SDM2

Drangapu

2024-04-02

1) **The "chorSub" data from the "cluster" package contains measurements of 10 chemicals in 61 geological samples from the Kola Peninsula. Cluster the data using k-means and hierarchical clustering. What is a good choice of "k" for each of these methods? Justify your selection**

```
if (!require(cluster)) {
  install.packages("cluster")
}
```

```
## Loading required package: cluster
```

```
library(cluster)
data("chorSub")
dats<-chorSub[]
```

Exploratory Data Analysis (EDA)

```
str(dats)
```

```
##  int [1:61, 1:10] 101 50 5 -40 -13 -49 44 285 4 -48 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:61] "190" "191" "192" "193" ...
##   ..$ : chr [1:10] "Al" "Ca" "Fe" "K" ...
```

```
summary(dats)
```

```
##        Al                Ca                Fe                K          
##  Min.   :-201.000   Min.   :-178.000   Min.   :-200.00   Min.   :-133.000  
##  1st Qu.: -49.000   1st Qu.: -59.000   1st Qu.: -75.00   1st Qu.: -74.000  
##  Median :  -1.000   Median :   2.000   Median : -37.00   Median : -17.000  
##  Mean   :   0.541   Mean   :  -2.066   Mean   : -14.64   Mean   :  -4.295  
##  3rd Qu.:  47.000   3rd Qu.:  59.000   3rd Qu.:  48.00   3rd Qu.:  51.000  
##  Max.   : 285.000   Max.   : 211.000   Max.   : 162.00   Max.   : 248.000  
##        Mg                Mn                Na                P          
##  Min.   :-155.00   Min.   :-139.00   Min.   :-242.00   Min.   :-102.000  
##  1st Qu.: -75.00   1st Qu.: -66.00   1st Qu.: -25.00   1st Qu.: -61.000  
##  Median : -30.00   Median : -41.00   Median :  47.00   Median : -36.000  
##  Mean   : -13.05   Mean   : -13.92   Mean   :  17.97   Mean   :  -6.623  
##  3rd Qu.:  31.00   3rd Qu.:   7.00   3rd Qu.:  81.00   3rd Qu.:   8.000  
##  Max.   : 254.00   Max.   : 354.00   Max.   : 187.00   Max.   : 406.000
```

```
##        Si                 Ti
##  Min.   :-223.000   Min.   :-190.000
##  1st Qu.: -46.000   1st Qu.: -73.000
##  Median :  22.000   Median : -24.000
##  Mean   :   8.328   Mean   :  -7.361
##  3rd Qu.:  60.000   3rd Qu.:  55.000
##  Max.   : 177.000   Max.   : 351.000
```

```
head(dats)
```

```
##      Al  Ca   Fe    K   Mg   Mn  Na   P  Si   Ti
## 190 101  11  -22  -17  -34  -41  27 -36 -58  -28
## 191  50 129   23  -82   47   33  61  90 -24    9
## 192   5  65  -22  -96  -33    7  47  49  30  -39
## 193 -40 -16 -158  -70 -104 -114  53 -61 103 -160
## 194 -13  30  -82 -113   26  -41  65 -90  43 -130
## 195 -49 -43   31  -74   -2   33 -72 -36  78  -51
```

```
dim(dats)
```

```
## [1] 61 10
```

Clustering the data using kmeans and hierarchical clustering

k-means Clustering

```
k <- 5
set.seed(123)
km<-kmeans(chorSub,centers=k,nstart=10)
print(km)
```

```
## K-means clustering with 5 clusters of sizes 10, 15, 11, 3, 22
##
## Cluster means:
##          Al         Ca          Fe         K        Mg        Mn         Na
## 1  39.00000 -115.40000   75.500000  67.10000  -7.50000  30.30000 -133.00000
## 2 -70.86667  -56.40000 -106.066667  14.73333 -88.06667 -85.66667   29.80000
## 3  47.54545  119.81818   78.000000 -64.27273  99.27273  39.54545   36.00000
## 4  53.66667  -14.00000    9.666667 212.66667 -86.33333 132.00000  118.33333
## 5   1.00000   27.18182  -42.909091 -49.31818 -10.59091 -31.72727   55.81818
##           P        Si        Ti
## 1 -20.70000 -12.00000  66.20000
## 2 -57.80000  93.13333 -87.33333
## 3  54.45455 -94.18182  59.54545
## 4 176.66667 -83.33333 193.33333
## 5 -20.86364  23.50000 -47.09091
##
## Clustering vector:
## 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209
##   5   3   5   2   5   5   1   3   2   3   1   1   2   4   2   1   1   5   3   5
## 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229
##   2   1   4   5   5   5   3   5   5   5   1   1   2   4   2   3   5   3   5   5
```

```
## 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249
##   2   2   2   5   2   5   5   5   2   1   3   5   1   5   5   3   3   2   2   3
## 250
##   2
##
## Within cluster sum of squares by cluster:
## [1] 366692.1 369687.3 414822.5 211533.3 463456.7
##  (between_SS / total_SS =  58.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
km$cluster
```

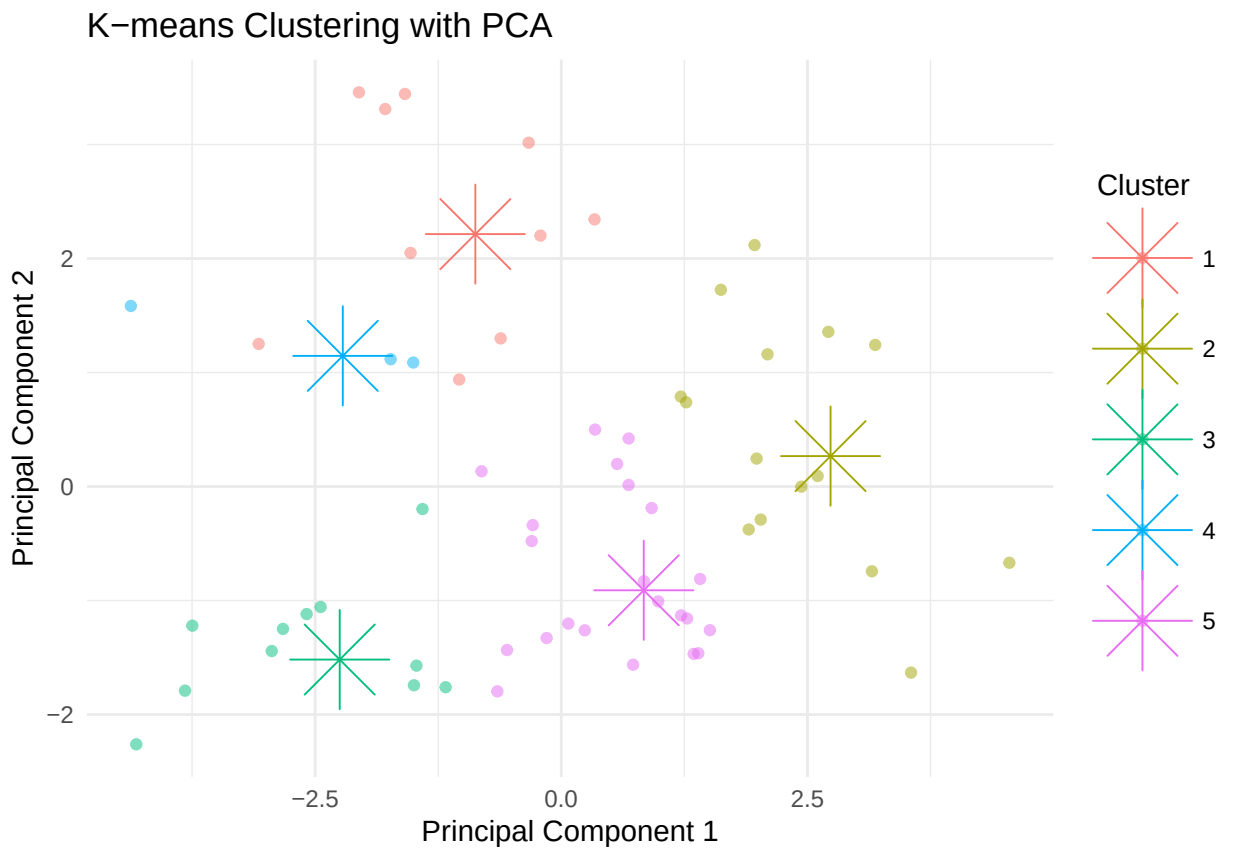
```
## 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209
##   5   3   5   2   5   5   1   3   2   3   1   1   2   4   2   1   1   5   3   5
## 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229
##   2   1   4   5   5   5   3   5   5   5   1   1   2   4   2   3   5   3   5   5
## 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249
##   2   2   2   5   2   5   5   5   2   1   3   5   1   5   5   3   3   2   2   3
## 250
##   2
```

```
km$centers
```

```
##          Al         Ca          Fe         K        Mg        Mn         Na
## 1  39.00000 -115.40000   75.500000  67.10000  -7.50000  30.30000 -133.00000
## 2 -70.86667  -56.40000 -106.066667  14.73333 -88.06667 -85.66667   29.80000
## 3  47.54545  119.81818   78.000000 -64.27273  99.27273  39.54545   36.00000
## 4  53.66667  -14.00000    9.666667 212.66667 -86.33333 132.00000  118.33333
## 5   1.00000   27.18182  -42.909091 -49.31818 -10.59091 -31.72727   55.81818
##           P        Si        Ti
## 1 -20.70000 -12.00000  66.20000
## 2 -57.80000  93.13333 -87.33333
## 3  54.45455 -94.18182  59.54545
## 4 176.66667 -83.33333 193.33333
## 5 -20.86364  23.50000 -47.09091
```

```
pca_scl <- prcomp(scale(dats), center = TRUE)
pca_data <- data.frame(pca_scl$x[,1:2])
pca_centers <- predict(pca_scl, newdata = km$centers)
pca_data$cluster <- as.factor(km$cluster)
pca_centers <- as.data.frame(pca_centers)
pca_centers$cluster <- as.factor(1:nrow(pca_centers))
```

```
library(ggplot2)
ggplot(pca_data, aes(x = PC1, y = PC2, color = cluster)) +
  geom_point(alpha = 0.5) +geom_point(data = pca_centers, aes(x = PC1, y = PC2, color = cluster),size =
  theme_minimal() +
  labs(title = "K-means Clustering with PCA",x = "Principal Component 1",
       y = "Principal Component 2") + scale_color_discrete(name = "Cluster")
```



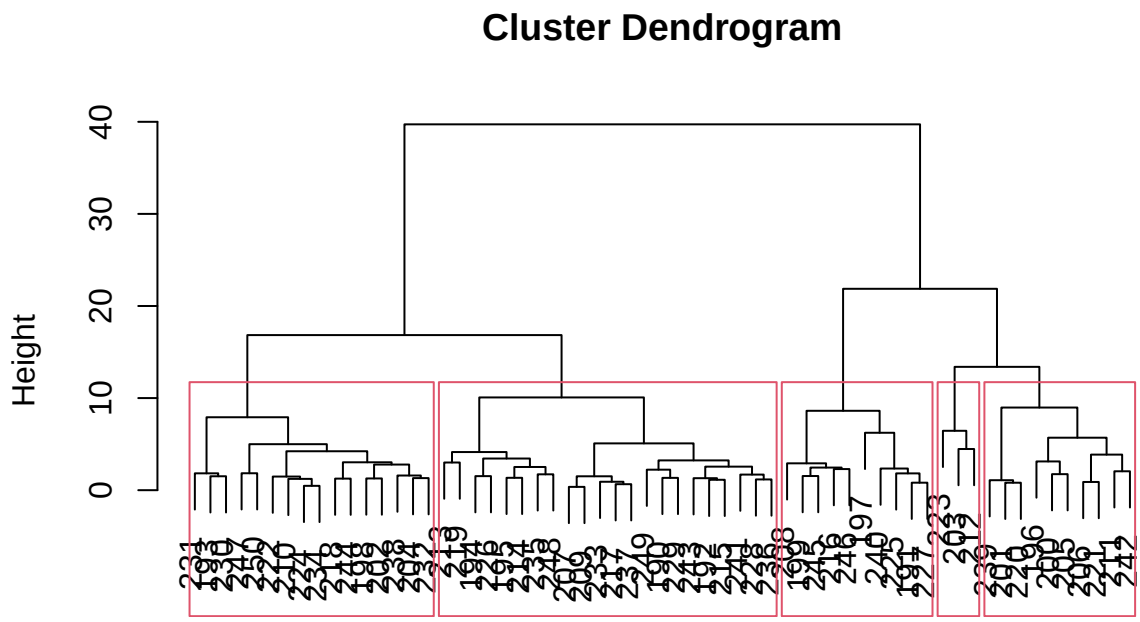
Hierarchical Clustering

```
library(cluster)
library(factoextra)
```

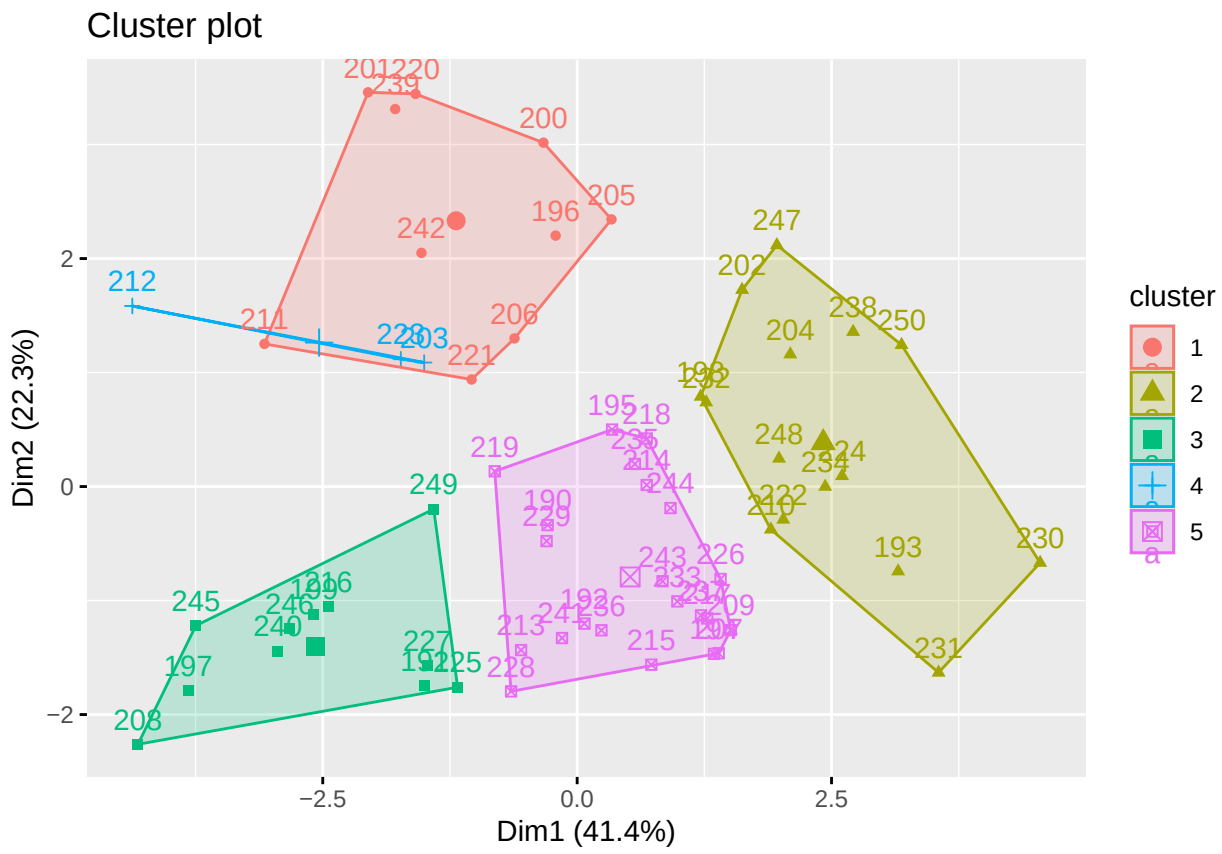
```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
data_scl<- scale(chorSub)
set.seed(123)
opt_clusters <- 5
km <- kmeans(data_scl, centers = opt_clusters, nstart = 25)
d <- dist(data_scl)
hc <- hclust(d, method = "ward.D")
```

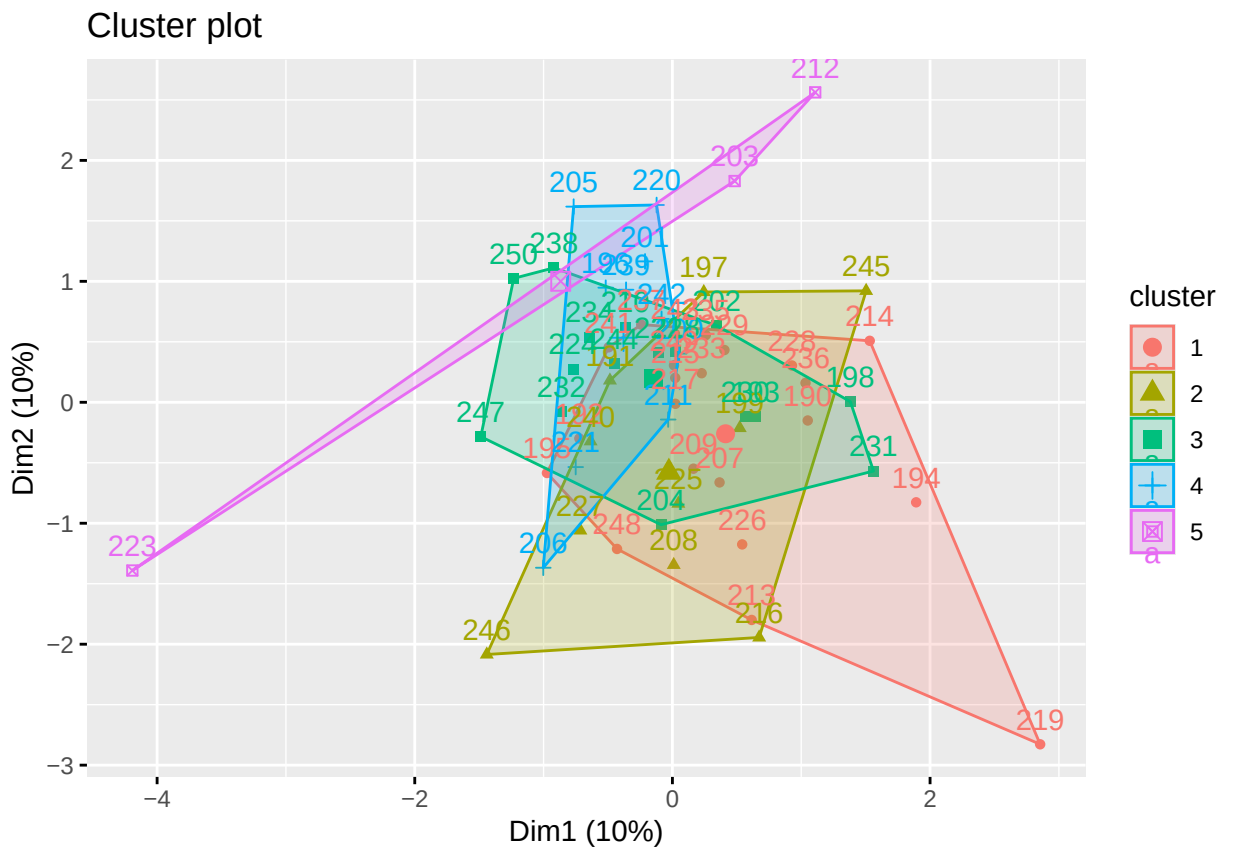
```
plot(hc)
rect.hclust(hc, k = opt_clusters)
```




```
grps <- cutree(hc, k = opt_clusters)
data_clstd <- data.frame(chorSub, KMeans_Cluster = km$cluster, Hierarchical_Cluster = grps)
k_plot<-fviz_cluster(list(data = data_scl, cluster = km$cluster))
print(k_plot)
```




```
pca_res <- prcomp(data_scl)
fviz_cluster(list(data = pca_res$x, cluster = grps))
```

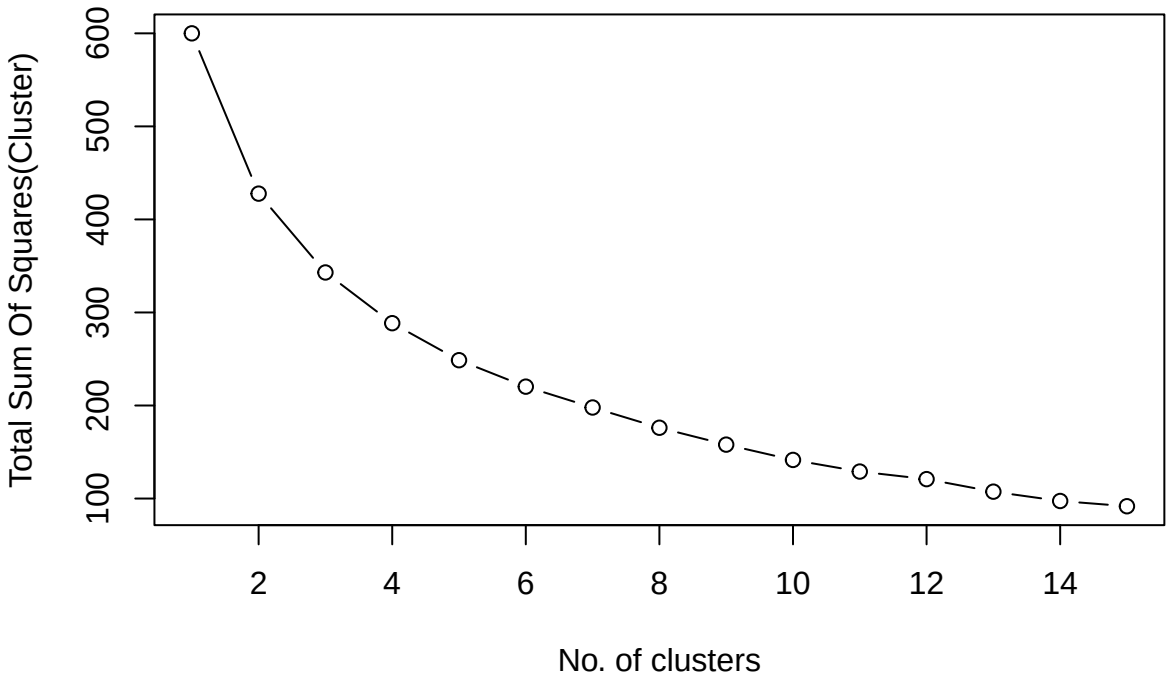


What is a good choice of "k" for each of these methods? Justify your selection.

For K-means clustering k can be choosen by analysing the data using elbow method, silhouette method and gap statistic method. here we will use Elbow method and Silhouette analysis.

Elbow method

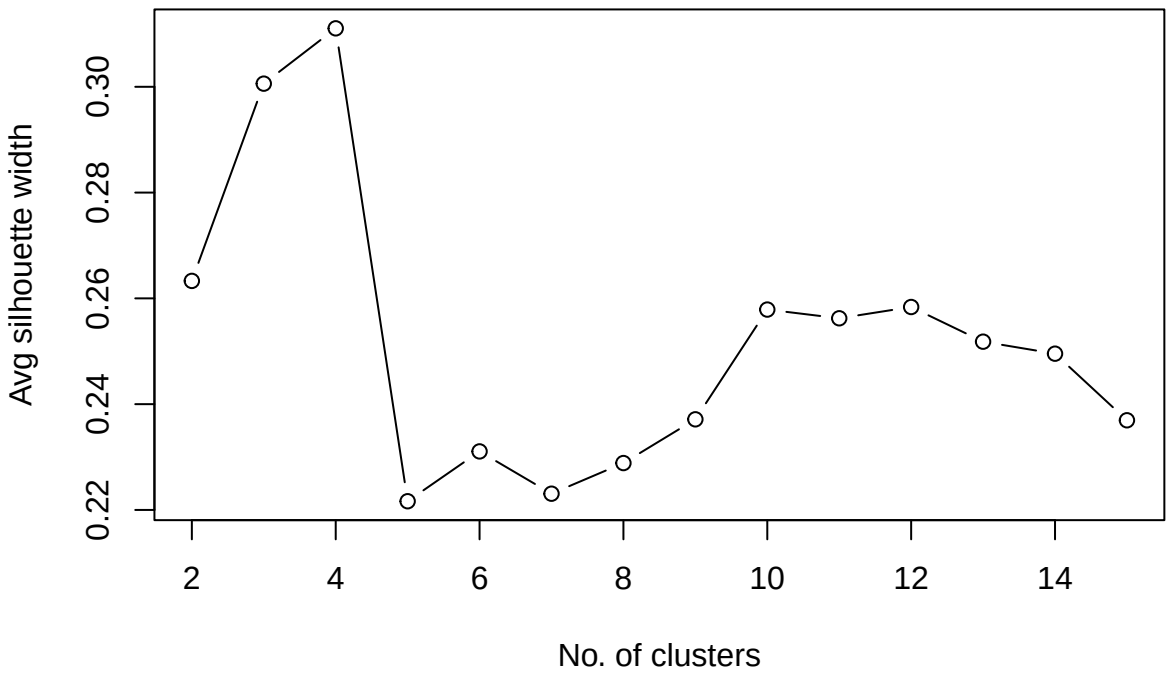
```
wss <- sapply(1:15, function(k){sum(kmeans(data_scl, k, nstart = 10)$withinss)})
plot(1:15, wss, type = "b", xlab = "No. of clusters", ylab = "Total Sum Of Squares(Cluster)")
```



This graph gradually smooths out as the number of clusters increases, but there's a slight bend at k=4. This suggests that adding more clusters after four will provide only marginal improvements to the model's fit.

Silhouette analysis

```
sil_scores <- sapply(2:15, function(k){
  km.res <- kmeans(data_scl, centers = k, nstart = 25)
  sil_score <- mean(silhouette(km.res$cluster, dist(data_scl))[, 3])
  return(sil_score)
})
plot(2:15, sil_scores, type = "b", xlab = "No. of clusters", ylab = "Avg silhouette width")
```

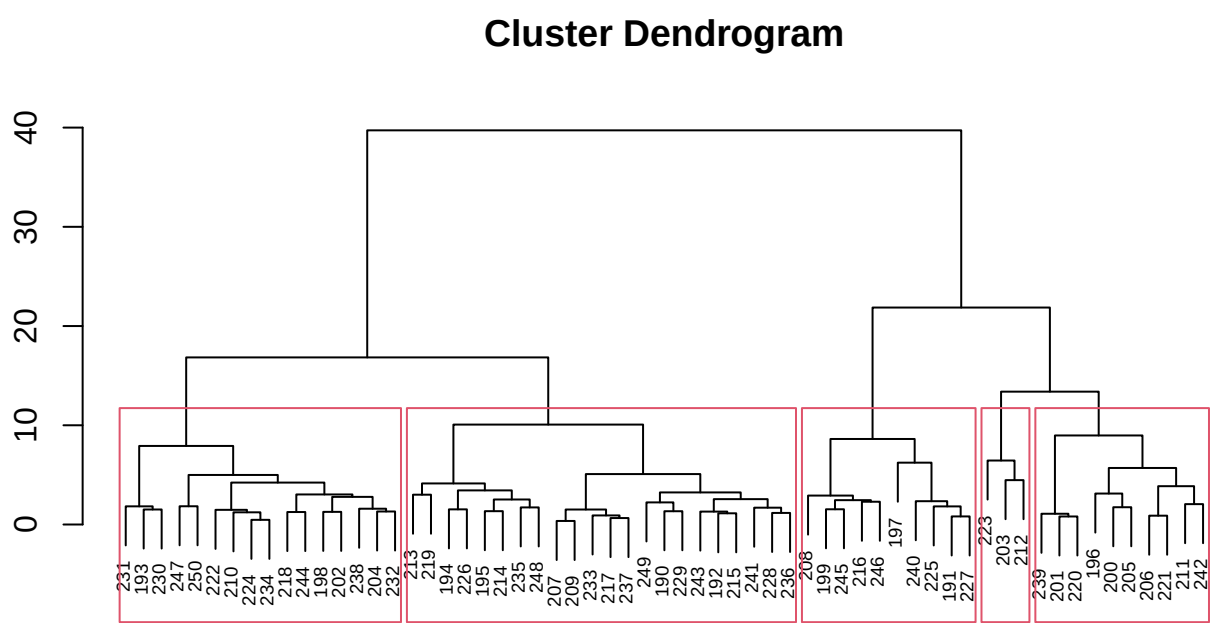


In Silhoutte analysis, the first peak is at k=4, indicating a strong level with two clusters that well-separate the data.

The elbow approach and the silhouette analysis both recommend that k=4 be used for k-means clustering. This choice is accurate. A massive peak can be seen in the silhouette plot at k=4, suggesting a robust structure with distinct clusters that are apart from one another. This is supported by the elbow plot, which exhibits a small bend at the same location and suggests that adding clusters beyond 4 will not significantly reduce variation. This methodological consistency confirms the correctness for k=4, which means a balanced and understandable clustering solution for the chorSub dataset that is both statistically and practically significant.For Hierarchical clustering we choose k by using dendrogram and silhouette method

dendrogram

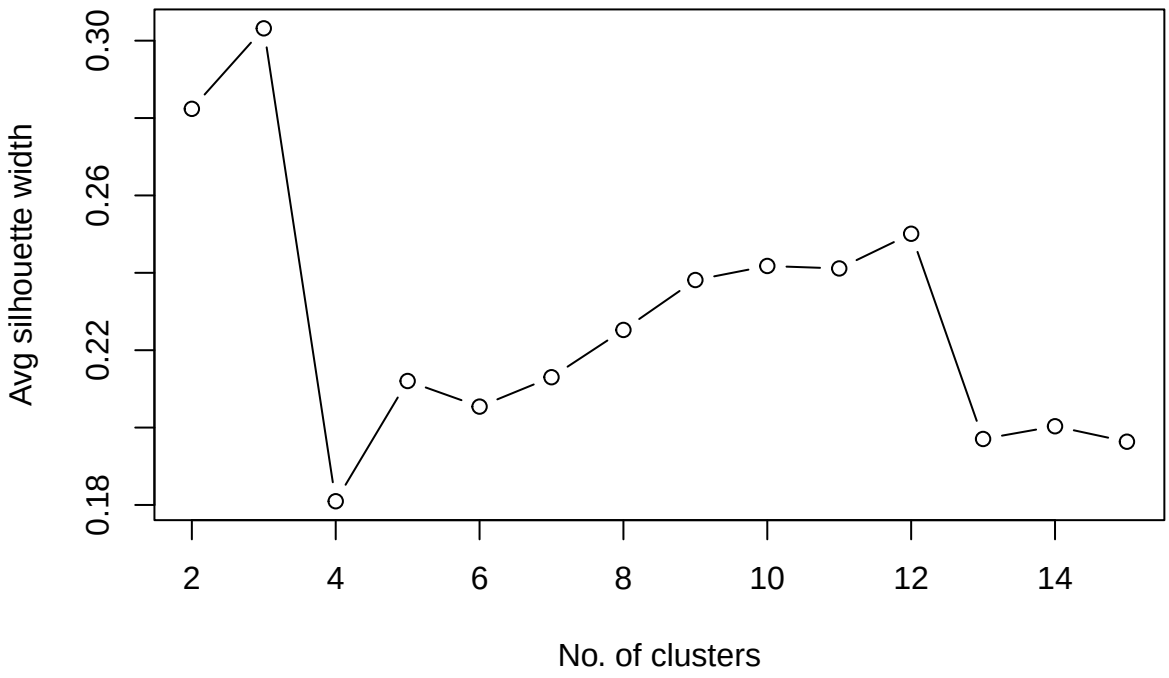
```
par(mar=c(5,2,4,0) + 0.1)
plot(hc, cex=0.6)
rect.hclust(hc, k = opt_clusters)
```



I searched for notable difference in the linkage distance, which frequently specify a natural cluster separation, when analyzing the dendrogram above. I analysed the dissimilarity between the data points that are clustered together based on the height of the merges. There are multiple levels at which clusters merge in this dendrogram, but when I pay close attention to the areas where there is a significant rise in merge height because these may be possible cuts for significant clusters.Five is the potential K value.

Silhouette analysis

```
sil_scores_hc <- sapply(2:15, function(k){
  clust <- cutree(hc, k)
  sil_score <- mean(silhouette(clust, dist(data_scl))[, 3])
  return(sil_score)
})
plot(2:15, sil_scores_hc, type = "b", xlab = "No. of clusters", ylab = "Avg silhouette width")
```



This plot presents a peak at k=3, which is another viable option but not as distinct as k=5

Although the silhouette analysis for hierarchical clustering indicates that k = 3 would be a good option, the dendrogram offers an alternative viewpoint, suggesting that k = 5 might offer more precise clustering. Due to significant jumps in the connection distance at this level, which indicate a more meaningful division of the data into natural groups , we selected k = 5 after analyzing the dendrogram.

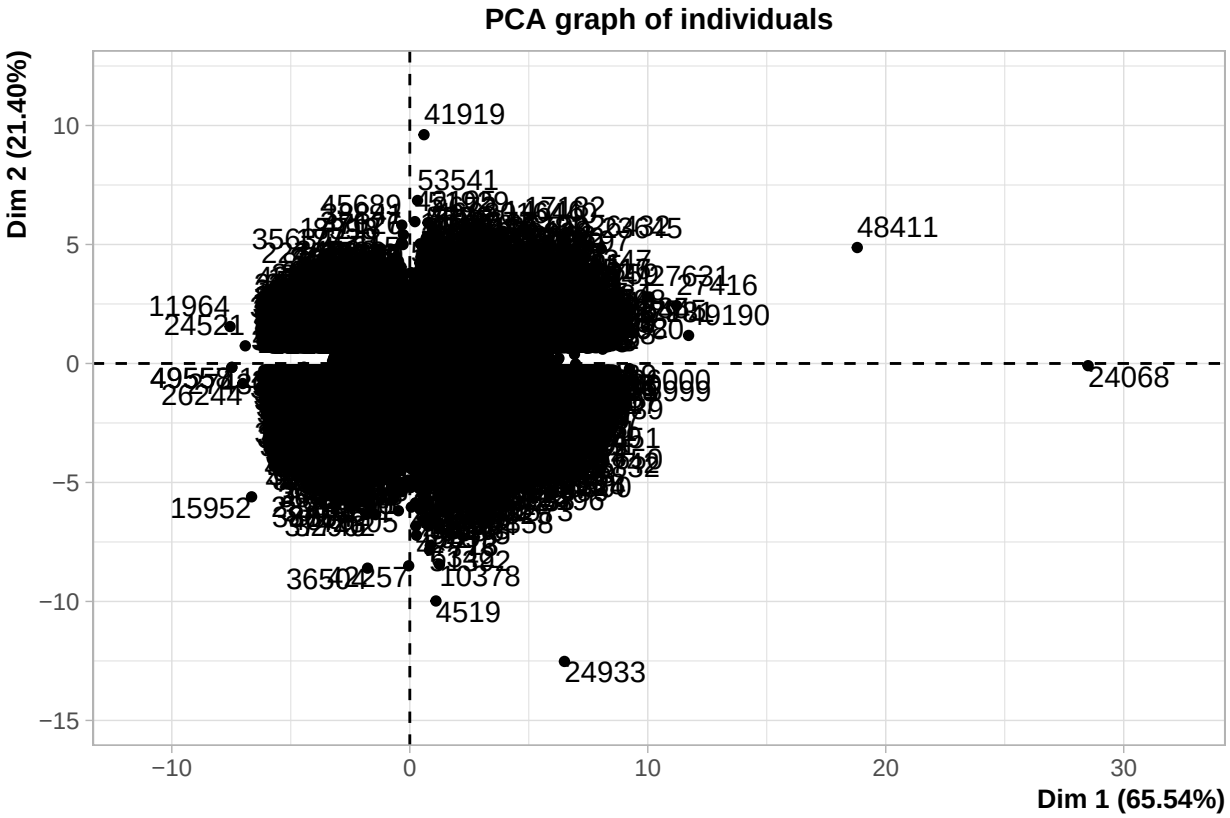
2) **Consider the "diamonds" data from ggplot2. Use principal components on the variables {caret, x, y, z, depth, table}, and answer the following questions.**

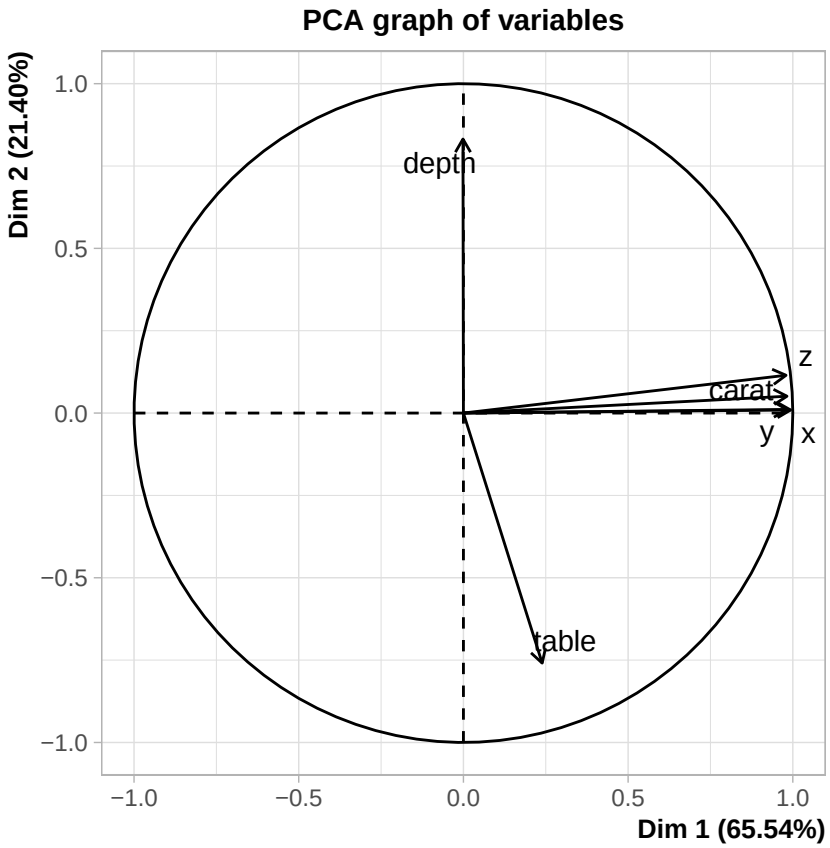
a) **How much of the total variance does the first principal component account for? How many components are needed to account for at least 90% of the total variance?**

```
library(ggplot2)
library(FactoMineR)
```

```
data(diamonds)
diamonds_data <- diamonds[, c("carat", "x", "y", "z", "depth", "table")]
```

```
pca_result <- PCA(diamonds_data, scale.unit = TRUE, ncp = 6)
```


PCA graph of individuals
Dim 2 (21.40%)
Dim 1 (65.54%)
41919
53541
48411
24068
11964
15952
4519
24933
10378
27416
10
5
0
−5
−10
−15
−10
0
10
20
30




```
exp_var <- pca_result$eig
print(exp_var[,2])
```

```
##     comp 1     comp 2     comp 3     comp 4     comp 5     comp 6
## 65.5356438 21.4008672 11.3899728  0.7900005  0.6633357  0.2201800
```

```
cum_var <- cumsum(exp_var[,2])
print(cum_var)
```

```
##    comp 1    comp 2    comp 3    comp 4    comp 5    comp 6
##  65.53564  86.93651  98.32648  99.11648  99.77982 100.00000
```

The first principal component accounts for about 65.54% of the total variance. To achieve at least 90% of the total variance explained, the first three components are necessary, cumulatively which is approximately 98.33%. This highlights the importance of these components in representing the dataset's variability.

b) **Judging by the loadings, what do the first two principal components measure?**

```
a <- pca_result$var$coord
print(a[, 1:2])
```

```
##              Dim.1        Dim.2
## carat  0.982294529  0.051139145
```

```
## x      0.993285835  0.009296089
## y      0.981997646  0.010943344
## z      0.979349447  0.114769965
## depth -0.001352864  0.831832602
## table  0.239108320 -0.759021094
```

The first primary component, which has large loadings on carat, x, y, and z, indicates the overall size of the diamonds. The second principal component evaluates features of cut quality and proportions orthogonal to size. It highlights variations in the form and cut of the diamonds by differing loadings on depth and table.

c) **What is the correlation between the first principal component and price?**

```
comp1_scores <- pca_result$ind$coord[,1]
corr_price <- cor(comp1_scores, diamonds$price)
print(corr_price)
```

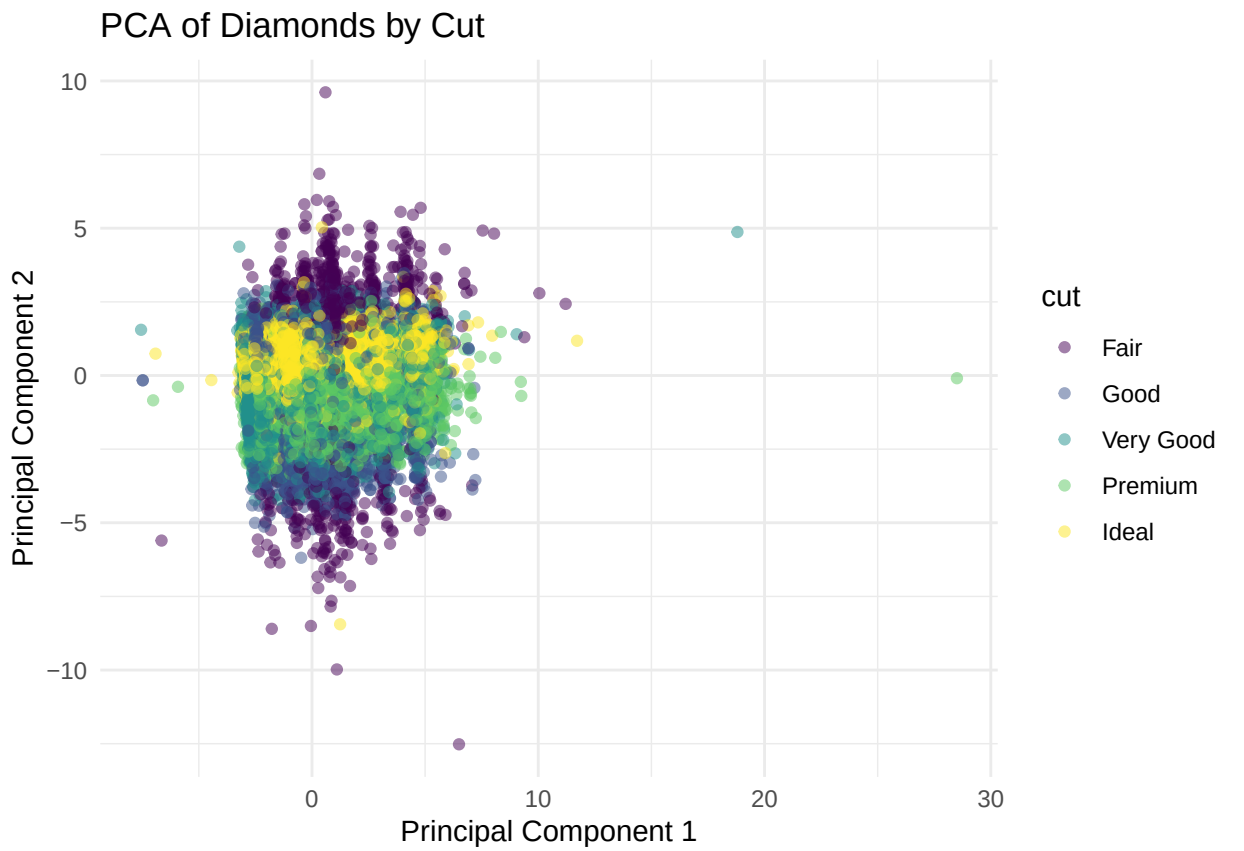
```
## [1] 0.8920056
```

The correlation between the first principal component and the price of the diamonds is 0.8920056, indicating a very strong positive relationship. This indicate that the principal component, which primarily measures the overall size of the diamonds, is a significant predictor of their price. The closer this value is to 1, the stronger the linear relationship. The size-related measures increase, so does the price, in a strong linear way.

d) **Can the first two principal components be used to distinguish between diamonds with different cuts?**

```
diamonds$PC1 <- pca_result$ind$coord[,1]
diamonds$PC2 <- pca_result$ind$coord[,2]
```

```
library(ggplot2)
ggplot(diamonds, aes(x = PC1, y = PC2, color = cut)) +
  geom_point(alpha = 0.5) +theme_minimal() +
  labs(title = "PCA of Diamonds by Cut", x = "Principal Component 1", y = "Principal Component 2")
```



The first two primary component's scatter plot, colored by diamond cut, shows how the various cut qualities overlap. There is some clustering, with 'Ideal' cuts appearing to be more central and denser, but no group is totally isolated from the others. This suggests that although there can be patterns linking specific cuts to areas inside the PCA space, the initial two principal components do not effectively distinguish diamonds based just on their cut quality.

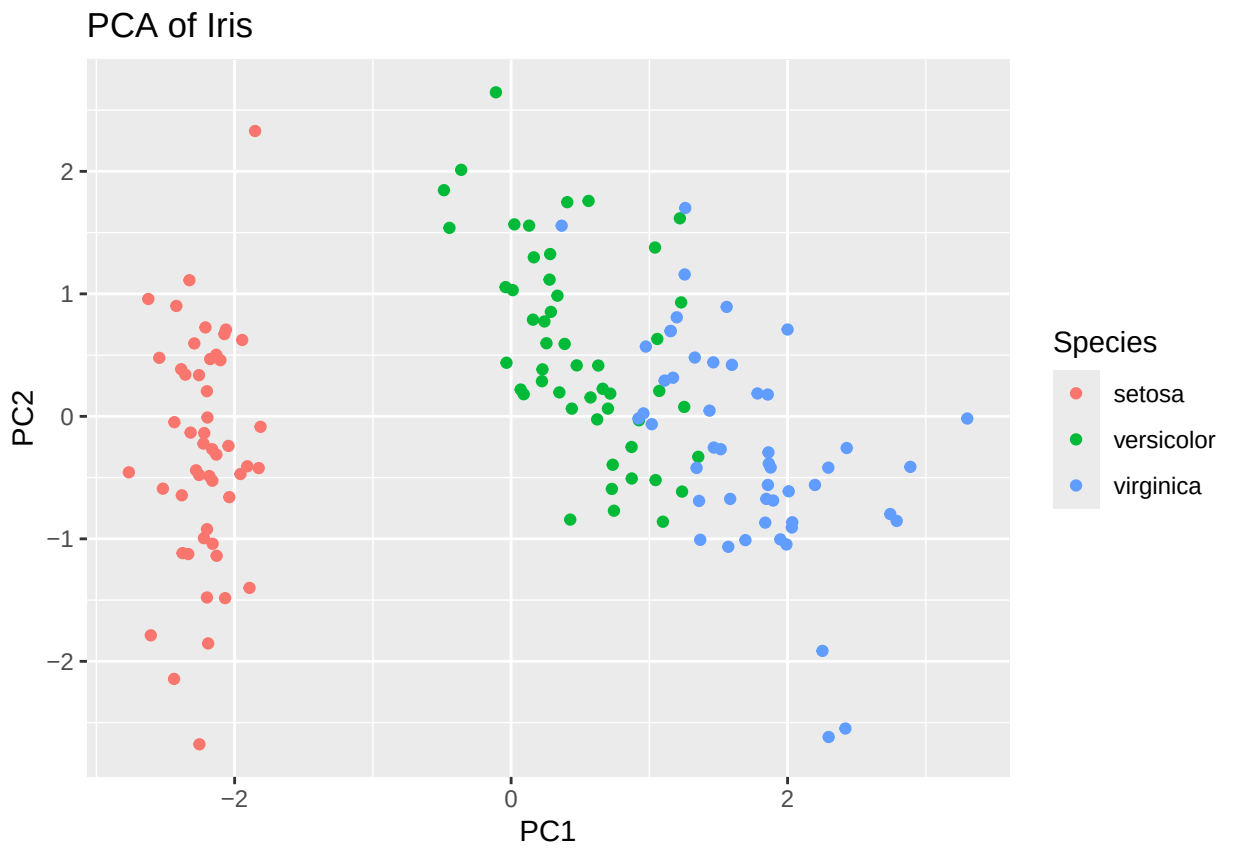
3) **Consider the Iris data data(iris)**

a) **Create a plot using the first two principal components, and color the iris species by class**

```
data(iris)
library(ggplot2)
```

```
iris.pca <- prcomp(iris[,1:4], center = TRUE, scale. = TRUE)
iris_pca_data <- data.frame(iris.pca$x, Species = iris$Species)
```

```
ggplot(iris_pca_data, aes(x = PC1, y = PC2, color = Species)) +
  geom_point() + labs(title = "PCA of Iris")
```

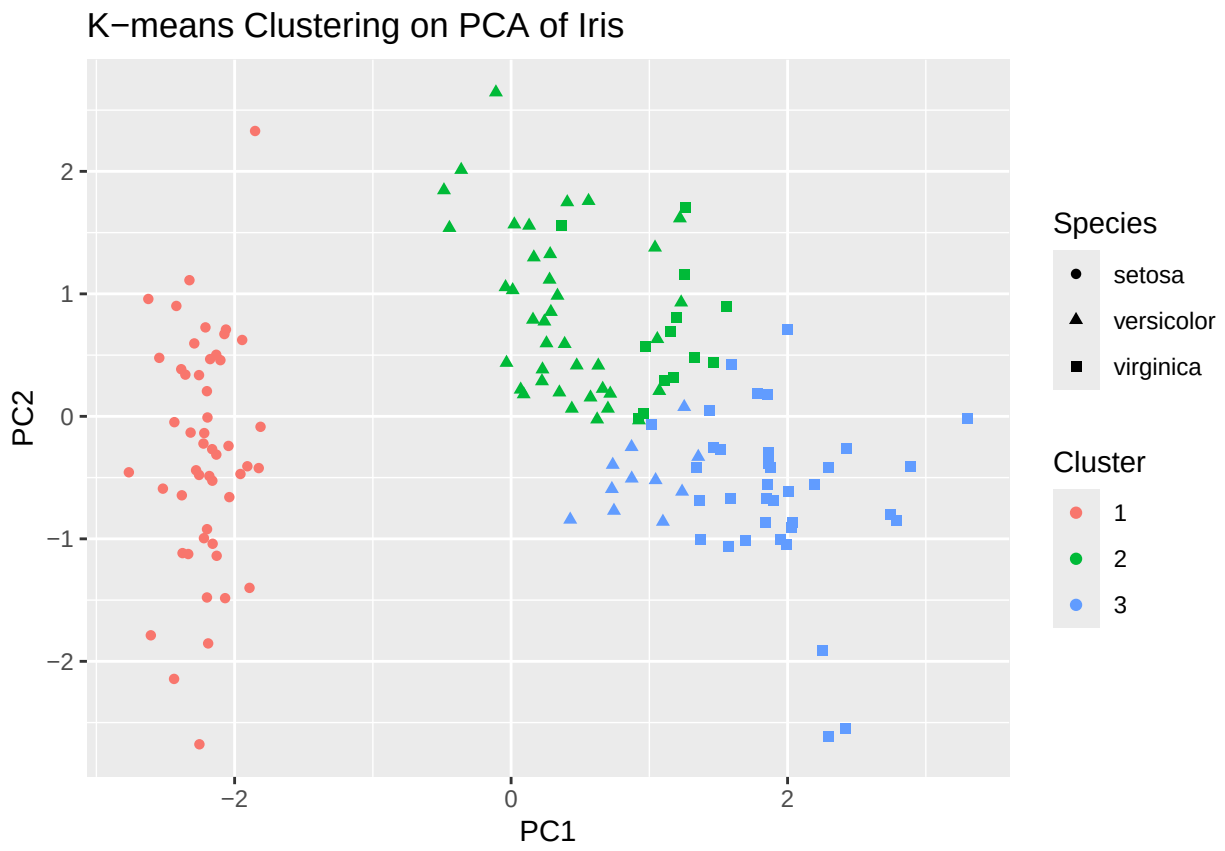


b) **Perform k-means clustering on the first two principal components of the iris data. Plot the clusters different colors, and the specify different symbols to depict the species labels.**

```
set.seed(123)
kmeans_result <- kmeans(iris.pca$x[,1:2], centers = 3)
```

```
iris_pca_data$Cluster <- as.factor(kmeans_result$cluster)
```

```
ggplot(iris_pca_data, aes(x = PC1, y = PC2, color = Cluster, shape = Species)) +geom_point() +labs(titl
```



c) **Use rand index and adjusted rand index to assess how well the cluster assignments capture the species labels.**

```
library(mclust)
```

```
## Package 'mclust' version 6.1
## Type 'citation("mclust")' for citing this R package in publications.
```

```
library(cluster)
library(clValid)
library(flexclust)
```

```
## Loading required package: grid
```

```
## Loading required package: lattice
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'modeltools'
```

```
## The following object is masked from 'package:clValid':
##
##     clusters
```

```
library(fpc)
library(fossil)
```

```
## Loading required package: sp

## Loading required package: maps

##
## Attaching package: 'maps'

## The following object is masked from 'package:mclust':
##
##     map

## The following object is masked from 'package:cluster':
##
##     votes.repub

## Loading required package: shapefiles

## Loading required package: foreign

##
## Attaching package: 'shapefiles'

## The following objects are masked from 'package:foreign':
##
##     read.dbf, write.dbf
```

```
library(bootcluster)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

##
## Attaching package: 'bootcluster'

## The following object is masked from 'package:clValid':
##
##     stability
```

```
# install.packages("clValid")
library(clValid)

species_num <- as.numeric(iris$Species)
randind <- rand.index(species_num, kmeans_result$cluster)
print(paste("Rand Index:",randind))
```

```
## [1] "Rand Index: 0.832214765100671"
```

```
adjind <- adj.rand.index(species_num, kmeans_result$cluster)
print(paste("Adjusted Rand Index:",adjind))
```

```
## [1] "Adjusted Rand Index: 0.620135180887038"
```

The Rand Index of 0.8322 indicates a high similarity between k-means clusters and the true iris species labels, while the Adjusted Rand Index of 0.6201, suggests a strong agreement. These values demonstrate that the clustering captures the natural groupings of the Iris dataset.

d) **Use the gap statistic and silhouette plots to determine the number of clusters**

```
library(cluster)
library(factoextra)
```

```
set.seed(123)
gap_stat <- clusGap(iris.pca$x[,1:2], FUN = kmeans, nstart = 25, K.max = 10)
```

```
opt_clusters <- maxSE(gap_stat$Tab[, "gap"], gap_stat$Tab[, "SE.sim"], method = "Tibs2001SEmax")
print(opt_clusters)
```

```
## [1] 3
```

The output indicates that the optimal number of clusters for your dataset, based on the gap statistic and its standard error is 3. This result aligns well with the structure of the Iris dataset, which contains three distinct species.

e) **Reflect on the results, especially c-d. What does this tell us about the clustering?**

The results of the Rand Index and Adjusted Rand Index show that k-means clustering and the genuine Iris species labels match well. the Adjusted Rand Index accounts for chance to give a more accurate assessment. The best number of clusters, as determined by the gap statistics is three,which matches with the actual number of species. This indicates that PCA and k-means were accurate in revealing the underlying structure of the Iris dataset. These results validate the strategy for exploratory analysis and pattern recognition by highlighting the ability of unsupervised learning approaches to identify occurring groupings within data.

Consider the wine quality data (https://archive.ics.uci.edu/dataset/186/wine+quality)

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## v readr     2.1.5

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::map()    masks maps::map(), mclust::map()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
red_wine <- read.csv("C:/msinub/sdm/winequality-red.csv", sep = ";")
white_wine <- read.csv("C:/msinub/sdm/winequality-white.csv", sep = ";")

red_wine$wine_color <- 'red'
white_wine$wine_color <- 'white'

wine_data <- bind_rows(red_wine, white_wine)

str(wine_data)
```

```
## 'data.frame':    6497 obs. of  13 variables:
##  $ fixed.acidity       : num  7.4 7.8 7.8 11.2 7.4 7.4 7.9 7.3 7.8 7.5 ...
##  $ volatile.acidity    : num  0.7 0.88 0.76 0.28 0.7 0.66 0.6 0.65 0.58 0.5 ...
##  $ citric.acid         : num  0 0 0.04 0.56 0 0 0.06 0 0.02 0.36 ...
##  $ residual.sugar      : num  1.9 2.6 2.3 1.9 1.9 1.8 1.6 1.2 2 6.1 ...
##  $ chlorides           : num  0.076 0.098 0.092 0.075 0.076 0.075 0.069 0.065 0.073 0.071 ...
##  $ free.sulfur.dioxide : num  11 25 15 17 11 13 15 15 9 17 ...
##  $ total.sulfur.dioxide: num  34 67 54 60 34 40 59 21 18 102 ...
##  $ density             : num  0.998 0.997 0.997 0.998 0.998 ...
##  $ pH                  : num  3.51 3.2 3.26 3.16 3.51 3.51 3.3 3.39 3.36 3.35 ...
##  $ sulphates           : num  0.56 0.68 0.65 0.58 0.56 0.56 0.46 0.47 0.57 0.8 ...
##  $ alcohol             : num  9.4 9.8 9.8 9.8 9.4 9.4 9.4 10 9.5 10.5 ...
##  $ quality             : int  5 5 5 6 5 5 5 7 7 5 ...
##  $ wine_color          : chr  "red" "red" "red" "red" ...
```

a) **Perform exploratory data analysis on the data. Summarize the data quality and characterisHcs. Discuss any apparent outliers and associaHons.**

EDA - Exploratory Data Analysis

```r
summary(wine_data)
```

```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   : 3.800   Min.   :0.0800   Min.   :0.0000   Min.   : 0.600
##  1st Qu.: 6.400   1st Qu.:0.2300   1st Qu.:0.2500   1st Qu.: 1.800
##  Median : 7.000   Median :0.2900   Median :0.3100   Median : 3.000
##  Mean   : 7.215   Mean   :0.3397   Mean   :0.3186   Mean   : 5.443
```

```
##  3rd Qu.: 7.700   3rd Qu.:0.4000   3rd Qu.:0.3900   3rd Qu.: 8.100
##  Max.   :15.900   Max.   :1.5800   Max.   :1.6600   Max.   :65.800
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.00900   Min.   :  1.00      Min.   :  6.0        Min.   :0.9871
##  1st Qu.:0.03800   1st Qu.: 17.00      1st Qu.: 77.0        1st Qu.:0.9923
##  Median :0.04700   Median : 29.00      Median :118.0        Median :0.9949
##  Mean   :0.05603   Mean   : 30.53      Mean   :115.7        Mean   :0.9947
##  3rd Qu.:0.06500   3rd Qu.: 41.00      3rd Qu.:156.0        3rd Qu.:0.9970
##  Max.   :0.61100   Max.   :289.00      Max.   :440.0        Max.   :1.0390
##        pH          sulphates         alcohol         quality
##  Min.   :2.720   Min.   :0.2200   Min.   : 8.00   Min.   :3.000
##  1st Qu.:3.110   1st Qu.:0.4300   1st Qu.: 9.50   1st Qu.:5.000
##  Median :3.210   Median :0.5100   Median :10.30   Median :6.000
##  Mean   :3.219   Mean   :0.5313   Mean   :10.49   Mean   :5.818
##  3rd Qu.:3.320   3rd Qu.:0.6000   3rd Qu.:11.30   3rd Qu.:6.000
##  Max.   :4.010   Max.   :2.0000   Max.   :14.90   Max.   :9.000
##   wine_color
##  Length:6497
##  Class :character
##  Mode  :character
##
##
##
```

```
sum(is.na(wine_data))
```

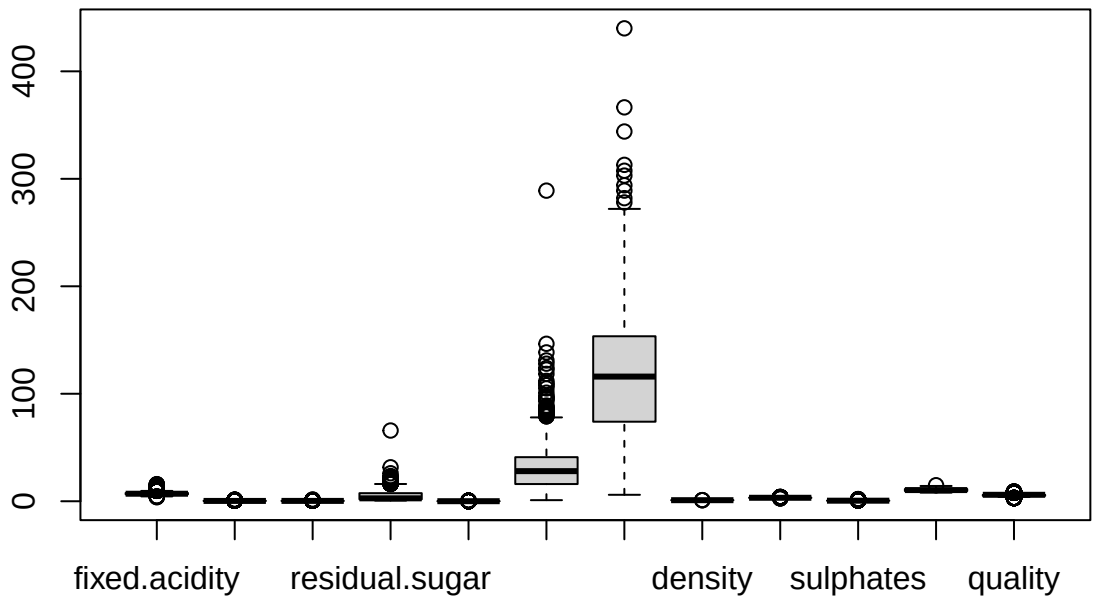
```
## [1] 0
```

```
sum(duplicated(wine_data))
```

```
## [1] 1177
```

```
wine_data <- wine_data[!duplicated(wine_data), ]
```

```
boxplot(wine_data[,sapply(wine_data, is.numeric)])
```

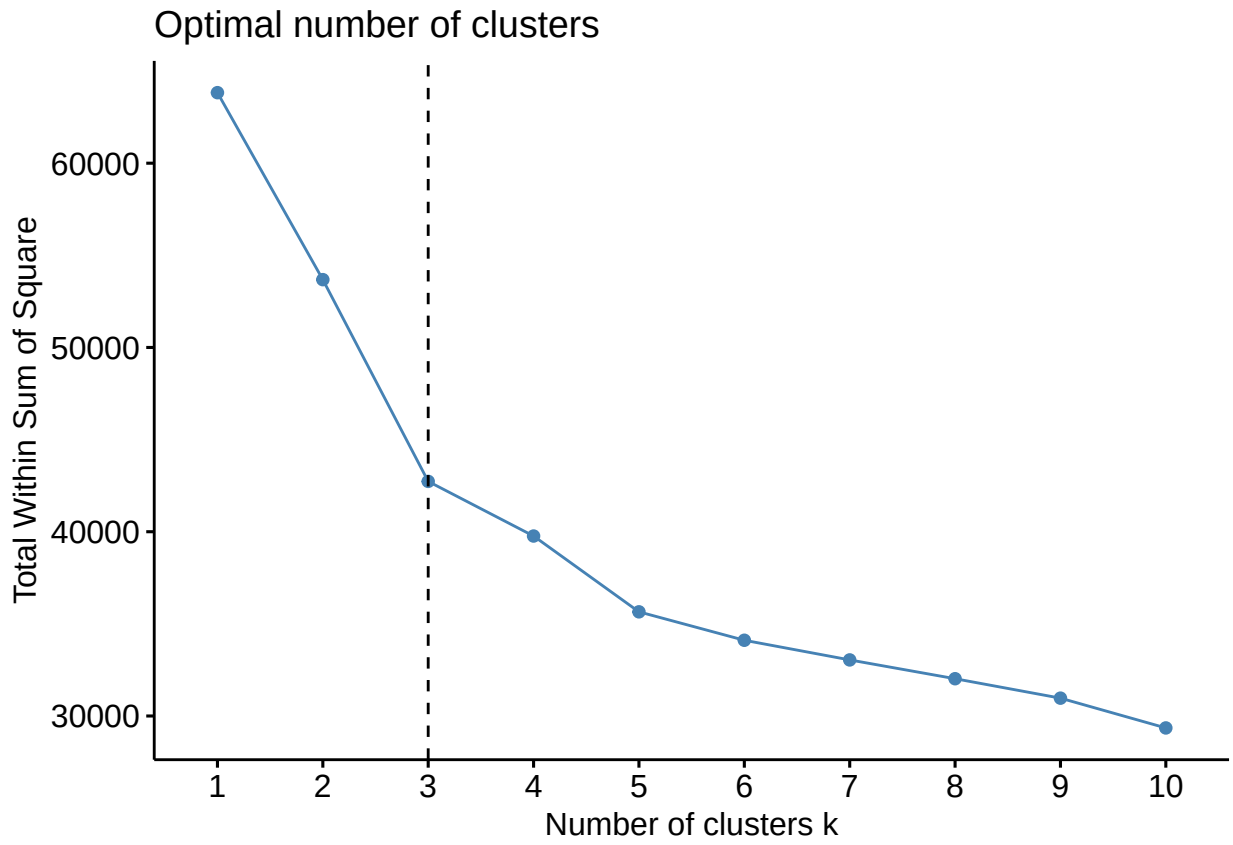


The boxplot for the wine quality dataset reveals numerous outliers, particularly in total.sulfur.dioxide, which could affect the clustering outcomes. Distributions vary across variables, indicating the need for data normalization.The dataset contains duplicates that should be addressed to ensure the integrity of Dataset.

b) **Perform k-means using Principal Components of the wine data. Justify your choice of "k". Visualize the result using a biplot and color the points (samples) according to "wine color".**

```
library(ggplot2)
library(factoextra)

wine_data_num <- wine_data[sapply(wine_data, is.numeric)]
wine_data_scl <- scale(wine_data_num)
```

```
pca_res <- prcomp(wine_data_scl, center = TRUE, scale. = TRUE)


k_plot<- fviz_nbclust(pca_res$x, kmeans, method = "wss") + geom_vline(xintercept = 3, linetype = 2)
print(k_plot)
```

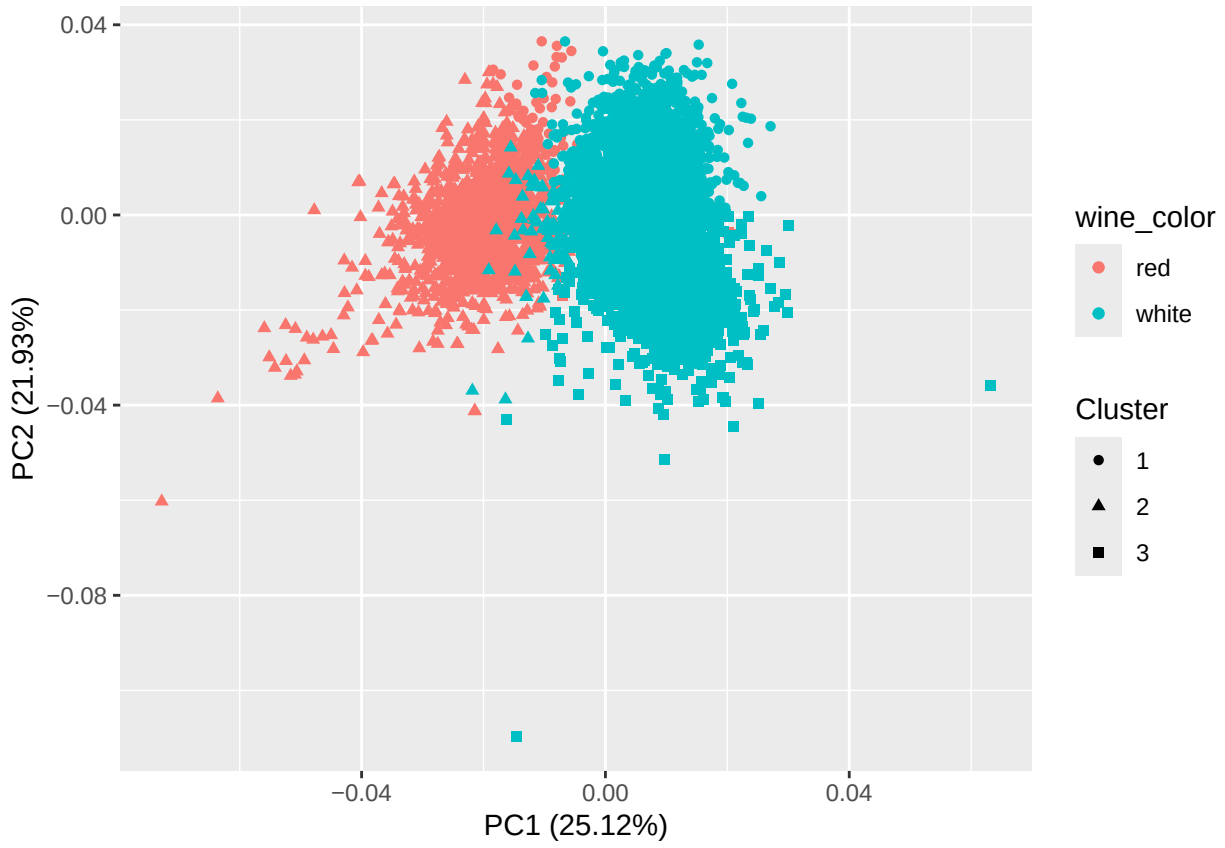



```
set.seed(123)
k <- 3
km_res <- kmeans(pca_res$x[, 1:2], centers = k)
wine_data$Cluster <- as.factor(km_res$cluster)
```

The elbow plot indicates an optimal k-value of 3 for clustering by the point where the within-cluster sum of squares begins to decline more slowly, marking the elbow. Choosing k = 3 will balance between minimizing within-cluster variance and avoiding overfitting with too many clusters. This number represents a meaningful separation in the wine dataset, capturing significant variance between the clusters while keeping the model simple.

performing k-means clustering with k set to 3 and create the biplot with clusters and the wine color:

```
library(ggfortify)
set.seed(123)
km_res <- kmeans(pca_res$x[, 1:2], centers = 3)
pca_cluster <- data.frame(pca_res$x, Cluster = factor(km_res$cluster))
pca_cluster$wine_color <- wine_data$wine_color
library(ggfortify)
autoplot(prcomp(wine_data_scl), data = pca_cluster, colour = 'wine_color', shape = 'Cluster')
```



About 47% of the variance is explained by the first two principal components, indicating that while they capture significant aspects of wine, they do not be considered for all of it. Wines are not strictly separated by color in k-means-formed clusters, suggesting that there are more complex elements to consider than just hue. This plot helps explain how wines are grouped according to their characteristics and may suggest more slight differences than color classification can convey. Clearer separations might result from additional research using more components or by using different clustering techniques.

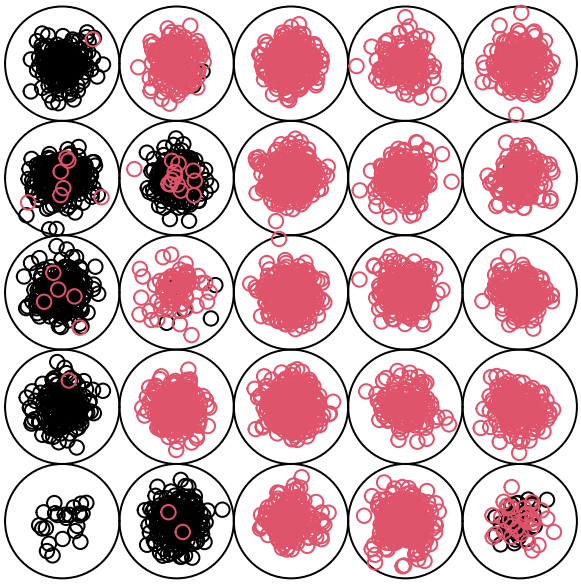
c) **Fit an SOM and color the samples according to wine color. Cluster the codebook vectors of the prototypes using hclust**.

```
if (!requireNamespace("kohonen", quietly = TRUE)) {
  install.packages("kohonen")
}
library(kohonen)
```

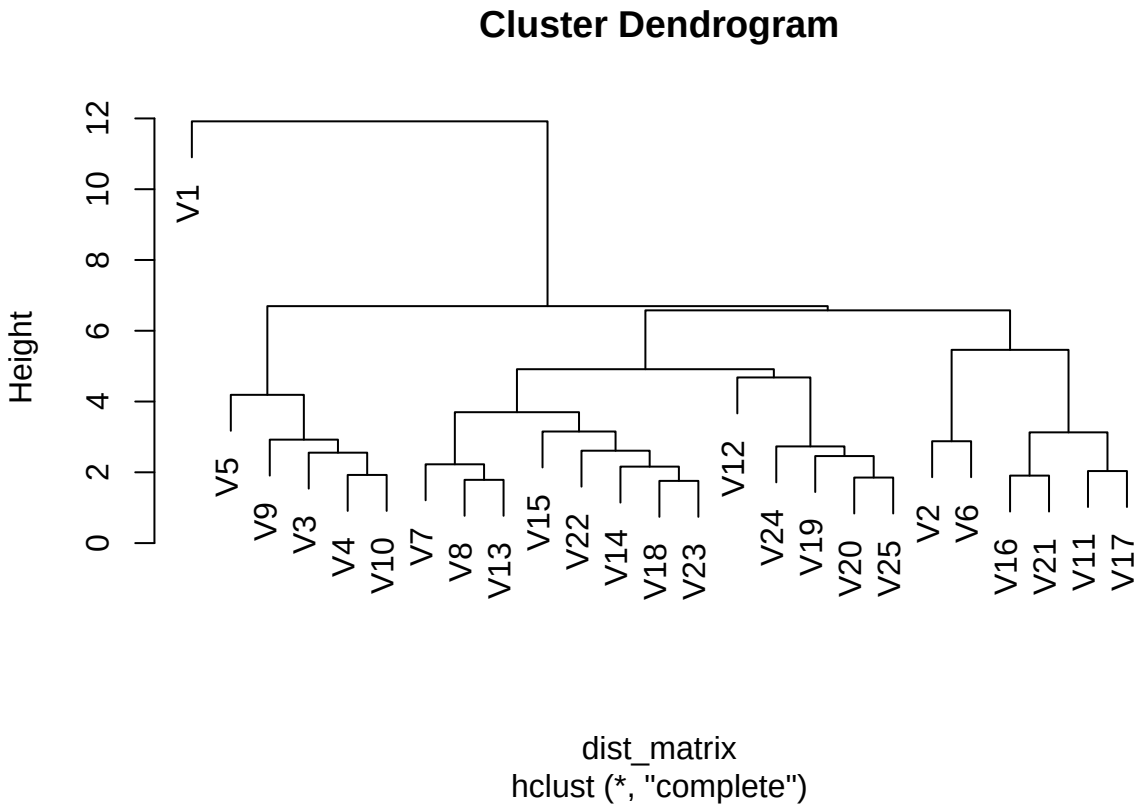
```
##
## Attaching package: 'kohonen'

## The following object is masked from 'package:purrr':
##
##     map

## The following object is masked from 'package:maps':
##
##     map
```

```
## The following object is masked from 'package:mclust':
##
##     map
```

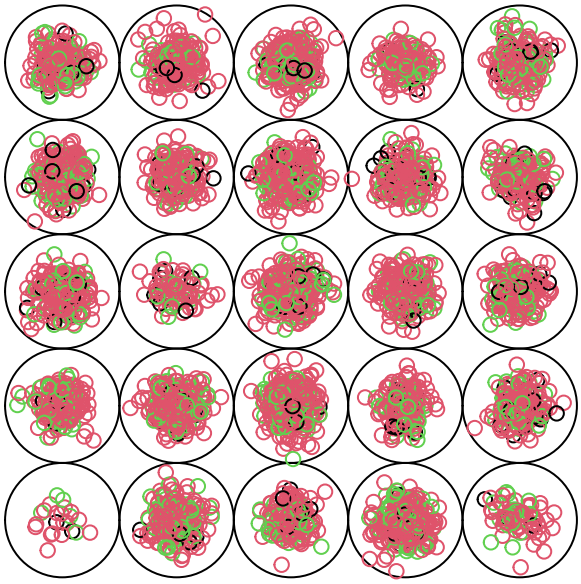
```
wine_data_num <- wine_data[, sapply(wine_data, is.numeric)]
wine_data_scl <- scale(wine_data_num)
som_x <- 5
som_y <- 5

som_grid <- somgrid(xdim = som_x, ydim = som_y, topo = "rectangular")
som_model <- som(wine_data_scl, grid = som_grid)

plot(som_model, type = "mapping", col = as.factor(wine_data$wine_color))
```




```
dist_matrix <- dist(som_model$codes[[1]])
hclust_res <- hclust(dist_matrix)

plot(hclust_res)
```

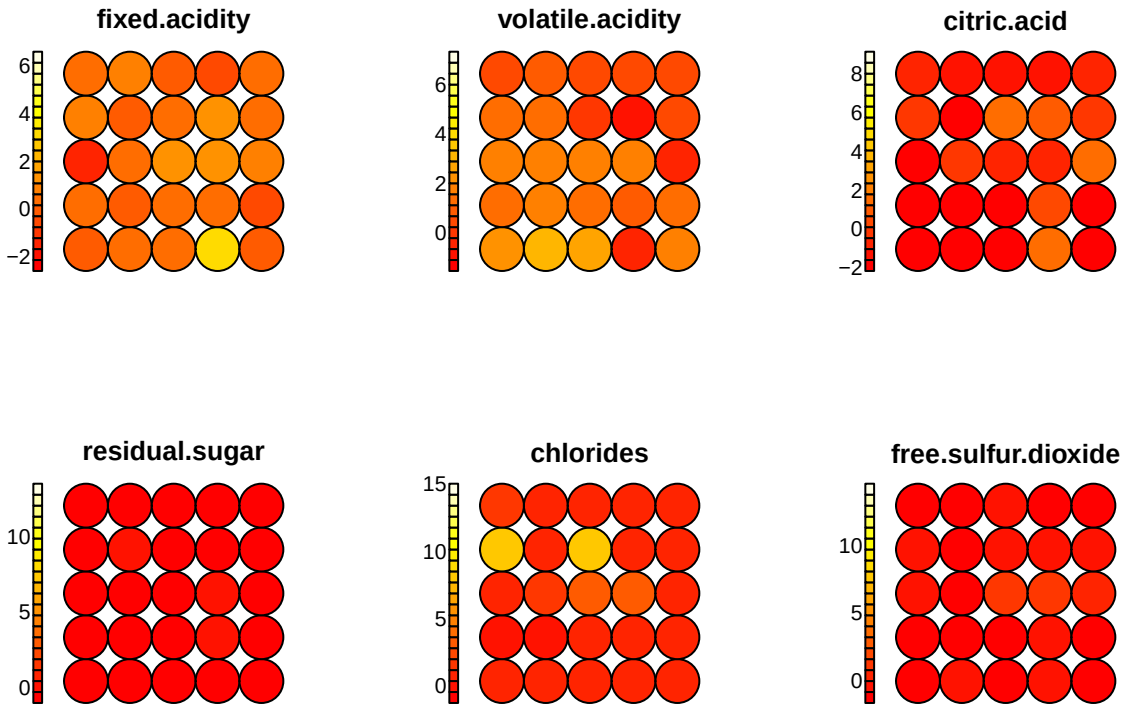


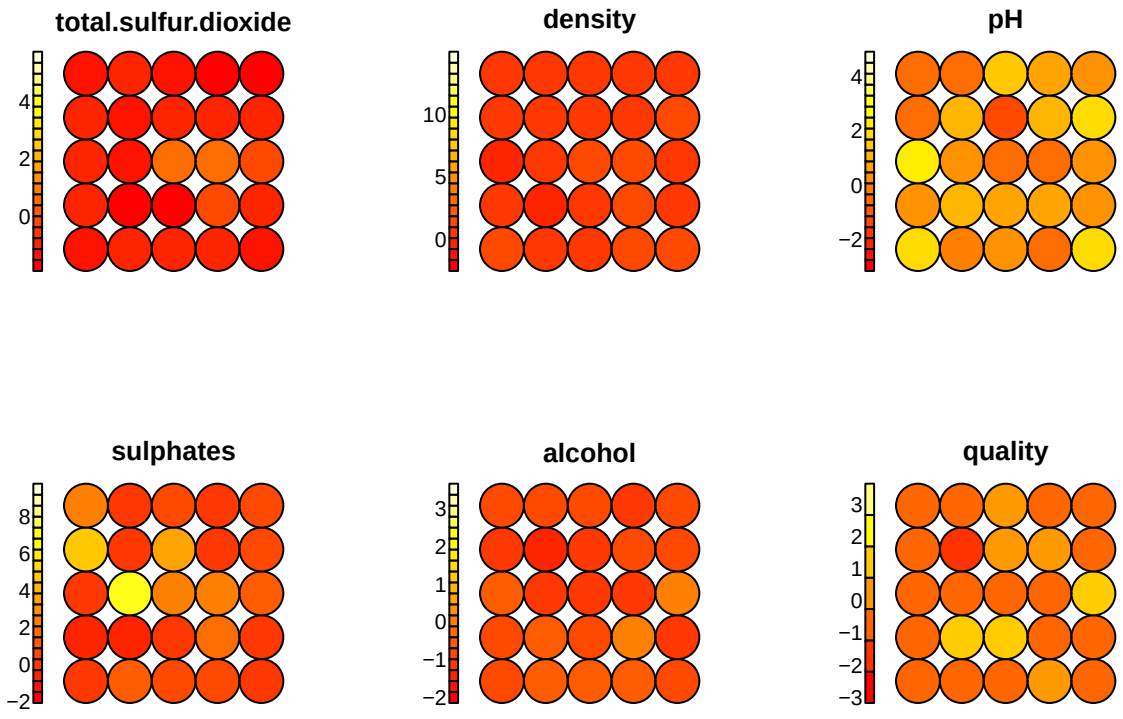

```
h_clusters <- 3
clusters <- cutree(hclust_res, h_clusters)

som_model$colors <- as.factor(clusters)
plot(som_model, type = "mapping", col = som_model$colors)
```



d) **Construct phase-plots (aka component planes) for some of the variables in the dataset.**

```
library(kohonen)

par(mfrow = c(2, 3))
for (i in 1:12) {

  comp_plane <- wine_data_scl[, i]

  var_name <- colnames(wine_data_scl)[i]

  plot(som_model, type = "property", property = comp_plane, main = var_name)
}
```


fixed.acidity
6
4
2
0
−2
volatile.acidity
6
4
2
0
citric.acid
8
6
4
2
0
−2
residual.sugar
10
5
0
chlorides
15
10
5
0
free.sulfur.dioxide
10
5
0



e) **Comment on the differences between b and c.**

The k-means clustering applied to the principal components of the wine data (b) indicated clusters that, while not strictly separating wines by color, did show some variation based on the most significant variances captured by PCA. As seen in the biplot, the k-means method offered a straightforward, linear segmentation of the data that is very helpful for locating large differences in the dataset.

The mapping of the wine data, however, was more complex and thorough in the SOM analysis (c). The SOM grid, which uses color coding based on wine color, revealed that it is more difficult to distinguish between red and white wines, pointing to a more intricate interaction between the many wine characteristics. Additional levels of structure were produced by the hierarchical clustering that was then applied to the SOM codebook vectors seen in the dendrogram. This highlighted a hierarchical organization within the data that is not revealed by k-means clustering.

Part (d)'s component planes provide additional information about the contributions of the individual variables throughout the grid, enhancing our comprehension of how each chemical measurement affects cluster formation.

The principal difference between the two methods is how they deal with data complexity. By distinct, non-overlapping clusters, the k-means method aims to reduce complexity and is perfect for spotting broad trends. SOMs have complexity, permitting smooth transitions between various wine properties and overlapping clusters, which may better capture the insights of the real world found in the data.

The analytical goals should guide the decision between SOM and k-means. K-means provides an easily interpreted high-level view that might serve as a useful foundation for additional investigation. SOMs, on the other hand, work well in exploratory analysis, where catching the complex patterns in the data is more crucial than achieving instant clarity. If the objective of the wine quality dataset is to establish

distinct market segments, then k-means might be a better fit. SOM would offer a deeper and more thorough viewpoint if the objective is to investigate the minute details of wine composition.