Ziming Dong

CSE 575

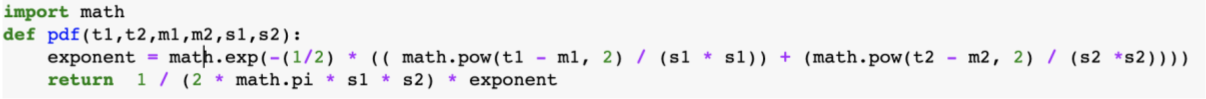
Project Part 1: Density Estimation and Classification Report

Instructor: Baoxin Li

Date: 9/22/2019

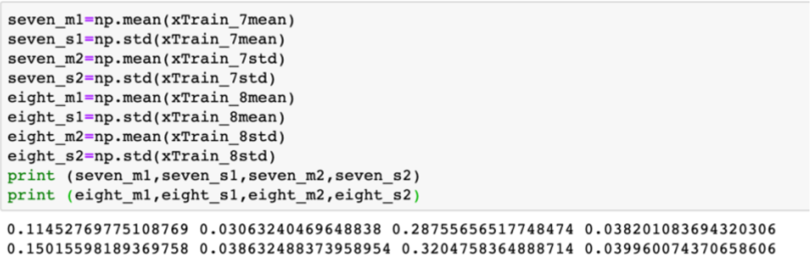
For this project, I use two classification methods to perform parameter estimation for given dataset, Naive Bayes and Logistic Regression classification. I get digit “7” and “8” images in training set and testing set, and I need to extract two features for each image, the average of all pixel values in the image and the standard deviation of all pixel values in the image. Then I begin to implement the classification on the dataset.

First, I would like to talk about how I calculated the classification accuracy for both “7” and “8” in the testing set. After I get two features for each image for “7” and “8”, I use MLE density Estimation to calculate each feature`s mean and standard deviation from training data, which is the 2-D normal distribution for each digit.  As we know, Naïve Bayes formula is $\underbrace{P(c\,|\,x)}_{\text{Posterior Probability}}=\frac{\overbrace{P(x\,|\,c)}^{\text{Likelihood}}\overbrace{P(c)}^{\text{Class Prior Probability}}}{\underbrace{P(x)}_{\text{Predictor Prior Probability}}}$. Thus, we need to get priority probability and pdf for digit “7” and “8” `s classifiers. The priority probability is easy to get: image (7 or 8) / all images. The formula we use to estimate the parameters shows below,



```
import math
def pdf(t1,t2,m1,m2,s1,s2):
    exponent = math.exp(-(1/2) * (( math.pow(t1 - m1, 2) / (s1 * s1)) + (math.pow(t2 - m2, 2) / (s2 *s2))))
    return  1 / (2 * math.pi * s1 * s2) * exponent
```

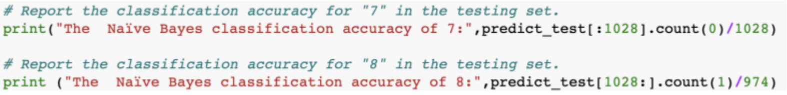
Which is similar $f(x)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$, but since we are using 2-D normal distribution, we will have two means and two standard deviations to imply this formula. After we get PDF and priority probability for classifier “7” and “8”, we apply the value of means and standard deviations which we estimate from training dataset to fill in PDF formula and multiply priority, I give the result name as p1 and p2 after I applying two features from testing



```
seven_m1=np.mean(xTrain_7mean)
seven_s1=np.std(xTrain_7mean)
seven_m2=np.mean(xTrain_7std)
seven_s2=np.std(xTrain_7std)
eight_m1=np.mean(xTrain_8mean)
eight_s1=np.std(xTrain_8mean)
eight_m2=np.mean(xTrain_8std)
eight_s2=np.std(xTrain_8std)
print (seven_m1,seven_s1,seven_m2,seven_s2)
print (eight_m1,eight_s1,eight_m2,eight_s2)
```

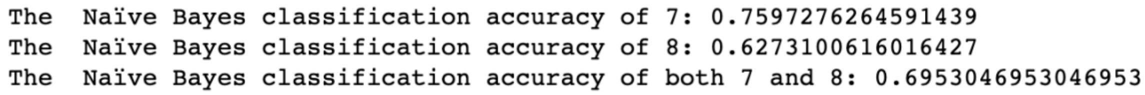
```
0.11452769775108769 0.03063240469648838 0.28755656517748474 0.038201083694320306
0.15015598189369758 0.038632488373958954 0.3204758364888714 0.039960074370658606
```

dataset.  I compare two ps value to identify the digit. Then I write a loop to account how many 0s and 1s in the prediction test. The accuracy of the classification for digits can be writen as:



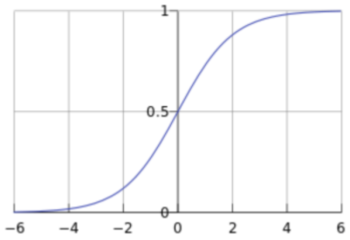
```
# Report the classification accuracy for "7" in the testing set.
print("The  Naïve Bayes classification accuracy of 7:",predict_test[:1028].count(0)/1028)

# Report the classification accuracy for "8" in the testing set.
print ("The  Naïve Bayes classification accuracy of 8:",predict_test[1028:].count(1)/974)
```

The final classification accuracy for both "7" and "8" for the testing set will be:



```
The  Naïve Bayes classification accuracy of 7: 0.7597276264591439
The  Naïve Bayes classification accuracy of 8: 0.6273100616016427
The  Naïve Bayes classification accuracy of both 7 and 8: 0.6953046953046953
```

Second, let us talk about steps on train a Logistic Regression model using gradient ascent and how it can report the classification accuracy for both "7" and "8" in the testing set. To implement the Logitstic Regression, we need to know Sigmoid function $g(z) = \frac{1}{1+e^{-z}}$, its function curve is as follows:



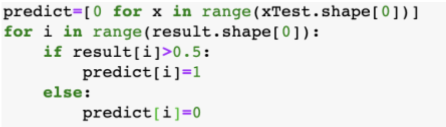
, As you can see from the above figure, the sigmoid function is an shaped curve whose value is between [0, 1]. The value of the function will be close to 0 or 1 when it is far away from 0. This feature is important for solving the two-category problem. Now we can turn the problem into a logistic, the best regression coefficient for regression. Since logistic regression can be regarded as a probability model, and the probability of output y occurring is related to the regression parameter θ, we can maximize the likelihood of θ, making y the most likely probability of occurrence. The θ is the optimal regression coefficient. On the entire data set request likelihood function obtained:

$$\ell(\theta) = \log L(\theta) = \sum_{i=1}^{m} [y^{(i)} log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

, and for the formula using gradient ascent method, to obtain the iterative formulas θ, the calculation result is

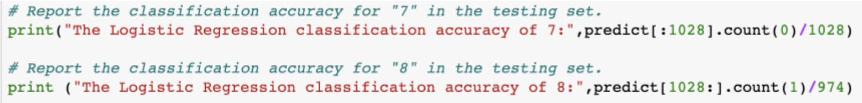
$$\theta := \theta + \alpha \sum_{i=1}^{m} \left( y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}$$

, a is the learning rate which I set value as 0.001, it determines how fast the function rises. In my code, I set up two matrixes, one is for the training data`s two features (mean and standard deviation), one is for the test dataset matrix with its two features. Once we get the best theta value by gradient ascent, I can get a new result by use sigmoid function with the test dataset matrix multiply theta. The last step is to compare the results:



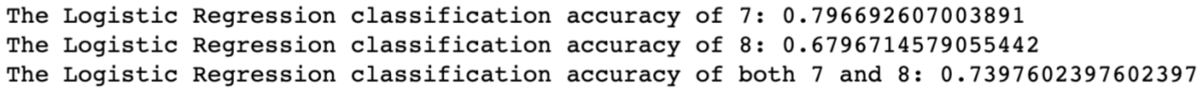
```
predict=[0 for x in range(xTest.shape[0])]
for i in range(result.shape[0]):
    if result[i]>0.5:
        predict[i]=1
    else:
        predict[i]=0
```

, then we can use the same way which I use in Naïve Bayes classification to count how many label 0s and 1s in the test dataset, to get each digit`s prediction accuracy, I use this method shows below:



```
# Report the classification accuracy for "7" in the testing set.
print("The Logistic Regression classification accuracy of 7:",predict[:1028].count(0)/1028)

# Report the classification accuracy for "8" in the testing set.
print ("The Logistic Regression classification accuracy of 8:",predict[1028:].count(1)/974)
```

The final classification accuracy for both “7” and “8” for the testing set are:



```
The Logistic Regression classification accuracy of 7: 0.796692607003891
The Logistic Regression classification accuracy of 8: 0.6796714579055442
The Logistic Regression classification accuracy of both 7 and 8: 0.7397602397602397
```

In conclusion, this project let me have a chance to review the material for Naïve Bayes and Logistic Regression algorithm, I consider there may be some factors to influence the accuracy of prediction, such as underfitting, overfitting. I also need to think about the set value of the learning rate and repeat cycles for Logistic Regression. I would like to test more cases by justifying the parameters in the future.