Programming Assignment

1. **Nearest Neighbor.** In this question, we will study the performance of the Nearest Neighbor (NN) algorithm on the MNIST dataset. The MNIST dataset consists of images of handwritten digits, along with their labels. Each image has 28×28 pixels, where each pixel is in grayscale scale, and can get an integer value from 0 to 255. Each label is a digit between 0 and 9. The dataset has 70,000 images. Althought each image is square, we treat it as a vector of size 784.

   The MNIST dataset can be loaded with `sklearn` as follows:

```
>>> from sklearn.datasets import fetch_openml
>>> mnist = fetch_openml('mnist_784')
>>> data = mnist['data']
>>> labels = mnist['target']
```

   Loading the dataset might take a while when first run, but will be immediate later. See `http://scikit-learn.org/stable/datasets/mldata.html` for more details. Define the training and test set of images as follows:

```
>>> import numpy.random
>>> idx = numpy.random.RandomState(0).choice(70000, 11000)
>>> train = data[idx[:10000], :].astype(int)
>>> train_labels = labels[idx[:10000]]
>>> test = data[idx[10000:], :].astype(int)
>>> test_labels = labels[idx[10000:]]
```

   It is recommended to use `numpy` and `scipy` where possible for speed, especially in distance computations.

   The k-NN algorithm is the first (and most trivial) classification algorithm we encounter in the course. In order to classify a new data point, it finds the k nearest neighbors of that point and classifies according to the majority label. More details can be found on Wikipedia.

   (a) Write a function that accepts as input: (i) a set of images; (ii) a vector of labels, corresponding to the images (ii) a query image; and (iii) a number k. The function will implement the k-NN algorithm to return a prediction of the query image, given the given label set of images. The function will use the k nearest neighbors, using the Euclidean L2 metric. In case of a tie between the k labels of neighbors, it will choose an arbitrary option.

   (b) Run the algorithm using the first $n = 1000$ training images, on each of the test images, using $k = 10$. What is the accuracy of the prediction (measured by 0-1 loss; i.e. the percentage of correct classifications)? What would you expect from a completely random predictor?

   (c) Plot the prediction accuracy as a function of k, for $k = 1, \ldots, 100$ and $n = 1000$. Discuss the results. What is the best k?

   (d) Using $k = 1$, run the algorithm on an increasing number of training images. Plot the prediction accuracy as a function of $n = 100, 200, \ldots, 5000$. Discuss the results.