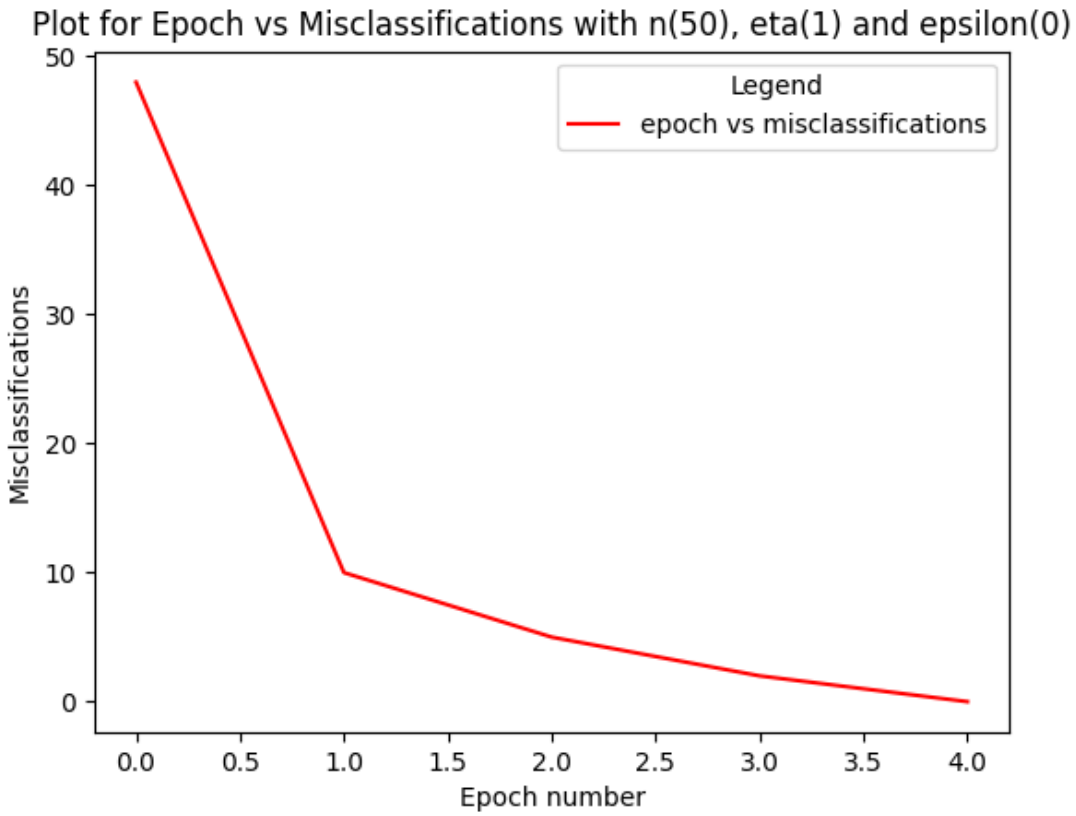
Name: Sai Anish Garapati
UIN: 650208577

Assignment_3:

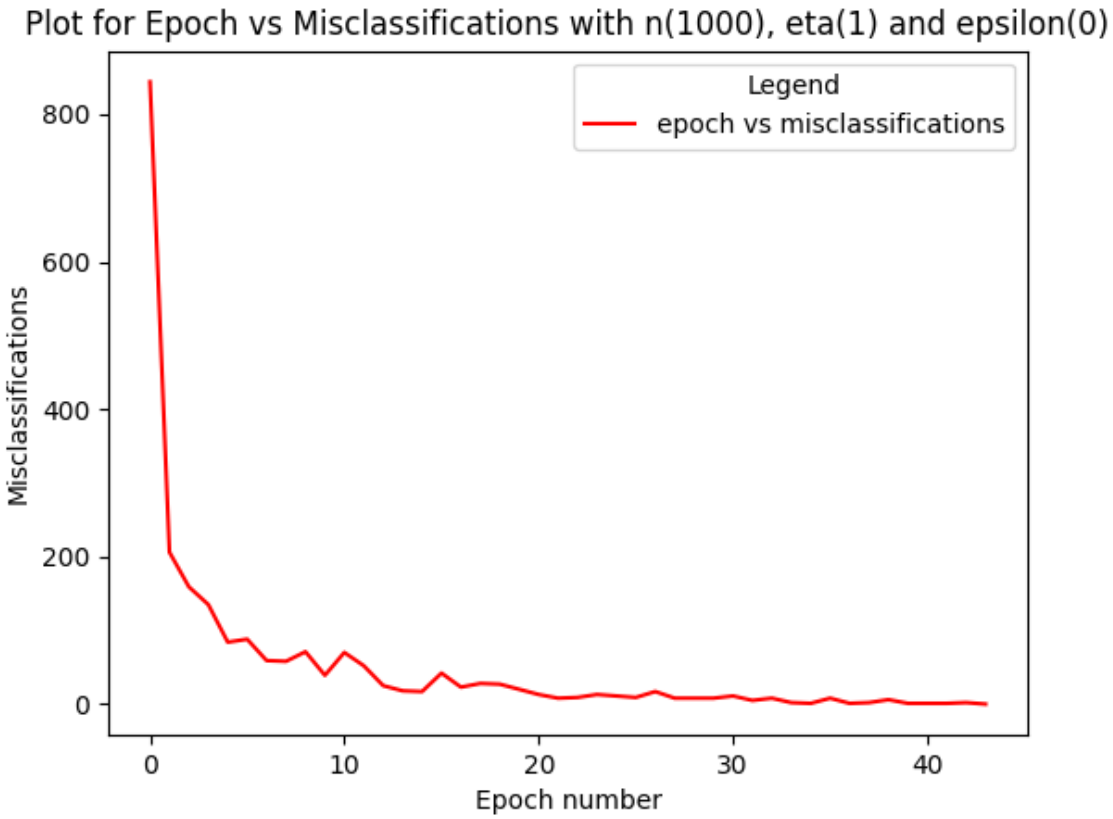
1.(f) Running PTA for n = 50, eta = 1 and epsilon = 0:
-> PTA terminated in 4 epochs with 0 errors(Error%: 0%) in the training set.



-> Percentage of misclassified test samples:45.14%

-> This discrepancy in error percentage in the training set(0%) and test set(45.14%) is because the sample size on which the model is trained (n = 50) is very less than the test set size (10000) and the model fails to predict properly on the test set.

1.(g) Running PTA for n = 1000, eta = 1 and epsilon = 0:
-> PTA terminated in 43 epochs with 0 errors(Error%: 0%) in the training set.



-> Percentage of misclassified test samples: 17.57%

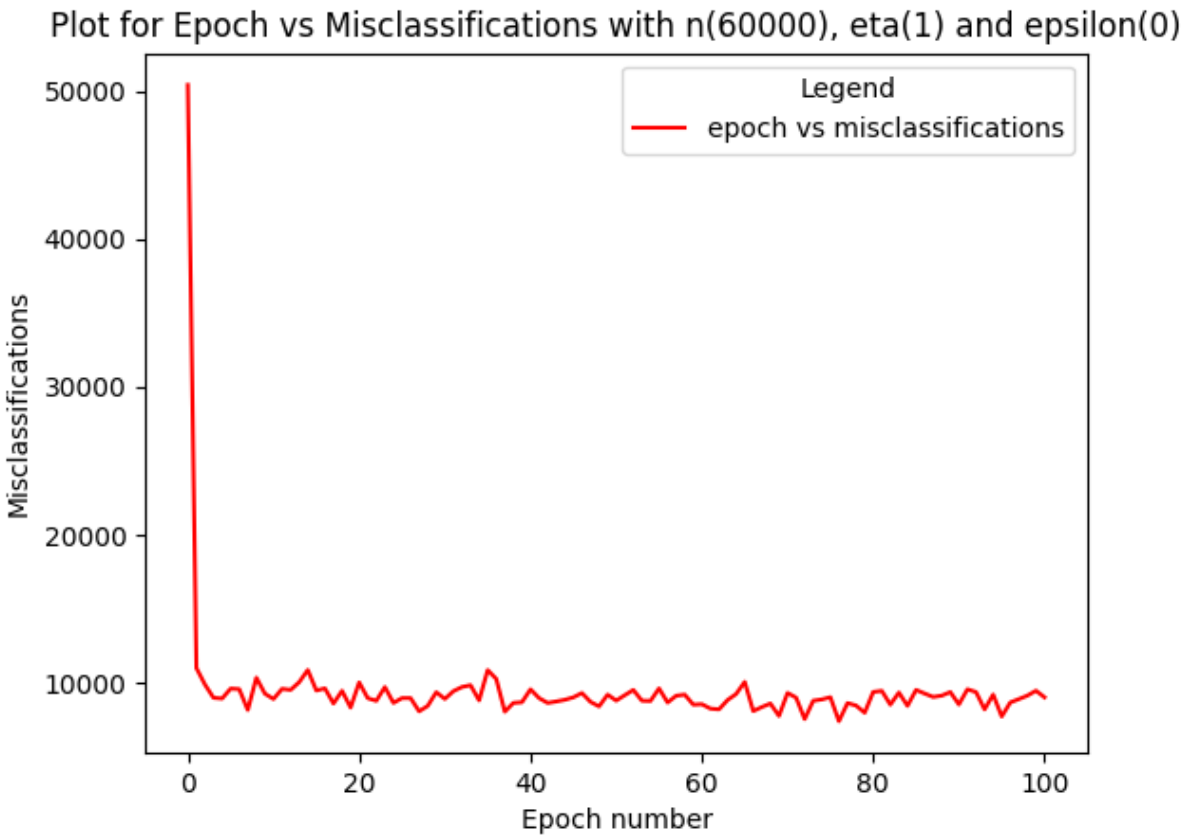
-> This discrepancy in error percentage in the training set(0%) and test set(17.57%) is because the sample size on which the model is trained (n = 1000) is very less than the test set size (10000) and the model fails to generalize for many examples.

-> The test error percentage in this case (17.57%) is better than the error percentage in the previous case(45.14%) due to an increase in the training set size (from 50 to 1000) which helps the model to get better weights.

1.(h) Running PTA for n = 60000, eta = 1 and epsilon = 0:

-> PTA fails to converge in this case which can be seen from the plot of epochs vs errors for the first 100 iterations. It keeps fluctuating at around 8000 errors.

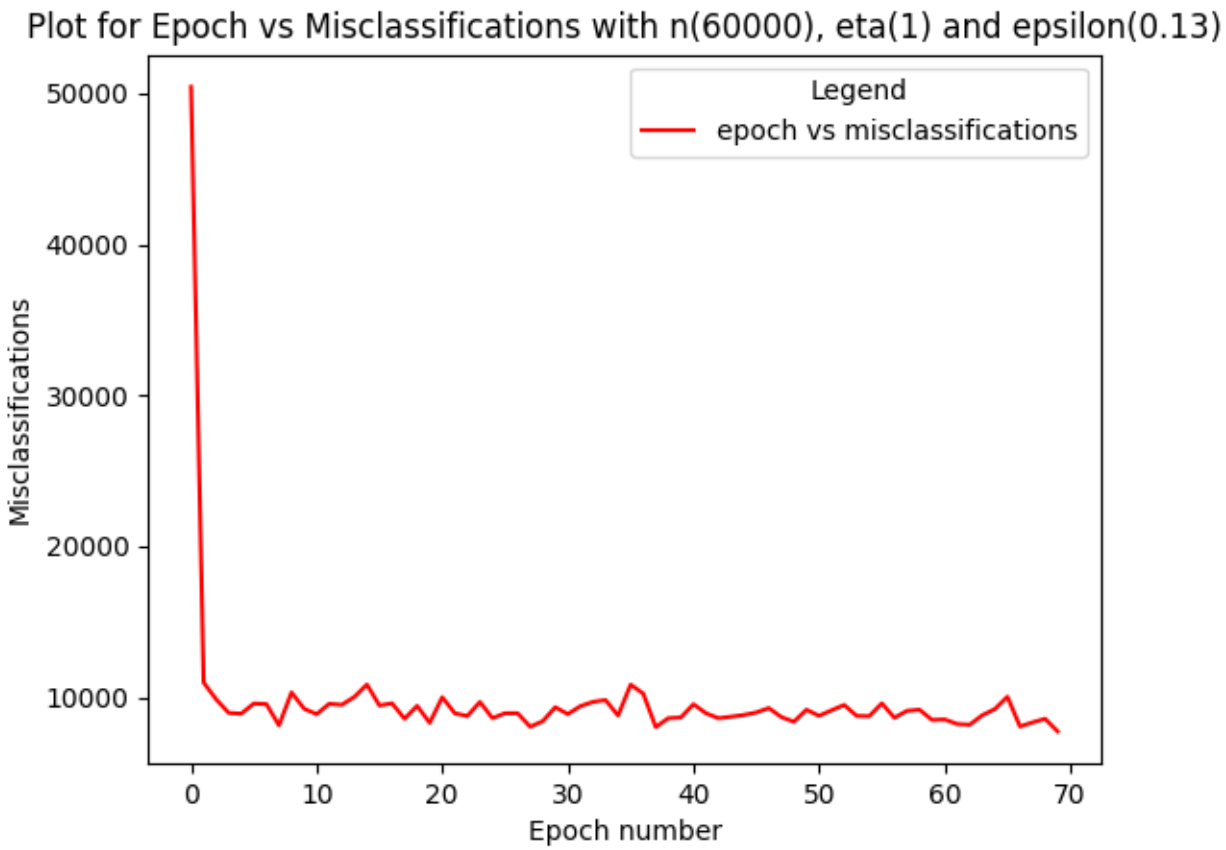
-> PTA was stopped after 100 epochs with 9003 errors(Error%: 15.005) in the training set.



-> Percentage of misclassified test samples: 15.44%

-> PTA fails to converge with 0 errors in the training set in this case due to the large training set (n=60000). Updating weights for a lot of misclassifications could be resulting in new errors due to larger delta in the weights after each iteration.

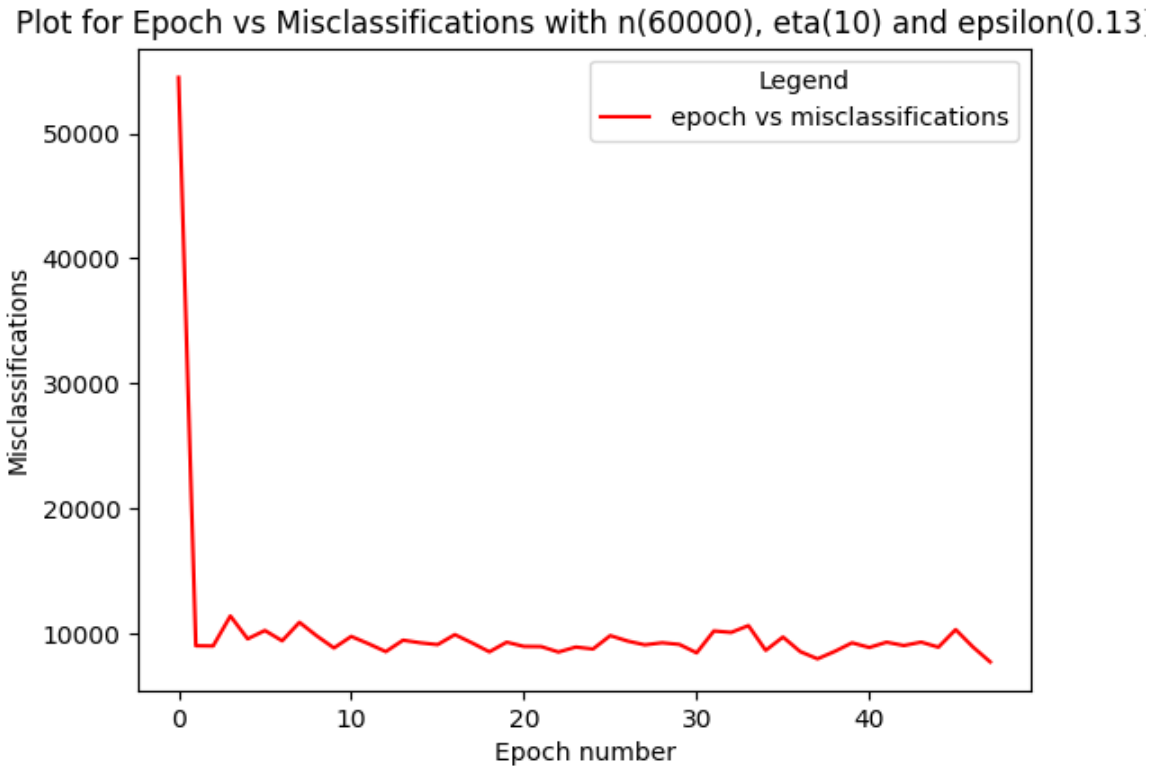
1.(i) By trial and error, an epsilon value of 0.13 is allowing the PTA to terminate.
1.(i).1 Running PTA on n = 60000, eta = 1, epsilon = 0.13:
-> PTA terminated after 69 epochs with 7745 errors(Error%: 12.91) in the training set



-> Percentage of misclassified test samples: 13.77%

1.(i).2 Running PTA on n = 60000, eta = 10, epsilon = 0.13:

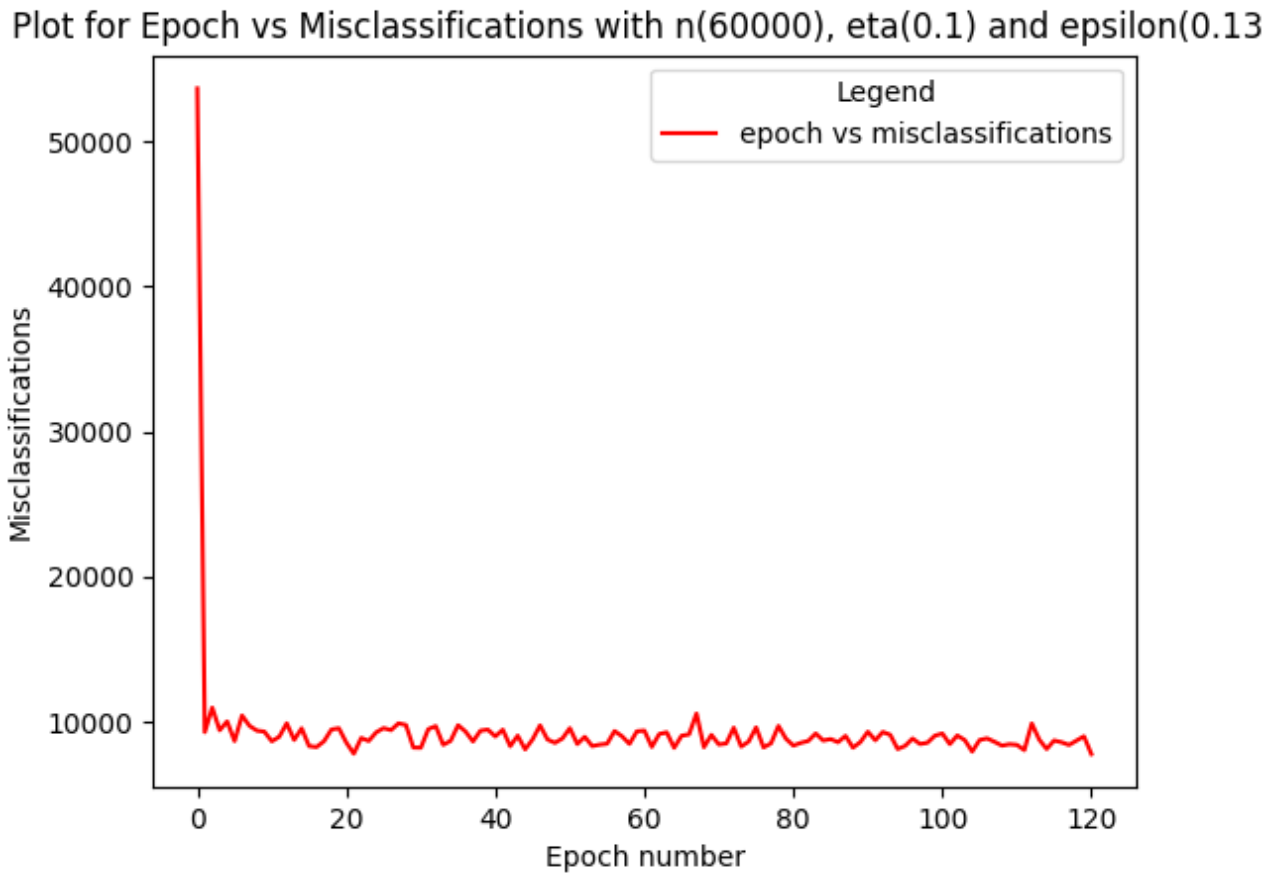
-> PTA terminated after 47 epochs with 7725 errors(Error%: 12.875%) in the training set



-> Percentage of misclassified test samples: 13.65%

1.(i).3 Running PTA on n = 60000, eta = 0.1, epsilon = 0.13:

-> PTA terminated after 120 epochs with 7775 errors(Error%: 12.96%) in the training set



-> Percentage of misclassified test samples: 13.78%

-> With any of the above choices of eta, irrespective of the differences in initial training weights, PTA terminated with a similar error percentage in the training set for epsilon=0.13. The test set error percentage is also almost the same. The only discrepancy in the above 3 cases is the number of epochs taken for the PTA to terminate which increased as the value of eta (learning rate) decreased.

Python code:

```
# Name: Sai Anish Garapati
# UIN: 650208577

import numpy as np
import matplotlib.pyplot as plt
from mlxtend.data import loadlocal_mnist

np.random.seed(2021)

def unit_activation(vector, W_train):
    local_field = np.dot(W_train, vector.reshape(784, 1))
    activated_output = np.array([int(lf >= 0) for lf in local_field]).reshape(10, 1)
    return activated_output

def calculate_errors(X_train, Y_train, W_train):
    errors = 0
    for vector, label in zip(X_train, Y_train):
        local_field = np.dot(W_train, vector.reshape(784, 1))
        if (local_field.argmax() != label):
            errors += 1
    return errors

def multicategory_PTA(n, eta, epsilon):
    X_train = train_set[:n, :]
    Y_train = train_labels[:n]
    W_train = np.random.randn(10, 784)
    epoch = 0
    errors = []
    print('Training model with n={}, eta={}, epsilon={}'.format(n, eta, epsilon))
    while (True):
        # Calculating misclassifications
        errors.append(calculate_errors(X_train, Y_train, W_train))

        # Condition to break for epsilon=0 for n=60000
        # if ((errors[-1]/n <= epsilon) or (epoch == 100)):
        #    break
        if (errors[-1]/n <= epsilon):
            break

        for vector, label in zip(X_train, Y_train):
            label_vector = np.zeros(shape=(10, 1))
            label_vector[label] = 1
            W_train = W_train + eta*np.dot((label_vector - unit_activation(vector,
W_train)), vector.reshape(1, 784))
        epoch += 1
    print('Training model completed in {} epochs with {} errors(Error%: {}%) in the
training set'.format(epoch, errors[-1], errors[-1]/n * 100))

    plt.title('Plot for Epoch vs Misclassifications with n({}), eta({}) and
```

```
epsilon({})'.format(n, eta, epsilon))
    plt.xlabel('Epoch number')
    plt.ylabel('Misclassifications')
    plt.plot([i for i in range(0, epoch + 1)], errors, 'r', label = 'epoch vs
misclassifications')
    plt.legend(title='Legend')
    plt.show()

    # Calculating Testset accuracy
    test_errors = calculate_errors(test_set, test_labels, W_train)
    print('Test Error percentage for n = {}, eta = {}, epsilon = {}: {}%'.format(n, eta,
epsilon, test_errors/test_set.shape[0] * 100))
    print("\n")

if (__name__ == '__main__'):
    train_set, train_labels = loadlocal_mnist(
            images_path='train-images.idx3-ubyte',
            labels_path='train-labels.idx1-ubyte')
    test_set, test_labels = loadlocal_mnist(
            images_path='t10k-images.idx3-ubyte',
            labels_path='t10k-labels.idx1-ubyte')

    print('\n')
    multicategory_PTA(n=50, eta=1, epsilon=0)
    multicategory_PTA(n=1000, eta=1, epsilon=0)
    # multicategory_PTA(n=60000, eta=1, epsilon=0)
    # epsilon=0.13 is chosen depending on the result from above

    multicategory_PTA(n=60000, eta=1, epsilon=0.13)
    multicategory_PTA(n=60000, eta=10, epsilon=0.13)
    multicategory_PTA(n=60000, eta=0.1, epsilon=0.13)
```