Assignment 1

Due: Feb. 7

Submission Instructions

- **Create a README file, with simple, clear instructions on how to compile and run your code.** *If the TA cannot run your program by following the instructions, you will receive 50% of programing score.*
- **Zip all your files (code, README, written answers, etc.) in a zip file named** $\{firstname\}\_\{lastname\}\_CS5790\_HW1.zip$ **and upload it to Blackboard**

In this assignment you will be implementing a straightforward classification algorithm known as the k-nearest neighbor (k-NN) classifier. Your implementation should accept two data files as input (both of which are posted in Blackboard together with this assignment):

1. A **training** dataset of class-labeled feature vectors
2. A **test** dataset of unlabeled feature vectors

For simplicity, you do NOT need to normalize the data for this exercise.

The training data contains examples of three different types of Iris (a type of flower), with each example having a class-label of either ***versicolor***, ***virginica*** or ***setosa***. Each example has four (numeric) features: Sepal Length, Sepal Width, Petal Length and Petal Width. Your classifier must examine each unlabeled example in the **test** set and classify it as one of the three classes of Iris. The classification will be based on an *unweighted* vote of its k nearest examples in the **training** set.

To determine nearest neighbors, you should measure all distances using regular Euclidean distance:

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_i (x_i - y_i)^2 \right)^{\frac{1}{2}}$$

If two or more classes receive the same (winning) number of votes, break the tie by choosing the class with the lowest total distance from the test point to its voting examples. For example, if class X has two votes (at distances 1 and 3) and class Y has two votes (at distances 2 and 4), then your algorithm should predict class X since (1+3)<(2+4).

Program Output

The output from your program must be in the same format as the **test** set file, except each line must have the k-NN predicted labels appended to it for $k = 1, 3, 5, 7, 9$ (in that order). For example, if the k-NN classifier predicts that example $\mathbf{x_1} = (6.7, 3.1, 4.4, 1.4)$ is of class ***setosa*** when $k = 1$, $k = 3$ and $k = 5$ but predicts class ***versicolor*** when $k = 7$ and $k = 9$ then your output line for that example should be:

```
6.7,3.1,4.4,1.4,x1,setosa,setosa,setosa,versicolor,versicolor
```

There is a likelihood that you will develop this classifier further in future assignments. To be able to easily make classifications with various different values of k, your core k-NN classification subprocedure/function/method should therefore take k as one of its parameters. This should help to reduce development effort not only for this assignment, but also for potential future assignments.