Clustering and Classification of IEEE audio signal for information Extraction

Karun Dhingra, MS, Data Science (IU)

Independent Studies Report Submission under **Professor Donald Williamson**

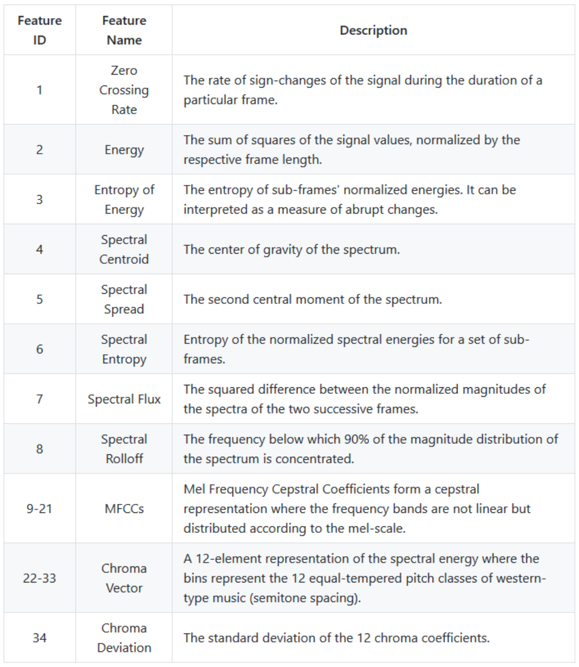
Abstract:

With the idea of audio signal processing from the point of view of clustering and classification, I have designed this report for using a IEEE audio signal dataset and some noise to extend it to a mixed signal dataset for trying out various algorithms like K Means, Gaussian Mixture Model (for clustering) and Multi-Layer Perceptron classifier to train these signals. I have kept the scope as for the level of SNR as my label for classification. Effective classification and classification is observed using PCA.

Keywords: K Mean, GMM, MLP, PCA

Data Configuration and Preparation:

PyAudioAnalysis is a Python library covering a wide range of audio analysis tasks. Through pyAudioAnalysis we extracted audio *features* and representations (e.g. mfccs, spectrogram, chromagram). I used this library to extract 34 features for each signal. Each signal turned out to be a matrix of $34 \times m$, where m depends on the length of the signal. For creating a matrix of interest, we flatten the matrix into a row, that will create a matrix for $15000 \times (34 * n)$. As m is different for each signal we need to bring all signals down to common number n, to make an even matrix for further processing and analysis.



Feature ID	Feature Name	Description
1	Zero Crossing Rate	The rate of sign-changes of the signal during the duration of a particular frame.
2	Energy	The sum of squares of the signal values, normalized by the respective frame length.
3	Entropy of Energy	The entropy of sub-frames' normalized energies. It can be interpreted as a measure of abrupt changes.
4	Spectral Centroid	The center of gravity of the spectrum.
5	Spectral Spread	The second central moment of the spectrum.
6	Spectral Entropy	Entropy of the normalized spectral energies for a set of sub-frames.
7	Spectral Flux	The squared difference between the normalized magnitudes of the spectra of the two successive frames.
8	Spectral Rolloff	The frequency below which 90% of the magnitude distribution of the spectrum is concentrated.
9-21	MFCCs	Mel Frequency Cepstral Coefficients form a cepstral representation where the frequency bands are not linear but distributed according to the mel-scale.
22-33	Chroma Vector	A 12-element representation of the spectral energy where the bins represent the 12 equal-tempered pitch classes of western-type music (semitone spacing).
34	Chroma Deviation	The standard deviation of the 12 chroma coefficients.

Table 1: Features used for clustering and classification

Unsupervised Approach:

So, the idea is to reduce this data to a level where we can achieve the training of the Neural Network without too much of data. One thing that comes to mind is clustering that can represent the same set of data with a representative and hence can be used instead of all that data. I took a clean set of noise signals (720) using these signals to mix with 10 different frames of noise signal on 3 different SNR level, creating 15000 mixed signals **[3][5]**.

Supervised Approach:

I take the same set of data, but we label them as per the SNR. For the sake of creating a dataset for our neural network by taking use of the label 1 for data having a positive SNR, and for the counter argument we take 0 as the classification of the audio signals that have SNR is negative. We are using the same dataset one we used for the unsupervised learning.

Audio Clustering Idea:

[1] There are three types of clustering algorithms for high-dimensional data: attribute reduction, subspace clustering, and co-clustering. One reduces the data dimensionality with attribute conversion or reduction and then, performs clustering. The effect of this method is heavily dependent on the degree of dimension reduction; if it is considerable, useful information may be lost, and if it is less, clustering cannot be done effectively. Other method divides the original space into several different subspaces and searches for cluster in the subspace. The third one implements clustering iteratively with respect to the content and feature alternately. The clustering result is adjusted as per the semantic relationship between the theme and characteristic, realizing a balance between data and attribute clustering

Curse of Dimensionality:

When we try to apply clustering to this matrix of 15000×2210 dimensions, it shows only one cluster as the concept of distance becomes less precise as the number of dimensions grows, since the distance between any two points in a given dataset converges. The discrimination of the nearest and farthest point becomes in a way meaningless to the observation for the algorithm.

PCA:

So, we moved the next best option of reducing the dimension, PCA comes out to be the best possible choice. We must first normalize the data for better reduction and avoid any outliers. Applying PCA I reduced the dimension to a few number of principle component as it represents the percentage of variance in decreasing order. I choose 3 of them, so the final matrix came out to be 2210×3; multiplying it with the original matrix of 15000×2210 it gives us the reduced dimension matrix 15000×3. The 3 Principle Component's variance amount to a value of 94%. The plot looks quite scattered but with not too much outliers which is good cue to go forward.

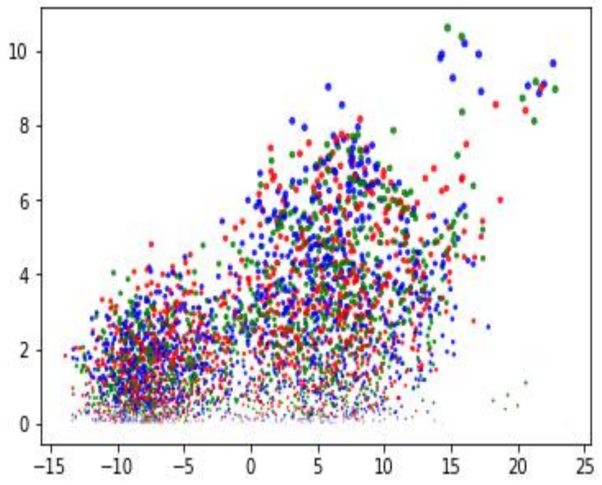


Figure 1: Data representation after PCA

K Means:

We have data set of 15000 rows with 3 PC's that we can cluster to see and find if there exists a right representative of these set of data. As our first choice of clustering by default is K mean, we go ahead and try to choose different values of clustering from 2 to 7. I choose k-mean ++ **[2]** that are in algorithms term is the version of Lloyd Algorithm. MSE is

Distance/Clusters	2	3	4	5	6	7
Cosine	0.150	.1286	.1148	.1286	.1291	.115
Correlation	.1283	.1126	.1031	.0827	.0907	.0953

taken as the criteria for choosing the right number of clusters. Above is the table for the comparison between the two distance function cosine and correlation to compare the best (leas MSE) configuration for the cluster formation and take that value of cluster as K.

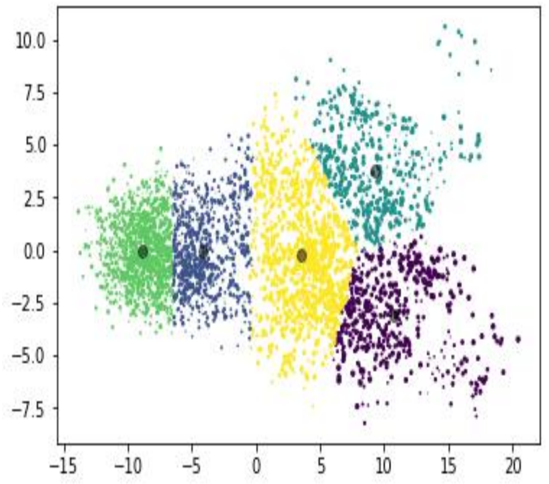


Figure 2: k mean cluster with centers

The result came out side for 5 clusters for we have select to the right distance criteria as well. I tried the Euclidean, Cosine and Correlation. We can measure for the different errors for the different set of the distances.

GMM Clustering:

GMM clustering is more flexible because you can view it as a *fuzzy* or *soft clustering* method. The algorithm that fits a GMM to the data can be sensitive to initial conditions. K-Means doesn't take into account the covariance of our data. K-Means would regard them as being equal, since it uses Euclidean distance. But it Seems certainly more likely that the point that follows the trend should match closer to the Gaussian than the point that doesn't. Hence it is always a wise choice to take GMM into account while clustering to check if there is any drastic change in results.

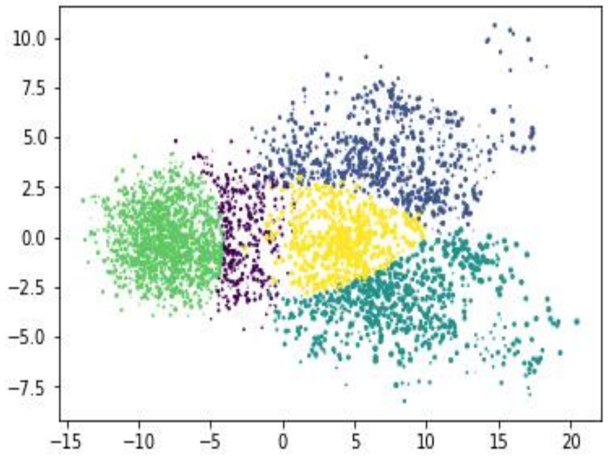


Figure 3: Clusters using GMM

Turned out for the GMM the data seems to fit just fine with 5 clusters for the covariance type as diagonal, it can be seen the for the outer layer of the clustering is same for both GMM and k means **[2]**. So, we are on the right tract as the dataset probabilistically has the same behavior as that of dissimilarity. There are multiple options in GMM as provided by scikit learn to try variations for GMM like for covariance type: 'spherical', 'tied', 'diag', 'full'.

Classification using MLP:

As we see in our dataset creation section we have a set of classified labels on the basis of SNR as the label (1 if SNR is positive and 0 if it's negative). I have used MLPClassifier from the scikit learn to train the data for a better activation function used ReLU. We can try with different variation of hidden layers. With a variation of data split of 70 – 30 we have created a confusion matrix. The dimension of the dataset is 15000 * 2210. It can be argued that it comes under high dimension. And we know that the behavior of the high dimensional data is different from that of the lower dimensions.

precision	recall	f1-score	support	
0	1.00	1.00	1.00	982
1	1.00	1.00	1.00	908
Avg/total	1.00	1.00	1.00	1890

Table 2: prediction matrix on actual dataset

It can be seen that something is not right over here looking at the precision matrix. It happened as we had a doubt for the higher dimension of our dataset.

Fewer Features (PCA):

For we assumed that MLP **[5]** is capable enough to handle this kind of arrangement as it takes care of the required features and ignore the irrelevant one, but I tried to compare the results of the MLP that we implemented with another dataset of the PCA applied to this data set a we have already calculated before in section 3.

precision	recall	f1-score	support	
0	0.98	0.95	0.97	972
1	0.95	0.98	0.97	918
Avg/total	0.97	0.97	0.97	1890

Table 3: prediction matrix after PCA

Conclusion:

We can see that the accuracy is good with the MLPClassifier which is around 97% on the reduced data set. It's obvious that the high dimensionality of the data doesn't let the classifier work fine. Coming to the clustering We try to setup the whole clustering around the original audio signal data converted to a matrix to the dimension of the extraction, but the high dimensionality of the matrix caused problem for the implementation of these algorithms as well, so we moved to the choice of taking a dimensionality reduction approach using PCA. It came out to be obvious as there are prominent features that take up the cluster area, it is hard to tell which features are more prominent than the others, but we can always go ahead and check using some f-test to see the impact of adding a feature. The same reduced dimension data is used to train a MLP Classifier which gave us a good accuracy for classification of the SNR on the positive and the negative values. Further work can include using various levels of Hidden Layers to see the effect on the training of the network.

References:

[1]: EURASIP Journal on Audio, Speech, and Music processing 20172017:26 https://doi.org/10.1186/s13636-017-0123-3© The Author(s). 2017 Wenfa Li, Gongming Wang and Ke Li

[2]: http://scikit-learn.org/stable/modules/clustering.html

[3]: Complex Ratio Masking for Monaural Speech Separation - Donald S. Williamson, *Student Member, IEEE*, Yuxuan Wang, and DeLiang Wang, *Fellow, IEEE[4]*

[4]:https://www.kdnuggets.com/2016/10/beginners-guide-neural-networks-python-scikit-learn.html/2

[5]: Wang Y., Narayanan A. and Wang D.L. (2014): On training targets for supervised speech separation.