Homework 3

Statistics W4240: Data Mining
Columbia University, Fall 2015
Due Wednesday, October 21

For your .R submission, submit a file for each question labeled `hw03_q1.R`, and so on. The write up should be saved as a .pdf and under 8MB.

DO NOT submit .rar, .tar, .zip, .docx, or other file types.

Problem 1. (20 Points) James 3.7.3

Problem 2. (20 Points) James 3.7.4

Problem 3. (20 Points) Load the data set `hw03_q3.csv`.

a. (5 Points) Use the function `dist()` to produce a matrix of distances between all pairs of points. Distances should be computed for the two-dimensional input points $x = [x_1, x_2]$ (y is the output variable). Print the results.

b. (5 Points) Use the first data point as the testing set and the rest of the data as a training set. Implement kNN regression using the distance matrix from (a) for $k = 1, 2, \ldots, 10$. This algorithm should predict the y value of the first data point (with some error). Compute the mean squared error for the testing set and the mean squared error for the training set for each value of k; denote these values as $MSE^{k,1}_{test}$ and $MSE^{k,1}_{train}$.

c. (5 Points) Rerun part (b). For each data point: use the ith data point as a testing set, the remaining data as a training set, and run kNN for $k = 1, 2, \ldots, 10$ for observations $i = 2, 3, \ldots, n$. For each value of k compute a mean squared error as follows:

$$MSE^{k}_{train} = \frac{1}{n}\sum_{i=1}^{n} MSE^{k,i}_{train}$$

$$MSE^{k}_{test} = \frac{1}{n}\sum_{i=1}^{n} MSE^{k,i}_{test}.$$

d. (5 Points) The results from part (c) are called *leave one out cross-validation* error. They are commonly used for estimating prediction error and selecting model parameters. Use these results to pick the optimal value for k. Should you make your choice based on MSE^{k}_{train} or MSE^{k}_{test}, and why? What is the optimal choice of k, and why?

Problem 4. (35 Points) In this problem, we will use 1NN classification and PCA to do facial recognition.

a. (5 Points) Load the views `P00A+000E+00`, `P00A+005E+10`, `P00A+005E-10`, and `P00A+010E+00` for all subjects in the CroppedYale directory. Convert each photo to a *vector*; store the collection as a matrix where each row is a photo. Give this matrix the name `face_matrix_6a`. Record the subject number and view of each row of `face_matrix_6a` in a data frame. The subject numbers will be used as our data labels.

Use the following commands to divide the data into training and testing sets:

```
fm_6a_size = dim(face_matrix_6a)
# Use 4/5 of the data for training, 1/5 for testing
ntrain_6a = floor(fm_6a_size[1]*4/5)
ntest_6a = fm_6a_size[1]-ntrain_6a
set.seed(1)
ind_train_6a = sample(1:fm_6a_size[1],ntrain_6a)
ind_test_6a = c(1:fm_6a_size[1])[-ind_train_6a]
```

Here `ind_train_6a` is the set of indices for the training data and `ind_test_6a` is the set of indices for the testing data. What are the first 5 files (rows) in the training set? What are the first 5 files in the testing set? Specify their subject and view indices.

b. (5 Points) Do PCA on your training set and use the first 25 scores to represent your data. Specifically, create the mean face from the training set, subtract off the mean face, and run `prcomp()` on the resulting image matrix. Project your testing data onto the first 25 loadings so that it is also represented by the first 25 scores. Do not rescale the scores. Use 1NN classification in the space of the first 25 scores to identify the subject for each testing observation. In class we discussed doing kNN classification by majority vote of the neighbors; in the 1NN case, there is simply one vote. How many subjects are identified correctly? How many incorrectly? Plot any subject photos that are misidentified next to the 1NN photo prediction.

c. (10 Points) Rerun parts (a) and (b) using the views `P00A-035E+15`, `P00A-050E+00`, `P00A+035E+15`, and `P00A+050E+00` for all subjects in the CroppedYale directory. Give this matrix the name `face_matrix_6c`. For each image, record the subject number and view in a data frame. Use the following commands to divide the data into training and testing sets:

```
fm_6c_size = dim(face_matrix_6c)
# Use 4/5 of the data for training, 1/5 for testing
ntrain_6c = floor(fm_6c_size[1]*4/5)
ntest_6c = fm_6c_size[1]-ntrain_6c
set.seed(2)
ind_train_6c = sample(1:fm_6c_size[1],ntrain_6c)
ind_test_6c = c(1:fm_6c_size[1])[-ind_train_6c]
```

Do PCA on your training set and use the first 25 scores to represent your data. Project your testing data onto the first 25 loadings so that it is also represented by the first 25 scores. Use 1NN in the space of the first 25 scores to identify the subject for each testing observation. Do not rescale the scores. How many subjects are identified correctly? How many incorrectly? Plot any subject photos that are misidentified next to the 1NN photo prediction.

d. (5 Points) Rerun part (c) with 10 different training and testing divides. Display the number of faces correctly identified and the number incorrectly identified for each. What do these numbers tell us?

e. (10 Points) Compare the results for parts (b) and (c). Are the testing error rates different? Observe that the views in (a) are closer to each other than those in (c), where the latter has much wider lighting ranges. What does this tell you about PCA? In general, when does PCA work better?

Problem 5. (20 Points) James 4.7.4

Problem 6. (20 Points) James 4.7.6

Problem 7. (20 Points) James 4.7.8