Eigen faces of MSAN faculty and students

Nishan Madawanarachchi

5/6/2018

Objective

Identify the four "eigen-faces" of the MSDS students and MSDS faculty and then identify similarities between the images of students and faculty.

Extracting data

Images of the students and the faculty were extracted from the MSDS website.
1. Student - `wget -K -p https://www.usfca.edu/arts-sciences/graduate-programs/data-science/our-students`
2. Faculty - `wget -K -p https://www.usfca.edu/arts-sciences/graduate-programs/data-science/faculty`

Sample images of the students

```
# get all students
students <- list.files(path = 'headshots_students/', pattern = '.png')
student_imgs <- list()
for(s in 1:length(students)) {
    student_imgs[[s]] <- readPNG(paste0('headshots_students/',students[s]))
}

# plot first four students
student_imgs_ras <- list()
for(i in 1:4) {
    student_imgs_ras[[i]] <- as.raster(student_imgs[[i]])
}

par(mfrow = c(1,4), mar = c(0,0,0,0))
for(ir in student_imgs_ras){
    plot(ir)
}
```

Sample images of the professors

```
faculty <- list.files(path = 'headshots_faculty/', pattern = '.png')
faculty_imgs <- list()
for(f in 1:length(faculty)) {
    faculty_imgs[[f]] <- readPNG(paste0('headshots_faculty/',faculty[f]))
}

# plot first four students
faculty_imgs_ras <- list()
for(i in 1:4) {
    faculty_imgs_ras[[i]] <- as.raster(faculty_imgs[[i]])
}

par(mfrow = c(1,4), mar = c(0,0,0,0))
for(ir in faculty_imgs_ras){
    plot(ir)
}
```

Aligning the pictures

We will align the faces and zoom into the faces using an image registration method. This is done using the python scripts `align_faces.py` and `converter.py`. Also we make sure that all the images have the same resolution.

Sample images of the students after aligning

```
# get all students
students <- list.files(path = 'headshots_students/aligned/', pattern = '.png')
student_imgs <- list()
for(s in 1:length(students)) {
    student_imgs[[s]] <- readPNG(paste0('headshots_students/aligned/',students[s]))
}

# plot first four students
student_imgs_ras <- list()
for(i in 1:4) {
    student_imgs_ras[[i]] <- as.raster(student_imgs[[i]])
}

par(mfrow = c(1,4), mar = c(0,0,0,0))
for(ir in student_imgs_ras){
```

```
    plot(ir)
}
```

Sample images of the faculty after aligning

```
# get all faculty
faculty <- list.files(path = 'headshots_faculty/aligned/', pattern = '.png')
faculty_imgs <- list()
for(f in 1:length(faculty)) {
    faculty_imgs[[f]] <- readPNG(paste0('headshots_faculty/aligned/',faculty[f]))
}

# plot first four students
faculty_imgs_ras <- list()
for(i in 1:4) {
    faculty_imgs_ras[[i]] <- as.raster(faculty_imgs[[i]])
}

par(mfrow = c(1,4), mar = c(0,0,0,0))
for(ir in faculty_imgs_ras){
    plot(ir)
}
```

Principal component analysis to find the eigen-faces

Turn the images in to grey scale for consistency

```
# covert to gray scale using avarage method

#students
student_imgs_grey <- list()
for(s in 1:length(student_imgs)){
    student_imgs_grey[[s]] <- (student_imgs[[s]][,,1]+student_imgs[[s]][,,2]+
                               student_imgs[[s]][,,3])/3
}
#faculty
faculty_imgs_grey <- list()
for(f in 1:length(faculty_imgs)){
    faculty_imgs_grey[[f]] <- (faculty_imgs[[f]][,,1]+faculty_imgs[[f]][,,2]+
                                 faculty_imgs[[f]][,,3])/3
}
```

Most important 4 eigen faces for students

```
# get average image
avg_student <- matrix(0, 150, 150)
for(s in student_imgs_grey){
    avg_student <- s + avg_student
}
avg_student <- avg_student/length(student_imgs_grey)

cat("Average MSDS student face")
```

```
## Average MSDS student face
```

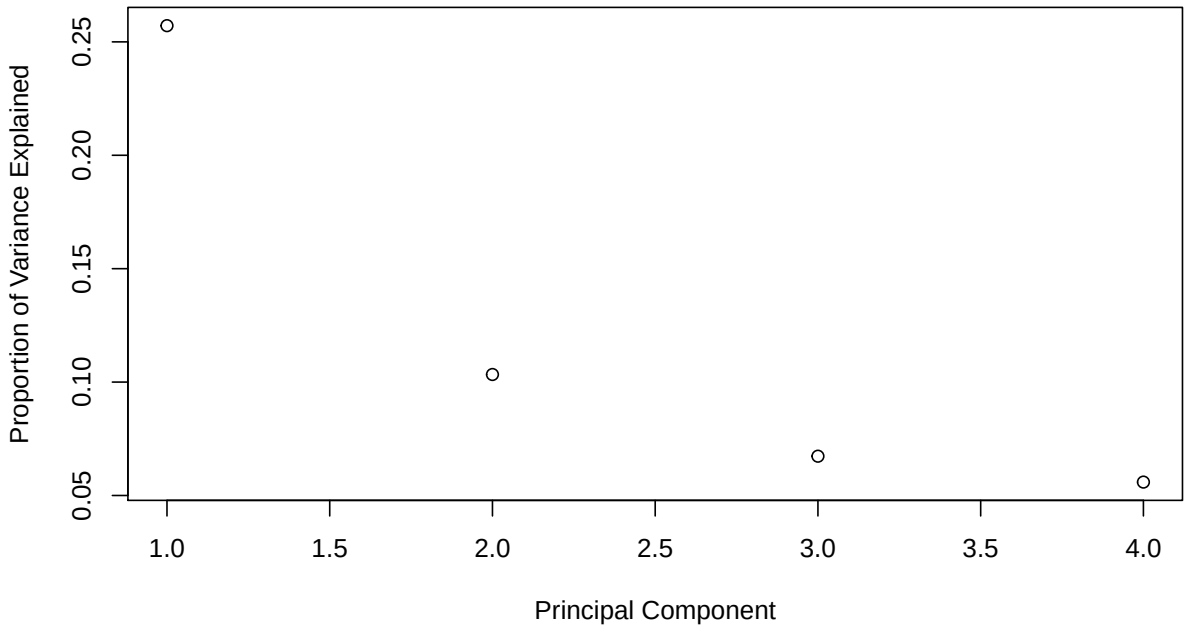
```
par(mfrow = c(1,2), mar = c(0,0,0,0))
plot(as.raster(avg_student))
```

```
# remove average image from each image
student_imgs_grey_scaled <- list()

for(s in 1:length(student_imgs_grey)){
    student_imgs_grey_scaled[[s]] <- student_imgs_grey[[s]] - avg_student
}

# convert to images to column format
student_imgs_grey_scaled_col <- list()
for(s in 1:length(student_imgs_grey_scaled)){
    d <- dim(student_imgs_grey_scaled[[s]])
```

```
    student_imgs_grey_scaled_col[[s]] <- student_imgs_grey_scaled[[s]]
    dim(student_imgs_grey_scaled_col[[s]]) <- c(1, d[1]*d[2])
}

# combine all images to one matrix
student_img_matrix <- do.call(rbind, student_imgs_grey_scaled_col)

# pc for students
pc_students <- prcomp(student_img_matrix, scale = FALSE)

# four most important eigen faces

par(mfrow = c(1,4), mar = c(0,0,0,0))
for(i in 1:4){
    pc <- pc_students$rotation[,i]
    dim(pc) <- c(150,150)

    scl <- (pc - min(as.numeric(pc)))
    scl <- scl / max(as.numeric(scl))
    plot(as.raster(scl))
}
```

Propotion of variability explained by each eigen face

```
# Proportion of the variability explained by student faces
pr.var <- pc_students$sdev^2
pve <- pr.var / sum(pr.var)
plot(pve[1:4], xlab = "Principal Component",
     ylab = "Proportion of Variance Explained")
```



Most important 4 eigen faces for faculty

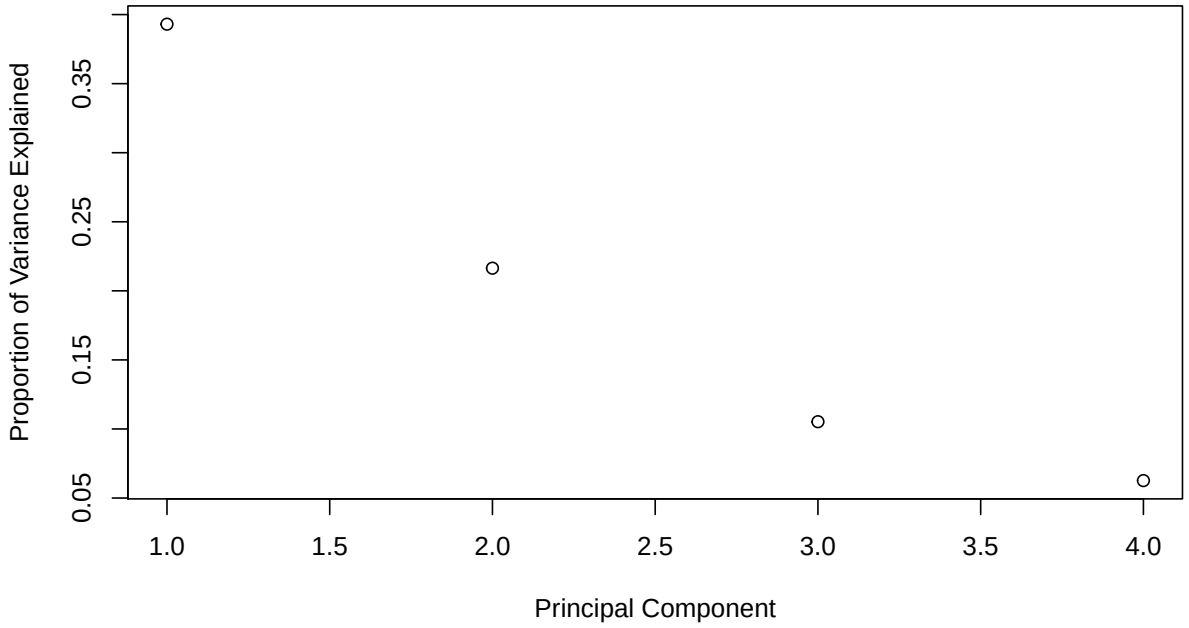
```
# get average image
avg_faculty <- matrix(0, 150, 150)
for(f in faculty_imgs_grey){
    avg_faculty <- f + avg_faculty
}
avg_faculty <- avg_faculty/length(faculty_imgs_grey)

par(mfrow = c(1,2), mar = c(0,0,0,0))
plot(as.raster(avg_faculty))
```

```
# remove average image from each image
faculty_imgs_grey_scaled <- list()

for(f in 1:length(faculty_imgs_grey)){
    faculty_imgs_grey_scaled[[f]] <- faculty_imgs_grey[[f]] - avg_faculty
}

# convert to images to column format
faculty_imgs_grey_scaled_col <- list()
for(f in 1:length(faculty_imgs_grey_scaled)){
    d <- dim(faculty_imgs_grey_scaled[[f]])
    faculty_imgs_grey_scaled_col[[f]] <- faculty_imgs_grey_scaled[[f]]
    dim(faculty_imgs_grey_scaled_col[[f]]) <- c(1, d[1]*d[2])
}

# combine all images to one matrix
faculty_img_matrix <- do.call(rbind, faculty_imgs_grey_scaled_col)

# pc for students
pc_faculty <- prcomp(faculty_img_matrix, scale = FALSE)

# four most important eigen faces
par(mfrow = c(1,4), mar = c(0,0,0,0))
for(i in 1:4){
    pc <- pc_faculty$rotation[,i]
    dim(pc) <- c(150,150)
```

```
    scl <- (pc - min(as.numeric(pc)))
    scl <- scl / max(as.numeric(scl))
    plot(as.raster(scl))
}
```

Propotion of variability explained by each eigen face

```
# Proportion of the variability explained by faculty faces
pr.var <- pc_faculty$sdev^2
pve <- pr.var / sum(pr.var)
plot(pve[1:4], xlab = "Principal Component", ylab = "Proportion of Variance Explained")
```



Interpreting faces as linear combinations of eigen faces

My original image and image constructed using first 4 eigen faces for students

```
# my orignial image
par(mfrow = c(1,2), mar = c(0,0,0,0))
plot(as.raster(student_imgs_grey[[35]]))

# all student images with only first 4 principal components
u_proj <- pc_students$rotation
#we will only use first 4 PC loadings so set the remaining to 0
u_proj[, 5:75] <- 0
projection_student <- (student_img_matrix%*%u_proj)%*%t(u_proj)

# my face with 4 eigen faces
my_face <- projection_student[35, ]
dim(my_face) <- c(150,150)

my_face_with_4pc <- avg_student+my_face # adding the average student
scl <- (my_face_with_4pc - min(as.numeric(my_face_with_4pc)))
scl <- scl / max(as.numeric(scl))
plot(as.raster(scl))
```

```
# find the linear combination of eigen faces
b = matrix(projection_student[35, ][1:4], 4)
X = matrix(unlist(c(pc_students$rotation[,1][1:4], pc_students$rotation[,2][1:4],
       pc_students$rotation[,3][1:4], pc_students$rotation[,4][1:4])), 4)
lin_comb <- round(solve(X,b),3)
```

Weights of the linear combination of student eigen faces that result in my face is 14.17, 4.361, -19.416, -5.79 respectively

Original image of computation statistics professor and image constructed using first 4 eigen faces of faculty

```
# James original image
par(mfrow = c(1,2), mar = c(0,0,0,0))
plot(as.raster(faculty_imgs_grey[[14]]))

# all faculty images with only first 4 principal components
u_proj <- pc_faculty$rotation
#we will only use first 4 PC loadings so set the remaining to 0
u_proj[, 5:15] <- 0
projection_faculty <- (faculty_img_matrix%*%u_proj)%*%t(u_proj)

# my face with 4 eigen faces
james_face <- projection_faculty[14, ]
dim(james_face) <- c(150,150)
```

```
james_face_with_4pc <- avg_faculty+james_face # adding the average faculty
scl <- (james_face_with_4pc - min(as.numeric(james_face_with_4pc)))
scl <- scl / max(as.numeric(scl))
plot(as.raster(scl))
```

```
# find the linear combination of eigen faces
b = matrix(projection_faculty[14, ][1:4], 4)
X = matrix(unlist(c(pc_faculty$rotation[,1][1:4], pc_faculty$rotation[,2][1:4],
                    pc_faculty$rotation[,3][1:4], pc_faculty$rotation[,4][1:4])), 4)
lin_comb <- round(solve(X,b),3)
```

Weights of the linear combination of faculty eigen faces that result in professor's face is -14.886, -9.333, 0.405, -10.831 respectively

Lets find similarity between faculty and students

Lets first calculate the weights for the linear combination of first four eigen faces for each person with their group. Then these weights for each group are standardized to be between 0 and 1. So the similarity measure we are going to use is the distance between standardized weight vectors for faculty and students. Advantage of this method is we are calculating distance using vectors of size 4 compared original images with vectors of size 22500. Hence this is can be computationally efficient for a very large dataset.

Derive similarity measures

```
student_weights <- list()
for(s in 1:75){
    b = matrix(projection_student[s, ][1:4], 4)
    X = matrix(unlist(c(pc_students$rotation[,1][1:4], pc_students$rotation[,2][1:4],
                        pc_students$rotation[,3][1:4], pc_students$rotation[,4][1:4])), 4)
    student_weights[[s]] <- round(solve(X,b),3)
}
student_weights <- matrix(unlist(student_weights), 75, byrow = T)
student_weights_scl <- (student_weights -
                            min(as.numeric(student_weights)))
student_weights_scl <- student_weights_scl /
    max(as.numeric(student_weights_scl))

faculty_weights <- list()
for(f in 1:15){
    b = matrix(projection_faculty[f, ][1:4], 4)
    X = matrix(unlist(c(pc_faculty$rotation[,1][1:4], pc_faculty$rotation[,2][1:4],
                        pc_faculty$rotation[,3][1:4], pc_faculty$rotation[,4][1:4])), 4)
    faculty_weights[[f]] <- round(solve(X,b),3)
}
faculty_weights <- matrix(unlist(faculty_weights), 15, byrow = T)
faculty_weights_scl <- (faculty_weights -
                            min(as.numeric(faculty_weights)))
faculty_weights_scl <- faculty_weights_scl /
    max(as.numeric(faculty_weights_scl))

distances <- list()
for(f in 1:length(faculty)){
    f_dist <- list()
    for(s in 1:length(students)){
        f_dist[[s]] <- dist(rbind(faculty_weights_scl[f, ],
                                  student_weights_scl[s,]))
    }
    distances[[f]] <- f_dist
}

dist_mat <- matrix(unlist(distances), 15, byrow = T)

# get cleaned student and faculty name vectors
student_names <- rep(0,75)
for(s in 1:length(student_names)){
    student_names[s] <- gsub("alignedmsan-student-", "", students[s])
    student_names[s] <- gsub(".png", "", student_names[s])
}

faculty_names <- rep(0,15)
for(f in 1:length(faculty_names)){
    faculty_names[f] <- gsub("aligned", "", faculty[f])
    faculty_names[f] <- gsub(".png", "", faculty_names[f])
}
```

Student whose closest to our computational statistic professor is ruchirawat-nicha

Graph to show the similarity between faculty and students

```
graph_mat <- dist_mat

graph_mat[graph_mat<=0.25] <- 1
graph_mat[graph_mat !=1] <- 0

graph_adj <- rep(0, (75+15)^2)
graph_adj <- matrix(graph_adj, (75+15))

graph_adj[76:90, 1:75] <- graph_mat
graph_adj[1:75, 76:90] <- t(graph_mat)

G <- graph_from_adjacency_matrix(graph_adj, mode = 'undirected')
V(G)$name <- c(student_names, faculty_names)
V(G)$role <- c(rep(2,75), rep(4,15))

G1 <- delete.vertices(G, degree(G)==0)

plot(G1, main=paste("Graph network"), usearrows = FALSE, edge.col = "grey50",
     vertex.size = 5, vertex.color = V(G1)$role, vertex.label.dist=1,
     vertex.label.cex = 0.8)
```

Graph network