Data Mining Project 1

Group:

Akshya Srinivasa Raghavan	1001547268
Balaji Paiyur Mohan	1001576836

Project File Name: project1.py

Execution Steps: python project1.py

Place all the files in the same directory as your code

Asks the user to enter the task

a/A	Task A
b/B	Task B
c/C	Task C
d/D	Task D

Explanation of the code

FUNCTION NAME	**USAGE**
taskA	Implements task A of the given question. It classifies the given input string using SVM, Centroid method, Linear and KNN method
taskB	Implements task B of the given question. It makes use of 5-fold cross validation to classify. It finds the accuracy and also displays the single accuracy for each classifier
taskC	Implements task C of the given question. Runs the centroid classifier to compute average test image classification accuracy. Plot these 7-average accuracies on one curve in a figure
pickDataClass(filename, class_ids)	This function takes the input file and the class ids of the given input string Now, it picks the data from the input file corresponding to the class ids of the input string This data is stored in tempfile.out Given: filename: char_string specifing the data file to read. For example, 'ATNT_face_image.txt' class_ids: array that contains the classes to be pick. For example: (3, 5, 8, 9) Returns: an multi-dimension array or a file, containing the data (both attribute vectors and class labels) of the selected classes We use this subroutine to pick a small part of the data to do experiments. For example for handwrittenletter data, we can pick classes "C" and "F" for a 2-class experiment. Or we pick "A,B,C,D,E" for a 5-class experiment.

	test_instances: the data instances in each class to be used as test data. We assume that the remaining data instances in each class (after the test data instances are taken out) will need training_instances
letter_2_digit_convert(input_string)	This function takes the input string and assigns class id to it. The letter is first converted into its corresponding ASCII value. Now, 64 is subtracted from it. Therefore A is now 1, B is 2 and so on These values that are now classids are appended to a list These values are then passed to pickDataClass function Given: "letter_2_digit_convert" that converts a character string to an integer array. For example,letter_2_digit_convert('ACFG') returns array (1, 3, 6, 7).
splitData2TestTrain(filename, number_per_class, test_instances)	This function is used to divide the data into train and test data The input string and the corresponding data which is stored in tempfile.out is passed as an argument It first reads this file , the test-instances are of the form 20:29 This means the data from 20 to 29 are the test data Now the remaining data i.e; 0-19 and 29-39 are training data These values are stored in test_data and train_data test_data[0] contains the classids of test_data train_data[0] contains the classids of train_data test_data[1:] contains the actual test data and train_data[1:] contains the train data These values are passed to store function to store the data in testData.txt and trainData.txt Given: splitData2TestTrain(filename, number_per_class, test_instances) filename: char_string specifing the data file to read. This can also be an array containing input data. number_per_class: number of data instances in each class (we assume every class has the same number of data instances) test_instances: the data instances in each class to be used as test data. We assume that the remaining data instances in each class (after the test data instances are taken out) will be training_instances Return/output: Training_attributeVector(trainX), Training_labels(trainY), Test_attributeVectors(testX), Test_labels(testY) The data should easily feed into a classifier. Example: splitData2TestTrain('Handwrittenletters.txt', 39, 1:20) Use entire 26-class handwrittenletters data. Each class has 39 instances. In every class, first 20 images for testing, remaining 19 images for training
store(class_ids, filedata, fileName)	This function is called from splitData2testntrain function The class ids and the data of test and train are passed

	It takes these data and stores them in testData.txt and trainData.txt respectively Given: This routine will store (trainX,trainY) into a training data file, and store (testX,testY) into a test data file. The format of these files is determined by student's choice: could be a matlab file, a text file, or a file convenient for Pythorn. These file should allow the data to be easily read and feed into a classifier.
svmClassifier(train_data, trainLabel, test_data, testLabel):	Implementing svm using scikit library
centroid(trainVector, trainLabel, testVector, testLabel)	Implementation of centroid method using Euclidian distance
linear(Xtrain, Xtest, Ytrain, Ytest)	Implementation of linear method using Euclidian distance
kNearestNeighbor(trainvector, trainlabel, testvector, k)	This function is used to implement k nearest neighbour using euclidean distance Find the least k out of the most repeated k

Task A:

Question:

Use the data-handler to select "A,B,C,D,E" classes from the hand-written-letter data.
From this smaller dataset, Generate a training and test data: for each class
using the first 30 images for training and the remaining 9 images for test.
Do classification on the generated data using the four classifers.

Output:

```
Final Accuracy for Task A:

SVM : 75.555556

Centroid : 66.666667

Linear : 75.555556

KNN (K=5): 80.000000
```

Task B:

Question:
On ATNT data, run 5-fold cross-validation (CV) using each of the

four classifiers: KNN, centroid, Linear Regression and SVM.

If you don't know how to partition the data for CV, you can use the data-handler to do that.
Report the classification accuracy on each classifier.
Remember, each of the 5-fold CV gives one accuracy. You need to present all 5 accuracy numbers for each classifier. Also, the average of these 5 accuracy numbers.

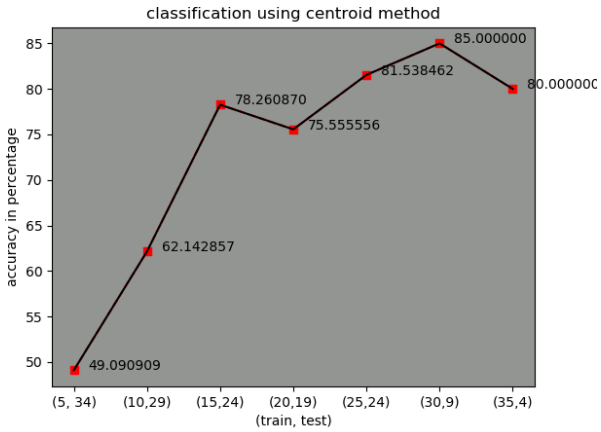
Output:

```
K-NN (k=5) : 94.500000
[90.0, 97.5, 95.0, 92.5, 97.5]

Linear : 88.000000
[80.0, 87.5, 90.0, 87.5, 95.0]

SVM : 71.500000
[65.0, 67.5, 65.0, 80.0, 80.0]

Centroid : 64.500000
[65.0, 65.0, 67.5, 60.0, 65.0]
```

Task C:

On handwritten letter data, fix on 10 classes. Use the data handler to generate training and test data files.
Do this for seven different splits: (train=5 test=34), (train=10 test=29), (train=15 test=24) , (train=20 test=19), (train=25 test=24) , (train=30 test=9) , (train=35 test=4).
On these seven different cases, run the centroid classifier to compute average test image classification accuracy. Plot these 7 average accuracy on one curve in a figure. What trend can you observe?
When do this task, the training data and test data do not need be written into files.

Output:



Trend:

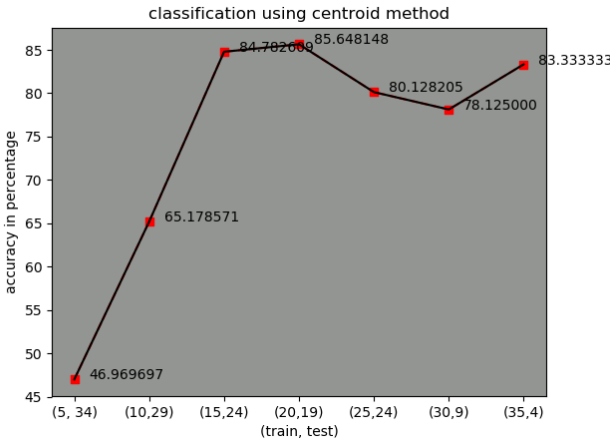
We see that when the train data is 5 and test is 34, the accuracy achieved is the least. This accuracy increases but drops when the test and train are almost equally distributed. It again increases at (30,9) and falls when the test is too small. This shows that as more data is trained, the accuracy increases.

Task D:

Question:

Repeat task (D) for another different 10 classes. You get another 7 average accuracy.
Plot them on one curve in the same figure as in task (D). Do you see some trend?

Output:



Trend:

We see that when the train data is 5 and test is 34, the accuracy achieved is the least. This accuracy increases but drops when the test and train are almost equally distributed. It again increases at (30,9) and falls when the test is too small. This shows that as more data is trained, the accuracy increases.