Intro to Deep Learning
Homework Assignment 1
Shounak Rangwala
Netid: snr85

1)As written in the comments of the hw1-q1.py file, we can get the array of distances of the test point from the 12 training data points.

```
shounakrangwala@nbp-212-1
[-1  1 -1]
[[1.73205081]
 [1.41421356]
 [2.        ]
 [2.23606798]
 [2.23606798]
 [2.44948974]
 [2.82842712]
 [3.        ]
 [3.        ]
 [3.31662479]
 [1.41421356]
 [2.82842712]]
1
shounakrangwala@nbp-212-1
```

For k=1:
We need to find the distance that is the least from the `dist` array. As we can see from the figure above, the least value is 1.4141 which is in the index 1 and 10 of the array. This index corresponds to class A or class C. So, the answer will be either A or C.

For k=2:
We need to find the 2 smallest distances in the `dist` array. These are again 1.414 in the index 1 and index 10 positions. However, since we have to make the classification decision based on 2 distances, in this case we cannot make a decision and the point will be in a position where no decision can be made (blank white space where it is equidistant from 2 points).

For k=3:
We need to find the 3 smallest distances in the `dist` array. In this case these are 1.414 in the index 1 and 10 position and 1.732 in the index 0 position. Since we are making the classification decision based on the 3 distances we can see that there

are 2 distances where its closest to class A and just one where it is close to class C. Thus taking the majority of the 3 distances we can classify the test point to be A.

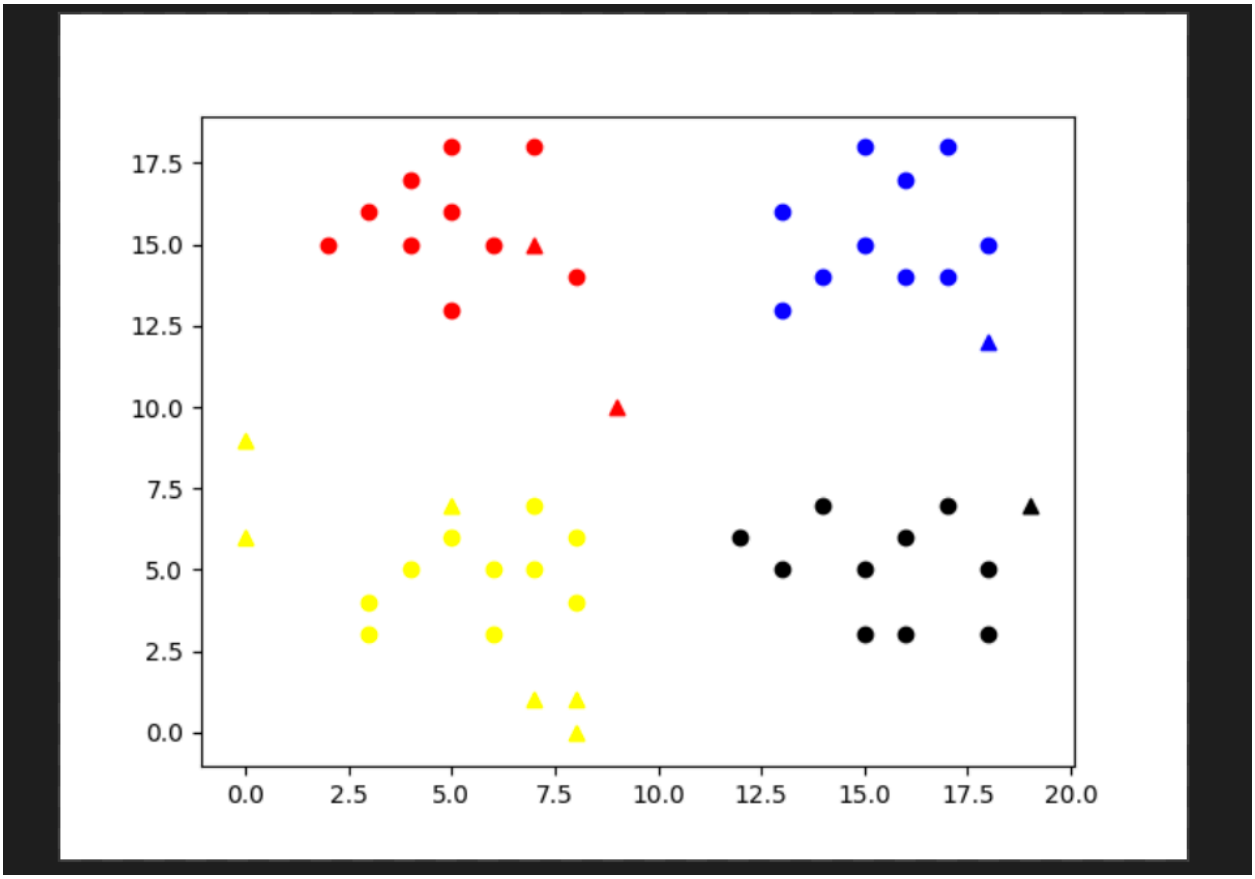
2)
The source code for this is as follows:

```
def kNNClassify(newInput, dataSet, labels, k):
    result=[]
    ########################
    # Input your code here #
    ########################
    dist=np.zeros((10,40))
    for i in range(len(newInput)):
        for j in range(len(dataSet)):
            d= np.linalg.norm(dataSet[j]-newInput[i])
            # as done before, we are making a 10x40 matrix of the distances between th
e test points and the training data.
            # Every row corresponds to 1 test data point and the 40 entries inside it
are its distances from the 40 training points.
            dist[i,j] = d



    # below are making an array called votes. THis array will have as many elements as
 there are classification classes.
    # In this case the classes are 4 in number

    for i in range(len(newInput)):
        votes = np.zeros(4)
        x= np.argsort(dist[i])[:k]
        # selects the k smallest values and returns an array with the indices of these
 values in the original unsorted array
        # print(x)
        for i in range(len(x)):
            num_label = labels[x[i]]
            votes[num_label]+=1
        print(votes)
        result.append(np.argmax(votes))
        # selects the index with the max value and returns it and appends it to the re
sult array.
    ####################
    # End of your code #
    ####################
    return result
```

The output for k = 6 in my case is as follows:

```
shounakrangwala@nbp-212-162 deepLearning HW1 % /usr/local/bin/pytho
[0, 2, 3, 2, 2, 2, 0, 2, 1, 2]
random test points are: [[ 7 15]
 [ 0  6]
 [19  7]
 [ 8  0]
 [ 7  1]
 [ 8  1]
 [ 9 10]
 [ 0  9]
 [18 12]
 [ 5  7]]
knn classfied labels for test: [0, 2, 3, 2, 2, 2, 0, 2, 1, 2]
shounakrangwala@nbp-212-162 deepLearning HW1 %
```

The image file is as follows:



However, I will still attach the source code so you can run it and confirm the results.

3)
The source code for this is as follows:

```
def kNNClassify(newInput, dataSet, labels, k):
    result=[]
    ########################
    # Input your code here #
    ########################
    test_len = len(newInput)
    train_len= len(dataSet)

    dist=np.zeros((test_len,train_len))
    for i in range(test_len):
        for j in range(train_len):
            d= np.linalg.norm(dataSet[j]-newInput[i])
            dist[i,j] = d

    # print(dist)

    # print(labels)
    for i in range(test_len):
        votes = np.zeros(10)
        x= np.argsort(dist[i])[:k]
        # print(x)
        for i in range(len(x)):
            num_label = labels[x[i]]
            # print(num_label)
            votes[num_label]+=1
        print(votes)
        result.append(np.argmax(votes))

    ####################
    # End of your code #
    ####################
    return result
```

In this case, we have taken votes as a 10 element array because the y_train label set ranged from 0-9 and thus contained 10 discreet labels.

I ran the code for the first 20 samples in the test data and kept k = 12 for the knn classification. The output I received was as follows (the 20 arrays here are the votes array I make for each test point):

```
51    outputlabels=kNNClassify(x_test[0:20],x_train,y_train,12)
52    print(outputlabels)
53    result = y_test[0:20] - outputlabels
54    result = (1 - np.count_nonzero(result)/len(outputlabels))
55    print ("---classification accuracy for knn on mnist: %s ---" %result)
56    print ("---execution time: %s seconds ---" % (time.time() - start_time))
57
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
%
shounakrangwala@nbp-212-162 deepLearning HW1 % /usr/local/bin/python3 "/Users/shouna
[ 0.  0.  0.  0.  0.  0.  0. 12.  0.  0.]
[ 0.  0. 12.  0.  0.  0.  0.  0.  0.  0.]
[ 0. 12.  0.  0.  0.  0.  0.  0.  0.  0.]
[12.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
[ 0.  0.  0.  0. 11.  0.  0.  0.  0.  1.]
[ 0. 12.  0.  0.  0.  0.  0.  0.  0.  0.]
[ 0.  0.  0.  0. 11.  0.  0.  0.  0.  1.]
[ 0.  0.  0.  0.  0.  0.  0.  0.  0. 12.]
[0. 0. 0. 0. 0. 9. 1. 0. 0. 2.]
[ 0.  0.  0.  0.  0.  0.  0.  1.  0. 11.]
[12.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
[ 0.  0.  0.  0.  0.  0. 12.  0.  0.  0.]
[ 0.  0.  0.  0.  0.  0.  0.  0.  0. 12.]
[12.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
[ 0. 12.  0.  0.  0.  0.  0.  0.  0.  0.]
[ 0.  0.  0.  2.  0. 10.  0.  0.  0.  0.]
[ 0.  0.  0.  0.  0.  0.  0.  0.  0. 12.]
[ 0.  0.  0.  0.  0.  0.  0. 12.  0.  0.]
[ 0.  0.  0. 12.  0.  0.  0.  0.  0.  0.]
[ 0.  0.  0.  0. 12.  0.  0.  0.  0.  0.]
[7, 2, 1, 0, 4, 1, 4, 9, 5, 9, 0, 6, 9, 0, 1, 5, 9, 7, 3, 4]
---classification accuracy for knn on mnist: 1.0 ---
---execution time: 13.569308042526245 seconds ---
shounakrangwala@nbp-212-162 deepLearning HW1 %
```