ECE448

MP3

Naive Bayes Classification

Date:2017/11/26

Credit: 3

Section: R

Team Member:

Jiaying Wu (jwu86)

Qin Li (qinli2)

Yuchen Liang (yliang35)

Part 1:

1.1 Single pixels as features

Description:

In this part, we need to train a digit classifier using single raw pixels as features. In our implementation, the train and test sets are described as a list of objects of class *digit*, which contains a numpy matrix that represents the pattern of the digit (denoted by feature matrix below) and its corresponding label. In the preprocessing step, the matrix element is set to one if it is not a background pixel at that index, and zero otherwise. In training, we get the probability of one at every pixel location (i.e. 28*28) by taking the mean of the sum of all feature matrix of a given digit class. Besides, we also obtain the prior of each class. During testing, we calculate the log a posteriori probability (see next paragraph for the smoothing constant) of an instance to lie in each class, and pick the class that maximizes the a posteriori probability.

To prevent taking the log of zero, we use Laplace smoothing by adding a small constant k on the numerator and 2*k (since the feature can take values 0 or 1) on the denominator when calculating probability of each class. The constant k is a hyper parameter that ranges from 0.1 to 10. As shown in the table below, the accuracy decreases as k increases from 0.1 to 10. Hence, to maximize overall accuracy, we choose k = 0.1.

k	0.1	0.2	0.3	0.4	0.5	1	2	3	5	10
Overall accuracy	.772	.772	.772	.772	.77	.77	.763	.762	.759	.757

The following results assume that k=0.1.

```
The corresponding k is:0.1
The overall accuracy is: 0.772
The confusion matrix is:
[[ 0.844  0.     0.011  0.     0.011  0.056  0.033  0.     0.044  0.    ]
 [ 0.     0.963  0.009  0.     0.     0.019  0.009  0.     0.     0.    ]
 [ 0.01   0.029  0.786  0.039  0.019  0.     0.058  0.01   0.049  0.    ]
 [ 0.     0.01   0.     0.8    0.     0.03   0.02   0.07   0.01   0.06  ]
 [ 0.     0.     0.009  0.     0.748  0.009  0.037  0.009  0.019  0.168]
 [ 0.022  0.011  0.011  0.13   0.033  0.685  0.011  0.011  0.022  0.065]
 [ 0.011  0.044  0.044  0.     0.044  0.066  0.769  0.     0.022  0.    ]
 [ 0.     0.047  0.038  0.     0.028  0.     0.     0.726  0.028  0.132]
 [ 0.01   0.01   0.029  0.136  0.029  0.087  0.     0.01   0.592  0.097]
 [ 0.01   0.01   0.     0.03   0.1    0.02   0.     0.02   0.01   0.8  ]]
```

Classification rate:

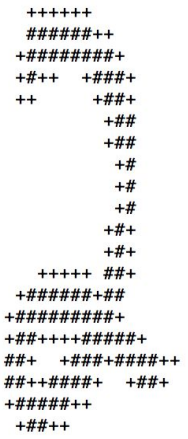
Class	0	1	2	3	4	5	6	7	8	9
Accuracy for each class	0.844	0.963	0.786	0.8	0.748	0.685	0.769	0.726	0.592	0.8

Test examples with highest posterior probabilities:

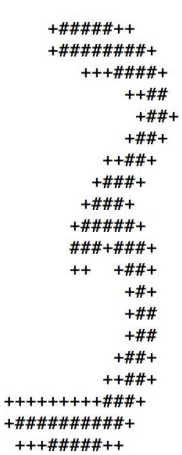
Max Class: 0 Index: 723

Max Class: 1 Index: 633

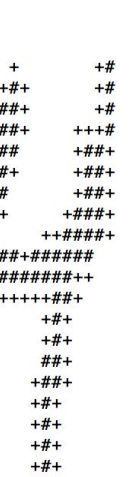
Max Class: 2 Index: 795



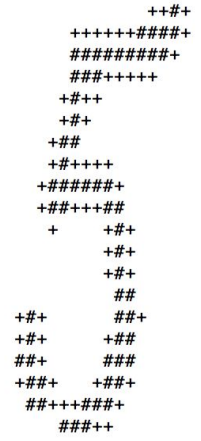
Max Class: 3 Index: 205



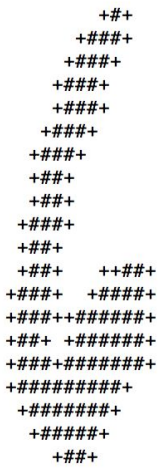
Max Class: 4 Index: 111



Max Class: 5 Index: 471



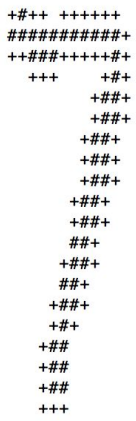
Max Class: 6 Index: 632

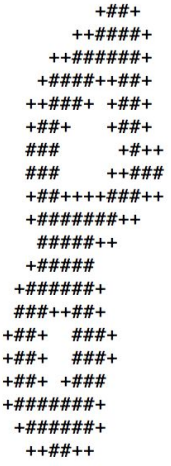


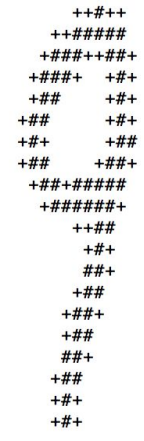
Max Class: 7 Index: 784

Max Class: 8 Index: 560

Max Class: 9 Index: 58

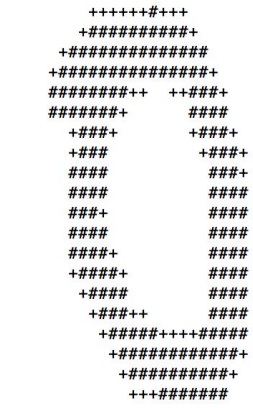




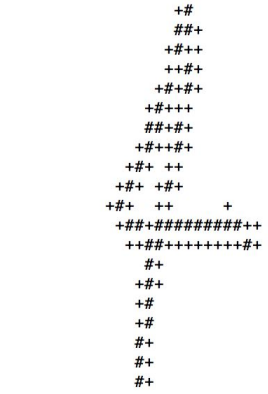


Test examples with lowest posterior probabilities:

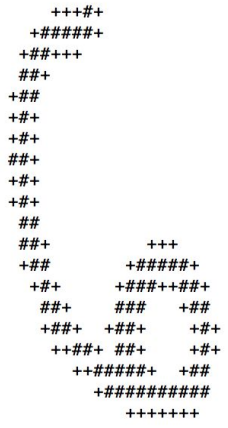
Min Class: 0 Index: 610



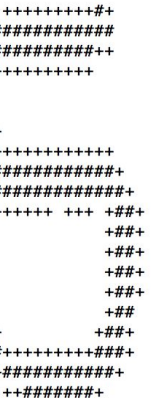
Min Class: 1 Index: 465



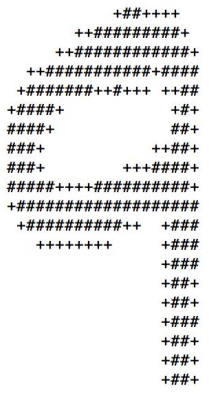
Min Class: 2 Index: 17



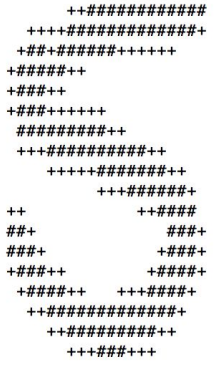
Min Class: 3 Index: 896



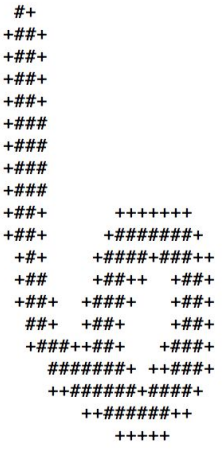
Min Class: 4 Index: 801



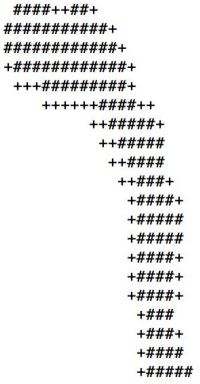
Min Class: 5 Index: 70



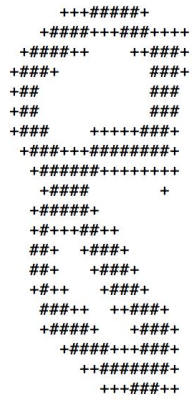
Min Class: 6 Index: 362



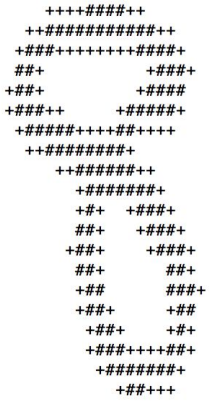
Min Class: 7 Index: 671



Min Class: 8 Index: 758

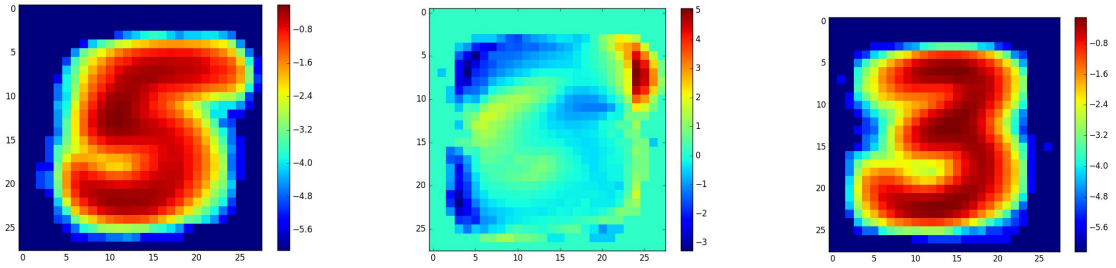


Min Class: 9 Index: 882



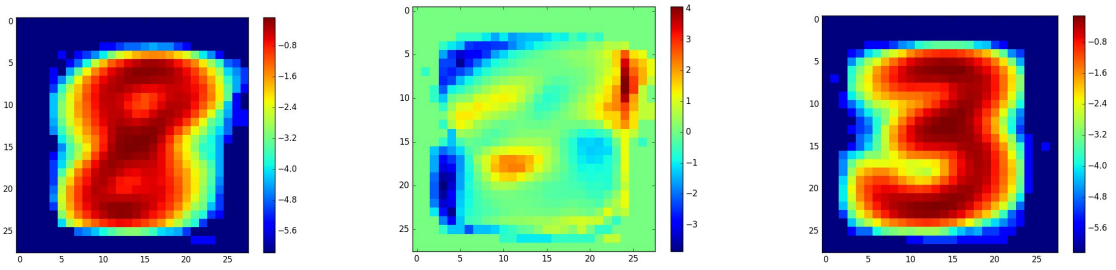
Odds Ratio:

1.Odds Ratio of 5 and 3: confusionmatrix[5][3] =0.13



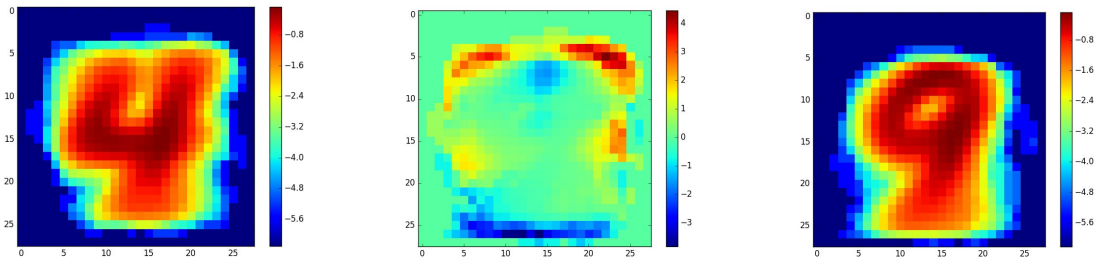
According to the plot above and the data of odds[5][3], we can see that 5 and 3 are pretty similar in the bottom part so that the odds ratio plot of the plot lower part are nearly 0, and the difference is significant at the upper part where the deep color and red color appears.

2.Odds Ratio of 8 and 3: confusionmatrix[8][3]=0.136



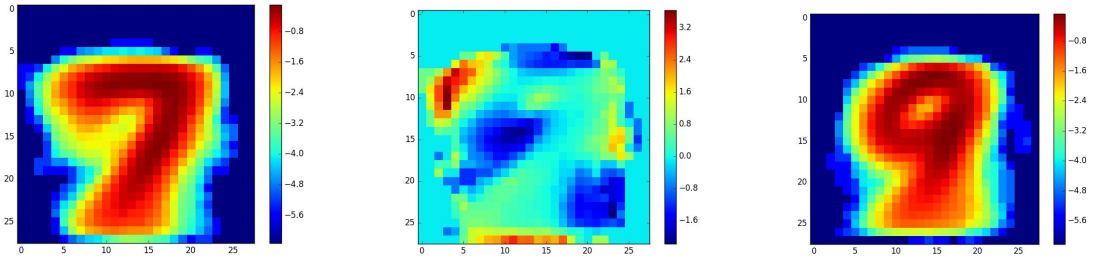
According to the plot above and the data of odds[8][3], we can see that 8 and 3 are pretty similar in the right part where few red or deep blue color appears, and the difference is significant at the left part where the deep color and red color appears.

3.Odds Ratio of 4 and 9: confusionmatrix[4][9]=0.168



According to the plot above and the data of odds[4][9], we can see that 4 and 9 are pretty similar in the bottom part so that the odds ratio plot of the plot lower part are nearly 0, and the difference is significant at the upper part where the deep color and red color appears.

4. Odds Ratio of 7 and 9: confusionmatrix[7][9]=0.132



According to the plot above and the data of odds[7][9], we can see that 7 and 9 are pretty similar in the bottom part so that the odds ratio plot of the plot lower part are nearly 0, and the difference is significant at the upper and middle part where the deep color and red color appears.

1.2 : Pixel groups as features (EXTRA CREDIT)

First, we show the results for the different kinds of desired patches below:

- **Disjoint:**

Disjoint patches size	Accuracies	Training time	Testing time
2*2	0.843	6.85s	4.07s

```
The overall accuracy is: 0.843
The confusion matrix is:
[[ 0.933  0.     0.     0.     0.     0.011  0.022  0.     0.033  0.   ]
 [ 0.     0.981  0.009  0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.01   0.01   0.835  0.019  0.01   0.01   0.039  0.019  0.039  0.01 ]
 [ 0.     0.01   0.     0.86   0.     0.03   0.02   0.02   0.02   0.04 ]
 [ 0.     0.009  0.     0.     0.869  0.     0.019  0.009  0.009  0.084]
 [ 0.011  0.011  0.     0.12   0.     0.783  0.011  0.011  0.033  0.022]
 [ 0.011  0.044  0.     0.     0.022  0.055  0.868  0.     0.     0.   ]
 [ 0.     0.047  0.047  0.     0.019  0.     0.     0.774  0.009  0.104]
 [ 0.019  0.019  0.039  0.126  0.01   0.019  0.01   0.01   0.709  0.039]
 [ 0.01   0.01   0.     0.02   0.08   0.01   0.     0.02   0.03   0.82 ]]
```

Disjoint patches size	Accuracies	Training time	Testing time
2*4	0.827	6.03s	2.72s

```
The overall accuracy is: 0.827
The confusion matrix is:
[[ 0.922  0.     0.     0.011  0.     0.     0.033  0.011  0.022  0.   ]
 [ 0.     0.981  0.009  0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.01   0.078  0.777  0.01   0.019  0.     0.039  0.019  0.039  0.01 ]
 [ 0.     0.02   0.     0.89   0.     0.01   0.02   0.06   0.     0.   ]
 [ 0.     0.009  0.     0.     0.935  0.     0.019  0.009  0.     0.028]
 [ 0.033  0.022  0.011  0.185  0.011  0.652  0.011  0.011  0.033  0.033]
 [ 0.022  0.055  0.     0.     0.022  0.033  0.868  0.     0.     0.   ]
 [ 0.     0.066  0.038  0.     0.028  0.     0.     0.774  0.009  0.085]
 [ 0.019  0.058  0.029  0.126  0.029  0.01   0.019  0.029  0.66   0.019]
 [ 0.01   0.01   0.     0.03   0.08   0.     0.     0.06   0.01   0.8  ]]
```

Disjoint patches size	Accuracies	Training time	Testing time
4*2	0.847	6.44s	2.67s

```
The overall accuracy is: 0.847
The confusion matrix is:
[[ 0.956  0.     0.     0.     0.     0.     0.022  0.011  0.011  0.   ]
 [ 0.     0.991  0.     0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.019  0.029  0.845  0.01   0.029  0.     0.019  0.019  0.029  0.   ]
 [ 0.     0.01   0.02   0.9    0.     0.     0.01   0.03   0.01   0.02 ]
 [ 0.     0.019  0.     0.     0.925  0.     0.019  0.009  0.     0.028]
 [ 0.022  0.011  0.011  0.185  0.043  0.641  0.011  0.011  0.033  0.033]
 [ 0.011  0.033  0.011  0.     0.022  0.033  0.89   0.     0.     0.   ]
 [ 0.     0.066  0.028  0.     0.009  0.     0.     0.821  0.009  0.066]
 [ 0.019  0.029  0.039  0.136  0.01   0.     0.019  0.01   0.689  0.049]
 [ 0.01   0.     0.     0.03   0.08   0.     0.     0.06   0.02   0.8  ]]
```

Disjoint patches size	Accuracies	Training time	Testing time
4*4	0.753	5.98s	1.87s

```
The overall accuracy is: 0.753
The confusion matrix is:
[[ 0.9    0.     0.     0.     0.     0.     0.078  0.011  0.011  0.   ]
 [ 0.     0.991  0.     0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.019  0.214  0.67   0.     0.029  0.     0.019  0.039  0.01   0.   ]
 [ 0.     0.1    0.     0.77   0.     0.     0.02   0.1    0.     0.01 ]
 [ 0.     0.037  0.     0.     0.897  0.     0.019  0.019  0.     0.028]
 [ 0.033  0.109  0.     0.293  0.065  0.37   0.033  0.022  0.033  0.043]
 [ 0.     0.066  0.     0.     0.044  0.011  0.879  0.     0.     0.   ]
 [ 0.     0.104  0.     0.     0.028  0.     0.     0.84   0.     0.028]
 [ 0.019  0.194  0.019  0.136  0.058  0.     0.029  0.087  0.427  0.029]
 [ 0.02   0.03   0.     0.02   0.1    0.     0.     0.07   0.     0.76 ]]
```

- **Overlapping:**

Overlapping patches size	Accuracies	Training time	Testing time
2*2	0.854	25.66s	15.22s

```
The overall accuracy is: 0.854
The confusion matrix is:
[[ 0.933  0.     0.     0.     0.     0.011  0.033  0.     0.022  0.   ]
 [ 0.     0.981  0.009  0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.01   0.019  0.825  0.029  0.01   0.     0.049  0.019  0.029  0.01 ]
 [ 0.     0.     0.     0.88   0.     0.02   0.01   0.03   0.02   0.04 ]
 [ 0.     0.009  0.     0.     0.869  0.     0.019  0.009  0.009  0.084]
 [ 0.011  0.     0.     0.12   0.     0.793  0.011  0.011  0.022  0.033]
 [ 0.011  0.033  0.     0.     0.022  0.066  0.868  0.     0.     0.   ]
 [ 0.     0.047  0.038  0.     0.019  0.     0.     0.783  0.019  0.094]
 [ 0.019  0.01   0.039  0.117  0.     0.019  0.     0.01   0.757  0.029]
 [ 0.01   0.     0.     0.02   0.06   0.01   0.     0.02   0.03   0.85 ]]
```

Overlapping patches size	Accuracies	Training time	Testing time
2*4	0.854	38.96s	17.13s

```
The overall accuracy is: 0.854
The confusion matrix is:
[[ 0.944  0.     0.     0.     0.     0.     0.033  0.     0.022  0.   ]
 [ 0.     0.991  0.     0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.019  0.058  0.786  0.     0.019  0.     0.039  0.019  0.039  0.019]
 [ 0.     0.01   0.     0.89   0.     0.02   0.01   0.07   0.     0.   ]
 [ 0.     0.009  0.     0.     0.944  0.     0.019  0.009  0.     0.019]
 [ 0.011  0.022  0.     0.141  0.022  0.728  0.     0.011  0.033  0.033]
 [ 0.011  0.055  0.     0.     0.011  0.033  0.89   0.     0.     0.   ]
 [ 0.     0.066  0.047  0.     0.019  0.     0.     0.811  0.     0.057]
 [ 0.019  0.039  0.029  0.126  0.019  0.     0.01   0.019  0.709  0.029]
 [ 0.01   0.01   0.     0.03   0.05   0.01   0.     0.04   0.01   0.84 ]]
```

Overlapping patches size	Accuracies	Training time	Testing time
4*2	0.859	40.19s	17.82s

```
The overall accuracy is: 0.859
The confusion matrix is:
[[ 0.944  0.     0.     0.     0.011  0.     0.022  0.011  0.011  0.   ]
 [ 0.     0.991  0.     0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.01   0.039  0.864  0.019  0.01   0.     0.019  0.019  0.019  0.   ]
 [ 0.     0.     0.     0.89   0.     0.     0.01   0.06   0.01   0.03 ]
 [ 0.     0.009  0.     0.     0.935  0.     0.019  0.009  0.     0.028]
 [ 0.022  0.011  0.011  0.174  0.011  0.685  0.011  0.011  0.022  0.043]
 [ 0.011  0.044  0.     0.     0.033  0.044  0.857  0.     0.011  0.   ]
 [ 0.     0.047  0.038  0.     0.009  0.     0.     0.84   0.009  0.057]
 [ 0.01   0.049  0.039  0.136  0.     0.01   0.01   0.019  0.709  0.019]
 [ 0.01   0.     0.     0.04   0.03   0.     0.     0.04   0.02   0.86 ]]
```

Overlapping patches size	Accuracies	Training time	Testing time
4*4	0.803	75.84s	22.36s

```
The overall accuracy is: 0.803
The confusion matrix is:
[[ 0.944  0.     0.     0.     0.011  0.     0.022  0.011  0.011  0.   ]
 [ 0.     0.991  0.     0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.019  0.136  0.728  0.     0.029  0.     0.039  0.029  0.019  0.   ]
 [ 0.     0.02   0.     0.87   0.     0.     0.01   0.09   0.     0.01 ]
 [ 0.     0.047  0.     0.     0.925  0.     0.019  0.     0.     0.009]
 [ 0.033  0.043  0.011  0.25   0.054  0.478  0.022  0.033  0.022  0.054]
 [ 0.011  0.055  0.     0.     0.044  0.011  0.868  0.011  0.     0.   ]
 [ 0.     0.094  0.009  0.     0.009  0.     0.     0.868  0.     0.019]
 [ 0.019  0.175  0.029  0.136  0.019  0.     0.019  0.039  0.544  0.019]
 [ 0.01   0.02   0.     0.04   0.06   0.     0.     0.08   0.     0.79 ]]
```

Overlapping patches size	Accuracies	Training time	Testing time
2*3	0.860	32.69s	16.11s

```
The overall accuracy is: 0.86
The confusion matrix is:
[[ 0.944  0.     0.     0.     0.     0.     0.033  0.     0.022  0.   ]
 [ 0.     0.981  0.009  0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.019  0.039  0.806  0.     0.019  0.     0.039  0.019  0.039  0.019]
 [ 0.     0.     0.     0.88   0.     0.02   0.01   0.05   0.01   0.03 ]
 [ 0.     0.009  0.     0.     0.897  0.     0.019  0.009  0.     0.065]
 [ 0.011  0.022  0.     0.12   0.     0.783  0.     0.011  0.033  0.022]
 [ 0.011  0.033  0.     0.     0.011  0.055  0.89   0.     0.     0.   ]
 [ 0.     0.057  0.047  0.     0.019  0.     0.     0.792  0.     0.085]
 [ 0.019  0.019  0.029  0.097  0.     0.01   0.     0.019  0.786  0.019]
 [ 0.01   0.     0.     0.02   0.06   0.01   0.     0.03   0.03   0.84 ]]
```

Overlapping patches size	Accuracies	Training time	Testing time
3*2	0.869	34.67s	16.61s

```
The overall accuracy is: 0.869
The confusion matrix is:
[[ 0.956  0.     0.     0.     0.011  0.     0.022  0.     0.011  0.   ]
 [ 0.     0.991  0.     0.     0.     0.     0.009  0.     0.     0.   ]
 [ 0.01   0.019  0.854  0.029  0.01   0.     0.029  0.019  0.029  0.   ]
 [ 0.     0.     0.     0.89   0.     0.01   0.01   0.04   0.01   0.04 ]
 [ 0.     0.009  0.     0.     0.935  0.     0.019  0.009  0.     0.028]
 [ 0.011  0.     0.     0.141  0.011  0.772  0.011  0.011  0.022  0.022]
 [ 0.011  0.044  0.     0.     0.022  0.044  0.868  0.     0.011  0.   ]
 [ 0.     0.038  0.047  0.     0.019  0.     0.     0.821  0.009  0.066]
 [ 0.01   0.019  0.039  0.136  0.     0.029  0.     0.01   0.728  0.029]
 [ 0.01   0.     0.     0.02   0.03   0.01   0.     0.03   0.03   0.87 ]]
```

Overlapping patches size	Accuracies	Training time	Testing time
3*3	0.837	46.48s	18.90s

```
The overall accuracy is: 0.837
The confusion matrix is:
[[ 0.944  0.     0.     0.     0.011  0.     0.022  0.011  0.011  0.    ]
 [ 0.     0.991  0.     0.     0.     0.     0.009  0.     0.     0.    ]
 [ 0.019  0.039  0.816  0.019  0.019  0.     0.039  0.019  0.019  0.01  ]
 [ 0.     0.01   0.     0.88   0.     0.     0.01   0.08   0.01   0.01  ]
 [ 0.     0.009  0.     0.     0.944  0.     0.019  0.009  0.     0.019]
 [ 0.022  0.022  0.011  0.174  0.033  0.598  0.022  0.033  0.033  0.054]
 [ 0.011  0.055  0.     0.     0.044  0.022  0.868  0.     0.     0.    ]
 [ 0.     0.057  0.047  0.     0.019  0.     0.     0.84   0.     0.038]
 [ 0.019  0.068  0.029  0.126  0.019  0.     0.01   0.019  0.68   0.029]
 [ 0.01   0.01   0.     0.05   0.06   0.     0.     0.07   0.01   0.79 ]]
```

From the results above, the average accuracy using overlapping patches is higher than that using disjoint patches. For the set of disjoint patches, patches of size 4*2 attain the maximum accuracy, 0.847, similar to the second highest, which is 0.843 with patch size 2*2. The other patches (2*4 & 4*4) have a lower accuracy because of the particular property of the training digit set. A patch with width 4 misclassifies much more 5 as 3 (i.e. nearly one third), and slightly more 8 as 1 and 3, lowering the overall accuracies. On the other hand, for the set of overlapping patches, the patch with size 3*2 obtains the highest accuracy, while 4*4 obtains the lowest. Generally, non-square patches obtain higher accuracy than square patches, and smaller patch sizes are more favored.

The training and testing running time is limited by the number of classes generated by the patch size and the number of patches to calculate. in general, the running time for the overlapping patches is much longer than that for the disjoint patches (i.e. about 4-8 times longer). For disjoint patches, a larger patch usually has shorter training and testing running time. For overlapping patches, a increase in patch size results in an enormous increase in both the training and testing running time.

Part 1 EXTRA

To increase the accuracy of the classification process, we use ternary feature. Instead of treating both '#' and '+' as one, we treat '+' as one and '#' as two. The method for calculation is the same as the method in part 1.2. Instead of having 2^(m*n) possibilities in one n*m sized patch, the number of possible values is 3^(m*n). Besides, the Laplace smoothing requires 3*k on the denominator. The accuracy of the model increased a little bit compared to that in part 1.2.

Disjoint patches size	Accuracy w/ binary feature	Accuracy w/ ternary feature
2*2	0.843	0.853

```
The overall accuracy is: 0.853
The confusion matrix is:
[[ 0.922  0.     0.     0.011  0.     0.     0.056  0.     0.011  0.    ]
 [ 0.     0.991  0.     0.     0.     0.     0.009  0.     0.     0.    ]
 [ 0.01   0.049  0.816  0.039  0.     0.     0.029  0.019  0.039  0.    ]
 [ 0.     0.01   0.     0.87   0.     0.02   0.02   0.04   0.     0.04 ]
 [ 0.     0.009  0.     0.     0.925  0.     0.019  0.009  0.     0.037]
 [ 0.022  0.011  0.     0.087  0.011  0.761  0.011  0.022  0.043  0.033]
 [ 0.022  0.022  0.     0.     0.033  0.033  0.89   0.     0.     0.    ]
 [ 0.     0.047  0.057  0.     0.009  0.009  0.     0.83   0.     0.047]
 [ 0.01   0.029  0.039  0.107  0.019  0.029  0.01   0.019  0.709  0.029]
 [ 0.01   0.01   0.     0.03   0.06   0.01   0.     0.05   0.02   0.81 ]]
```

Part 2: Audio Classification

Part 2.1:

Description:

In this part, we need to train a classifier to classify yes/no based on the spectrogram of their sound. In our implementation, the train and test sets are described as a list of lists of no/yes data, each described by a numpy matrix. In the preprocessing step, the matrix element is set to one if it is of high frequency, and zero otherwise. In training, we get the probability of one at every pixel location (i.e. 25*10) by taking the mean of the sum of all feature matrix of a given yes/no class. Besides, we also obtain the prior of each class. During testing, we calculate the log a posteriori probability (see next paragraph for the smoothing constant) of an instance to lie in each class, and then pick the class that maximizes the a posteriori probability.

To prevent taking the log of zero, we use Laplace smoothing by adding a small constant k on the numerator and 2*k (since the feature can take values 0 or 1) on the denominator when calculating probability of each class. The constant k is a hyper parameter that ranges from 0.1 to 10. As shown in the table below, the accuracy first increases and later decreases as k increases from 0.1 to 10, and attains the maximum at k = 3. Hence, to maximize overall accuracy, we choose k = 3.

k	0.1	0.2	0.3	0.4	0.5	1	2	3	5	10
Overall accuracy	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.97	0.96	0.94

```
The overall accuracy is: 0.97
The confusion matrix is:
[[ 0.96  0.04]
 [ 0.02  0.98]]
```

Part 2.2: (EXTRA CREDIT)

Description:

In this part, we need to train a classifier to classify digits 1-5 based on the Mel-frequency Cepstrum Coefficients (MFCC) of their sound. In our implementation, the train and test sets are described as a list of *word* objects, each described by a numpy matrix and a label showing digit 1-5. In the preprocessing step, the matrix element is set to one if it is higher than the threshold, and zero otherwise. In training, we get the probability of one at every pixel location (i.e. 30*13) by taking the mean of the sum of all feature matrix of a given digit class (5 in all). Besides, we also obtain the prior of each class. During testing, we calculate the log a posteriori probability (see next paragraph for the smoothing constant) of an instance to lie in each class, and then pick the class that maximizes the a posteriori probability.

To prevent taking the log of zero, we use Laplace smoothing by adding a small constant k on the numerator and 2*k (since the feature can take values 0 or 1) on the denominator when calculating probability of each class. The constant k is a hyper parameter that ranges from 0.1 to 10. As shown in the table below, the accuracy first increases and later decreases as k increases from 0.1 to 10, and attains the maximum between 0.3 and 5. Hence, to maximize overall accuracy, we choose k = 1 .

k	0.1	0.2	0.3	0.4	0.5	1	2	3	5	10
Overall accuracy	0.825	0.825	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.825

```
The overall accuracy is: 0.85
The confusion matrix is:
[[ 0.875  0.     0.     0.     0.125]
 [ 0.     1.     0.     0.     0.   ]
 [ 0.     0.     1.     0.     0.   ]
 [ 0.     0.375  0.     0.625  0.   ]
 [ 0.125  0.     0.125  0.     0.75 ]]
```

Statement of individual contribution

Jiaying Wu

- Odds ratio visualization
- Data recording for part 1 and 2
- Debugging for part 1

Qin Li

- Highest and lowest posterior probabilities pattern
- Running time for testing and training
- Part1 debugging and extra credit

Yuchen Liang

- Finish file I/O for part 1 and 2
- Design the data structure
- Implement training and testing algorithm with other group members