Homework 3
CS498daf, Spring 2016
Varun Somani, David Young and Cybelle Smith

For each of the classifiers below, the training was performed on all data from "pubfig_train_50000_pairs.txt". Where applicable and when time permitted, parameter tuning was performed using the "pubfig_kaggle_#.txt" files. The classification accuracy of the trained classifier is shown for the same training data, one example of the validation data, and the test/evaluation data. For the test accuracy, the value was reported by kaggle as we do not have access to the ground-truth labels.

Table 1: Accuracies for all classifiers.

Classifier	**Train Accuracy** (on pubfig_train_50000_pairs.txt)	**Validation Accuracy** (on pubfig_kaggle_1.txt)	**Test Accuracy** (on pubfig_kaggle_eval.txt, as reported by kaggle)
Linear SVM	0.77552	0.75915	0.7643
Naïve Bayes	0.752	0.74	0.74
Random Forests	1	NA	0.7652
Approximate Nearest Neighbor	1	NA	1
Radial SVM w/ Gamma 0.02	0.88064	0.76135	0.7675 (Highest Accuracy)
Radial SVM w/ Gamma 0.01	NA	0.76265	0.7674
K-Means clustering + Linear SVM	NA	0.7523	NA
Combined Voting of SVM, RBF SVM, NB & RandForest	NA	NA	0.7657

During the initial prototyping of various classifiers, a smaller subset of data was used (~20% -50% of the training dataset). This subset was split into training and testing portions for the various training procedures. Surprisingly, the results from these smaller training sets were better than the final results using the entire training set and the validation sets for testing. For example we obtained over 80% accuracy using an SVM on a smaller training portion split into training and test sets, but did not obtain 80% accuracy with any of our SVMs when training on all the data. This may be due to human bias in selectively reporting our results to ourselves. Prototyping on these smaller sets was necessary however, as re-training a classifier such as the RBF SVM on the entire data set for each of many gamma values would have taken a long time.

The key to an initial performance jump was the pre-processing of features. Each example was made up of two feature vectors, which we replaced by the difference between the two feature vectors (i.e. the scaled Euclidian distance between the two feature vectors in high dimensional space). A linear SVM worked surprisingly well given that the data intuitively seems to extend radially out from an ideal 0 distance.

We did not expect a linear SVM to do so well. Instead we believed an SVM using a radial-basis-function (RBF) would perform better, which it did. An RBF SVM is capable of choosing a non-linear decision boundary by mapping to a higher dimension feature space. But, although the RBF SVM yielded the highest accuracy, the performance gain was minimal and at the cost of significantly longer training time. Both versions of the SVM were tried a second time with appended features equal to the square of the pre-processed feature vector. This did not improve performance, and the results were not generated for the entire dataset so they are absent. We believe that these additional features were not as beneficial given our initial pre-processing.

The naïve bayes and random forest classifiers took a long time to train, but performed well. The naïve bayes did have the lowest final accuracy. The random forest classifier was nearly as accurate as the SVMs, with a training time somewhere in between the linear SVM and the RBF SVM.

The classification strategy for part 2 (comparison of labels from the lookup table) using nearest neighbors was guaranteed to yield 100% accuracy. The question was whether an approximate nearest neighbor package could yield nearly the same performance as a nearest neighbor package. The results would indicate yes. The training accuracy and the testing accuracy were both 100%. This would indicate the correct nearest neighbor was being accurately selected via the approximate algorithm, and significantly faster.

Lastly, we tried to implement the classifier described in lecture: a K-Means Clustering with a corresponding SVM for each cluster. But the performance here was right on par with the other SVMs. A voting system didn't help much, and we were unable to get a soft/fuzzy clustering package to work properly for our purposes. Neither approach yielded better performance than the RBF SVM.

We believe the RBF SVM worked well because the data was appropriately pre-processed and the RBF SVM somewhat subsumes both the linear and polynomial variants of an SVM classifier. Unfortunately the slightly better performance is not likely worth the much longer training time required by the RBF SVM.