Exploratory Analysis of MNIST Dataset

Description of training and testing image files:

The images data is stored in a very simple file format. The data files train-images-idx3-ubyte and t10k-images-idx3-ubyte are taken from the mnist website. These files can be read using any file reader. The actual information about the pixels starts from the 16th position. The first 4 characters provide information about endianness and the next 4 characters are about the count of images followed by row and column count(of each image) in the next 8 characters respectively. Also, pixels are organized row-wise with values ranging from 0 to 255. The training set contains 60000 examples, and the test set 10000 examples. Out of all the images, the first 30k images are taken from NIST's Special Database 3(SD-3, which was collected among Census Bureau employees) and the remaining 30k are from NIST's Special Database 1(SD-1, which was collected among high-school students). Furthermore, the test set has 5,000 images taken from SD-3 and 5,000 images from SD-1.

Description of training and testing label files:

The labels data is also stored in a very simple file format. On the original website, two files - train-labels-idx1-ubyte and t10k-labels-idx1-ubyte are provided. These files can be read using any file reader however, the actual information about the labels starts from the 8th position. The first 4 characters give information about endianness and the next 4 characters give information about the number of items.

Count of images available for each label(in train-data):

To make sure that we have almost the same number of examples for each image(so that the model will not be biased), we read the train labels file. Upon reading this file we found the following information - {'1': 6742, '2': 5958, '3': 6131, '4': 5842, '5': 5421, '6': 5918, '7': 6265, '8': 5851, '9': 5949, '0': 5923}. From these counts, we can be sure that the model will not get overtrained on a particular digit.

Outlier Detection:

While doing exploratory analysis we were very curious to see what if some images are very different from others. For example, few people write 7 with bar in middle so, how our model will classify these outliers.

Finding outliers for numerical data is very straight forward but for images we were looking for a though so that we could find these outliers, if exist. On doing some research, we got to point where thought we will never get to a conclusion if we think in the direction of images. So, we moved to the basic unit of an image. Which is pixel.

As we are using MNIST dataset, on above analysis we found out that in our image we have total 784 pixels in 1D array or 28x28 shape in 2D array. And each pixel value is between 0-255. Having this idea, we then thought of generating an image that will have average pixel values of all pixels from images of that class known as centroid images.

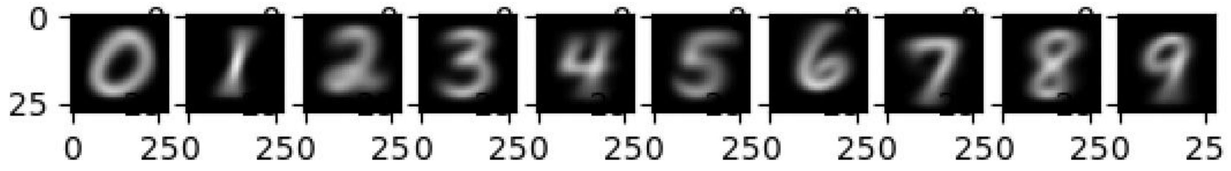


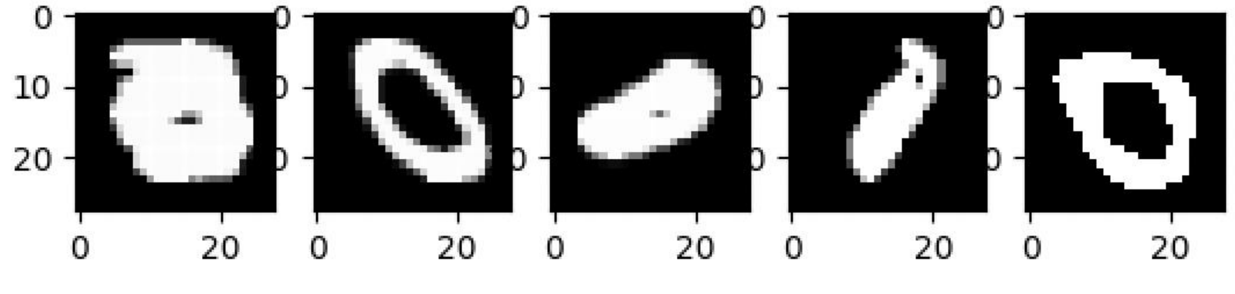
Figure 1 Generated Centroid Images after averaging pixels of all images of that class.

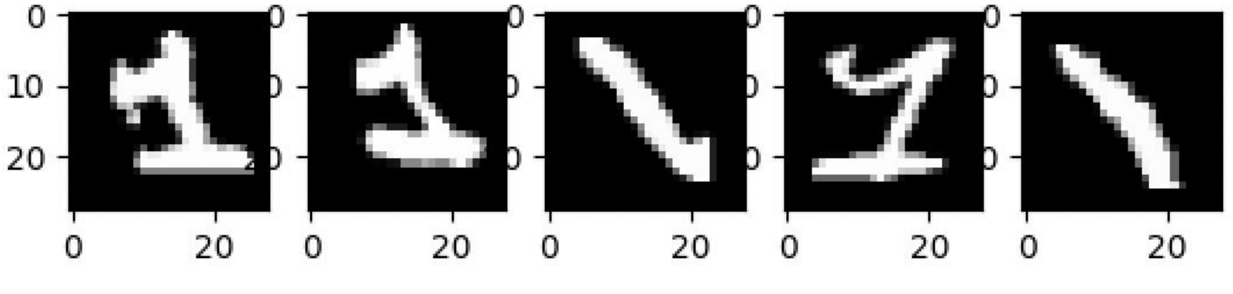
As from Figure 1 we can see that these centroid images look almost like numbers that we usually write. As we created these images with average pixel values these came out to be blurred, as expected.

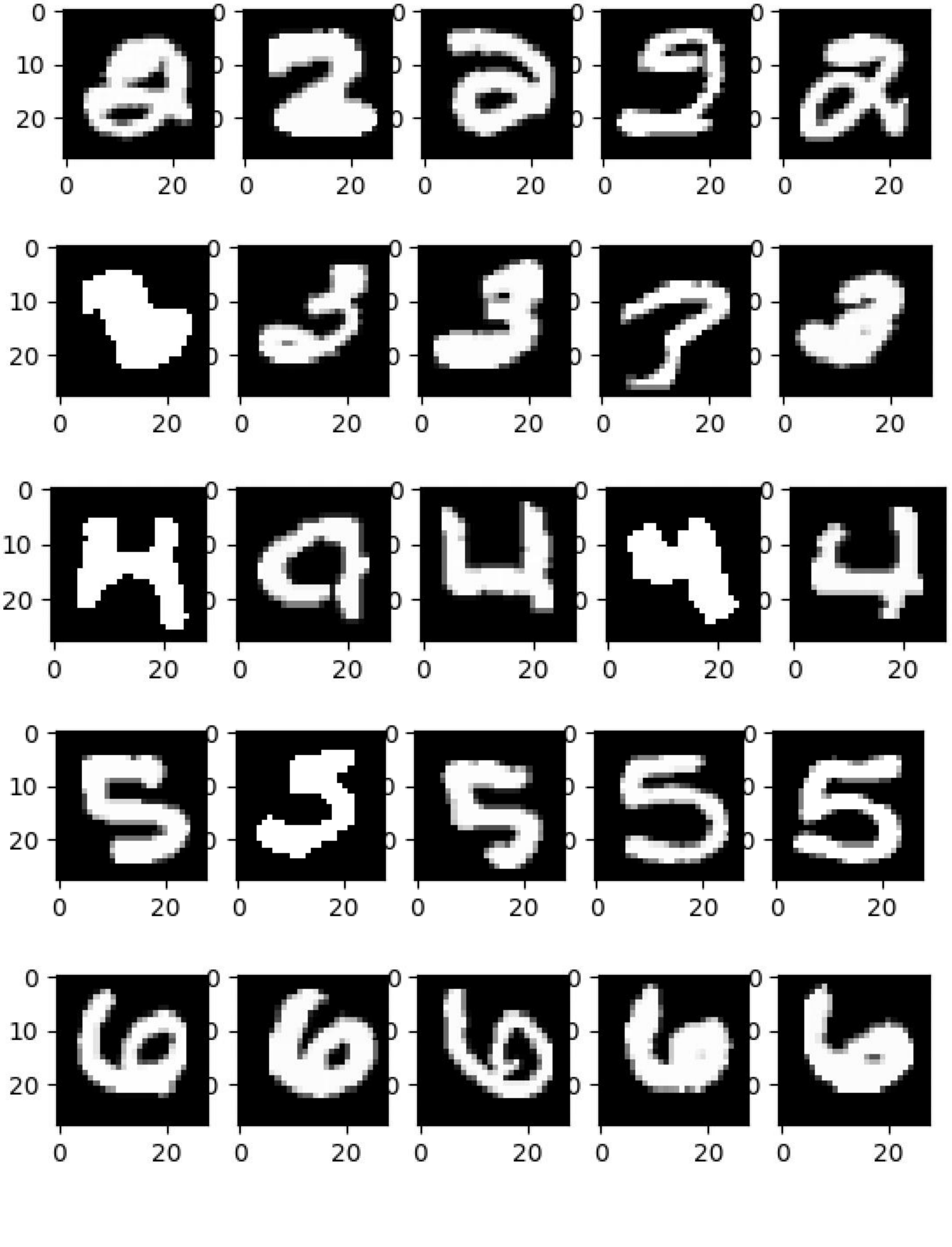
After finding these centroid images, we found Euclidean distance of all images of a class to the centroid image that we generated for that class. And we had an intuition that those images that will have more Euclidean distance will look totally different to our centroid images. And we will get our outliers.

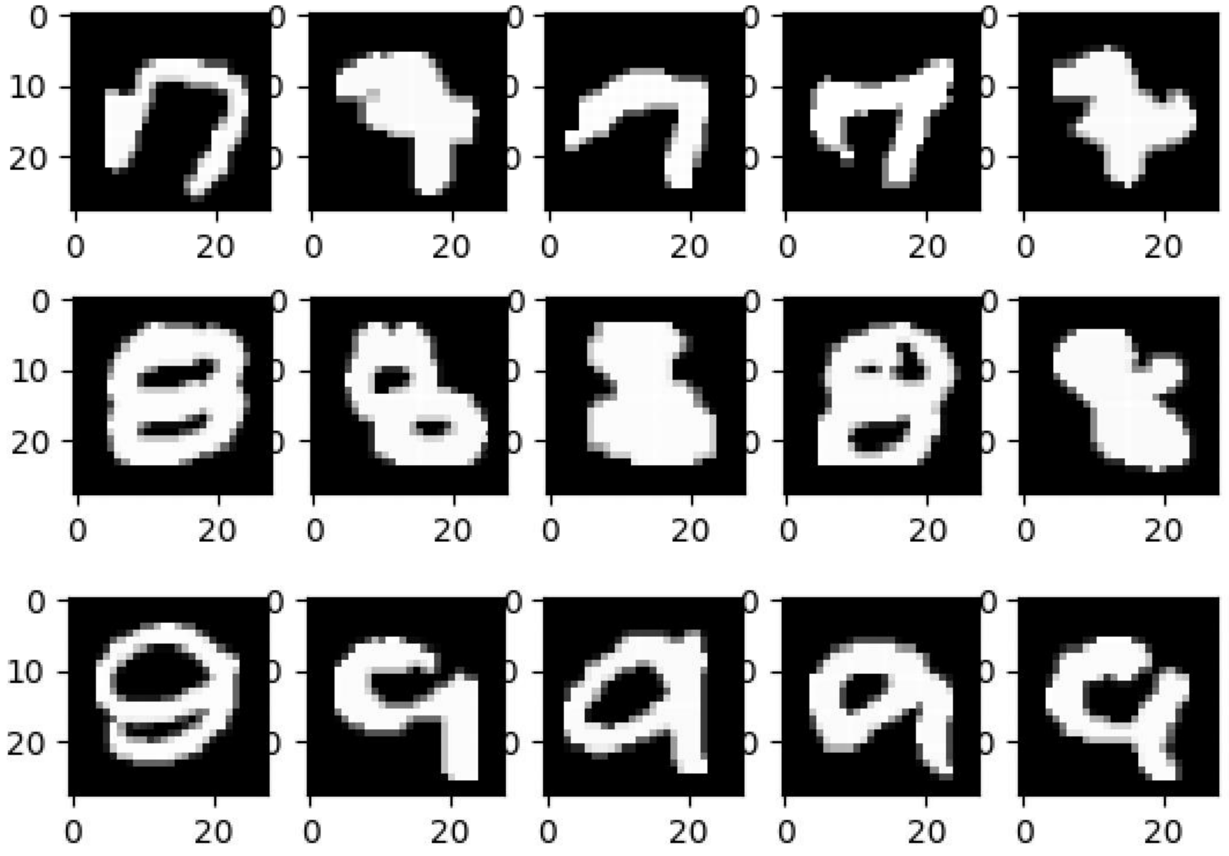
From our findings, we got the same results that we expected. The images that were more distant from centroid images looks much different from others. Below are the top 5 images of each class that had greatest Euclidean distance from the centroid image of that class.

From these findings, we can find out that on training our model if these images are classified correctly which will show that our model is not generalizing well and model is overfitting the training data.



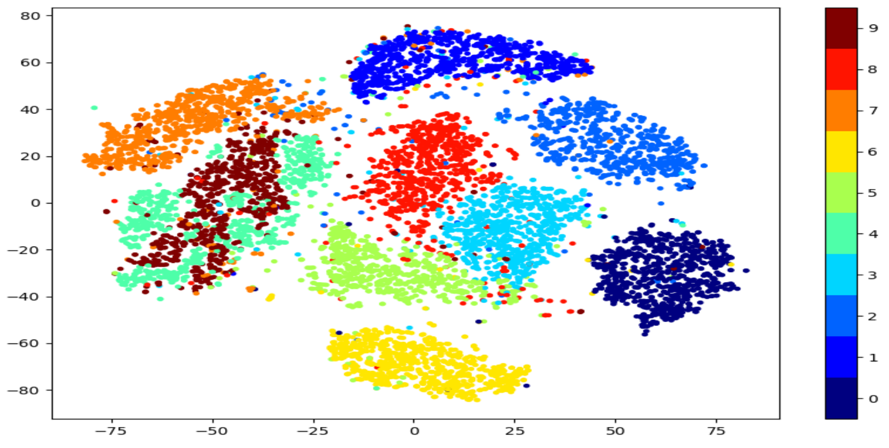






Visualizing the data in higher dimension:

Plotting t-sne (t-distributed stochastic neighbor embedding) for MNIST would help us visualize the data in a higher dimension by giving each image a location in a 2D space. We see that all the similar-looking digits are falling under the same cluster and the majority of the clusters seem to be having well-defined cluster boundaries. Indicating that the data can be classified easily without needing many transformations. Thus, simple neural networks would be sufficient to classify most images accurately.


80
60
40
20
0
−20
−40
−60
−80
−75
−50
−25
0
25
50
75
9
8
7
6
5
4
3
2
1
0