Submitted By: Abhishek Ayachit (862188073)
Deep Learning: Assignment 1
Date: 04/24/2020

1. **Write the code for downloading and formatting the data.**

```
[66] (x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()

[67] x_train = x_train[:50000]
     y_train = y_train[:50000]

[68] x_train = x_train.reshape(50000, 784)
     x_test = x_test.reshape(10000, 784)

[69] x_train = x_train/255
     x_test = x_test/255

[70] n_classes = 10
     y_train_targets = y_train.reshape(-1)
     y_train = np.eye(n_classes)[y_train_targets]

[71] n_classes = 10
     y_test_targets = y_test.reshape(-1)
     y_test = np.eye(n_classes)[y_test_targets]
```

2. **Write the code for minibatch SGD implementation for your linear MNIST classifier.**

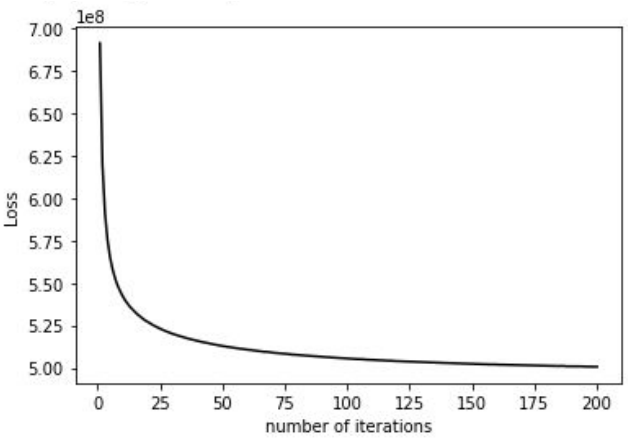
```
n = len(y)
loss_arr = np.zeros(iteration)
G=0
for iter in range(iteration):
    loss =0.0
    rand = np.random.permutation(n)
    X = X[rand]
    y = y[rand]
    for i in range(0,n,batch_size):
        x_new = X[i:i+batch_size]
        y_new = y[i:i+batch_size]

        pred = np.dot(x_new,weight)
        G += ( x_new.T.dot((pred - y_new)))
    G = (1/batch_size)*G
    weight = weight - l_r*G
    loss += calculate_loss(weight,X,y)
    loss_arr[iter]  = loss
```

3. **The role of batch size: Run your code with batch sizes B = 1, 10, 100, 1000. For each batch size,**

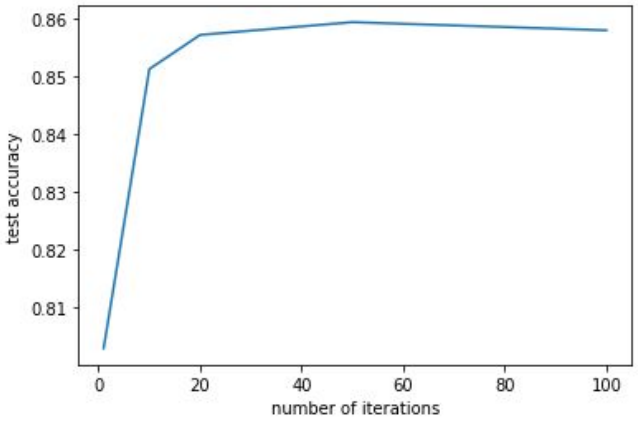
The best learning rate was identified as 0.5 and it is used for the following experiments and iterations were considered as 100 (experimentally).

For batch size = 1

- Time: 139
- Loss vs Iteration:

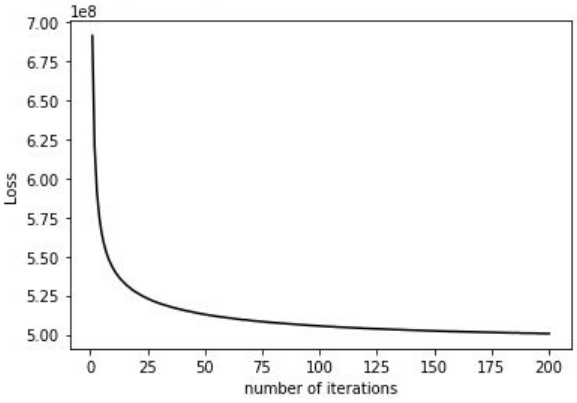


- Accuracy vs Iteration:

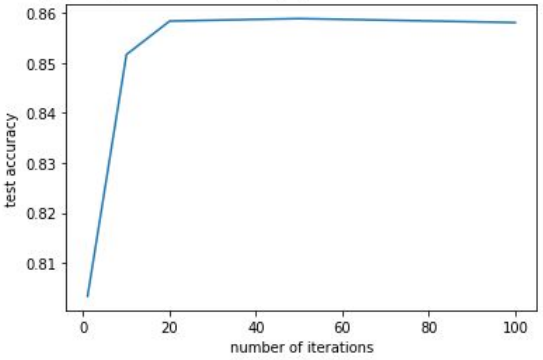


For batch size = 10

- Time: 123
- Loss vs Iteration:

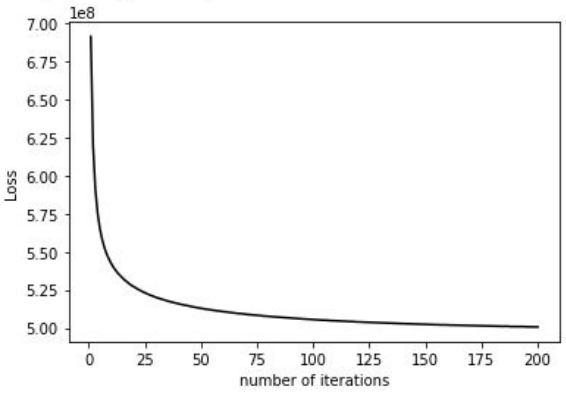


- Accuracy vs Iterations:

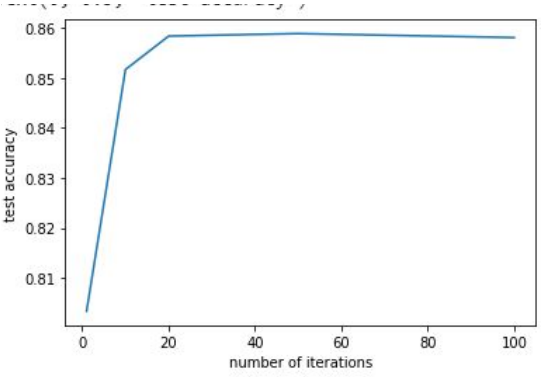


For batch size = 100

- Time: 89
- Loss vs Iteration:

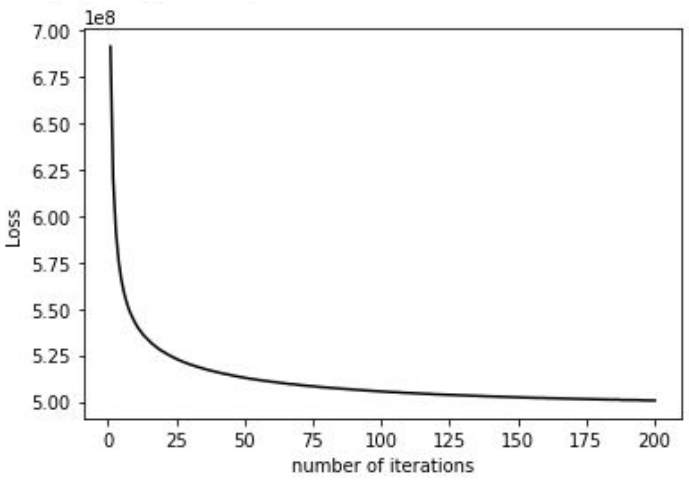


- Accuracy vs Iterations:

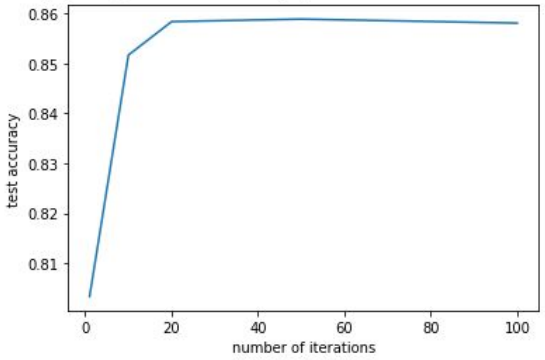


For batch size = 1000

- Time: 68
- Loss vs Iteration:

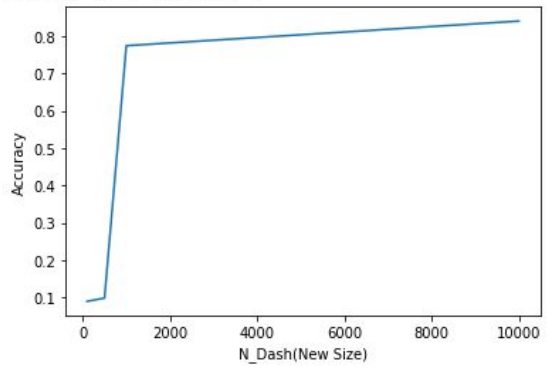


- Accuracy vs Iterations:



4. **Comment on Batch Size:** Batch size controls the accuracy of the estimate of the error gradient when training neural networks.
   The batch size affects the time as well as the accuracy. Therefore, as the batch size increases, the time reduces and the accuracy decreases as well.

5. **The role of training dataset size:** Let us reduce the training dataset size.Instead ofN = 50, 000, let us pick a subset S′ of size N′ from the original dataset without replacement and uniformly at random. Fix batch size to B = 100. Repeat the steps above for N′ ∈ {100, 500, 1000, 10000}.Comment on the accuracy as a function of dataset size.



6. **Run the linear MNIST classifier with batchsize B = 100 over the full dataset by using PyTorch or Tensorflow. Verify that it is consistent with your handcoded algorithm by comparing your results (the accuracy and training loss plots).**

```
[20] import numpy as np
     import mnist
     import matplotlib.pyplot as plt
     from keras.models import Sequential
     from keras.layers import Dense
     from keras.utils import to_categorical

[21] train_images = mnist.train_images()
     train_labels = mnist.train_labels()
     test_images = mnist.test_images()
     test_labels = mnist.test_labels()

[22] train_images = (train_images/255) - 0.5
     test_images = (test_images/255) - 0.5
     train_images = train_images.reshape(-1, 784)
     test_images = test_images.reshape(-1, 784)

[9]  model = Sequential()
     model.add(Dense(64, activation = 'relu', input_dim = 784))
     model.add(Dense(64, activation = 'relu'))
     model.add(Dense(10, activation = 'softmax'))

     model.compile(optimizer = 'sgd',loss = 'mean_squared_error',metrics = ['accuracy'])
     loss=model.fit(train_images,to_categorical(train_labels),epochs = 25,batch_size = 100)

     accuracy = model.evaluate(test_images,to_categorical(test_labels))
```