

FAQ's DMG2 Assignment

Q7.

P7 : Pair-wise Classifier Features

- For all pairs of classes in MNIST data
- Compute the Fisher Discriminant for that pair of classes
- This is again a D dimensional vector where each dimension corresponds to a pixel.
- Convert the D-dimensional vector into an image – scale it and draw the image
- Comment on how Fisher discriminant for a few pairs of classes is learning to “focus” on different pixels in the image.

You start this question with 45 Pairs of Dataset, and each task you need to do 45 Times.

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255, inclusive

Here every Pixel corresponds to one dimension.

For pair 1.

Step1: First you remove all the pixels (Dimensions), where the sum of variance of both the classes is equal to 0. Because as per the formula for Fisher discriminant, if the sum of variance of both classes are equal then it will bring error to system.

Step2: Then the Data will come down by fewer Dimension from 784. Then you run a fisher Discriminant analysis on the given data. After running the model then you bring back those pixels that were taken in first step.

Step3: When you run the model you will have the D -Dimensional vector, which you need to project as a single image. Once you project that image you will find the areas where it is blurred and where it is dark. You need to comment on the part where the dimensions are least separable.

Q8.

P8 : Binary Hierarchical Classifier

- Build a Bottom-up-Binary Hierarchical Classifier
- Use Linear SVM to build each of the pair-wise classifiers
- Merge two classes that give the highest training error
- Draw the entire tree that was discovered automatically
- For each node in the tree – write the training and test accuracies.

So in this question you start with 45 pairs of dataset.

Step1: You build an SVM model, on each pair. One with least accuracy which means highest training error will get merged.

Suppose 1 and 7 has the maximum training error amongst all the pairs, so it will get merged, then all the pairs containing these two classes like (1vs2, 1vs3,... ,7vs9) will disappear, and a new class will be formed that will be 1_7 the merged class.

Then all the pairs barring the class 1 and 7 will remain including new pairs with the merged class.

1_7 vs 2, 1_7 vs 3, 1_7 vs 6, 1_7vs8, 1_7 vs9.

Now again amongst the remaining pair whosoever has the maximum training error, will get merged and the process will continue until we get the final merged class.

You need to build a tree of this process.