ps7 borui sun

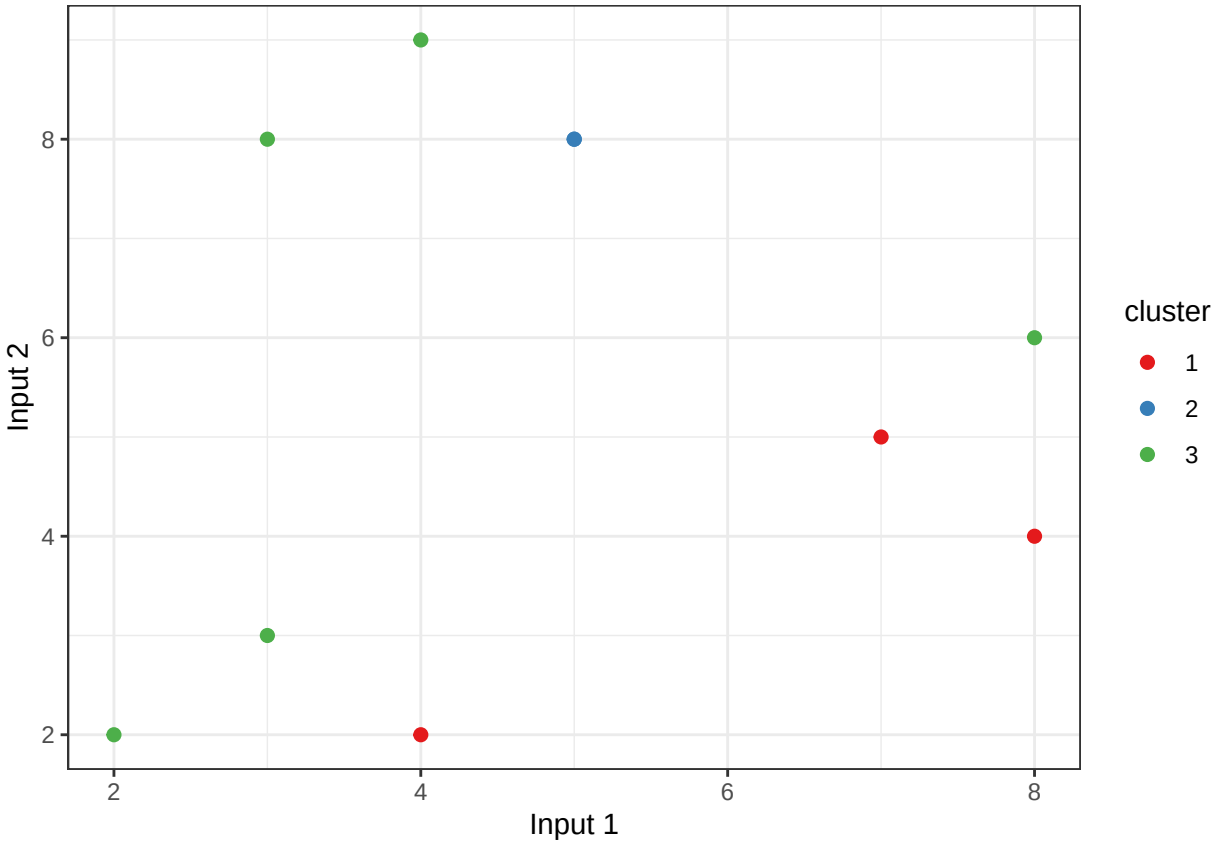
borui sun

3/15/2020

k-Means Clustering "By Hand"

1. (5 points) Imitate the k-means random initialization part of the algorithm by assigning each observation to a cluster at random.

```
sampledata <- tibble(
  input_1 = c(5,8,7,8,3,4,2,3,4,5),
  input_2 = c(8,6,5,4,3,2,2,8,9,8),
  cluster = sample(1:3, 10, replace = TRUE))

ggplot(sampledata, aes(x = input_1, y = input_2, color = as.factor(cluster))) +
  geom_point(size = 2) +
  scale_color_brewer(palette = "Set1") +
  labs(x = "Input 1", y = "Input 2", color = "cluster")
```

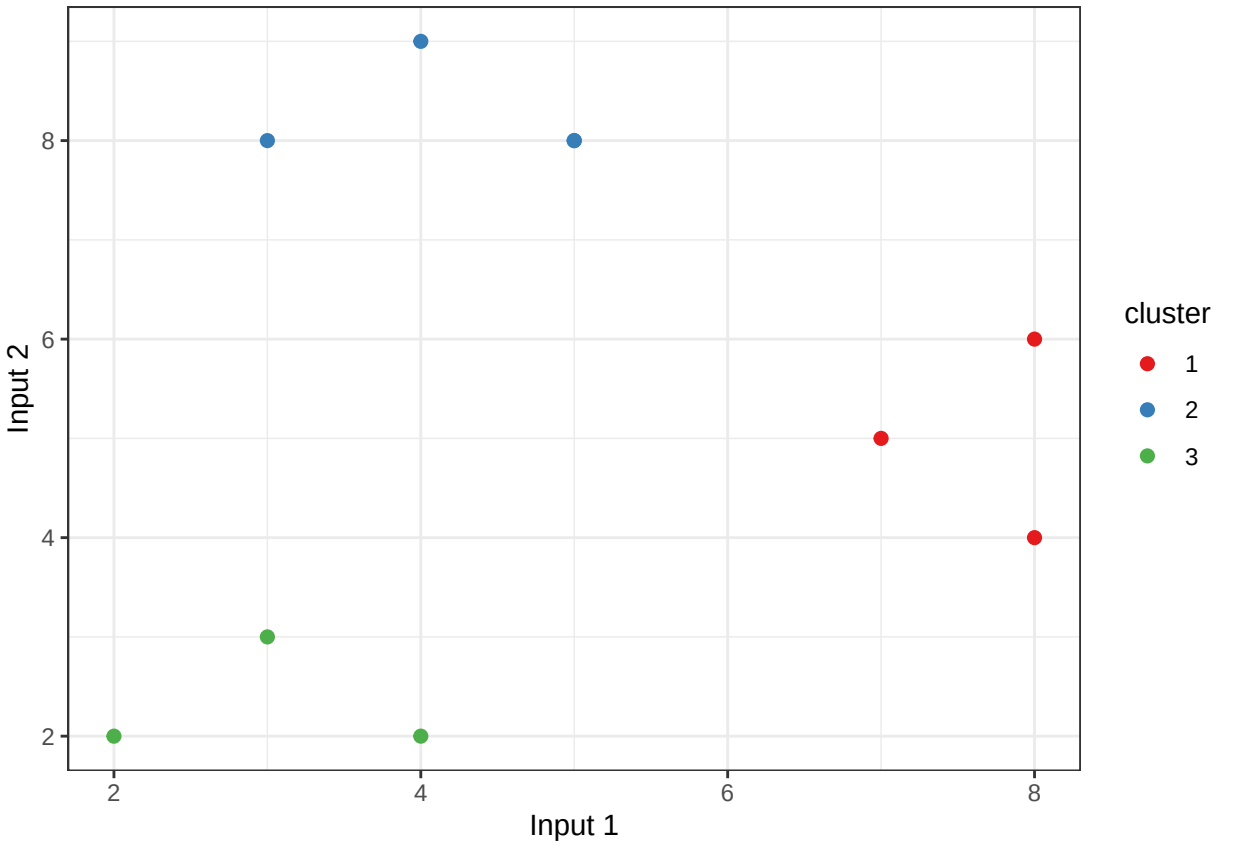


2. (5 points) Compute the cluster centroid and update cluster assignments for each observation iteratively based on spatial similarity.

```
for (i in 1:10000000){

  cluster.old <- sampledata$cluster

  centroids <- sampledata %>%
    group_by(cluster) %>%
    summarise(centroid.1 = mean(input_1), centroid.2 = mean(input_2))

  for (i in 1:nrow(sampledata)) {
    distance.c1 <- sqrt((sampledata$input_1[i]-centroids$centroid.1[1])^2
                        + (sampledata$input_2[i]-centroids$centroid.2[1])^2)
    distance.c2 <- sqrt((sampledata$input_1[i]-centroids$centroid.1[2])^2
                        + (sampledata$input_2[i]-centroids$centroid.2[2])^2)
    distance.c3 <- sqrt((sampledata$input_1[i]-centroids$centroid.1[3])^2
                        + (sampledata$input_2[i]-centroids$centroid.2[3])^2)
    distance.min <- c(distance.c1, distance.c2, distance.c3) %>% min()

    if(distance.min == distance.c1){sampledata$cluster[i] <- 1}
    else if(distance.min == distance.c2){sampledata$cluster[i] <- 2}
    else if(distance.min == distance.c3){sampledata$cluster[i] <- 3}
}
  cluster.new <- sampledata$cluster

  if(sum(cluster.new - cluster.old) == 0) {break}
}
```

3. (5 points) Present a visual description of the final, converged (stopped) cluster assignments.

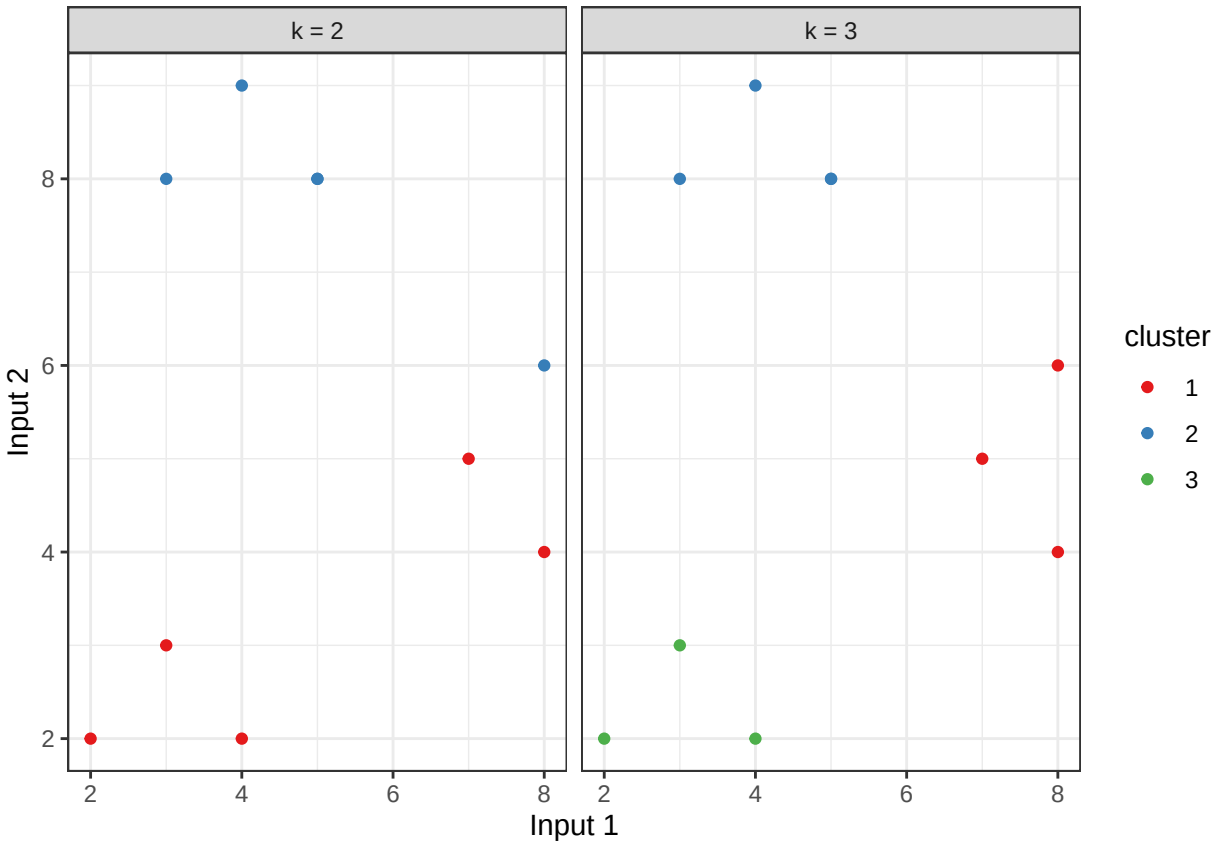
```
ggplot(sampledata, aes(x = input_1, y = input_2, color = as.factor(cluster))) +
  geom_point(size = 2) +
  labs(x = "Input 1", y = "Input 2", color = "cluster") +
  scale_color_brewer(palette = "Set1")
```



4. (5 points) Now, repeat the process, but this time initialize at `k = 2` and present a final cluster assignment visually next to the previous search at `k = 3`.

```
sampledata.k2 <- tibble(
  input_1 = c(5,8,7,8,3,4,2,3,4,5),
  input_2 = c(8,6,5,4,3,2,2,8,9,8),
  cluster = sample(1:2, 10, replace = TRUE))


for (i in 1:10000000){

  cluster.old <- sampledata.k2$cluster

  centroids <- sampledata.k2 %>%
    group_by(cluster) %>%
    summarise(centroid.1 = mean(input_1), centroid.2 = mean(input_2))

  for (i in 1:nrow(sampledata.k2)) {
    distance.c1 <- sqrt((sampledata.k2$input_1[i]-centroids$centroid.1[1])^2
                        + (sampledata.k2$input_2[i]-centroids$centroid.2[1])^2)
    distance.c2 <- sqrt((sampledata.k2$input_1[i]-centroids$centroid.1[2])^2
                        + (sampledata.k2$input_2[i]-centroids$centroid.2[2])^2)

    distance.min <- c(distance.c1, distance.c2) %>% min()
```

```
    if(distance.min == distance.c1){sampledata.k2$cluster[i] <- 1}
    else if(distance.min == distance.c2){sampledata.k2$cluster[i] <- 2}
}
  cluster.new <- sampledata.k2$cluster

  if(sum(cluster.new - cluster.old) == 0) {break}
}

sampledata %>% bind_cols(sampledata.k2[,"cluster"]) %>%
  `colnames<-`(c("input_1", "input_2", "k = 3", "k = 2")) %>%
  pivot_longer(cols = starts_with("k"),
               values_to = "cluster",
               names_to = "k") %>%
  ggplot(aes(x = input_1, y = input_2, color = as.factor(cluster))) +
  geom_point() +
  scale_color_brewer(palette = "Set1") +
  labs(x = "Input 1", y = "Input 2", color = "cluster") +
  facet_wrap(~k)
```



5. (10 points) Did your initial hunch of 3 clusters pan out, or would other values of `k`, like 2, fit these data better? Why or why not?

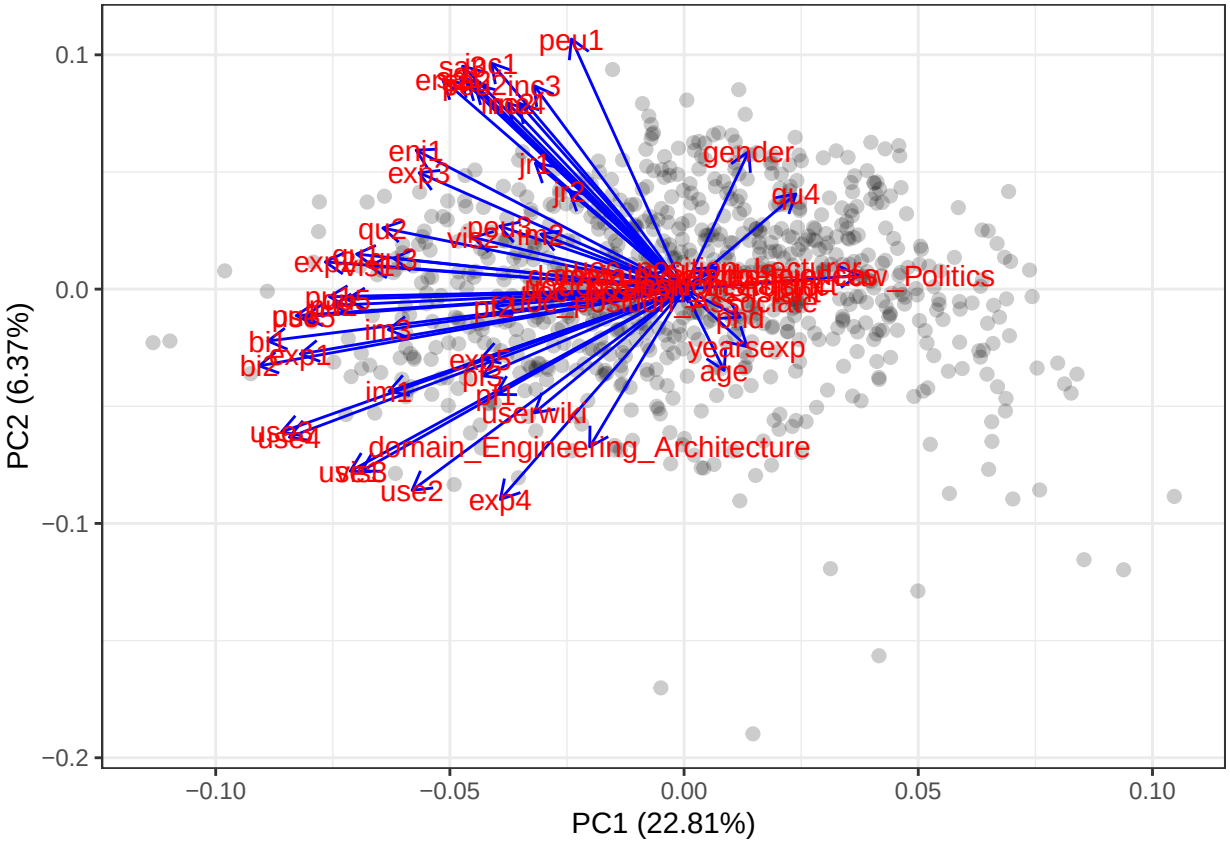
Based on the figure shown above, using a total of 3 clusters seems to describe the data distribution better than using a cluster size of 2, as there is a clear spatial separation of the data into 3 groups. The three dots concentrated in the right-middle of the plot are neither too close to the "top" cluster nor the "bottom" cluster. Coercing them into either cluster does not seem appropriate.

Dimension reduction

6. (15 points) Perform PCA on the dataset and plot the observations on the first and second principal components. Describe your results, e.g.,
   - What variables appear strongly correlated on the first principal component?
   - What about the second principal component?

bi2, bi1, use3, use4, pu3, exp1 are strongly correlated on the first principal component, while peu1, inc1, sa3, sa1, exp4 and enj2 are strongly correlated on the second principal component.

```
wiki <- read.csv("./data/wiki.csv") %>%
  mutate_all(scale)

wiki_pca <- prcomp(wiki)

autoplot(wiki_pca, loadings = TRUE, loadings.label = TRUE,
         loadings.colour = "blue", alpha = 0.2, size = 2,
         loadings.label.size = 4)
```




```
wiki_pca$rotation[, 1] %>% as.data.frame() %>%
  rownames_to_column() %>% `colnames<-`(c("Variable", "Comp1")) %>%
  arrange(desc(abs(Comp1))) %>% head() %>% kable()
```

Variable	Comp1
bi2	-0.2309240
bi1	-0.2261931
use3	-0.2188092
use4	-0.2145584
pu3	-0.2108626
exp1	-0.2085917

```
wiki_pca$rotation[, 2] %>% as.data.frame() %>%
  rownames_to_column() %>% `colnames<-`(c("Variable", "Comp2")) %>%
  arrange(desc(abs(Comp2))) %>% head() %>% kable()
```

Variable	Comp2
peu1	0.2717413
inc1	0.2454398
sa3	0.2423254
sa1	0.2299260
exp4	-0.2284943
enj2	0.2276024

7. (5 points) Calculate the proportion of variance explained (PVE) *and* the cumulative PVE for all the principal components. **Approximately how much of the variance is explained by the first two principal components?**

About 22.81% of variance are explained by the first principal component and 6.372% are explained by the second principal component. 29.183% of variance are explained by the first two principal together.

```
wik_pca_sum <- summary(wiki_pca)
wik_pca_sum$importance %>% as.data.frame() %>% {.[2:3,1:2]} %>% kable()
```

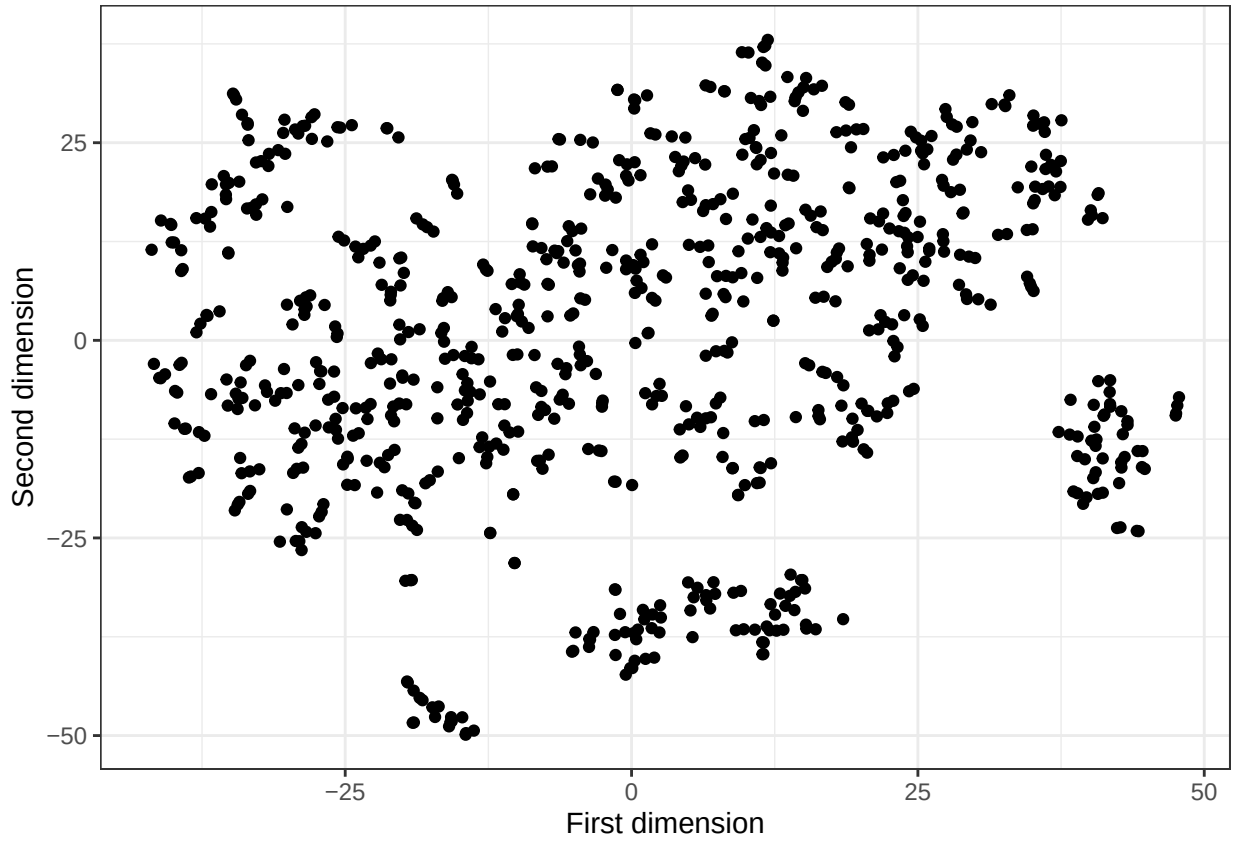
	PC1	PC2
Proportion of Variance	0.22811	0.06372
Cumulative Proportion	0.22811	0.29183

8. (10 points) Perform t-SNE on the dataset *and* plot the observations on the first and second dimensions. Describe your results.

From the TSNE plot below, we can see that there are 3~4 small clusters on the peripheral and a large cluster in the center, even though it is not too evident.

```
wiki_tsne <- Rtsne(as.matrix(wiki),
                   perplexity = 5)

wiki %>%
  mutate(tsne1 = wiki_tsne$Y[,1],
         tsne2 = wiki_tsne$Y[,2]) %>%
  ggplot(aes(tsne1, tsne2)) +
```

```
  geom_point() +
  labs(x = "First dimension",
       y = "Second dimension")
```

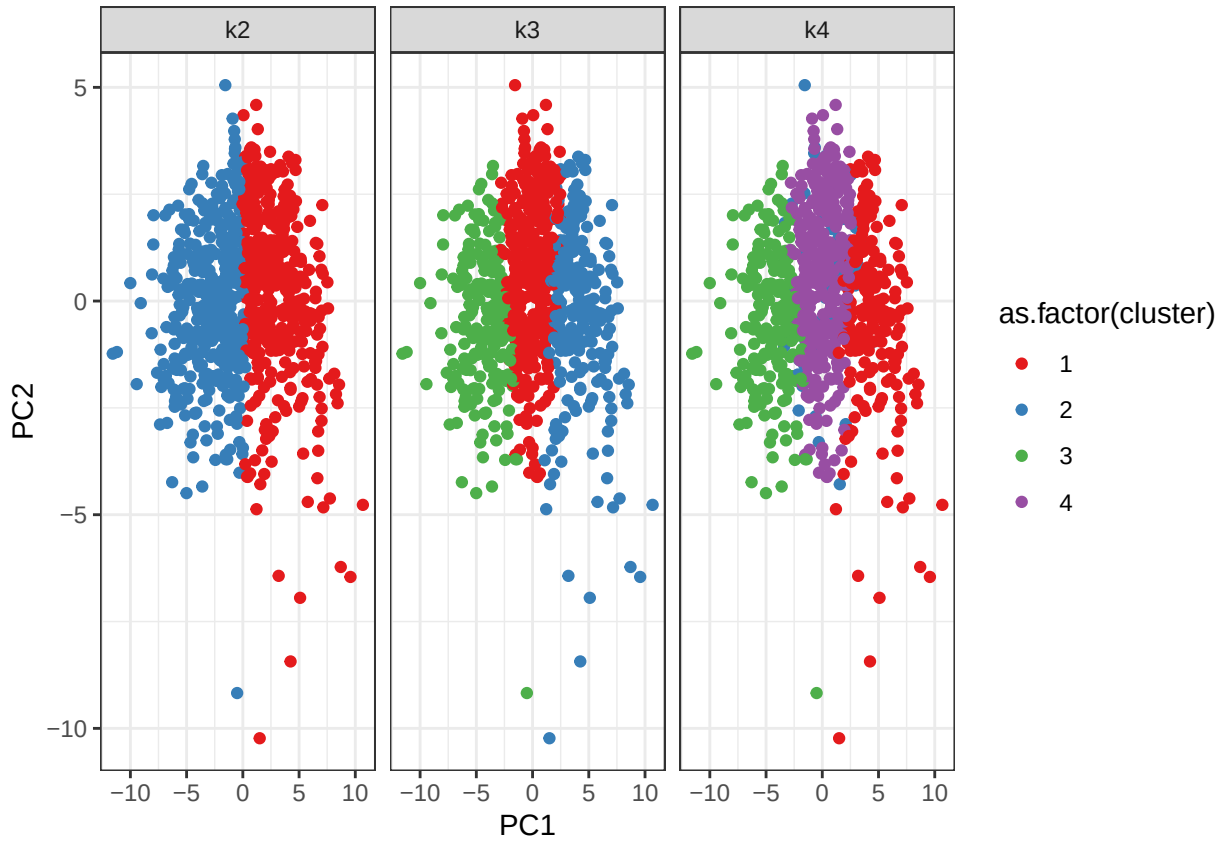


Clustering

9. (15 points) Perform k-means clustering with $k = 2, 3, 4$. Be sure to scale each feature (i.e.,mean zero and standard deviation one). *Plot* the observations on the first and second principal components from PCA and *color-code* each observation based on their cluster membership. *Discuss* your results.

From the figure below, we can see that the division of k cluster is mainly based on the first principal component. When k = 2, there is a clear cut of the data between first principal component greater than 0 and first principal component less than 0. Similar pattern can also be seen with using k = 3 and k = 4.

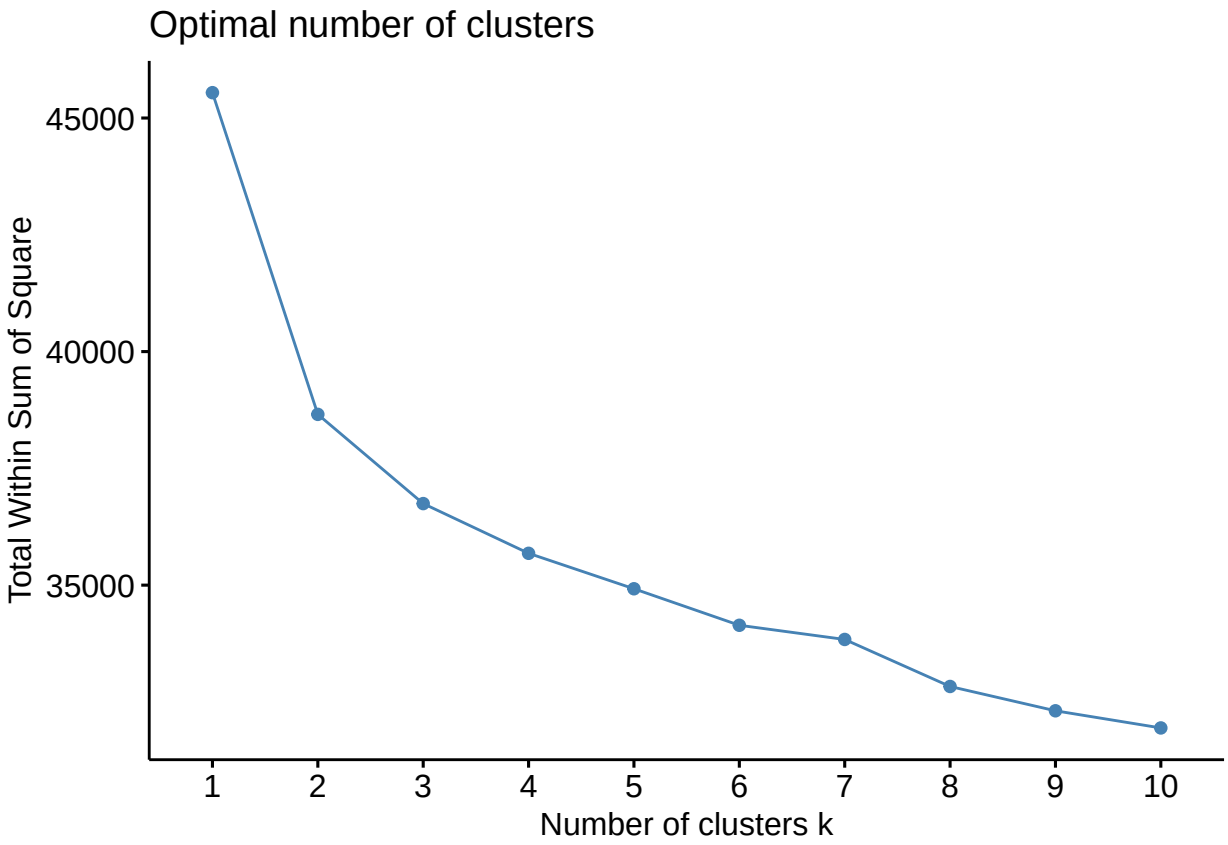
```
c(2:4) %>% map(~kmeans(wiki, centers = .)$cluster) %>% as.data.frame() %>% `colnames<-`(c("k2","k3","k4
  bind_cols(as.data.frame(wiki_pca$x[,1:2])) %>%
  pivot_longer(cols = starts_with("k"),
               names_to = "k",
               values_to = "cluster") %>%
  ggplot(aes(x = PC1, y = PC2, color = as.factor(cluster))) +
  geom_point() +
  scale_color_brewer(palette = "Set1") +
  facet_wrap(~k)
```



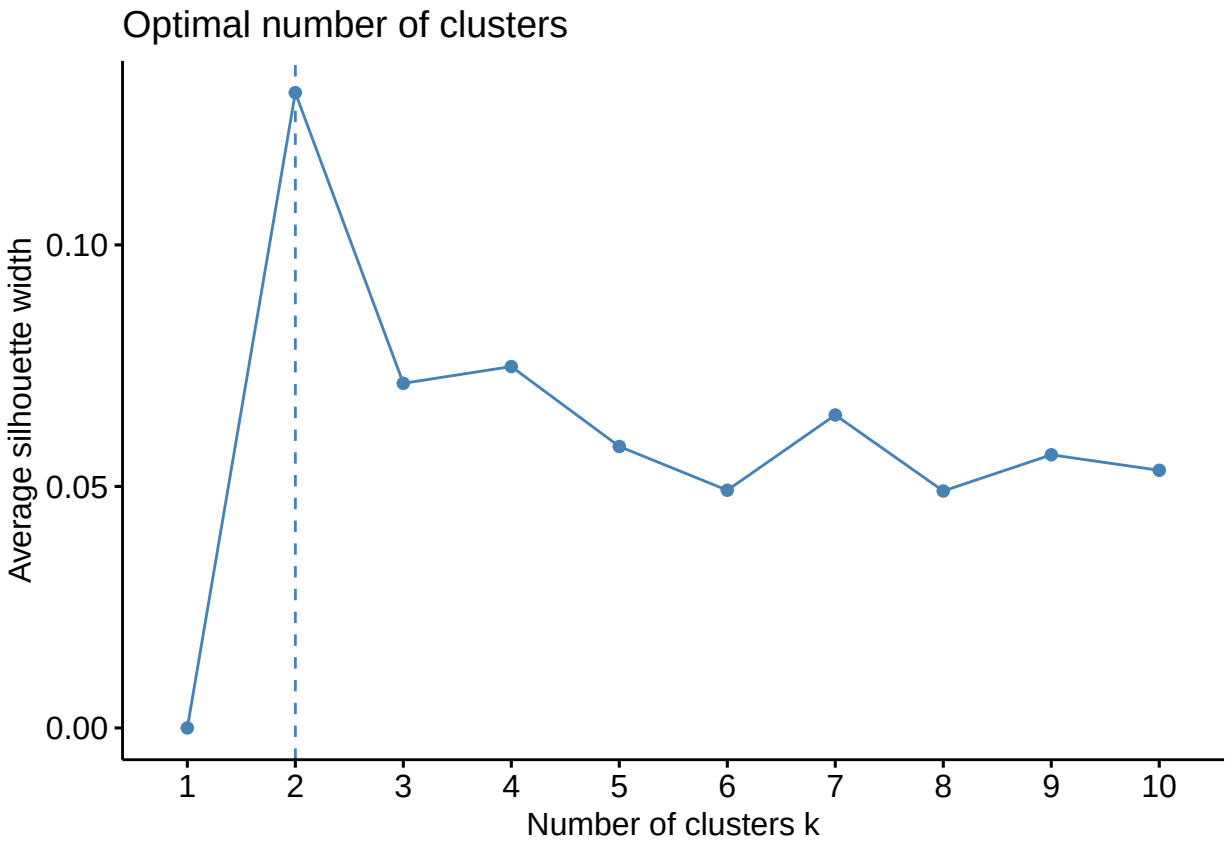
10. (10 points) Use the elbow method, average silhouette, and/or gap statistic to identify the optimal number of clusters based on k-means clustering with scaled features.

Both average silhouette and gap statistic identify the optimal number of clusters as 2. However, the results from elbow method is a little bit confusing. There is no obvious cutpoint followed by a linear curve. k of 2 or k of 3 seems to be the optimal options.

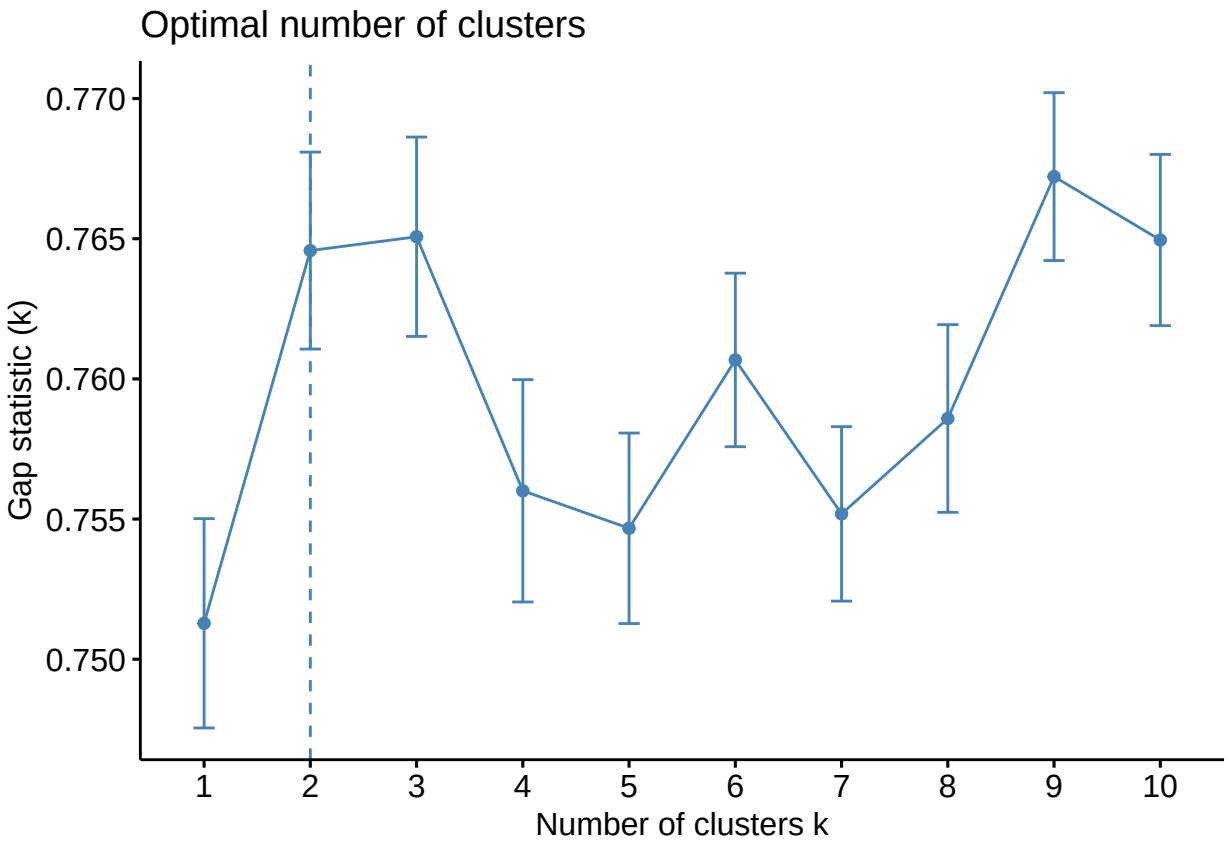
```
fviz_nbclust(wiki, kmeans, method = "wss")
```




```
fviz_nbclust(wiki, kmeans, method = "silhouette")
```



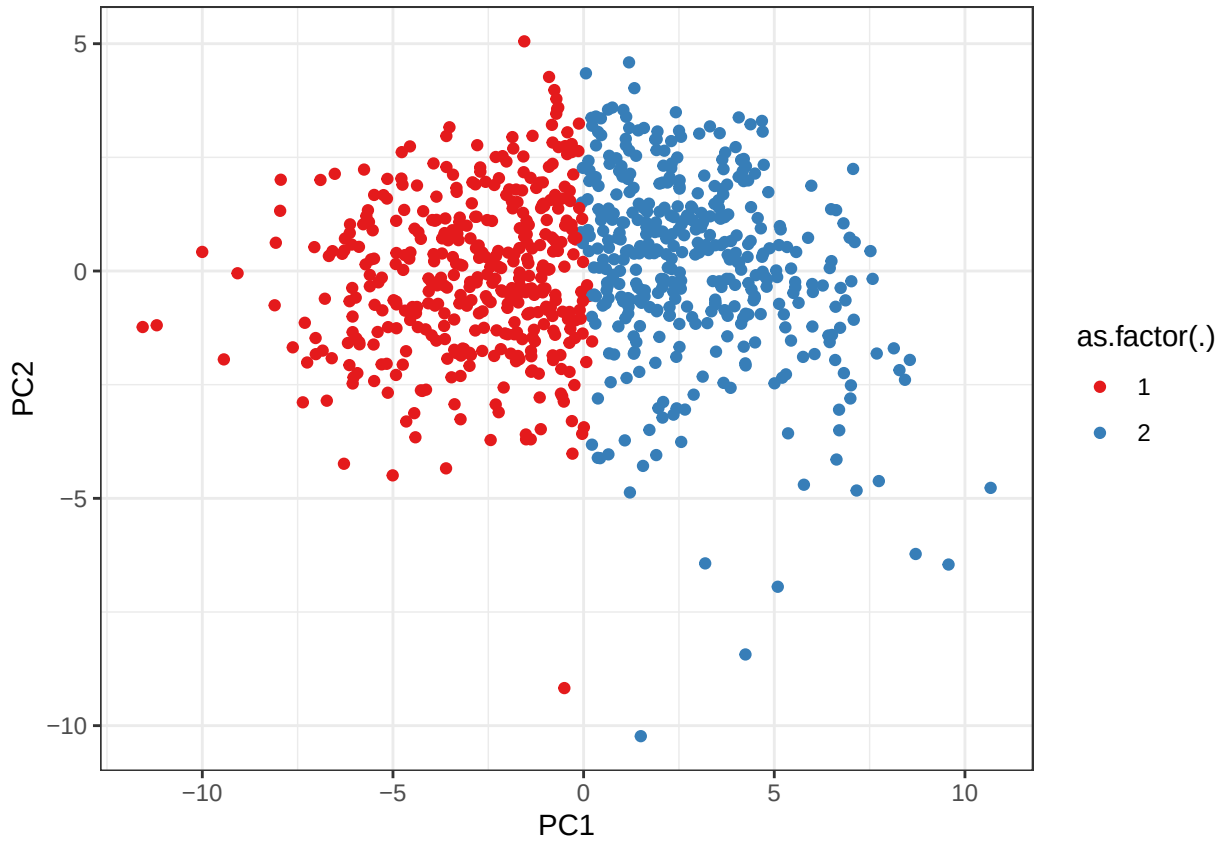

```
fviz_nbclust(wiki, kmeans, method = "gap_stat")
```



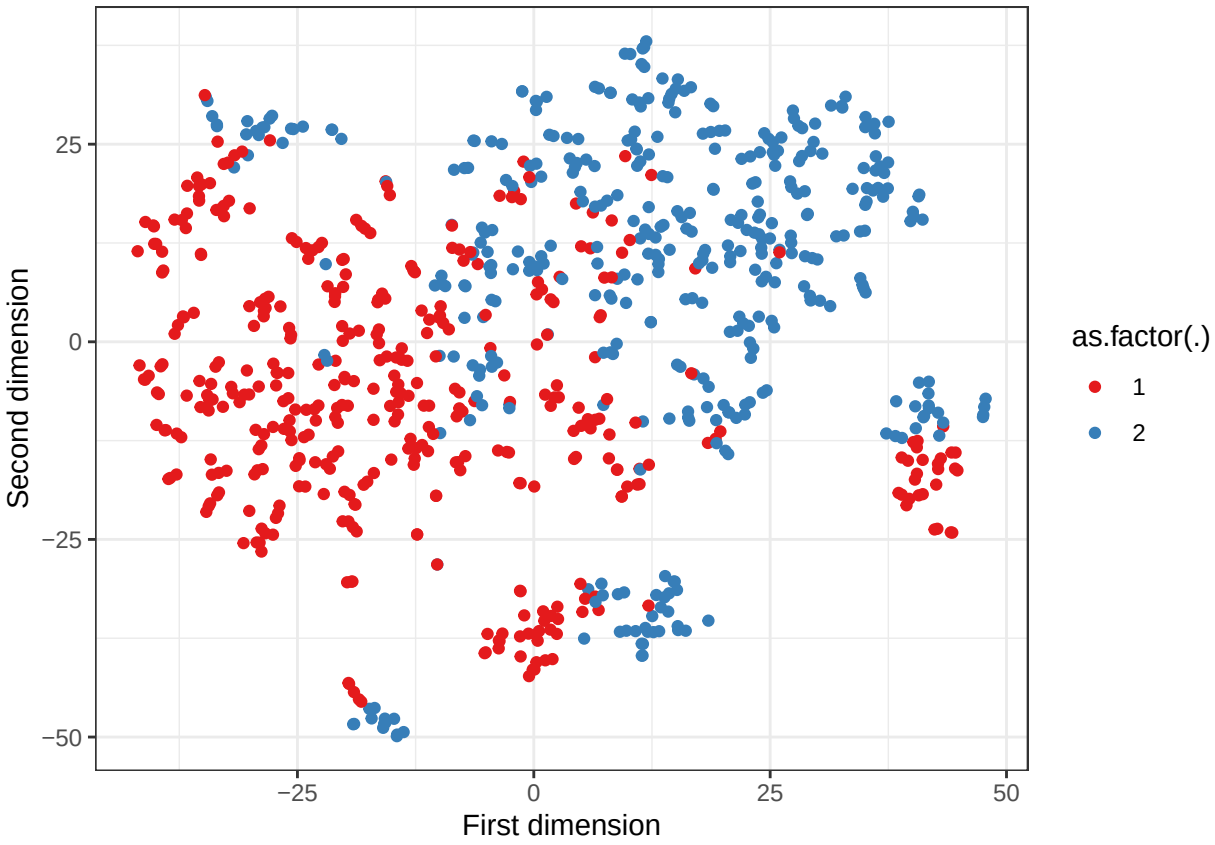
11. (15 points) Visualize the results of the optimal $\hat{k}$-means clustering model. **First** use the first and second principal components from PCA, and color-code each observation based on their cluster membership. **Next** use the first and second dimensions from t-SNE, and color-code each observation based on their cluster membership. **Describe your results. How do your interpretations differ between PCA and t-SNE?**

As shown in the two figures below, the PCA plot has a clearer and more global seperation with less overlappings than TSNE plot. Specifically, in the PCA plot, the kmeans clusters are seperated based on whether the first principal component is greater or less than 0. On the other hand, in the TSNE plot, there is one large cluster in the center with 4 smaller clusters on the peripheral, and the kmeans clustering results are equally assigned to each cluster groups. This observed phenomenon can be explained by the fact that TSNE focuses more on local structures by placing neighbors close to each other, while PCA emphasizes more on global structure.

```
kmeans(wiki, centers = 2)$cluster %>%
  as.data.frame() %>%
  bind_cols(as.data.frame(wiki_pca$x[,1:2])) %>%
  ggplot(aes(x = PC1, y = PC2, color = as.factor(.))) +
  scale_color_brewer(palette = "Set1") +
  geom_point()
```




```
kmeans(wiki, centers = 2)$cluster %>%
  as.data.frame() %>%
  bind_cols(as.data.frame(wiki_tsne$Y[,1:2])) %>%
  ggplot(aes(x = V1, y = V2, color = as.factor(.)))+
  geom_point() +
  scale_color_brewer(palette = "Set1") +
  labs(x = "First dimension",
       y = "Second dimension")
```


Second dimension
25
0
−25
−50
−25
0
25
50
First dimension
as.factor(.)
1
2