Dhanesh Shah
1001554982

Parth Joshi
1001556783

CSE5334 – DATA MINING : Project

By

Dhanesh Shah(1001554982)

Parth Joshi(1001556783)

Task A:

In this task we are using **HandWrittenLetters.txt** data, we are performing Classification for 4 classifiers – KNN, Centroid, SVM and Logistic Regression.

Observation:

Average Accuracies (%)	KNN	Centroid	SVM	Logistic Regression
	91.11	86.67	91.11	93.33

*Table: 1.1 Average Accuracies for KNN, Centroid, SVM and Logistic Regression for 2,3,5,10-Fold CV for **HandWrittenLetters**.txt*

Observations:

1. Accuracy for Logistic Regression classifier is always highest than SVM, KNN and Centroid. As you can see the accuracy for Logistic Regression classifier is highest and it always falls under range 92-100% (Refer Figure 1.1)

Task B:

In this task we are using **HandWrittenLetters.txt** data, we are performing 5-fold Cross Validation for 4 classifiers – KNN, Centroid, SVM and Logistic Regression.

Observation:

Average Accuracies (%)	KNN	Centroid	SVM	Logistic Regression
1-Fold	78.75	87.5	100.00	97.50
2-Fold	75.00	90.0	98.95	95.00
3-Fold	78.05	98.75	98.50	96.00
4-Fold	75.0	93.75	93.75	98.50
5-Fold	68.75	93.75	90.75	95.75

Table: 1.2 Average Accuracies for KNN, Centroid, SVM and Logistic Regression *for 5-Fold CV for HandWrittenLetters.txt*

Observations :

1. Accuracy for KNN classifier is lowest in this case than SVM, Logistic Regression and Centroid.
   As you can see from the figure, average accuracies for KNN classifier is lowest and it always falls under range 67-80% (Refer Figure 1.2)
2. Accuracy with 2-Fold cross validation is always low.
3. KNN has lowest accuracy.

Task C: In this task we fix 10 classes. We use the data handler to generate training and test data files. We have split the data in 7 parts and named each one as split-1, split-2 and so on. The classes that we fix are based on the input string.

After running multiple iterations, we see a trend that the accuracy is highest when the number of training and testing dataset is almost equal. For example, in the output the split-5 has training dataset = 25 and testing dataset = 24.

Output:

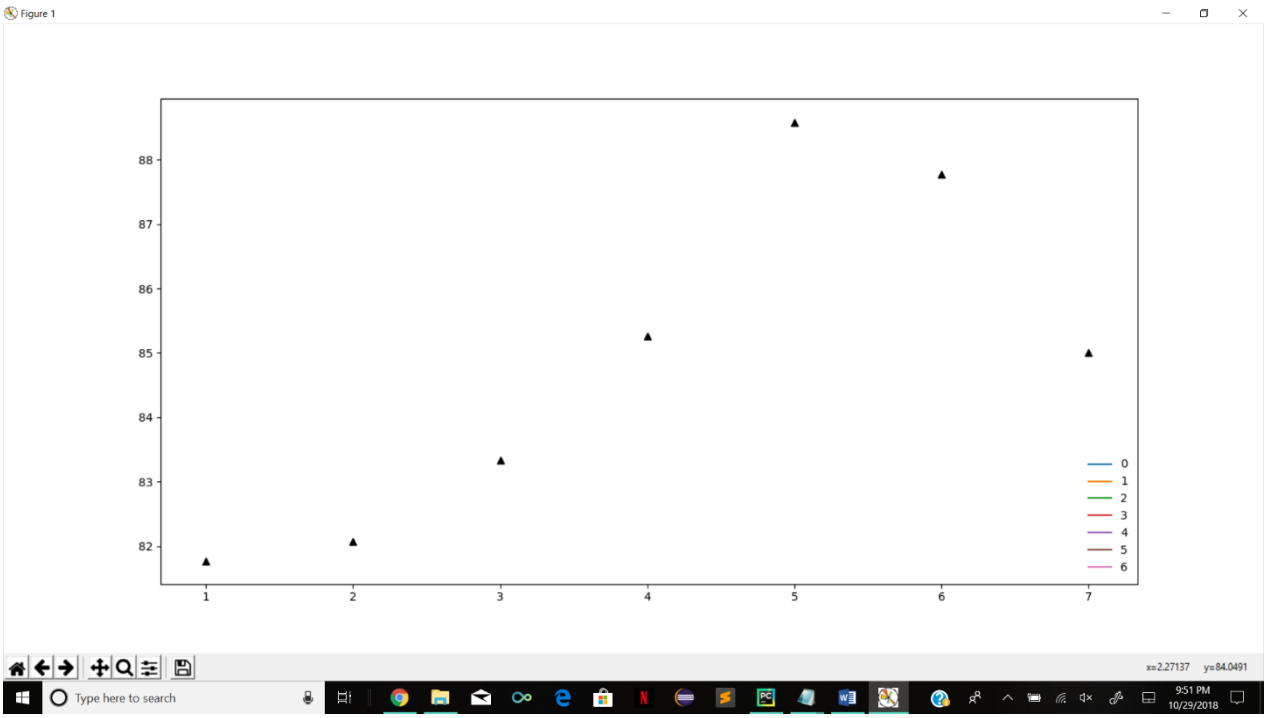

Figure1.3: Result of 7 accuracies of different train and test data splits on 10 different classes for HandWrittenData.txt

1–**Split1** 2- **Split2** 3- **Split3** 4- **Split4** 5- **Split5** 6- **Split6** 7- **Split7**

Task D: This task is similar to Task C. We use the data handler to generate training and test data files. We have split the data in 7 parts and named each one as split-1, split-2 and so on. The input string that we use to fix classes is different from Task C.

After running multiple iterations, we conclude that the accuracy is stable and highest when the number of training and testing dataset is almost equal. For example, in the output Split-4 and Split-5 readings are stable and highest as the Training and Testing dataset is almost equal.

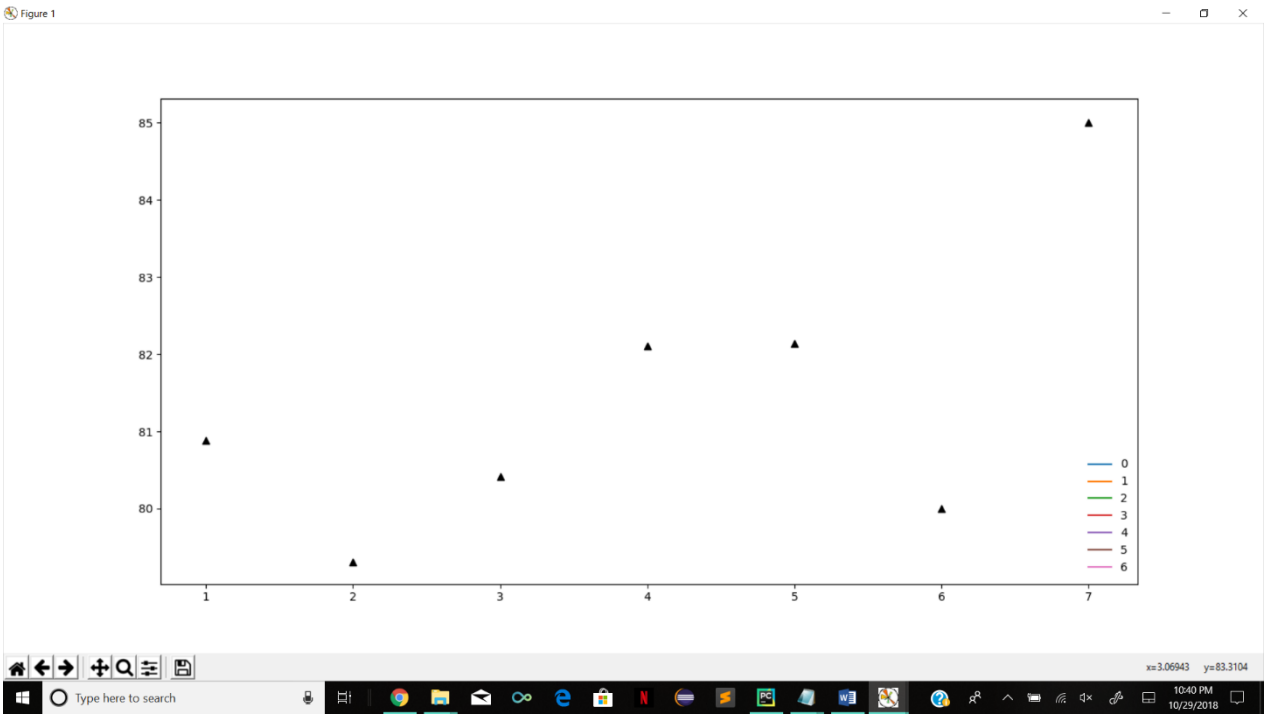

Figure1.4: Result of 7 accuracies of different train and test data splits on 10 different classes for HandWrittenData.txt

1–**Split1** 2- **Split2** 3- **Split3** 4- **Split4** 5- **Split5** 6- **Split6** 7- **Split7**

Task E:

In this task, we have made a Data_Handler. In this task we have defined 3 subroutines working on HandWrittenLetters.txt

1. pickData() :

   Parameters: filename, class

   Description: In this subroutine, we are generating train and test data. The train and test data are generated for the specified array of class numbers with respect to the number of training instances and test instances on the given file.

2. storeData():

   Parameters: train_X, train_Y, test_X, test_Y

Description: In this subroutine, we collected the train and test data in to different files namely, trainData.txt and testData.txt respectively. We can use these file and feed into classifier.

3. letter_2_digit_convert()

   Parameters: A string of characters

   Description: In this subroutine, we convert the characters from the string into their respective class values and return as an array.