**ALGORITHMS USED TO CALSSIFY THE ALPHABETS USING PERSISTENT HOMOLOGY**

STEPS:

CALCULATING BETTI NUMBERS FOR THE GIVEN DATASET

1.     Developed a function which calculates betti numbers for 0 dimensional and 1 dimensional simplicial complex using ripser filtration.

2.     The manually labelled data is used as a point cloud and fed as input to rips filtration and their by generating 0-dim and 1-dim betti numbers.

3.     The betti numbers function takes data set (0s and 1s) as input and outputs a tuple which contains betti 0s and betti 1s based on multiple scanning types.

4.     Following are the scanning types used to calculate betti numbers:

      Left to Right or Right to Left

      Uptown or Down up, for downup scanning fix the scanstop at the last index of the list of 0’s and 1’s and by altering scanstart.

      Scanstart for updown and left-right scanning is fixed as the 1st index of the list.

      Middle out scanning by gradually incrementing scanstart and scanstop at the same time.

5.     The maximum distance is ribs filtration is set to 100.

6.     Plotted the diagrams by using the above-mentioned scanning techniques.

CREATING CLASSIFICATION VECTORS FOR THE DATASETS

1.     Developed a function to classify the data using above mentioned scanning types.

2.     This function takes dataset as input and a flag which controls single vector analysis or whole dataset to analyze.

3.     This function returns a matrix containing vectors of weight responding to the signatures from the datasets.

4.     The classification vector function iterated through all the scanning types for each character in dataset and outputs weight vectors.

      After calculating betti numbers and classification vectors for the dataset, developed a comparison function which takes a list of weight vectors for the known letters and returns a string containing the classified the unknown letter as output.

      Finally, added noise to the data by changing 0’s and 1’s and also generated noise data randomly.

      The noise test function takes a string of 0’s and 1’s as noise and returns classification rate which refers to the proportion of correctly classified letters to a total number of letters as output.

      Higher the classification rate (close to 1) better is the prediction for the unknown letters.

Finally, performed 100 tests of classifying the dataset with noise added and the average classification rate is calculated. The classification seems to be reasonably good with 60% accuracy.

The accuracy of the classification rate can be improved by making use of optimal scanning techniques and using advanced filtration method