Assignment: Naive Bayes

65 points

For this problem you cannot use an existing Naive Bayes classifier implementation or package.

Implement a Naive Bayes classifier in R or Python to apply it to the task of classifying handwritten digits. Files mnist-train and mnist-test contain training and test digits, together with their ground truth labels (first column). Each row in these files corresponds to a different digit.

Each image is 28x28, hence there are 784 pixel in every image. Columns 2-785 in the data files correspond to the pixel intensity, a value between 0 to 255. Column 1 corresponds to the correct label for each digit.

You should convert the pixel intensities to a single binary indicator feature (F_i) for each pixel. Specifically, if the intensity is smaller than 255/2 map it to a zero, otherwise to a one.

1. (10 points) Estimate the priors $P(class)$ based on the frequencies of different classes in the training set. **Report the values in a table. Round to 3 decimal places.**

2. (15 points) Estimate the likelihoods $P(F_i|class)$ for every pixel location i and for every digit class from 0 to 9. The likelihood estimate is

   $P(F_i = f|class) =$(Number of times pixel i has value f in training examples from this class) / (Total number of training examples from this class)

   Note that you have to smooth the likelihoods to ensure that there are no zero counts. Laplace smoothing is a very simple method that increases the observation count of every value f by some constant k. This corresponds to adding k to the numerator above, and k*V to the denominator (where V is the number of possible values the feature can take on). The higher the value of k, the stronger the smoothing. Experiment with different integer values of k from 1 to 5. While you need to find all the likelihoods for k=1 to 5, I'd like you to report the following values in your report: **For k=1 and k=5** $P(F_{682} = 0|class = 5)$ **and** $P(F_{772} = 1|class = 9)$**. Round to 3 decimal places.**

3. (25 points) Perform maximum a posteriori (MAP) classification of test digits according to the learned Naive Bayes modeles. Suppose a test image has feature values f_1, f_2, ... , f_{784}. According to this model, the posterior probability (up to scale) of each class given the digit is given by:

   $P(class)P(f_1|class)(f_2|class)...P(f_{784}|class)$

Note that in order to avoid underflow, you need to work with the log of the above quantity:

$$\log P(class) + \log P(f_1|class) + \log P(f_2|class) + ... + \log P(f_{784}|class)$$

Compute the above decision function values for all ten classes for every test image, then use them for MAP classification. **For the first test image, report the log posterior probability of** $P(class = 5|f_1, f_2, ..., f_{784})$ **and** $P(class = 7|f_1, f_2, ..., f_{784})$ **for k=1 and k=5.**

4. (10 points) Use the true class labels of the test images from the mnist_test file to check the correctness of the estimated label for each test digit. **Report your performance in terms of the classification rate (percentage of all test images correctly classified) for each value of k from 1 to 5.**

5. (5 points) **Report your confusion matrix for the best k.** This is a 10x10 matrix whose entry in row r and column c is the percentage of test images from class r that are classified as class c. (Tip: You should be able to achieve at least 70% accuracy on the test set.)