

# Pattern Recognition Assignment

Siddhartha Singh Bhadauriya(2k18/se/123)

## Problem Statement

In this assignment, the problem is basically to design a Bayes classifier for classifying the numbers in the MNIST handwritten digit recognition databases.

### Objectives:

- (a) Design a classifier to distinguish between 0 and 1. Use the training database available for both to design the classifier.
- (b) Use the testing database to compute the classification accuracy  $((TP+TN)/(TP+TN+FP+FN))$  of the Bayesian model.
- (c) Plot the ROC curves between FAR vs GAR.
- (d) Repeat the above three parts for classifying the digits 3 and 8.
- (e) Compare and contrast the results of the two classifiers.

## Solution:

**Dataset used:** We will use Mnist dataset, which can be taken from the given link (<http://yann.lecun.com/exdb/mnist/>)

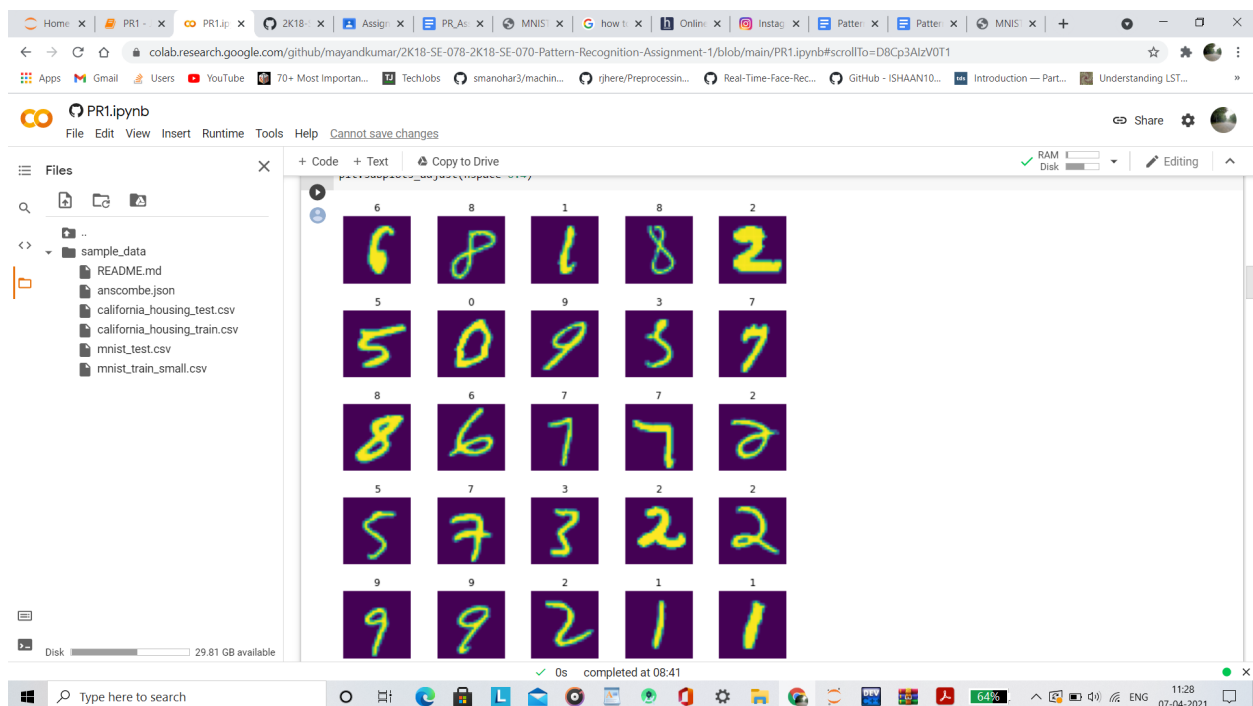
## Dataset Description:

The MNIST database of handwritten digits, available from this page, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on preprocessing and formatting.

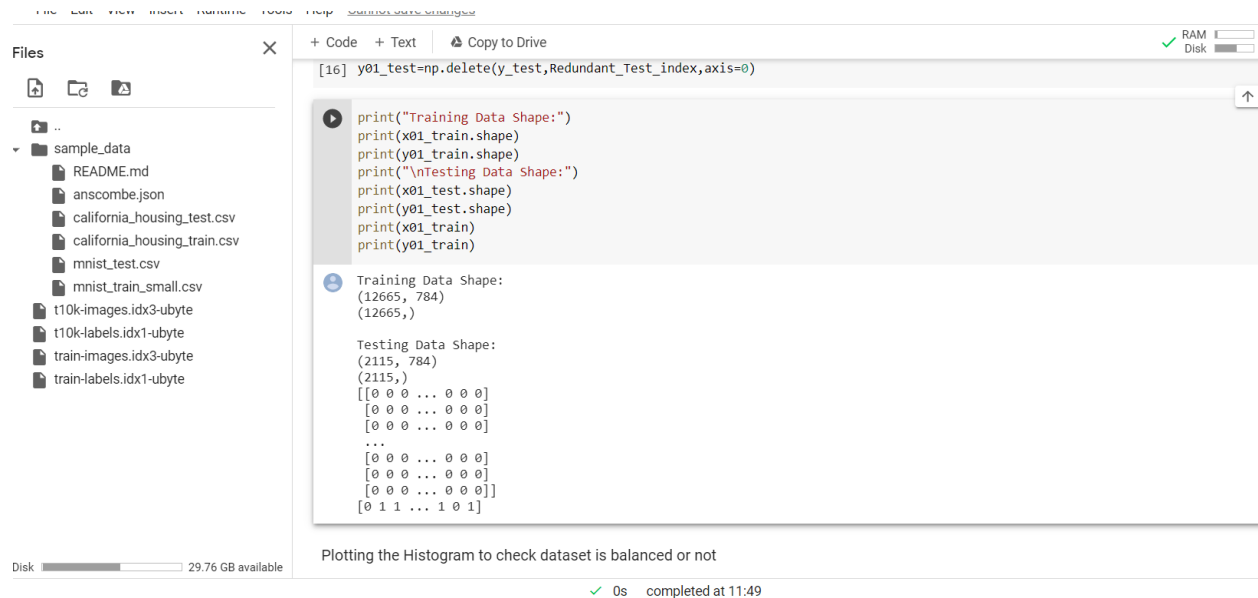
## Dataset Preprocessing:

Initially dataset tuples were in the form of  $28 \times 28$  matrix. So, to make dataset tuple suitable for input in our model, we flattened the tuples resulting in an array of 784 elements where each element corresponds to a pixel of the image.



## Task is to make data frame for classification between 0 and 1:

For this, we will have to make a dataframe, looking into the labels of our given dataset, so after pruning from there, we will get a new dataframe, having 12665 train and 2115 in test set.



The screenshot shows a Jupyter Notebook environment. On the left, a file explorer displays a directory structure with files like 'sample\_data', 'README.md', 'anscombe.json', 'california\_housing\_test.csv', 'california\_housing\_train.csv', 'mnist\_test.csv', 'mnist\_train\_small.csv', 't10k-images.idx3-ubyte', 't10k-labels.idx1-ubyte', 'train-images.idx3-ubyte', and 'train-labels.idx1-ubyte'. The main area contains a code cell with the following Python code:

```
[16] y01_test=np.delete(y_test,Redundant_Test_index,axis=0)

print("Training Data Shape:")
print(x01_train.shape)
print(y01_train.shape)
print("\nTesting Data Shape:")
print(x01_test.shape)
print(y01_test.shape)
print(x01_train)
print(y01_train)
```

The output of the code cell shows the shapes of the training and testing data:

```
Training Data Shape:
(12665, 784)
(12665,)

Testing Data Shape:
(2115, 784)
(2115,)
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
[[0 1 1 ... 1 0 1]
```

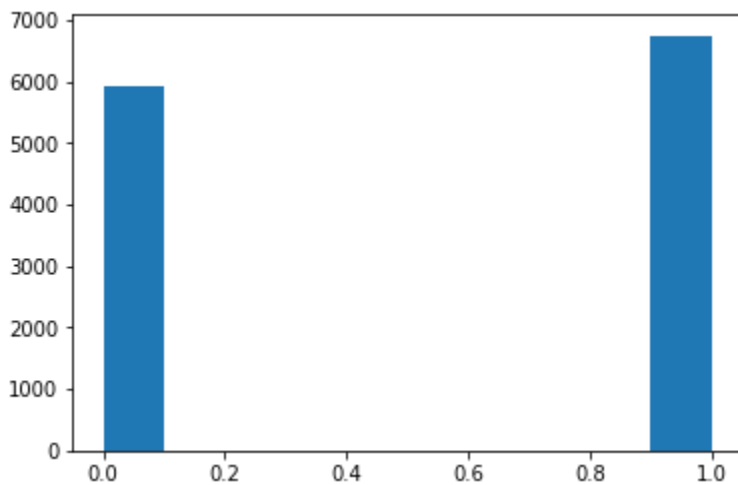
At the bottom, a status bar indicates 'Plotting the Histogram to check dataset is balanced or not' and '0s completed at 11:49'.

Similarly, when we do for 3 and 8 classification, we see the number of items in the train are 11982 and items in the test are 1984.

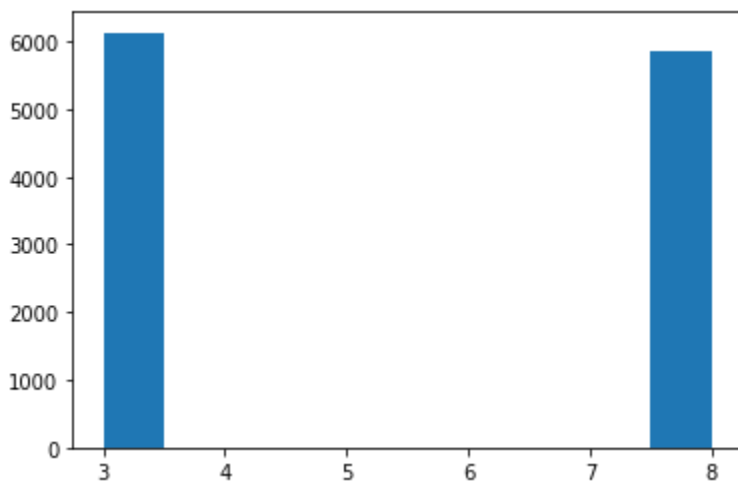
```
Training Data Shape:
(11982, 784)
(11982,)

Testing Data Shape:
(1984, 784)
(1984,)
```

To visualize the number of 0 and 1 in form of a histogram:



To visualize 3 and 8 in form of numbers:



### **Classifier Used:**

Gaussian Naive Bayes:

In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is also called Normal distribution.

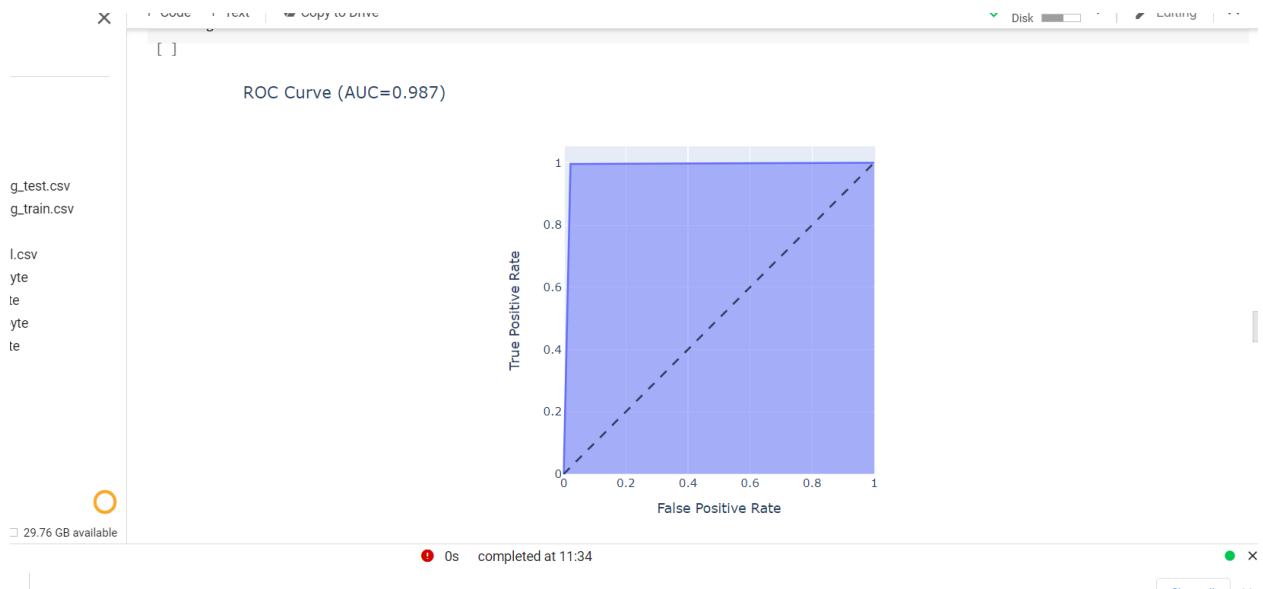
### **Performances Measure:**

We basically used accuracy, precision , recall , f1-score as our performance

measures . We have evaluated the confusion matrix. Along with this,we will also plot the ROC curve for this.

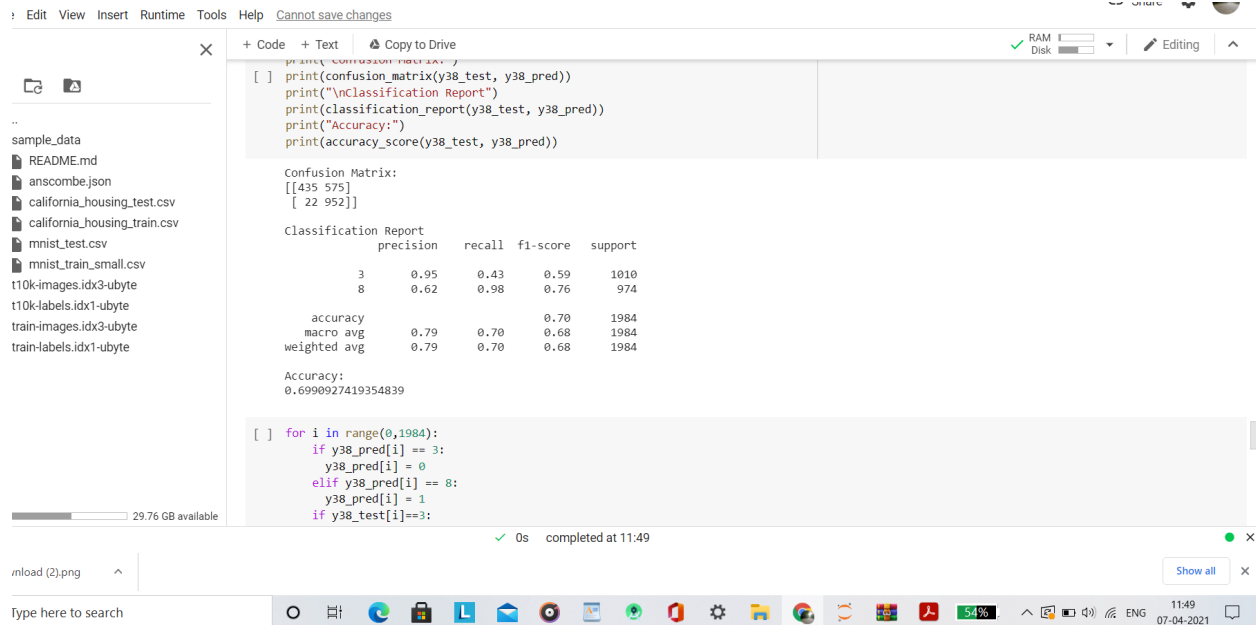


ROC curve for 0 and 1 classification:



Now, for classification between 3 and 8:

We will show our accuracy scores and ROC curve:



The screenshot shows a Jupyter Notebook interface with a file explorer on the left, a code editor in the center, and a terminal at the bottom. The code in the notebook prints the confusion matrix, classification report, and accuracy score for a model trained on digits 3 and 8. The results are as follows:

```
print(confusion_matrix(y38_test, y38_pred))
print("\nClassification Report")
print(classification_report(y38_test, y38_pred))
print("Accuracy:")
print(accuracy_score(y38_test, y38_pred))
```

Confusion Matrix:

```
[[435 575]
 [ 22 952]]
```

Classification Report

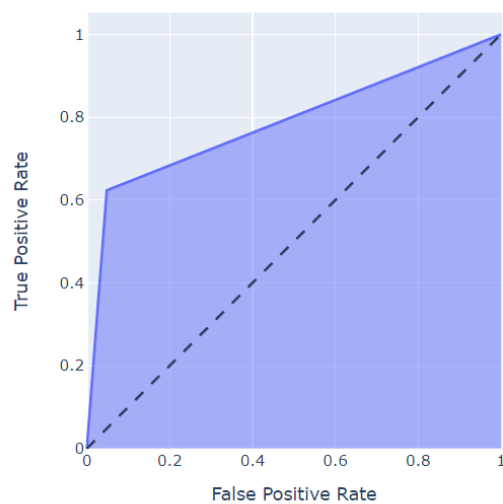
	precision	recall	f1-score	support
3	0.95	0.43	0.59	1010
8	0.62	0.98	0.76	974
accuracy			0.70	1984
macro avg	0.79	0.70	0.68	1984
weighted avg	0.79	0.70	0.68	1984

Accuracy:

```
0.6990927419354839
```

The terminal at the bottom shows the command `mload (2).png` and the status `0s completed at 11:49`.

Now, ROC curve for 3 and 8 classification:



## **Results and Conclusion:**

What we analysed from the result of performance measures that we took into consideration for this problem was that our Bayesian Model performed poor in classifying the digits.

We noticed in case of 3 and 8(0.699) which are quite similar in appearance, there was much less accuracy as compared to accuracy in case of 0 and 1(0.98).