Problem Set 7

Pete Cuppernull

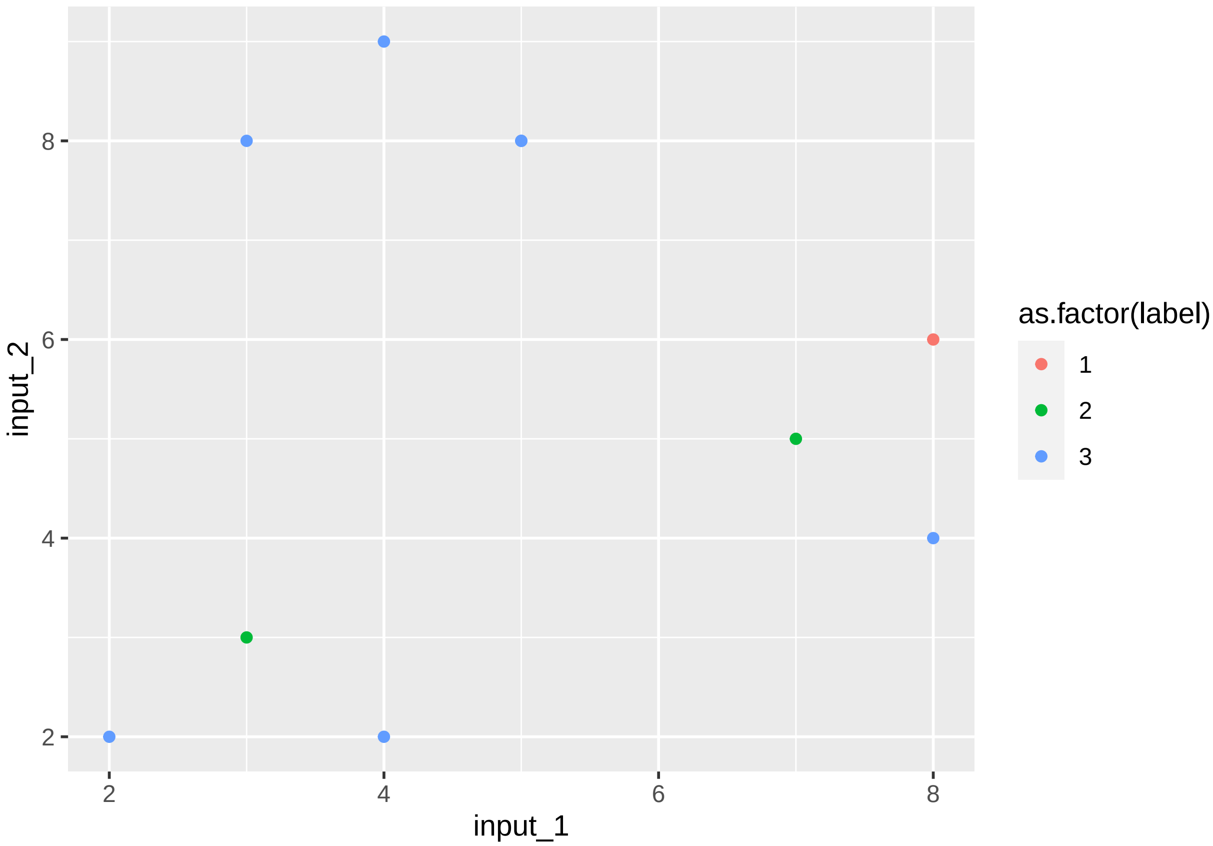
3/15/2020

```
library(tidyverse)
library(ggfortify)
library(tsne)
library(clValid)
```

K-Means by hand

1. Initialize process, assign random clusters

```
set.seed(10)
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)


#create assignments
label <- as.factor(sample(1:3, 10, replace=TRUE))

#assign labels to obs
x <- cbind(input_1, input_2, label)

ggplot(x, aes(input_1, input_2, color = as.factor(label))) +
  geom_point()
```



2. Write iterative function for centroids and assignments

```
#create function to make DF, determine cluster centroids,
#attach centroids to original df and assign label based on closest centroid
#return new dataframe, and then we can repeat this function until convergence
iterate <- function(original_df){
  x_df <- as.data.frame(x)

centroids <- x_df %>%
  group_by(label) %>%
  mutate(mean(input_1), mean(input_2)) %>%
  select(`mean(input_1)`, `mean(input_2)`) %>%
  distinct()

x1 <- centroids[1,2]
x2 <- centroids[2,2]
x3 <- centroids[3,2]
y1 <- centroids[1,3]
y2 <- centroids[2,3]
y3 <- centroids[3,3]
xs <- cbind(x1, x2, x3)
ys <- cbind(y1, y2, y3)
points <- cbind(xs, ys)
colnames(points) <- c("x1", "x2", "x3", "y1", "y2", "y3")
```

```r
df_cluster <- as.data.frame(cbind(x_df, points))

new_df <- df_cluster %>%
  mutate(label = if_else((((abs(input_1-x1) + abs(input_2-y1)) / 2) <
                           ((abs(input_1-x2) + abs(input_2-y2)) / 2)) &
                          (((abs(input_1-x1) + abs(input_2-y1)) / 2) <
                            ((abs(input_1-x3) + abs(input_2-y3)) / 2)), 1,
                          if_else((((abs(input_2-x2) + abs(input_2-y2)) / 2) <
                                    ((abs(input_1-x3) + abs(input_2-y3)) / 2)), 2, 3))) %>%
  select(input_1, input_2, label)

new_df
}

x2 <- iterate(x)
x == x2 #not converged :(
```

```
##       input_1 input_2 label
##  [1,]    TRUE    TRUE FALSE
##  [2,]    TRUE    TRUE FALSE
##  [3,]    TRUE    TRUE FALSE
##  [4,]    TRUE    TRUE  TRUE
##  [5,]    TRUE    TRUE FALSE
##  [6,]    TRUE    TRUE  TRUE
##  [7,]    TRUE    TRUE  TRUE
##  [8,]    TRUE    TRUE FALSE
##  [9,]    TRUE    TRUE FALSE
## [10,]    TRUE    TRUE FALSE
```

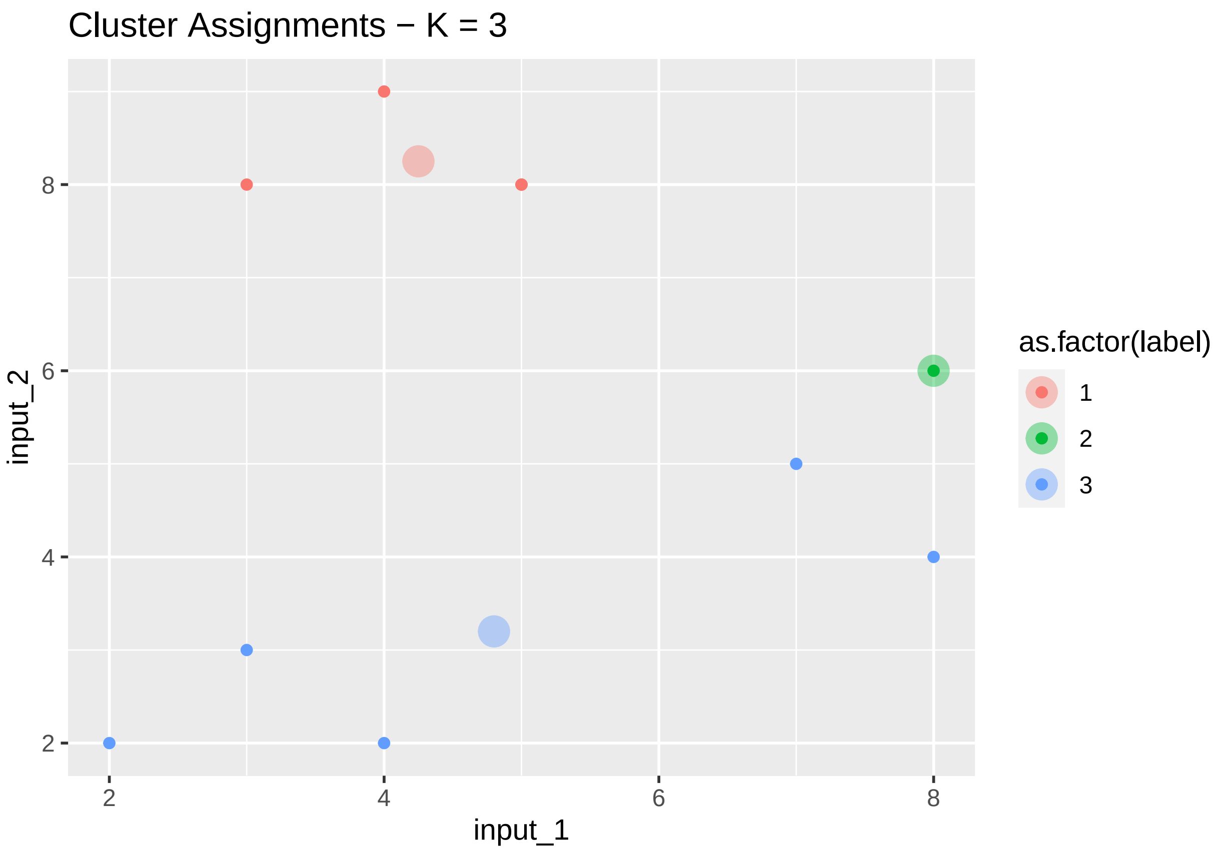
```r
x3 <- iterate(x2)
x2 == x3 #converged! :)
```

```
##       input_1 input_2 label
##  [1,]    TRUE    TRUE  TRUE
##  [2,]    TRUE    TRUE  TRUE
##  [3,]    TRUE    TRUE  TRUE
##  [4,]    TRUE    TRUE  TRUE
##  [5,]    TRUE    TRUE  TRUE
##  [6,]    TRUE    TRUE  TRUE
##  [7,]    TRUE    TRUE  TRUE
##  [8,]    TRUE    TRUE  TRUE
##  [9,]    TRUE    TRUE  TRUE
## [10,]    TRUE    TRUE  TRUE
```

3. Visualize cluster assignments

```r
final_centroids <- as.data.frame(x3) %>%
  group_by(label) %>%
  mutate(mean(input_1), mean(input_2)) %>%
  select(`mean(input_1)`, `mean(input_2)`) %>%
  distinct()

clusters_plot <- ggplot() +
```

```
  geom_point(data = as.data.frame(x3),
             mapping = aes(input_1, input_2, color = as.factor(label))) +
  geom_point(data = final_centroids,
             mapping = aes(`mean(input_1)`, `mean(input_2)`, color = as.factor(label)),
             size = 5,
             alpha = .4)  +
  labs(title = "Cluster Assignments - K = 3")

clusters_plot
```



4. Repeat process for k = 2

```
iterate2 <- function(original_df){
  x_df <- as.data.frame(x)

centroids <- x_df %>%
  group_by(label) %>%
  mutate(mean(input_1), mean(input_2)) %>%
  select(`mean(input_1)`, `mean(input_2)`) %>%
  distinct()

x1 <- centroids[1,2]
x2 <- centroids[2,2]
y1 <- centroids[1,3]
y2 <- centroids[2,3]
```

```r
xs <- cbind(x1, x2)
ys <- cbind(y1, y2)
points <- cbind(xs, ys)
colnames(points) <- c("x1", "x2", "y1", "y2")

df_cluster <- as.data.frame(cbind(x_df, points))

new_df <- df_cluster %>%
  mutate(label = if_else((((abs(input_1-x1) + abs(input_2-y1)) / 2) <
                          ((abs(input_1-x2) + abs(input_2-y2)) / 2)), 1, 2)) %>%
  select(input_1, input_2, label)

new_df
}

x2b <- iterate2(x)
x == x2b
```

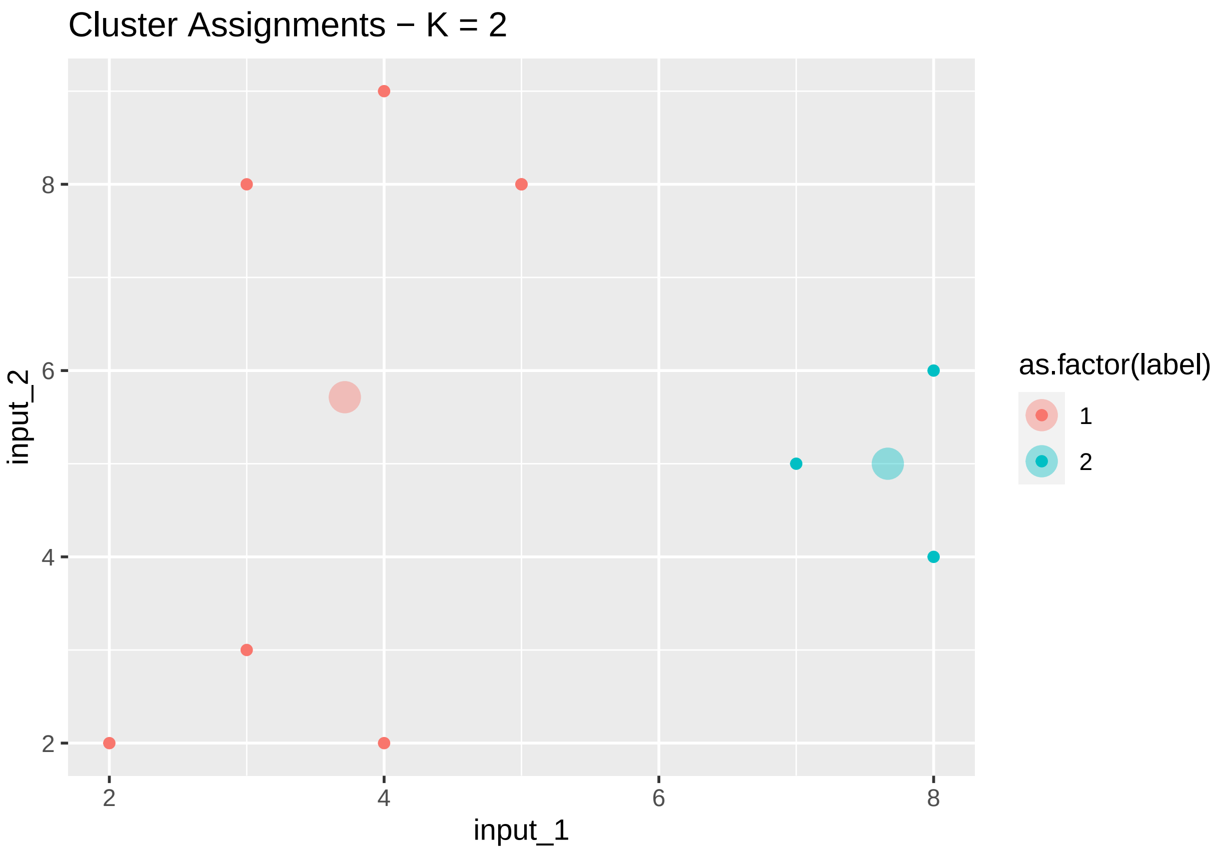
```
##       input_1 input_2 label
##  [1,]    TRUE    TRUE FALSE
##  [2,]    TRUE    TRUE FALSE
##  [3,]    TRUE    TRUE  TRUE
##  [4,]    TRUE    TRUE FALSE
##  [5,]    TRUE    TRUE FALSE
##  [6,]    TRUE    TRUE FALSE
##  [7,]    TRUE    TRUE FALSE
##  [8,]    TRUE    TRUE FALSE
##  [9,]    TRUE    TRUE FALSE
## [10,]    TRUE    TRUE FALSE
```

```r
x3b <- iterate2(x2b)
x2b == x3b
```

```
##       input_1 input_2 label
##  [1,]    TRUE    TRUE  TRUE
##  [2,]    TRUE    TRUE  TRUE
##  [3,]    TRUE    TRUE  TRUE
##  [4,]    TRUE    TRUE  TRUE
##  [5,]    TRUE    TRUE  TRUE
##  [6,]    TRUE    TRUE  TRUE
##  [7,]    TRUE    TRUE  TRUE
##  [8,]    TRUE    TRUE  TRUE
##  [9,]    TRUE    TRUE  TRUE
## [10,]    TRUE    TRUE  TRUE
```

```r
final_centroids2 <- as.data.frame(x3b) %>%
  group_by(label) %>%
  mutate(mean(input_1), mean(input_2)) %>%
  select(`mean(input_1)`, `mean(input_2)`) %>%
  distinct()

ggplot() +
  geom_point(data = as.data.frame(x3b),
             mapping = aes(input_1, input_2, color = as.factor(label))) +
  geom_point(data = final_centroids2,
```

```
            mapping = aes(`mean(input_1)`, `mean(input_2)`, color = as.factor(label)),
            size = 5,
            alpha = .4) +
  labs(title = "Cluster Assignments - K = 2")
```



5. Discussion

Our initial hunch of 3 clusters did pan out, as the mean distance between points and the cluster centroids is small relative to only using 2 clusters. This is also visually represented in the two plots, where the centroids and groups of data are more intuitively clustered together in the k = 3 plot. **Note: while it appears as if the centroid for Label 1 is off center in the k = 3 plot, the reason for this is that data points 1 and 10 are in the exact same place, which is visually skewing the position of that cluster centroid.

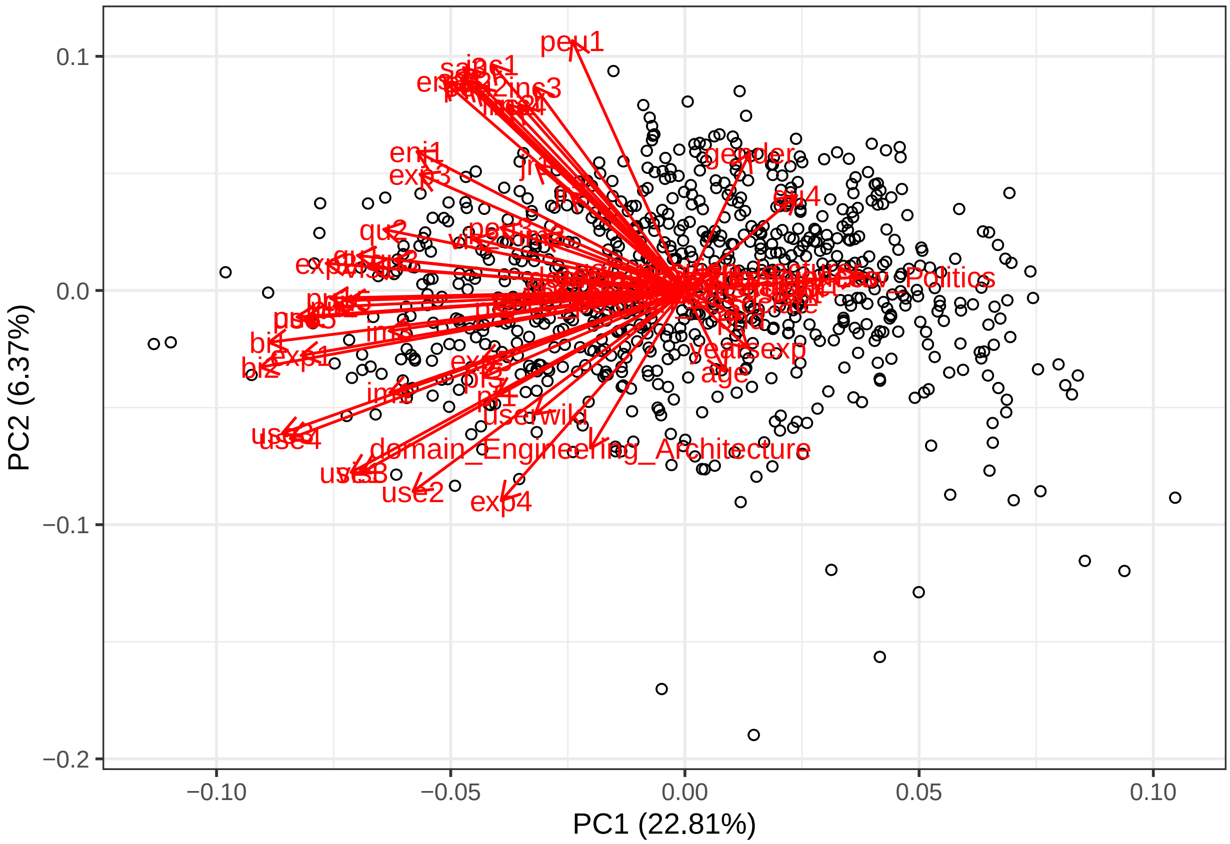
Application

Dimension Reduction

6. Perform PCA

```
wiki <- read_csv("/Users/petecuppernull/Dropbox/UChicago/2019-20/Winter/Computatonal Modeling/Repos/Prol

wiki_fit <- prcomp(wiki,
                   scale = TRUE)
```

```r
# visualize
autoplot(wiki_fit,
         shape = TRUE, # names instead of points
         loadings.label = TRUE) + # show the loading directions
theme_bw()
```



The variables that appear strongly correlated with the first component are exp4 (faculty who "contribute to Wikipedia") and faculty who work in engineering and architecture. The second principle component appears to capture direct opinions about the use of Wikipedia for teaching: these variables include bi1 and bi2 (recommending the use of Wikipedia to colleagues and students and intending to use Wikipedia for teaching in the future, respectively), as well as use3 and use4 (currently recommending to colleagues to use Wikipedia and knowing that their students currently use it, respectively).

7. Calculate PVE

```r
summary(wiki_fit)
```

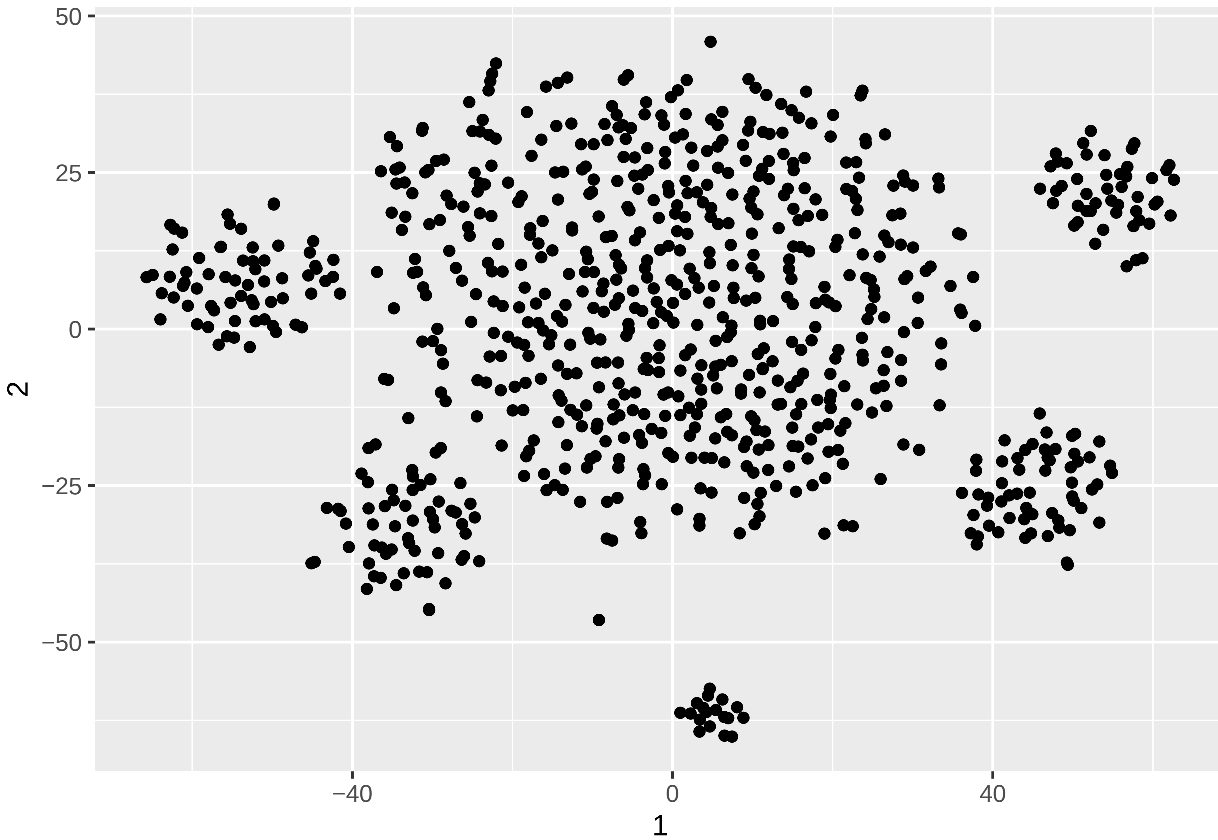
```
## Importance of components:
##                           PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     3.6058 1.90586 1.69219 1.52365 1.46547 1.38228 1.31487
## Proportion of Variance 0.2281 0.06372 0.05024 0.04073 0.03768 0.03352 0.03033
## Cumulative Proportion  0.2281 0.29183 0.34207 0.38280 0.42047 0.45399 0.48433
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
```

```
## Standard deviation     1.20613 1.17385 1.16779 1.13641 1.08654 1.07515 1.04158
## Proportion of Variance 0.02552 0.02417 0.02393 0.02266 0.02071 0.02028 0.01903
## Cumulative Proportion  0.50985 0.53402 0.55795 0.58060 0.60132 0.62160 0.64063
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     1.01080 0.99808 0.99197 0.96071 0.93339 0.91156 0.90379
## Proportion of Variance 0.01792 0.01748 0.01726 0.01619 0.01528 0.01458 0.01433
## Cumulative Proportion  0.65855 0.67603 0.69329 0.70949 0.72477 0.73935 0.75368
##                          PC22    PC23    PC24    PC25    PC26    PC27    PC28
## Standard deviation     0.8771 0.85951 0.82448 0.80853 0.80231 0.78661 0.75050
## Proportion of Variance 0.0135 0.01296 0.01193 0.01147 0.01129 0.01086 0.00988
## Cumulative Proportion  0.7672 0.78014 0.79206 0.80353 0.81482 0.82568 0.83556
##                           PC29    PC30    PC31   PC32    PC33   PC34    PC35
## Standard deviation     0.73659 0.70268 0.70157 0.6878 0.68203 0.6708 0.64653
## Proportion of Variance 0.00952 0.00866 0.00864 0.0083 0.00816 0.0079 0.00733
## Cumulative Proportion  0.84508 0.85374 0.86238 0.8707 0.87884 0.8867 0.89407
##                           PC36    PC37    PC38    PC39    PC40    PC41    PC42
## Standard deviation     0.64385 0.62790 0.62332 0.61367 0.59686 0.57617 0.57549
## Proportion of Variance 0.00727 0.00692 0.00682 0.00661 0.00625 0.00582 0.00581
## Cumulative Proportion  0.90134 0.90826 0.91507 0.92168 0.92793 0.93375 0.93956
##                          PC43    PC44    PC45    PC46    PC47   PC48    PC49
## Standard deviation     0.5650 0.55613 0.55423 0.54026 0.53701 0.5231 0.51546
## Proportion of Variance 0.0056 0.00543 0.00539 0.00512 0.00506 0.0048 0.00466
## Cumulative Proportion  0.9452 0.95059 0.95598 0.96110 0.96616 0.9710 0.97562
##                           PC50    PC51    PC52    PC53    PC54    PC55    PC56
## Standard deviation     0.50816 0.49831 0.46804 0.46300 0.43911 0.36615 0.33486
## Proportion of Variance 0.00453 0.00436 0.00384 0.00376 0.00338 0.00235 0.00197
## Cumulative Proportion  0.98015 0.98451 0.98835 0.99211 0.99549 0.99784 0.99981
##                           PC57
## Standard deviation     0.10351
## Proportion of Variance 0.00019
## Cumulative Proportion  1.00000
```

Approximately 29.2% of the variance is expalined by the first two components.

8. T-SNE

```r
wiki_scaled <- scale(wiki)

tsne <- tsne(wiki_scaled, k = 2)

tsne %>%
  ggplot(aes(`1`, `2`)) +
  geom_point()
```



Here, we see distinct clustering in the data. There appear to be five smaller, well-defined clusters, and one broad loosely correlated cluster. This would suggest that there are certain subsets of faculty which have distinct views on using Wikipedia in the classroom.

Clustering

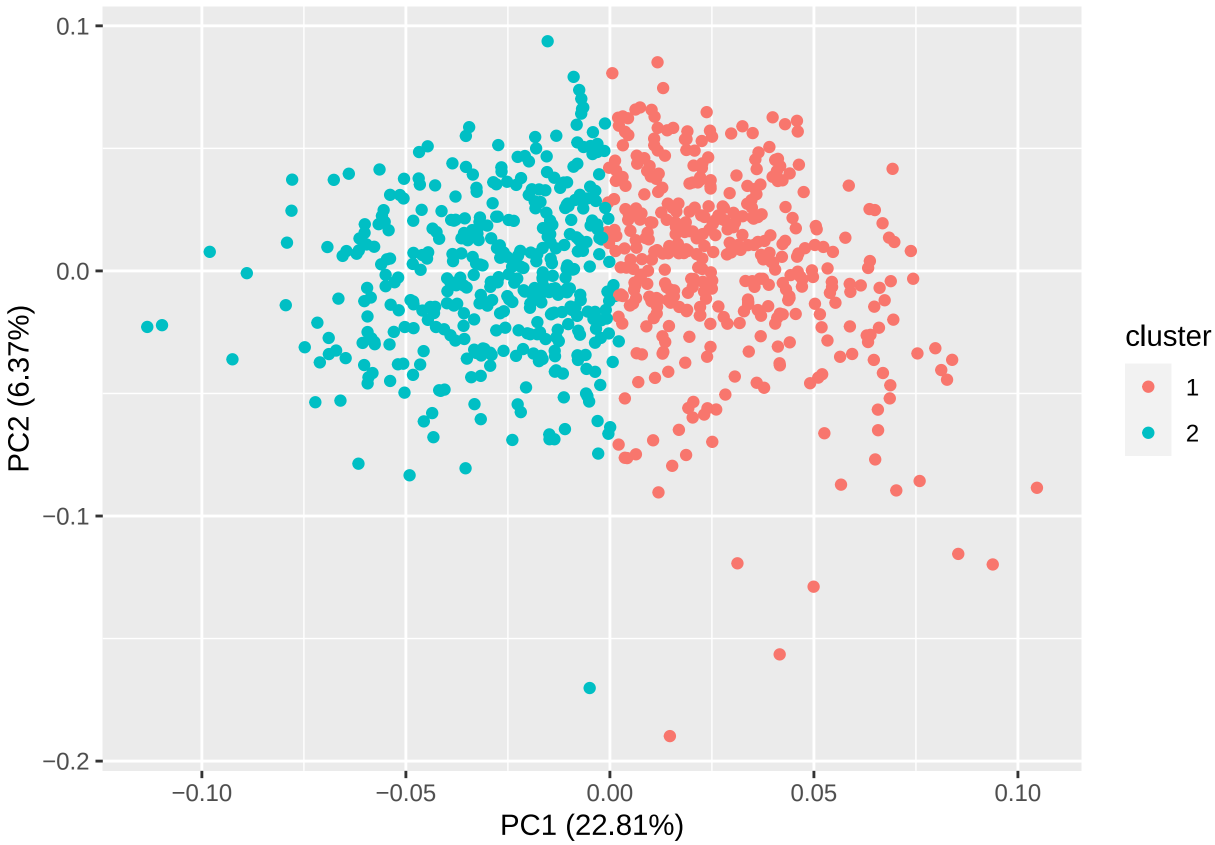
9. Create K-Means Models

```
#2 clusters
kmeans2 <- kmeans(wiki_scaled,
                  centers = 2,
                  nstart = 15)

kmeans3 <- kmeans(wiki_scaled,
                  centers = 3,
                  nstart = 15)

kmeans4 <- kmeans(wiki_scaled,
                  centers = 4,
                  nstart = 15)
```

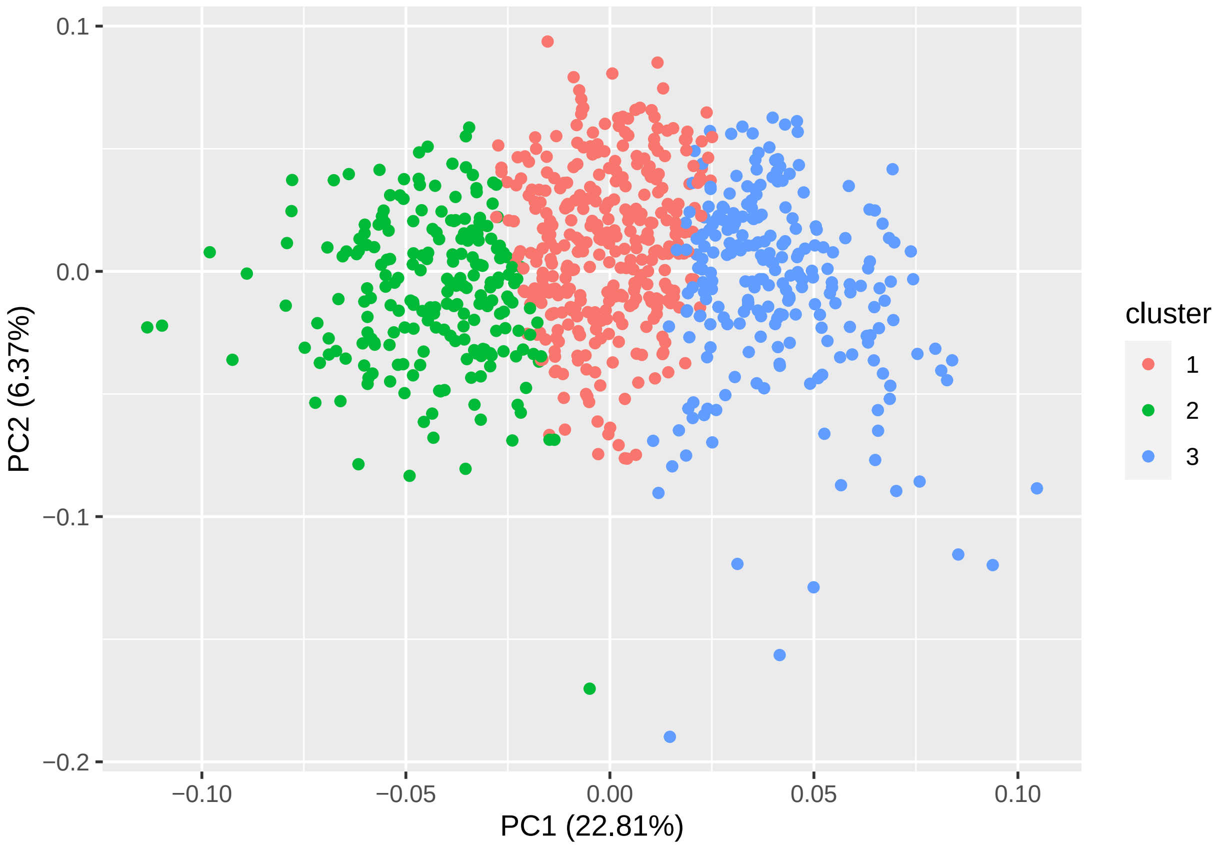
Now map cluster assignments onto plots

```
##Two Clusters
wiki_kmeans2 <- wiki_scaled %>%
  cbind(as.data.frame(kmeans2$cluster)) %>%
  rename(cluster = `kmeans2$cluster`) %>%
```

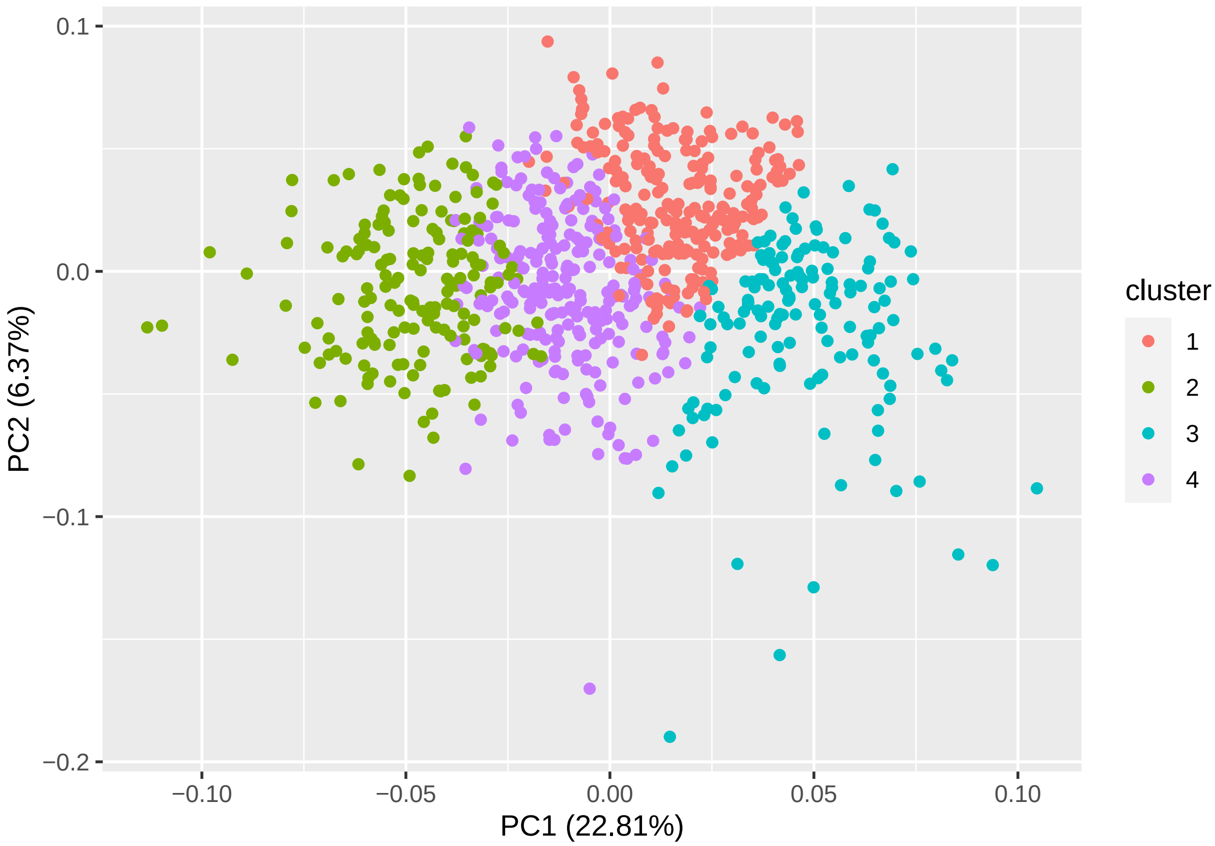
```r
  mutate(cluster = as.factor(cluster))

autoplot(prcomp(as.data.frame(wiki_scaled)),
         data = wiki_kmeans2,
         colour = 'cluster'
         )
```




```r
##Three Clusters
wiki_kmeans3 <- wiki_scaled %>%
  cbind(as.data.frame(kmeans3$cluster)) %>%
  rename(cluster = `kmeans3$cluster`) %>%
  mutate(cluster = as.factor(cluster))

autoplot(prcomp(as.data.frame(wiki_scaled)),
         data = wiki_kmeans3,
         colour = 'cluster'
         )
```




```
##Four Clusters
wiki_kmeans4 <- wiki_scaled %>%
  cbind(as.data.frame(kmeans4$cluster)) %>%
  rename(cluster = `kmeans4$cluster`) %>%
  mutate(cluster = as.factor(cluster))

autoplot(prcomp(as.data.frame(wiki_scaled)),
         data = wiki_kmeans4,
         colour = 'cluster'
         )
```

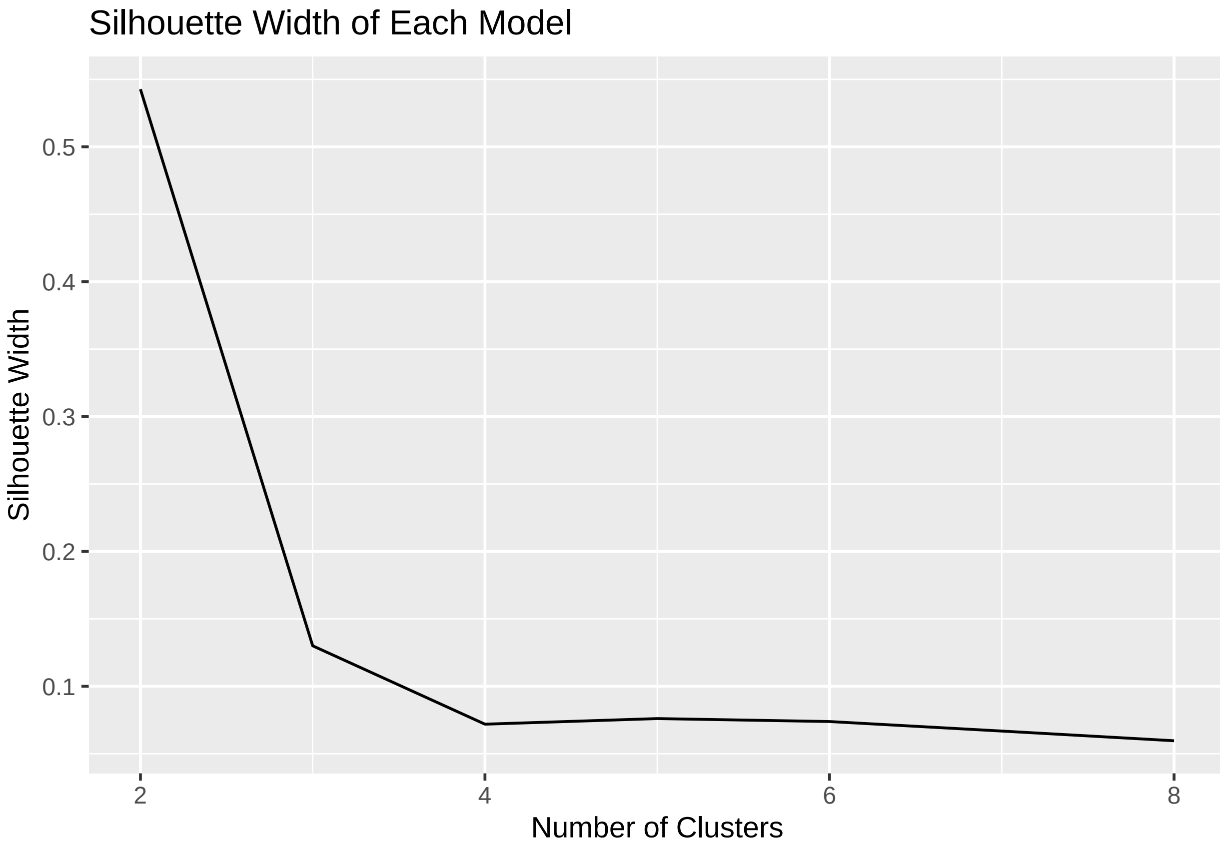


We can see that the k-means clustering creating results that spoke to similar variance in the data as did the PCA. In the plot using two clusters, k-means split the data nearly perfectly down the center of the first principle component. The model with three clusters similarly shows very distinct classifications; the model with four clusters, however, begins to show more of a fuzzy boundary between the clusters, potentially suggesting that the optimal number of clusters for the most intuitive classification is two or three.

10. Identify optimum number of clusters - silhouette width

```
#sample half of DF so doc will knit
wiki_scaled_samp <- wiki_scaled[sample(nrow(wiki_scaled),size=400,replace=FALSE),]
kmeans_valid <- clValid(as.matrix(wiki_scaled_samp),
        nClust = 2:8,
        clMethods = "kmeans",
        validation = "internal")

km_df <- as.data.frame(measures(kmeans_valid))

colnames(km_df) <- c("2", "3", "4", "5", "6", "7","8")

km_final_df <- km_df %>%
  rownames_to_column() %>%
  gather(key = "clusters",
         value = "score",
         2:8) %>%
  mutate(model = "kmeans") %>%
  mutate(clusters = as.numeric(clusters))
```

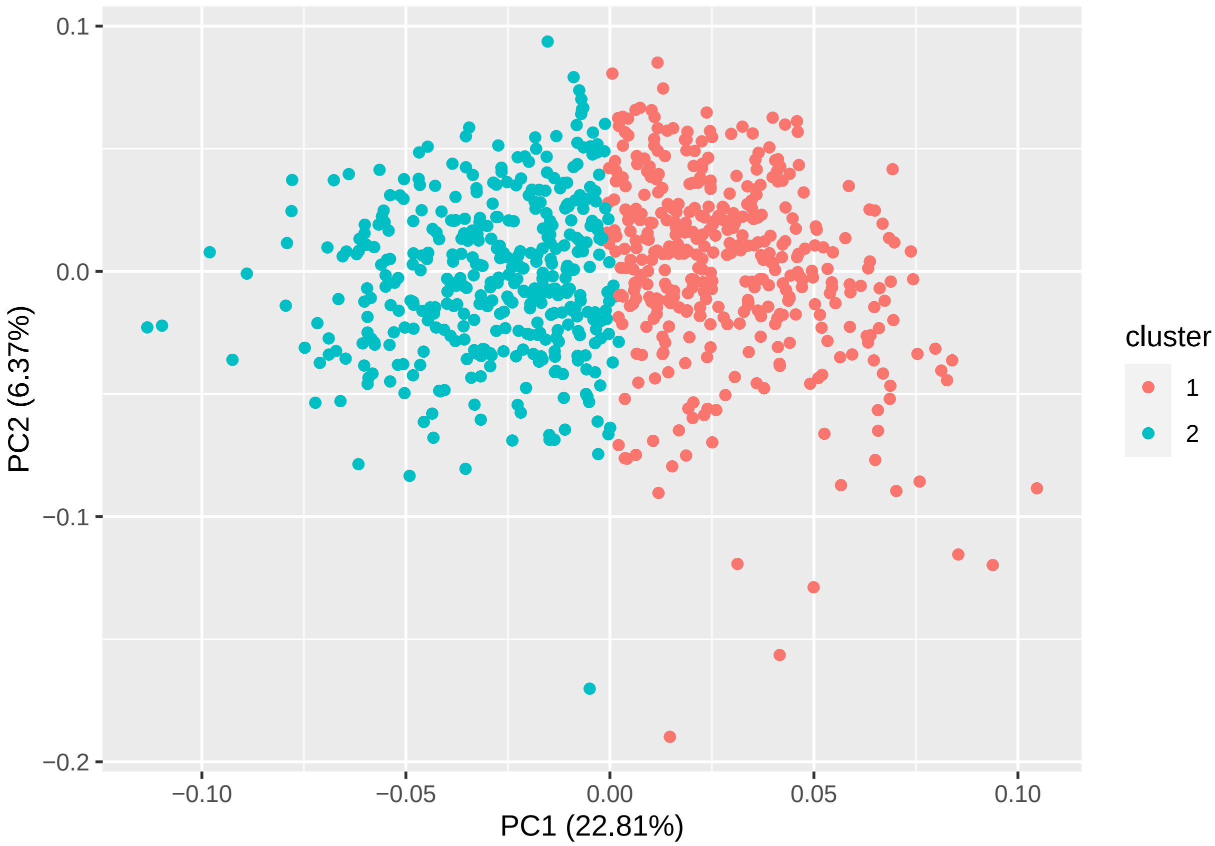
```
km_final_df %>%
  filter(rowname == "Silhouette") %>%
ggplot(aes(clusters, score)) +
  geom_line() +
  labs(title = "Silhouette Width of Each Model",
       x = "Number of Clusters",
       y = "Silhouette Width")
```



The Silhouette Width validation shows that two clusters is the optimal number, as measured strictly by how well-matched the cluster is to the within-cluster data. This insight was also present in the visuals plots, which showed a strong distinction between the two classes.

11. Visualize optimal k-means (2 clusters)

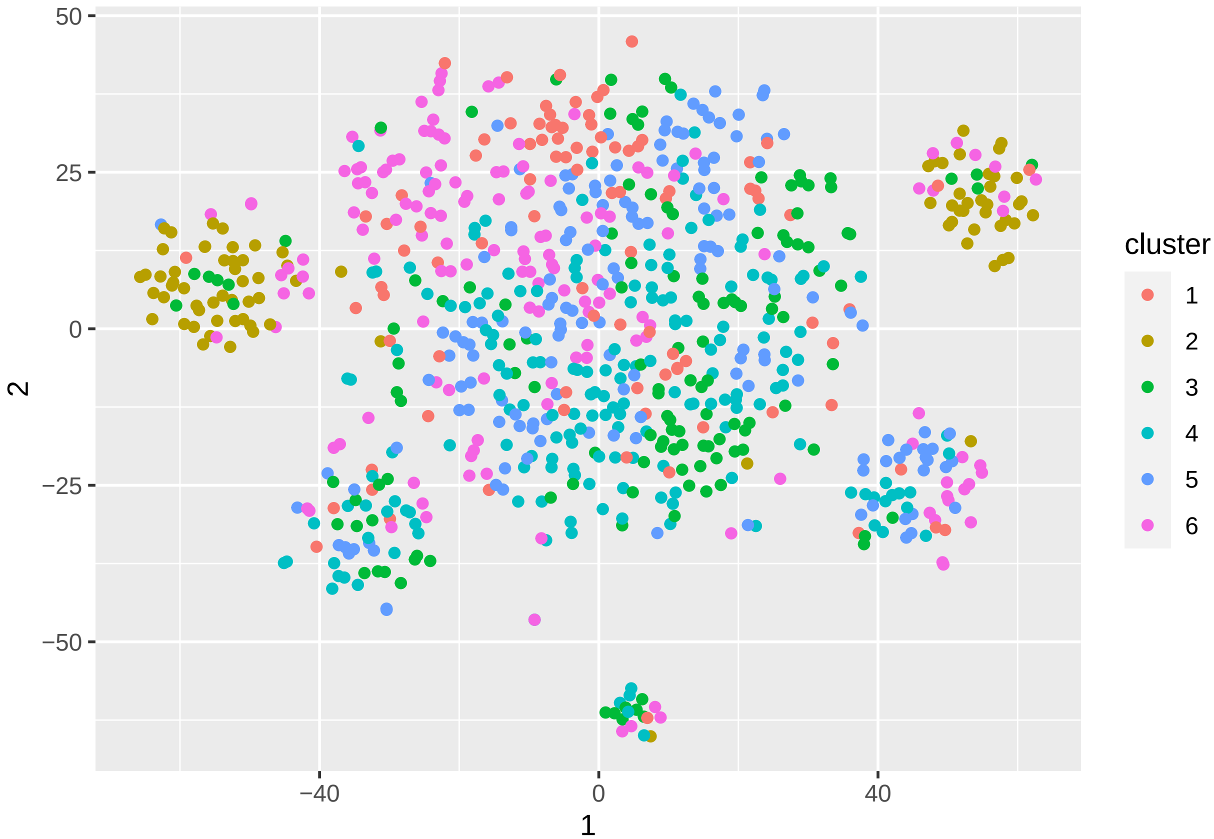
```
##PCA
autoplot(prcomp(as.data.frame(wiki_scaled)),
         data = wiki_kmeans2,
         colour = 'cluster'
         )
```




```
##t-SNE
tsne %>%
  cbind(as.data.frame(kmeans2$cluster)) %>%
  rename(cluster = `kmeans2$cluster`) %>%
  mutate(cluster = as.factor(cluster)) %>%
  ggplot(aes(`1`, `2`, color = cluster)) +
  geom_point()
```

There is a significant difference in interpretation between the PCA and t-SNE results. The PCA results appear to be mirroring the k-means clusters – the observations in the PCA plot form a single cloud, and k-means clustering splits that cloud in half. With t-SNE however, there appears to be six distinct clouds, and so it is unsurprising that the k-means cluster assignments (for two clusters) appear to be mixed within the cluster assignments from t-SNE. As an additional exercise, below I plot the same t-SNE results with k-means for six clusters.

```
kmeans6 <- kmeans(wiki_scaled,
                  centers = 6,
                  nstart = 15)

tsne %>%
  cbind(as.data.frame(kmeans6$cluster)) %>%
  rename(cluster = `kmeans6$cluster`) %>%
  mutate(cluster = as.factor(cluster)) %>%
  ggplot(aes(`1`, `2`, color = cluster)) +
  geom_point()
```



Here, we still do not see any intuitive matching between the k-means cluster assignments and the t-SNE groupings, suggesting that t-SNE is capturing different variation from both k-means and PCA. It would require further investigation to parse out the exact variation captured by t-SNE.