assignment1

October 13, 2019

1 Venkatesh Prasad Venkataramanan A53318036 Assignment 1

[2]:
```
import numpy as np
from matplotlib import pyplot as plt
```

[5]:
```
import MNISTtools
```

[6]:
```
help(MNISTtools.load)
```

```
Help on function load in module MNISTtools:

load(dataset='training', path=None)
    Import either the training or testing MNIST data set.
    It returns a pair with the first element being the collection of
    images stacked in columns and the second element being a vector
    of corresponding labels from 0 to 9.

    Arguments:
        dataset (string, optional): either "training" or "testing".
            (default: "training")
        path (string, optional): the path pointing to the MNIST dataset
            If path=None, it looks succesively for the dataset at:
            '/datasets/MNIST' and './MNIST'. (default: None)

    Example:
        x, lbl = load(dataset="testing", path="/Folder/for/MNIST")
```

1.1 Question 1

[7]:
```
print ("Answer 1")

xtrain, ltrain = MNISTtools.load(dataset="training", path="/datasets/
 ↪MNIST")#loaded data

print ("The shape of xtrain is :",xtrain.shape)
```

```
print ("The shape of ltrain is :",ltrain.shape)
print ("The size of the training dataset is : ",xtrain.shape[1])
print ("The feature dimension is : ",xtrain.shape[0])
```

```
Answer 1
The shape of xtrain is : (784, 60000)
The shape of ltrain is : (60000,)
The size of the training dataset is :  60000
The feature dimension is :  784
```

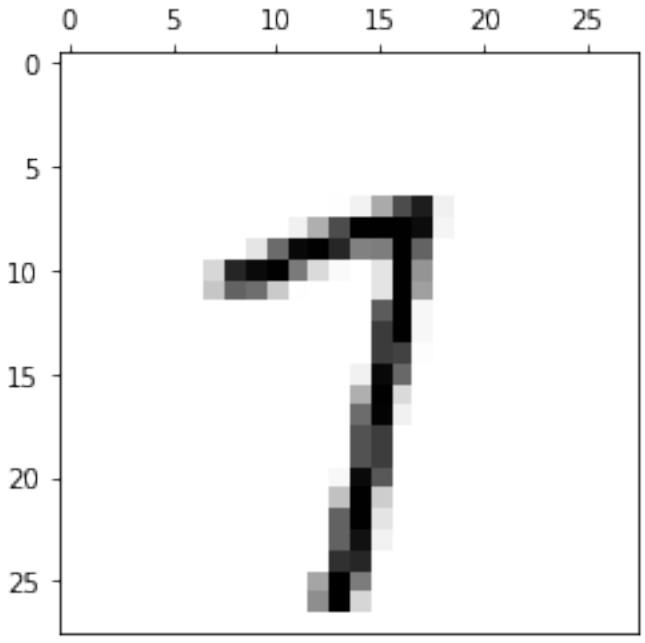
1.2 Question 2

[93]:
```
print ("Answer 2")

MNISTtools.show(xtrain[:, 42])#displaying the image at that index

print ("ltrain value for that index is :",ltrain[42])
print ("They are similar")
```

```
Answer 2
```




```
ltrain value for that index is : 7
They are similar
```

1.3 Question 3

[94]:
```
print ("Answer 3")

print("The minimum of xtrain is :",np.amin(xtrain))
print("The maximum of xtrain is :",np.amax(xtrain))
print("The type of xtrain is :",xtrain.dtype)
```

```
Answer 3
The minimum of xtrain is : 0
The maximum of xtrain is : 255
The type of xtrain is : uint8
```

1.4 Question 4

[95]:
```
print ("Answer 4")

xtrain = xtrain.astype(np.float32)#float conversion

def normalize_MNIST_images(x):
    x = -1.0 + (2*x)/255#mapping [0,255] to [-1,1]
    return x


xtrain = normalize_MNIST_images(xtrain)
```

```
Answer 4
```

1.5 Question 5

[96]:
```
print ("Answer 5")

def label2onehot(lbl):#function to convert labels to one-hot codes
    d = np.zeros((lbl.max() + 1, lbl.size))
    d[lbl, np.arange(lbl.size)] = 1
    return d

dtrain = label2onehot(ltrain)
print (dtrain[:,42])
print (ltrain[42])
print ("We can see that the one-hot code matches.")
```

```
Answer 5
[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
```

```
7
We can see that the one-hot code matches.
```

1.6 Question 6

[97]:
```
print ("Answer 6")

def onehot2label(d):#function to convert one-hot codes back to label
    lbl = d.argmax(axis=0)
    return lbl

ltrain = onehot2label(dtrain)
print("Converting back from one-hot to label :",ltrain[42])
print ("We can see that it matches ltrain[42]")
```

```
Answer 6
Converting back from one-hot to label : 7
We can see that it matches ltrain[42]
```

1.7 Question 7

[98]:
```
print ("Answer 7")

def softmax(a):#softmax function
    M = np.amax(a)
    num = den = np.exp(a - M)
    y= ((num)/(den.sum(axis=0)))
    return y
```

```
Answer 7
```

1.8 Question 10

[99]:
```
print ("Answer 10")

def softmaxp(a,e):
    y =  softmax(a)
    element_wise = np.multiply(y,e)
    ans = ((element_wise) - (element_wise.sum(axis=0)*y))
    return ans
```

```
Answer 10
```

1.9 Question 11

[101]:
```
print ("Answer 11")

eps         = 1e-6                          # finite difference step
a           = np.random.randn(10, 200)      # random inputs
e           = np.random.randn(10, 200)      # random directions
diff        = softmaxp(a, e)
diff_approx = (softmax(a + eps*e) - softmax(a))/(eps)#the approximation formula
rel_error   = np.abs(diff - diff_approx).mean() / np.abs(diff_approx).mean()

print(rel_error, 'should be smaller than 1e-6')
print("As we can see, it is smaller than the said value")
```

```
Answer 11
4.915187031417891e-07 should be smaller than 1e-6
As we can see, it is smaller than the said value
```

1.10 Question 12

[102]:
```
print ("Answer 12")

def relu(a):
    return np.maximum(a, 0)

def relup(a,e):
    y = (a > 0) * e
    return y

eps         = 1e-6                          # finite difference step
a           = np.random.randn(10, 200)      # random inputs
e           = np.random.randn(10, 200)      # random directions
diff        = relup(a, e)
diff_approx = (relu(a + eps*e) - relu(a))/(eps)
rel_error   = np.abs(diff - diff_approx).mean() / np.abs(diff_approx).mean()

print(rel_error, 'should be smaller than 1e-6')
print ("As we can see, it is smaller than the said value")
```

```
Answer 12
4.2143301280670875e-11 should be smaller than 1e-6
As we can see, it is smaller than the said value
```

1.11 Question 13

[103]:
```
print ("Answer 13")

def init_shallow(Ni, Nh, No):#initiate a network of random values
    b1 = np.random.randn(Nh, 1)   / np.sqrt((Ni+1.)/2.)
    W1 = np.random.randn(Nh, Ni)  / np.sqrt((Ni+1.)/2.)
    b2 = np.random.randn(No, 1)   / np.sqrt((Nh+1.))
    W2 = np.random.randn(No, Nh)  / np.sqrt((Nh+1.))
    return W1, b1, W2, b2

Ni = xtrain.shape[0]
Nh = 64
No = dtrain.shape[0]
netinit = init_shallow(Ni, Nh, No)
```

```
Answer 13
```

1.12 Question 14

[104]:
```
print ("Answer 14")

def forwardprop_shallow(x, net):#forward propagate over the network
    W1 = net[0]
    b1 = net[1]
    W2 = net[2]
    b2 = net[3]

    a1 = W1.dot(x) + b1
    #COMPLETE
    h1      = relu(a1)
    a2      = W2.dot(h1) + b2
    y       = softmax(a2)

    return y

yinit = forwardprop_shallow(xtrain, netinit)
```

```
Answer 14
```

1.13 Question 15

[106]:
```
print ("Answer 15")

def eval_loss(y, d):#evaluates average cross-entropy loss
    loss_vector = np.multiply(d,np.log(y))
    loss = - np.mean(loss_vector)

    return loss

print(eval_loss(yinit, dtrain), 'should be around .26')
print("Hence we can see that it satisfies the criteria")
```

```
Answer 15
0.2609550551344122 should be around .26
Hence we can see that it satisfies the criteria
```

1.14 Question 16

[112]:
```
print ("Answer 16")

def eval_perfs(y, lbl):#calculates percentage of misclassified samples
    c = np.equal(onehot2label(y), lbl)
    print (c)
    c = np.sum(c)
    c = (c/60000)*100
    return 100 - c

print("The percentage of misclassified samples is : ",eval_perfs(yinit, ltrain))
print("This is because since a random network is used for classification, there␣
 ↪is 1/10 probability that the prediction is correct. Hence we are getting 90%␣
 ↪misclassified samples.")
```

```
Answer 16
[False False False … False False False]
The percentage of misclassified samples is :  90.09166666666667
This is because since a random network is used for classification, there is 1/10
probability that the prediction is correct. Hence we are getting 90%
misclassified samples.
```

1.15 Question 17

[111]:
```
print ("Answer 17")

def update_shallow(x, d, net, gamma=.05):#updating weights and biases
    W1 = net[0]
    b1 = net[1]
    W2 = net[2]
    b2 = net[3]
    Ni = W1.shape[1]
    Nh = W1.shape[0]
    No = W2.shape[0]
    gamma = gamma / x.shape[1] # normalized by the training dataset size
    #
    a1 = W1.dot(x) + b1
    h1 = relu(a1)
    a2 = W2.dot(h1) + b2
    y = softmax(a2)
    #
    d2 = softmaxp(a2, -d/y)
    d1 = relup(a1, W2.T.dot(d2))
    #
    W2 -= gamma*d2.dot(h1.T)
    W1 -= gamma*d1.dot(x.T)
    b2 -= gamma*d2.sum(axis=1).reshape(No,1)
    b1 -= gamma*d1.sum(axis=1).reshape(Nh,1)

    return W1, b1, W2, b2

print ("The proof has been attached.")
```

```
Answer 17
The proof has been attached.
```

1.16 Question 18

[115]:
```
print ("Answer 18")

def backprop_shallow(x, d, net, T, gamma = 0.05):#backpropagation function
    lbl = onehot2label(d)
    for t in range(T):
        net = update_shallow(x,d,net,gamma)
        y = forwardprop_shallow(x,net)
        loss = eval_loss(y,d)
        training_error = eval_perfs(y,lbl)
        print ("The loss is :",loss)
```

```
        print ("The training error is :",training_error)

    return net

nettrain = backprop_shallow(xtrain, dtrain, netinit, 2)
nettrain = backprop_shallow(xtrain, dtrain, netinit, 100)
```

```
Answer 18
[False  True False … False False False]
The loss is : 0.21196815881309206
The training error is : 74.94833333333334
[False  True False … False False False]
The loss is : 0.20412524002677554
The training error is : 68.97333333333333
[False  True False … False False False]
The loss is : 0.19673498489298583
The training error is : 63.888333333333335
[False  True False … False False False]
The loss is : 0.18968055332466394
The training error is : 59.035
[False  True False … False False False]
The loss is : 0.1828885552937374
The training error is : 54.80833333333333
[False  True False … False False False]
The loss is : 0.17630116601378062
The training error is : 50.995000000000005
[False  True False … False  True False]
The loss is : 0.1698981187684028
The training error is : 47.695
[False  True False … False  True False]
The loss is : 0.16368890653742793
The training error is : 44.708333333333336
[False  True False … False  True False]
The loss is : 0.15774713542889324
The training error is : 42.14833333333333
[False  True False … False  True False]
The loss is : 0.15222607745008035
The training error is : 39.83833333333333
[False  True False … False  True False]
The loss is : 0.14708445665639763
The training error is : 37.85666666666667
[False  True False … False  True False]
The loss is : 0.14221798486630013
The training error is : 36.068333333333335
[False  True False …  True  True False]
The loss is : 0.137599080272382
The training error is : 34.516666666666666
```

```
[False  True  True …  True  True False]
The loss is : 0.13321443165406735
The training error is : 32.985
[False  True  True …  True  True False]
The loss is : 0.12905205175468595
The training error is : 31.723333333333343
[False  True  True …  True  True False]
The loss is : 0.12511249981382477
The training error is : 30.40166666666667
[False  True  True …  True  True  True]
The loss is : 0.12138561013641953
The training error is : 29.393333333333345
[False  True  True …  True  True  True]
The loss is : 0.11786070041988013
The training error is : 28.233333333333334
[False  True  True …  True  True  True]
The loss is : 0.11452527771707793
The training error is : 27.45833333333333
[False  True  True …  True  True  True]
The loss is : 0.11136571917549799
The training error is : 26.49333333333334
[False  True  True …  True  True  True]
The loss is : 0.1083721365630602
The training error is : 25.86500000000001
[False  True  True …  True  True  True]
The loss is : 0.10553875779518819
The training error is : 24.951666666666668
[False  True  True …  True  True  True]
The loss is : 0.1028609102027906
The training error is : 24.58333333333333
[False  True  True …  True  True  True]
The loss is : 0.10034004100489889
The training error is : 23.568333333333342
[False  True  True …  True  True  True]
The loss is : 0.09797809896271195
The training error is : 23.51166666666667
[ True  True  True …  True  True  True]
The loss is : 0.09577722378254441
The training error is : 22.48166666666667
[False  True  True …  True  True  True]
The loss is : 0.09376015270453793
The training error is : 22.763333333333335
[ True  True  True …  True  True  True]
The loss is : 0.09191206660160393
The training error is : 21.678333333333327
[ True  True  True …  True  True  True]
The loss is : 0.09038252307447846
The training error is : 22.58333333333333
```

```
[ True  True  True …  True  True  True]
The loss is : 0.08897658026384024
The training error is : 21.536666666666676
[ True  True  True …  True  True  True]
The loss is : 0.08834451422507462
The training error is : 23.578333333333333
[ True  True  True …  True  True  True]
The loss is : 0.08727754477710613
The training error is : 22.14
[False  True  True …  True  True  True]
The loss is : 0.0881235375443347
The training error is : 25.346666666666664
[ True  True  True …  True  True  True]
The loss is : 0.08619749083325977
The training error is : 22.760000000000005
[False  True  True …  True  True  True]
The loss is : 0.08792034438280122
The training error is : 26.27333333333334
[ True  True  True …  True  True  True]
The loss is : 0.08407044410629883
The training error is : 22.09166666666667
[False  True  True …  True  True  True]
The loss is : 0.08499692917695983
The training error is : 25.620000000000005
[ True  True  True …  True  True  True]
The loss is : 0.0809605277762379
The training error is : 21.12166666666667
[False  True  True …  True  True  True]
The loss is : 0.08117727961826345
The training error is : 24.224999999999994
[ True  True  True …  True  True  True]
The loss is : 0.07768765605532381
The training error is : 20.155
[False  True  True …  True  True  True]
The loss is : 0.07751947114858851
The training error is : 22.570000000000007
[ True  True  True …  True  True  True]
The loss is : 0.07476588812290835
The training error is : 19.398333333333326
[ True  True  True …  True  True  True]
The loss is : 0.07442723866737068
The training error is : 21.126666666666665
[ True  True  True …  True  True  True]
The loss is : 0.07228789022840565
The training error is : 18.763333333333335
[ True  True  True …  True  True  True]
The loss is : 0.07188489256842213
The training error is : 20.03333333333333
```

```
[ True  True  True …  True  True  True]
The loss is : 0.07018449846483125
The training error is : 18.328333333333333
[ True  True  True …  True  True  True]
The loss is : 0.06976971344317204
The training error is : 19.176666666666662
[ True  True  True …  True  True  True]
The loss is : 0.06837200412655316
The training error is : 17.968333333333334
[ True  True  True …  True  True  True]
The loss is : 0.06797308376977461
The training error is : 18.46166666666666
[ True  True  True …  True  True  True]
The loss is : 0.0667906188639469
The training error is : 17.703333333333333
[ True  True  True …  True  True  True]
The loss is : 0.0664171332503026
The training error is : 17.938333333333333
[ True  True  True …  True  True  True]
The loss is : 0.06538880410541455
The training error is : 17.5
[ True  True  True …  True  True  True]
The loss is : 0.06503965124916795
The training error is : 17.540000000000006
[ True  True  True …  True  True  True]
The loss is : 0.06412573771169654
The training error is : 17.281666666666666
[ True  True  True …  True  True  True]
The loss is : 0.06379091082659624
The training error is : 17.183333333333323
[False  True  True …  True  True  True]
The loss is : 0.06297039700115127
The training error is : 17.096666666666664
[ True  True  True …  True  True  True]
The loss is : 0.06264554908482686
The training error is : 16.88166666666666
[False  True  True …  True  True  True]
The loss is : 0.061896026714692465
The training error is : 16.924999999999997
[ True  True  True …  True  True  True]
The loss is : 0.061572085604326206
The training error is : 16.583333333333343
[False  True  True …  True  True  True]
The loss is : 0.06087845909024812
The training error is : 16.73166666666667
[ True  True  True …  True  True  True]
The loss is : 0.060547732585492293
The training error is : 16.308333333333337
```

```
[False  True  True …  True  True  True]
The loss is : 0.05990122775419289
The training error is : 16.52333333333334
[ True  True  True …  True  True  True]
The loss is : 0.05956548294785646
The training error is : 16.054999999999993
[False  True  True …  True  True  True]
The loss is : 0.0589599579689484
The training error is : 16.313333333333333
[ True  True  True …  True  True  True]
The loss is : 0.058619772749316096
The training error is : 15.766666666666666
[False  True  True …  True  True  True]
The loss is : 0.05805229455743454
The training error is : 16.09166666666667
[ True  True  True …  True  True  True]
The loss is : 0.05771010025382859
The training error is : 15.546666666666667
[False  True  True …  True  True  True]
The loss is : 0.05717775199622686
The training error is : 15.86333333333333
[ True  True  True …  True  True  True]
The loss is : 0.05683682353951389
The training error is : 15.310000000000002
[False  True  True …  True  True  True]
The loss is : 0.056340278430066455
The training error is : 15.625
[ True  True  True …  True  True  True]
The loss is : 0.056005351529891195
The training error is : 15.060000000000002
[False  True  True …  True  True  True]
The loss is : 0.05554121102580211
The training error is : 15.403333333333336
[ True  True  True …  True  True  True]
The loss is : 0.05521584931076593
The training error is : 14.873333333333335
[False  True  True …  True  True  True]
The loss is : 0.054783165461671564
The training error is : 15.169999999999987
[ True  True  True …  True  True  True]
The loss is : 0.05446872200002141
The training error is : 14.701666666666668
[False  True  True …  True  True  True]
The loss is : 0.05406306848831048
The training error is : 14.976666666666674
[ True  True  True …  True  True  True]
The loss is : 0.05375937322013152
The training error is : 14.510000000000005
```

```
[False  True  True …  True  True  True]
The loss is : 0.05338138857862652
The training error is : 14.791666666666671
[ True  True  True …  True  True  True]
The loss is : 0.05308883054224841
The training error is : 14.338333333333324
[False  True  True …  True  True  True]
The loss is : 0.052734346009654834
The training error is : 14.614999999999995
[ True  True  True …  True  True  True]
The loss is : 0.0524528054206503
The training error is : 14.168333333333337
[False  True  True …  True  True  True]
The loss is : 0.05212056740002614
The training error is : 14.450000000000003
[ True  True  True …  True  True  True]
The loss is : 0.05184999332269487
The training error is : 14.034999999999997
[False  True  True …  True  True  True]
The loss is : 0.05154133757765732
The training error is : 14.258333333333326
[ True  True  True …  True  True  True]
The loss is : 0.05128141586252163 4
The training error is : 13.909999999999997
[False  True  True …  True  True  True]
The loss is : 0.05099244090552137
The training error is : 14.12166666666667
[ True  True  True …  True  True  True]
The loss is : 0.05074286932371008
The training error is : 13.77000000000001
[False  True  True …  True  True  True]
The loss is : 0.05046884144247903
The training error is : 13.985
[ True  True  True …  True  True  True]
The loss is : 0.0502296477173934
The training error is : 13.674999999999997
[False  True  True …  True  True  True]
The loss is : 0.04997047046291217
The training error is : 13.838333333333324
[ True  True  True …  True  True  True]
The loss is : 0.04974005108415713
The training error is : 13.560000000000002
[False  True  True …  True  True  True]
The loss is : 0.04949497121686797
The training error is : 13.671666666666667
[ True  True  True …  True  True  True]
The loss is : 0.04927326109189782
The training error is : 13.456666666666663
```

```
[False  True  True … True  True  True]
The loss is : 0.04904097189509447
The training error is : 13.59833333333333
[ True  True  True … True  True  True]
The loss is : 0.04882795128776544
The training error is : 13.333333333333329
[False  True  True … True  True  True]
The loss is : 0.04860665962539456
The training error is : 13.521666666666661
[ True  True  True … True  True  True]
The loss is : 0.04840240419511806
The training error is : 13.239999999999995
[False  True  True … True  True  True]
The loss is : 0.04819137105557332
The training error is : 13.40166666666667
[ True  True  True … True  True  True]
The loss is : 0.04799487902771333
The training error is : 13.163333333333341
[False  True  True … True  True  True]
The loss is : 0.0477940030431145
The training error is : 13.303333333333327
[ True  True  True … True  True  True]
The loss is : 0.04760496071251265
The training error is : 13.053333333333327
[False  True  True … True  True  True]
The loss is : 0.04741332033484262
The training error is : 13.189999999999998
```

[118]:
```
print ("Answer 18 Contd")
print ("As we can see, the network has achieved training error of 13.19% after␣
 ↪100 iterations")
```

```
Answer 18 Contd
As we can see, the network has achieved training error of 13.19% after 100
iterations
```

1.17 Question 19

[120]:
```
print ("Answer 19")

xtest, ltest = MNISTtools.load(dataset = "testing", path = "/datasets/MNIST")
xtest = normalize_MNIST_images(xtest)
dtest = label2onehot(ltest)

print("The shape of xtest is:",xtest.shape)
print("The shape of ltest is:",ltest.shape)
```

```
print("The size of the testing dataset is : 10000")

y = forwardprop_shallow(xtest,nettrain)#testing on test data
loss = eval_loss(y,dtest)
testing_error = eval_perfs(y,ltest)

print("The loss is : ",loss)
print ("The testing error is: ",training_error)
```

```
Answer 19
The shape of xtest is: (784, 10000)
The shape of ltest is: (10000,)
The size of the testing dataset is : 10000
[ True  True  True …  True  True  True]
The loss is :  0.09060106537555575
The testing error is:  87.39
```

1.18 Question 20

[124]:
```
print ("Answer 20")

def backprop_minibatch_shallow(x, d, net, T, B=100, gamma=.05):#minibatch␣
 ↪gradient descent method
    N = x.shape[1]
    lbl = onehot2label(d)
    NB =int((N+B-1)/B)

    for t in range(T):
        for l in range(NB):
            idx = np.arange(B*l, min(B*(l+1), N))
            net = update_shallow(x[:,idx],d[:,idx],net,gamma)
        y = forwardprop_shallow(x, net)
        loss = eval_loss(y,d)
        training_error = eval_perfs(y,lbl)
        print(loss)
        print(training_error)
    return net
```

```
Answer 20
```

1.19 Question 21

[125]:
```
print ("Answer 21")

netminibatch = backprop_minibatch_shallow(xtrain, dtrain, netinit, 5, B=100)
y = forwardprop_shallow(xtest,netminibatch)
loss = eval_loss(y,dtest)
testing_error = eval_perfs(y,ltest)

print("The loss is :",loss)
print ("The testing error is :", testing_error)
```

```
Answer 21
[ True  True  True …  True  True  True]
0.00638118087250069
2.001666666666665
[ True  True  True …  True  True  True]
0.00620677356344962
1.9533333333333331
[ True  True  True …  True  True  True]
0.0059672864272528525
1.88333333333334
[ True  True  True …  True  True  True]
0.005843186752013941
1.836666666666659
[ True  True  True …  True  True  True]
0.005717637543621029
1.806666666666672
[ True  True  True …  True  True  True]
The loss is : 0.027083609668869593
The testing error is : 84.75166666666667
```