ENGR-E 511 Fall 2018: Assignment #4

Due on Sunday, November 11th, 11:59P

Professor Minje Kim

Abhilash Kuhikar (akuhikar@iu.edu)

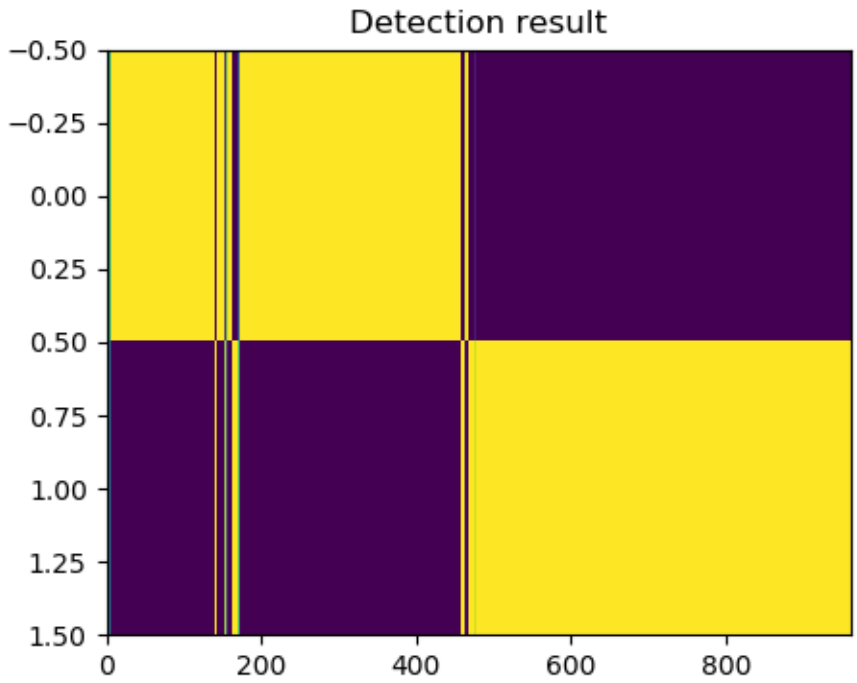
November 12, 2018

Contents

Problem 1: When to applaud?

The objective is to use naive smoothing and viterbi to come out with the smooth posterior probabilities of when to clap.

(a) **The initial posterior probabilities of piano and clap sound waves**:
Let's look at the plot of initial posterior probabilities.

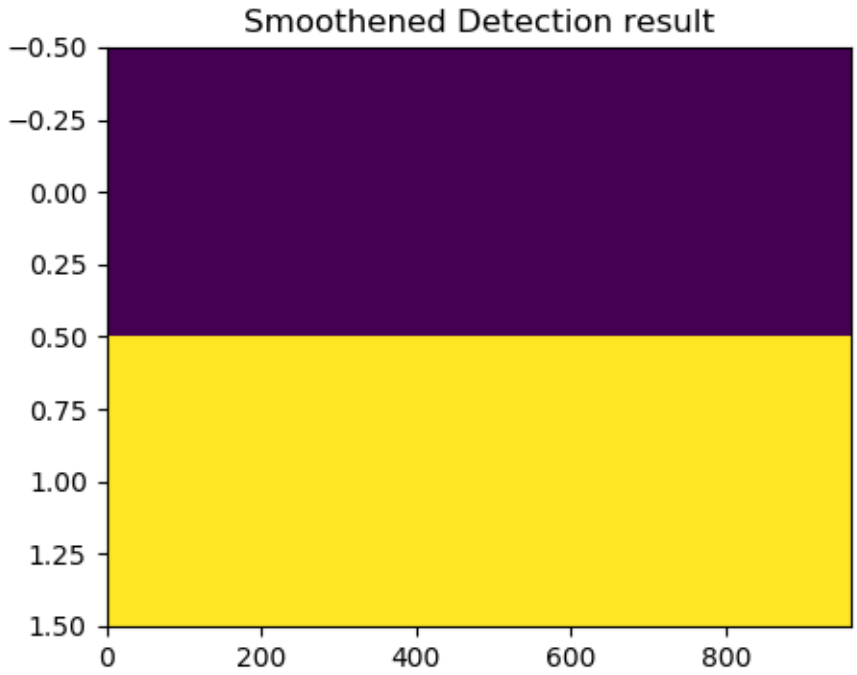


We can see that there are some probabilities corresponding to the clap at around 160th frame

(b) **Naive smoothening using some transition probabilities**

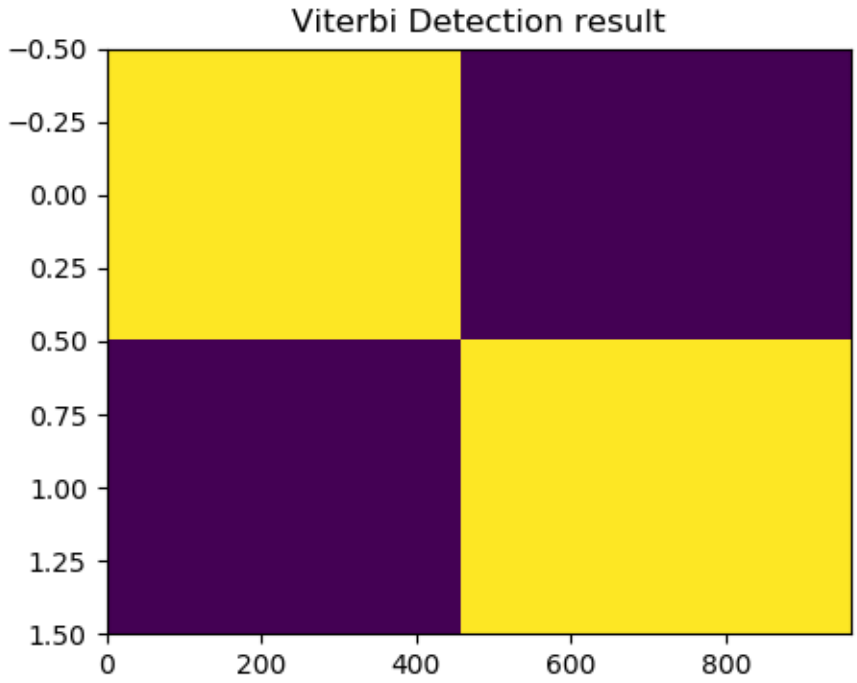
Here we use the naive smoothening to get the posterior probabilities. Here we make a hard decision at every frame looking at just the previous frame class.

Let's look at the smoothened detection result

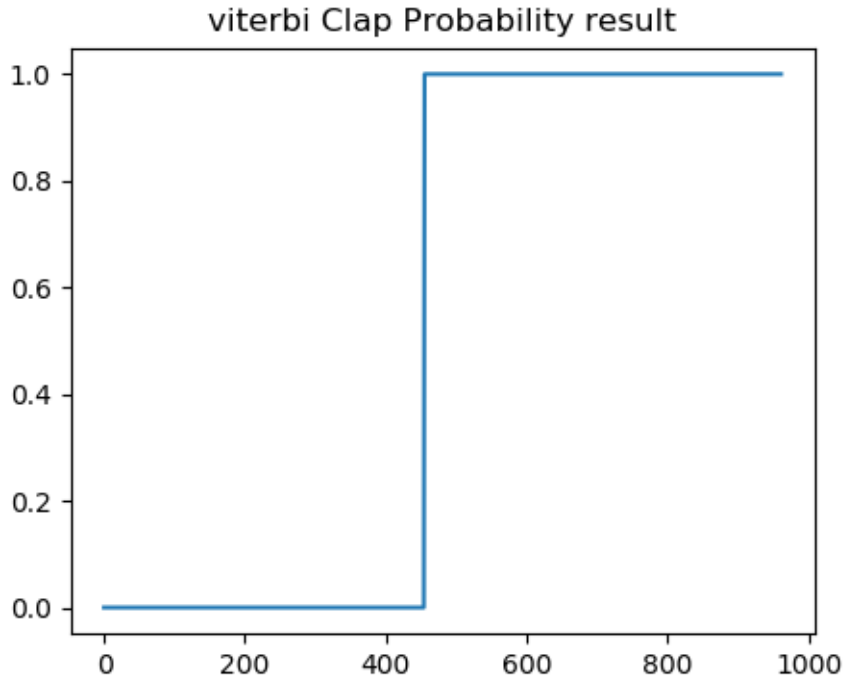


We can see that as the first frame probability was more for the clap, all the subsequent probabilities are assigned to the clap class. We will use viterbi to smoothen out this noisy claps frames.

(c) **Detection using the viterbi smoothening**
We use the viterbi algorithm to smooth out the noisy clap frames.
Let's look at the transition of the piano and clap probabilities.



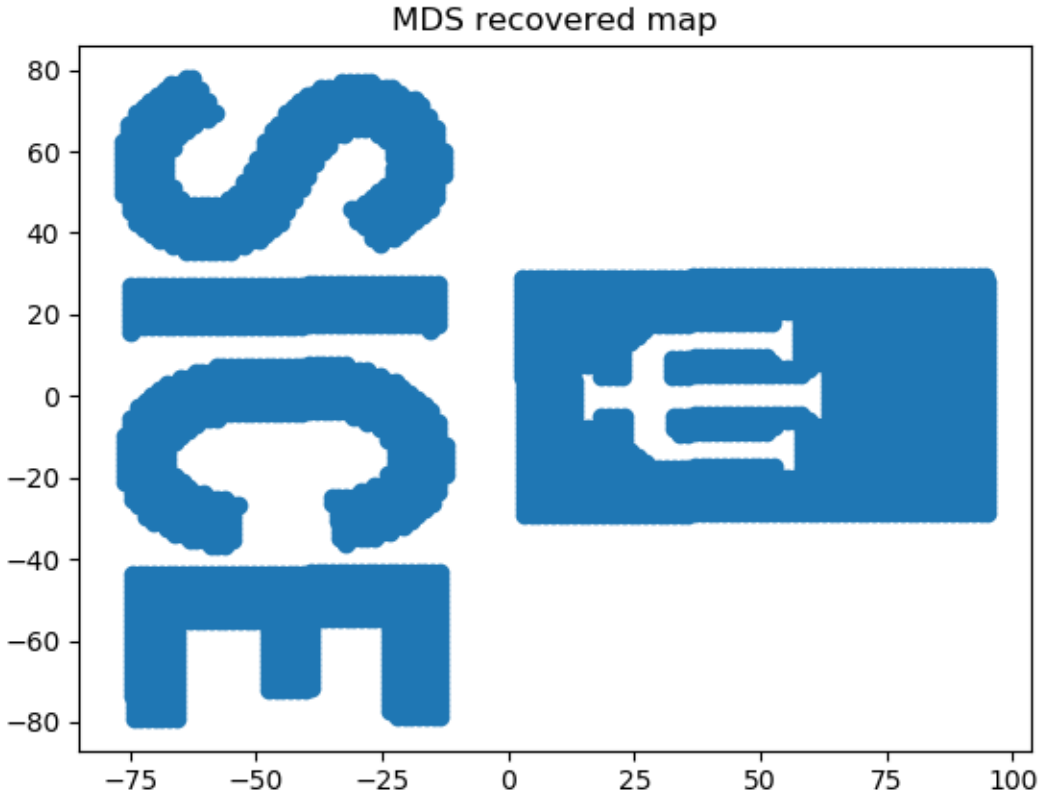
Let's look at the posterior clap probability found using the viterbi algorithm



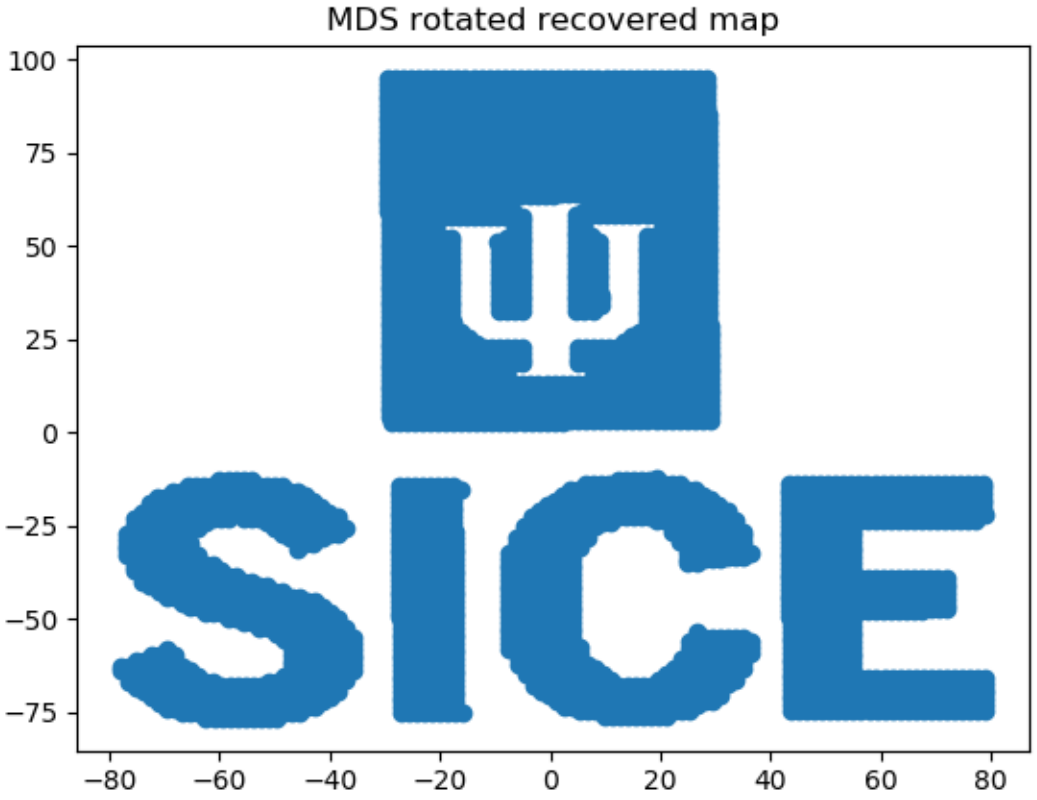
We can see that the hidden state sequence found by the viterbi makes much more sense than the naive smoothening. We can see that at 455th frame, the probability of clap changes from 0 to 1 and that is when the clapping should be started.

Problem 2: Multidimensional Scaling

The objective is to use MDS to recover the original map out of the pairwise euclidean distance matrix
Let's look at the recovered plot of the map

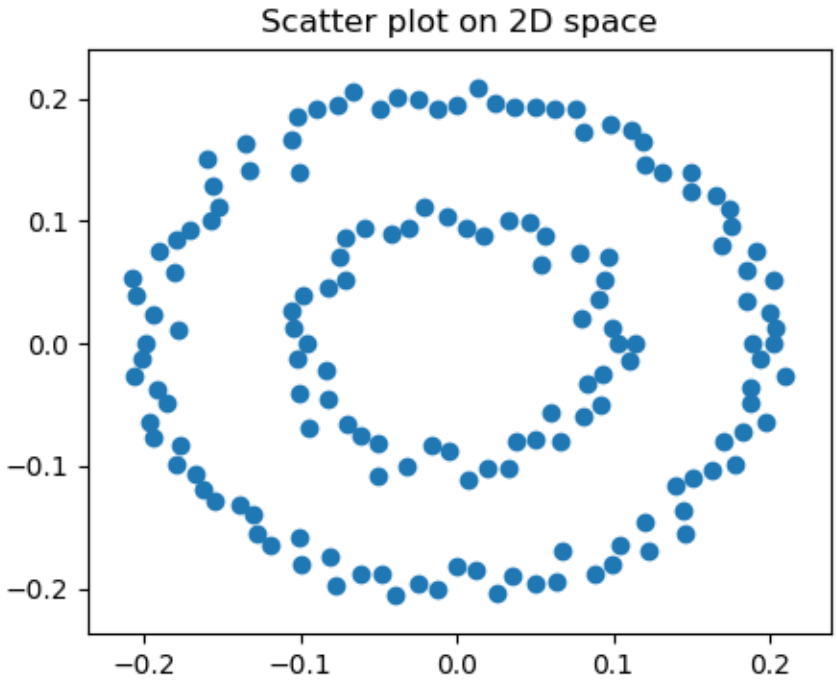


We can rotate this to get the below result

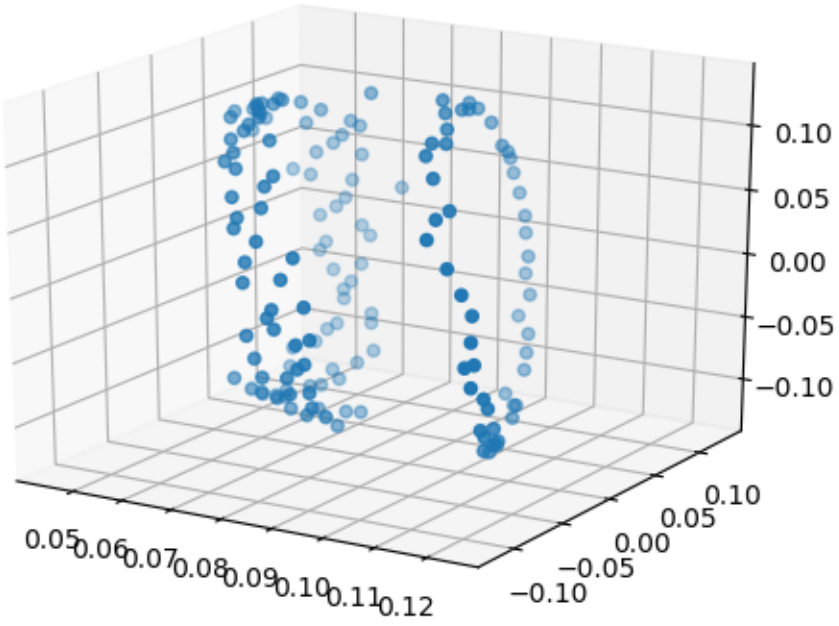


Problem 3: Kernel PCA

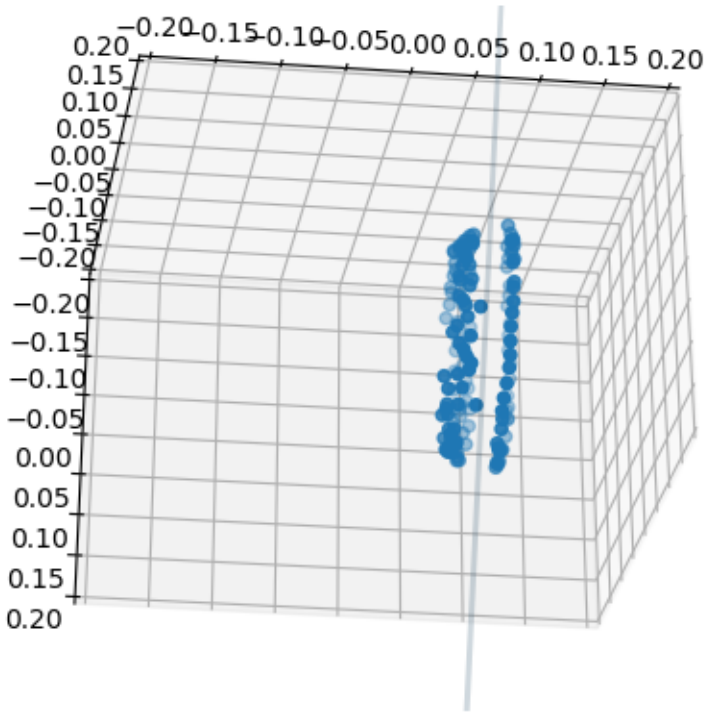
The objective is to use Kernel PCA to transform the lower dimensional data into higher dimensional and then project the data onto this higher dimensional space. This higher dimensional space will allow us to linearly separate the data samples which otherwise would not have been possible in the lower dimensional space.
Let's look at the original data points in their original feature space



We can see that the data samples are not linearly separable.
Now we will use RBF kernel to transform this data into infinite feature space and then do the kernel PCA on this kernel matrix to get the largest three eigen vectors.

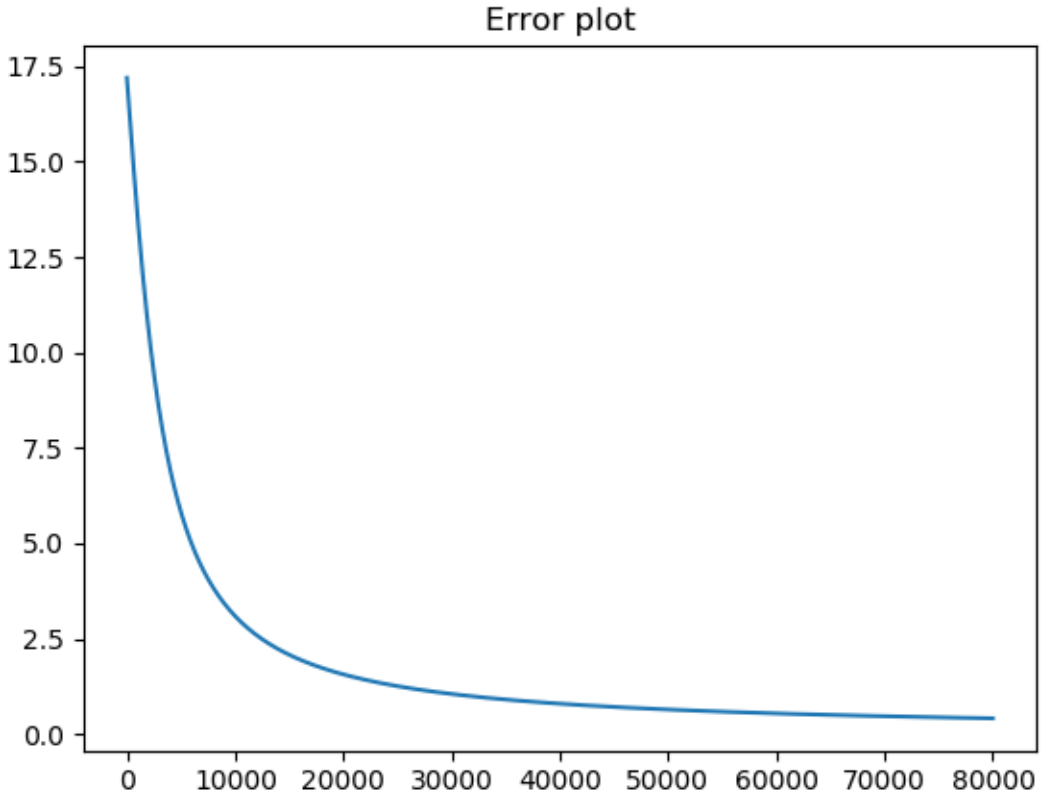

0.10
0.05
0.00
−0.05
−0.10
0.05
0.06
0.07
0.08
0.09
0.10
0.11
0.12
0.10
0.05
0.00
−0.05
−0.10

We can now see that the data is now linearly separable and train the simple perceptron to learn the parameters. After learning we get the following result.



We can see that the plane linearly separates the data samples into two. The plane is the learned parameters by the perceptron. I have tried some different values for σ^2 and found out that at $\sigma^2 = 0.05$ the data samples were linearly separable in 3D space.
The learning rate is 0.05 The learned parameters are as follows:
W = [-156.33821067, -20.72214572, -6.62144019]
b = 13.90218439

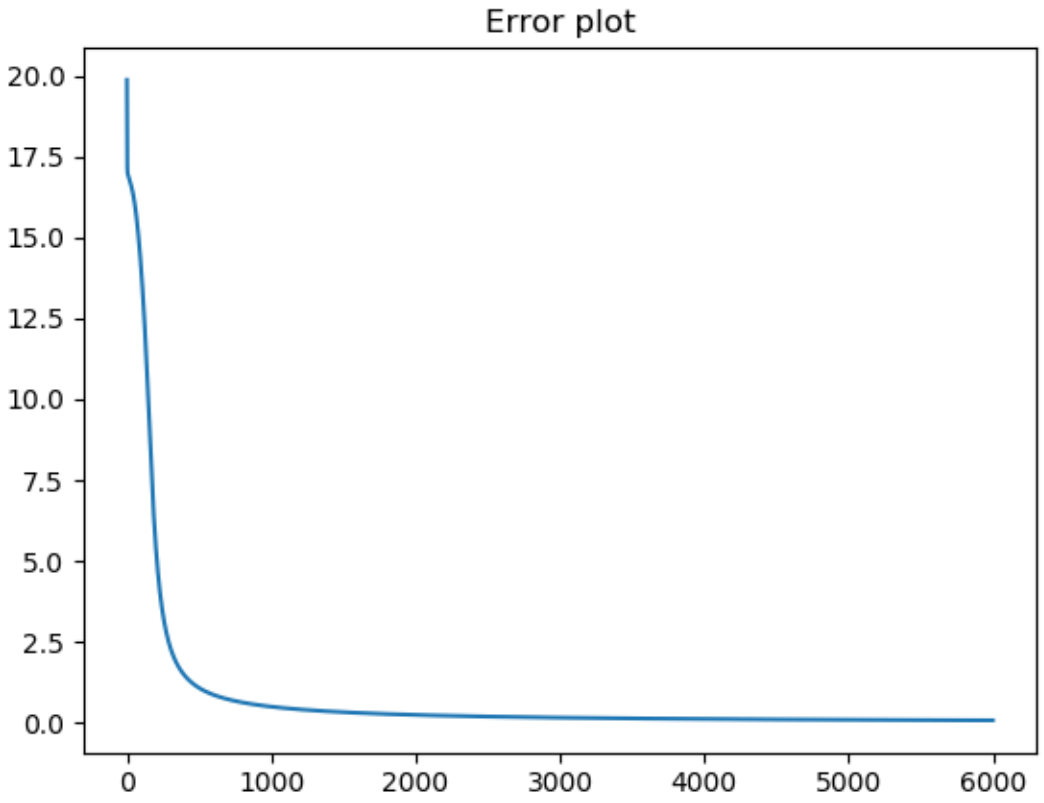
Let's look at the error plot for this:



We can see that the error is converged to 0.414 which means the accuracy is about 100

Problem 4: Neural Networks

The objective is to use the neural network with a single hidden layer to transform the 2d data into 3d space(because 3 hidden units in the hidden layer).
The backpropagation is used to train the parameters of all the layers together. I have set the learning rate $\alpha = 0.05$
The error plot is as follows.



The learned parameters for the different layers are as follows:
First layer:
weights, W1 = [9.35623744, 0.62051901
3.04431474, 5.23175853
5.67624867, -4.17400045]
biases, B1 = [3.80826768, -3.82122869, -3.53942453]
Second Layer:
weights, W2 = [-15.03554034, 15.05490477, 15.39899703]
bias, b2 = 4.11773946

The accuracy is 100
The error converged to 0.08

References