Naïve Bayes Classifier for classifying handwritten digits.

- In this task, we implemented Naive Bayes Classifier with laplace smoothing
- Each digit is represented by 28x28 pixels. So we implemented a dictionary with digits as the key and the 28x28 matrix as its value
- We decided to consider a '+' and '#' as 1 feature and ' ' as another feature for identification of digits
- In the initial step, we parse the data of training images. To handle zero counts, we implemented laplace smoothing with smoothing factor k = 1. So for each digit, the pixels in matrix have value as 1 (line 10 in naivebayes.py)
- Whenever we encounter a ' ' at location (i,j) , we do not increment value for the matrix of that digit. Whenever we encounter a '+' or a '#' , we increment the value for that pixel by 1 (line 36 in naive bayes.py)
- Also, the prior probability of each digit is stored (line 23 and 25 in naivebayes.py)
- After the training is done, we've the prior probability as well as probability of seeing a '+' or '#' at each of 28x28 pixel for each digit i.e we have
  - P(D = 0), P(D = 1) ... P(D=9)
  - P(F = '+' or F = '#' , L = (0,0) | D = 0) ..P(F = '+' or F = '#' , L = (27,27) | D = 0) for D = 0 to 9
- Also, the prior probability of each digit is stored (line 23 and 25 in naivebayes.py)
- For testing the naive bayes classifier on each image of testimages.txt , we store all the images in a dictionary. The key is the index of the test image and value is 28x28 matrix as described previously
- To classify each image, we do
  - P(D = 1| F = ' ', L = (0,0) ..F = '+', L = (27,27)) = P(F = ' ', L = (0,0) | D = 1)* .. P(F = '+', L = (27,27)* P(1) (line 84 to 122 in naivebayes.py)
  - We take log to compute the actual value of above equation. This computation is done for each digit
  - The digit which has the highest probability is selected as the label for this test image (line 125,126 in naivebayes.py)
- We compare our assigned label with the label given and if it matches, we increase the value of counter
- In the end, we get an overall accuracy of **77.1 %**
- The accuracy for each digit is shown in below table :

Digit	FrequencyInTestData	LabelledCorrectly	Accuracy
0	90	76	84.444444
1	108	104	96.296296
2	103	80	77.669903
3	100	79	79.000000
4	107	82	76.635514

```
5          92             62        67.391304

6          91             69        75.824176

7         106             77        72.641509

8         103             62        60.194175

9         100             80        80.000000

-----------------------------------------------------------------------------
```