2a)

The best C is 0.01 giving 94.35% accuracy.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| C value | 0.001 | 0.01 | 0.1 | 0.5 | 1.0 | 2.0 | 5.0 | 10.0 | 100.0 |
| Test Accuracy | 91.95 | 94.45 | 93.25 | 93.0 | 92.7 | 92.7 | 92.7 | 92.7 | 92.7 |

2b)

We learned a model with all 2000 instances in groups2.train using C = 0.01. Then we classified all 2000 instances in groups2.test using that model. Reading off the results from svm\_classify, we received accuracies of:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | 1 | 2 | 3 | 4 |
| Train Accuracy | 96.85 | 95.85 | 96.75 | 97.65 |
| Test Accuracy | 94.75 | 93.05 | 94.80 | 94.95 |

2c)

Normalization should increase accuracy since it removes the problem where some feature vectors have much higher magnitude. After normalizing the feature vectors, the accuracy increased.

2d)

Slack variables measure the distance an instance goes past the margin. If you normalize the feature vectors then their distances from the margin are also normalized. This prevents the case where certain features vectors have much higher magnitudes than others because they have more features. The slack variables of each instance then give a normalized distance from the margin.

C controls how hard the margin is and the complexity of the classifier. A smaller C means a bigger and softer margin. If we normalize the instances then the margin is normalized too since the support vectors are normalized. Then, to optimize the margin, we have to reconsider C since it was set based on a different margin before normalization.