Lab 7: K means and Decision trees

Makesh Srinivasan

3/31/2022

Registration number: 19BCE1717 Faculty: Prof. Parvathi R Slot: L55 + L56 Course code: CSE3020

K-means clustering and Decision trees

Sections:

- Part 1: (Unsupervised) K-means clustering
- Part 2: (Supervised) Decision trees

K-means clustering and visualisation

Instructions: Other than IRIS dataset, use a dataset to perform clustering and visualise the same I have used CRABS datase from the MASS package.

1. Load the crabs dataset and view the data.

```
rm(list=ls())
library(MASS)
data("crabs") # load crabs Dataset
str(crabs) #view structure of dataset
```

```
## 'data.frame':    200 obs. of  8 variables:
##  $ sp   : Factor w/ 2 levels "B","O": 1 1 1 1 1 1 1 1 1 1 ...
##  $ sex  : Factor w/ 2 levels "F","M": 2 2 2 2 2 2 2 2 2 2 ...
##  $ index: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ FL   : num  8.1 8.8 9.2 9.6 9.8 10.8 11.1 11.6 11.8 11.8 ...
##  $ RW   : num  6.7 7.7 7.8 7.9 8 9 9.9 9.1 9.6 10.5 ...
##  $ CL   : num  16.1 18.1 19 20.1 20.3 23 23.8 24.5 24.2 25.2 ...
##  $ CW   : num  19 20.8 22.4 23.1 23 26.5 27.1 28.4 27.8 29.3 ...
##  $ BD   : num  7 7.4 7.7 8.2 8.2 9.8 9.8 10.4 9.7 10.3 ...
```

2. Display the Statistical Summary of the dataset

```
#2. Display the Statistical Summary of the dataset
summary(crabs) #view statistical summary of dataset
```

```
##  sp       sex          index            FL              RW              CL
##  B:100   F:100   Min.   : 1.0   Min.   : 7.20   Min.   : 6.50   Min.   :14.70
##  O:100   M:100   1st Qu.:13.0   1st Qu.:12.90   1st Qu.:11.00   1st Qu.:27.27
##                  Median :25.5   Median :15.55   Median :12.80   Median :32.10
##                  Mean   :25.5   Mean   :15.58   Mean   :12.74   Mean   :32.11
##                  3rd Qu.:38.0   3rd Qu.:18.05   3rd Qu.:14.30   3rd Qu.:37.23
##                  Max.   :50.0   Max.   :23.10   Max.   :20.20   Max.   :47.60
##        CW              BD
##  Min.   :17.10   Min.   : 6.10
##  1st Qu.:31.50   1st Qu.:11.40
##  Median :36.80   Median :13.90
##  Mean   :36.41   Mean   :14.03
##  3rd Qu.:42.00   3rd Qu.:16.60
##  Max.   :54.60   Max.   :21.60
```

```
head(crabs) #view top  rows of dataset
```

```
##   sp sex index   FL  RW   CL   CW  BD
## 1  B   M     1  8.1 6.7 16.1 19.0 7.0
## 2  B   M     2  8.8 7.7 18.1 20.8 7.4
## 3  B   M     3  9.2 7.8 19.0 22.4 7.7
## 4  B   M     4  9.6 7.9 20.1 23.1 8.2
## 5  B   M     5  9.8 8.0 20.3 23.0 8.2
## 6  B   M     6 10.8 9.0 23.0 26.5 9.8
```

3. Apply the preprocessing to remove the class attribute (sp)

```
crabs.new <- crabs[,c(3,4,5,6,7,8)]
crabs.class <- crabs[,"sp"]
head(crabs.new)
```

```
##   index   FL  RW   CL   CW  BD
## 1     1  8.1 6.7 16.1 19.0 7.0
## 2     2  8.8 7.7 18.1 20.8 7.4
## 3     3  9.2 7.8 19.0 22.4 7.7
## 4     4  9.6 7.9 20.1 23.1 8.2
## 5     5  9.8 8.0 20.3 23.0 8.2
## 6     6 10.8 9.0 23.0 26.5 9.8
```

```
head(crabs.class)
```

```
## [1] B B B B B B
## Levels: B O
```

#4. Normalisation

```
# Create a function to normalize the data before clustering
normalize <- function(x){
  return ((x-min(x))/(max(x)-min(x)))
}

crabs.new$FL<- normalize(crabs.new$FL)
crabs.new$RW<- normalize(crabs.new$RW)
crabs.new$CL<- normalize(crabs.new$CL)
crabs.new$CW<- normalize(crabs.new$CW)
crabs.new$BD<- normalize(crabs.new$BD)
```

```
crabs.new$index<- normalize(crabs.new$index)
head(crabs.new)
```

```
##        index         FL         RW         CL         CW         BD
## 1 0.00000000 0.05660377 0.01459854 0.04255319 0.05066667 0.05806452
## 2 0.02040816 0.10062893 0.08759124 0.10334347 0.09866667 0.08387097
## 3 0.04081633 0.12578616 0.09489051 0.13069909 0.14133333 0.10322581
## 4 0.06122449 0.15094340 0.10218978 0.16413374 0.16000000 0.13548387
## 5 0.08163265 0.16352201 0.10948905 0.17021277 0.15733333 0.13548387
## 6 0.10204082 0.22641509 0.18248175 0.25227964 0.25066667 0.23870968
```

5. Apply k-means clustering algorithm with k = 2

```
result<- kmeans(crabs.new,2) #aplly k-means algorithm with no. of centroids(k)=2
```

6. Find the number of records in each cluster

```
result$size # gives no. of records in each cluster
```

```
## [1] 101  99
```

7. Display the cluster center data point values

```
result$centers # gives value of cluster center datapoint value(2 centers for k=2)
```

```
##        index        FL        RW        CL        CW        BD
## 1 0.2564154 0.3542562 0.3132182 0.3549610 0.3444224 0.3378473
## 2 0.7485055 0.7037037 0.6003834 0.7066409 0.6891313 0.6889541
```

8. Display the cluster vector showing the cluster where each record falls

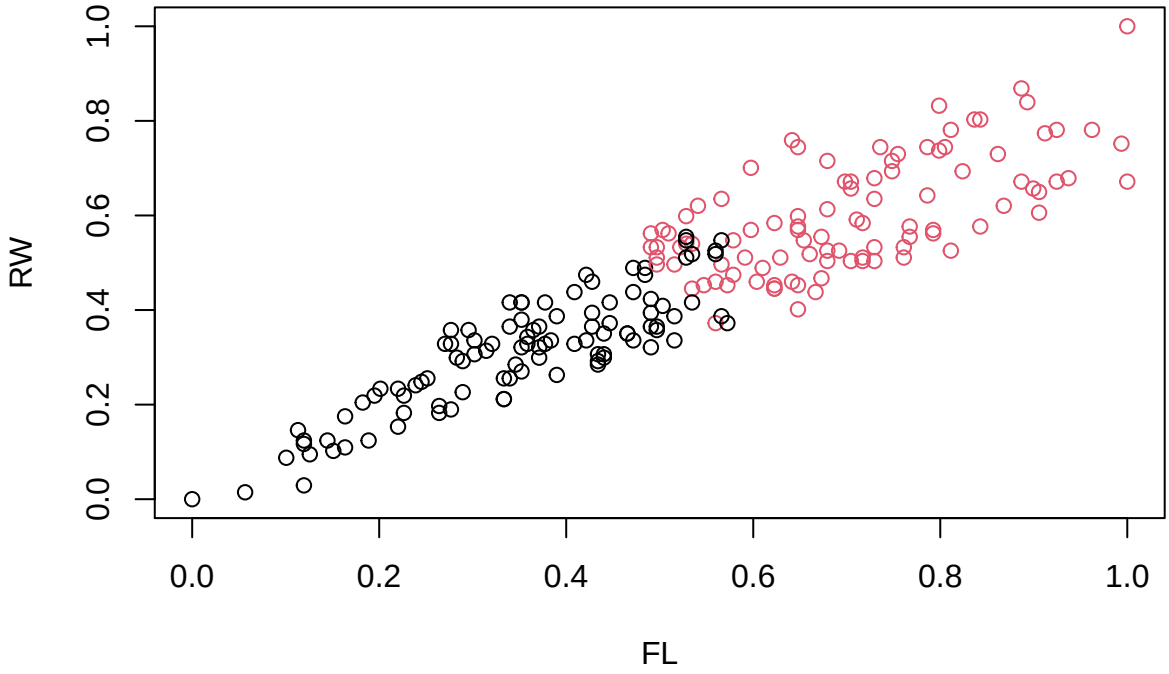
```
result$cluster #gives cluster vector showing the cluster where each record falls
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##   1   1   1   1   1   1   1   2   2   2   2   2   2   2   2   2   2   2   2   2
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   2   2   2   2   2   2   2   2   2   2   1   1   1   1   1   1   1   1   1   1
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   1   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   1   1   1   1   1   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##   2   2   2   2   2   2   2   2   2   2   1   1   1   1   1   1   1   1   1   1
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   1   1   1   1   1   1   1   1   2   2   2   2   2   2   2   2   2   2   2   2
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

```
# Verify results of clustering
par(mfrow=c(2,2), mar=c(5,4,2,2))
```

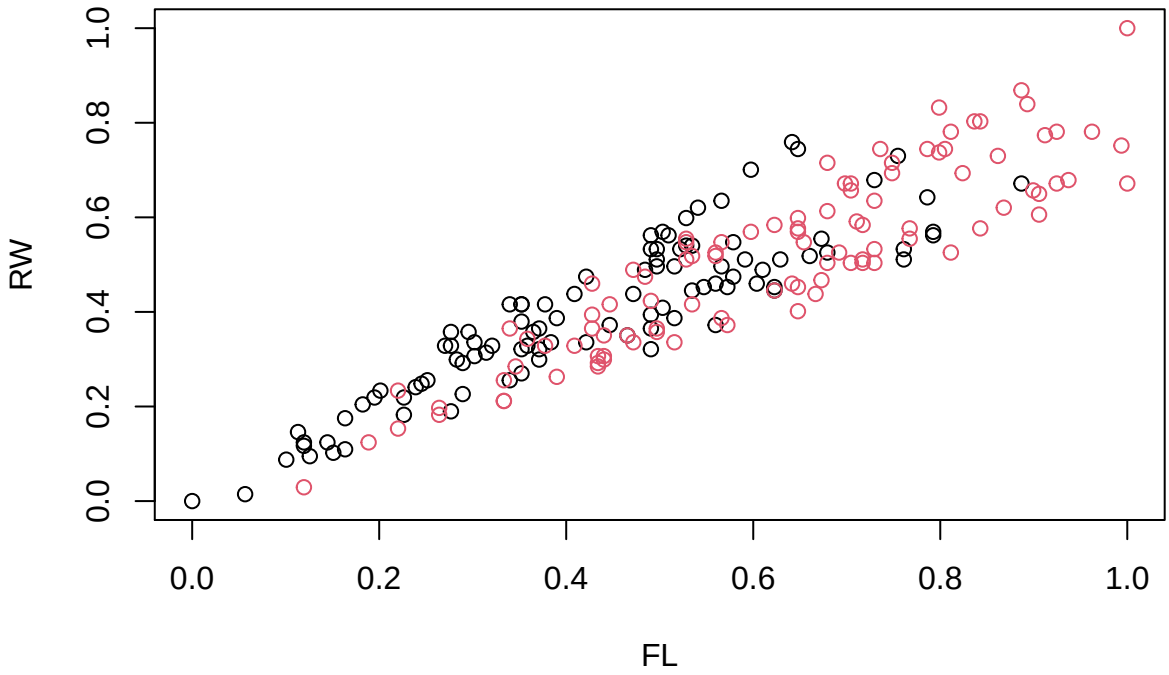
#9. Plot to see how FL and RW data points have been distributed in clusters

```
plot(crabs.new[c(2,3)], col=result$cluster)
```



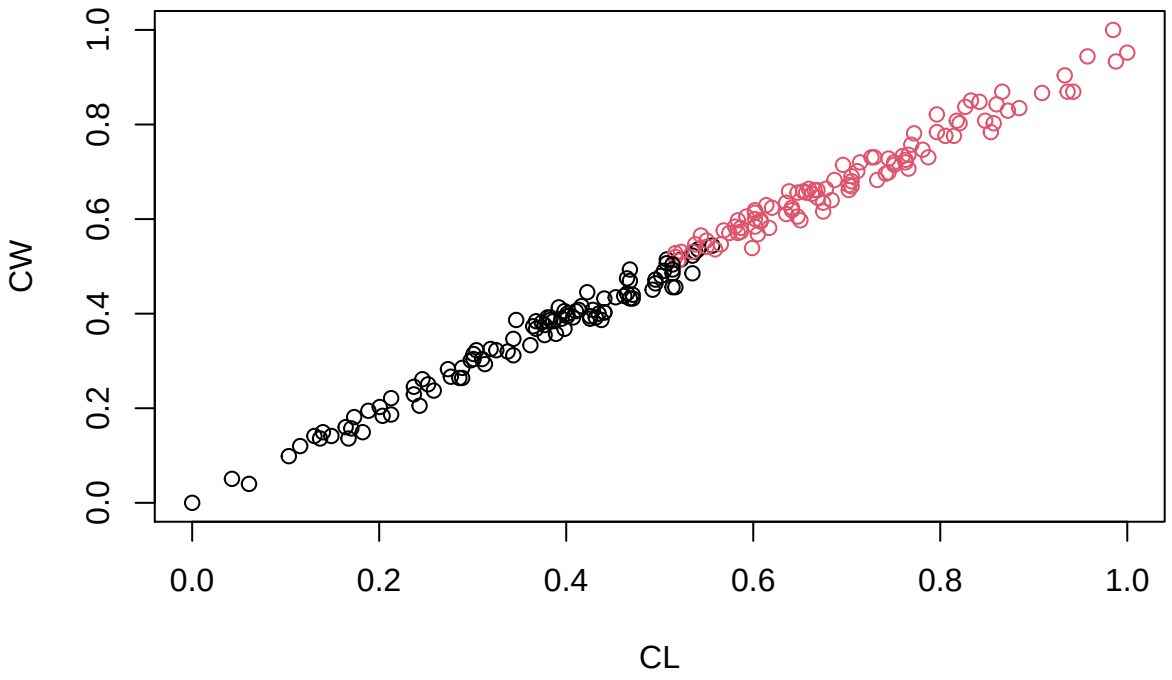
10. Plot to see how FL and RW data points have been distributed originally as per "class" attribute in dataset

```
plot(crabs.new[c(2,3)], col=crabs.class)
```



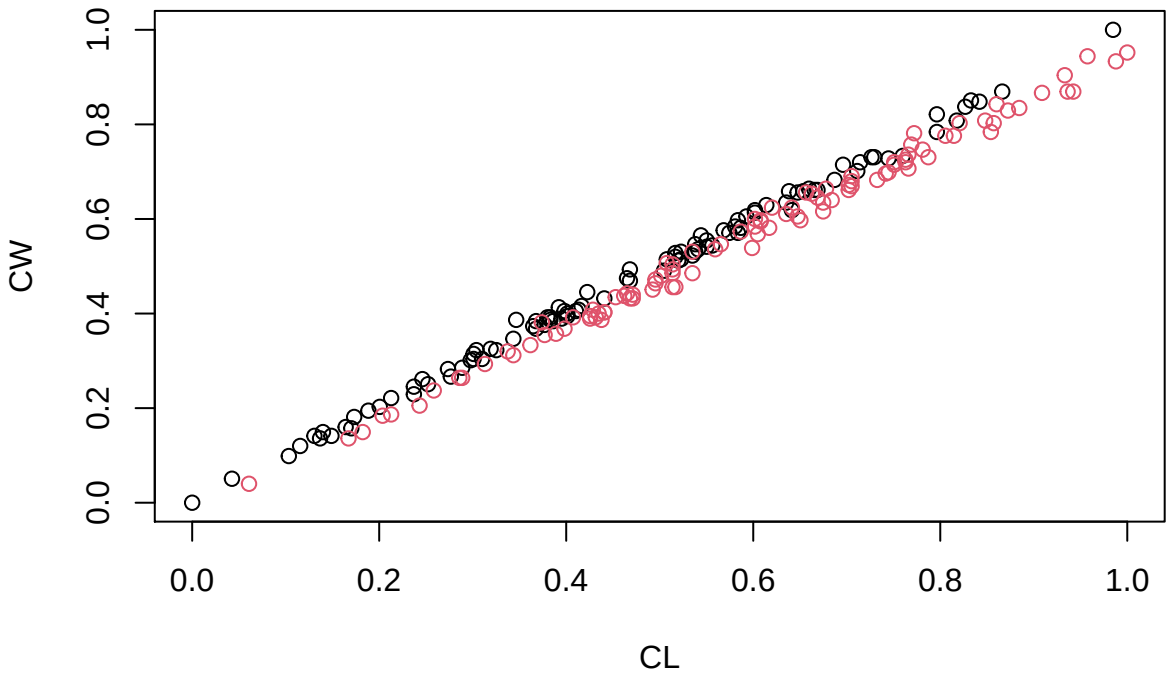
11.Plot to see how CL and CW data points have been distributed in clusters

```
plot(crabs.new[c(4,5)], col=result$cluster)
```



12.Plot to see how CL and CW data points have been distributed originally as per "class" attribute in dataset

```
plot(crabs.new[c(4,5)], col=crabs.class)
```



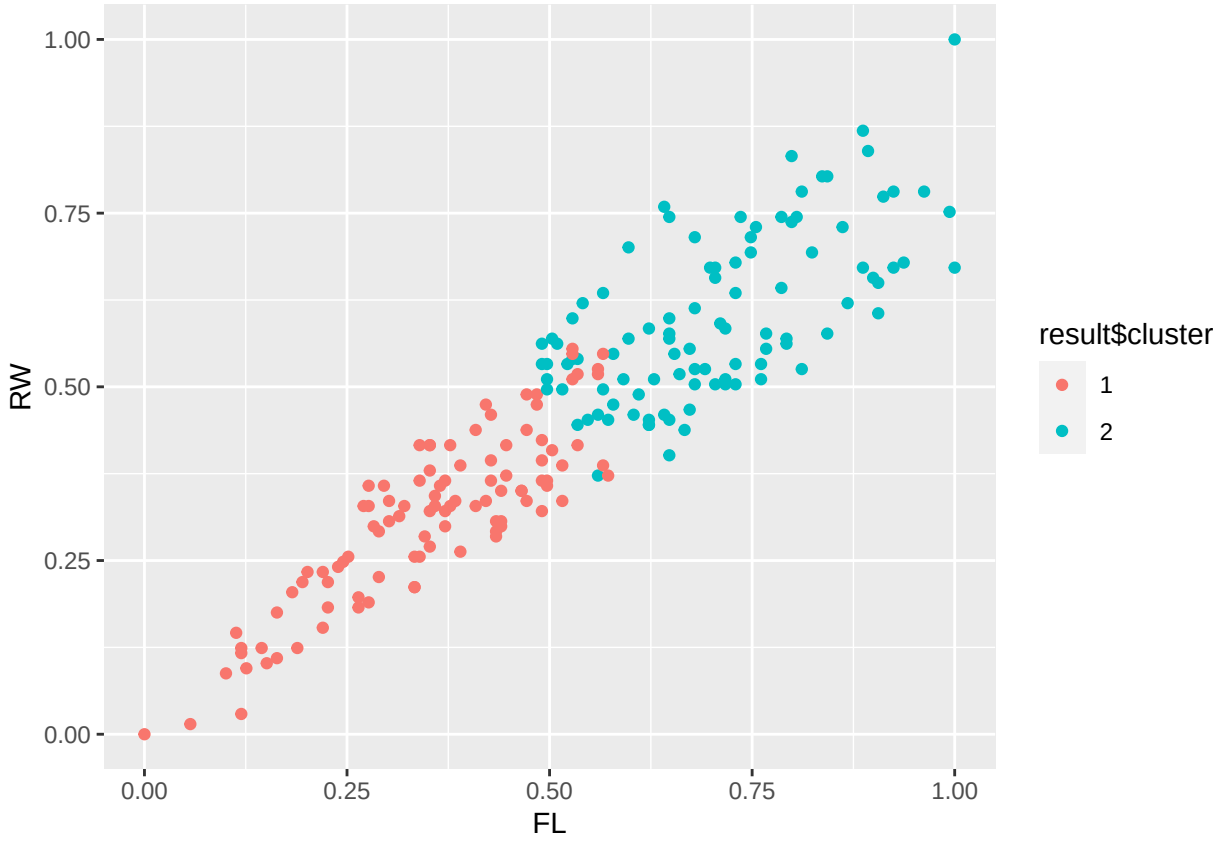

```
result$cluster <- as.factor(result$cluster)
```

```
library(ggplot2)
```

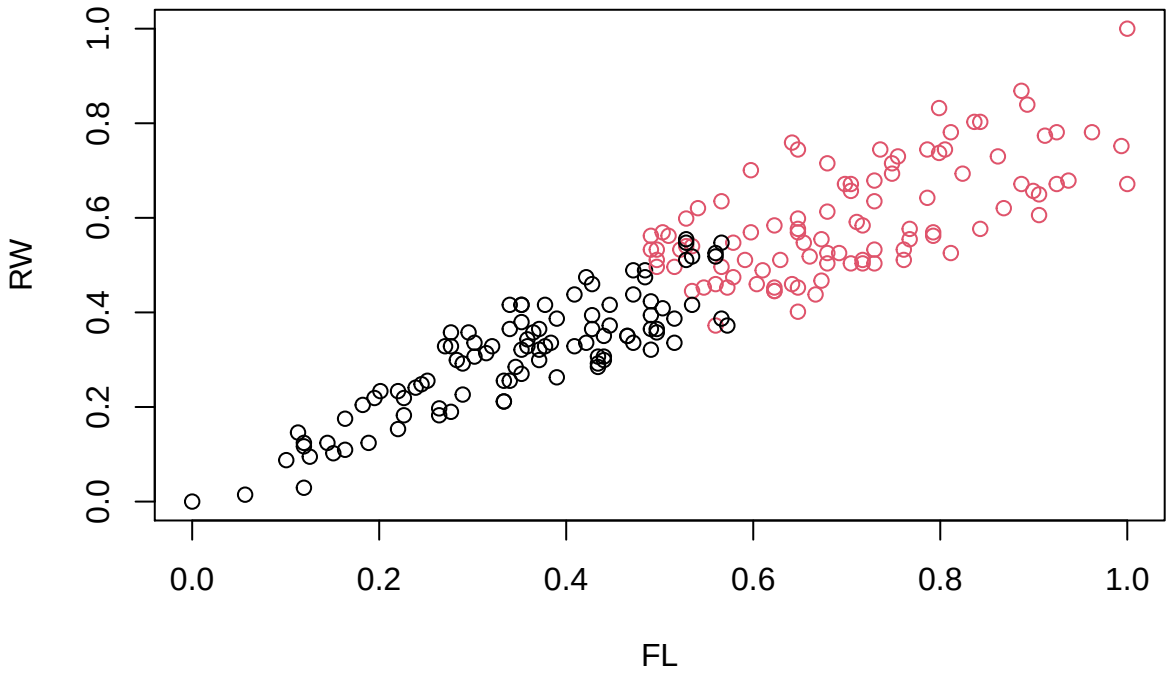
```
## Warning: package 'ggplot2' was built under R version 4.0.2
```

14. Plot the clusterresults using ggplot

```
ggplot(crabs.new, aes(FL, RW, color = result$cluster)) + geom_point()
```

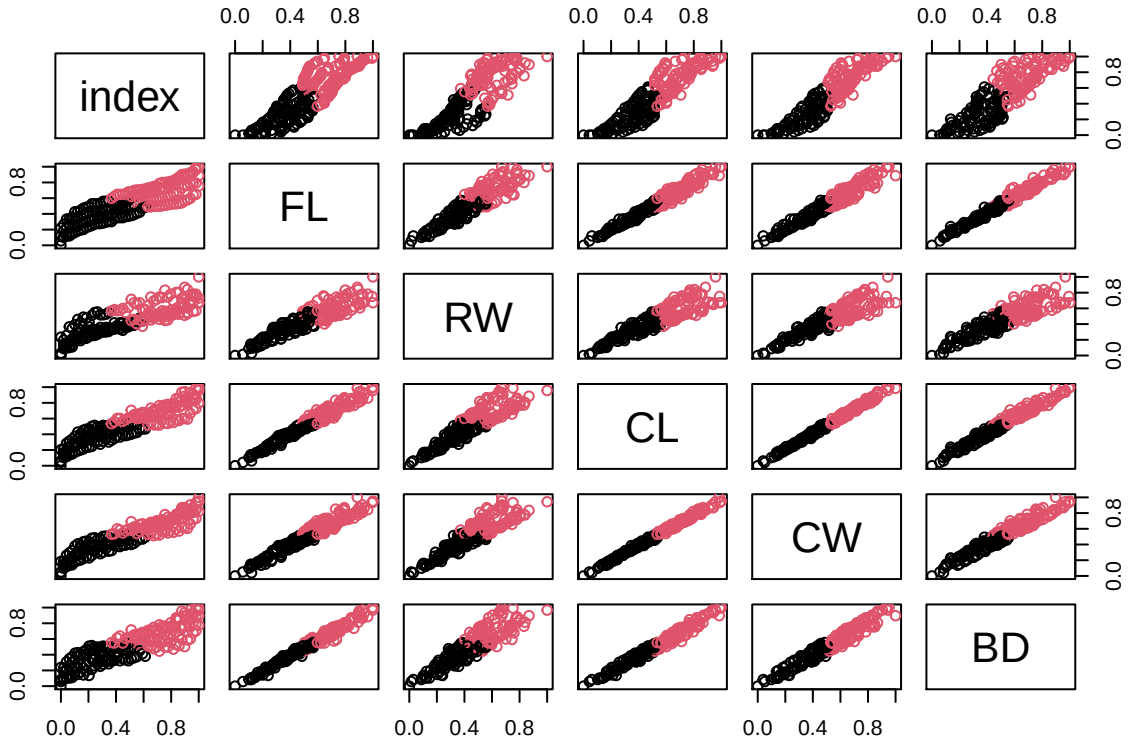



```
plot(crabs.new[c("FL", "RW")], col=result$cluster)
```



15. Display the clustering results with all parameters

```
plot(crabs.new[,], col=result$cluster)
```



16. Display the results in table

```
table(result$cluster,crabs.class) # Result of table shows that Cluster 1 corresponds to B, and Cluster
```

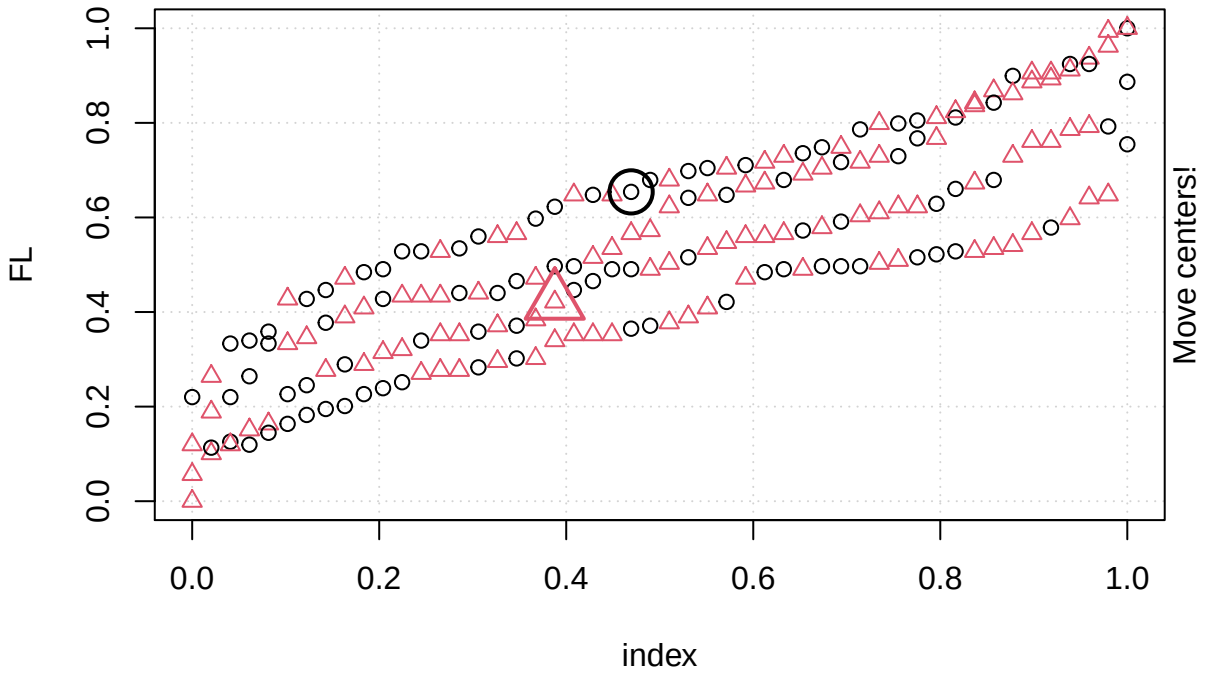
```
##    crabs.class
##      B  O
##   1 58 43
##   2 42 57
```

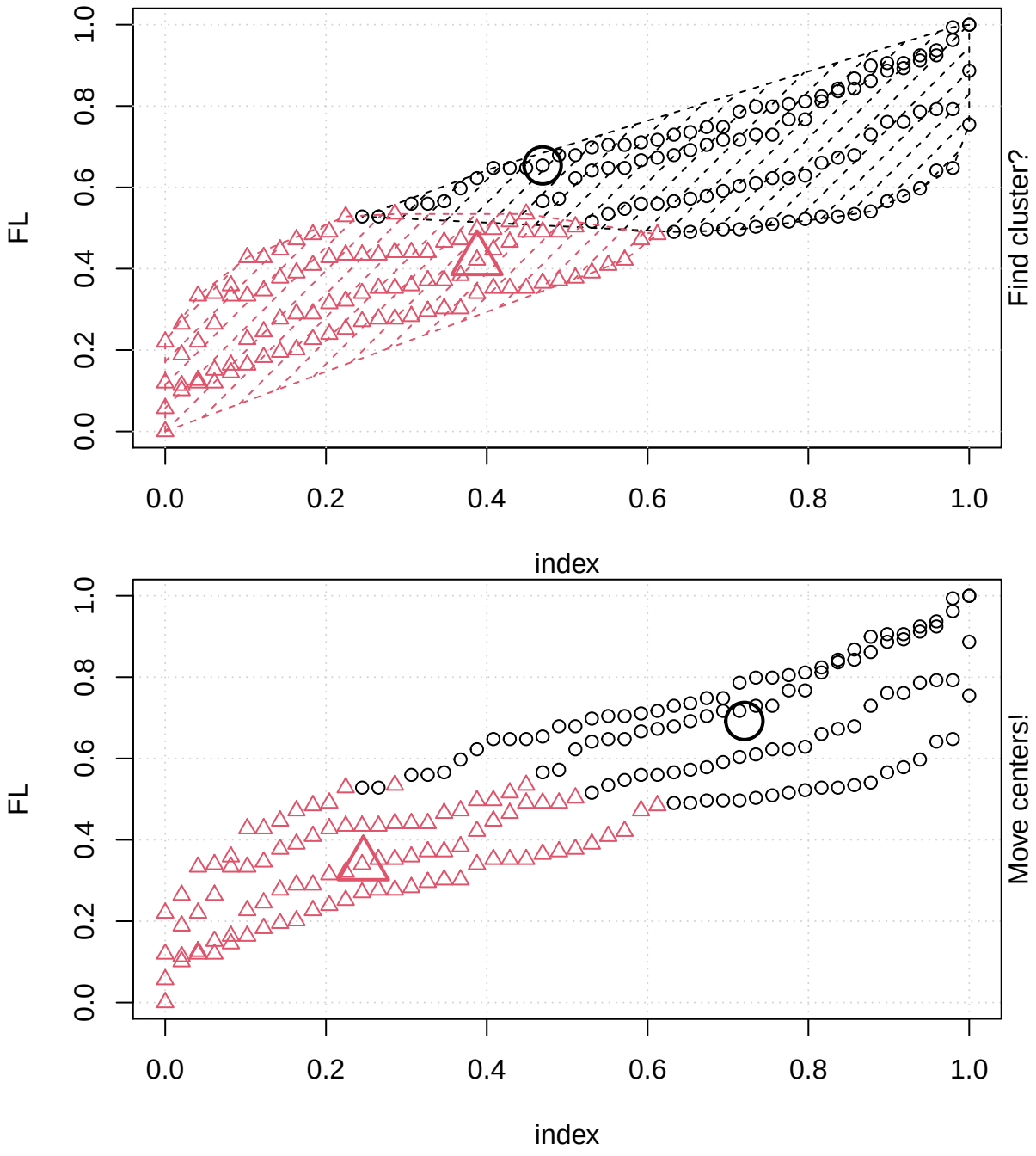
17. Display the K Means Algorithm with Animation and visualize the changes in the cluster center

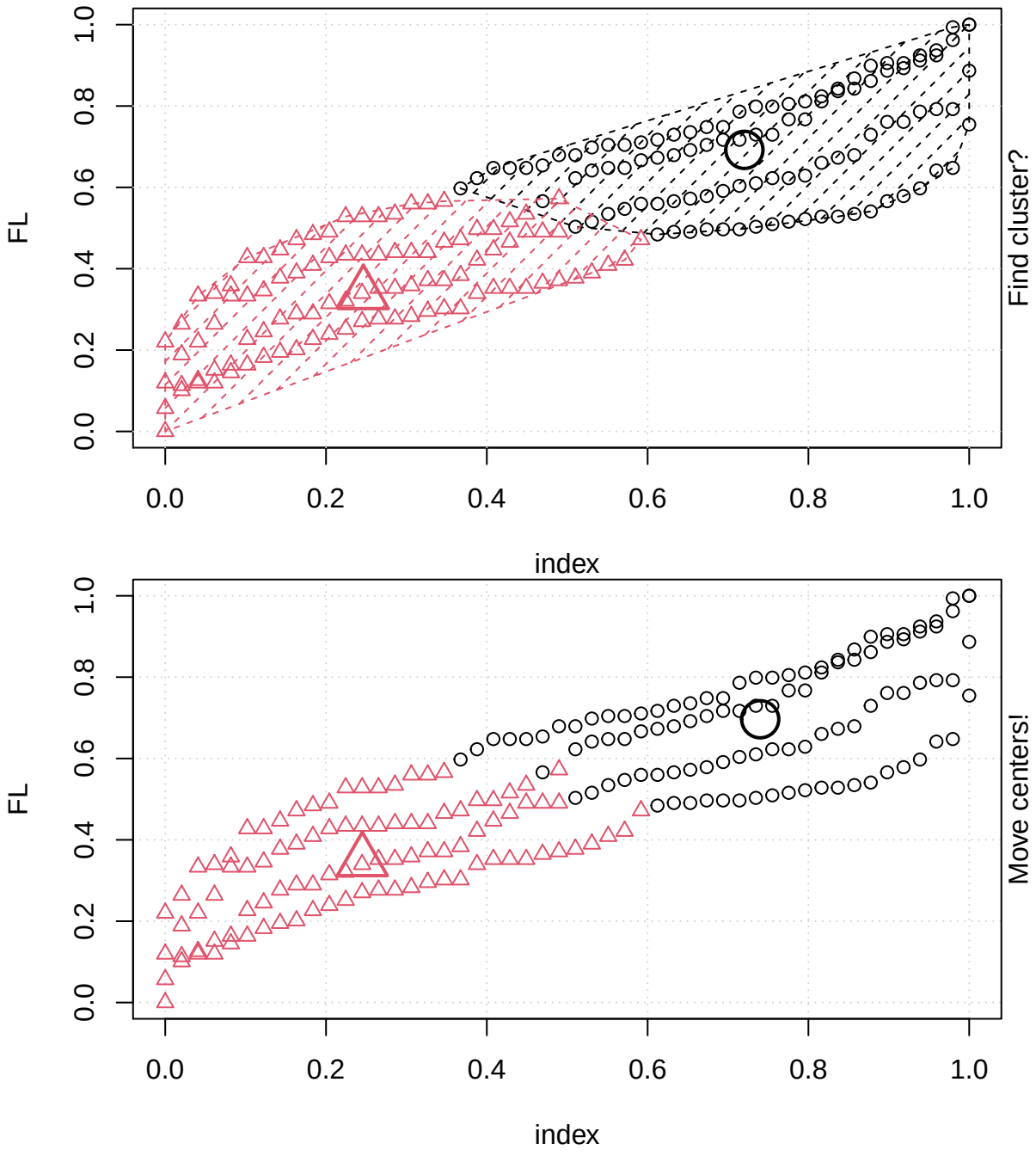
```
library(animation)
```

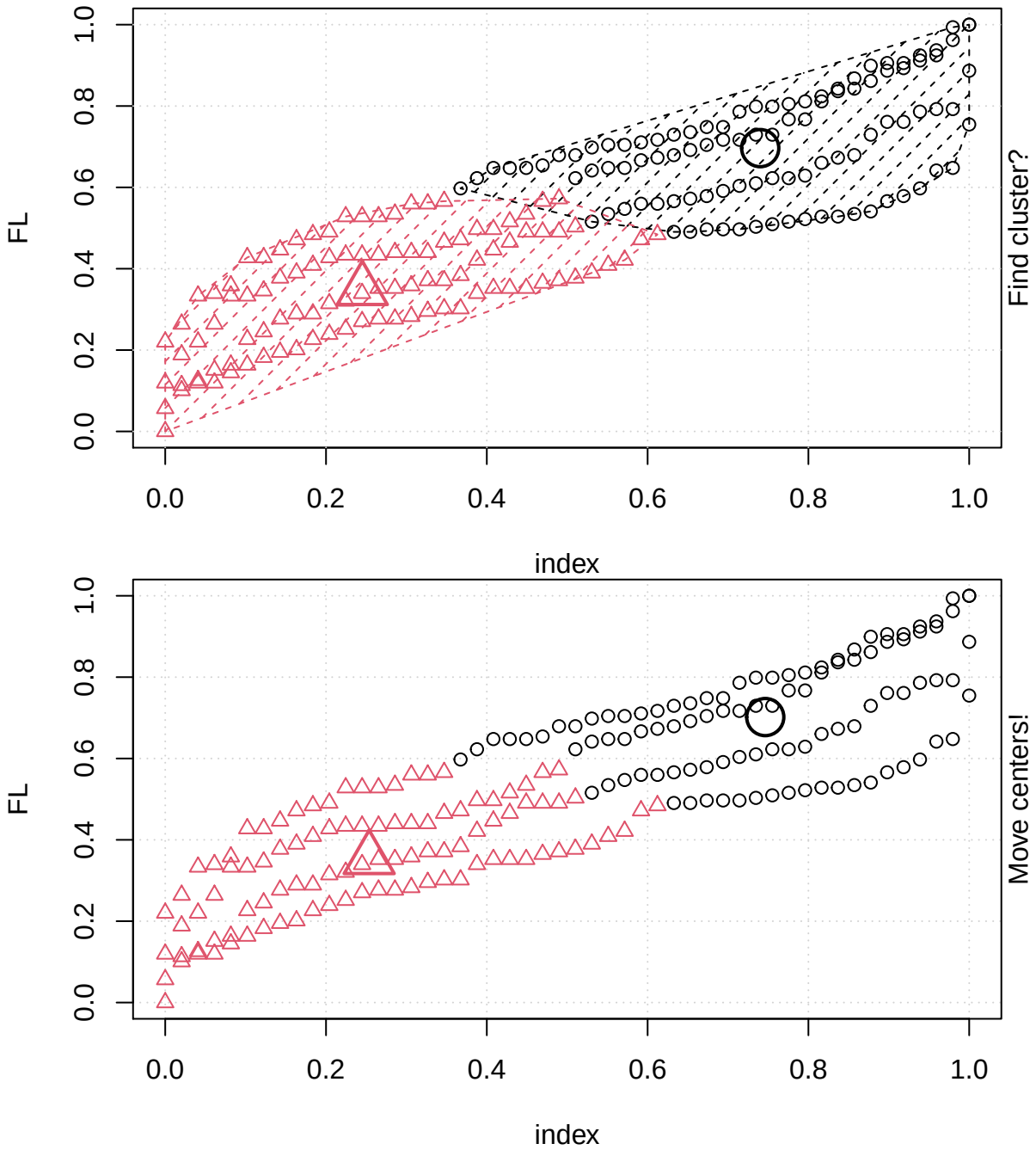
```
## Warning: package 'animation' was built under R version 4.0.2
```

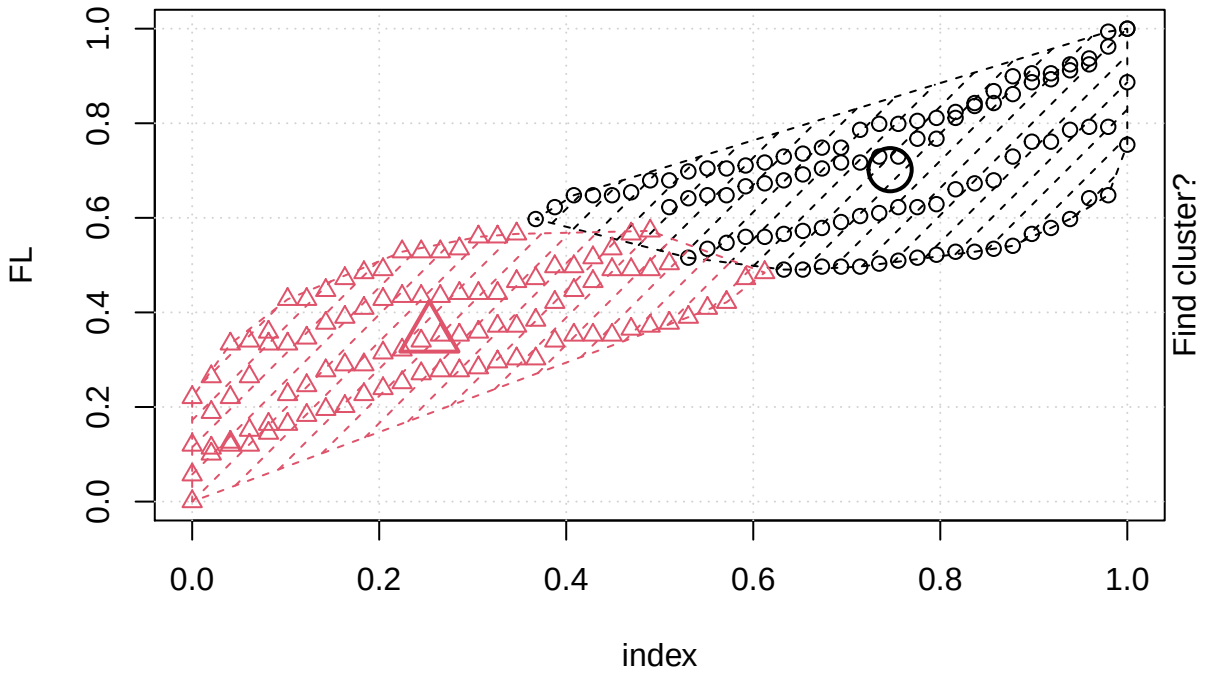
```
km1<-kmeans.ani(crabs.new,2)
```




FL
0.0
0.2
0.4
0.6
0.8
1.0
index
Find cluster?
Move centers!


FL
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
index
Find cluster?
FL
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
index
Move centers!


FL
0.0
0.2
0.4
0.6
0.8
1.0
index
Find cluster?
Move centers!



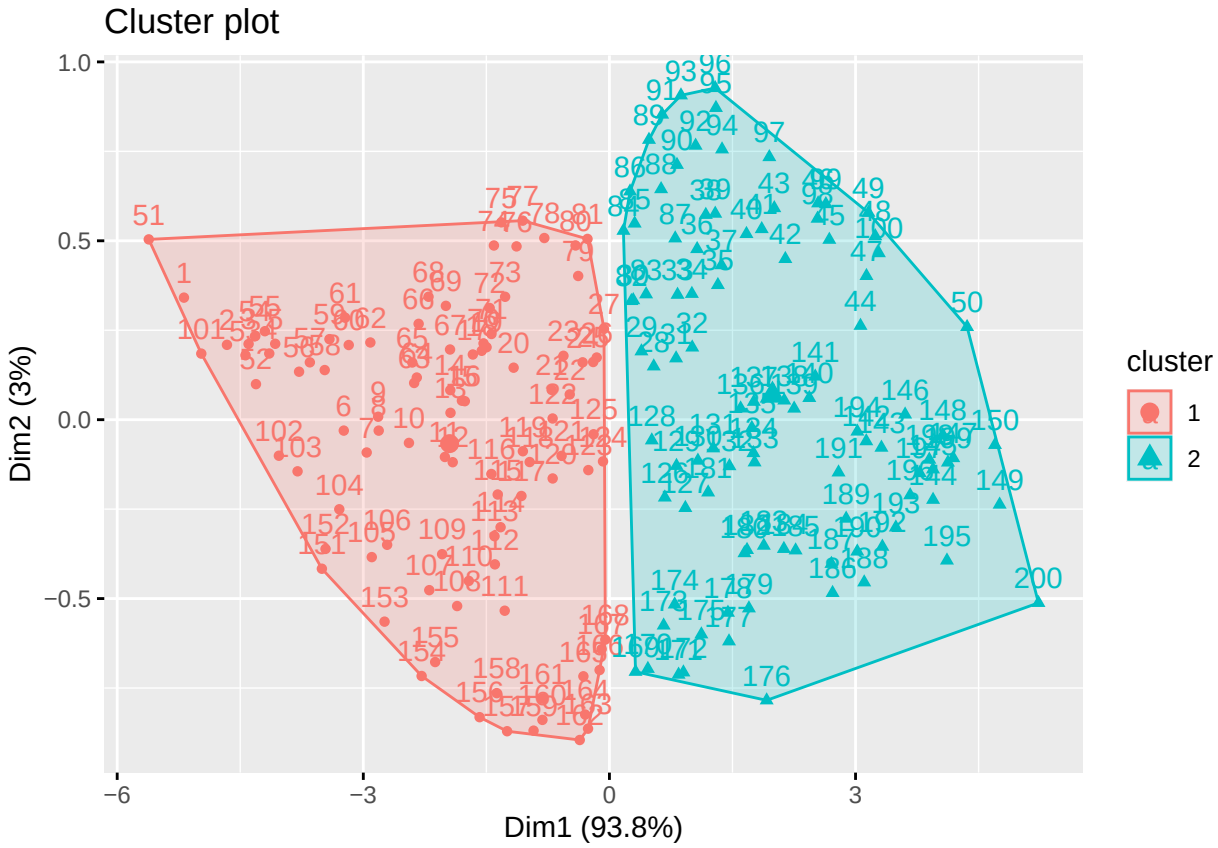
18. Import factoextra package and visualize the cluster result

```
library(factoextra) # clustering algorithms & visualization
```

```
## Warning: package 'factoextra' was built under R version 4.0.2
```

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

```
fviz_cluster(result, data = crabs.new)
```



19. Explore the cluster analysis result with various value of k like 3,4,5

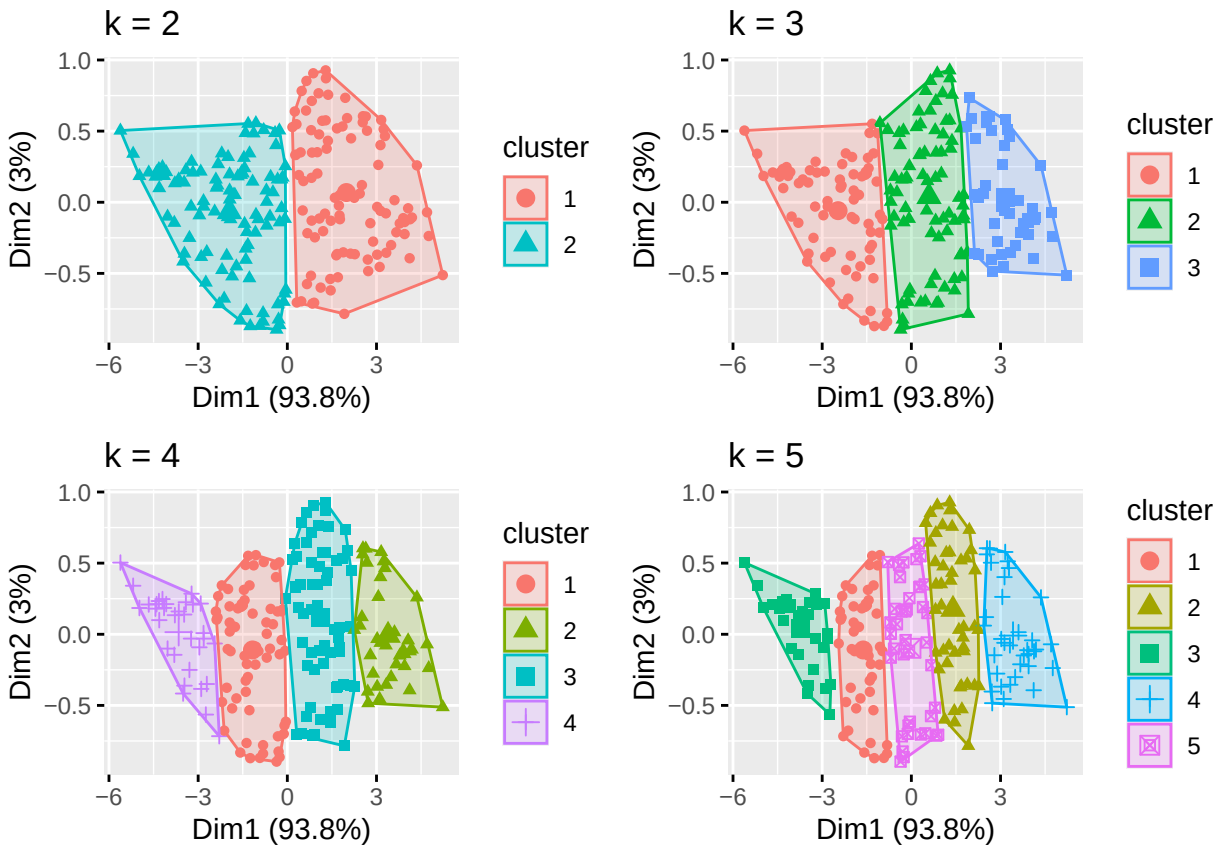
```
k2 <- kmeans(crabs.new, centers = 2, nstart = 25)
k3 <- kmeans(crabs.new, centers = 3, nstart = 25)
k4 <- kmeans(crabs.new, centers = 4, nstart = 25)
k5 <- kmeans(crabs.new, centers = 5, nstart = 25)
```

plots to compare

```
p1 <- fviz_cluster(k2, geom = "point", data = crabs.new) + ggtitle("k = 2")
p2 <- fviz_cluster(k3, geom = "point",  data = crabs.new) + ggtitle("k = 3")
p3 <- fviz_cluster(k4, geom = "point",  data = crabs.new) + ggtitle("k = 4")
p4 <- fviz_cluster(k5, geom = "point",  data = crabs.new) + ggtitle("k = 5")
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.0.2
```

```
grid.arrange(p1, p2, p3, p4, nrow = 2)
```



Decision trees

Instructions: Train a decision tree model and visualise the tree

Load the dataset

```
data <- crabs
shuffle_index <- sample(1:nrow(data))
data <- data[shuffle_index, ]
head(data)
```

```
##     sp sex index   FL   RW   CL   CW   BD
## 6    B   M     6 10.8  9.0 23.0 26.5  9.8
## 176  O   F    26 18.0 16.3 37.9 43.0 17.2
## 101  O   M     1  9.1  6.9 16.7 18.6  7.4
## 32   B   M    32 16.2 13.3 36.0 41.7 15.4
## 18   B   M    18 13.1 10.9 28.3 32.4 11.2
## 185  O   F    35 19.1 16.3 37.9 42.6 17.2
```

Train and test split

```
create_train_test <- function(d, size = 0.8, train = TRUE) {
    n_row = nrow(d)
    total_row = size * n_row
    train_sample <- 1: total_row
    if (train == TRUE) {
```

```
        return (d[train_sample, ])
    } else {
        return (d[-train_sample, ])
    }
}
```

80:20 ratio for the dataset

```
# Dataset is split into train and test
data_train <- create_train_test(data, 0.8, train = TRUE)
data_test <- create_train_test(data, 0.8, train = FALSE)
dim(data_train)
```

```
## [1] 160   8
```

Dimension of the test data

```
dim(data_test)
```

```
## [1] 40  8
```

8 columns and 40 rows

proportion of the train and test data with respect to species (label):

```
prop.table(table(data_train$sp))
```

```
##
##       B       O
## 0.51875 0.48125
```

```
prop.table(table(data_test$sp))
```

```
##
##     B     O
## 0.425 0.575
```

We can see that the distribution of B and O classes are almost similar. Hence we can proceed without having to do Sampling of the dataset.

Visualise the model

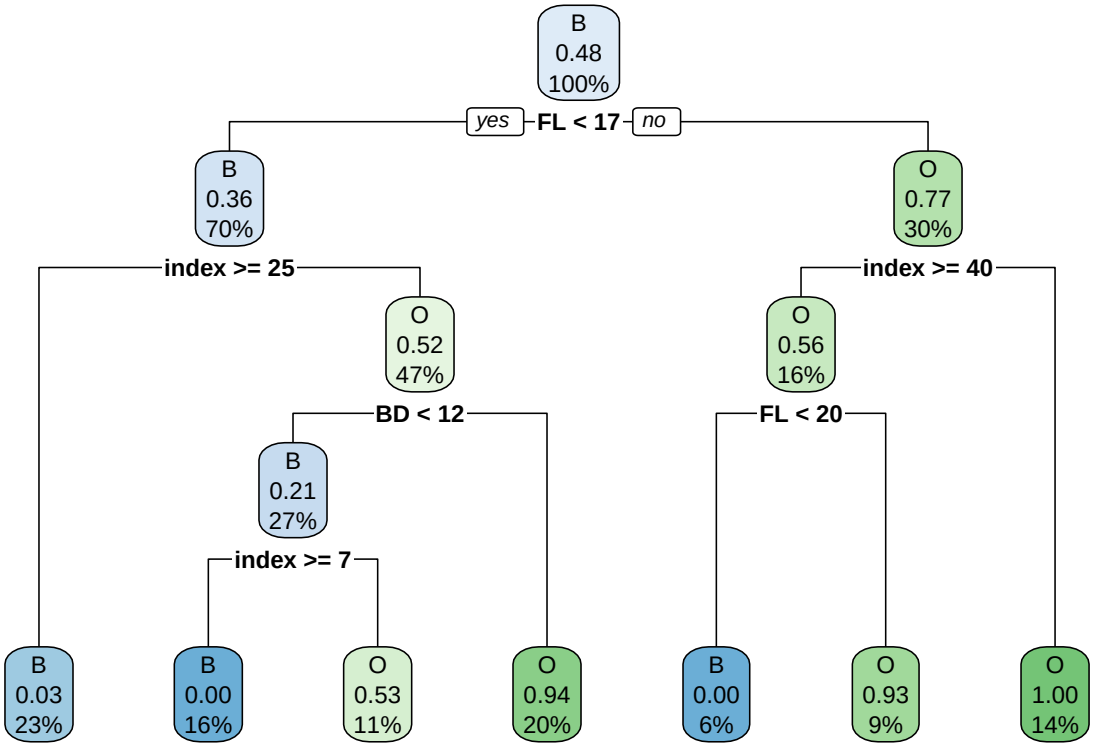
```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 4.0.2
```

```
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.0.2
```

```
fit <- rpart(sp~., data = data_train, method = 'class')
rpart.plot(fit, extra = 106)
```



The Decition tree is shown above. The model was generated and the output is as follows.

Confusion matrix of the predictions

```
predict_unseen <-predict(fit, data_test, type = 'class')
table_mat <- table(data_test$sp, predict_unseen)
table_mat
```

```
##    predict_unseen
##      B  O
##   B  9  8
##   O  3 20
```

We see that the first and last cells show the right predictions (True positives and negatives)

Accuracy (performance measure)

```
accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
print(paste('Accuracy for test', accuracy_Test))
```

```
## [1] "Accuracy for test 0.725"
```

Thus, our model is able to describe most of the dataset accurately.
