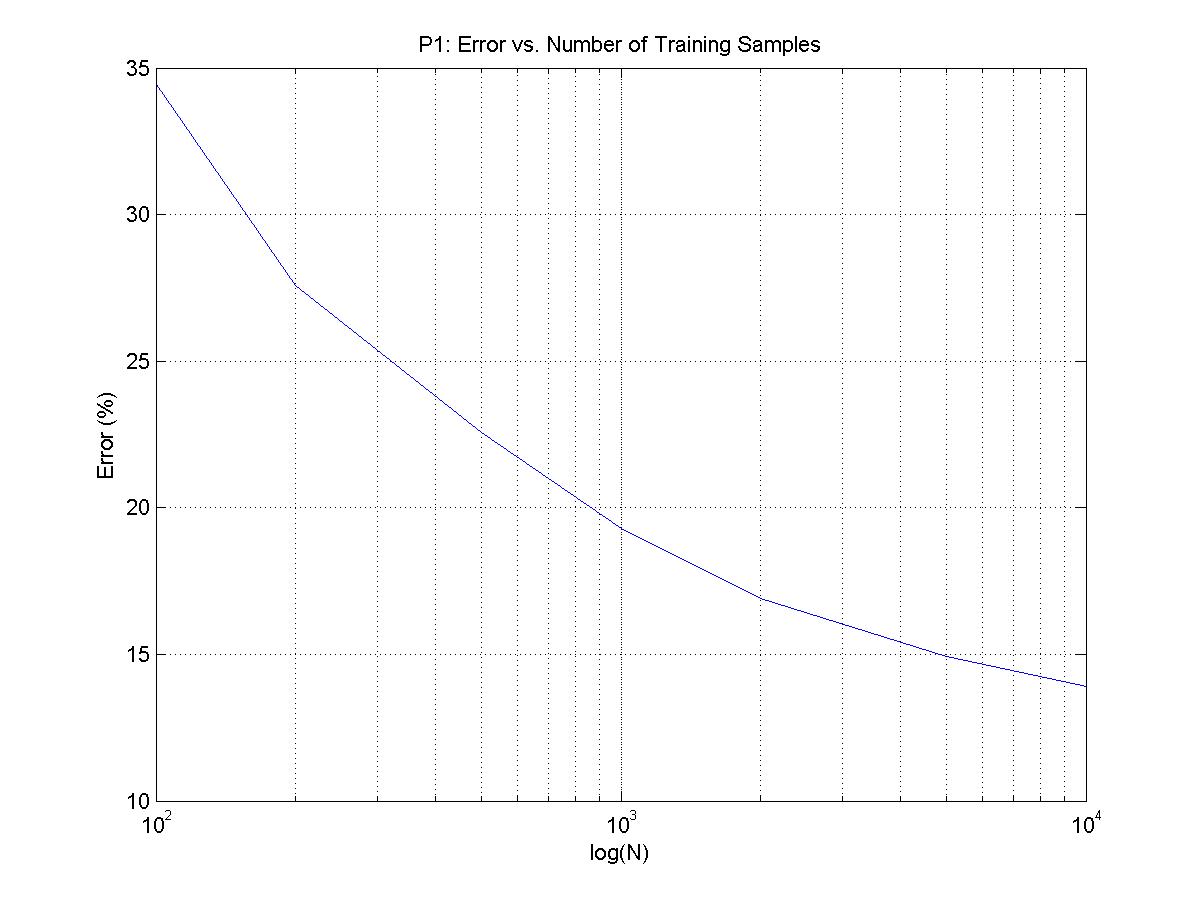
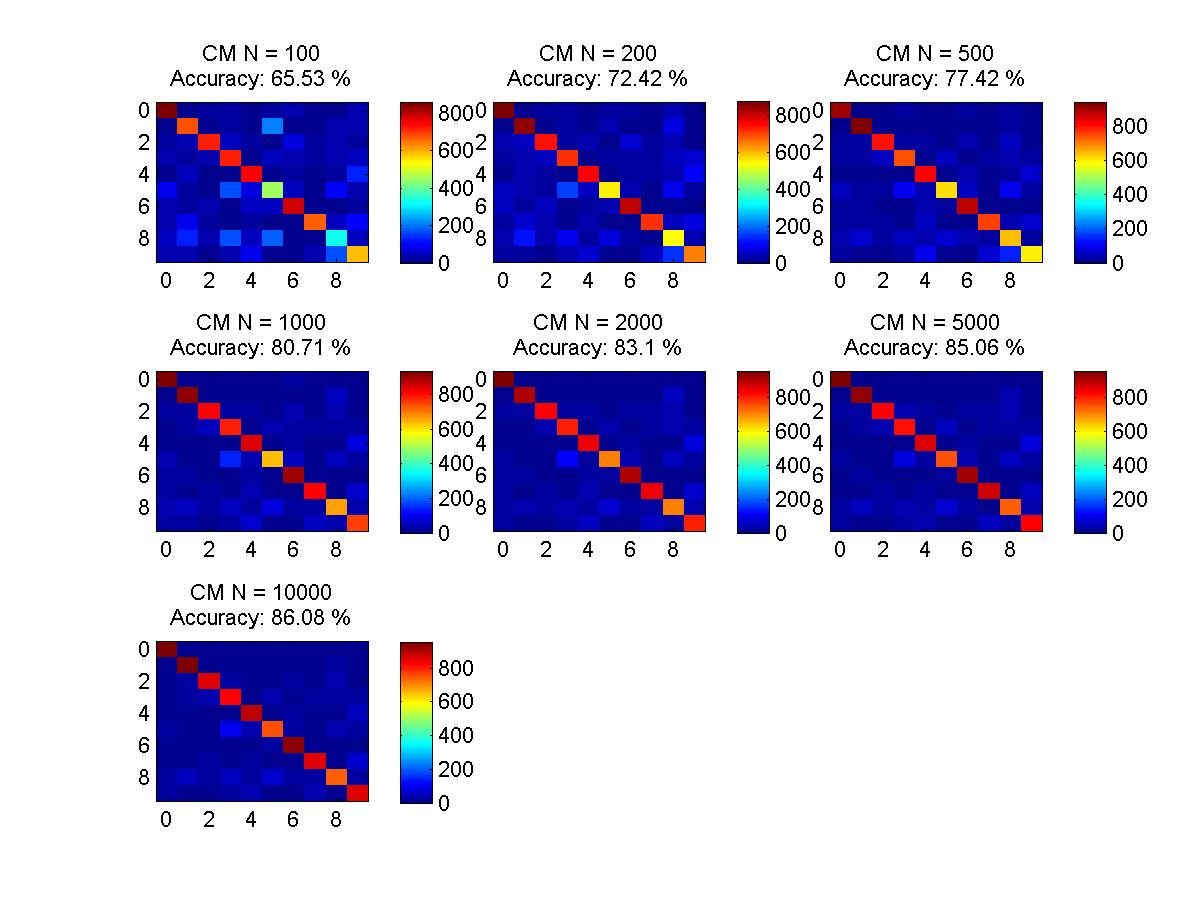
P1

Here we built a simple support vector machine (SVM) using the default SVM of LIBLINEAR. From the digits data, I evenly sampled 10k images for validation. Training data were sampled from the remaining 50k images. Below is a plot of the error rate vs. the number of training samples.



P2

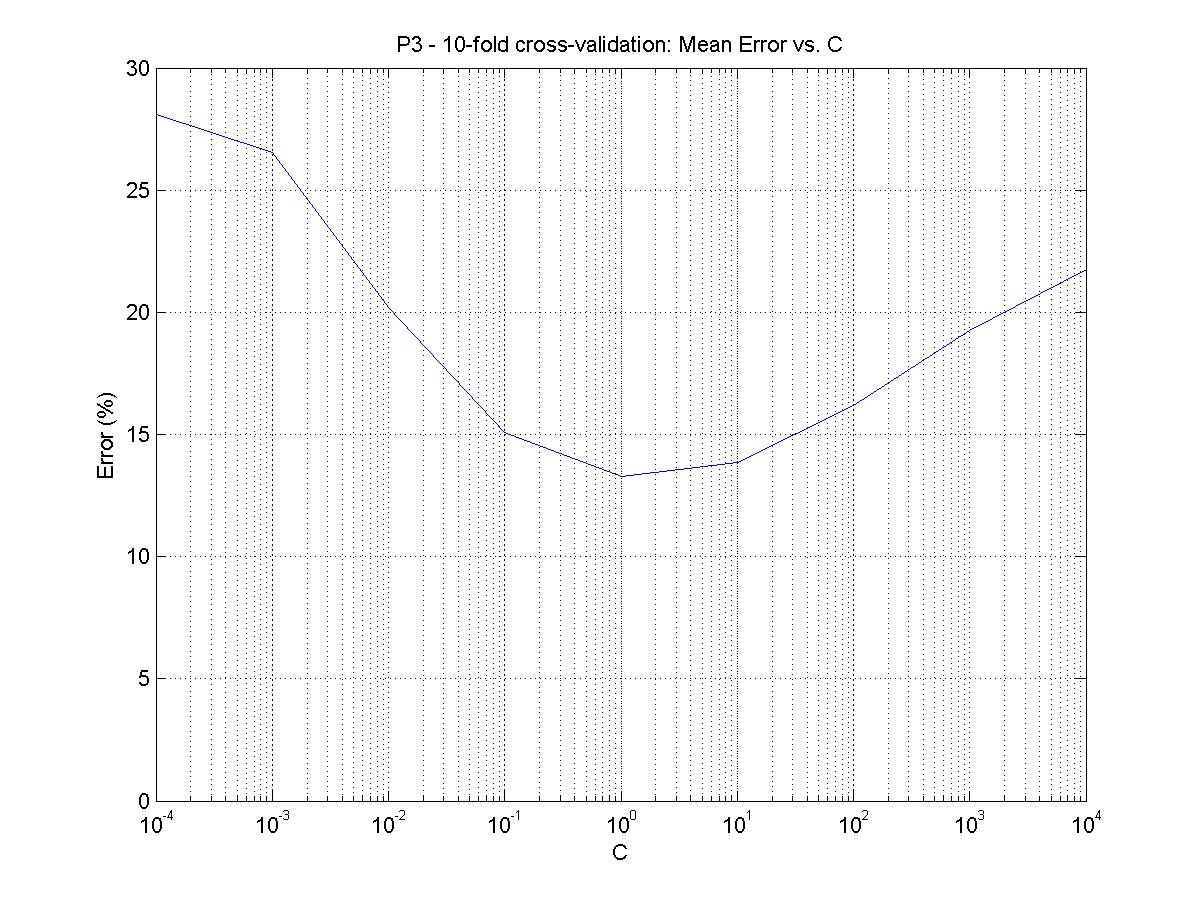
In this problem, we built confusion matrices for each of the experiments in P1. Below are the matrices. We can see that the biggest problems we have are misclassifying “5” for “2” and “8” for “9”. Confusion matrices can give us great intuition about misclassification, and could help us tune our models.



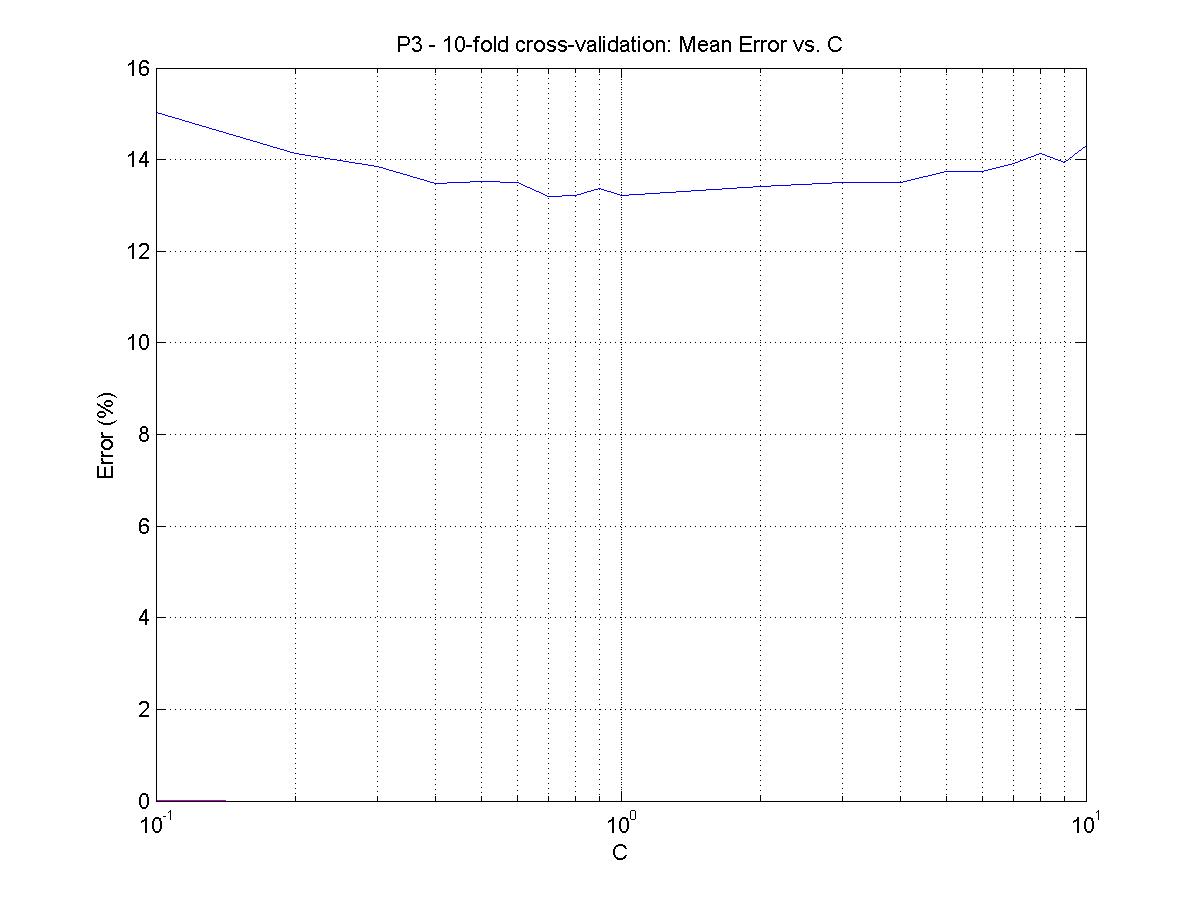
P3

Here we were tasked to use 10-fold cross-validation to optimize the parameter C of our model. Cross-validation randomly samples from a training data and splits in into k parts for each C parameter. Randomly sampling ensures that we are not biasing our sample, and validating k-fold times and computing the mean error makes the current C parameter being tested robust against outliers. helps in two ways: optimize for model parameters and

Initial Run: C = [.0001 .001 .01 .1 1 10 100 1e+3 1e+4]. We can see that the optimal value seems to be between .1 and 10.



Second Run: C = [.1 .2 .3 .4 .5 .6 .7 .8 .9 1 2 3 4 5 6 7 8 9 10]. Here the minimum C occurs at C = 0.7 (mean\_err = 13.2 %).



P4

I started with all features, and the average error and average confusion matrix were as follows:

