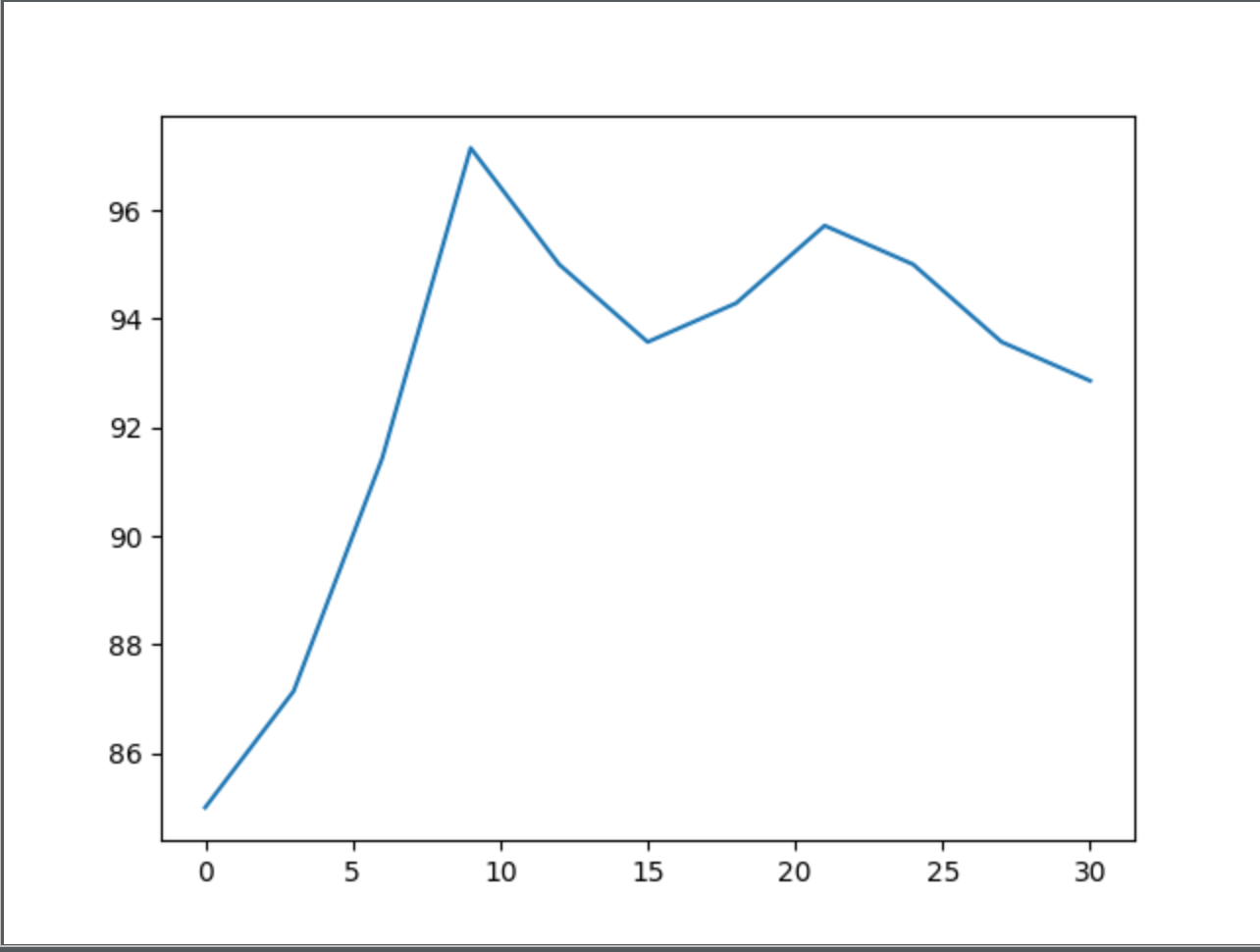
HOMEWORK-4

Q3) Testing Algorithms on Data

1. Neural Network

**Using Sigmoid activation function:**



Accuracy vs No. of hidden nodes

Here I = No. of hidden nodes

FOR I= 3 Test accuracy: 85.00%

FOR I= 6 Test accuracy: 87.14%

FOR I= 9 Test accuracy: 91.43%

FOR I= 12 Test accuracy: 97.14%

FOR I= 15 Test accuracy: 95.00%

FOR I= 18 Test accuracy: 93.57%

FOR I= 21 Test accuracy: 94.29%

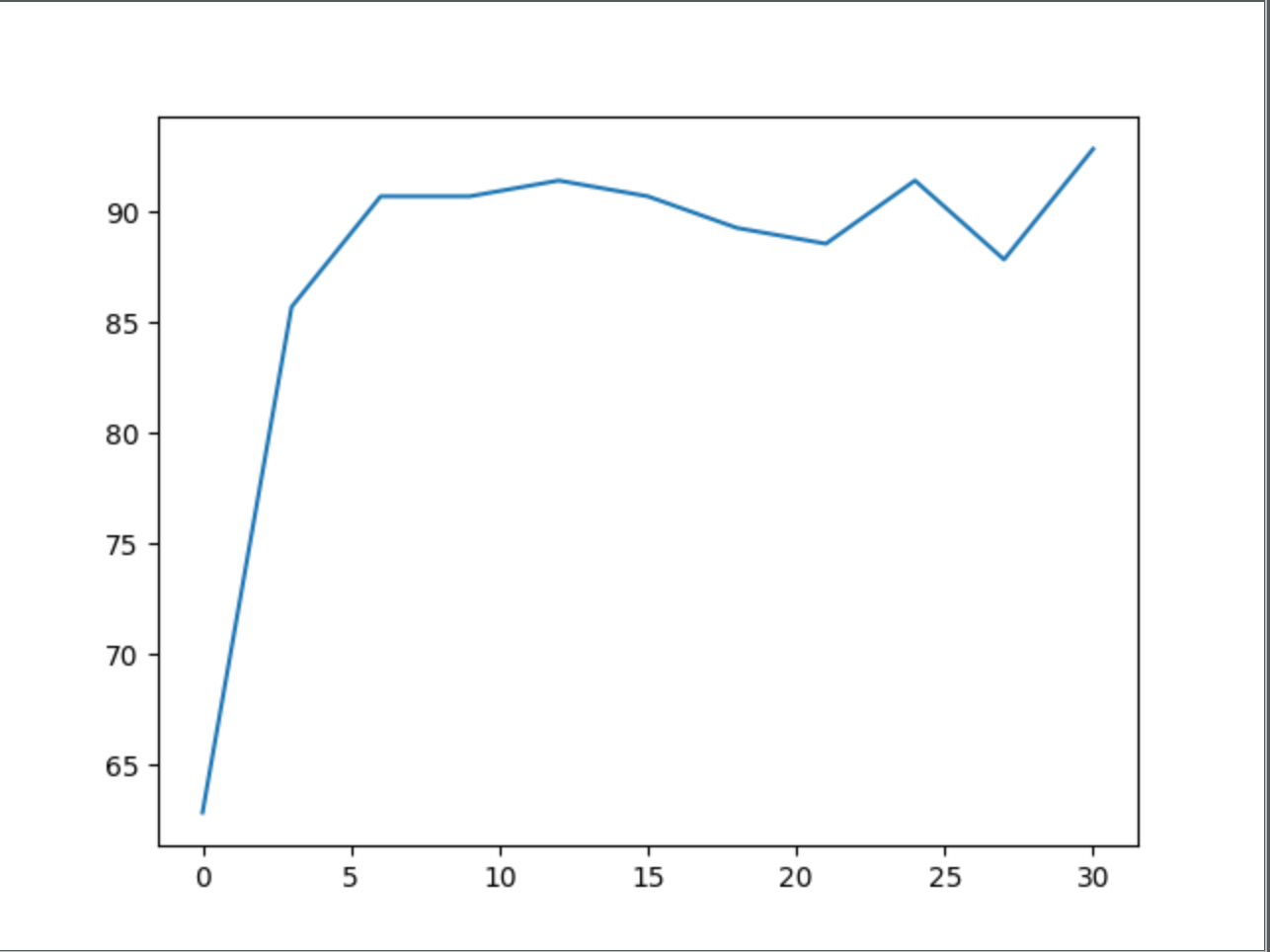
FOR I= 24 Test accuracy: 95.71%

FOR I= 27 Test accuracy: 95.00%

FOR I= 30 Test accuracy: 93.57%

FOR I= 33 Test accuracy: 92.86%

**Using RELU as the activation function:**

****

Accuracy vs. No. of hidden nodes

Here I= No. of hidden nodes

FOR I= 3 Test accuracy: 62.86%

FOR I= 6 Test accuracy: 85.71%

FOR I= 9 Test accuracy: 90.71%

FOR I= 12 Test accuracy: 90.71%

FOR I= 15 Test accuracy: 91.43%

FOR I= 18 Test accuracy: 90.71%

FOR I= 21 Test accuracy: 89.29%

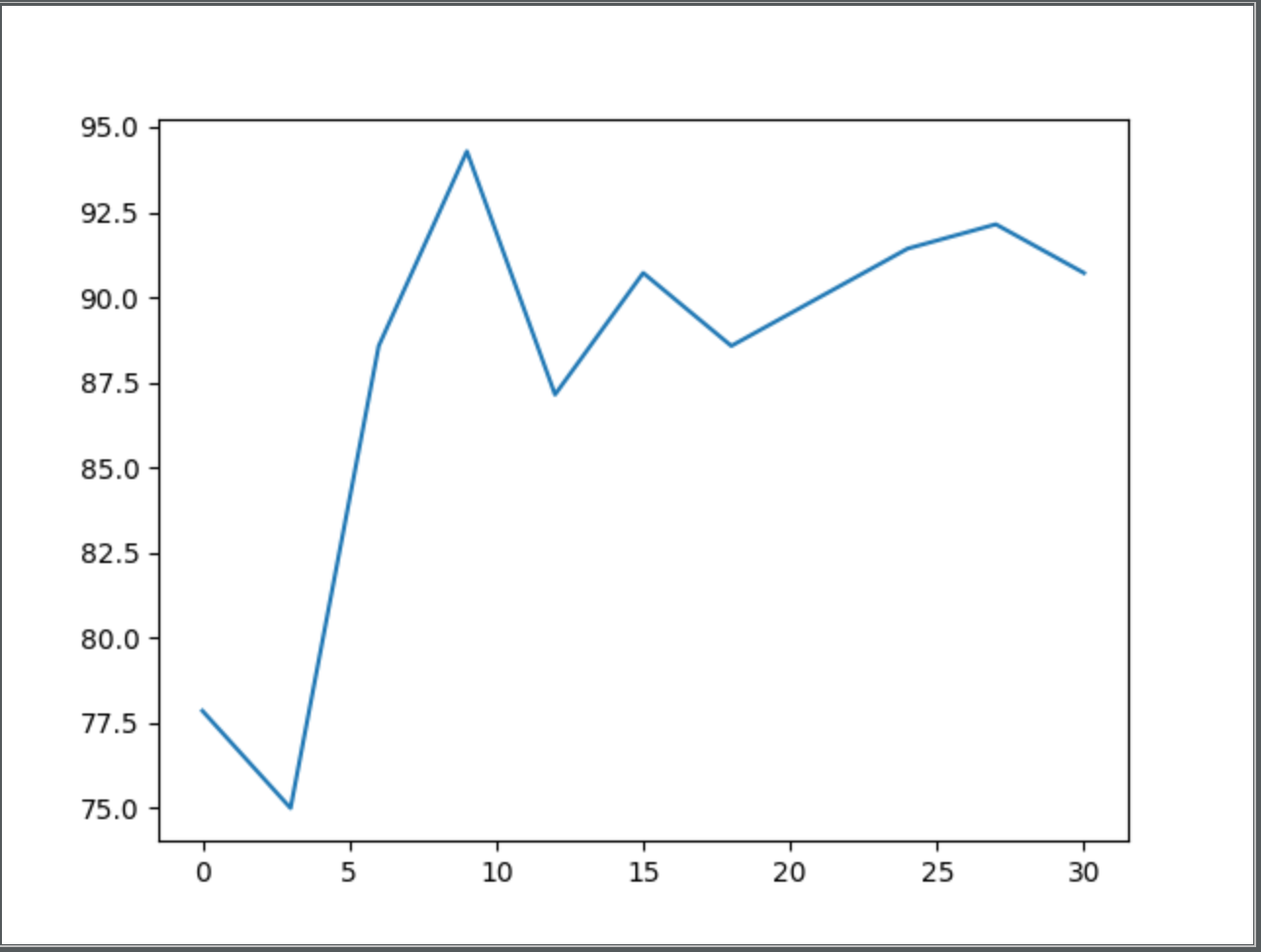
FOR I= 24 Test accuracy: 88.57%

FOR I= 27 Test accuracy: 91.43%

FOR I= 30 Test accuracy: 87.86%

FOR I= 33 Test accuracy: 92.86%

**Using TanH as the activation function:**

****

Accuracy vs. No. of hidden nodes

FOR I= 3 Test accuracy: 77.86%

FOR I= 6 Test accuracy: 75.00%

FOR I= 9 Test accuracy: 88.57%

FOR I= 12 Test accuracy: 94.29%

FOR I= 15 Test accuracy: 87.14%

FOR I= 18 Test accuracy: 90.71%

FOR I= 21 Test accuracy: 88.57%

FOR I= 24 Test accuracy: 90.00%

FOR I= 27 Test accuracy: 91.43%

FOR I= 30 Test accuracy: 92.14%

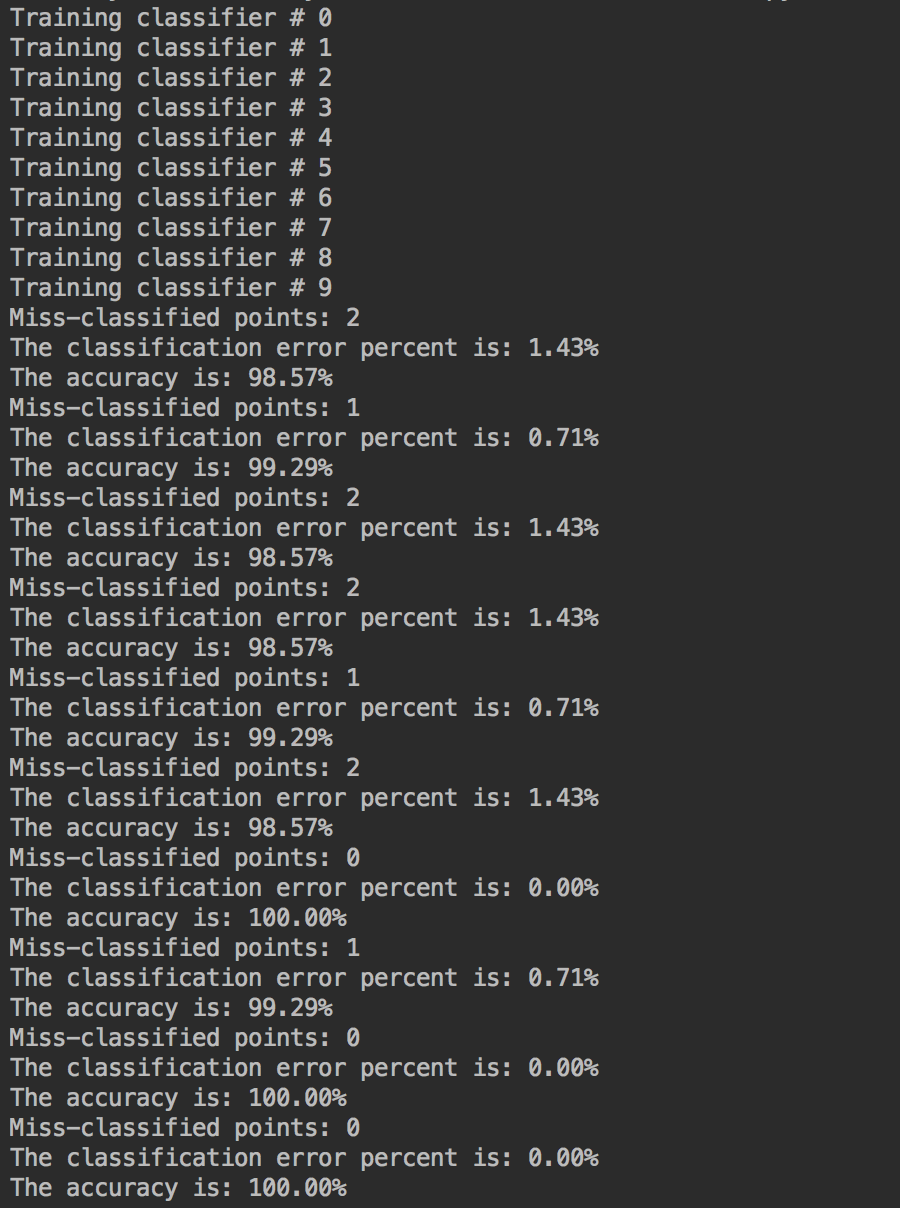
FOR I= 33 Test accuracy: 90.71%

Above mentioned are the results and graphs that my implementation of the FNN algorithm gives me for different activation functions. As you can see there is a trend here. As the no. of hidden nodes increase there is an increase in accuracy only till a certain point, after which the accuracy seems to decrease on the test set. What I can infer from the above results is that as the number of the hidden nodes increase, so does the complexity of the model hence it is mapping each function better and able to predict my test data in a more accurate way. But as you increase the complexity of the model, the chance that you are overfitting the training data also increase. So that is why you can see the sudden dip in accuracy for the test data as you are overfitting the training sample.

As far as the activation functions, using Sigmoid is performing better than Tanh and ReLU.

This is cause the loss function I am using is more suited towards sigmoid activation function.

1. 1 vs all SVM



Final classification error on test data: 11 misclassifications

1. 1 vs all Logistic regression

Final classification error on test data: 23 misclassifications

1. PCA followed by classification

**Neural Nets**

**Using Sigmoid as the activation function:**

FOR I= 3 Test accuracy: 9.29%

FOR I= 6 Test accuracy: 7.86%

FOR I= 9 Test accuracy: 13.57%

FOR I= 12 Test accuracy: 11.43%

FOR I= 15 Test accuracy: 12.86%

FOR I= 18 Test accuracy: 15.00%

FOR I= 21 Test accuracy: 12.86%

FOR I= 24 Test accuracy: 20.00%

FOR I= 27 Test accuracy: 19.29%

FOR I= 30 Test accuracy: 14.29%

FOR I= 33 Test accuracy: 7.86%

**Using Relu as the activation function:**

FOR I= 3 Test accuracy: 6.43%

FOR I= 6 Test accuracy: 9.29%

FOR I= 9 Test accuracy: 14.29%

FOR I= 12 Test accuracy: 9.29%

FOR I= 15 Test accuracy: 12.86%

FOR I= 18 Test accuracy: 5.00%

FOR I= 21 Test accuracy: 10.00%

FOR I= 24 Test accuracy: 10.71%

FOR I= 27 Test accuracy: 14.29%

FOR I= 30 Test accuracy: 12.86%

FOR I= 33 Test accuracy: 9.29%

**Using TanH as the activation function:**

FOR I= 3 Test accuracy: 5.71%

FOR I= 6 Test accuracy: 6.43%

FOR I= 9 Test accuracy: 14.29%

FOR I= 12 Test accuracy: 11.43%

FOR I= 15 Test accuracy: 14.29%

FOR I= 18 Test accuracy: 17.86%

FOR I= 21 Test accuracy: 10.00%

FOR I= 24 Test accuracy: 12.86%

FOR I= 27 Test accuracy: 16.43%

FOR I= 30 Test accuracy: 17.14%

FOR I= 33 Test accuracy: 12.14%

**1 vs all SVM classifier**

Final classification error on test data: 223 misclassifications

**1 vs all Logistic regression**

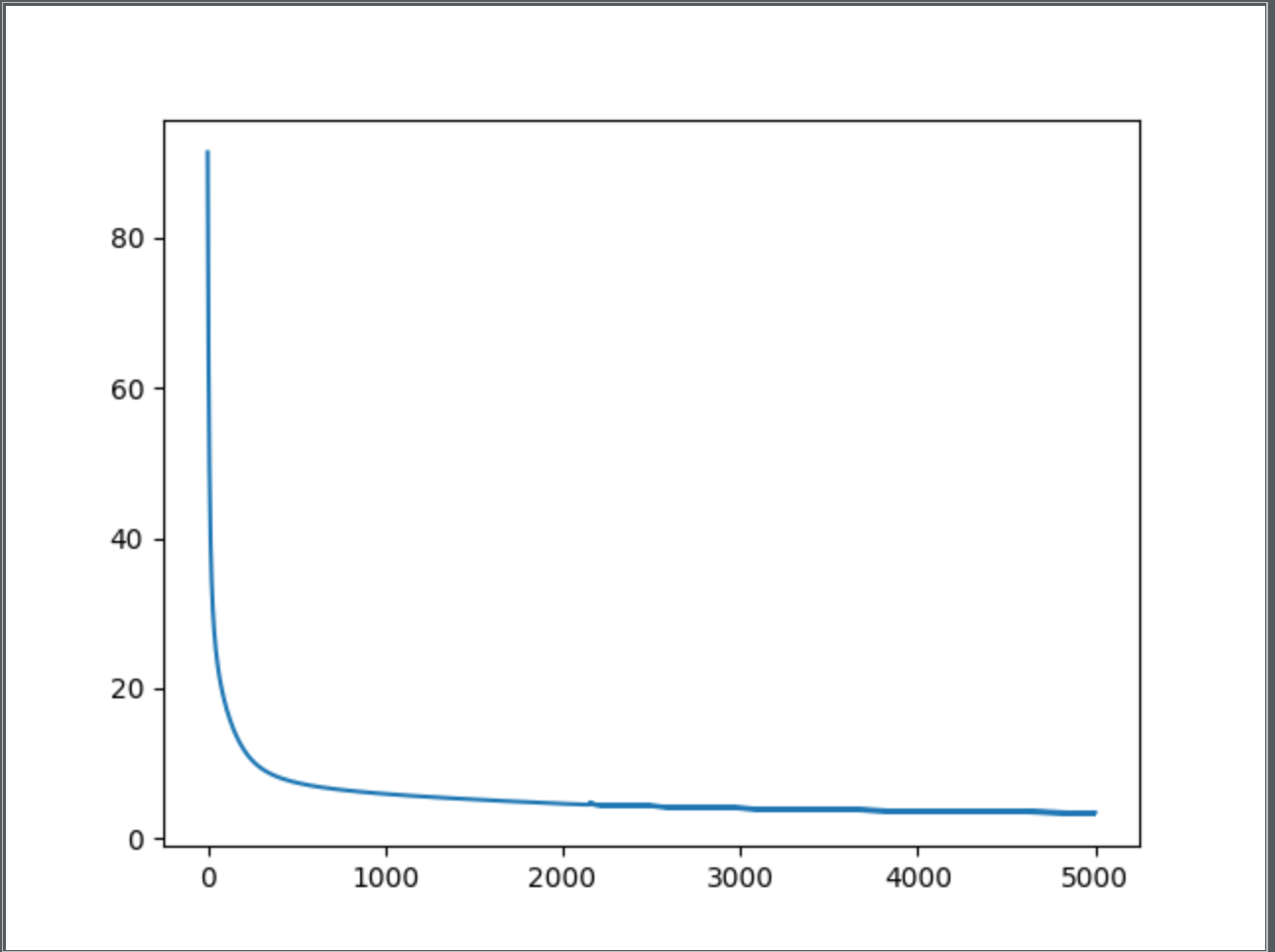
Final classification error on test data: 265 misclassifications

After applying PCA and then classifying, we see there is a drastic reduction in prediction accuracy on test data. PCA gets the top 100 dimensions which store the maximum variance regarding the data and this is not enough to properly classify the dataset. This is because the classification information does depend on even the lowly affecting dimensions and not just the top 100.

1. Auto Encode followed by classification

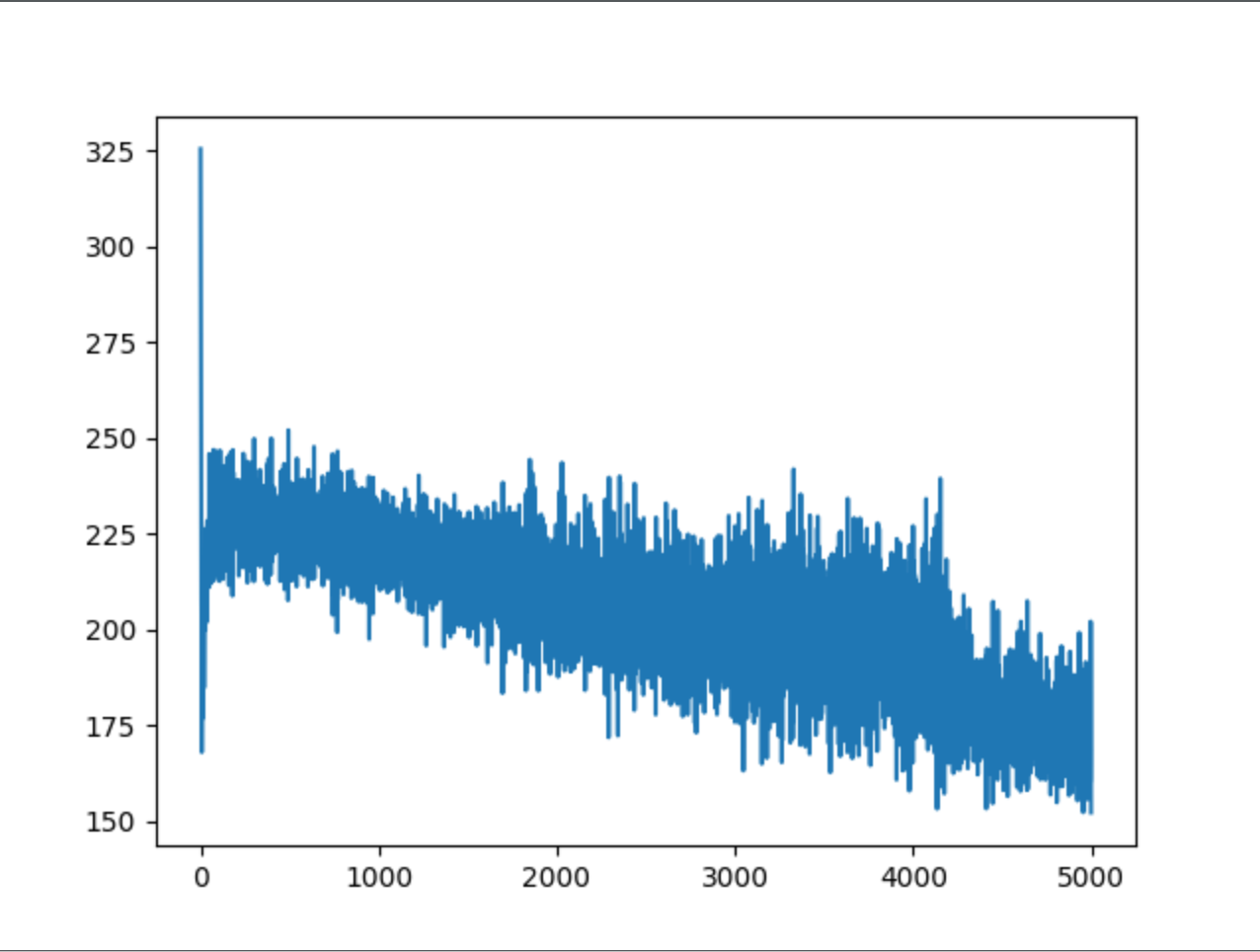
Following are the plots for MSE between input and output layer after the encoding, decoding phase.

**Autoencoder using Sigmoid:**



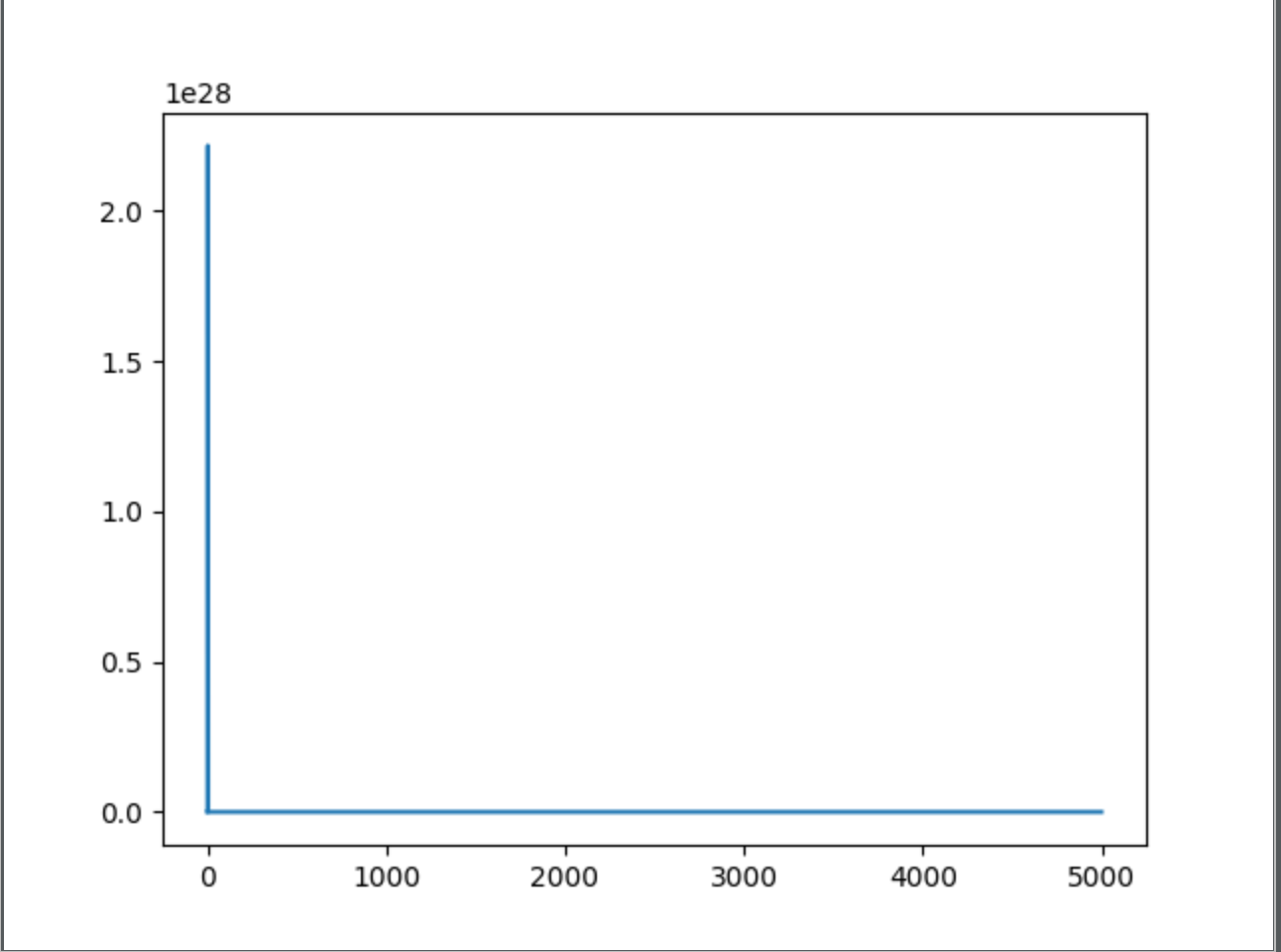
MSE vs Epoch

**Autoencoder using tanh:**

****

MSE vs Epoch

**Autoencoder using Relu:**



MSE vs Epoch

**Neural Nets**

**Using Sigmoid as the activation function:**

FOR I= 3 Test accuracy: 19.29%

FOR I= 6 Test accuracy: 17.86%

FOR I= 9 Test accuracy: 23.57%

FOR I= 12 Test accuracy: 25.43%

FOR I= 15 Test accuracy: 22.86%

FOR I= 18 Test accuracy: 25.00%

FOR I= 21 Test accuracy: 22.86%

FOR I= 24 Test accuracy: 30.00%

FOR I= 27 Test accuracy: 9.29%

FOR I= 30 Test accuracy: 14.29%

FOR I= 33 Test accuracy: 7.86%

**Using Relu as the activation function:**

FOR I= 3 Test accuracy: 16.43%

FOR I= 6 Test accuracy: 19.29%

FOR I= 9 Test accuracy: 24.29%

FOR I= 12 Test accuracy: 29.29%

FOR I= 15 Test accuracy: 2.86%

FOR I= 18 Test accuracy: 5.00%

FOR I= 21 Test accuracy: 20.00%

FOR I= 24 Test accuracy: 20.71%

FOR I= 27 Test accuracy: 24.29%

FOR I= 30 Test accuracy: 22.86%

FOR I= 33 Test accuracy: 19.29%

**Using TanH as the activation function:**

FOR I= 3 Test accuracy: 15.71%

FOR I= 6 Test accuracy: 16.43%

FOR I= 9 Test accuracy: 24.29%

FOR I= 12 Test accuracy: 31.43%

FOR I= 15 Test accuracy: 24.29%

FOR I= 18 Test accuracy: 17.86%

FOR I= 21 Test accuracy: 20.00%

FOR I= 24 Test accuracy: 22.86%

FOR I= 27 Test accuracy: 26.43%

FOR I= 30 Test accuracy: 27.14%

FOR I= 33 Test accuracy: 22.14%

**1 vs all SVM classifier**

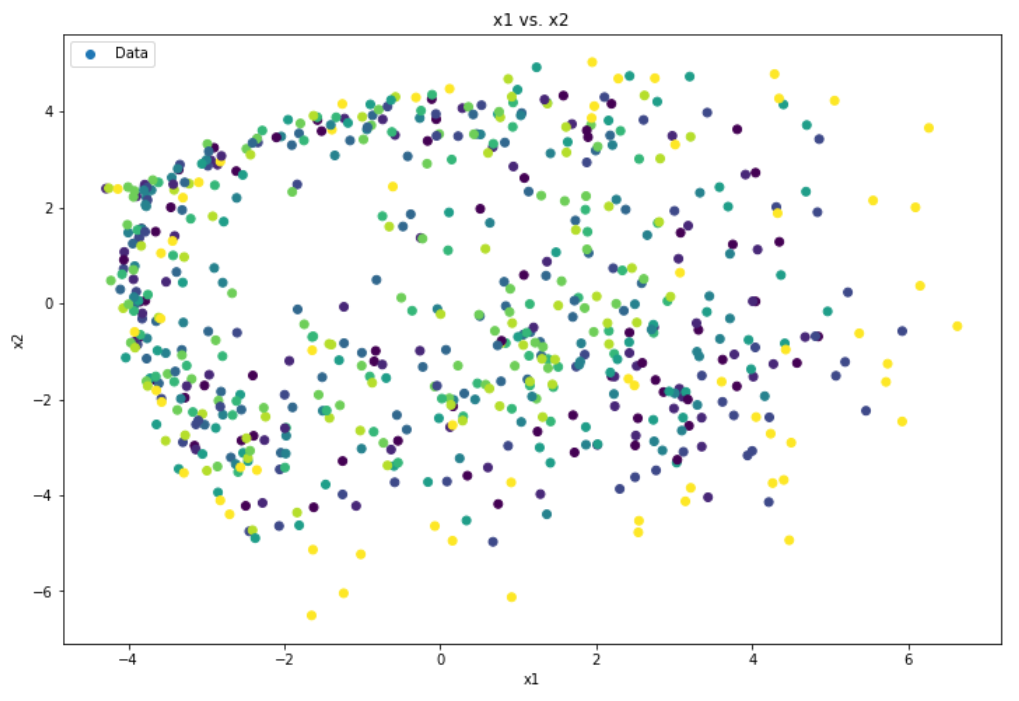
Final classification error on test data: Gave me an error multiply real and complex numbers

**1 vs all Logistic regression**

Final classification error on test data: 181 misclassifications

After applying AutoEncoder and then classifying, we see there is a drastic reduction in prediction accuracy on test data. PCA gets the top 100 dimensions which store the maximum variance regarding the data and this is not enough to properly classify the dataset. This is because the classification information does depend on even the lowly affecting dimensions and not just the top 100.

1. Visualize data after applying PCA

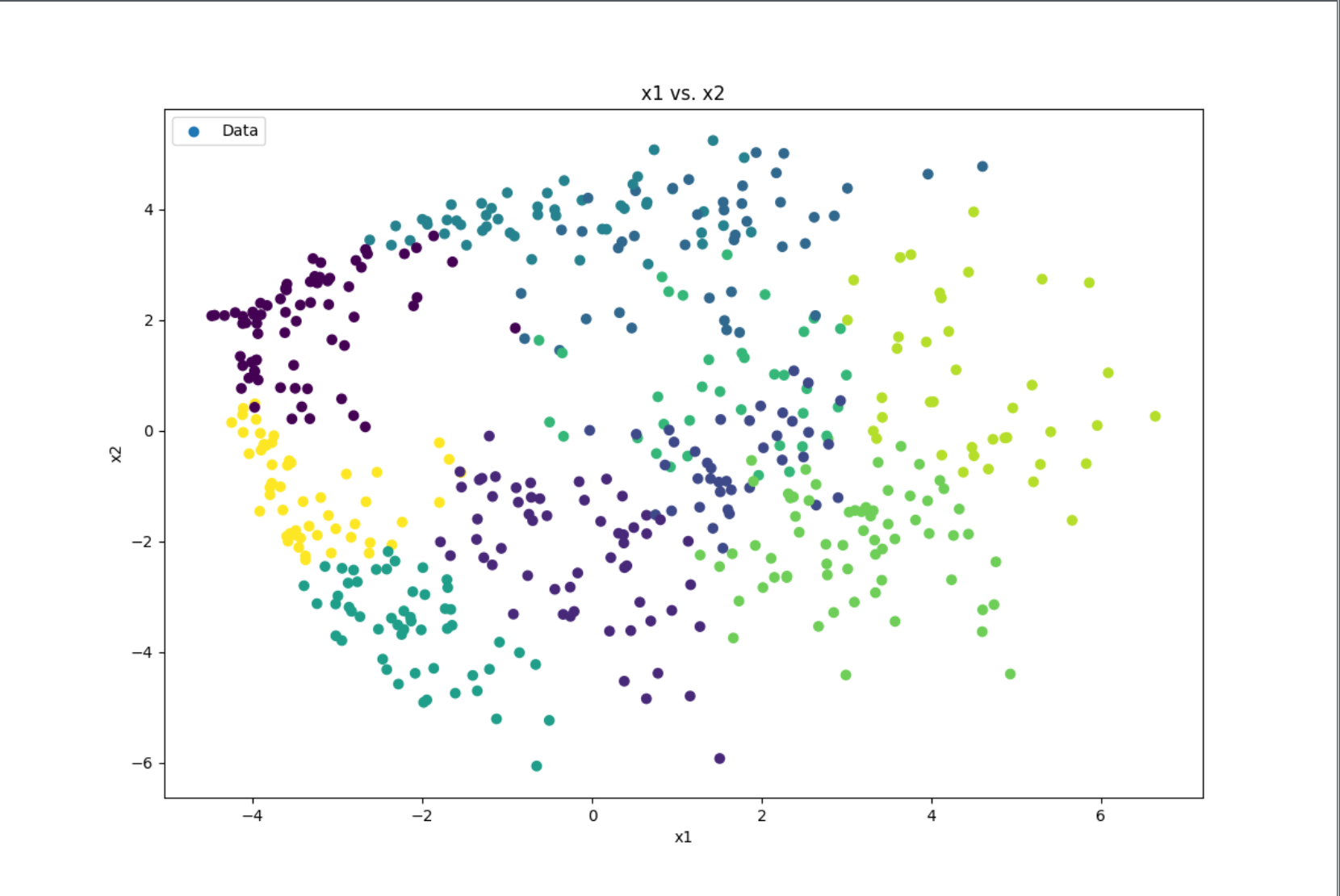


After reducing the dimensionality of data to 2 using PCA, we can see that there is no visible structure and data is not separated according to classes. Here each color represents a class and the colors are mixed up in the plot.

PCA transforms the data into a new lower dimension and captures the information regrading the maximum variance in the data. In this example, as we can see that the top 2 Eigen vectors do not capture any information about the true classification of the data points. Hence, the rendered plot does not separate the data according to their classes. This plot gives more insight to the previous poor classification results after applying PCA.

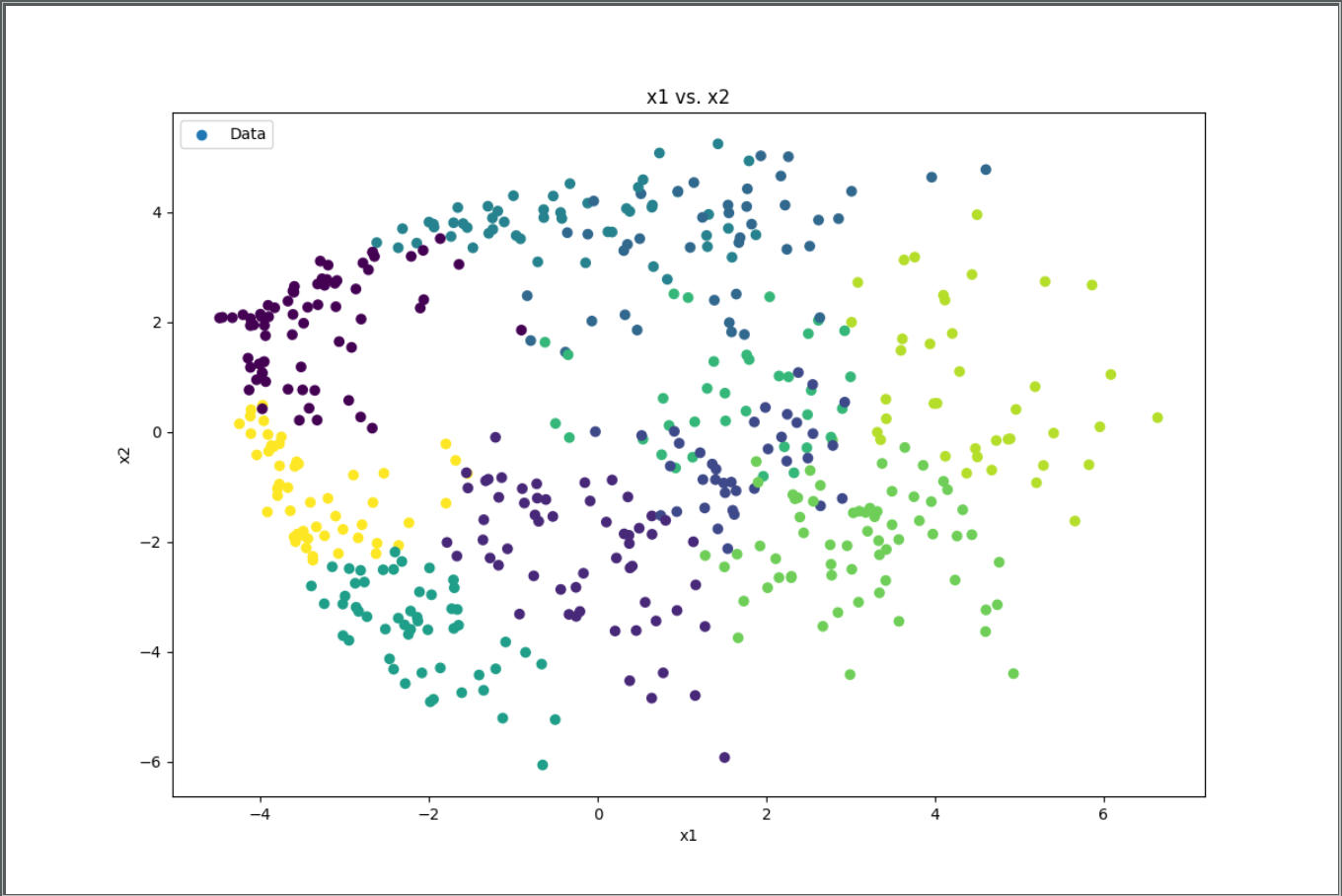
1. K-MEANS

**K means of original data:**

****

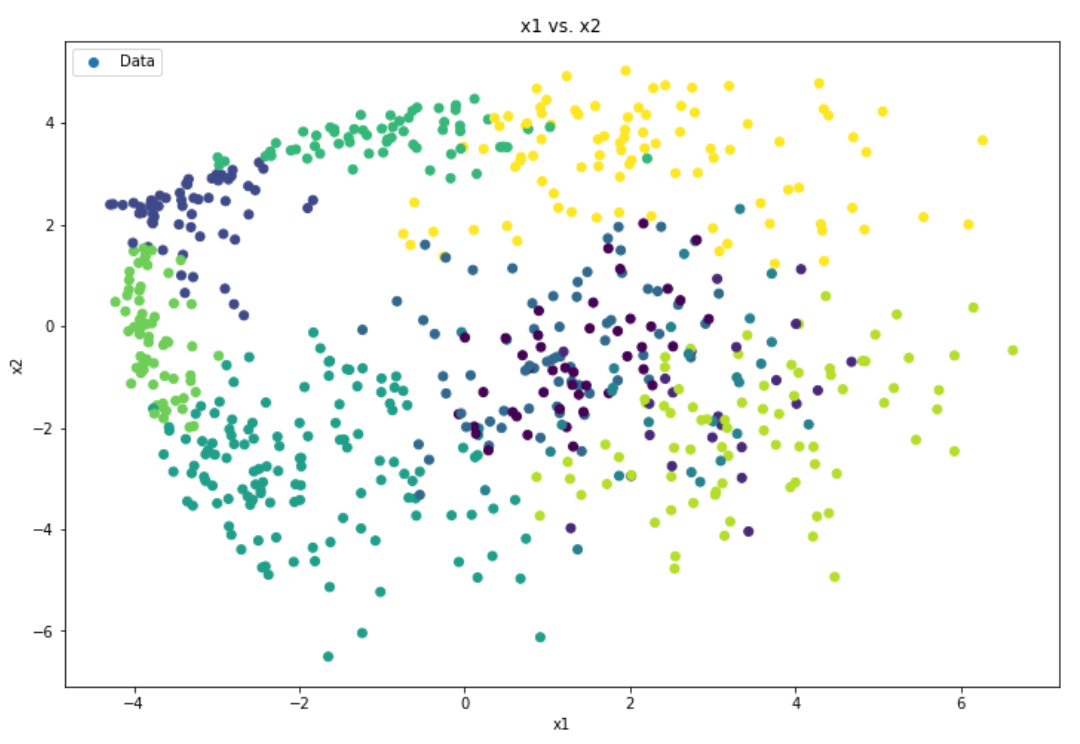
There are 563 misclassifications when k-means is used to cluster the given data. K-means is not able to recover the true clustering of the given data since is relies on the distance of each point from the cluster centers. K-means being unsurpervised, it does not learn the actual labels of each of the data points. Thus, it fails to classify the previous given data even though forming them into clusters. K-means uniformly distributes the data points to the specified number of clusters without learning the true labels.

**K means after applying PCA d =100:**



After applying PCA to reduce the dimensionality of the original data to 2 and then applying k-means, we can see that there is no change in the clustering and it the same as applying k-means on original data. PCA gives directions that capture maximum variance in the given data and combined with k-means it won’t affect the true classification capabilities.

1. Spectral Clustering



Spectral clustering, being an unsupervised learning task, tries to cluster the by finding the KNN and then building a similarity graph when then used k-means over. Hence, it is not a reliable method to classify the given dataset of images since it does not actually learn from the true labels provided in the data

The above clustering was determined using K = 10 nearest neighbors and σ = 25. The number of misclassifications were 603.

There is no particular difference between the misclassification rate between Spectral and K means. They both give different values for different values of their hyper parameters. Since both of them are unsupervised and clustering techniques it doesn’t make sense comparing them for a classification problem.

Keeping K = 10

|  |  |
| --- | --- |
| *σ2* value | # misclassifications |
| 0.5 | 578 |
| 5 | 642 |
| 10 | 633 |
| 25 | 601 |
| 50 | 580 |
| 75 | 571 |
| 100 | 592 |

We see that there is no particular pattern here since the clustering strategy learned by the algorithm is different from the classification strategy we have specified on the images.

Similarly, keeping σ2 = 10 and varying the value of K,

|  |  |
| --- | --- |
| K value | # misclassifications |
| 5 | 523 |
| 10 | 540 |
| 15 | 626 |
| 25 | 640 |
| 30 | 614 |
| 35 | 617 |
| 40 | 589 |

Once again, we see that there is no particular pattern here in the number of misclassifications due to the clustering strategy being different than classification.

NOTE: To run any of the following results you can do so by running the main() in q3.py and putting desired question like a(), b() method in the main.