DATA605 - Week 4

Keith Colella

2023-09-25

```
library(doParallel)
library(foreach)
library(jpeg)
library(EBImage)
library(xROI)
library(tidyverse)
```

Assignment

The goal is to take a series of images of shoes from a single brand, and build and visualize eigenimagery that accounts for 80% of the variability.

I'll largely follow the example code provided by Prof. Fulton in this assignment, instead focusing on interpreting results!

View Images

First, we'll load in the images, set a few key parameters (height, width, scale). We then add those images to a large array.

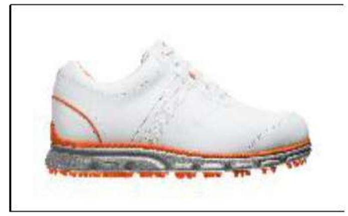
```
files = list.files('data/shoes')

height = 1200
width = 2500
scale = 10

im = array(
  rep(0, length(files) * height / scale * width / scale * 3),
  dim = c(length(files), height / scale, width / scale, 3)
)

for (i in 1:length(files)) {
  tmp = paste0('data/shoes/', files[i])
  temp = EBImage::resize(readJPEG(tmp), height / scale, width / scale)
  im[i,,,] = array(temp, dim = c(1, height / scale, width / scale, 3))
}
```

We then loop through that array to plot the shoe images.

```
par(mfrow=c(3,3))
par(mai=c(.3,.3,.3,.3))
for (i in 1:length(files)) {
  plotJPEG(writeJPEG(im[i,,,]))
}
```














Generate Principal Components

Next, we'll use R's native `princcomp` function to decompose the image arrays into principal components (i.e. eigenvectors).

```
newdata = im
dim(newdata) = c(length(files), height * width*3 / scale^2)
pca = princomp(t(as.matrix(newdata)), score = TRUE, cor = TRUE)
```

We can examine the standard deviation of these components to see to what degree they explain the variability across images. If we look at the cumulative sum, we see that the sum of just the first three components exceeds our 80% threshold.

```
components = pca$sdev^2 / sum(pca$sdev^2)

print(cumsum(components))
```

```
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 0.6916309 0.7911549 0.8442585 0.8714110 0.8903922 0.9070168 0.9213980 0.9335995
##    Comp.9   Comp.10   Comp.11   Comp.12   Comp.13   Comp.14   Comp.15   Comp.16
## 0.9433646 0.9523060 0.9606716 0.9685299 0.9760625 0.9826631 0.9891121 0.9950668
##   Comp.17
## 1.0000000
```

```
sum(components[1:2])
```

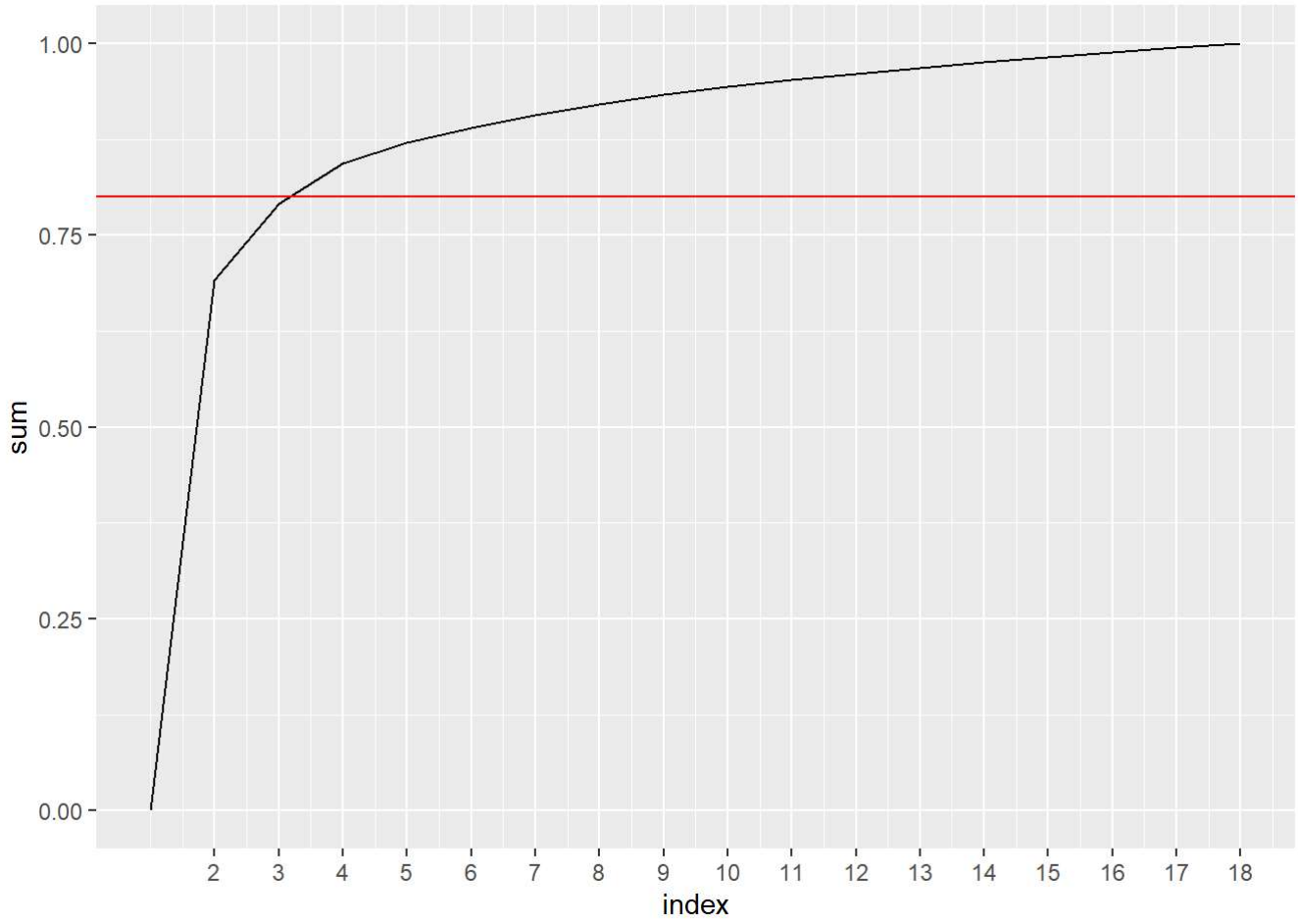
```
## [1] 0.7911549
```

```
sum(components[1:3])
```

```
## [1] 0.8442585
```

We can actually plot the cumulative sum to see how much variability is explained by each incremental component.

```
data.frame(index = seq(1,length(components)+1), sum = c(0,cumsum(components))) %>%
  ggplot(aes(index, sum)) +
  geom_line() +
  geom_hline(yintercept = 0.8, color = 'red') +
  scale_x_continuous(breaks=1:length(components)+1)
```



Visualize Eigenshoes

Finally, we can then plot these three components to re-create the key “archetypes” for these shoe images.

```
pca2 = t(pca$scores)
dim(pca2) = c(length(files), height / scale, width / scale, 3)
par(mfrow=c(1,3))
par(mai = c(0.1, 0.1, 0.1, 0.1))

for (i in 1:3) {
  jpg = readJPEG(writeJPEG(pca2[i,,,]))
  res = dim(jpg)
  plot(
    1, 1,
    xlim=c(1,res[1]),
    ylim=c(1,res[2]),
    asp=1, type='n',
    xaxs='i',yaxs='i',
    xaxt='n',yaxt='n',
    xlab='',ylab='',
    bty='n'
  )
  rasterImage(jpg, 1, 1, res[1], res[2])
}
```