STAT-5504 Multivariate Statistical Methods
Take-home Final Project

Linjun Li, Department of Statistics, Virginia Tech

Task 1: Multi-response Regression of Image Pixels

Consider the pixels of images with "happy" expression as responses (Y), and pixels of images with "wink", "surprised" expressions as predictors (X). Perform a multi-response regression and conduct inferences (i.e., parameter estimation accuracy, prediction accuracy, prediction interval, hypothesis testing) for the estimated model.

From reading the RGB data of the .gif images, it is found that each image is consisted of 243×320 pixels. To form the multi-response matrix Y, I picked some important points in each of the "happy" images where different facial expressions will lead to different RGB values. See Figure 1 for the points that are picked.

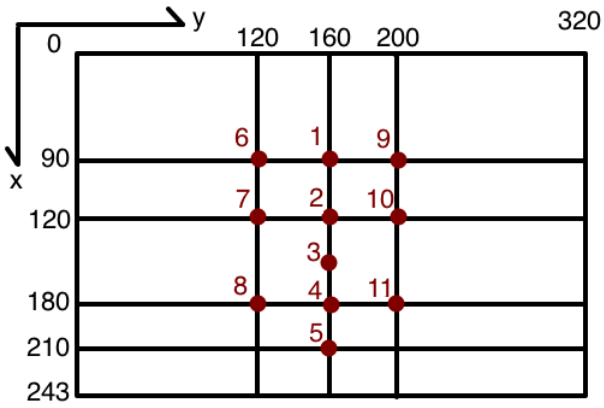

Figure 1: Plot of the achieved coverage of the Wald interval determined from Monte Carlo simulation.

Thus, for each of the 15 "happy" figures we can obtain 11 data points whose RGB values (range from 0 to 255) as the multi-response. Here we denote the new matrix with reduced dimension as Y_n. (The 15×11 Y_n matrix can be found in the comment section of the code.) To form the X matrix, we start with the "wink" figures. Each row of RGB values of a "wink" figure are flattened so that for each figure we have one array with 243×320(=77760) RGB values. In the same way, we can obtain the arrays (with 77760 elements) for the "surprised" figures. For the same person I concatenate the "wink" array and "surprised" array so that each row of the matrix will have 155520 elements. With data of 15 persons, we can obtain a X matrix with a shape

of 15×155520. In the X matrix, a lot of covariates are highly correlated, such as pixels at the upper-left and upper-right conners, their values are almost all 255. To avoid the problem collinearity during regression, PCA is applied. The scree plot of the PCA look as follows:

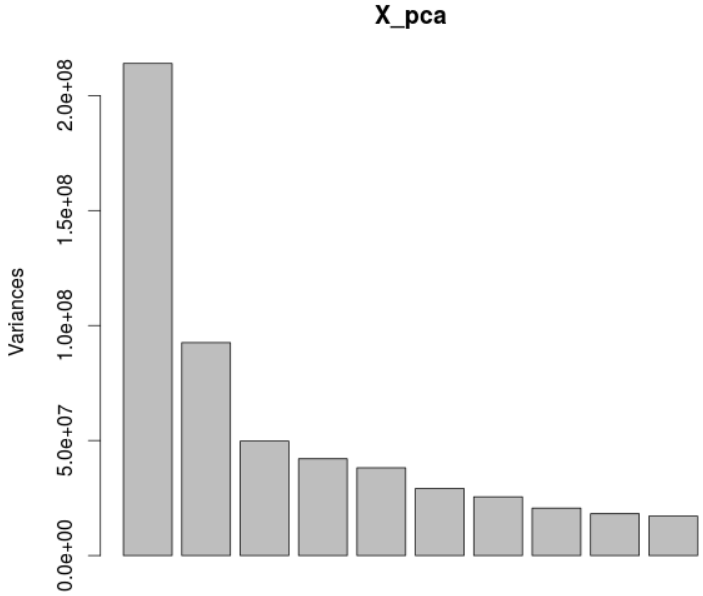

Figure 2: Scree plot of PCA result. The first ten principal component explains 91.68% of the variance.

The detailed result of the PCA calculation can be found in the comment section of the code. It is can be found that the first 15 principal components have the proportion of variance explained as $0.3585, 0.1551, 0.08336, 0.0705$, $0.06393, 0.04883$, 0.04277, 0.03458, 0.03048, 0.02874, 0.02408, $0.0227, 0.01859, 0.01786, 0.00000$. Also, from the data of cumulative proportions, we can find more than 90% of variance are explained when the principal component number reaches 10. Thus I choose the first 10 principal components of the PCA transformed data as the new predictor variables (which form X_n). Here X_n is a matrix of size 15×10.

With Y_n and X_n multi-response regression is performed using R. From the output of the $lm(.)$ routine in R we can draw the regression result for each of the 11 variables in Y_n. For example, we can pick point 5 (see Figure 1), which represents a point on the chin if the person does not open his/her mouth. In the regression result for point 5, we find the R^2 is 0.9479, which means the model with the estimated $\hat{B}$ explain 94.79% of the variation of the fifth response variable in Y_n. The P-value for the F statistics test is 0.03535, meaning the model is acceptable for explaining the variations in the 5th column of Y_n. Also, we can find that in addition to the parameter estimation of the intercept, the parameter estimation of the 4th and 8th principal components

are also significant. The standard error for these three parameters are 4.7518907, 0.0007582 and 0.0010827 respectively. The standard error of residual is 18.4.

Given an arbitrary x_{new} data, we want to predict the corresponding y value on the point 5. This can be done by calculating $x'_{new}\hat{B}$, the prediction for the 11 Y values given a certain x'_{new} (see the comment section in code) is (176.3837, 84.75026, 129.8396, 59.86544, 86.72901, 69.55481, 100.2671, 132.6298, 107.7541, 70.61056, 45.55909). The standard error of prediction for each element of Y requires the knowledge of the standard error of residual for that element of Y. For example $\sigma_5 = 18.4$ for the fifth element.

$$\sigma_5* = \sigma_5\sqrt{x'_{new}(X'_{n1}X_{n1})^{-1}x_{new} + 1}, X_{n1} = [\mathbf{1}, X_n] \quad (1)$$

The σ_5* is found to be 19.82849 in our case. Thus, the prediction interval can be estimated as

$$PI_5 = [86.729 - 19.828 \times t_{4,0.975}, 86.729 + 19.828 \times t_{4,0.975}] = [31.676, 141.782] \quad (2)$$

The range of PI_5 looks a little large. However, a color of dark grey to grey is predicted, instead of plain white (255) or some other extremely light grey color (200 to 255).

In addition, tests of models basing on the 'Wilks' statistics are performed. It shows that models with the first one, first two or first three columns of X_n (in the direction of principal components) are valid for explaining the variations in Y_n. Their P-values for the F-statistics are 0.02996, 0.02633 and 0.04462. The hypothesis test won't work for models with even larger number of covariates for lack of degrees of freedom for the residuals. (See test results in code) If we can include more images of some other people (larger n) in our image data set, we will be able to do a hypothesis test for the model with 10 covariates.

Task 2: Facial Expression Classification

Using the subset of data with "sad", "happy", and "wink" expressions, conduct a three-class classification. Please evaluate the classification accuracy of your method.

Before classification, we want to build the 15×77760 matrix for the 15 "sad" images with the pixels in each of the image. This can be done by lining up the 243 rows of the pixel data of each image into one row vector and concatenate such vectors for all the "sad" images by rows. In the same way, we build the 15×77760 matrices for the "happy" images and "wink" images. Then, we concatenate these three 15×77760 matrices by rows and form a 45×77760 matrix. Between each pair of the 45 records (data points) in this matrix, "Bray Curtis" distances are calculated. With the distances between the 45 points in 77760 dimensions, we have a network in high dimensions. Using IsoMap we can simplify this network so that the connections of only the nearest 3 neighbors of a data point can be retained. Also I set the parameter "ndim" to be 10 so that this network will be projected into a space of 10 dimensions using Multi-dimensional Scaling. The two dimensional projection of the 10-dimensional map is presented in Figure 3.

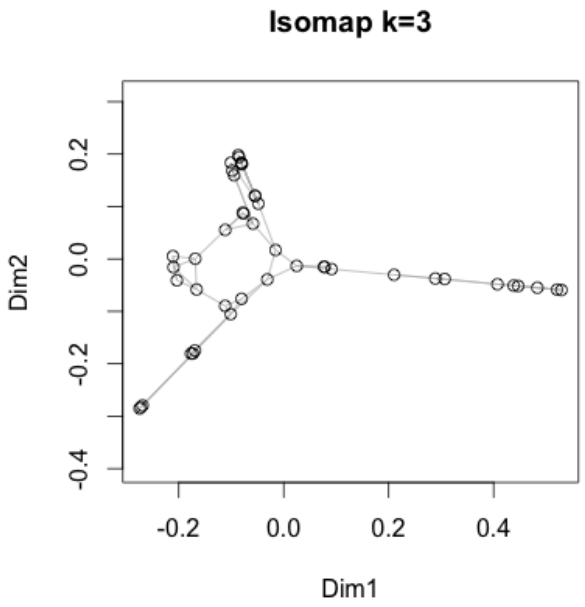

Figure 3: Two dimensional projection of the ten-dimensional IsoMap for the 45 data points, the distance metric used in the original space is Bray Curtis distance.

With the data in the 10-dimensional map, we have the 45×10 data matrix. Now we can zip the 45×10 matrix with the vector (1,1,1,1,1,1,1,1,1,1,1,1,1,1,1, 2,2,2,2,2,2,2,2,2, 2,2,2,2,2,2,2,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3)' by column. Here "1" is the label for "sad" images, "2" is the label for "happy" images and, "3" is the label for "wink" images. With the 45×11 matrix we shuffle the records/rows. Within the shuffled 45×11 matrix, the first 30 rows are chosen as training set and the last 15 rows are chosen as testing set.

In the 30×11 training set, the data points are sparsely spaced in feature space, thus I used SVM to do the classification. After trying different kernels (such as "linear" kernel, the "polynomial" kernel, the "radial" kernel and the "sigmoid" kernel), I found the "sigmoid" kernel gives the best performance. Then I used 5-fold cross validation to tune the best misclassification penalty parameter (the 'cost' parameter). Plugging the best parameter directly into the SVM routine we can obtain the optimized fitted model "svmModel". Using "svmModel" we are able to make predictions for the testing data, which is a 15×10 matrix. The prediction result is as follows:

$$3, 2, 2, 2, 2, 3, 3, 1, 2, 1, 3, 3, 2, 3, 2 \tag{3}$$

and their real labels are:

$$3, 1, 2, 3, 3, 3, 2, 1, 1, 2, 3, 3, 2, 1, 2 \tag{4}$$

The misclassification rate for the testing data is 0.4666667, which is considerably larger than expected. One possible reason for the large error rate is the author's lack of experience of finding the optimal kernel; another possible reason for the large error

rate is the lack of training data; a third possible reason is that I didn't find a better distance metric to build a distance/dissimilarity matrix which can better represent the distinctions between the images.

Task 3: Image Clustering

Using all the image data, conduct a clustering method to group the data. Evaluate the performance of your clustering method.

Before clustering the images I first performed PCA and reduced the dimension for each image from 77760 into 6. The detailed procedures and results of PCA can be found in Task 4. As can be found in Figure 8, the structure of the data is complicated.

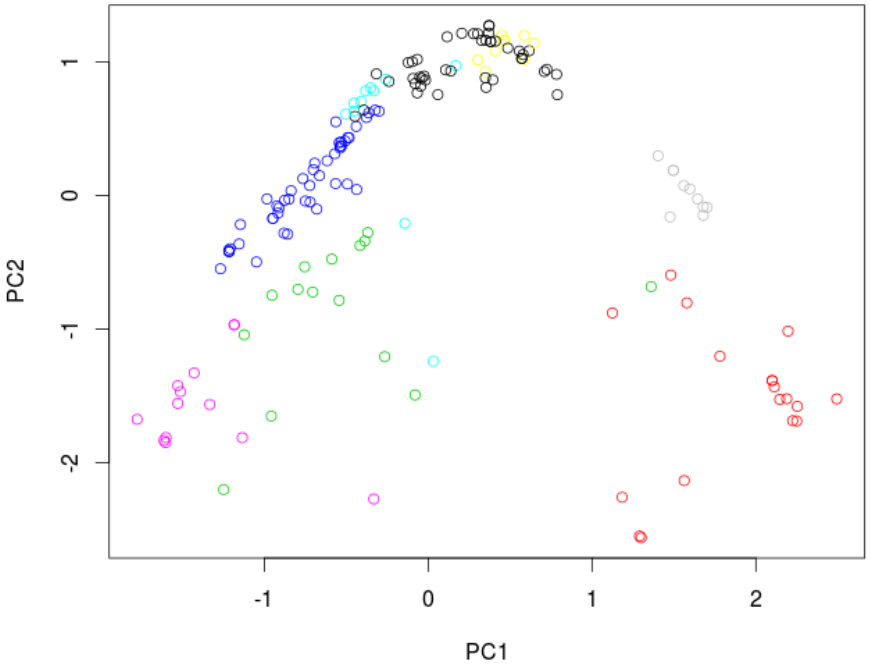

Figure 4: Clustering result of Spectral Clustering as displayed in a scatter plot in the dimensions of PC1 and PC2, after scaling the data of six dimensions.

Zooming in one of the plots in Figure 8, here Figure 4 displays the scaled data in the dimension of the first two principal components. Since the outline of the data in this plot is "V" shaped, we can imagine that the clustering algorithm which requires convex-shaped outlines of the data (such as K-means) will not work very well. However, the algorithm of Spectral Clustering, which is dedicated to cluster non-convex data, can be used for the current task. Since there are 11 expressions for each of the 15 persons, I set the total number of clusters to be 11. The result of Spectral Clustering are represented by the different colors in Figure 4. (The detailed cluster membership by Spectral Clustering can be found in the comment section of the code.) The Within Sum of Squares for each cluster is as follows: 53.28955, 181.07577,

68.17774, 180.11255, 96.37523, 143.88464, 43.08548, 146.86978, 167.48242, 169.36351, 190.01196. And the total Within Sum of Squares is 1439.729.

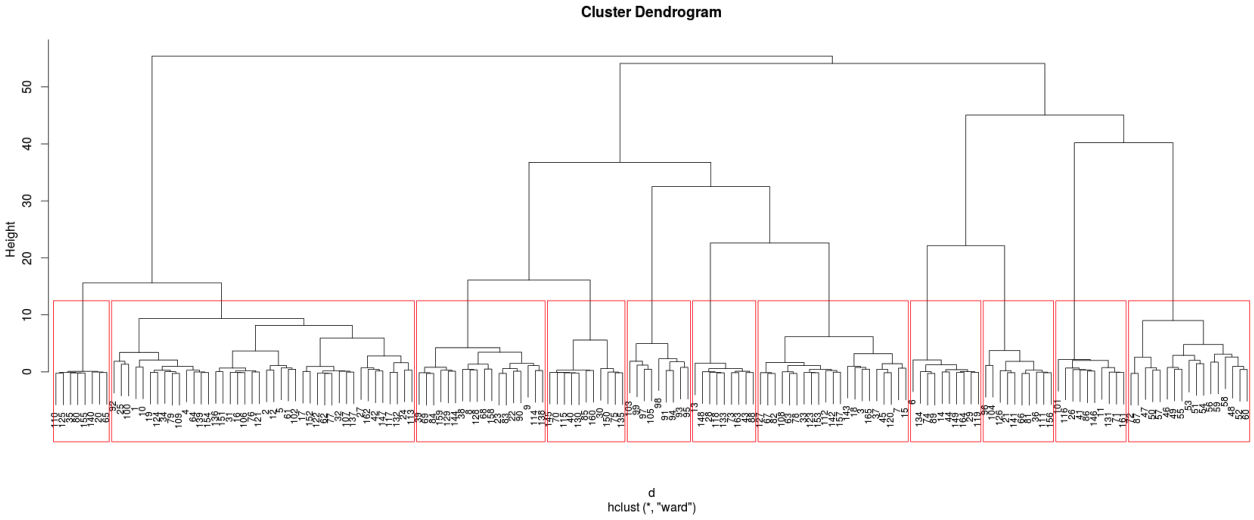

Figure 5: Dendrogram of the Agglomerative Clustering algorithm with Ward's Method.

In addition to Spectral Clustering, Agglomerative Clustering with Ward's minimum variance method is performed, also with the reduced data from PCA in Task 4. As have been done for Spectral Clustering, the data in 6 dimensions (the dimensions of the first 6 principal components) are also scaled. This algorithm starts with 165 clusters (the number of data points) and merge two clusters into a bigger one if the increase of total Within Sum of Squares is the smallest in that iteration. The distance in this clustering method is set to be Euclidean. The dendrogram of Agglomerative Clustering can be found in Figure 5. (The detailed cluster membership by this clustering algorithm can be found in the comment section of the code.) The Within Sums of Squares in each cluster is: 163.3609 55.84344 146.8698 97.13051 190.012 100.9495 56.63406 169.3635 114.8533 225.2949 196.288. And the total Within Sums of Squares is 1516.6, which is a little larger than that of the Spectral Clustering algorithm. This is because Agglomerative Clustering with Ward's Method is a greedy algorithm, which prefers convex-shaped clusters.

As a result, the performance of Spectral Clustering beats that of the Agglomerative Clustering with Ward's Method.

Task 4: PCA of the Image Data

Using all the image data, perform the principal component analysis and interpret your result properly.

In the same way as described at the beginning of Task 2, we can build the 15×77760 matrices for each of the center-light, glasses, happy, left-light, no glasses, normal, right-light, sad, sleepy, surprised, and wink expressions (eleven in total).

Then we concatenate the eleven 15×77760 matrices (by rows) in to a 165×77760 matrix. From the image data set we can see each pixel value ranges from 0 to 255,

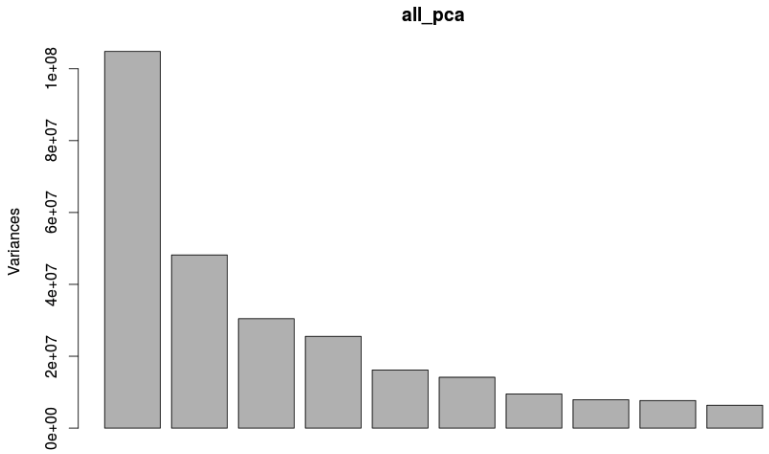

Figure 6: Scree plot of PCA result with all the images in the "Yale Face" folder. The first ten principal component explains 78.89% of the variance.

indicating black to white. This means the RGB values of each pixel vary in the same range. Also, we can observe that there are locations in the pictures where all the 165 images show pure white color (RGB=255). With all the 165 images having the same value of RGB in a specific pixel, we obtain a variance of 0. In this case, the data scaling is not applicable since 0 will appear in denominator of the calculation. Data scaling is neither encouraged because it will change the "shape" of the data and thus change the result of PCA. For both reasons as mentioned above, I perform PCA with R without scaling the data in the feature space. (It should be noticed that the PCA routine in R default the center parameter to be TRUE and the scale parameter to be FALSE, so the centering procedure is automatically performed.) The scree plot of the PCA can be found in Figure 6. The Cumulative Proportion of Variance can be found in Figure 7.

From the PCA result (can be found in the comment section of the code), we can see the first principal component explains 30.54% of the variance in the rotated feature space, where the covariance matrix of the data is diagonal, with the diagonal elements ordered from large to small. By adding the second principal component, which is uncorrelated to the first component, additional 14.04% of variance is explained. If we include the first 6 principal components (which are uncorrelated with each other), 69.73%≈70% of the variances can be explained/represented. In Task 3, I used the dimensions of the first 6 principal components to represent the complete image data (in 77760 dimensions) for clustering.

In Figure 8, the scatter plots of the data described by the first 4 principal components are shown in pairs. It can be found from the plots that the outlines of the data are neither linear or oval. What can be observed include the "V" shaped outlines, the "Ω" shaped outline, the arrow shaped outline and the "S" shaped outlines. This

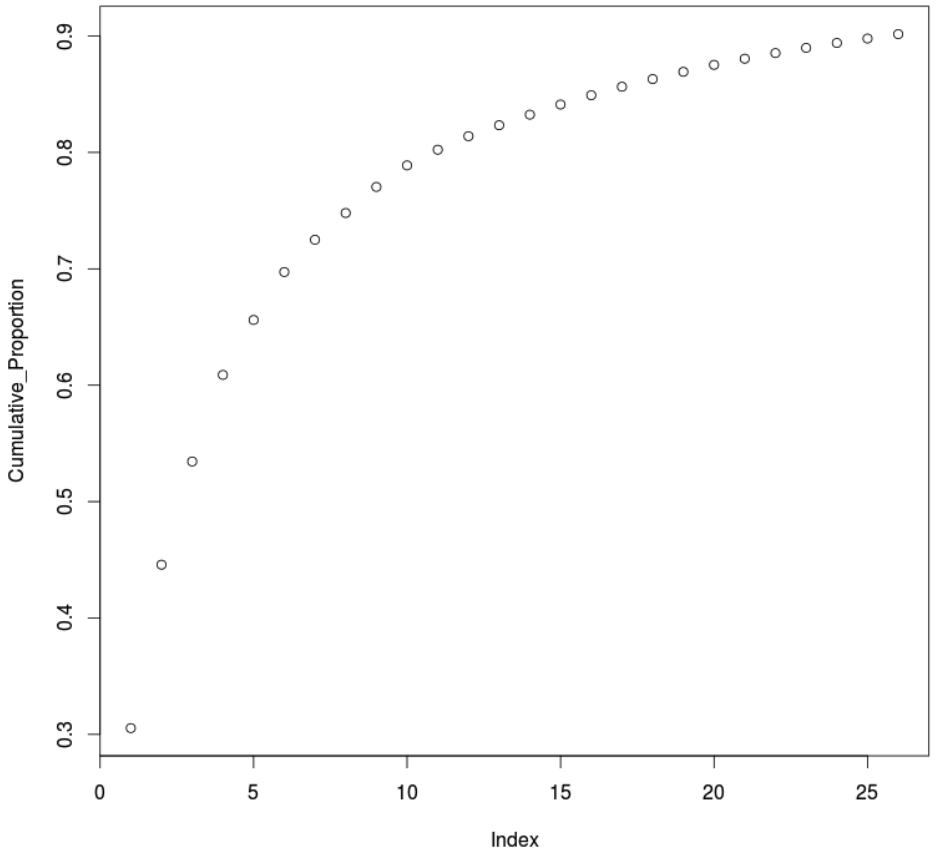

Figure 7: Cumulative Proportion of Variance of the principal components, from PC1 to PC26.

means the structure of the data is complicated, a problem cannot be resolved by simple rotation, translation or scaling of the data.

Task 5: CCA of the Image Data

Suppose one group of data is the images with "sleepy" expression, and the other group data is the images with "happy". Perform the canonical correlation analysis for these two groups of data and explain the results appropriately.

Canonical Correlation Analysis aims to find the basis vectors $\mathbf{a}$ and $\mathbf{b}$ such that when data set X and Y are projected to them respectively, the correlation (between the vectors $U_1 = \mathbf{a_1}'X$ and $V_1 = \mathbf{b_1}'Y$) become the largest. When the first pair U_1, V_1 are found, the second pair of U_2, V_2 are to be found for largest correlation, under the condition that $\text{cov}(U_1, U_2)$=0 and $\text{cov}(V_1, V_2)$=0, etc.

To tackle with this task, PCA is performed for the 15×77760 "happy" matrix. According to the scree plot out of PCA, the values in the dimensions of the first 6 principal components are retained to represent the complete "happy" dataset, see Figure 9. Same procedures have been done for the "sleepy" dataset.

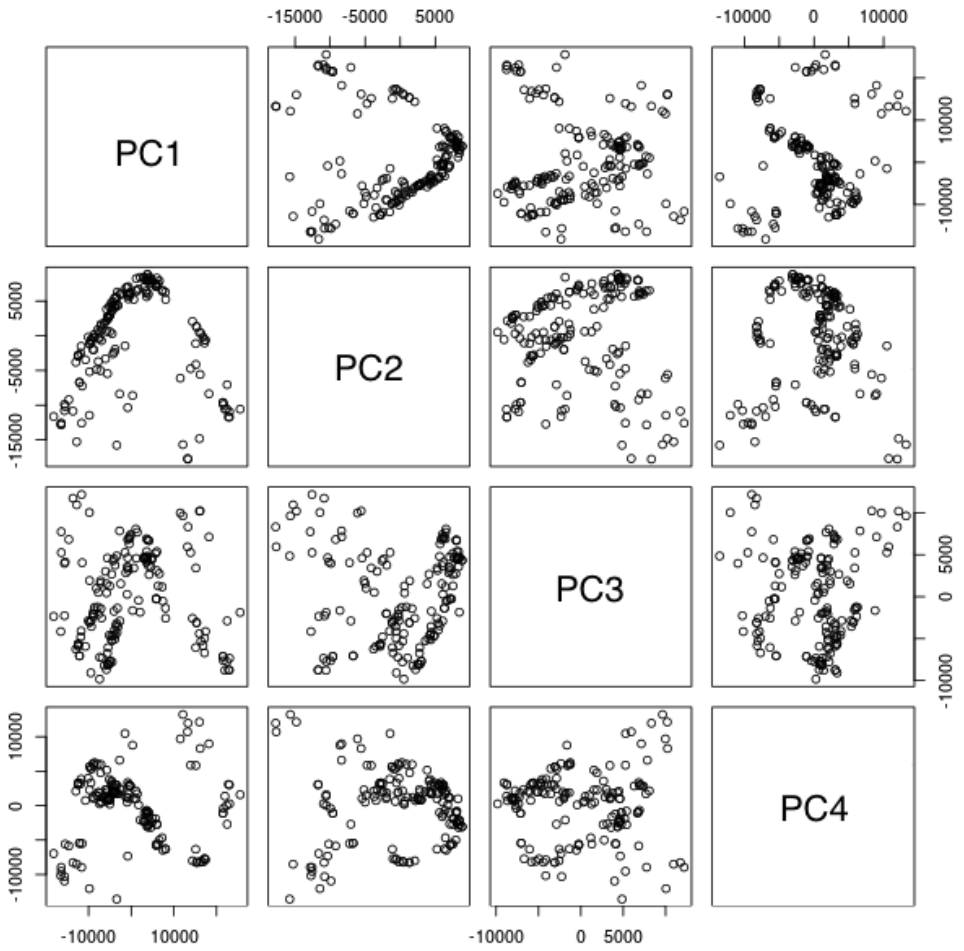

Figure 8: Scattering plots of the image data in the directions of the first four principal components.

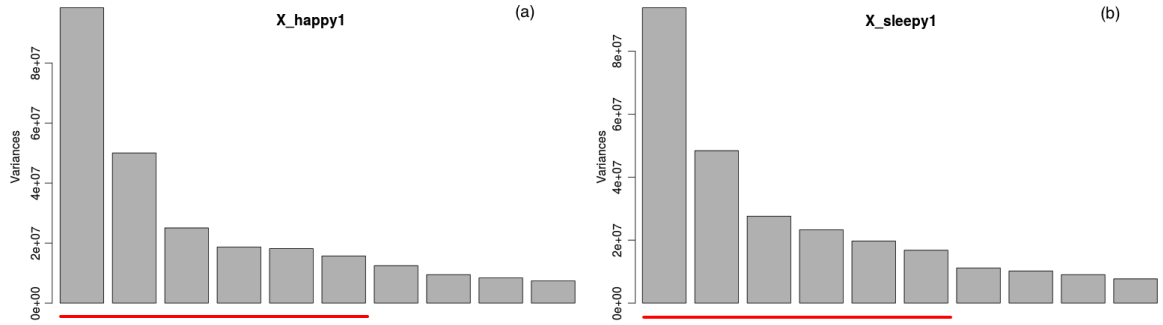

Figure 9: (a) Scree plot of the "happy" dataset. (b) Scree plot of the "sleepy" dataset.

After performing the Canonical Correlation Analysis in R, we obtained the coef-

ficient matrix $A = (\mathbf{a_1}, \mathbf{a_2}, \mathbf{a_3}, \mathbf{a_4}, \mathbf{a_5}, \mathbf{a_6})$ and $B = (\mathbf{b_1}, \mathbf{b_2}, \mathbf{b_3}, \mathbf{b_4}, \mathbf{b_5}, \mathbf{b_6})$ as:

$$A = \begin{bmatrix} -9.02e-05 & 2.34e-05 & -1.20e-05 & -1.43e-05 & -3.08e-05 & 1.37e-05 \\ 3.02e-05 & 6.61e-05 & 1.44e-05 & -1.18e-04 & 1.77e-05 & 1.55e-05 \\ -3.75e-05 & -1.57e-04 & -3.92e-05 & -9.97e-05 & 4.04e-05 & -2.67e-05 \\ -1.03e-05 & -4.57e-05 & 6.74e-05 & 1.57e-05 & 5.46e-05 & 2.09e-04 \\ -9.59e-06 & -4.81e-05 & 2.01e-04 & -2.57e-05 & -9.66e-05 & -4.87e-05 \\ -8.56e-05 & 4.26e-05 & 8.74e-05 & 3.23e-05 & 1.99e-04 & -7.78e-05 \end{bmatrix} \tag{5}$$

and

$$B = \begin{bmatrix} -9.13e-05 & 2.69e-05 & -5.75e-06 & -2.58e-05 & -2.39e-05 & 1.81e-05 \\ -4.07e-05 & -4.37e-05 & 2.33e-05 & 1.24e-04 & -3.58e-05 & -3.83e-06 \\ -3.21e-05 & -1.44e-04 & 3.94e-05 & -4.77e-05 & 6.44e-05 & 8.10e-05 \\ -5.87e-05 & -5.92e-05 & -1.86e-05 & -2.63e-05 & 5.52e-05 & -1.79e-04 \\ 1.48e-05 & -7.76e-05 & -1.96e-04 & -6.73e-06 & -7.50e-05 & 1.91e-05 \\ 3.84e-05 & -6.42e-05 & 9.82e-05 & -8.43e-05 & -1.87e-04 & -4.71e-05 \end{bmatrix} \tag{6}$$

(The more accurate values for A and B can be found in the comment section of the code.) Then we can calculate the $U = XA$ and $V = YB$ and have a look at the correlation between U and V, which is:

$$corr(U, V) = \begin{bmatrix} 9.97e-01 & -8.17e-17 & -1.02e-16 & 1.42e-16 & 1.37e-16 & 1.98e-16 \\ 5.79e-17 & 9.95e-01 & -5.36e-16 & -1.31e-16 & -1.37e-16 & 1.80e-16 \\ 4.11e-17 & -1.68e-16 & 9.78e-01 & 1.59e-17 & -4.62e-17 & 3.57e-17 \\ -1.21e-16 & 2.08e-16 & 5.34e-17 & 9.71e-01 & 3.22e-16 & -1.86e-16 \\ 2.35e-16 & 2.92e-16 & 1.21e-16 & 1.31e-16 & 8.91e-01 & 1.04e-17 \\ -1.09e-17 & 3.03e-16 & -6.32e-17 & 3.73e-16 & 3.33e-17 & 6.94e-01 \end{bmatrix} \tag{7}$$

with pair-wise correlations $corr(U_i, V_i)$ =0.9969858, 0.9954081, 0.9776235, 0.9705297, 0.8906162 and 0.6941896 for $i = 1, 2, ..., 6$. It is not hard to find, for $i = 1, 2, 3, 4$, the (U_i, V_i) pairs are highly correlated, having the correlation values bigger than 0.97. For $i = 5, 6$, the correlation between U_i and V_i are still large and non-negligible. This means, after transforming the reduced "happy" and "sleepy" datasets, we obtain two new datasets U and V. The columns of each of U and V are uncorrelated, yet the corresponding columns between U and V are highly correlated (the matrices U and V should be almost the same if the columns of both matrices are standardized).

This means, the reduced "happy" and "sleepy" datasets can be transformed by CCA such that we can match the images from the two sets very easily (by calculating the inner products of the records in the transformed datasets). Detailed speaking, if we partition the "happy" and "sleepy" datasets into training and testing datasets, we can use the training sets to find A and B. With the trained A and B, we can transform the testing sets. Given a transformed record from the "happy" testing set, we can retrieve the same person's "sleepy" record by simply calculating the inner products between itself and the transformed records in the "sleepy" testing set. The

record of the largest inner product should be the matched "sleepy" image, which should be of the same person.