Pre-processing tasks

For pre-processing I decided to remove outliers first before scaling. This is because scaling can be affected by outliers.

The method I used to remove outliers was calculating the interquartile range and finding the limits for outliers, then using that to remove those above and below the limits. The upper limit is defined by :

The third quartile + 1.5 * IQR
The first quartile – 1.5 * IQR.

This method was applied to each column. The result was that there were 814 rows left in the data frame, meaning there were 50 outliers removed (see below).

To perform scaling, I used the normalise function on each column.

Code:

```
outliers <- function(x) {

  Q1 <- quantile(x, probs=.25)
  Q3 <- quantile(x, probs=.75)
  iqr = Q3-Q1

  upper_limit = Q3 + (iqr*1.5)
  lower_limit = Q1 - (iqr*1.5)

  x > upper_limit | x < lower_limit
}

remove_outliers <- function(df, cols = names(df)) {
  for (col in cols) {
    df <- df[!outliers(df[[col]]),]
  }
  df
}


vehicles <- remove_outliers(vehicles,
c('Comp','Circ','D.Circ','Rad.Ra','Pr.Axis.Ra','Max.L.Ra','Scat.Ra','Elong','Pr.Axis.Rect',

'Max.L.Rect','Sc.Var.Maxis','Sc.Var.maxis','Ra.Gyr','Skew.Maxis','Skew.maxis','Kurt.maxis','Ku
rt.Maxis','Holl.Ra'))
```

```
vehicles[c('Comp','Circ','D.Circ','Rad.Ra','Pr.Axis.Ra','Max.L.Ra','Scat.Ra','Elong','Pr.Axis.Rect',

'Max.L.Rect','Sc.Var.Maxis','Sc.Var.maxis','Ra.Gyr','Skew.Maxis','Skew.maxis','Kurt.maxis','Ku
rt.Maxis','Holl.Ra')] <-
lapply(vehicles[c('Comp','Circ','D.Circ','Rad.Ra','Pr.Axis.Ra','Max.L.Ra','Scat.Ra','Elong','Pr.Axi
s.Rect',

'Max.L.Rect','Sc.Var.Maxis','Sc.Var.maxis','Ra.Gyr','Skew.Maxis','Skew.maxis','Kurt.maxis','Ku
rt.Maxis','Holl.Ra')], function(x) c(normalise(x)))

vehicles[c(2:19)] <- lapply(vehicles[c(2:19)], function(x) c(normalise(x)))

> dim(vehicles)
> vehicles
```

Output:

```
[1] 814  20

# A tibble: 814 x 20
   Samples  Comp   Circ D.Circ Rad.Ra Pr.Axis.Ra Max.L.Ra Scat.Ra Elong Pr.Axis.Rect
Max.L.Rect Sc.Var.Maxis Sc.Var.maxis Ra.Gyr Skew.Maxis Skew.maxis Kurt.maxis
     <dbl> <dbl>  <dbl>  <dbl>  <dbl>     <dbl>    <dbl>   <dbl> <dbl>       <dbl>
<dbl>       <dbl>  <dbl>     <dbl>     <dbl>     <dbl>
 1      1 0.512 0.577   0.597  0.521     0.862     0.7  0.333  0.457      0.273
0.586      0.291      0.240  0.472      0.393     0.316     0.4
 2      2 0.419 0.308   0.611  0.261     0.345     0.6  0.247  0.543      0.182
0.357      0.253      0.179  0.308      0.464     0.474     0.35
 3      3 0.721 0.654   0.917  0.739     0.655     0.7  0.633  0.171      0.545
0.571      0.589      0.554  0.698      0.5       0.737     0.225
 4      4 0.465 0.308   0.583  0.387     0.552     0.6  0.213  0.571      0.182
0.357      0.190      0.154  0.113      0.143     0.316     0.25
 5      6 0.791 0.923   0.917  0.479     0.103     0.3  0.953  0          1
0.729      0.949      0.950  0.975      0.929     0.263     0.225
 6      7 0.558 0.385   0.458  0.486     0.621     0.3  0.273  0.457      0.182
0.357      0.291      0.217  0.396      0.25      0.684     0.025
 7      8 0.395 0.385   0.361  0.373     0.621     0.6  0.167  0.629      0.0909
0.4        0.203      0.119  0.346      0.286     0.158     0.075
 8      9 0.302 0.0385  0.306  0.254     0.483     0.4  0.0667 0.8        0
0.129      0.0696     0.0479 0.0189     0.179     0.105     0.35
 9     10 0.465 0.423   0.806  0.655     0.517     0.8  0.473  0.286      0.455
0.4        0.456      0.394  0.270      0.179     0.211     0.35
10     11 0.302 0.115   0.417  0.275     0.483     0.6  0.14   0.686      0.0909
0.171      0.146      0.101  0.113      0.25      0.105     0.25
# … with 804 more rows, and 3 more variables: Kurt.Maxis <dbl>, Holl.Ra <dbl>, Class <chr>
```

Define the number of cluster centres by showing all necessary steps/methods (via manual & automated tools)

I used two methods. One was bar plots using a table constructed from the NbClust method. For this, I varied between Manhattan distance and Euclidean distance. The second was plotting a line graph using WSS. The Euclidean bar plot showed that 2 was the best number of clusters, and the Manhattan bar plot had the same result with the graph looking very similar. The line graph showed the same results, where 2 is the elbow of the graph.

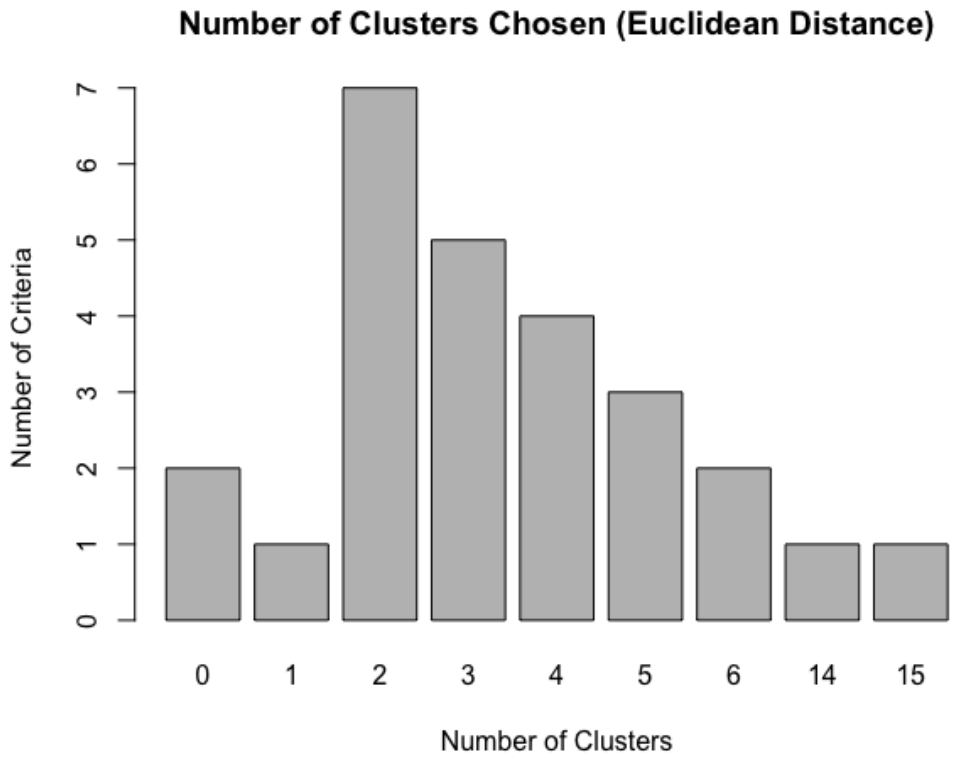
Code:

```
euclidean <- NbClust(data.train,
                     distance="euclidean",
                     min.nc=2,
                     max.nc=15,
                     method="kmeans",
                     index="all")

manhattan <- NbClust(data.train,
                     distance="manhattan",
                     min.nc=2,
                     max.nc=15,
                     method="kmeans",
                     index="all")

table(euclidean$Best.n[1,])

barplot(table(euclidean$Best.n[1,]),
        xlab="Number of Clusters",
        ylab="Number of Criteria",
        main="Number of Clusters Chosen (Euclidean Distance)")

table(manhattan$Best.n[1,])

barplot(table(manhattan$Best.n[1,]),
        xlab="Number of Clusters",
        ylab="Number of Criteria",
        main="Number of Clusters Chosen (Manhattan Distance)")wss <- 0
for (i in 1:15){
  wss[i] <-
    sum(kmeans(data.train, centers=i)$withinss)
}
plot(1:15,
     wss,
     type="b",
     xlab="Number of Clusters",
     ylab="Within groups sum of squares")
```

Output:

```
> euclidean <- NbClust(data.train,distance="euclidean",
min.nc=2,max.nc=15,method="kmeans",index="all")
*** : The Hubert index is a graphical method of determining the number of clusters.
        In the plot of Hubert index, we seek a significant knee that corresponds to a
        significant increase of the value of the measure i.e the significant peak in Hubert
        index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
        In the plot of D index, we seek a significant knee (the significant peak in Dindex
        second differences plot) that corresponds to a significant increase of the value of
        the measure.

*******************************************************************
* Among all indices:
* 7 proposed 2 as the best number of clusters
* 5 proposed 3 as the best number of clusters
* 4 proposed 4 as the best number of clusters
* 3 proposed 5 as the best number of clusters
* 2 proposed 6 as the best number of clusters
* 1 proposed 14 as the best number of clusters
* 1 proposed 15 as the best number of clusters

             ***** Conclusion *****

* According to the majority rule, the best number of clusters is  2


*******************************************************************
> table(euclidean$Best.n[1,])

 0  1  2  3  4  5  6 14 15
 2  1  7  5  4  3  2  1  1

> barplot(table(euclidean$Best.n[1,]),
+       xlab="Number of Clusters",
+       ylab="Number of Criteria",
```

```
+       main="Number of Clusters Chosen (Euclidean Distance)")
```




```
> manhattan <- NbClust(data.train,
+                 distance="manhattan",
+                 min.nc=2,
+                 max.nc=15,
+                 method="kmeans",
+                 index="all")
*** : The Hubert index is a graphical method of determining the number of clusters.
        In the plot of Hubert index, we seek a significant knee that corresponds to a
        significant increase of the value of the measure i.e the significant peak in Hubert
        index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
        In the plot of D index, we seek a significant knee (the significant peak in Dindex
        second differences plot) that corresponds to a significant increase of the value of
        the measure.

*******************************************************************
* Among all indices:
* 7 proposed 2 as the best number of clusters
* 5 proposed 3 as the best number of clusters
* 4 proposed 4 as the best number of clusters
* 3 proposed 5 as the best number of clusters
```

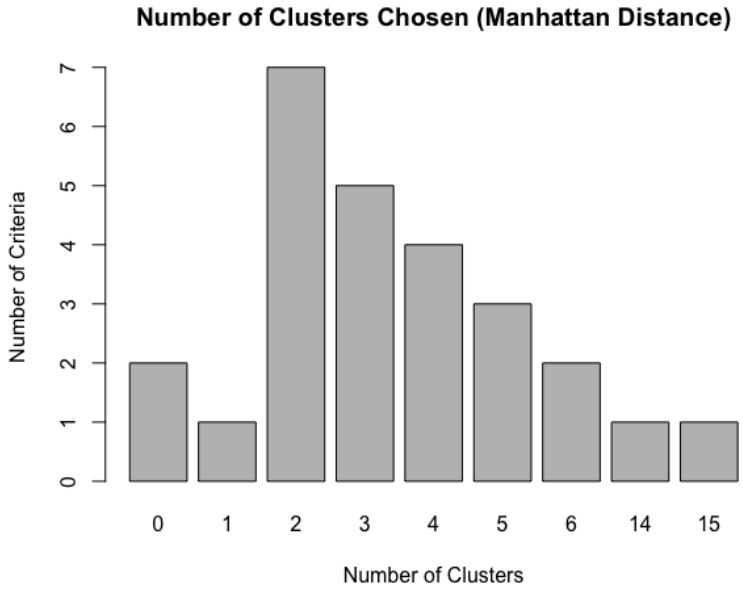
* 2 proposed 6 as the best number of clusters
* 1 proposed 14 as the best number of clusters
* 1 proposed 15 as the best number of clusters

***** Conclusion *****

* According to the majority rule, the best number of clusters is 2

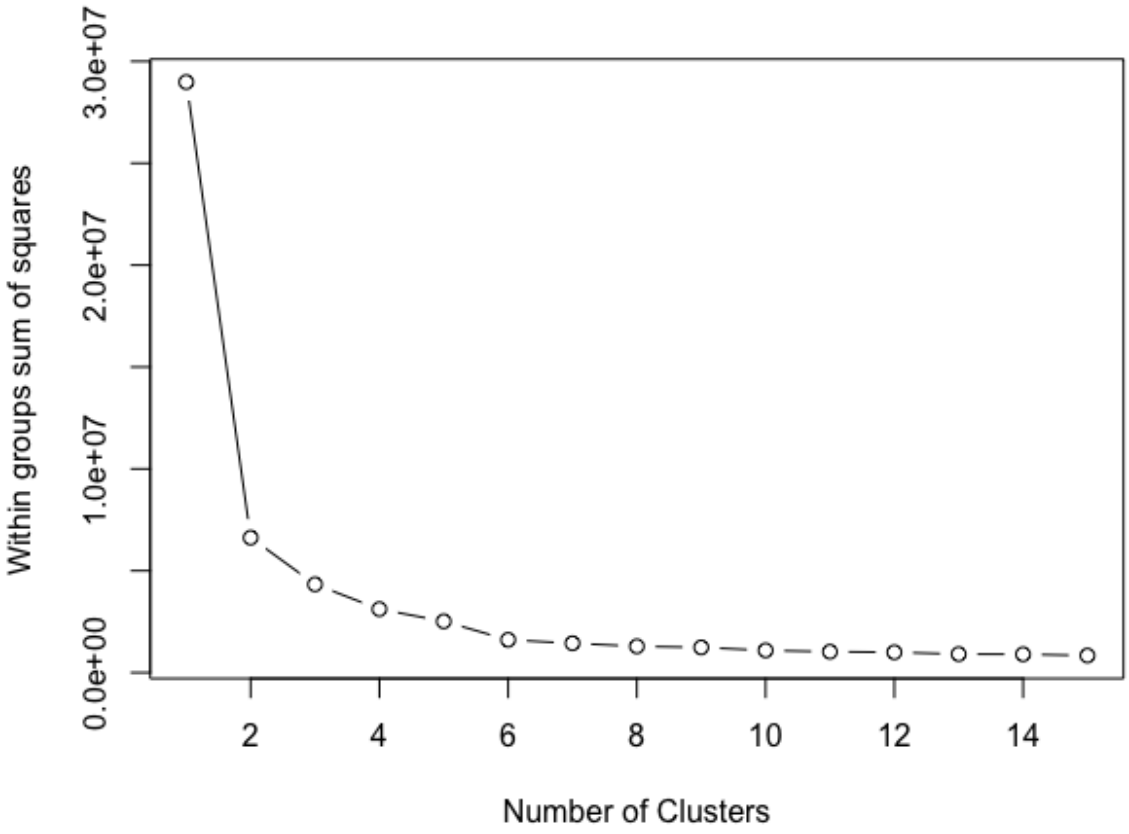
****

```
> table(manhattan$Best.n[1,])

 0  1  2  3  4  5  6 14 15
 2  1  7  5  4  3  2  1  1
>
> barplot(table(manhattan$Best.n[1,]),
+         xlab="Number of Clusters",
+         ylab="Number of Criteria",
+         main="Number of Clusters Chosen (Manhattan Distance)")
```




```
> plot(1:15,
+      wss,
+      type="b",
+      xlab="Number of Clusters",
```

+ ylab="Within groups sum of squares")



K-means analysis for each attempt

2 Clusters:

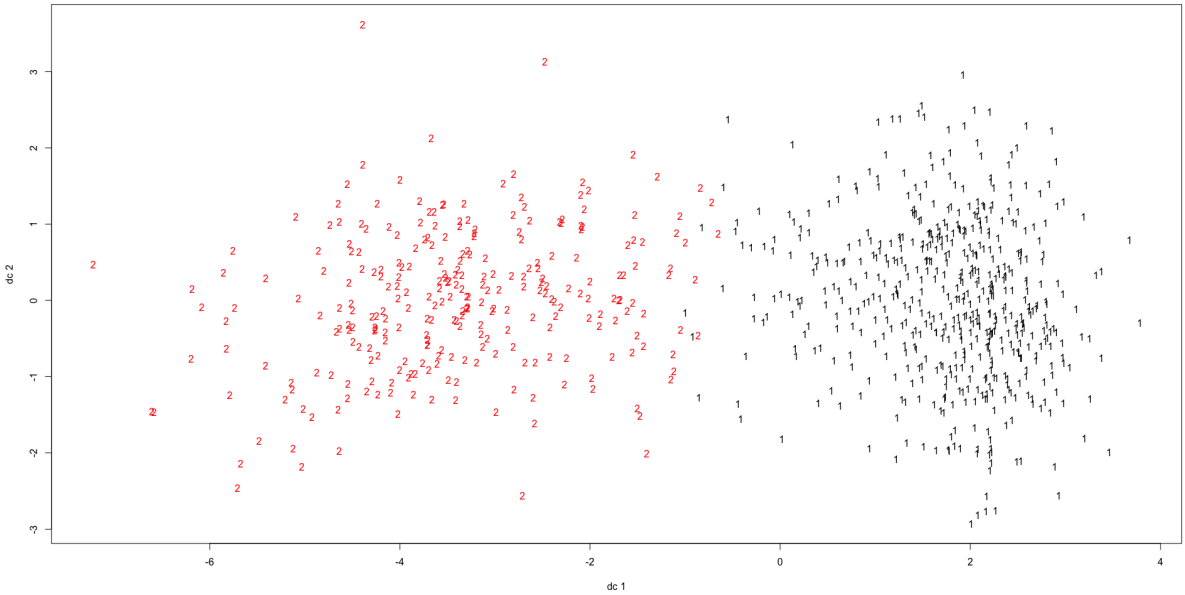
Code:

```
fit.km <- kmeans(data.train, 2)

fit.km

plotcluster(data.train, fit.km$cluster)

parcoord(data.train, fit.km$cluster)

confuseTable.km <- table(vehicles$Class, fit.km$cluster)
confuseTable.km
```

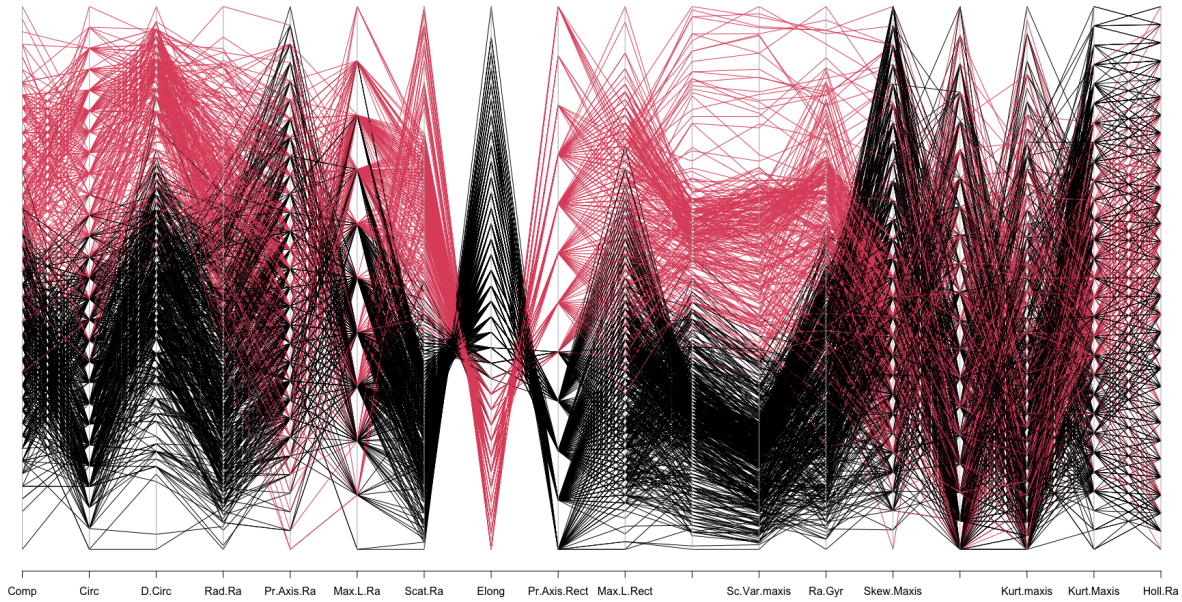
Output:

```
> fit.km
K-means clustering with 2 clusters of sizes 536, 278

Cluster means:
       Comp       Circ     D.Circ     Rad.Ra Pr.Axis.Ra   Max.L.Ra    Scat.Ra      Elong Pr.Axis.Rect
Max.L.Rect Sc.Var.Maxis Sc.Var.maxis     Ra.Gyr  Skew.Maxis
1 -0.5598279 -0.5739833 -0.5976565 -0.5401398 -0.1332193 -0.3350492 -0.6357373
0.6061737   -0.6377141 -0.5309226   -0.6163387   -0.6389155 -0.5260507  0.04452436
2  1.0793803  1.1066728  1.1523161  1.0414205  0.2568545  0.6459941  1.2257381 -
1.1687378    1.2295496  1.0236493    1.1883365    1.2318659  1.0142560 -0.08584552
  Skew.maxis Kurt.maxis  Kurt.Maxis    Holl.Ra
1 -0.0559197 -0.1190294 -0.02651187 -0.1038261
2  0.1078164  0.2294956  0.05111642  0.2001828

Clustering vector:
  [1] 1 1 2 1 2 1 1 1 2 1 1 1 1 2 2 1 1 2 2 1 1 1 1 2 1 1 2 2 1 1 1 2 1 1 2 1 2 1 1 1 1 1 1 1 1 2
1 2 1 2 1 2 1 2 1 1 1 2 1 1 2 1 2 2 2 1 1 1 2 1 1 2 1 2 1 1 1 2
  [82] 1 1 1 1 2 1 2 1 1 2 1 1 2 1 1 1 1 2 2 2 1 1 2 1 1 1 1 1 2 2 1 1 1 2 2 1 1 1 1 1 1 1 1 2 2 1 1 2 1 1 1 1
1 1 2 1 1 2 1 1 1 1 2 2 1 2 1 2 1 1 1 1 1 2 1 1 2 2 1 2 1 1 2 2 1
[163] 2 1 1 1 1 1 1 2 1 1 1 2 1 1 1 2 1 1 1 2 1 1 2 1 1 2 1 1 1 1 1 2 2 1 1 1 1 1 2 1 1 1 2 1 1 1 2 1 1 2 1
2 1 1 1 2 1 2 1 1 1 2 1 1 1 1 2 1 1 1 1 2 1 1 1 1 2 1 1 1 1 2 1 1 2 1 1 1 2
[244] 1 1 2 2 1 1 1 1 2 1 1 1 1 1 1 2 1 1 2 1 1 1 2 1 2 1 1 2 1 1 1 1 2 1 1 1 1 2 1 2 1 1 1 2 1 1 1 1
1 2 2 2 2 2 1 1 2 1 1 1 2 1 2 2 2 1 2 1 1 2 1 1 1 1 2 2 1 2 2 1 2
[325] 1 1 1 1 1 2 2 2 1 1 1 2 1 1 1 2 1 1 2 1 2 2 2 1 1 1 2 1 1 1 1 1 1 1 1 1 2 2 1 1 2 1 1 2 1 1 2 1 1 2 1
2 1 1 1 2 1 2 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1
[406] 1 1 2 1 2 1 2 2 1 1 2 1 1 1 2 2 2 1 1 2 2 1 2 2 2 1 1 1 1 1 2 1 1 1 2 1 1 2 1 1 2 1 1 2 2 1 1 2
2 2 1 2 2 1 2 1 2 2 1 1 1 1 2 1 1 2 2 1 1 2 2 1 2 1 2 2 2 1 1 2 2
[487] 1 1 2 1 1 2 1 1 1 2 1 1 1 1 1 2 1 2 2 1 1 1 2 2 1 1 1 2 1 2 2 1 1 1 1 1 1 1 1 2 1 1 1 1 2 2 2 1
1 2 2 1 2 1 1 2 2 1 1 1 2 1 1 1 2 2 2 2 1 1 1 1 2 2 2 1 2 1 1 2 1
[568] 1 1 1 1 2 1 1 1 1 1 1 2 1 1 2 1 1 1 1 2 1 1 1 1 1 1 1 1 2 2 1 1 1 2 1 1 2 1 1 2 1 2 1 1 1 1 2 2
2 1 2 1 1 1 1 1 2 1 2 1 1 1 2 1 1 1 2 1 2 1 1 1 1 2 2 1 1 2 1 2 1
[649] 1 2 1 2 1 1 1 1 1 2 1 1 1 2 1 2 1 1 2 1 2 1 1 1 1 1 2 2 1 1 2 2 1 1 2 2 2 2 1 2 1 1 2 2 1 2 1 2 1
2 1 1 2 1 1 2 2 2 1 2 1 1 2 2 2 1 2 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1
[730] 2 1 1 1 2 1 2 2 1 1 1 2 1 1 2 2 1 1 1 2 2 2 1 2 1 2 2 1 1 2 1 2 1 1 1 2 2 1 1 2 1 1 1 1 2 2 1 1
2 1 1 2 1 1 1 1 1 2 2 2 1 2 1 2 2 1 2 2 1 1 1 1 2 1 1 1 1 1 1 1 1
[811] 1 2 1 1

Within cluster sum of squares by cluster:
[1] 5737.082 2654.387
 (between_SS / total_SS =  42.7 %)

Available components:

[1] "cluster"     "centers"     "totss"       "withinss"    "tot.withinss" "betweenss"    "size"
"iter"        "ifault"
```

> plotcluster(data.train, fit.km$cluster)



> parcoord(data.train, fit.km$cluster)



> confuseTable.km

	1	2
bus	155	54
opel	92	116
saab	100	108
van	189	0

<u>3 Clusters:</u>

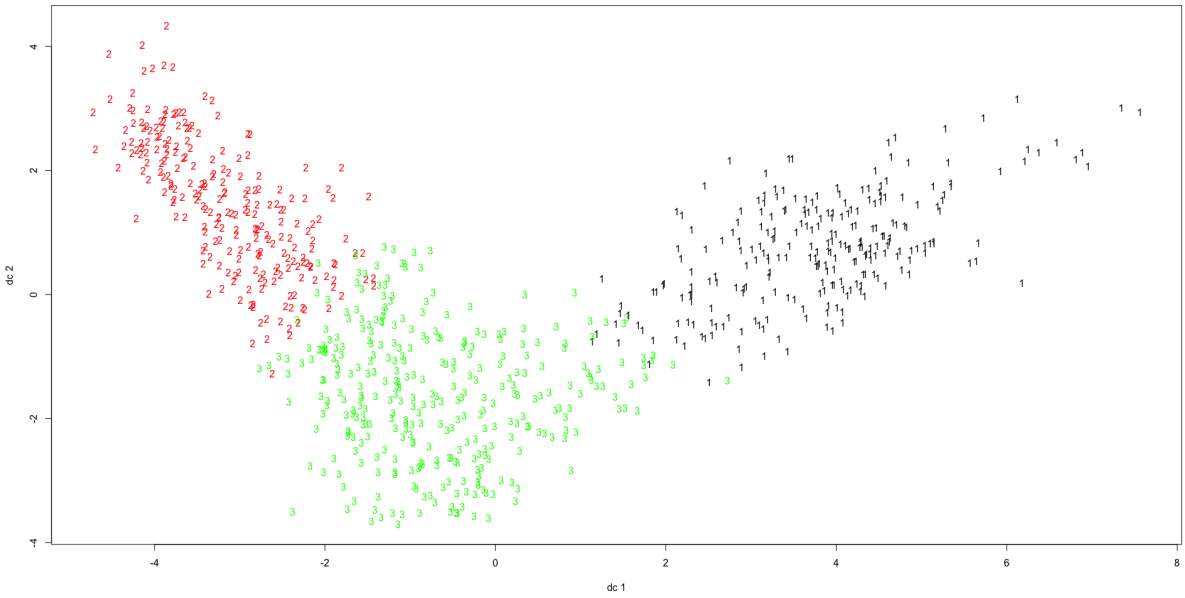
Output:

```
> fit.km
K-means clustering with 3 clusters of sizes 254, 233, 327

Cluster means:
       Comp      Circ     D.Circ       Rad.Ra Pr.Axis.Ra   Max.L.Ra    Scat.Ra      Elong Pr.Axis.Rect
Max.L.Rect Sc.Var.Maxis Sc.Var.maxis     Ra.Gyr Skew.Maxis
1  1.1619572  1.1879128  1.2225756  1.050718567  0.2138098  0.7067637  1.3091738 -
1.2232515    1.3137764  1.1093979    1.2629923    1.3202566  1.0951042 -0.0257191
2 -0.9430706 -0.5488373 -0.9193176 -1.146726125 -0.7492660 -0.5337781 -0.7946580
0.8907934   -0.7608432 -0.5073911   -0.8126318   -0.7971448 -0.4172906  0.9885816
3 -0.2305862 -0.5316537 -0.2945969  0.000931716  0.3678022 -0.1686474 -0.4506875
0.3154466   -0.4783570 -0.5001986   -0.4020087   -0.4575242 -0.5532959 -0.6844246
   Skew.maxis  Kurt.maxis  Kurt.Maxis    Holl.Ra
1  0.13978075  0.27448001 -0.02710884  0.1535188
2 -0.10227776 -0.26106869 -1.06074174 -1.1245542
3 -0.03569905 -0.02718324  0.77687606  0.6820409

Clustering vector:
  [1] 3 3 1 3 1 3 3 3 3 3 3 3 3 1 2 3 1 1 2 2 3 3 1 3 2 1 1 2 3 3 3 1 3 3 2 1 2 1 2 2 3 2 2 2 3 2 3 1
3 1 3 3 2 1 2 1 2 2 2 3 2 2 1 3 1 1 1 3 2 3 1 3 2 1 2 1 3 2 3 1
 [82] 3 2 3 2 1 3 1 3 2 1 2 2 1 2 3 3 2 1 1 1 2 2 3 3 3 2 2 3 1 1 2 3 2 2 3 3 3 2 3 1 1 3 2 1 2 3 2 3
3 2 1 2 3 1 3 3 3 3 1 3 3 1 3 1 3 2 3 3 2 1 3 3 1 1 3 1 2 2 1 1 3
[163] 1 3 3 3 3 3 2 1 2 3 2 1 3 3 3 1 3 3 3 1 3 2 1 2 2 2 3 3 1 1 3 3 3 2 2 1 3 3 3 1 2 3 2 1 2 3 1 2
1 2 2 3 1 3 1 2 2 2 1 3 2 3 2 1 2 3 3 2 1 2 2 3 3 1 2 2 1 2 3 3 1
[244] 3 3 1 1 2 3 3 3 1 2 2 3 3 2 2 3 3 3 1 3 2 2 1 3 3 2 2 1 2 3 3 2 1 2 2 3 3 1 3 1 2 3 3 1 3 3 3 2
3 1 1 1 1 1 2 3 1 2 2 2 3 2 1 1 1 2 1 3 2 1 2 3 3 3 1 1 2 1 1 2 1
[325] 3 3 3 2 2 1 1 1 3 3 3 1 2 3 2 3 3 3 1 3 1 1 1 3 3 2 1 3 2 2 3 3 3 3 3 2 1 1 2 2 1 2 2 1 3 3 3 3
1 2 3 3 3 3 1 3 3 3 3 1 3 1 3 2 2 3 3 3 2 2 3 2 1 3 3 2 3 2 1 3 2
[406] 3 3 1 3 1 3 1 1 2 2 1 3 2 2 3 1 1 2 2 1 1 2 1 1 1 3 3 3 3 3 1 2 2 3 1 3 3 1 3 2 1 2 2 1 1 3 2 1
1 1 2 1 1 3 3 2 1 1 3 3 2 2 1 3 2 1 1 3 2 1 1 3 1 2 1 1 1 2 2 1 1
[487] 3 3 1 2 3 1 3 2 2 1 2 3 3 2 3 1 3 1 1 3 2 3 1 1 2 2 3 1 3 1 1 3 3 3 3 2 2 3 3 1 2 2 3 2 1 3 1 2
2 1 1 3 1 3 3 3 1 3 2 3 1 3 3 2 1 1 1 1 3 2 2 2 1 1 1 3 1 2 3 1 2
[568] 2 2 3 2 3 3 3 3 3 3 3 1 3 3 1 3 3 3 2 1 2 2 3 2 3 3 2 2 1 1 2 3 2 1 3 3 1 3 2 1 2 1 2 2 3 2 3 1
1 3 1 3 3 2 3 2 1 3 1 2 3 3 3 2 2 3 1 3 1 2 3 3 3 3 1 3 2 1 3 1 3
[649] 3 1 2 1 2 3 3 3 2 1 3 2 3 1 2 1 3 3 1 2 3 2 3 3 2 3 1 1 3 3 1 1 3 2 3 1 1 1 1 3 1 3 3 1 1 3 1 3
1 3 2 1 3 2 1 1 1 3 1 2 2 1 1 1 3 1 3 3 1 3 2 3 2 3 1 3 2 3 3 3 2
[730] 1 2 2 2 1 2 1 1 2 3 3 1 3 2 1 1 2 3 3 1 1 1 2 1 3 1 1 2 2 1 2 1 3 2 3 1 1 3 2 1 3 3 2 3 3 1 2 3
1 2 2 1 2 3 2 2 3 1 1 3 2 1 3 1 1 2 3 1 2 2 3 3 1 2 2 2 3 3 3 3 3
[811] 3 1 3 2

Within cluster sum of squares by cluster:
[1] 2284.200 1618.092 2660.204
 (between_SS / total_SS =  55.2 %)
```

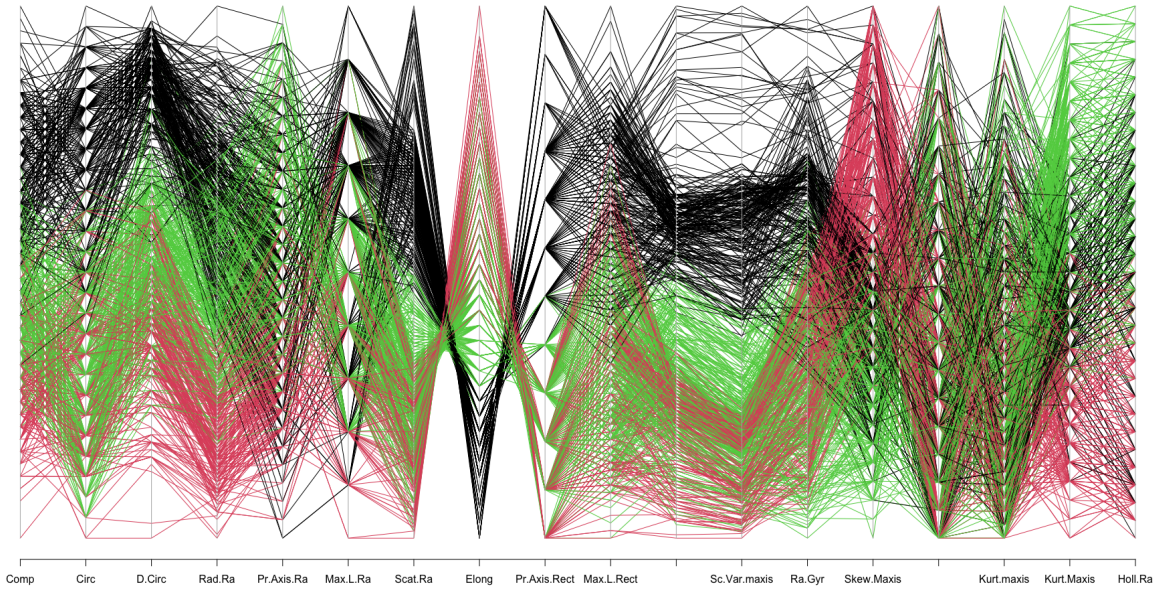
Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss" "betweenss" "size" "iter" "ifault"

> plotcluster(data.train, fit.km$cluster)



> parcoord(data.train, fit.km$cluster)



> confuseTable.km

	1	2	3
bus	47	80	82
opel	110	35	63
saab	97	38	73
van	0	80	109

<u>4 Clusters:</u>

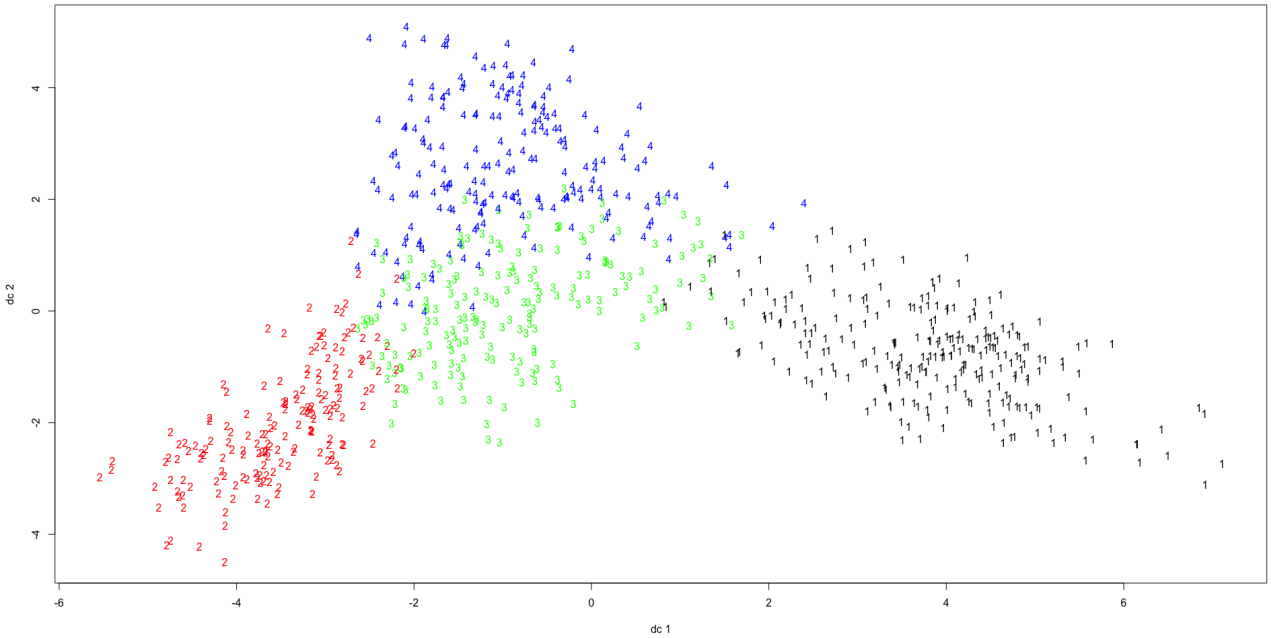
Output:

> fit.km
K-means clustering with 4 clusters of sizes 252, 171, 186, 205

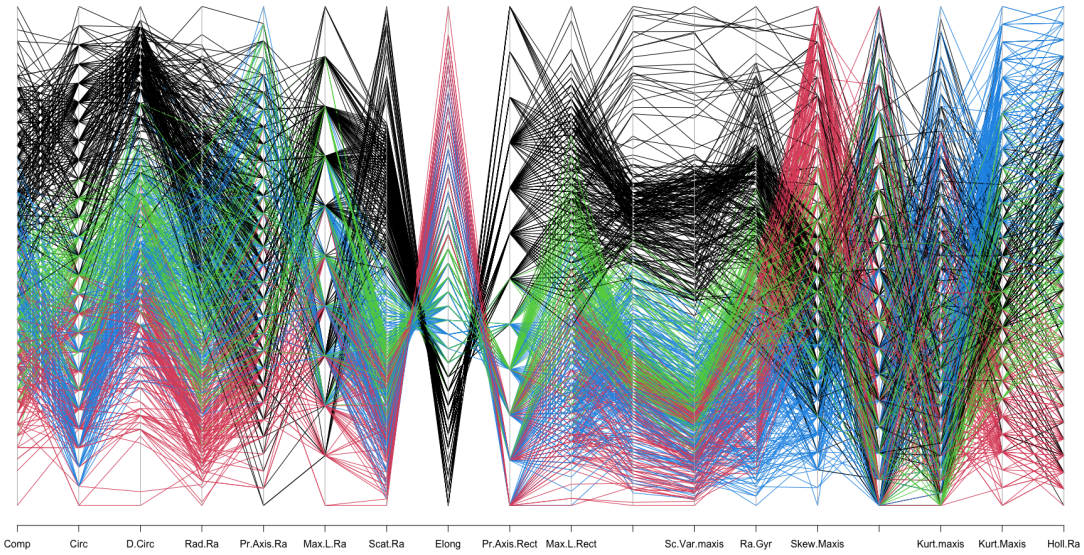
Cluster means:

	Comp	Circ	D.Circ	Rad.Ra	Pr.Axis.Ra	Max.L.Ra	Scat.Ra	Elong	Pr.Axis.Rect	Max.L.Rect	Sc.Var.Maxis	Sc.Var.maxis	Ra.Gyr	Skew.Maxis
1	1.1703267	1.197796392	1.2275280	1.05051696	0.2073337	0.7128347	1.3143767	-1.2259126	1.3198460	1.12065656	1.2658739	1.3256528	1.10476570	-0.02778674
2	-1.0349954	-0.752637128	-1.1075391	-1.26830251	-0.8908388	-0.8204129	-0.8937088	1.0407070	-0.8555864	-0.77044290	-0.9105507	-0.8771187	-0.56278443	1.18768009

```
3 -0.4092418  0.007147697 -0.1462744 -0.21471245  0.2431166  0.1517948 -0.3341041
0.1994023   -0.3530323  0.06163186   -0.2943911   -0.3835297  0.04621098 -0.01475183
4 -0.2039958 -0.851088847 -0.4523943 -0.03860502  0.2676374 -0.3296467 -0.5670993
0.4579525   -0.5884483 -0.79084509   -0.5294600   -0.5499545 -0.93053688 -0.94315706
    Skew.maxis Kurt.maxis  Kurt.Maxis     Holl.Ra
1  0.144639638  0.2754261 -0.02992877  0.15709480
2 -0.047307620 -0.2119942 -1.18353747 -1.34986388
3 -0.005273194 -0.4241544 -0.15964000 -0.04202179
4 -0.133554984  0.2231041  1.16887804  0.97099945

Clustering vector:
  [1] 3 3 1 4 1 4 4 4 4 4 4 3 3 3 1 2 4 1 1 3 2 4 4 1 3 2 1 3 2 3 4 4 1 4 3 2 1 3 1 2 3 4 3 3 3 4 2 4 1
4 1 4 3 3 1 2 1 2 2 2 3 2 2 1 3 1 1 1 4 2 3 1 3 2 1 3 1 4 2 4 1
 [82] 3 2 4 2 1 3 1 3 3 1 3 3 1 2 4 4 2 1 1 1 2 2 3 4 3 2 3 3 1 1 2 3 2 3 4 3 3 3 3 1 1 4 2 1 3 4 2 3
4 2 1 2 4 1 3 4 4 4 1 4 4 1 3 1 4 2 4 3 2 1 4 4 1 1 4 1 2 2 1 1 4
[163] 1 3 4 4 3 4 2 1 2 4 2 1 4 4 3 1 4 3 3 1 4 2 1 2 2 2 3 4 1 1 4 3 3 2 3 1 4 4 4 1 2 3 2 1 3 4 1 3
1 2 2 4 1 4 1 2 2 2 1 4 3 4 2 1 3 4 4 2 1 3 2 4 4 1 2 2 1 3 4 3 1
[244] 4 4 1 1 2 4 4 4 1 2 2 4 4 2 2 3 4 4 1 4 3 2 1 4 3 3 2 1 3 3 3 3 1 2 4 3 4 1 4 1 2 4 4 1 3 4 4 2
3 1 1 1 1 1 3 3 1 2 2 2 4 2 1 1 1 2 1 4 2 1 3 4 4 4 1 1 2 1 1 3 1
[325] 4 3 4 2 2 1 1 1 4 4 4 1 2 3 3 3 3 3 1 3 1 1 1 3 3 3 1 3 2 2 3 4 4 3 4 2 1 1 2 2 1 2 2 1 3 3 3 3
1 2 4 3 3 3 1 3 4 4 4 1 4 1 4 2 2 3 4 3 2 2 4 3 1 4 4 3 4 2 1 4 2
[406] 4 4 1 4 1 4 1 1 2 2 1 4 2 2 3 1 1 3 4 1 1 3 1 1 1 4 3 4 4 4 1 2 2 3 1 4 4 1 3 2 1 2 2 1 1 4 3 1
1 1 2 1 1 3 3 2 1 1 3 4 3 3 1 3 2 1 1 4 2 1 1 3 1 3 1 1 1 2 2 1 1
[487] 4 3 1 2 4 1 4 2 2 1 2 4 3 2 4 1 3 1 1 4 2 3 1 1 2 2 3 1 4 1 1 3 4 4 4 3 2 4 4 1 2 2 3 2 1 3 1 2
2 1 1 3 1 4 4 4 1 3 2 4 1 3 3 3 1 1 1 1 4 2 2 2 1 1 1 4 1 2 4 1 2
[568] 2 2 4 2 3 3 4 3 4 4 4 1 4 4 1 4 4 4 2 1 3 2 4 2 4 4 2 2 1 1 3 4 4 1 3 3 1 4 2 1 2 1 2 2 4 2 3 1
1 3 1 4 3 3 4 2 1 4 1 2 4 4 4 3 3 4 1 4 1 2 3 4 3 4 1 3 2 1 4 1 4
[649] 4 1 3 1 2 3 3 4 3 1 4 2 4 1 2 1 4 3 1 3 4 2 3 4 3 3 1 1 3 4 1 1 4 3 4 1 1 1 1 4 1 4 3 1 1 4 1 3
1 3 4 1 4 2 1 1 1 3 1 2 2 1 1 1 3 1 4 4 1 3 2 3 2 3 1 4 2 4 4 4 2
[730] 1 2 2 3 1 3 1 1 2 4 4 1 4 2 1 1 2 4 4 1 1 1 2 1 4 3 1 2 2 1 2 1 3 2 3 1 1 3 2 1 4 3 2 3 3 1 2 3
1 3 2 1 2 4 3 2 3 1 1 3 2 1 4 1 1 2 3 1 2 3 4 4 1 3 2 3 3 3 4 4 3
[811] 3 1 4 2

Within cluster sum of squares by cluster:
[1] 2255.6906  997.2905 1167.3448 1559.0218
 (between_SS / total_SS =  59.1 %)

Available components:

[1] "cluster"     "centers"     "totss"       "withinss"    "tot.withinss" "betweenss"    "size"
"iter"        "ifault"

> plotcluster(data.train, fit.km$cluster)
```



> parcoord(data.train, fit.km$cluster)



> confuseTable.km

	1	2	3	4
bus	46	62	48	53
opel	109	32	27	40
saab	97	35	26	50
van	0	42	85	62

Evaluation of the produced outputs against 19th column

2 Clusters:

	1	2
bus	155	54
opel	92	116
saab	100	108
van	189	0

Recall/Sensitivity per class:

Bus: 155/209 = 74.1%

Opel: 116/208 = 55.8%

Saab: 108/208 = 51.9%

Van: 189/189 = 100%

Precision per cluster:

Cluster 1: (155 + 189)/536 = 64.2%

Cluster 2: (116 + 108)/278 = 80.6%

Accuracy of Clustering:

(155 + 116 + 108 + 189)/814 = 69.8%

3 Clusters:

	1	2	3
bus	47	80	82
opel	110	35	63
saab	97	38	73
van	0	80	109

Recall/Sensitivity per class:

Bus: 80/209 = 38.3%

Opel: 110/208 = 52.9%

Saab: 97/208 = 46.6%

Van: 109/189 = 57.7%

Precision per cluster:

Cluster 1: (110 + 97)/254 = 81.5%

Cluster 2: 80/233 = 34.3%

Cluster 3: 109/327 = 33.3%

Accuracy of Clustering:

(80 + 110 + 97 + 109)/814 = 48.6%

4 Clusters:

	1	2	3	4
bus	46	62	48	53
opel	109	32	27	40
saab	97	35	26	50
van	0	42	85	62

Recall/Sensitivity per class:

Bus: 53/209 = 25.4%

Opel: 109/208 = 52.4%

Saab: 35/208 = 16.8%

Van: 85/189 = 45%

Precision per cluster:

Cluster 1: 109/252 = 43.3%

Cluster 2: 35/171 = 20.5%

Cluster 3: 85/186 = 45.7%

Cluster 4: 53/205 = 25.9%

Accuracy of Clustering:

(53 + 109 + 35 + 85)/ 814 = 34.6%

Define final winner cluster case and provide brief explanation of evaluation indices

The winner cluster case is with 2 clusters. This is largely due to the highest accuracy (69.8%) compared to the others (48.6%, 34.6%). Accuracy is the proportion of instances that are correctly classified into the classes Bus, Opel, Saab and Van.

Sensitivity/Recall is a measure of ability to detect positivity. In this context it means for example for the Van class, how many times it predicts Van. So, for k = 2 it had by far the highest recall with all above 50%, and Van actually being 100%. For k = 3 and k = 4, the sensitivity for Bus dropped dramatically.

Precision in this case is when the algorithm predicts that an instance is an Opel, sensitivity shows how many are correct i.e., how many are actually Opel out of the Opel predictions.

Illustrate the coordinates of each centre for each clustering group

2 Clusters:

```
> fit.km$centers
      Comp      Circ    D.Circ    Rad.Ra Pr.Axis.Ra  Max.L.Ra   Scat.Ra     Elong Pr.Axis.Rect
Max.L.Rect Sc.Var.Maxis Sc.Var.maxis    Ra.Gyr Skew.Maxis
1 0.6813370 0.7176730 0.8346276 0.6818365  0.5388703 0.6437276 0.6482676 0.1671275
0.6125774  0.6397849    0.5962298    0.5791437 0.6188093  0.4461086
2 0.3747446 0.3194105 0.4559190 0.3299197  0.4632936 0.4431776 0.2396012 0.5593057
0.1770603  0.3180507    0.2482669    0.1770948 0.3038030  0.4758344
  Skew.maxis Kurt.maxis Kurt.Maxis   Holl.Ra
1  0.3537068  0.3666667  0.4754224 0.5388292
2  0.3117560  0.2886449  0.4578772 0.4638006
```

3 Clusters:

```
> fit.km$centers
      Comp      Circ    D.Circ    Rad.Ra Pr.Axis.Ra  Max.L.Ra   Scat.Ra      Elong Pr.Axis.Rect
Max.L.Rect Sc.Var.Maxis Sc.Var.maxis    Ra.Gyr Skew.Maxis
1 0.7807309 0.8296703 0.8516865 0.5857646  0.2992611 0.2714286 0.8669048 0.04387755
0.8636364  0.6918367    0.8573689    0.8403387 0.8158131  0.8482143
2 0.6630118 0.6920152 0.8253063 0.6881862  0.5657532 0.6817490 0.6142966 0.18837588
0.5734532  0.6233026    0.5595370    0.5401201 0.5860057  0.3942151
3 0.3716039 0.3171790 0.4510038 0.3237955  0.4608690 0.4393881 0.2356023 0.56421743
0.1733009  0.3161704    0.2447540    0.1735099 0.3025241  0.4810844
  Skew.maxis Kurt.maxis Kurt.Maxis   Holl.Ra
```

1 0.3327068 0.3767857 0.2806122 0.1119048
2 0.3602161 0.3665399 0.5051602 0.5925222
3 0.3086445 0.2863767 0.4529500 0.4579350

4 Clusters:

> fit.km$centers
      Comp   Circ   D.Circ   Rad.Ra Pr.Axis.Ra Max.L.Ra   Scat.Ra   Elong Pr.Axis.Rect
Max.L.Rect Sc.Var.Maxis Sc.Var.maxis   Ra.Gyr Skew.Maxis
1 0.2868217 0.3034188 0.3592172 0.1851259 0.3416928 0.3575758 0.1925589 0.6389610
0.1354454 0.2920635   0.2015407   0.1337267 0.3149101 0.7113997
2 0.4169727 0.2240216 0.4563840 0.3982374 0.5176447 0.4362573 0.2197271 0.5695906
0.1541733 0.2426065   0.2346214   0.1651077 0.1987569 0.2564745
3 0.4628342 0.4577642 0.5994938 0.4850928 0.5678376 0.5448598 0.3530218 0.4129506
0.2888700 0.4351802   0.3528333   0.2777733 0.4120085 0.4028705
4 0.7075405 0.7565768 0.8628547 0.6847448 0.5217197 0.6696970 0.6831457 0.1453309
0.6521055 0.6762523   0.6218149   0.6154632 0.6522639 0.4679963
  Skew.maxis Kurt.maxis Kurt.Maxis   Holl.Ra
1 0.3062201 0.2579545 0.2175325 0.1784512
2 0.2726993 0.3460526 0.7107352 0.7189084
3 0.3489916 0.2754673 0.5190254 0.5607477
4 0.3615858 0.3788961 0.4412492 0.5203463

Appendix

```
library(readxl)
library(NbClust)
library(tidyverse)
library(gridExtra)
library(ggplot2)
library(rlang)
library(fpc)
library(MASS)
library(flexclust)
library(caret)
vehicles <- read_excel("/Users/Adz/Documents/University/Year 2/2. Machine Learning and
Data Mining/Coursework/vehicles.xlsx")

set.seed(1)

normalise <- function(x)
{
  return ((x - min(x)) / (max(x) - min(x)))
}

# https://www.statology.org/remove-outliers-from-multiple-columns-in-r/
```

```
outliers <- function(x) {

  Q1 <- quantile(x, probs=.25)
  Q3 <- quantile(x, probs=.75)
  iqr = Q3-Q1

  upper_limit = Q3 + (iqr*1.5)
  lower_limit = Q1 - (iqr*1.5)

  x > upper_limit | x < lower_limit
}

remove_outliers <- function(df, cols = names(df)) {
  for (col in cols) {
    df <- df[!outliers(df[[col]]),]
  }
  df
}


vehicles <- remove_outliers(vehicles,
c('Comp','Circ','D.Circ','Rad.Ra','Pr.Axis.Ra','Max.L.Ra','Scat.Ra','Elong','Pr.Axis.Rect',

'Max.L.Rect','Sc.Var.Maxis','Sc.Var.maxis','Ra.Gyr','Skew.Maxis','Skew.maxis','Kurt.maxis','Ku
rt.Maxis','Holl.Ra'))


vehicles[c(2:19)] <- lapply(vehicles[c(2:19)], function(x) c(normalise(x)))


data.train <- vehicles[2:19]


euclidean <- NbClust(data.train,
                     distance="euclidean",
                     min.nc=2,
                     max.nc=15,
                     method="kmeans",
                     index="all")

manhattan <- NbClust(data.train,
                     distance="manhattan",
                     min.nc=2,
                     max.nc=15,
                     method="kmeans",
                     index="all")
```

```
table(euclidean$Best.n[1,])

barplot(table(euclidean$Best.n[1,]),
        xlab="Number of Clusters",
        ylab="Number of Criteria",
        main="Number of Clusters Chosen (Euclidean Distance)")

table(manhattan$Best.n[1,])

barplot(table(manhattan$Best.n[1,]),
        xlab="Number of Clusters",
        ylab="Number of Criteria",
        main="Number of Clusters Chosen (Manhattan Distance)")

wss <- 0
for (i in 1:15){
  wss[i] <-
    sum(kmeans(data.train, centers=i)$withinss)
}
plot(1:15,
     wss,
     type="b",
     xlab="Number of Clusters",
     ylab="Within groups sum of squares")

#2 Clusters
fit.km <- kmeans(data.train, 2)

fit.km

plotcluster(data.train, fit.km$cluster)

parcoord(data.train, fit.km$cluster)

confuseTable.km <- table(vehicles$Class, fit.km$cluster)
confuseTable.km

fit.km$centers

#3 Clusters
fit.km <- kmeans(data.train, 3)

fit.km

plotcluster(data.train, fit.km$cluster)
```

```
parcoord(data.train, fit.km$cluster)

confuseTable.km <- table(vehicles$Class, fit.km$cluster)
confuseTable.km

fit.km$centers

#4 Clusters
fit.km <- kmeans(data.train, 4)

fit.km

plotcluster(data.train, fit.km$cluster)

parcoord(data.train, fit.km$cluster)

confuseTable.km <- table(vehicles$Class, fit.km$cluster)
confuseTable.km

fit.km$centers
```