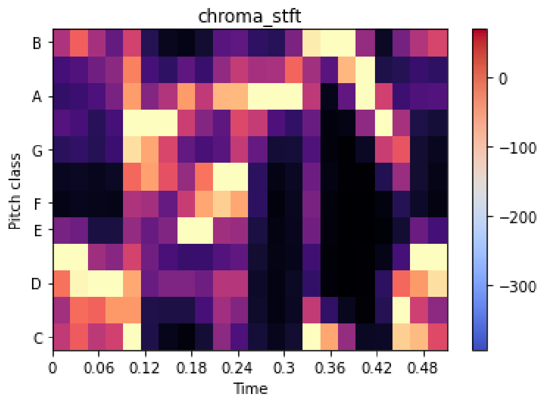
Data exploration

The dataset is composed by 30,821 recording extracted from different audiobooks: each recording last approximately 0.5 sec and is sampled at 24 kHz(for approximately 12000 samples per recording). The recording are collected from a total of 10 speakers identified by the label a,b,c,d,e,f,g,h,i,j .
The train set contains 24449 recordings grouped in 10 different folder (each corresponds to a label) in WAV format , the evaluation set contains 5832 recording in WAV format.
The train test is balanced among labels as we can see in the figure below.

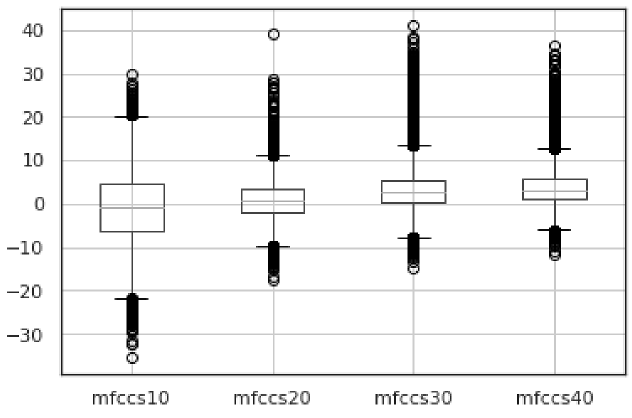


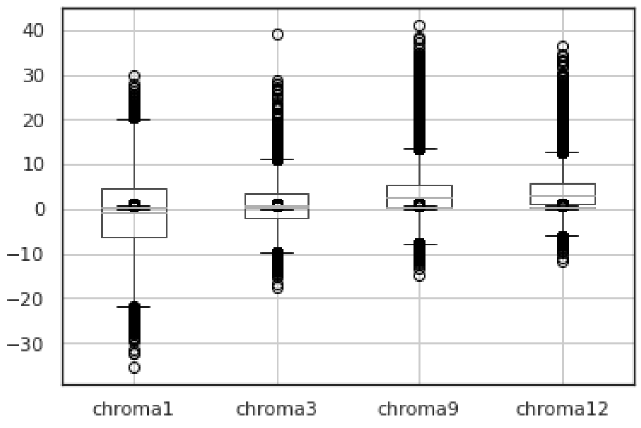
I also inspect the length of each file of the training set the min value is 0.08 s, the max value is 0.5 s , the mean value is 0.499s. I can conclude that quite all the data has the same duration. There are only 7 recording for the training set that has a duration inferior than 0.48 sec and 3 for the evaluation set. But I will work in the frequencies domain and the same number of feature will be extracted for each record independently of its length. So I opt to consider as error this short record (I dropped them out , only for the training set of course).
I decide to work in the frequencies domain instead of the time ones, because after reading lot of useful paper seems to be the best choice in order to represent wave with proper features. I choose the librosa package in order to extract different type of features . In particular I work with the following features:
1-MFCC : is a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a non-linear mel-cale of frequency. Our voice/sound is dependent on the shape of our vocal tract including tongue, teeth etc. If we can determine this shape accurately, we can recognize who is speaking. MFCC is a representation of the short-term power spectrum of a sound, which in simple terms represents the shape of the vocal tract. For each wav file 40 MFCCs are extracted;
2-Chroma-gram :they capture harmonic and melodic characteristics of music. For each record the 12 chroma-gram are extracted.
I also extract zero_crossing, spectral_bandwidth, spectral_centroids ,roll_off . But later I discover that those are not so meaningful. In order to represent chroma-grams and mfccs I plot some examples for just one recording:





I also plot the box-plot representation of some key value as follow :



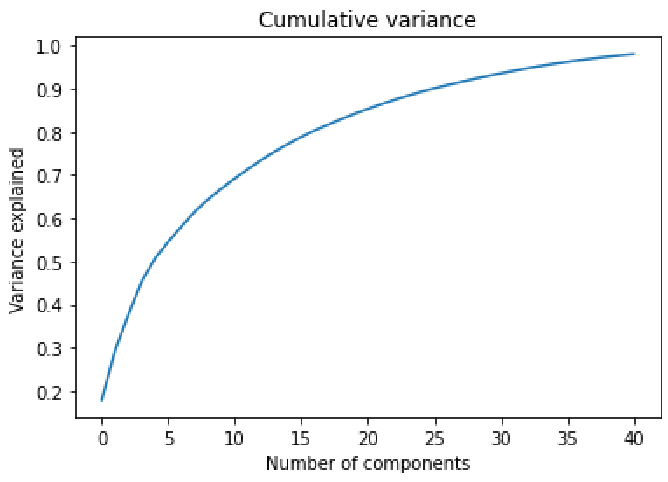


As we can see the variance decreases among with the coefficient index (10.20,30 /1,3,9…) , so in the next step we expect that those coefficient influence more during principal component analysis. I also adopt an outlier detection strategy on the training set thanks to the LocalOutlierFactor() and modules provided by sklearn setting the number of nearest neighbour to 300 . I also try with RandomIsolationForest() that is tree based algorithm for outlier detection , but the first ones seems to work better, so all the further steps are obtained after using this. After this phase the dataset is composed by 24200 record.

Preprocessing

The first step of my preprocessing phase is the data normalization, a really important set having a data set so spread in term of values. I try both the StandardScaler() module and the RobustScaler() module provided by sklearn. The first one normalize features thanks to the mean and standard deviation values, the second ones by the median and interquartile values. In this particular case the RobustScaler() seems to be more stable and I will use it for the next steps.
Not all the features are relevant in out analysis , some of them can be redundant. In order to reduce the number of features (actually 56) I opt for the PCA. The PCA (Principal Components Analysis) is a technique of dimensionality reduction that try to preserve the highest variance among the data but reducing the number of features.
Principal components are a new coordinate system. Given data on D variables, the hope is that the data points will lie mainly in a linear subspace of dimension d < D. The PCs decompose the total variance of the data, so the sum of the variance among all the PCs corresponds to the total variance among original data.
The first PCs is the one which have the higher variance, the last PCs is the one having lower variance.
Each PC is an orthogonal linear transformation of the original data.

In the plot below we can see that first 40 component represent quite the 98% of the variance and so the others can be discard.



I also try to adopt a noise-reduction strategy tanks to apply a filter on the MFCCs coefficient : the filter receives an audio matrix and returns the matrix after gain reduction on noise.
But it seems it doesn’t give the expected improvement and computationally it is really hard. So in term of balance of performance and computation time appears to be useless.
I don’t need for this phase to fill missing values because as said before librosa provide a frequencies representation of the wave form in input even if it’s not ‘complete’.

Algorithm choice

In order to choose the appropriate model for the speaker recognitions task some different classifier has been evaluated.
First of all I exclude the use of the KNN classifier , because really susceptible to the dimensionality. In this case we have after the PCA still 40 dimensions and so the data tends to be sparse in a such high dimensional space.
I also exclude to use Naive-Bayes classifier and the Logistic regression that works better with binary classifications. This first 3 classifier are tested and the performance in terms of accuracy score confirm those preliminary observations. The parameters are all set with the default values.
I finally opt for comparing the Random Forest Classifier and SVM classifier (with non linear kernel in order to better fit the non-linear problem).
In order to choose between the two algorithms I train both the model on a 80% of the train dataset , and test it on the left 20%. This set is realized with the train_test_split() module provided by sklearn. With the Random Forest Classifier (using the default settings) is reached an f1_score of 0.9351. With the SVM with radial basis kernel is reached an accuracy of 0.9562 .
I finally opt for tuning with the SVM classifier because of the best performance.
In the following graph we can see the confusion matrix that confirm the best performance of the SVM classifier.

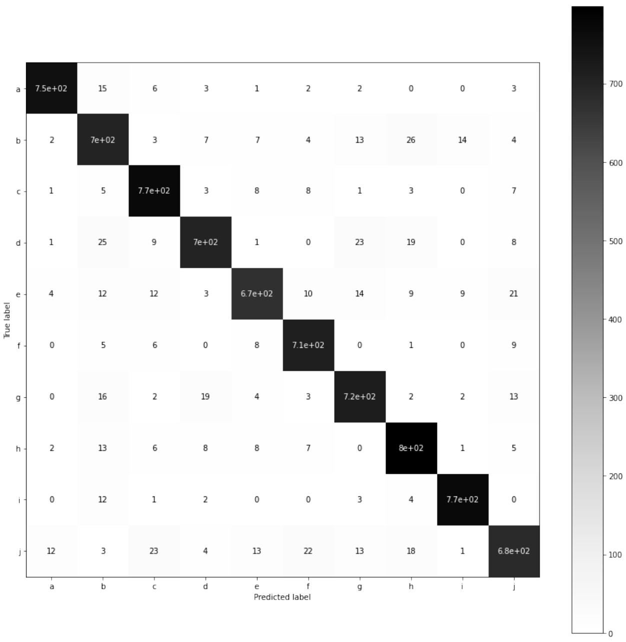


Figure 2: Random Forest

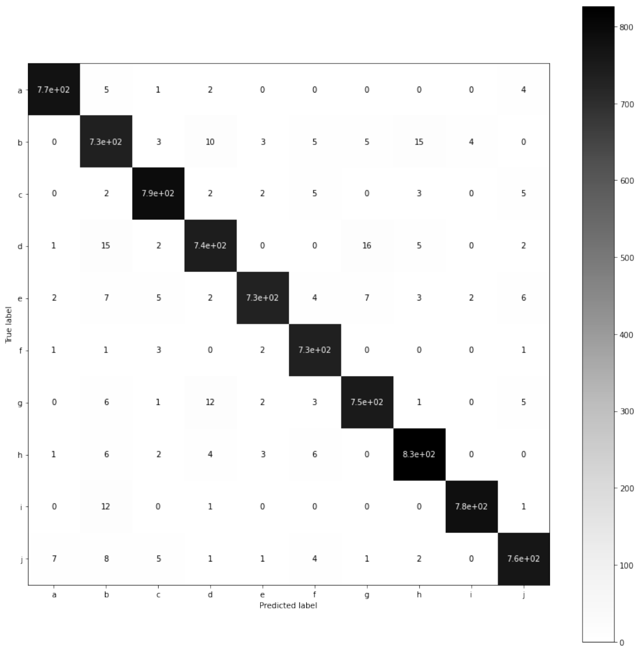


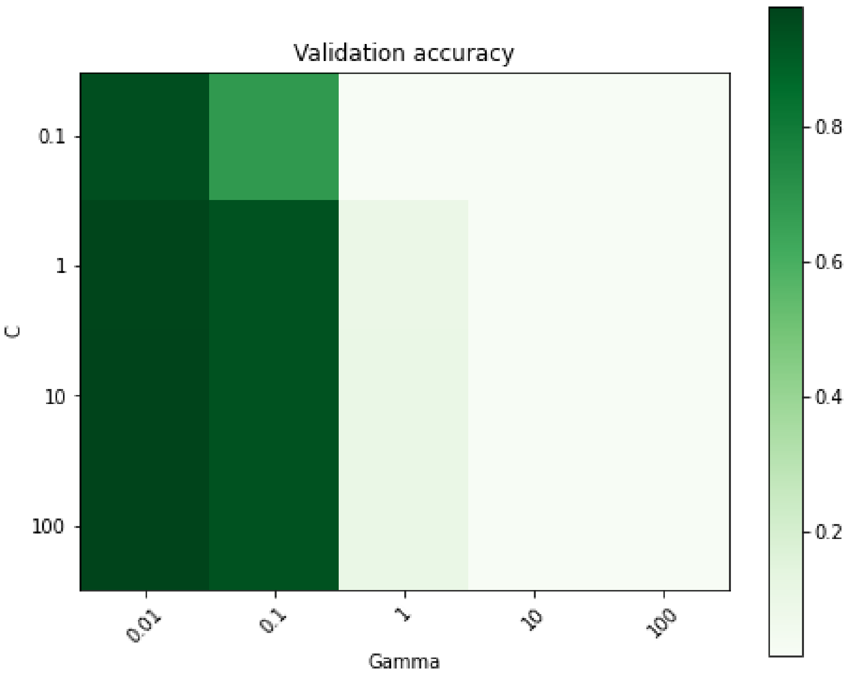
Figure 1: SVM

Tuning and Validation

For the SVM Classifier a 5 fold grid search cross validation has been used to fine tune the following hyper-parameters:

- **C**=0.01, 0.1, 1, 10, 100;
- **gamma**=0.001, 0.01, 0.1, 0.0001;
- **kernel**='rbf',’poli’.

I choose the stratified k-fold cross validation in order to avoid the over-fitting of the data.
The result are stored in the following heat-map (I show only the case with ‘rbf’ kernel):



The best f1_score is registered with C=10, gamma=0.1 and radial basis kernel.
With this hyper-heptameters I also study some different score as follow:

Label	**Precision**	**Recall**	**f1-score**	**Support**
a	0.98	0.99	0.98	969
b	0.98	0.98	0.98	963
c	0.98	0.98	0.98	1032
d	0.96	0.98	0.97	913
e	0.97	0.97	0.97	979
f	0.97	0.99	0.98	945
g	0.99	0.97	0.98	964
h	0.98	0.98	0.98	999
i	0.99	0.99	0.99	912
l	0.98	0.96	0.96	1011

As we can see from the table the different scores are well spread among the different labels. So the classifiers fit well all the classes, and the decision boundaries are also balanced.
The mean value of the accuracy is 0.98.
After applying the whole model trained on the training set on the evaluation set , and submitting the result I reached an f1_scored of 0.9564. There is a loss in term of this score probably due to the over-fitting of the data.

References

https://scikit-learn.org/stable/
https://librosa.org/doc/latest/index.html
https://seaborn.pydata.org/generated/seaborn.countplot.html
https://github.com/dodiku/noise_reduction/blob/master/noise.py
https://www.intechopen.com/books/intelligent-system-and-computing/voice-identification-using-classification-algorithms
https://towardsdatascience.com/ok-google-how-to-do-speech-recognition-f77b5d7cbe0b