MNIST Digit Recognition Using Naive Bayes

Saba Kanwal(18L-1860)

December 8, 2018

Abstract

In this report, simplest probability distribution approach Naive Bayes is used for digit recognition task. .And I used binary images of MNIST digit dataset for this task and Laplace smoothing is applied on probabilities to avoid probability zero issue. This method correctly classify 8427 testing examples and total examples are 10,000. So, accuracy achieved by this method is 84%. And all of the computations takes less than one minute. This method proves really fast all previous implemented methods.

Chapter 1

Naive Bayes

Chain rule is a way to write joint distribution as an incremental product of conditional distributions. Chain rule is given in below equation:

$$P(x_1, x_2, ..., x_n) = \prod_i P(x_i|x_1, ..., x_{i-1}) \tag{1.1}$$

Naive Bayes is a simple assumption that says if we know the actual label of example in classification task, then all features of this example are independent of each other. Then this can be written in simple equation as:

$$P(Y|F_1, F_2, ..., F_{784}) = P(Y).\prod_i P(F_i|Y) \tag{1.2}$$

Chapter 2

Experiments

In this experiment, i converted gray scale images of digit in to binary image by simply applying threshold. The threshold value used is 127. The values less than 127 are taken as 0 and greater values are taken as 1 in new images. In single pass, all of the probabilities computed. I defined digit_distribution and probability_distribution two arrays. Digit_distribution array is one dimensional and has length 10. For each training label we increment digit_distribution[label] by one. Probability_distribution array is of two dimensional and its dimensions are 10*784. First row contain count of each feature that is on with label 0. Second row contain count of each feature on given that label=1 and so on. Now, counts maintained in both of the arrays. These counts are converted into probabilities by dividing each count with total examples in case of digit_distribution or total digit count in case of probability_distribution. These two tables are used for classification of testing examples. Probabilities zero issue is resolved by applying Laplace smoothing. How much smoothing is performed? I tried different values of k, the most accuracy achieved at k =1. The results are shown in table below

Training size	Testing size	Accuracy	K for smoothing	Time
60,000	10,000	84.25%	1	within one minute
60,000	10,000	84.12%	5	within one minute
60,000	10,000	83.95%	10	within one minute
60,000	10,000	83.95%	15	within one minute
60,000	10,000	83.90%	20	within one minute
60,000	10,000	83.79%	30	within one minute

Chapter 3

Conclusion

From above experiment, we conclude that very less smoothing is required but it's essential with out smoothing most of the probabilities becomes zero. The highest accuracy achieved by chain rule is 84.25% . This method is easier to implement and faster than all of the previous studied methods. It perform training and testing within one minute.