Assignment 7 By Team 1

Odianosen Akhibi, Shun-Lung Chang, Juliana Nair

This study was conducted in R, and the source code can be found here.

1. Generate a simulated data set with 20 observations in each of three classes (i.e. 60 observations total), and 50 variables.

In this study, the normal, uniform and exponential data are selected as the distributions of the random number generators. The detailed parameter information is listed below. Three random number matrices for each distribution are then created by `rnormal()`, `runif()` and `rexp()`, and concatenated by `rbind()`.

1. $X1 \sim Normal(\mu = 500, \sigma^2 = 25), E(X1) = 500, V(X1) = 25$
2. $X2 \sim Uniform(min = -75, max = -25), E(X2) = -50, V(X2) = (-75 - (-25))^2/12 = 208.33$
3. $X3 \sim Exponential(\lambda = 1), E(X3) = 1, V(X3) = 1$

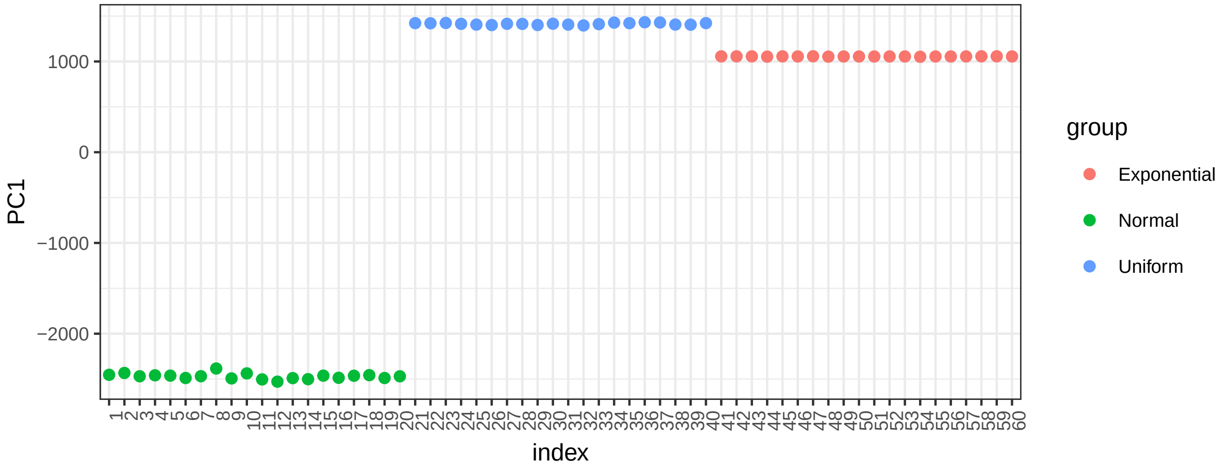
```r
# set the random number seed
set.seed(45)

x1 <- rnorm(1000, mean = 500, sd = 25) %>%
    matrix(20, 50)
x2 <- runif(1000, min = -75, max = -25) %>%
    matrix(20, 50)
x3 <- rexp(1000, rate = 1) %>%
    matrix(20, 50)

dat <- rbind(x1, x2, x3)

dim(dat)
```

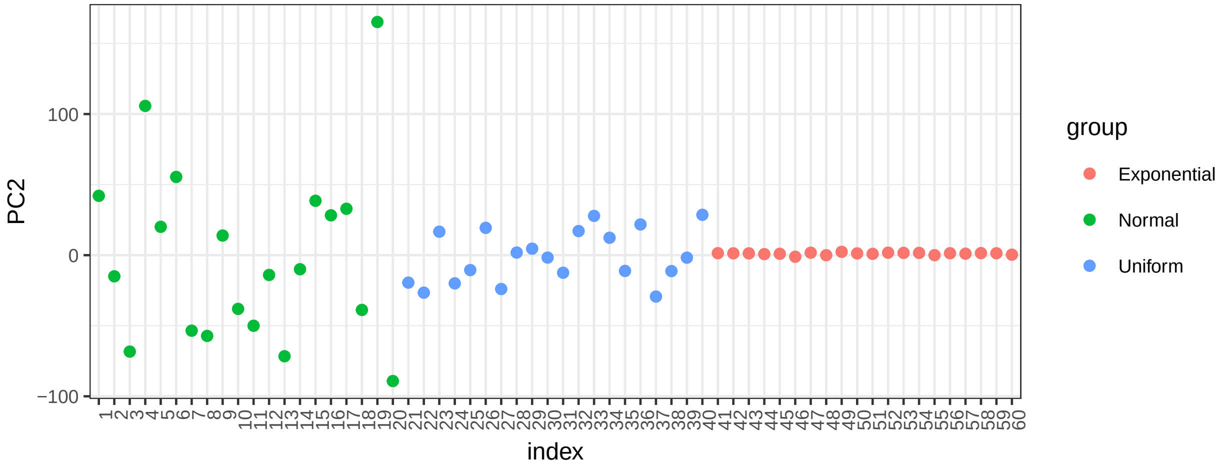
```
[1] 60 50
```

2. Perform PCA on the 60 observations and plot the first two principal component score vectors. Use a different color to indicate the observations in each of the three classes.

In R, `prcomp()` can be used to perform priciple component analysis (PCA hereafter). As can be seen from the plots below, the values of three classes differ notably in first two score vectors after the PCA. Also, the significant difference can be used for cluster analysis.

```r
pca <- prcomp(dat)
```





3. Perform K-means clustering of the observations with K = 3. How well do the clusters that you obtained in K-means clustering compare to the true class labels?

By applying `kmeans()` on the dataset, we can obtain the results of K-means cluster analysis. The following table indicates that the K-means clusters are consistent with the real groups although the order differs.

```r
set.seed(42)
kc_3 <- kmeans(dat, center = 3)

table(kc_3$cluster, c(rep(1, 20), rep(2, 20), rep(3, 20))) %>% kable()
```

1	2	3
0	20	0
0	0	20
20	0	0

4. Perform K-means clustering with K = 2. Describe your results.

If the number of cluster is two, as can be seen from the table, numbers generated from uniform and exponential distributions are grouped into one. The results could be attributed to the means of the two distributions, set in the first task, are much closer.

```r
set.seed(42)
kc_2 <- kmeans(dat, center = 2)

table(kc_2$cluster, c(rep(1, 20), rep(2, 20), rep(3, 20))) %>% kable()
```

1	2	3
0	20	20
20	0	0

5. Now perform K-means clustering with K = 4, and describe your results.

If the number of centers is four, initialization of centers, namely choosing which rows are the initial centers, would play a important role in determing the clusters. The following three tables shows that as the random seed varies, the results would change with respect with different initial centers.

```r
set.seed(42)
kc_4 <- kmeans(dat, center = 4)

table(kc_4$cluster, c(rep(1, 20), rep(2, 20), rep(3, 20))) %>% kable()
```

1	2	3
0	20	0
0	0	17
20	0	0
0	0	3

```r
set.seed(44)
kc_4 <- kmeans(dat, center = 4)

table(kc_4$cluster, c(rep(1, 20), rep(2, 20), rep(3, 20))) %>% kable()
```

1	2	3
0	0	20
20	0	0
0	10	0
0	10	0

```r
set.seed(46)
kc_4 <- kmeans(dat, center = 4)

table(kc_4$cluster, c(rep(1, 20), rep(2, 20), rep(3, 20))) %>% kable()
```

1	2	3
6	0	0
7	0	0
0	20	20
7	0	0

6. Now perform K-means clustering with K = 3 on the first two principal component score vectors, rather than on the raw data. That is, perform K-means clustering on the 60 × 2 matrix of which the first column is the first principal component score vector, and the second column is the second principal component score vector. Comment on the results.

To perform this task, we extracted the first two score vectors and then applied K-means on them. As the considerable difference we indicated in task 2, the K-means groups align with the real groups.

```r
set.seed(42)
kc_pc_3 <- kmeans(cbind(two_pcs$PC1, two_pcs$PC2), centers = 3)

table(kc_pc_3$cluster, c(rep(1, 20), rep(2, 20), rep(3, 20))) %>% kable()
```

1	2	3
0	20	0
0	0	20
20	0	0

7. Using the `scale()` function, perform K-means clustering with K = 3 on the data after scaling each variable to have standard deviation one. How do these results compare to those obtained in (3)? Explain.

The K-means groups derived from the scaled dataset are consistent with the real groups. The results should not be unexpected, given that the scale of each variable is reduced to the same, so the results would be more stable, that is, be more likely to match the real groups, when K-meanss clustering is applied.

```r
dat_scaled <- scale(dat)
set.seed(42)
kc_3 <- kmeans(dat_scaled, center = 3)

table(kc_3$cluster, c(rep(1, 20), rep(2, 20), rep(3, 20))) %>% kable()
```

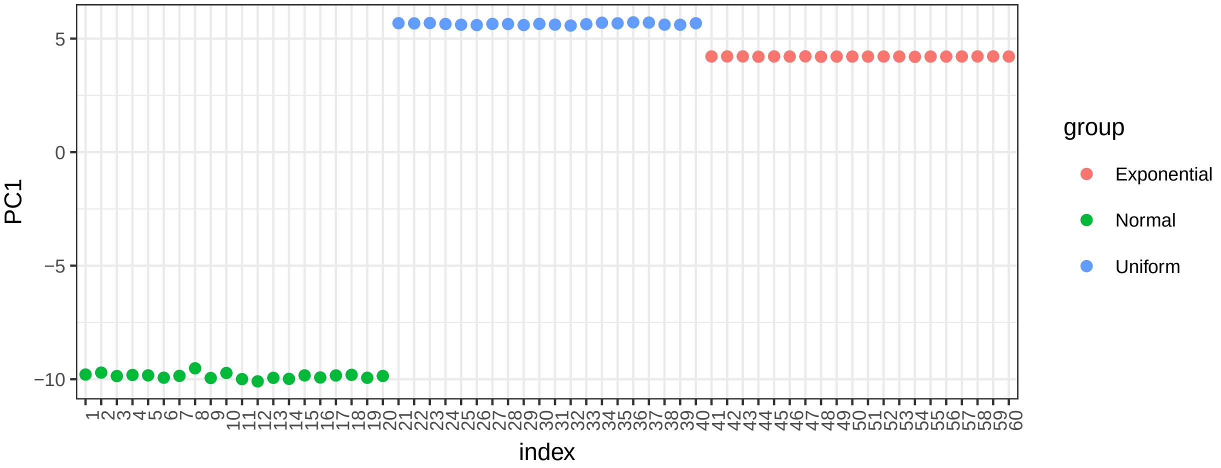
1	2	3
0	20	0
0	0	20
20	0	0

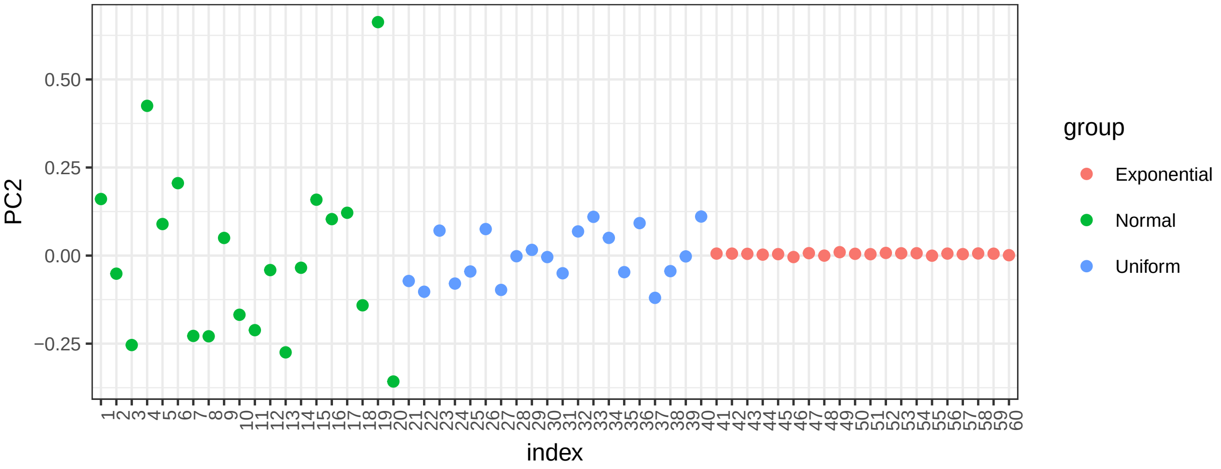
8. Use the scaled variables and run a PCA on them. Now perform K-means clustering with K = 3 on the first two principal component score vectors, rather than on the raw data. How do these results compare to those obtained in (3) and (7)? Explain.

The K-meanss clusters obtained in this task are the same with those in (3) and (7). The two plots below show that significant difference exists in the values of the first two score vectors. Thus the difference results in the K-meanss clusters that match the real groups.

```r
pca_scaled <- prcomp(dat, center = TRUE, scale. = TRUE)

set.seed(42)
kc_pc_scaled_3 <- kmeans(data.frame(pca_scaled$x[, 1], pca_scaled$x[, 2]), centers = 3)

table(kc_pc_scaled_3$cluster, c(rep(1, 20), rep(2, 20), rep(3, 20))) %>% kable()
```

1	2	3
0	20	0
0	0	20
20	0	0


PC1
index
group
● Exponential
● Normal
● Uniform


PC2
index
group
● Exponential
● Normal
● Uniform