

# K-Means

*Seung Ah Ha, Jaymo Kim, Wonbin Song*

Algorithm: First randomly assign a group label from 1 to 3 to each observations. Then calculate the current group centers and re-assign each observation to the cluster whose center it is the closest to (using the Euclidean distance). And we iterate until the assignment of observations to groups stops changing.

We implemented the algorithm by first creating the function “initialft” that calculates the group centers and the Euclidean distances and returns new group label where each elements of the new group label is the group whose center it is closest to. And we created the function “kmeans\_ft” that iterates the function “initialft” to compare new group label to previous group label and returns the new group label when the new group label equals the previous group label.

```
# install.packages('rattle')
data(wine, package="rattle")

initialft <- function(mat.data, n.row, n.col, G.label, new.data)
{
  # calculates the group centres
  mat.mu <- matrix(0,3,(n.col+1))
  for(j in 2:(n.col+1)){
    for(i in 1:3){
      mat.mu[i,j] <- mean(new.data[G.label==i,j])
    }
  }
  mat.mu[1,1]<- 1
  mat.mu[2,1]<- 2
  mat.mu[3,1]<- 3

  # calculates the Euclidean distance
  distance <- matrix(0,n.row,3)
  new.label <- rep(0,n.row)
  for(i in 1:n.row){
    for(j in 2:(n.col+1)){
      distance[i,1] <- sqrt(sum((new.data[i,j]-mat.mu[1,j])^2))
      distance[i,2] <- sqrt(sum((new.data[i,j]-mat.mu[2,j])^2))
      distance[i,3] <- sqrt(sum((new.data[i,j]-mat.mu[3,j])^2))
    }
  }

  # re-assign
  for (i in 1:n.row){
    new.label[i] <- which(distance[i,]==min(distance[i,]))
  }

  return(new.label)
}

kmeans_ft <- function(data)
{
  mat.data <- data.matrix(data)
  n.row <- nrow(mat.data)
```

```

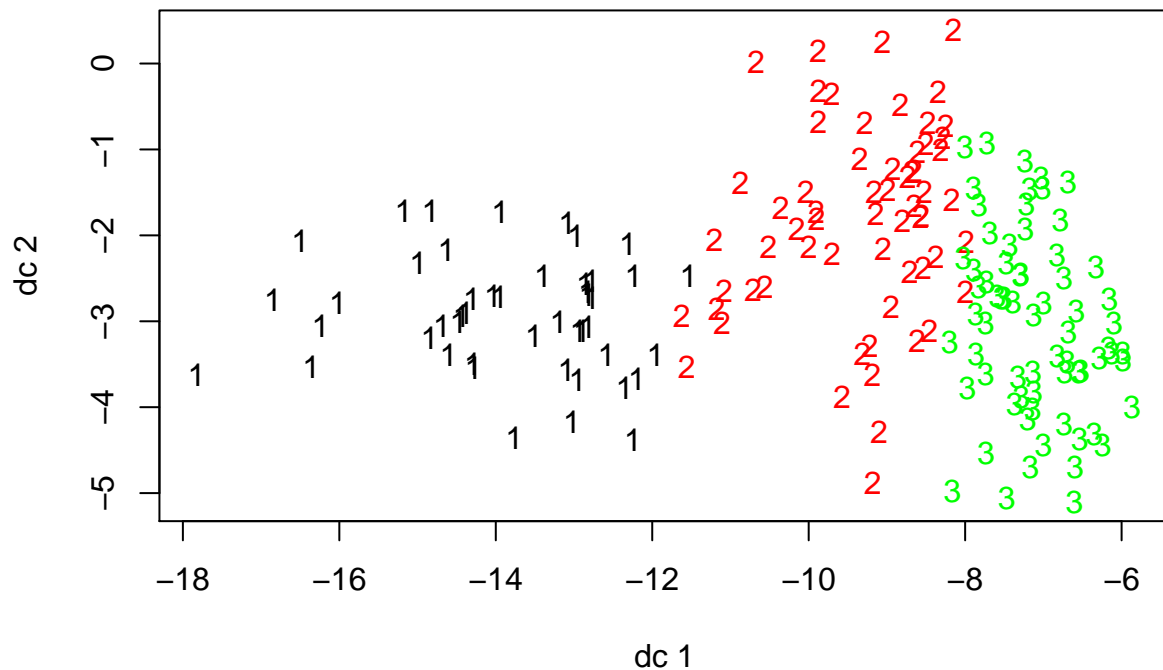
n.col <- ncol(mat.data)

# randomly assigning a group label
G.label <- rep(0,n.row)
for(i in 1:n.row){
  G.label[i] <- sample(1:3,1)
}
new.data <- cbind(G.label,mat.data)

new.l <- initialft(mat.data, n.row, n.col, G.label, new.data)
n.iter <- 1
while(any(new.l != G.label))
{
  G.label <- new.l
  new.l <- initialft(mat.data, n.row, n.col, G.label, new.data)
  n.iter <- n.iter + 1
}
return(list(number_of_iter = n.iter, new_group = new.l))
}

# install.packages("cluster")
# install.packages("fpc")
library(cluster)
library(fpc)
cluster <- kmeans_ft(wine[-1])
plotcluster(wine[-1], cluster$new_group)

```



```
cluster
```

```
## $number_of_iter
## [1] 6
##
## $new_group
## [1] 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 1 1 2 2 1 1 2 1 1 1 1 1 1
## [36] 2 2 1 1 2 2 1 1 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 3 2 3 2 3 3 2 3 3 2 2
## [71] 2 3 3 1 2 3 3 3 2 3 3 2 2 3 3 3 3 3 2 2 3 3 3 3 3 2 2 3 2 3 2 3 2 3 3 2
## [106] 3 3 3 3 2 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 2 2 2 2 3 3 3
## [141] 2 2 3 3 2 2 3 2 2 3 3 3 3 2 2 2 3 2 2 2 3 2 3 2 2 3 2 2 2 2 3 3 2 2 2
## [176] 2 2 3
```

```
wine[,1][which(cluster$new_group==1)]
```

```
## [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [36] 1 1 1 1 1 1 1 1 1 1 1 1 2
## Levels: 1 2 3
```

```
wine[,1][which(cluster$new_group==2)]
```

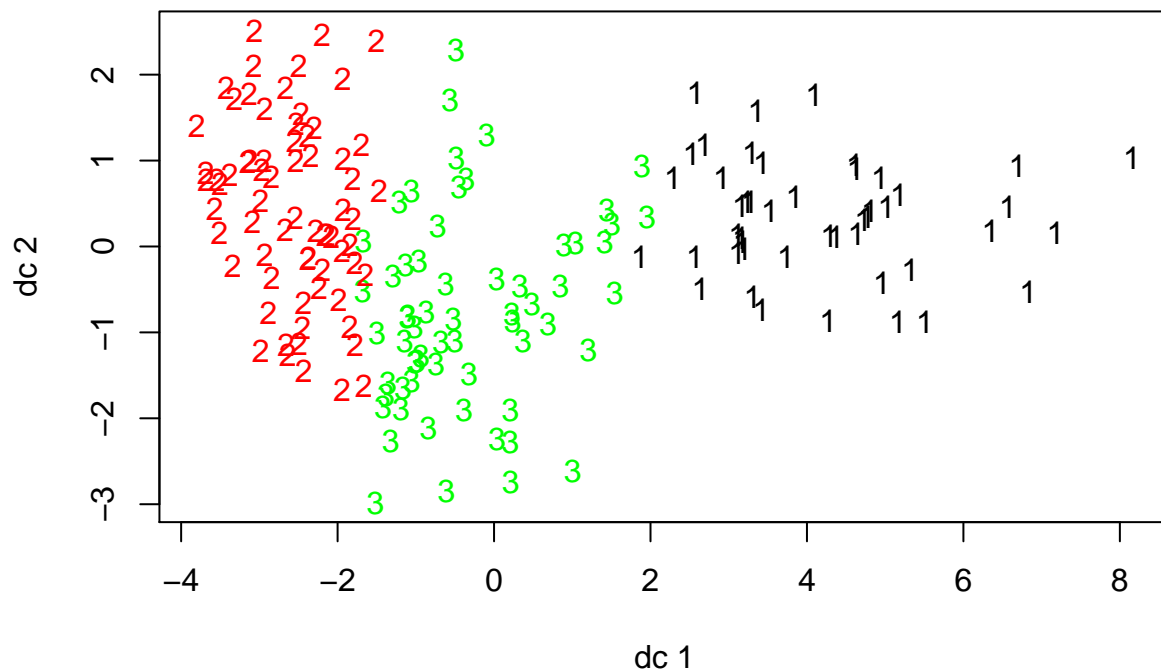
```
## [1] 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3
## [36] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## Levels: 1 2 3
```

```
wine[,1][which(cluster$new_group==3)]
```

```
## [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [36] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## Levels: 1 2 3
```

From the plot, We can see that the clusters seem well separated but failed to distinguish wine type 2 from wine type 3.

```
data.train <- scale(wine[-1])
cluster.scale <- kmeans_ft(data.train)
plotcluster(data.train, cluster.scale$new_group)
```



```
cluster.scale <- kmeans_ft(data.train)
wine[,1][which(cluster.scale$new_group==1)]
```

```
## [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3
## [36] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## Levels: 1 2 3
```

```
wine[,1][which(cluster.scale$new_group==2)]
```

```
## [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [36] 1 1 1 1 1 1 1 1 1 1 1 2
## Levels: 1 2 3
```

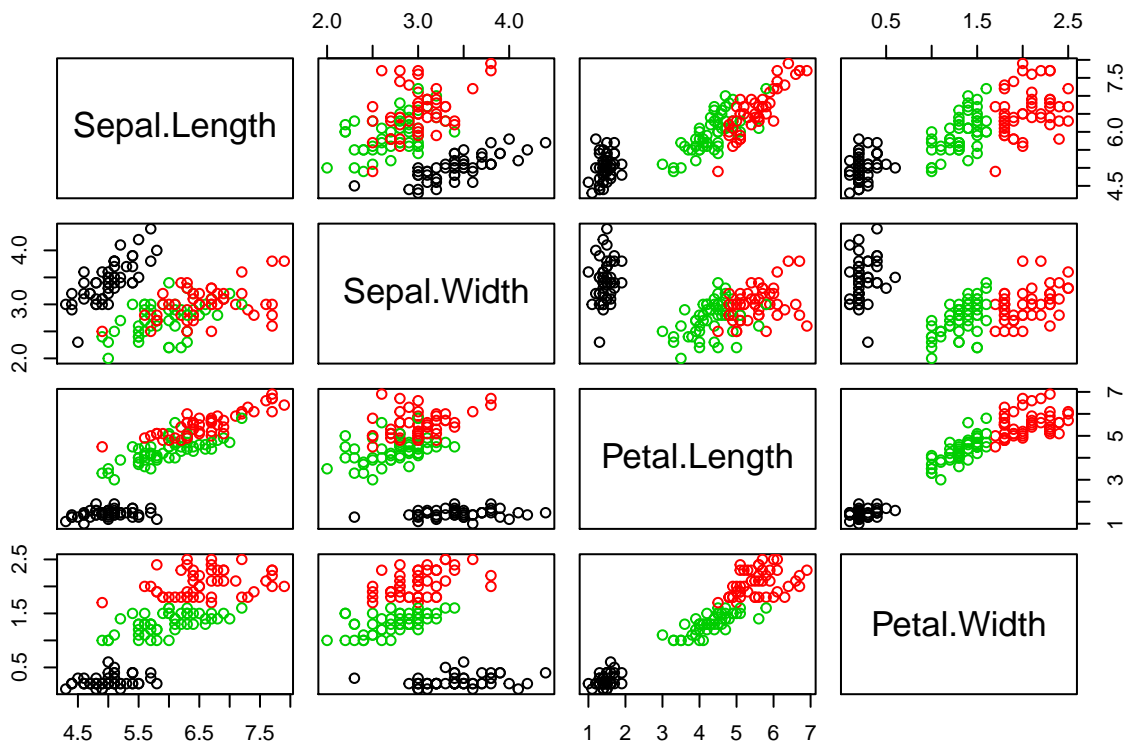
```
wine[,1][which(cluster.scale$new_group==3)]
```

```
## [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [36] 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## Levels: 1 2 3
```

“scale” function calculates each column’s mean and standard deviation, then scale each element by subtracting the mean and dividing by the standard deviation.

The scaled data improved the clustering results, but still failed to distinguish from wine type 2 and wine type3.

```
cluster2 <- kmeans_ft(iris[,-5])
with(iris, pairs(iris[-5], col=c(1:3)[cluster2$new_group]))
```



```
k <- kmeans_ft(iris[,1:4])
k
```

```
## $number_of_iter
## [1] 5
##
## $new_group
## [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [36] 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
## [71] 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1
## [106] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 2 2 1 1 1 1
## [141] 1 1 1 1 1 1 1 1 1 1
```

```
iris[,5][which(k$new_group==1)]
```

```
## [1] versicolor versicolor virginica virginica virginica virginica
## [7] virginica virginica virginica virginica virginica virginica
## [13] virginica virginica virginica virginica virginica virginica
## [19] virginica virginica virginica virginica virginica virginica
## [25] virginica virginica virginica virginica virginica virginica
## [31] virginica virginica virginica virginica virginica virginica
## [37] virginica virginica virginica virginica virginica virginica
## [43] virginica virginica virginica virginica virginica virginica
## Levels: setosa versicolor virginica
```

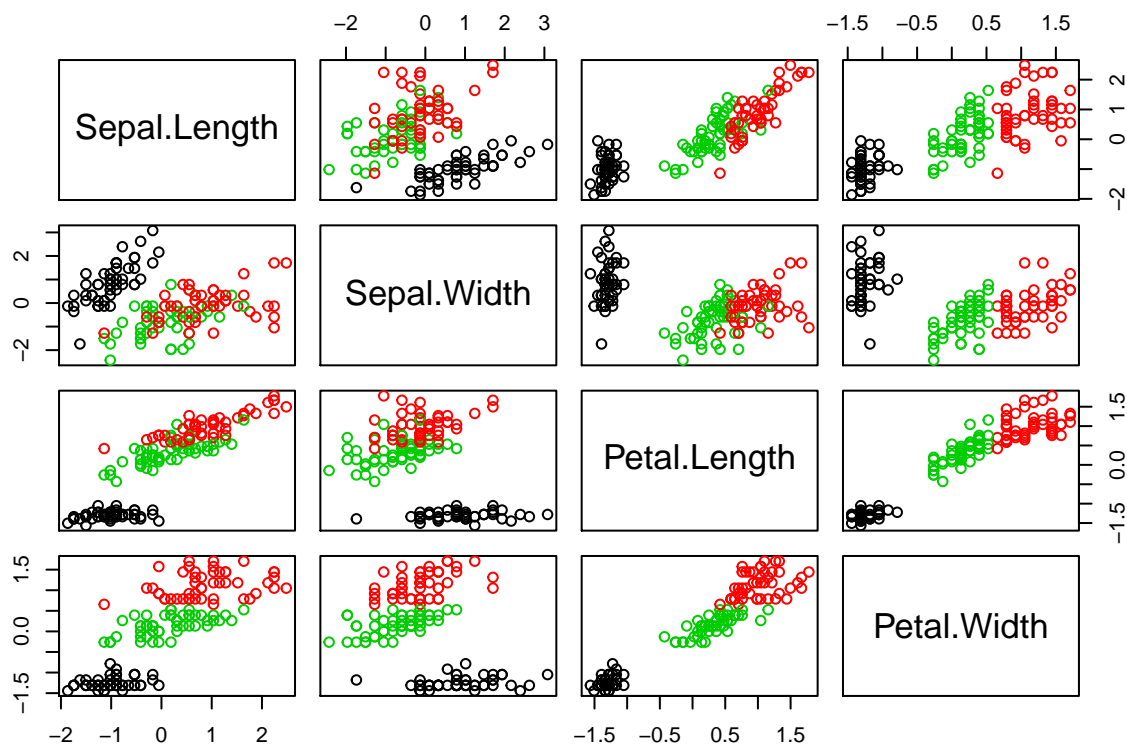
```
iris[,5][which(k$new_group==2)]
```

```
## [1] versicolor versicolor versicolor versicolor versicolor versicolor
## [7] versicolor versicolor versicolor versicolor versicolor versicolor
## [13] versicolor versicolor versicolor versicolor versicolor versicolor
## [19] versicolor versicolor versicolor versicolor versicolor versicolor
## [25] versicolor versicolor versicolor versicolor versicolor versicolor
## [31] versicolor versicolor versicolor versicolor versicolor versicolor
## [37] versicolor versicolor versicolor versicolor versicolor versicolor
## [43] versicolor versicolor versicolor versicolor versicolor versicolor
## [49] virginica virginica virginica virginica
## Levels: setosa versicolor virginica
```

```
iris[,5][which(k$new_group==3)]
```

```
## [1] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## [11] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## [21] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## [31] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## [41] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## Levels: setosa versicolor virginica
```

```
cluster2.scale <- kmeans_ft(scale(iris[,1:4]))
with(iris, pairs(scale(iris[,1:4]), col=c(1:3)[cluster2.scale$new_group]))
```



```
k2 <- kmeans_ft(scale(iris[,1:4]))
k2
```

```
## $number_of_iter
## [1] 5
##
## $new_group
## [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [36] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [71] 2 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2
## [106] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2
## [141] 2 2 2 2 2 2 2 2 2 2
```

```
iris[,5][which(k2$new_group==1)]
```

```
## [1] versicolor versicolor versicolor versicolor versicolor versicolor
## [7] versicolor versicolor versicolor versicolor versicolor versicolor
## [13] versicolor versicolor versicolor versicolor versicolor versicolor
## [19] versicolor versicolor versicolor versicolor versicolor versicolor
## [25] versicolor versicolor versicolor versicolor versicolor versicolor
## [31] versicolor versicolor versicolor versicolor versicolor versicolor
## [37] versicolor versicolor versicolor versicolor versicolor versicolor
## [43] versicolor versicolor versicolor versicolor versicolor versicolor
## [49] virginica virginica virginica virginica
## Levels: setosa versicolor virginica
```

```
iris[,5][which(k2$new_group==2)]
```

```
## [1] versicolor versicolor virginica virginica virginica virginica
## [7] virginica virginica virginica virginica virginica virginica
## [13] virginica virginica virginica virginica virginica virginica
## [19] virginica virginica virginica virginica virginica virginica
## [25] virginica virginica virginica virginica virginica virginica
## [31] virginica virginica virginica virginica virginica virginica
## [37] virginica virginica virginica virginica virginica virginica
## [43] virginica virginica virginica virginica virginica virginica
## Levels: setosa versicolor virginica
```

```
iris[,5][which(k2$new_group==3)]
```

```
## [1] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## [11] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## [21] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## [31] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## [41] setosa setosa setosa setosa setosa setosa setosa setosa setosa setosa
## Levels: setosa versicolor virginica
```

Clusters of both scaled and unscaled data classified the three different species of Iris data almost perfectly.