ASSIGNMENT-4

"Classification Accuracy on MNIST handwritten digit data by NBC with and without feature extraction (using mutual information)"

Submitted By

Sourabh Kumar

Stud ID: 15/8

Course Name- CPI 015 Pattern Recognition- Spring 2020

Under the guidance of:

Prof. MN Murthy

Department of Computer Science and Automation

Indian Institute of Science, Bangalore, India

Problem Statement

1. Download MNIST handwritten digit data. There are 10 classes (corresponding to digits 0, 1, ..., 9) and each digit is viewed as an imgae of size 28×28 (= 784) pixels; each pixel having values 0 to 255. There are around 6000-digit training patterns and around 1000 test patterns in each class and the class label is also provided for each of the digits. Visit http://yann.lecun.com/exdb/mnist/ for more details.

2. Run naïve Bayes classifier (NBC) based on the following:

(a) Consider classes 0 (digit zero) and 1 (digit one). Convert each of the patterns (both training and test) to binary images/strings by replacing each pixel value with a 0 or a 1. This conversion is done by using 0 if the original value is in the range [0,127] and by using the value 1 otherwise (that is use 1 if the pixel value is in the range [128,255]). Compute the accuracy of NBC on the binary data. (b) Repeat the experiment in step 1 with the pair of classes 7 and 9. (c) Use mutual information to extract the best 80 features (out of 784 (28×28)) in each of the above cases and compute accuracy on the test dataset using NBC.

3. Report your results appropriately using tables and graphs for different scenarios.

4. The report must be brief giving a page on the resources used and how they are used. Two-three pages on the results of your experiments.

Technology and Programming Resources Used

- Spyder Programming Editor
- Python Programming Language 3.7
- Following popular sklearn python libraries for machine learning
    a. sklearn.datasets for fetching MNIST data (fetches data internally from the source web site- http://yann.lecun.com/exdb/mnist/)
    b. sklearn. GaussianNB for Gaussian Naïve Bayes classifier
    c. sklearn.preprocessing for binarizing the data based on below logic
        1. range [0,127] – Binary value 0
        2. range [128,255] – Binary value 1
    d. sklearn.feature_selection for extracting best 80 features using mutual information method
    e. matplotlib.pyplot library for plotting charts
- MNIST hand written digit data with
    a. Total Features -784 (pixel grid size- 28x28)
    b. Total Classes- 10 (Digit 0 to Digit 9)
    c. Total Training data- 60000 (6000 per class)
    d. Total Test data- 10000 (1000 per class)
    e. Two pairs of Class data used for experiments- “Class 0 & Class 1” and “Class 7 & Class 9” together

Dataset Pre-Processing: -

Data set – Class 0 & Class 1 data

Training data - First 12,000 records (starting from 0 to 11,999 row indices) and all 784 feature fields

Training target- First 12,000 records (starting from 0 to 11,999 row indices) and last785th target field.

Test data – First 2,000 records (starting from 60,000 to 61,999 row indices) and all 784 feature fields

Test target - First 2,000 records (starting from 60,000 to 61,999 row indices) and last 785th target field

Data set – Class 7 & Class 9 data

Training data - 6,000 records (starting from 42,000 to 47,999 row indices) and

6,000 records (starting from 54,000 to 59,999 row indices) for all 784 feature fields

Training target- 6,000 records (starting from 42,000 to 47,999 row indices) and 6,000 records (starting from 54,000 to 59,999 row indices) for last785th target field.

Test data – 1,000 records (starting from 67,000 to 67,999 row indices) and

1,000 records (starting from 69,000 to 69,999 row indices) and all 784 feature fields

Test target - 1,000 records (starting from 67,000 to 67,999 row indices) and 1,000 records (starting from 69,000 to 69,999 row indices) for last 785th target field

Mutual Information process for selecting 80 or more features: -

```
# Calculating multual information and returns the value in an signle dimentsional array
mi_features_0_1 = mutual_info_classif(mnist_training_data_0_1, mnist_training_target_0_1,random_state=0)
mi_features_7_9 = mutual_info_classif(mnist_training_data_7_9, mnist_training_target_7_9,random_state=0)

top_80_features_based_0_1 = mi_features_0_1.argsort()[::-1][:80]
top_80_features_based_7_9 = mi_features_7_9.argsort()[::-1][:80]

total_indices = range(len(top_80_features_based_0_1))

# Making blank training data set and test data set for class pair- (0 & 1)
new_mnist_training_data_0_1 = np.array(np.empty((12000,))).reshape(-1,1)
new_mnist_test_data_0_1 = np.array(np.empty((2000,))).reshape(-1,1)

# Making blank training data set and test data set for class pair- (7 & 9)
new_mnist_training_data_7_9 = np.array(np.empty((12000,))).reshape(-1,1)
new_mnist_test_data_7_9 = np.array(np.empty((2000,))).reshape(-1,1)
```

Experiment-1

Check Classification Accuracy for first pair (Class 0 & Class 1) and second pair (Class 7 & Class 9) respectively using NBC (soft and hard) with all 784 features used

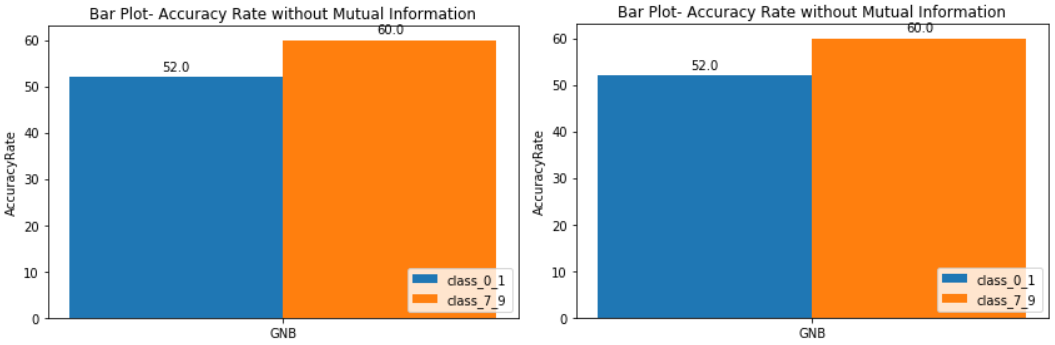
Program Name- Assignment4_without MI.py

Data set – (Class 0 & Class 1) data & (Class 7 & Class 9)

Result Table:

Dataset	Algorithms	Accuracy %
Class 0 & 1	NBC	52.0
Class 7 & 9	NBC	60.0

Plots:



Experiment-2

Check Classification Accuracy for first pair (Class 0 & Class 1) and second pair (Class 7 & Class 9) respectively using NBC (soft and hard) with all 80 features used (after mutual information feature extraction pre-processing)

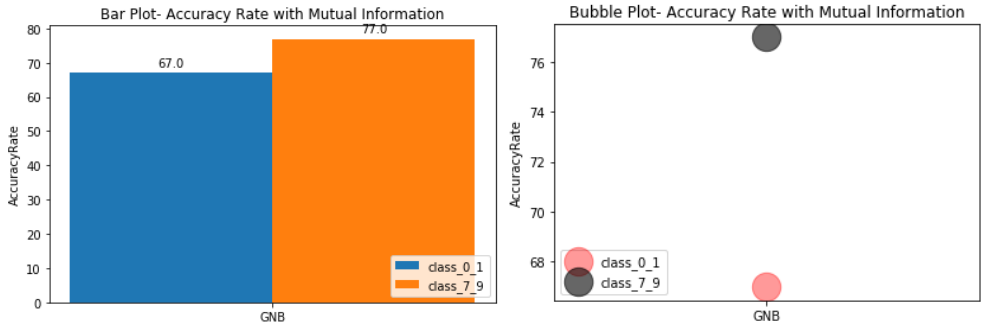
Program Name- Assignment4_with MI.py

Data set – (Class 0 & Class 1) data & (Class 7 & Class 9)

Result Table:

Dataset	Algorithms	Accuracy %
Class 0 & 1	NBC	67.0
Class 7 & 9	NBC	77.0

Plots:



Experiment-3

Check Classification Accuracy for first pair (Class 0 & Class 1) and second pair (Class 7 & Class 9) respectively using NBC (soft and hard) with all 160 features used (after mutual information feature extraction pre-processing)

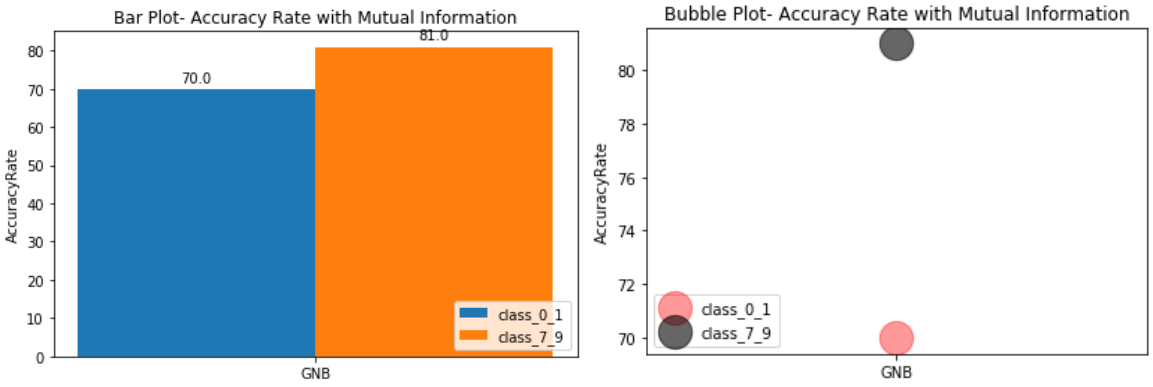
Program Name- Assignment4_with MI.py

Data set – (Class 0 & Class 1) data & (Class 7 & Class 9)

Result Table:

Dataset	Algorithms	Accuracy %
Class 0 & 1	NBC	70.0
Class 7 & 9	NBC	81.0

Plots:



Experiment-4

Check Classification Accuracy for first pair (Class 0 & Class 1) and second pair (Class 7 & Class 9) respectively using NBC (soft and hard) with all 240 features used (after mutual information feature extraction pre-processing)

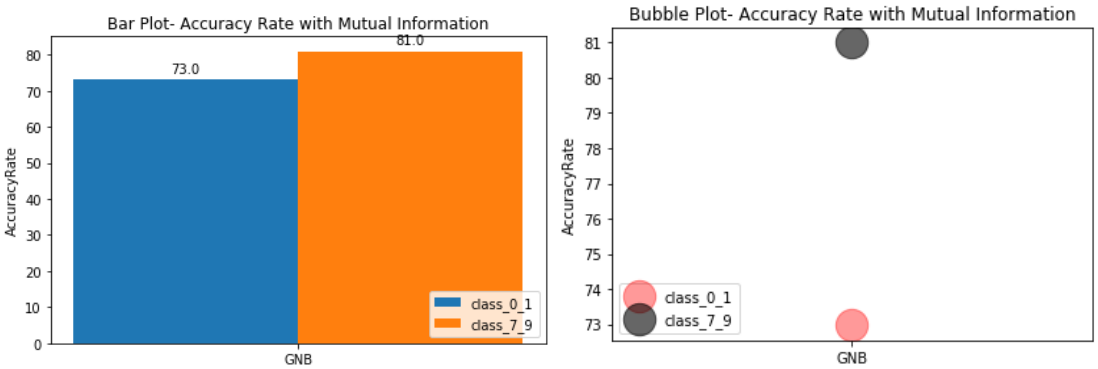
Program Name- Assignment4_with MI.py

Data set – (Class 0 & Class 1) data & (Class 7 & Class 9)

Result Table:

Dataset	Algorithms	Accuracy %
Class 0 & 1	NBC	73.0
Class 7 & 9	NBC	81.0

Plots:



Experiment-5

Check Classification Accuracy for first pair (Class 0 & Class 1) and second pair (Class 7 & Class 9) respectively using NBC (soft and hard) with all 320 features used (after mutual information feature extraction pre-processing)

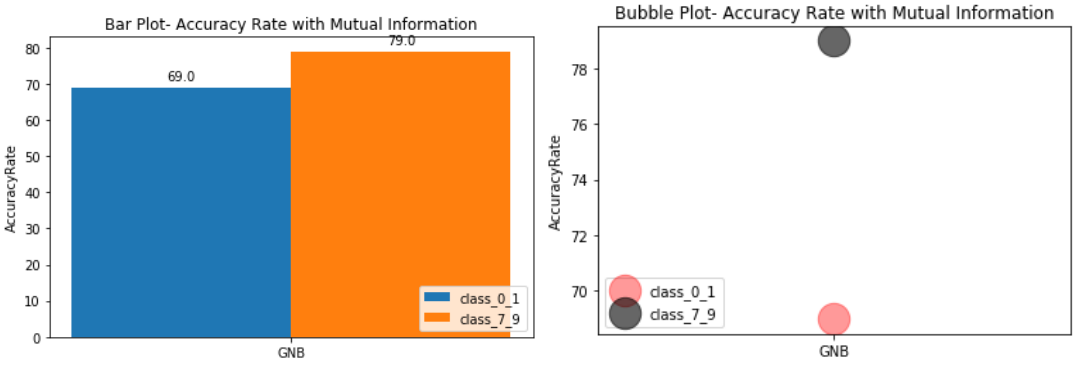
Program Name- Assignment4_with MI.py

Data set – (Class 0 & Class 1) data & (Class 7 & Class 9)

Result Table:

Dataset	Algorithms	Accuracy %
Class 0 & 1	NBC	69.0
Class 7 & 9	NBC	79.0

Plots:



Experiment-6

Check Classification Accuracy for first pair (Class 0 & Class 1) and second pair (Class 7 & Class 9) respectively using NBC (soft and hard) with all 280 features used (after mutual information feature extraction pre-processing)

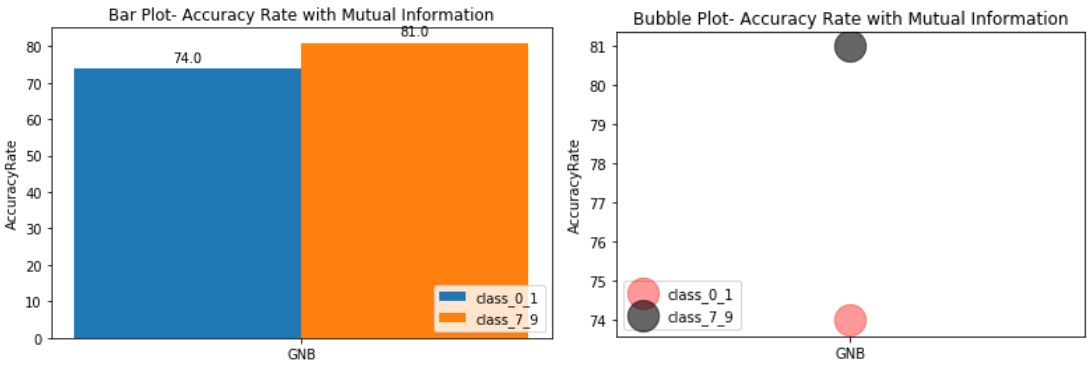
Program Name- Assignment4_with MI.py

Data set – (Class 0 & Class 1) data & (Class 7 & Class 9)

Result Table:

Dataset	Algorithms	Accuracy %
Class 0 & 1	NBC	74.0
Class 7 & 9	NBC	81.0

Plots:



Experiment-7

Check Classification Accuracy for first pair (Class 0 & Class 1) and second pair (Class 7 & Class 9) respectively using NBC (soft and hard) with all 300 features used (after mutual information feature extraction pre-processing)

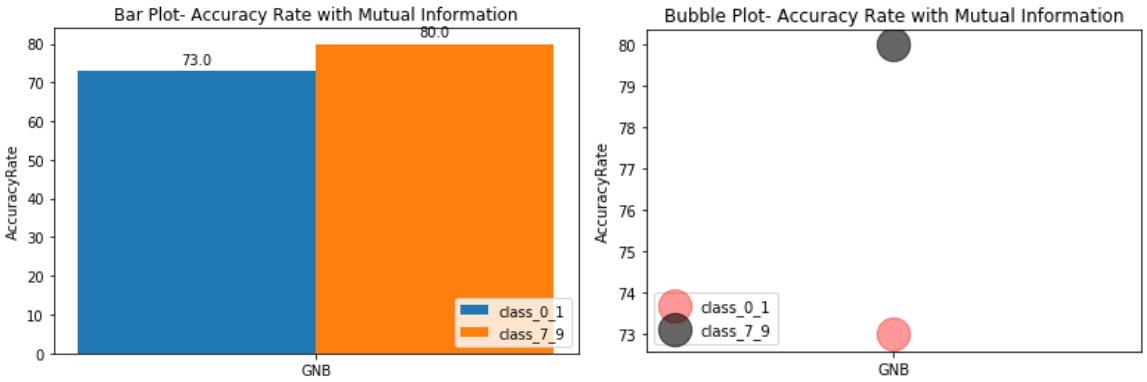
Program Name- Assignment4_with MI.py

Data set – (Class 0 & Class 1) data & (Class 7 & Class 9)

Result Table:

Dataset	Algorithms	Accuracy %
Class 0 & 1	NBC	73.0
Class 7 & 9	NBC	80.0

Plots:



Experiment-8

Check Classification Accuracy for first pair (Class 0 & Class 1) and second pair (Class 7 & Class 9) respectively using NBC (soft and hard) with all 290 features used (after mutual information feature extraction pre-processing)

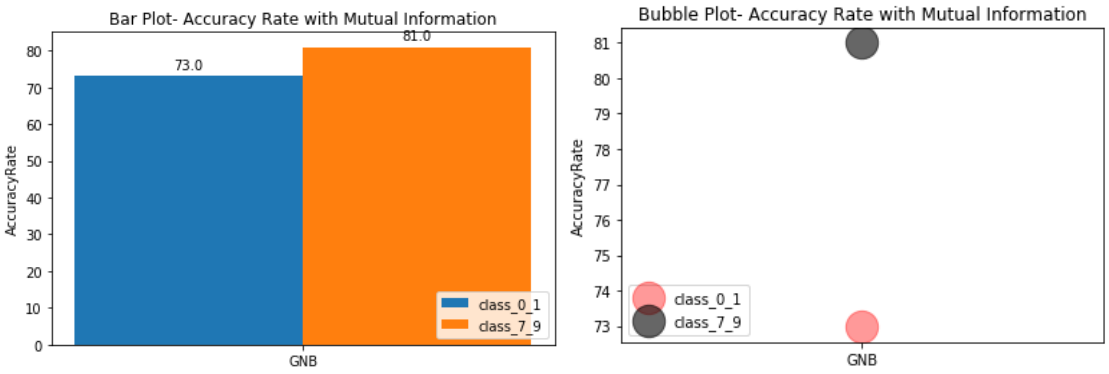
Program Name- Assignment4_with MI.py

Data set – (Class 0 & Class 1) data & (Class 7 & Class 9)

Result Table:

Dataset	Algorithms	Accuracy %
Class 0 & 1	NBC	73.0
Class 7 & 9	NBC	81.0

Plots:



Final Conclusion (combined for all experiments based on the separate results tables and plots):

1. Experiment 1 (with 784 features) accuracies achieved are **52 %** and **60 %** for class pair 0 & 1 and class pair 7 & 9 respectively.
2. With other experiments done with mutual information feature extraction data pre-processing step. 80, 160, 240, 280, 290, 300, 320 mutual information features were selected in different experiments from the experiment # 2 to 8.
    - With only **80 TOP features**, max accuracy achieved for (class 0 & 1 pair) and (class 7 & 9 pair) - **67 % and 77 %** respectively
3. When features were increased, then
    - class pair – 7 & 9 showed max accuracy – 81 % with 160 features and
    - class pair – 0 & 1 showed max accuracy – 74 % with 280 features.
4. Based on all the experiments (2 to 8), selection of **features** based on a mutual information method look appropriate in this problem out of total 784 features, for achieving good reasonable accuracy (~ 74 % to 81 %) for all the algorithms.
5. Out of all the experiments, one thing is pretty evident that original data set in MNIST database for class pair 7 & 9 are highly distinct and relevant and less redundant compared to the class pair 0 & 1.
6. With too high number of dimensions, this algorithm seems to be performing low (mostly curse of dimensionality).
7. With number of features from 80 to 280, accuracy seems increasing and after that slope starts dipping for both class pairs (Below graph).