```
TestSetA
Fingerprint actual: 10  Predicted as 10
Fingerprint actual: 11  Predicted as 11
Fingerprint actual: 12  Predicted as 12
Fingerprint actual: 13  Predicted as 13
Fingerprint actual: 14  Predicted as 11
Fingerprint actual: 15  Predicted as 15
Fingerprint actual: 16  Predicted as 15
Fingerprint actual: 17  Predicted as 17
Fingerprint actual: 18  Predicted as 18
Fingerprint actual: 19  Predicted as 19
Fingerprint actual: 1  Predicted as 1
Fingerprint actual: 20  Predicted as 20
Fingerprint actual: 21  Predicted as 21
Fingerprint actual: 2  Predicted as 2
Fingerprint actual: 3  Predicted as 3
Fingerprint actual: 4  Predicted as 4
Fingerprint actual: 5  Predicted as 5
Fingerprint actual: 6  Predicted as 6
Fingerprint actual: 7  Predicted as 7
Fingerprint actual: 8  Predicted as 8
Fingerprint actual: 9  Predicted as 9
Model Accuracy: 90.48%
```

The performance I was able to achieve was higher than expected. I am a little bit suspicious that there is some overfitting going on with my classifier, but it seems to work well for the test set A. I am anxious to see how it performs on the set B. I started by trying to 'normalize' the images that we received. Reading the paper by Xuan Xu he points out that the darkness variance needs to be removed so that it is not a decision in the classification of an image and that it is simply noise. I followed the steps that he explained in how to do this. I also used StandardScaler to try to help normalize. I used the sklearn onevsRestClassifier with svm.SVC having its kernel set to rbf. I first ran it just base without altering any parameters and ended up around a 72% accuracy rate for the data. After some research on the parameters I decided to use a GridSearchCV to try to find the 'best' parameters for the highest accuracy. I started to run into Memory errors when I tried to do a range on C and Gamma and decided to stick with just C parameter changes for now. I think that this might be due to the default mem_cache of 200mb and have increase it but have yet to retest if I can get C and Gamma to run together. Running the GridSearchCV I end up getting around an 86% accuracy with a C=.01 for the 'best' parameter. I found this to be odd since increasing the C value will increase the penalty for misclassification. I ended up testing a range of values to end up finding that C=100 gives me a better Accuracy at 90.48%. I have been reading up and trying to figure out why the GridSearchCV would select the .01 and not the 100 for C since they are both included in the range of values I test. I think that it might come down to how the 'scoring' of the GridSearchCV decides what the 'best' parameters are and not just off accuracy.

I have tried to also crop the image yet have been unsuccessful in increasing my accuracy. Cropping the image tends to drop my accuracy by a decent amount. I have been playing with PCA and Gabor Filter along

with some other interesting solutions I have found from researching but have not had any luck increasing my accuracy over 90.48%