In Class Assignment

Jen Johnson and Tina Chen

12/5/2017

```
data <- read.csv("hwscores.csv")
```

Function for binding columns of different length together.

```
cbind.fill <- function(...){
    nm <- list(...)
    nm <- lapply(nm, as.matrix)
    n <- max(sapply(nm, nrow))
    do.call(cbind, lapply(nm, function (x)
        rbind(x, matrix(, n-nrow(x), ncol(x)))))
}
```

Part 1

```
# get list of 24 people
a <- data.frame(colnames(data))
a$new <- substr(a$colnames.data., 1, 3)
people <- unique(a$new)
people2 <- data.frame(people)
```

```
# set up df where each col == 1 person's genuine scores
genuine.data <- list()

for(i in 1:length(people)){
    id1 <- people[i]
    # filter col
    current <- select(data, starts_with(id1))
    names <- colnames(current)
    # filter row using names
    temp <- current[names, ]
    # only use the top half of the matrix
    temp[lower.tri(temp)] <- NA
    # remove NA and 0s
    v <- unlist(temp)
    d <- data.frame(v)
    d2 <- na.exclude(d)
    d3 <- d2[d2!=0]
    d4 <- as.data.frame(d3)
    # add to genuine.data
    genuine.data <- cbind.fill(genuine.data, d4)
}
```

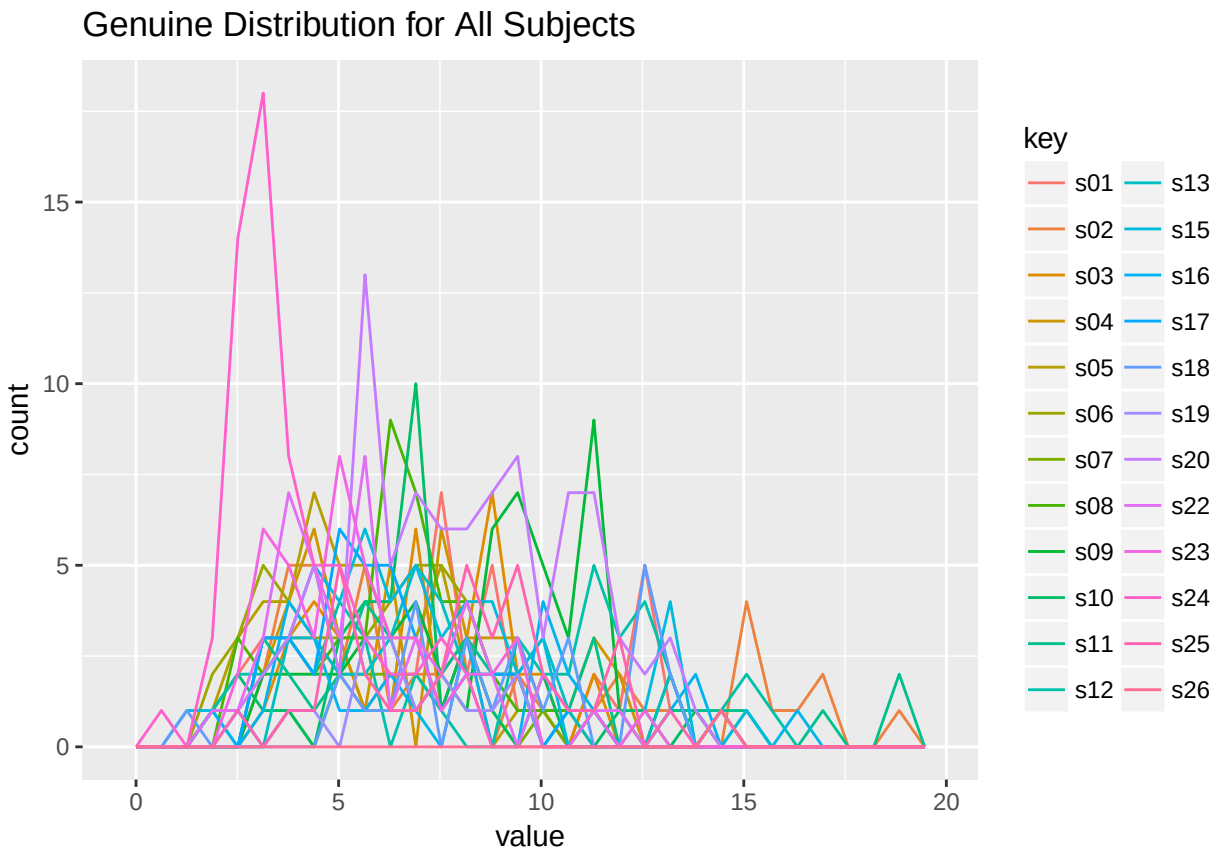
```
# remove empty first col
genuine.data <-subset(genuine.data, select= -c(1))
# set colnames for legend
colnames(genuine.data) <- people2$people
genuine.data <- data.frame(genuine.data)
```

```r
# convert to long format
temp <- gather(genuine.data, key = "key", value = "value")
temp$value <- as.numeric(temp$value)

# plot
ggplot(data = temp, aes(x = value, color = key)) +
  geom_freqpoly() +
  ggtitle("Genuine Distribution for All Subjects")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1209 rows containing non-finite values (stat_bin).
```



Part 2

```r
# convert all columns to numeric
numeric <- lapply(genuine.data, as.numeric)

means <- data.frame(lapply(numeric, mean, na.rm = TRUE))
ranges <- data.frame(lapply(numeric, range, na.rm = TRUE))

# print nicely
kable(means)
```

s01	s02	s03	s04	s05	s06	s07	s08	s09	s10	s11
7.568867	8.700097	7.072527	6.829411	5.432658	5.622851	6.808968	6.292376	8.64306	6.775542	8.627768

```
kable(ranges)
```

s01	s02	s03	s04	s05	s06	s07	s08	s09	s10	
2.023148	3.031599	3.015516	1.98071	1.960503	2.102067	3.882557	2.493492	2.85584	1.760282	3
14.018000	18.619373	11.462518	12.20853	11.067879	9.278427	10.045925	10.383258	14.28027	13.529818	18

Part 3

```
all.scores <- unlist(numeric)
all.scores.mean <- data.frame(mean(all.scores, na.rm = TRUE))
colnames(all.scores.mean)<- c("Mean")
kable(all.scores.mean)
```

Mean
7.059704

Part 4

```
# use Part 3 results and which.max find the index
largest.mean.index <- which.max(means)
largest.mean.person <- data.frame(people[largest.mean.index])
smallest.mean.index <- which.min(means)
smallest.mean.person <- data.frame(people[smallest.mean.index])

t.ranges <- t(ranges)
t.ranges <- as.data.frame(t.ranges)
t.ranges$new <- abs(t.ranges$V1 - t.ranges$V2)

largest.range.index <- which.max(t.ranges$new)
largest.range.person <- data.frame(people[largest.range.index])

kable(largest.mean.person) # s18
```

people.largest.mean.index.
s26

```
kable(smallest.mean.person)
```

people.smallest.mean.index.
s24

```
kable(largest.range.person)
```

people.largest.range.index.
s11

Highest average: While s26 gave us the highest average matching score, we treated it as an outlier because there was only 1 matching score in the s26 column of genuine.data. If we were to recalculate, the highest average matching score would actually be s18 (Chloe). This makes more sense because the images are all taken at very similar angles, with very similar lighting, and little variation in facial expression.

Lowest average: s24 (Jocelyn) has the lowest average matching score, because there are images with and without glasses, different facial expressions, hair styles (some up and some down), and various degrees of lighting.

Largest range: s11 (Jen) has the largest range, because there there are many different poses, facial expression, lighting, and hair styles.

Part 5

```
# convert matrix into list format
all.data <- as.matrix(data)
scores_list <- melt(all.data)[melt(upper.tri(all.data))$value,]

# get imposter data
scores_list$seq_1_subject <- substr(scores_list$X1, 2, 3)
scores_list$seq_2_subject <- substr(scores_list$X2, 2, 3)
imposter_data <- scores_list[!scores_list$seq_1_subject==scores_list$seq_2_subject, ]

# get the minimum distance score
min_imposter <- min(imposter_data$value)
min_imposter_pair <- which(imposter_data$value == min_imposter)

kable(imposter_data[min_imposter_pair,])
```

	X1	X2	value	seq_1_subject	seq_2_subject
46149	s23d5.png	s24d6.png	1.331537	23	24

The most similar looking imposter pair was s23d5 and s24d6. The two images share the same facial expression, lighting, and pose. The hair fringe at the top of the face and eyebrow angle are similar.

Part 6

```
genuine_data <- scores_list[scores_list$seq_1_subject==scores_list$seq_2_subject, ]

min_genuine <- min(genuine_data$value)
min_genuine_pair <- which(genuine_data$value == min_genuine)

kable(genuine_data[min_genuine_pair,])
```

	X1	X2	value	seq_1_subject	seq_2_subject
46604	s24d3.png	s24d8.png	0.6464693	24	24

The lowest non-zero distance score was between s24's d8 and d3 photos. The photos are the same, but 1 had been rotated and relabelled.