Face Matching Report

Part 1

Part 1 consists of training three models, naive bayes, linear support vector machine, and random forest, to detect whether two faces are the same. The model that performed the best was the random forest model, whereas the naive bayes model and the linear support vector machine had an accuracy slightly over 50%. All algorithms use packages from python's scikit-learn.

For the Naive Bayes Model, I used the function sklearn.naive_bayes.GaussianNB(). This function is the only continuous input Naive Bayes model that sklearn has, but it assumes that the probability models are all normally distributed. I standardized the data before fitting it to the Naive Bayes model, and the training accuracy came out to be 52.62%, while the testing accuracy on the evaluation set was 51.84%.

For the linear support vector machine, I used the function sklearn.svm.LinearSVC(). I evaluated the model with different regularization parameters on the datasets pubfig_kaggle_1.txt and pubfig_kaggle_2.txt using a zero mean, one variance dataset from pubfig_train_50000_pairs.txt. I used pubfig_kaggle_3.txt as a test set for my model. I could have used grid search or randomized grid search but both method took too much time with a search as small as deciding between four possible regularization parameters: 0.1, 1, 10, 100. Using the validation sets on multiple runs, I found the best regularization parameter to be equal to 1.0. The model performed at an accuracy of 51.754% on the training set and 50.080% on the testing set. The accuracy tells us that the data is not linearly separable.

For the random forest model, I used randomized search cross validation with the function sklearn.grid_search.RandomizedSearchCV() and the function sklearn.ensemble.RandomForestClassifier(). Randomized search was done over the

parameters: max_features, min_sample_split, and min_samples_leaf over random samples from the range 1 to 12. The skearn documentation recommended to look for parameters between 1 to the square root of the feature vector length (which is about 12). The best parameters were: max_feature = 10, min_samples_leaf = 2, and min_samples_split =5. The accuracy of the classifier on training set was about 76.5% and the accuracy on the testing set was about 65.980%. I am not sure why overfitting has occurred since I evaluated the model on the pubfig_kaggle_*, which was not used for training or cross validation, and there was no signs of overfitting. Random forest performed better than the other two models because it did not assume that the feature was gaussian distributed or linearly separable.

	Training	Evaluation
Naive Bayes	52.62%	51.840%
Linear SVM	51.754%	50.080%
Random Forest	76.5%	65.980%

Part 2

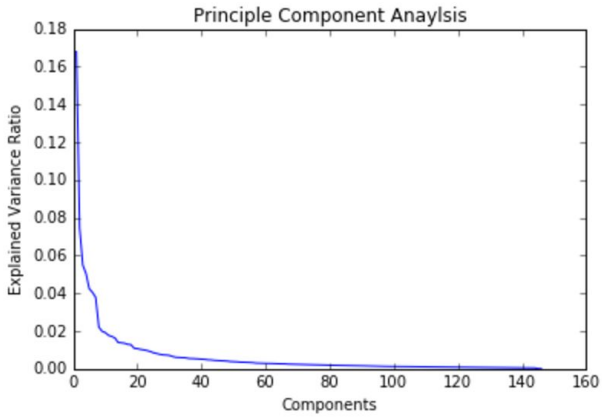
Part two consists of using nearest neighbor search to match each face attribute to a name and comparing whether the two names are equal. The implementation used in part two was scikit-learn's k nearest neighbors algorithm with n_neighbors equal two. Usually models

that use one neighbor to classify an example are prone to overfitting, but since we are not worried about this technique's performance on unseen celebrity faces not in the reference dataset, one nearest neighbors is the most appropriate. It should also be noted that mean centering and feature scaling was not employed for part two due to the fact that it might not find the correct nearest neighbors if all features have equal variance.

Training Accuracy	Evaluation Accuracy
100%	100%

Part 3

For part 3, I used a support vector machine with RBF kernel method. For the purposes of time, I could not run randomized search cross validation, grid search cross validation, or k-fold cross validation. I also tried to use techniques such as PCA and random sampling 60% of the training set in order to speed up the svm training process at the expense of reduced accuracy. I picked about 40 principal components since after 40 components the explained variance ratio of each component becomes less than 5%. The testing accuracy I found from this method is about 75.140% on the evaluation set.



Training Accuracy	Evaluation Accuracy
0.8388	75.140%