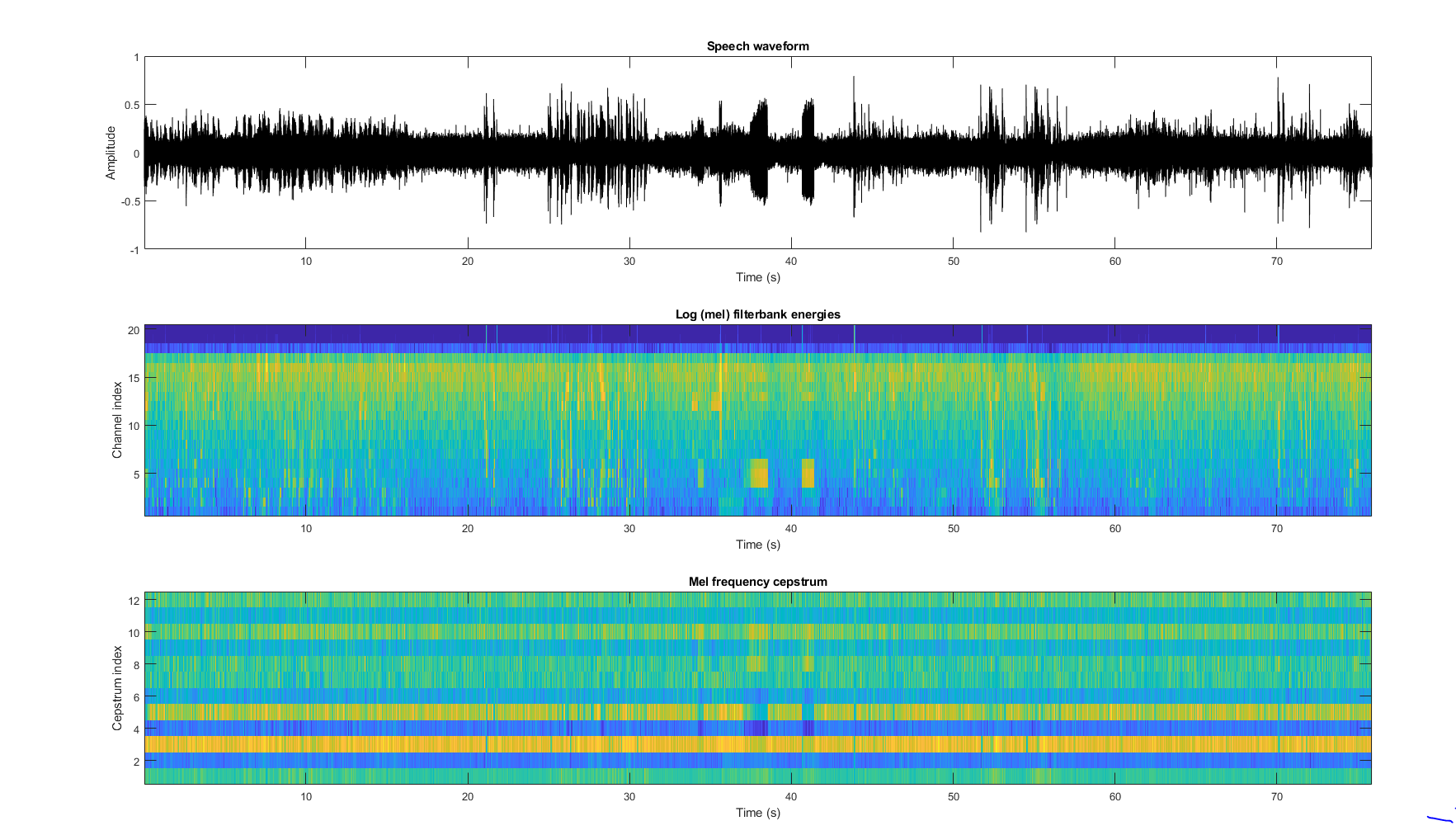
***Surgical mask detection***

Voica Mihai Cristian, 231

My machine learning project, “Surgical mask detection”, was made using the Librosa’s “librosa.core.load” function and the Mel-Frequency Cepstral Coefficients (MFCC) extracted from the audio files using another Librosa function, “librosa.feature. mfcc”. Before using the MFCCs I trained different models using the array of 22050 amplitudes, the default sampling rate of the “librosa.core.load” function (22050hz). The results were really bad, around 0.54 accuracy on the validation set. After using the MFCCs, I started seeing better results. Spectrograms are a useful technique for visualizing the spectrum of frequencies of a sound and how they vary during a very short period of time. A spectrogram uses a linear spaced frequency scale (so each frequency bin is spaced an equal number of Hertz apart), whereas an MFCC uses a quasi-logarithmic spaced frequency scale, which is more similar to how the human auditory system processes sounds.

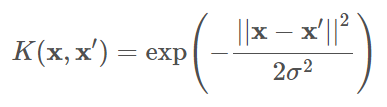


In the process of making the project, I tried using different machine learning techniques including Naive Bayes Multinomial, K-Nearest Neighbors, SVM and Keras Neural Network.

On the NB implementation I tried to digitize the MFCCs arrays in many ways but the best accuracy for the validation set was 0.57. Afterwards, I used the KNN model that we made at the laboratory with 3 neighbors and I got a score of 0.635 on validation and 0.59 on the test set. The Keras model was a Sequential that had 3 layers. First hidden layer with 12 nodes using ‘relu’ activation function, second layer with 8 nodes and ‘relu’ activation function and the output layer had one node with the sigmoid activation function. The loss function was 'binary\_crossentropy', and the optimizer ‘adam’. I tried fitting the model with different number of epochs and the best result I got was with 100 epochs, 0.81 accuracy on the validation set. Unfortunately, the model got a bad score on the test set, only 0.61.

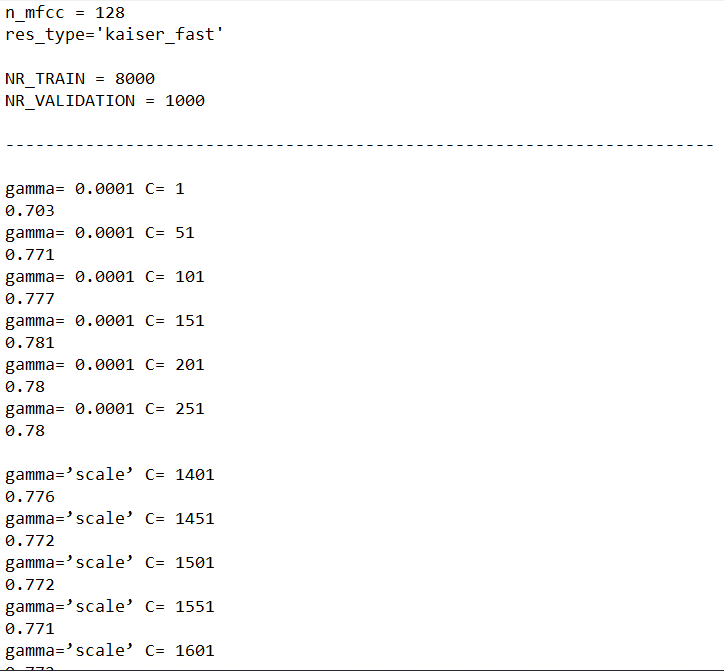
The machine learning model that I got the best score with was Support Vector Machine with Support Vector Classification from sklearn with ‘rbf’ kernel. Support Vectors Classifier tries to find the best hyperplane to separate the different classes by maximizing the distance between sample points and the hyperplane.

Radial Basis Function is a commonly used kernel in SVC:



I trained the model using different gamma and C parameters after I saw how much they influence the accuracy on the validation set. ‘gamma’ is a parameter of the RBF kernel and can be thought of as the ‘spread’ of the kernel and therefore the decision region. When gamma is low, the ‘curve’ of the decision boundary is very low and thus the decision region is very broad. When gamma is high, the ‘curve’ of the decision boundary is high, which creates islands of decision-boundaries around data points. ‘C’ is a parameter of the SVC learner and is the penalty for misclassifying a data point. When C is small, the classifier is okay with misclassified data points (high bias, low variance). When C is large, the classifier is heavily penalized for misclassified data and therefore bends over backwards to avoid any misclassified data points (low bias, high variance).

I tried a lot of values (gamma from 0.00001 to 100 and C from 0.01 to 100000) using a for loop and writing the scores in a text file to guess the best values for the test set.

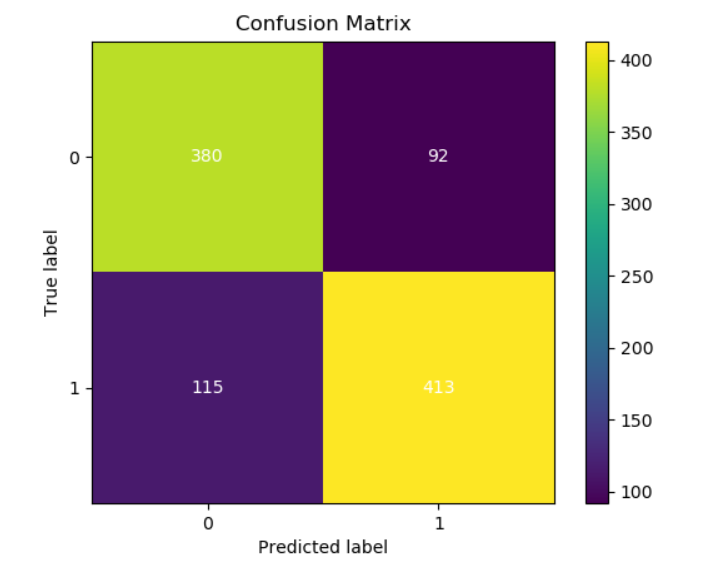


…

After a lot of submissions, the best score I got on the test set is 0.65333 with gamma=‘scale’ and C=85000 after it had a good score on the validation set of 0. 0.793. I thought that a very big C will overfit the data, but apparently that was a good thing for this data set.

Below is some information about the run on the validation set.





Bibliography:

* <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>
* <https://medium.com/@mikesmales/sound-classification-using-deep-learning-8bc2aa1990b7>
* <https://chrisalbon.com/machine_learning/support_vector_machines/svc_parameters_using_rbf_kernel/>