JUNE 4, 2021

PARTITIONING AND HIERARCHICAL CLUSTERING

ANWESH PRAHARAJ

Table of Contents

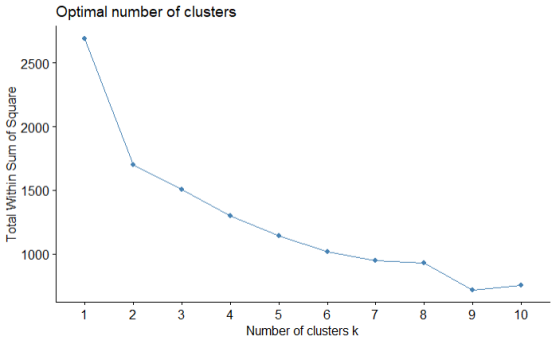
1. Partitioning Clustering

1.1. Using Kmean clustering

1.1.1. Find out optimal Clustering

used Elbow, silhouette method to find out the optimal cluster in dataset.

```{r}
set.seed(823)
factoextra::fviz_nbclust(
  x = scale_M,
  FUNcluster = kmeans,
  method = "wss"
)
```




```{r}
set.seed(823)
factoextra::fviz_nbclust(
  x = scale_M,
  FUNcluster = kmeans,
  method = "silhouette"
)
```

Left side graph is Elbow graph and right one is silhouette graph

In Elbow graph there is steep change of slope in k = 2. So, considering 2 as cluster. Also silhouette graph illustrate the same cluster as 2.

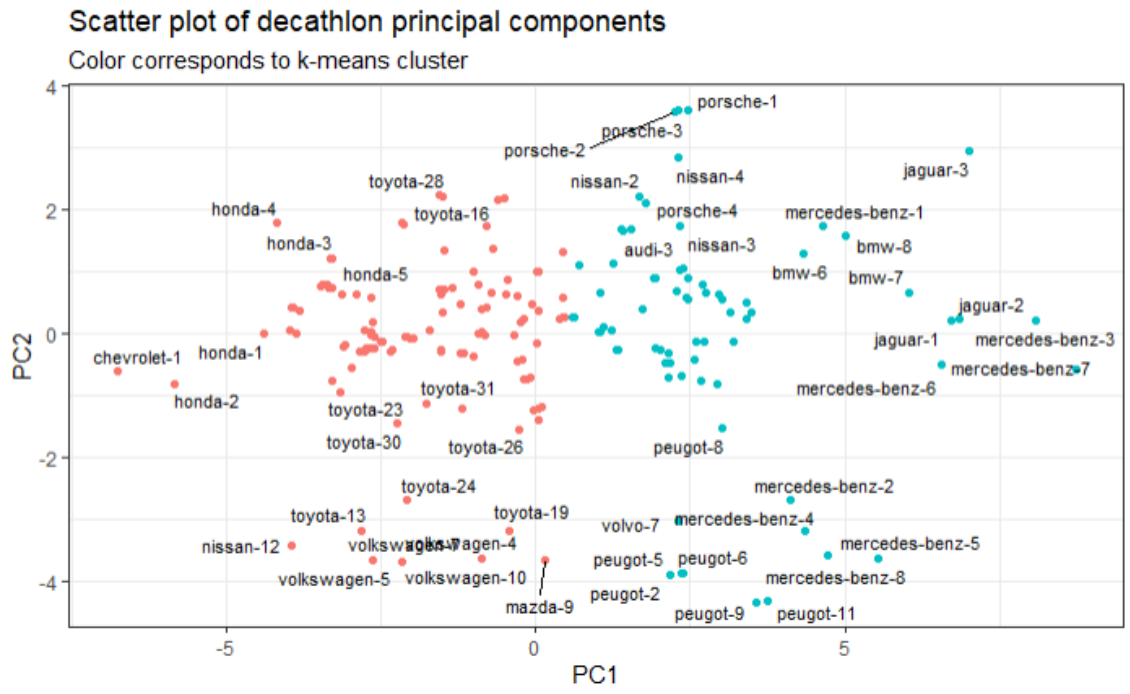
1.1.2. Kmean with 2 cluster

```{r}
kmeans_Car_2 <- kmeans(
  x = scale_M,
  centers = 2
)
kmeans_Car_2
```

Use PC1 and PC2 to plot scatter plot .

```{r}
prcomp_M_2 <- data.frame(
  prcomp(
    x = scale_M,
    center = FALSE,
    scale. = FALSE
  )$x[,1:2],
  Name = rownames(Dataset),
  Cluster = as.character(kmeans_Car_2$cluster),
  stringsAsFactors = FALSE
)
```

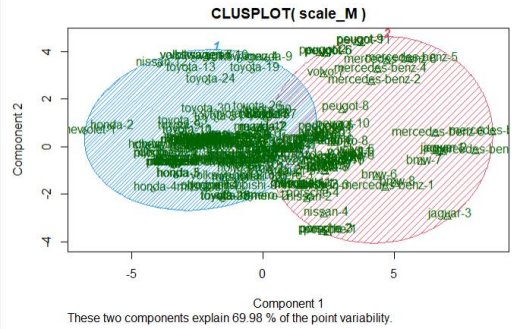
```{r}
require(ggplot2)
require(ggforce)
ggplot(prcomp_M_2) +
  aes(x = PC1,y = PC2,color = Cluster,fill = Cluster,label = Name,group = Cluster) +
  geom_point() +
  ggrepel::geom_text_repel(color = "black",size = 3) +
  ggtitle("Scatter plot of decathlon principal components","Color corresponds to k-means cluster") +
  theme_bw() +
  theme(legend.position = "none")
```



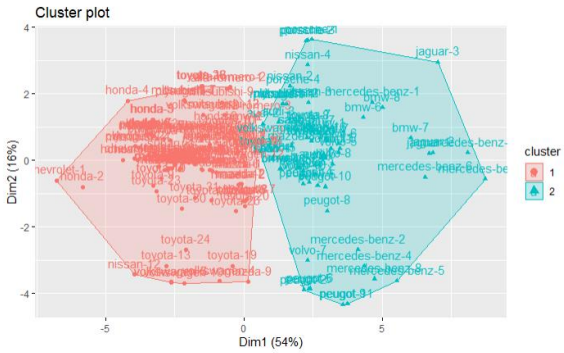
Here we have two groups in red and teal.

To know whether there are any overlap or not we will use below graphs (clustplot and fviz_cluster)

```{r}
cluster::clusplot(
  scale_M,
  kmeans_Car_2$cluster,
  color=TRUE,
  shade=TRUE,
  labels=2,
  lines=0
)
```




```{r}
factoextra::fviz_cluster(
  object = kmeans_Car_2,
  data = scale_M
)
```



Fviz_cluster plot shows very less overlap between two cluster group.

Calculate max_diameter and min_separation

```{r}
Kmean_stat_2 <- fpc::cluster.stats(
  d = dist(scale_M),
  clustering = kmeans_Car_2$cluster,
  G2 = TRUE,
  G3 = TRUE)

Kmean_stat_2$max.diameter
Kmean_stat_2$min.separation
```

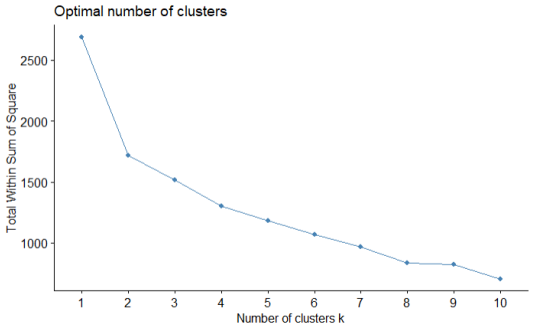
```
[1] 9.561109
[1] 0.285369
```

1.2. Using Clara clustering

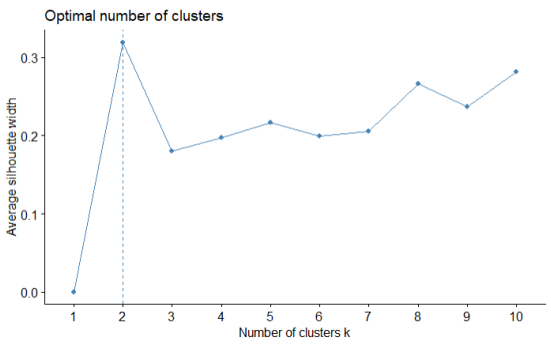
1.2.1. Find out optimal Clustering

Here also used Elbow , silhouette method to find the cluster strength.

```r
set.seed(823)
factoextra::fviz_nbclust(
  x = scale_M,
  FUNcluster = cluster::clara,
  method = "wss"
)
```




```r
set.seed(823)
factoextra::fviz_nbclust(
  x = scale_M,
  FUNcluster = cluster::clara,
  method = "silhouette"
)
```



Both the method plot show cluster 2 is good for Clara clustering.

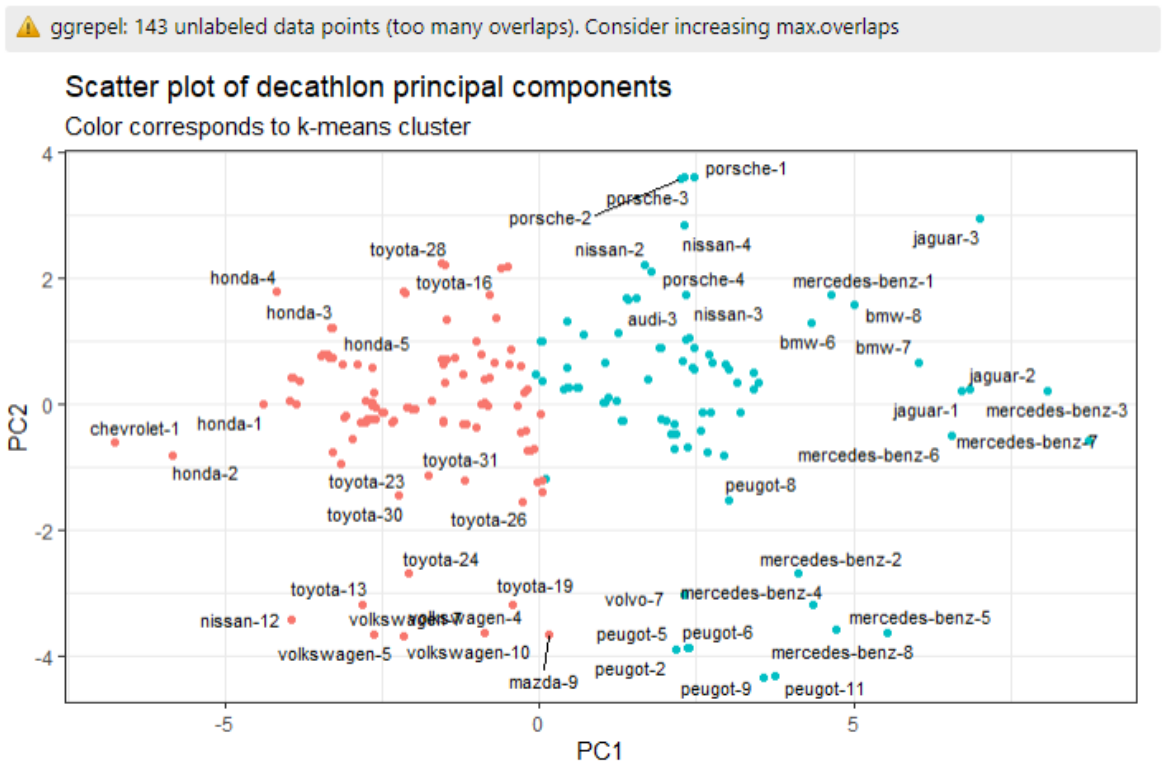
1.2.2 Clara with 2 cluster

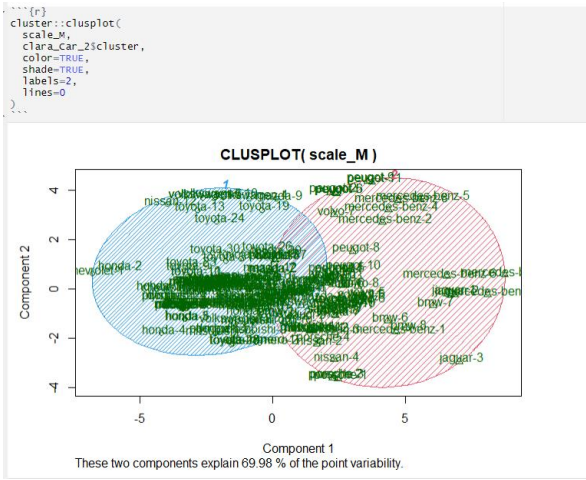
```r
clara_car_2 <- cluster::clara(
  x = scale_M,
  k = 2
)
clara_car_2
```

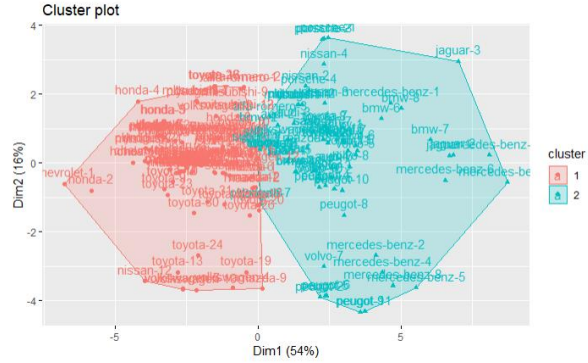
```
Call:    cluster::clara(x = scale_M, k = 2)
Medoids:
          wheel_base     length      width     height curb_weight engine_size       bore     stroke compression_ratio horsepower  peak_rpm   city_mpg highway_mpg       price
nissan-5  -0.719041 -0.6993116 -0.9794119 -0.2379970  -1.0129253  -0.7483533 -0.6631130 0.13042856        -0.1869588 -0.9083711 0.2139114  0.8881853   0.9113272 -0.67820122
nissan-8   0.239933  0.8232959  0.2835714  0.5136408   0.9464443   1.2713410  0.3648453 0.06702122        -0.2875247  1.2781348 0.2139114 -0.9903874  -0.8490012  0.02645221
Objective function:      2.856906
Clustering vector:       Named int [1:193] 1 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 ...
 - attr(*, "names")= chr [1:193] "alfa-romero-1" "alfa-romero-2" "alfa-romero-3" "audi-1" "audi-2" "audi-3" "audi-4" ...
Cluster sizes:           108 85
Best sample:
  [1] alfa-romero-2   audi-3          audi-5          bmw-2           bmw-3           bmw-5           bmw-6           dodge-6         honda-1         honda-5         honda-6
 [12] honda-13        mercedes-benz-8 mitsubishi-1    mitsubishi-2    mitsubishi-3    mitsubishi-11   nissan-3        nissan-5        nissan-8        nissan-13       nissan-18
 [23] peugot-3        peugot-5        peugot-7        plymouth-3      plymouth-4      plymouth-6      plymouth-7      saab-3          saab-4          subaru-7        subaru-9
 [34] toyota-2        toyota-5        toyota-8        toyota-9        toyota-23       toyota-25       toyota-26       toyota-29       volkswagen-3    volkswagen-6    volkswagen-10

Available components:
 [1] "sample"     "medoids"    "i.med"      "clustering" "objective"  "clusinfo"   "diss"       "call"       "silinfo"    "data"
```

```r
prcomp_M_Clara_2 <- data.frame(
  prcomp(
    x = scale_M,
    center = FALSE,
    scale. = FALSE
  )$x[,1:2],
  Name = rownames(Dataset),
  Cluster = as.character(clara_car_2$cluster),
  stringsAsFactors = FALSE
)
```

```r
require(ggplot2)
require(ggforce)
ggplot(prcomp_M_Clara_2) +
  aes(x = PC1,y = PC2,color = Cluster,fill = Cluster,label = Name,group = Cluster) +
  geom_point() +
  ggrepel::geom_text_repel(color = "black",size = 3) +
  ggtitle("Scatter plot of decathlon principal components","Color corresponds to k-means cluster") +
  theme_bw() +
  theme(legend.position = "none")
```







In above graph there are overlap of observation between two groups. Which means some of the observation in group 1 may be related to group 2 and vice versa.

Below is the max diameter and min separation for clara statistics for number of cluster = 2

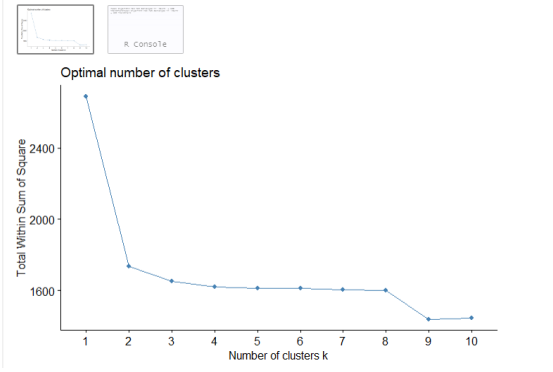
```r
clara_stat_2 <- fpc::cluster.stats(
  d = dist(scale_M),
  clustering = clara_car_2$cluster,
  G2 = TRUE,
  G3 = TRUE)

clara_stat_2$max.diameter
clara_stat_2$min.separation
```

```
[1] 10.53744
[1] 0.9923048
```

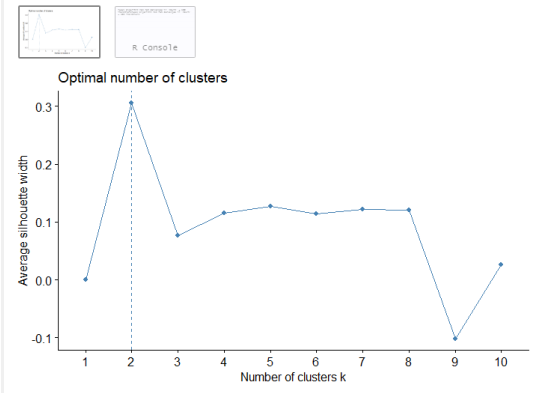
1.3. Using Fanny clustering

1.3.1. Find out optimal Clustering

```r
factoextra::fviz_nbclust(
  x = scale_M,
  FUNcluster = cluster::fanny,
  method = "wss"
)
```




```r
x = scale_M,
FUNcluster = cluster::fanny,
method = "silhouette"
)
```



Also used Elbow and Silhouette plot to identify the cluster. As per graph it is 2.

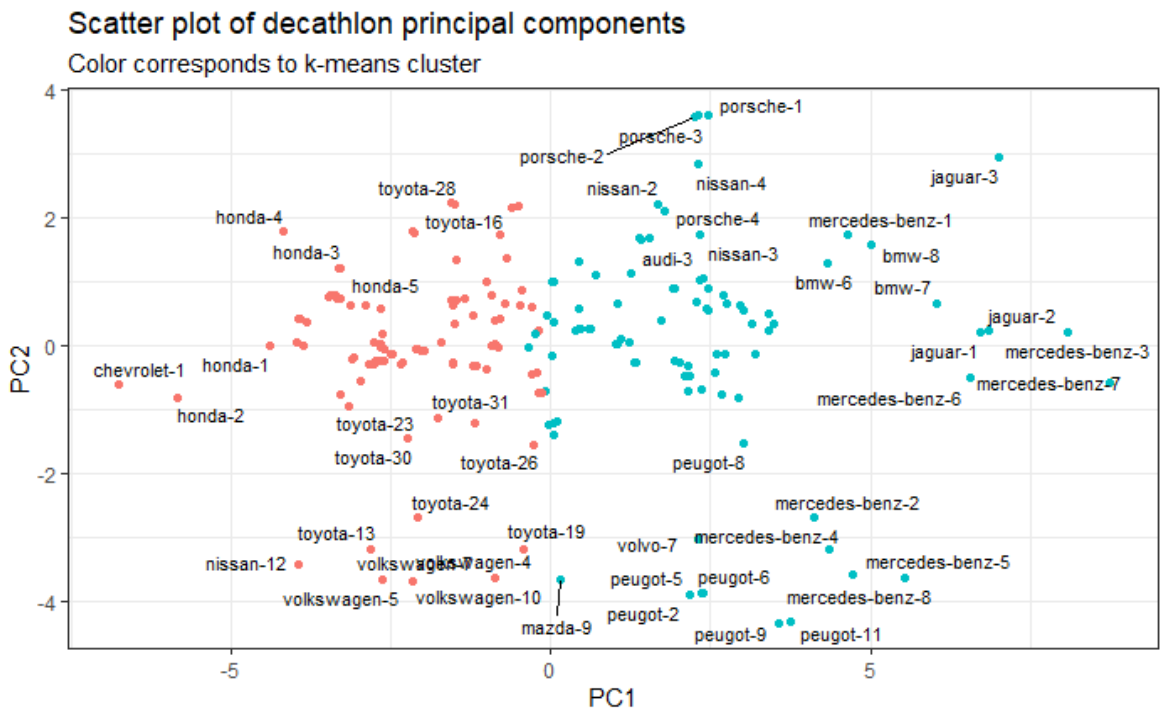
1.3.2. Fanny with 2 cluster

```r
fanny_Car_2 <- cluster::fanny(
  x = scale_M,
  k = 2
)
fanny_Car_2
```

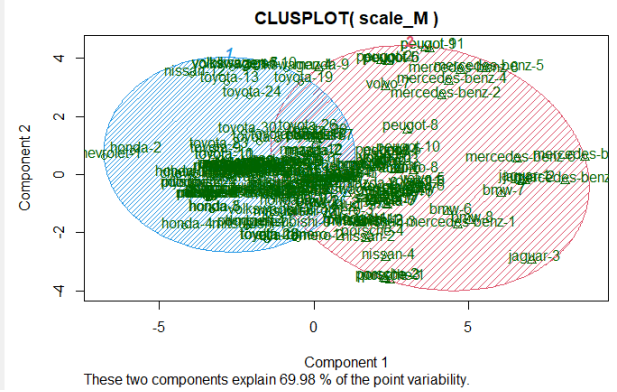
```r
prcomp_M_fanny_2 <- data.frame(
  prcomp(
    x = scale_M,
    center = FALSE,
    scale. = FALSE
  )$x[,1:2],
  Name = rownames(Dataset),
  Cluster = as.character(fanny_Car_2$cluster),
  stringsAsFactors = FALSE
)
```

```r
require(ggplot2)
require(ggforce)
ggplot(prcomp_M_fanny_2) +
  aes(x = PC1,y = PC2,color = Cluster,fill = Cluster,label = Name,group = Cluster) +
  geom_point() +
  ggrepel::geom_text_repel(color = "black",size = 3) +
  ggtitle("Scatter plot of decathlon principal components","Color corresponds to k-means cluster") +
  theme_bw() +
  theme(legend.position = "none")
```

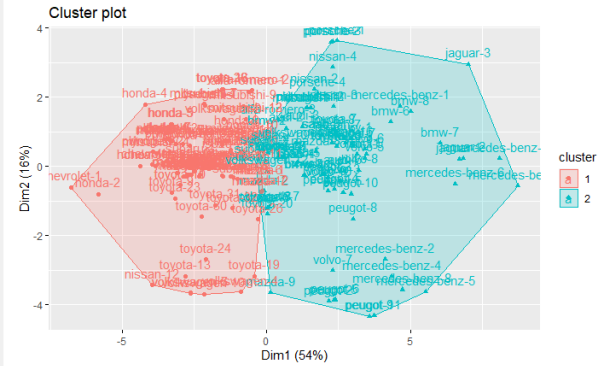
Scatter plot shows us the distributions of observations in different group.



```r
cluster::clusplot(
  scale_M,
  fanny_car_2$cluster,
  color=TRUE,
  shade=TRUE,
  labels=2,
  lines=0
)
```




```r
factoextra::fviz_cluster(
  object = fanny_car_2,
  data = scale_M
)
```



Here also there are overlaps of observation between the groups.

```r
fanny_stat_2 <- fpc::cluster.stats(
  d = dist(scale_M),
  clustering = fanny_car_2$cluster,
  G2 = TRUE,
  G3 = TRUE)

fanny_stat_2$max.diameter
fanny_stat_2$min.separation
```

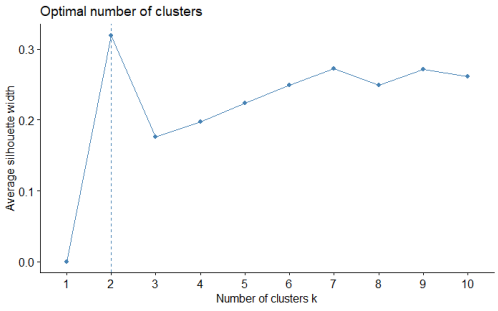
```
[1] 10.53744
[1] 0.1268614
```

1.4. Using Pam clustering

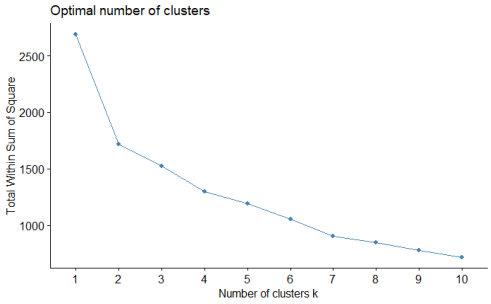
1.4.1. Find out optimal Clustering

Both Elbow and silhouette plot show the optimal cluster as 2.

```r
set.seed(823)
factoextra::fviz_nbclust(
  x = scale_M,
  FUNcluster = cluster::pam,
  method = "silhouette"
)
```




```r
set.seed(823)
factoextra::fviz_nbclust(
  x = scale_M,
  FUNcluster = cluster::pam,
  method = "wss"
)
```



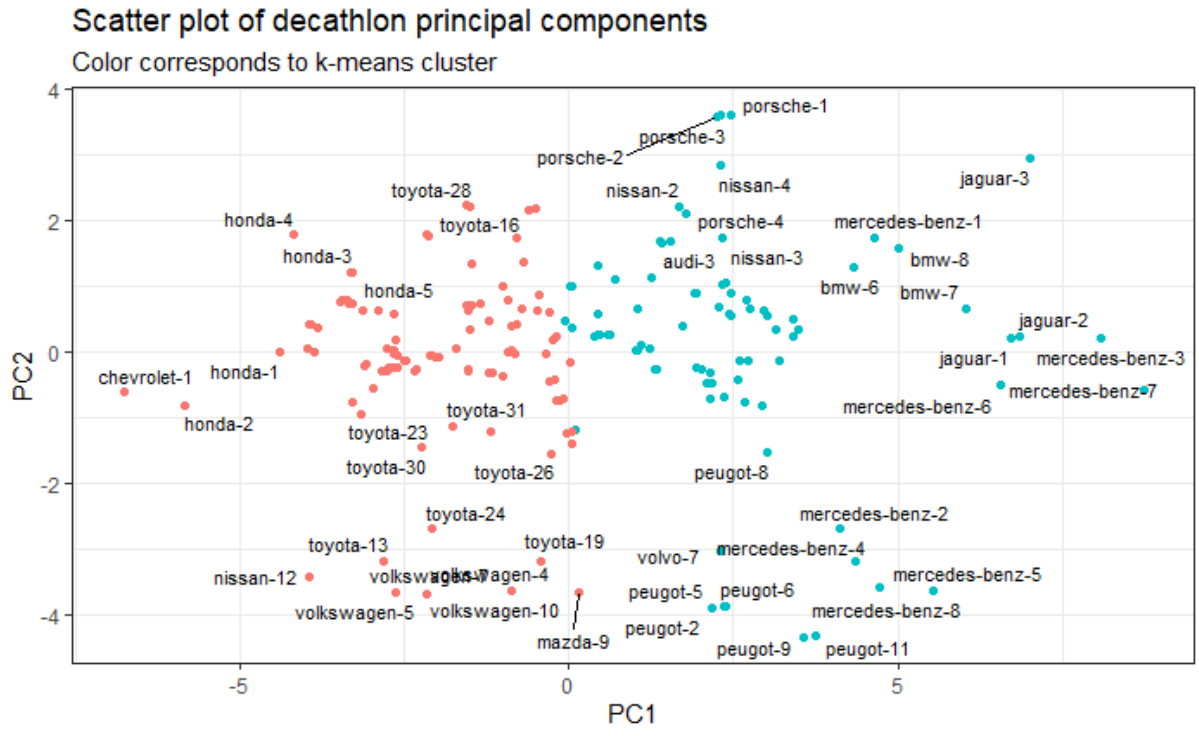
1.4.2. Pam with 2 cluster

```r
pam_Car_2 <- cluster::pam(
  x = scale_M,
  k = 2
)
pam_Car_2
```

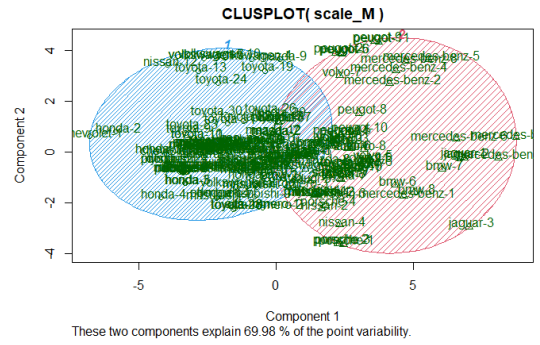
```
Medoids:
         ID wheel_base     length      width     height curb_weight engine_size       bore     stroke compression_ratio horsepower  peak_rpm   city_mpg highway_mpg       price
nissan-5 85  -0.719041 -0.6993116 -0.9794119 -0.2379970  -1.0129253  -0.7483533 -0.6631130 0.13042856        -0.1869588 -0.9083711 0.2139114  0.8881853   0.9113272 -0.67820122
nissan-8 88   0.239933  0.8232959  0.2835714  0.5136408   0.9464443   1.2713410  0.3648453 0.06702122        -0.2875247  1.2781348 0.2139114 -0.9903874  -0.8490012  0.02645221
Clustering vector:
  alfa-romero-1   alfa-romero-2   alfa-romero-3         audi-1         audi-2         audi-3         audi-4         audi-5         audi-6          bmw-1          bmw-2
              1               1               2              1              2              2              2              2              2              2              2
          bmw-3          bmw-4          bmw-5          bmw-6          bmw-7          bmw-8     chevrolet-1    chevrolet-2    chevrolet-3        dodge-1        dodge-2
              2              2              2              2              2              2              1              1              1              2              1
        dodge-3        dodge-4        dodge-5        dodge-6        dodge-7        dodge-8        honda-1        honda-2        honda-3        honda-4        honda-5
              1              1              1              1              1              2              1              1              1              1              1
        honda-6        honda-7        honda-8        honda-9       honda-10       honda-11       honda-12       honda-13        isuzu-1        isuzu-2       jaguar-1
              1              1              1              1              1              1              1              1              1              1              2
       jaguar-2       jaguar-3       mazda-1        mazda-2        mazda-3        mazda-4        mazda-5        mazda-6        mazda-7        mazda-8        mazda-9
              2              2              1              1              1              1              1              1              1              2              1
       mazda-10       mazda-11       mazda-12 mercedes-benz-1 mercedes-benz-2 mercedes-benz-3 mercedes-benz-4 mercedes-benz-5 mercedes-benz-6 mercedes-benz-7 mercedes-benz-8
              1              1              1              2              2              2              2              2              2              2              2
     mercury-1    mitsubishi-1    mitsubishi-2    mitsubishi-3    mitsubishi-4    mitsubishi-5    mitsubishi-6    mitsubishi-7    mitsubishi-8    mitsubishi-9   mitsubishi-10
              2              2              2              2              1              1              1              1              1              1              1
  mitsubishi-11   mitsubishi-12   mitsubishi-13       nissan-1       nissan-2       nissan-3       nissan-4       nissan-5       nissan-6       nissan-7       nissan-8
              1              1              1              1              2              2              2              1              1              2              2
       nissan-9      nissan-10      nissan-11      nissan-12      nissan-13      nissan-14      nissan-15      nissan-16      nissan-17      nissan-18       peugot-1
              1              1              1              1              1              1              1              2              1              1              2
       peugot-2       peugot-3       peugot-4       peugot-5       peugot-6       peugot-7       peugot-8       peugot-9      peugot-10      peugot-11     plymouth-1
              2              2              2              2              2              2              2              2              2              2              2
     plymouth-2     plymouth-3     plymouth-4     plymouth-5     plymouth-6     plymouth-7      porsche-1      porsche-2      porsche-3      porsche-4         saab-1
              1              1              1              1              1              2              2              2              2              2              2
         saab-2         saab-3         saab-4         saab-5         saab-6       subaru-1       subaru-2       subaru-3       subaru-4       subaru-5       subaru-6
              2              2              2              2              1              1              1              2              1              1              1
       subaru-7       subaru-8       subaru-9      subaru-10      subaru-11      subaru-12       toyota-1       toyota-2       toyota-3       toyota-4       toyota-5
```

```{r}
prcomp_M_pam_2 <- data.frame(
  prcomp(
    x = scale_M,
    center = FALSE,
    scale. = FALSE
  )$x[,1:2],
  Name = rownames(Dataset),
  Cluster = as.character(pam_Car_2$cluster),
  stringsAsFactors = FALSE
)
```

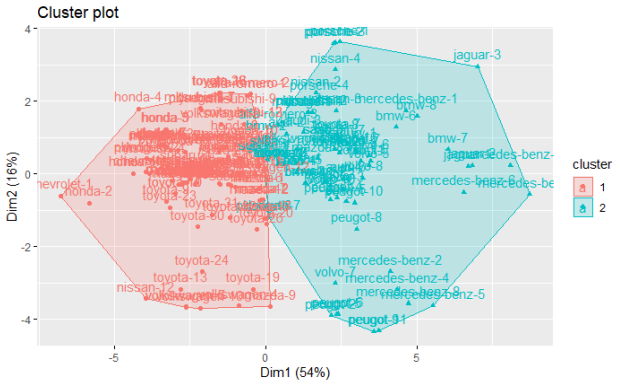
```{r}
require(ggplot2)
require(ggforce)
ggplot(prcomp_M_pam_2) +
  aes(x = PC1,y = PC2,color = Cluster,fill = Cluster,label = Name,group = Cluster) +
  geom_point() +
  ggrepel::geom_text_repel(color = "black",size = 3) +
  ggtitle("Scatter plot of decathlon principal components","Color corresponds to k-means cluster") +
  theme_bw() +
  theme(legend.position = "none")
```



```r
cluster::clusplot(
  scale_M,
  pam_Car_2$cluster,
  color=TRUE,
  shade=TRUE,
  labels=2,
  lines=0
)
```




```r
factoextra::fviz_cluster(
  object = pam_Car_2,
  data = scale_M
)
```



There is a overlap of observations in the groups.

```r
pam_stat_2 <- fpc::cluster.stats(
  d = dist(scale_M),
  clustering = pam_Car_2$cluster,
  G2 = TRUE,
  G3 = TRUE)

pam_stat_2$max.diameter
pam_stat_2$min.separation
```

```
[1] 10.53744
[1] 0.9923048
```

1.5. Conclusion

I have used Kmean, Clara, Fanny and Pam algorithm for Partitioning clustering of the dataset. Scatter plots describe the distribution of each observation separated by groups. In our case the scatter plot does not show much difference between the algorithms. That is why I choose fviz_cluster graph which display the clear separation and overlap between two groups. Out of all 4 algorithm, Kmean clustering is the best model for this dataset because the overlap between the groups is very minimal (may be this is because of multi dimension) followed by Clara. While comparing the Maximum diameter and minimum separation, I found that the value is min 9.561109 for Max diameter for Kmean with 2 clustering.

To find out optimal clustering for each algorithm, I have used Elbow and Silhouette plot. For this dataset all the algorithm Elbow plot has a steep bend at K = 2 and similarly Silhouette plot give cluster as 2 by drawing a vertical line.

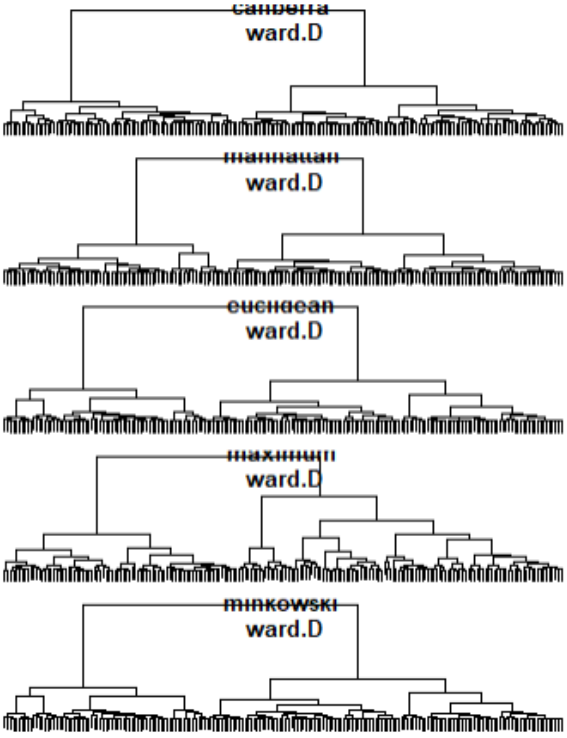
2. Hierarchical Clustering

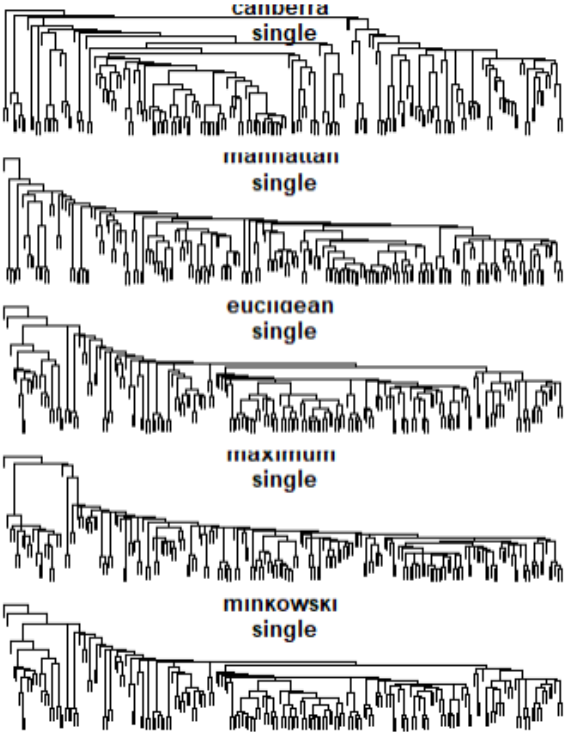
2.1. Using hclust()

2.1.1. Finding out best dist matrix and method

Used "canberra","manhattan","euclidean","maximum","minkowski" as distance matrix and "ward.D","single" as cluster method, trying to find out best coefficient . I used ward.D method for partitioning data into equal sized group and “single” for outlier detector.

```{r}
v_dist <- c(
  "canberra","manhattan","euclidean","maximum","minkowski"
)

list_dist <- lapply(
  X = v_dist,
  FUN = function(distance_method) dist(
    x = scale_M,
    method = distance_method
  )
)

names(list_dist) <- v_dist
v_hclust <- c(
  "ward.D","single"
)

list_hclust <- list()
for(j in v_dist) for(k in v_hclust) list_hclust[[j]][[k]] <- hclust(
  d = list_dist[[j]],
  method = k
)
par(
  mfrow = c(length(v_dist),length(v_hclust)),
  mar = c(0,0,0,0),
  mai = c(0,0,0,0),
  oma = c(0,0,0,0)
)
for(j in v_dist) for(k in v_hclust) plot(
  x = list_hclust[[j]][[k]],
  labels = FALSE,
  axes = FALSE,
  main = paste("\n",j,"\n",k)
)

```






```r
M_coef <- matrix(
  data = NA,
  nrow = length(v_dist),
  ncol = length(v_hclust)
)
rownames(M_coef) <- v_dist
colnames(M_coef) <- v_hclust
for(j in v_dist) for(k in v_hclust) try({
  M_coef[j,k] <- cluster::coef.hclust(
    object = list_hclust[[j]][[k]]
  )
})
M_coef
```

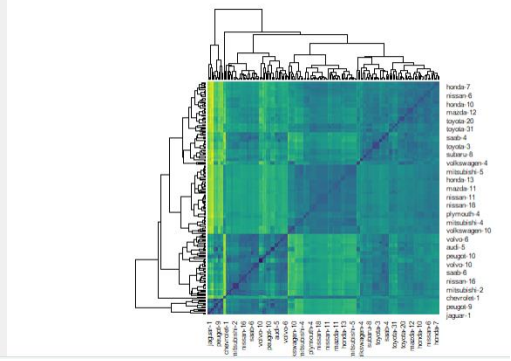
```
             ward.D    single
canberra  0.9964903 0.8320666
manhattan 0.9974664 0.8787228
euclidean 0.9952938 0.8365360
maximum   0.9909364 0.8399699
minkowski 0.9952938 0.8365360
```

In this case

- Manhattan & Ward.D gave highest score so considered to find equal size grouping
- Canberra & Single gave lowest score so considered to find outlier detection

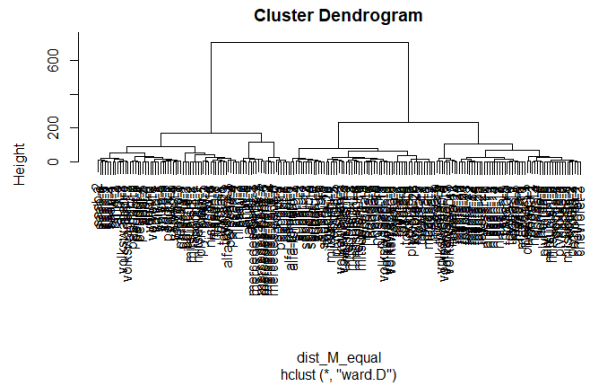
```r
dist_M_equal <- dist(
  x = scale_M,
  method = "manhattan"
)
```

```r
heatmap(
  x = as.matrix(dist_M_equal),
  col = viridis::viridis(256)
)
```



This heat map show the distance matrix and distance between each observations. Dark teal color shows a relation between the observations. In this case we can identify 4 squares which illustrate the groups.

```r
hclust_M_equal <- hclust(
  d = dist_M_equal,
  method = "ward.D"
)
plot(
  x = hclust_M_equal
)
```



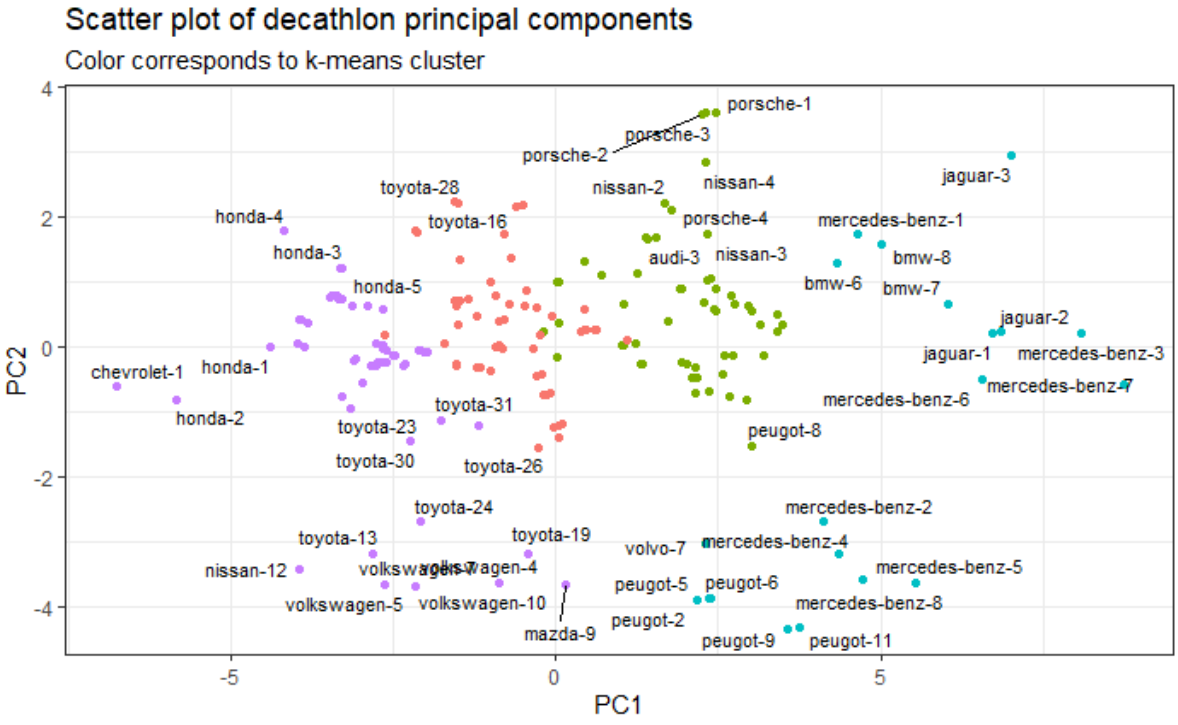
As there are more observations we are not able to see the levels clearly. But this Dendrogram give us an idea how each observations are grouped. If we cut our tree in two or four groups then might be able to get equal distributions.

2.1.2 Group Equal Proportion : Tree cut 4

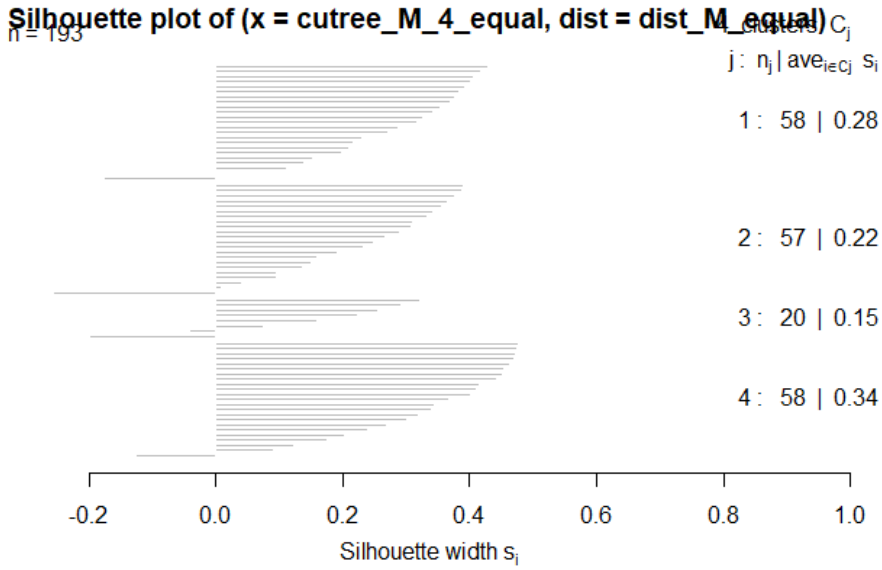
```{r}
cutree_M_4_equal <- cutree(
  tree = hclust_M_equal,
  k = 4
)
```

```{r}
prcomp_hir_M_equal <- data.frame(
  prcomp(
    x = scale_M,
    center = FALSE,
    scale. = FALSE
  )$x[,1:2],
  Name = rownames(Dataset),
  Cluster = as.character(cutree_M_4_equal),
  stringsAsFactors = FALSE
)
```

```{r}
require(ggplot2)
ggplot(prcomp_hir_M_equal) +
  aes(x = PC1,y = PC2,color = Cluster,fill = Cluster,label = Name,group = Cluster) +
  geom_point() +
  ggrepel::geom_text_repel(color = "black",size = 3) +
  ggtitle("Scatter plot of decathlon principal components","Color corresponds to k-means cluster") +
  theme_bw() +
  theme(legend.position = "none")
```



```r
silhouette_M_4_equal <- cluster::silhouette(
  x = cutree_M_4_equal,
  dist = dist_M_equal
)
plot(
  x = silhouette_M_4_equal
)
```



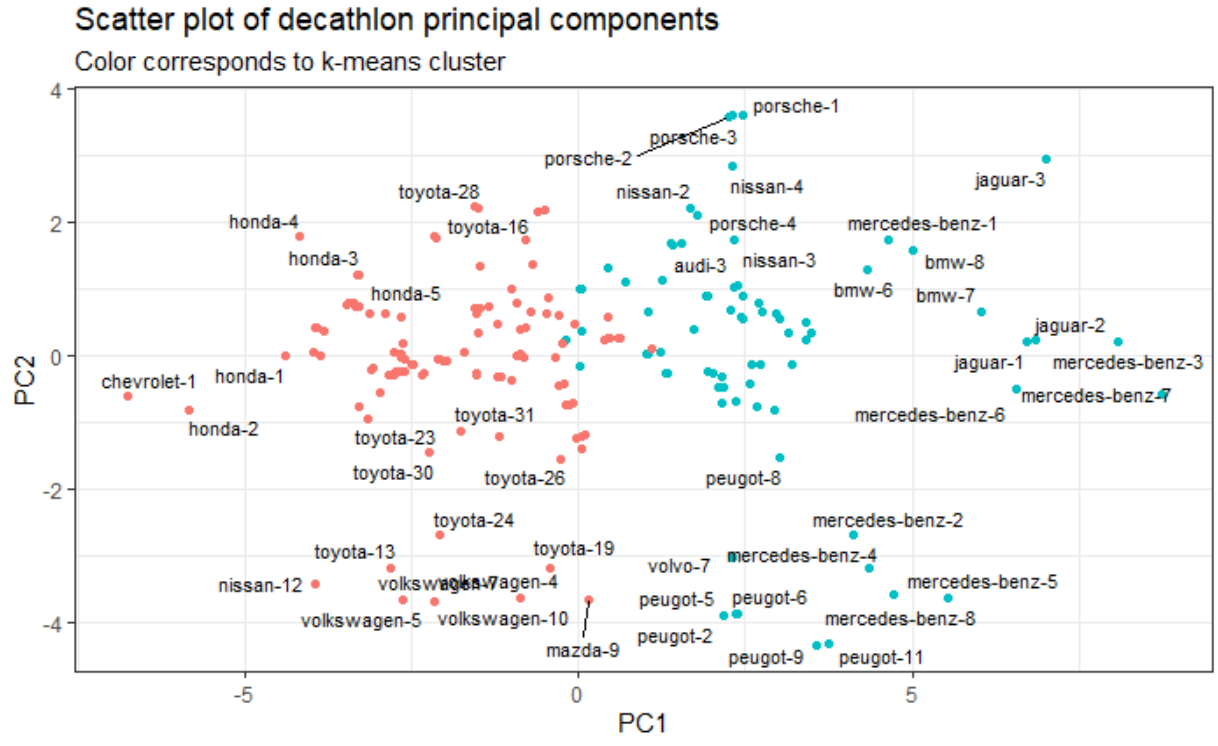
Silhouette plot will show whether the groups are good or not. If there are more bars in left side that means those observations are wrongly grouped.

2.1.2 Group Equal Proportion: Tree cut 2

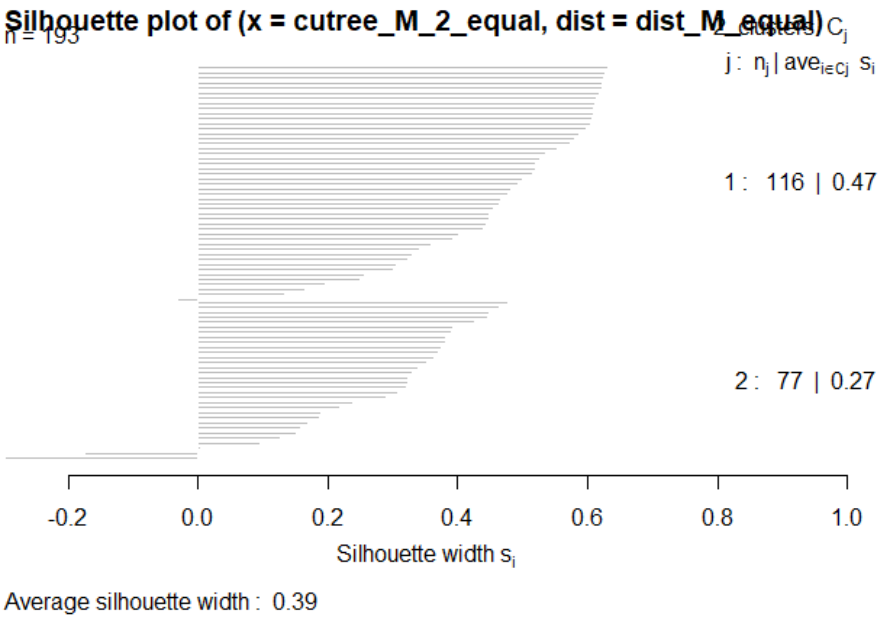
```{r}
cutree_M_2_equal <- cutree(
  tree = hclust_M_equal,
  k = 2
)
```

```{r}
prcomp_hir_M_equal_2 <- data.frame(
  prcomp(
    x = scale_M,
    center = FALSE,
    scale. = FALSE
  )$x[,1:2],
  Name = rownames(Dataset),
  Cluster = as.character(cutree_M_2_equal),
  stringsAsFactors = FALSE
)
```

```{r}
require(ggplot2)
ggplot(prcomp_hir_M_equal_2) +
  aes(x = PC1,y = PC2,color = Cluster,fill = Cluster,label = Name,group = Cluster) +
  geom_point() +
  ggrepel::geom_text_repel(color = "black",size = 3) +
  ggtitle("Scatter plot of decathlon principal components","Color corresponds to k-means cluster") +
  theme_bw() +
  theme(legend.position = "none")
```



```r
silhouette_M_2_equal <- cluster::silhouette(
  x = cutree_M_2_equal,
  dist = dist_M_equal
)
plot(
  x = silhouette_M_2_equal
)
```

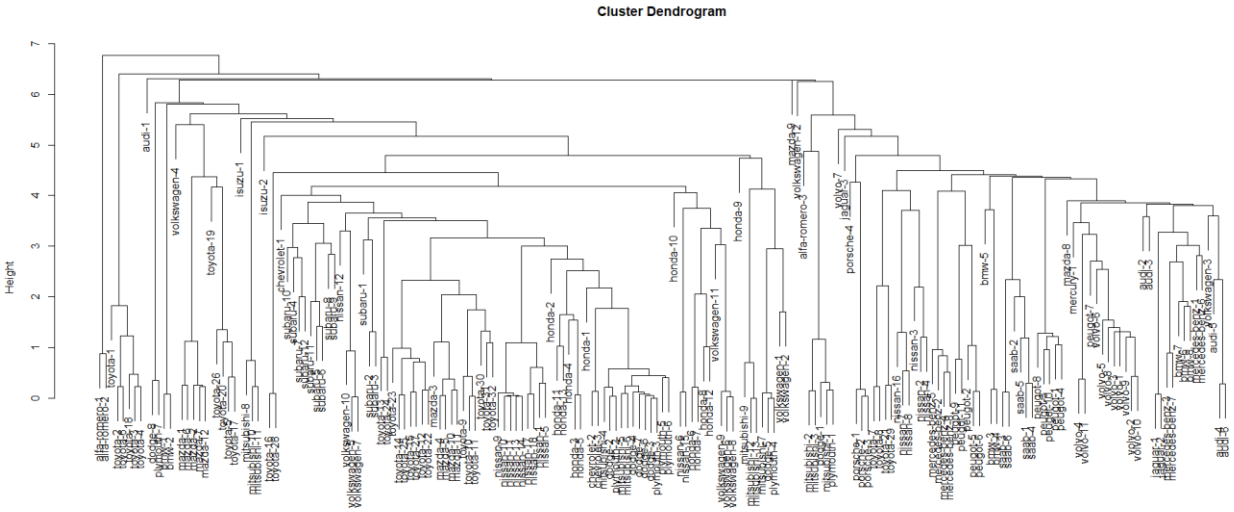


In case of cut tree in 2 groups there are less number of observations in the left side while comparing the tree cut in 4 groups.

2.1.2. Finding Outlier: Tree cut 2

```{r}
dist_M_outlier <- dist(
  x = scale_M,
  method = "canberra"
)
```

```{r}
hclust_M_outlier <- hclust(
  d = dist_M_outlier,
  method = "single"
)
plot(
  x = hclust_M_outlier
)
```



From above tree we can identify Audi-1 , Mazda – 9 , Volkswagan-12 and alfa-romero-1/2 are the outliers.

2.2. Using cluster::agnes()

2.2.1. Finding out best method

```{r}
v_pc <- prcomp(scale_M)$x[,1]
scale_M <- scale_M[order(v_pc),]
```

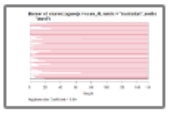
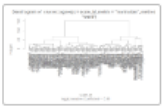
```{r}
v_methods <- c( "average", "single", "complete", "ward")
v_metric <- c( "canberra","manhattan","euclidean","maximum","minkowski" )
names(v_methods) <- c( "average", "single", "complete", "ward")
names(v_metric) <- c( "canberra","manhattan","euclidean","maximum","minkowski" )


ac <- function(x) {
  cluster::agnes(scale_M, method = x)$ac
}

purrr::map_dbl(v_methods, ac)

```

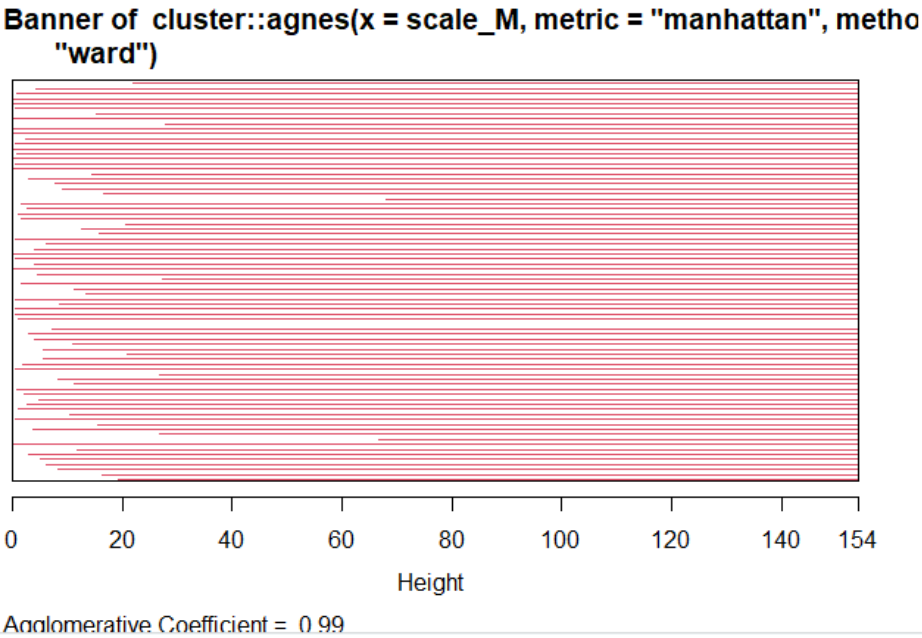
```
   average    single  complete      ward
 0.8958852 0.8365360 0.9485575 0.9815502
```

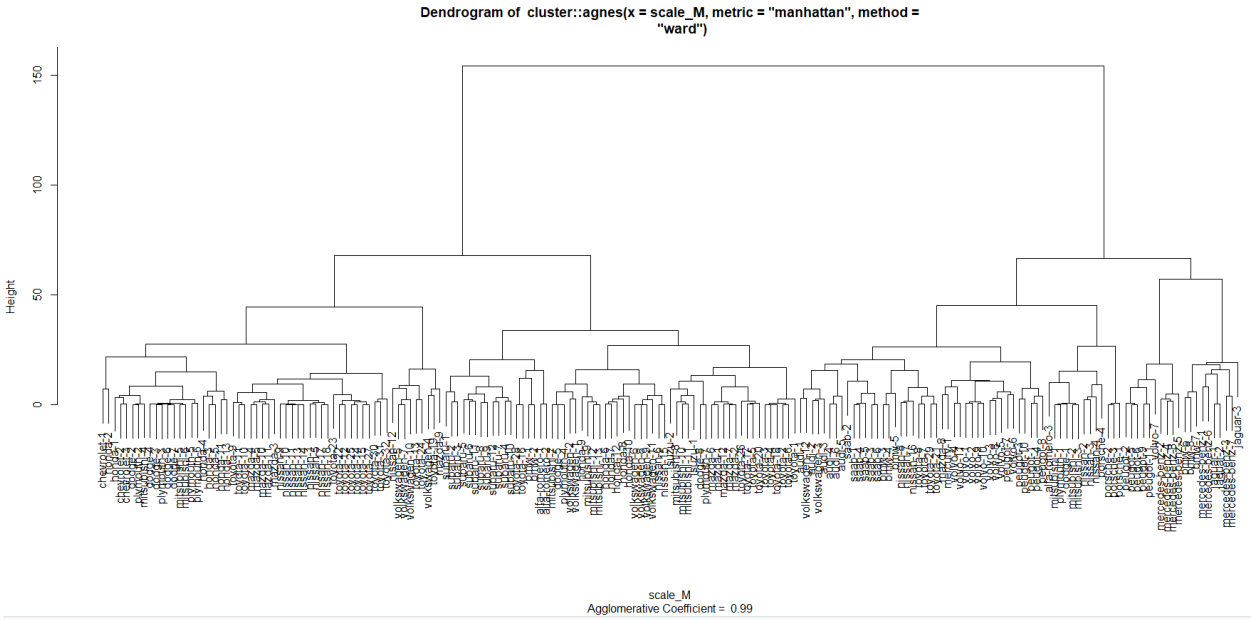
Ward has highest score so used this method for Equal proportion group and single has least score so used to find outlier detector.

2.2.2. Group Equal Proportion

```r
agnes_M_ward <- cluster::agnes(scale_M,metric = "manhattan", method = "ward")
plot(agnes_M_ward)
```





```{r}
print(agnes_M_ward)
```

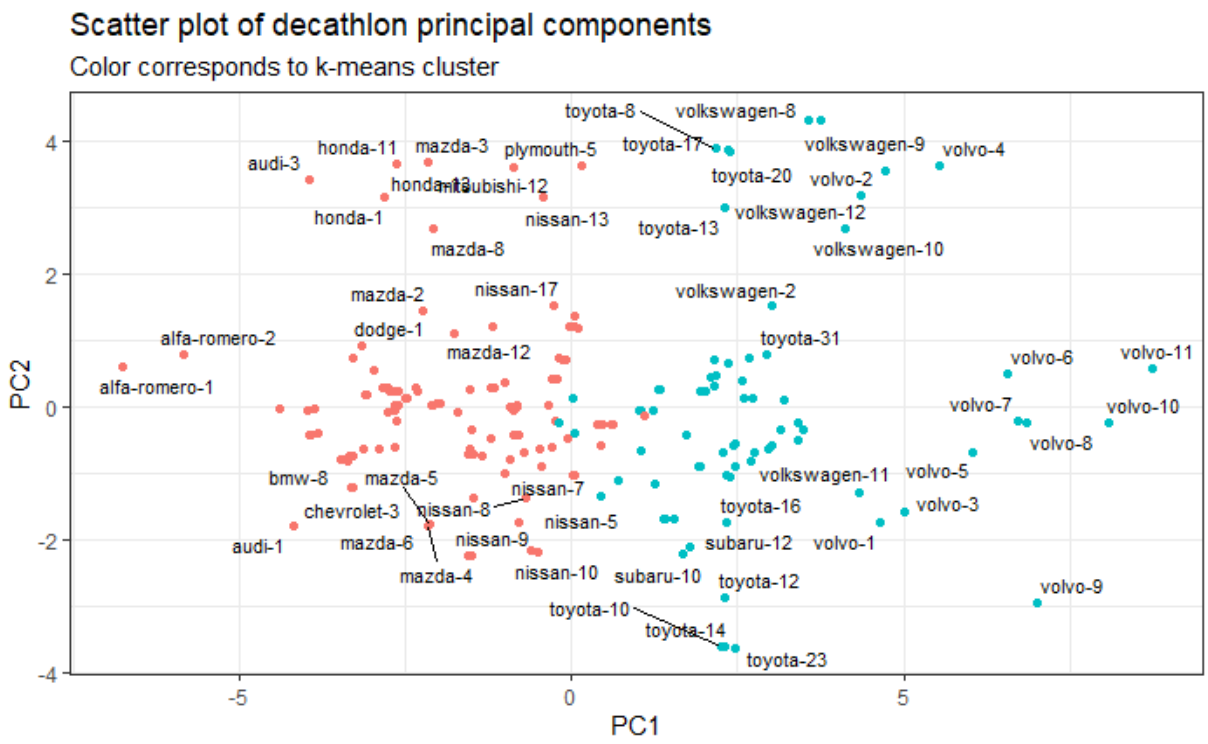
```
Call:    cluster::agnes(x = scale_M, metric = "manhattan", method = "ward")
Agglomerative coefficient:  0.9888474
Order of objects:
  [1] chevrolet-1      honda-2          honda-1          chevrolet-2      chevrolet-3      dodge-2          plymouth-2       mitsubishi-4     dodge-4          dodge-3
 [11] plymouth-3       dodge-6          dodge-7          mitsubishi-5     mitsubishi-6     plymouth-5       plymouth-6       honda-4          honda-3          honda-5
 [21] honda-11         honda-13         toyota-9         toyota-11        toyota-10        mazda-4          mazda-5          mazda-10         mazda-11         mazda-3
 [31] nissan-9         nissan-10        nissan-11        nissan-13        nissan-14        nissan-1         nissan-5         nissan-17        nissan-18        toyota-23
 [41] toyota-22        toyota-12        toyota-25        toyota-14        toyota-15        toyota-27        toyota-30        toyota-31        toyota-32        nissan-12
 [51] volkswagen-5     volkswagen-7     volkswagen-10    toyota-13        toyota-24        volkswagen-4     toyota-19        mazda-9          subaru-1         subaru-2
 [61] subaru-3         subaru-5         subaru-6         subaru-11        subaru-8         subaru-9         subaru-12        subaru-7         subaru-4         subaru-10
 [71] toyota-28        toyota-16        bmw-1            bmw-2            alfa-romero-1    alfa-romero-2    mitsubishi-7     dodge-5          plymouth-4       volkswagen-2
 [81] volkswagen-1     honda-9          mitsubishi-9     mitsubishi-12    mitsubishi-13    honda-6          honda-7          honda-12         honda-8          honda-10
 [91] volkswagen-6     volkswagen-8     volkswagen-9     volkswagen-11    nissan-6         nissan-15        isuzu-2          mitsubishi-8     mitsubishi-10    mitsubishi-11
[101] isuzu-1          dodge-8          plymouth-7       mazda-6          mazda-1          mazda-12         mazda-7          mazda-2          toyota-26        toyota-17
[111] toyota-5         toyota-20        toyota-3         toyota-4         toyota-18        toyota-2         toyota-6         toyota-1         audi-1           volkswagen-12
[121] audi-2           volkswagen-3     audi-3           audi-4           audi-6           audi-5           saab-2           saab-1           saab-4           saab-5
[131] saab-3           saab-6           bmw-3            bmw-4            bmw-5            nissan-8         nissan-7         nissan-16        toyota-8         toyota-7
[141] toyota-29        toyota-21        mazda-8          mercury-1        volvo-4          volvo-11         volvo-1          volvo-2          volvo-9          volvo-10
[151] volvo-3          volvo-8          volvo-5          peugot-7         volvo-6          peugot-3         peugot-10        peugot-1         peugot-4         peugot-8
[161] alfa-romero-3    mitsubishi-1     plymouth-1       dodge-1          mitsubishi-2     mitsubishi-3     nissan-2         nissan-3         nissan-4         porsche-4
[171] porsche-2        porsche-3        porsche-1        peugot-2         peugot-5         peugot-6         peugot-9         peugot-11        volvo-7          mercedes-benz-2
[181] mercedes-benz-4  mercedes-benz-8  mercedes-benz-5  bmw-6            bmw-8            bmw-7            mercedes-benz-1  mercedes-benz-6  jaguar-1         jaguar-2
[191] mercedes-benz-3  mercedes-benz-7  jaguar-3
Height (summary):
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
  0.0000   0.4009   2.3848   7.2276   8.0518 154.0565

Available components:
[1] "order"     "height"    "ac"        "merge"     "diss"      "call"      "method"    "order.lab" "data"
```

2.2.3. Group Equal Proportion – cut tree 2

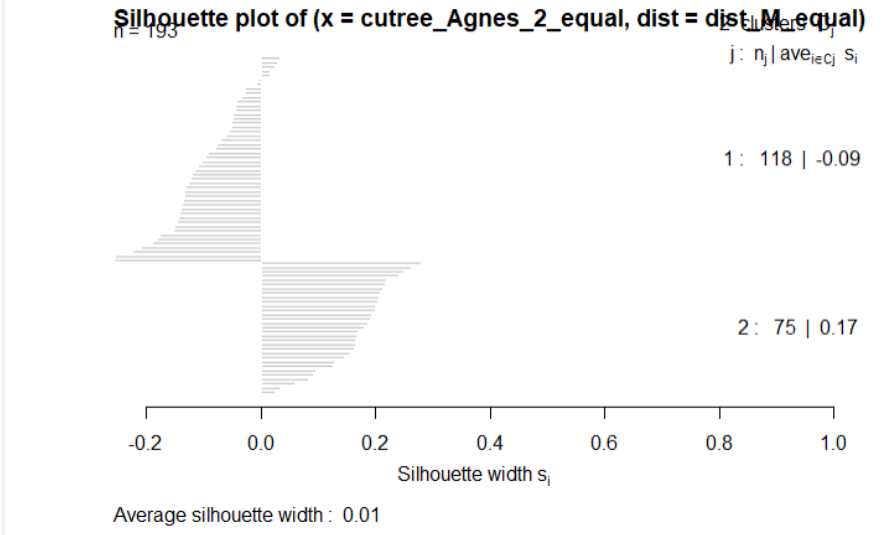
```{r}
cutree_Agnes_2_equal <- cutree(
  tree = agnes_M_ward,
  k = 2
)
```

```{r}
prcomp_Agnes_equal_2 <- data.frame(
  prcomp(
    x = scale_M,
    center = FALSE,
    scale. = FALSE
  )$x[,1:2],
  Name = rownames(Dataset),
  Cluster = as.character(cutree_Agnes_2_equal),
  stringsAsFactors = FALSE
)
```

```{r}
require(ggplot2)
ggplot(prcomp_Agnes_equal_2) +
  aes(x = PC1,y = PC2,color = Cluster,fill = Cluster,label = Name,group = Cluster) +
  geom_point() +
  ggrepel::geom_text_repel(color = "black",size = 3) +
  ggtitle("Scatter plot of decathlon principal components","Color corresponds to k-means cluster") +
  theme_bw() +
  theme(legend.position = "none")
```



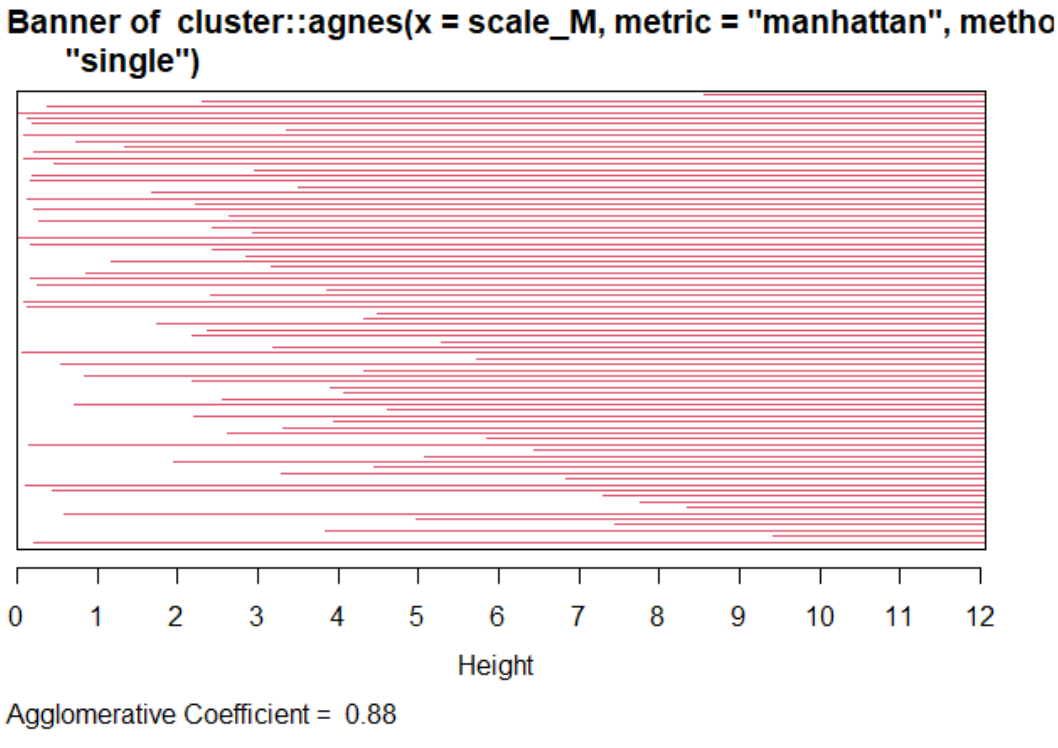

```{r}
silhouette_Agnes_2_equal <- cluster::silhouette(
  x = cutree_Agnes_2_equal,
  dist = dist_M_equal
)
plot(
  x = silhouette_Agnes_2_equal
)
```

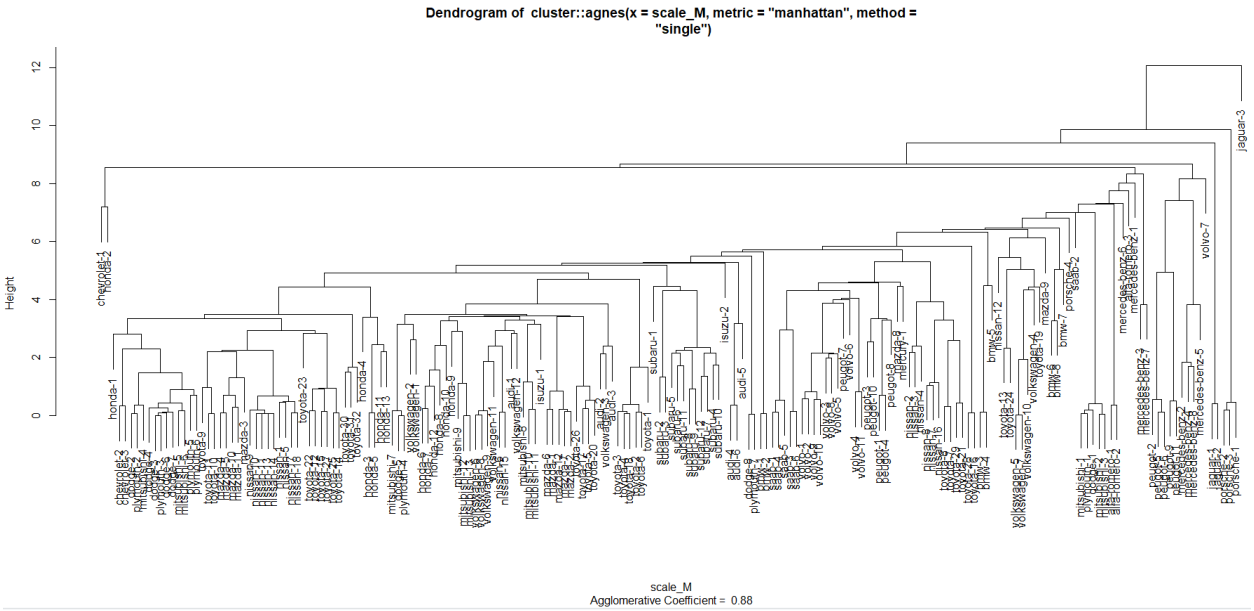


This has more number of observations on Left side.

2.2.4. Finding out Outlier

```r
agnes_M_single <- cluster::agnes(scale_M,metric = "manhattan", method = "single")
plot(agnes_M_single)
```



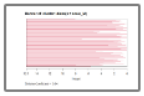


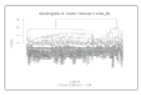
This tree graph shows the outlier clearly as Jaguar-3 , Honda-2 and Chevrolet-1 .

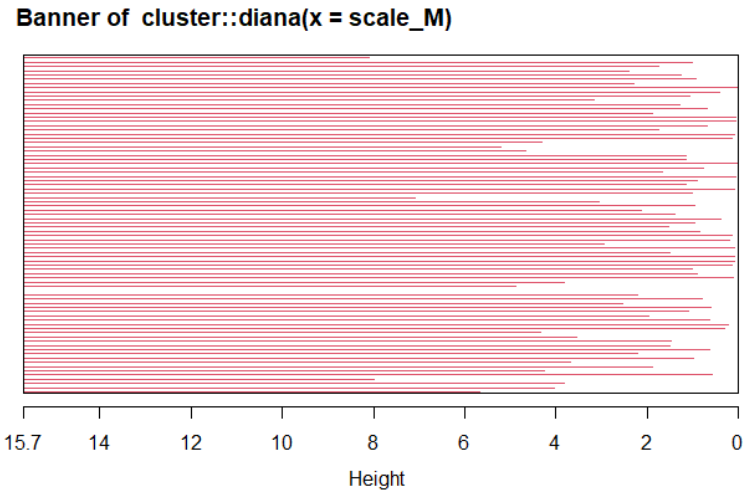
2.3. Using cluster::diana()

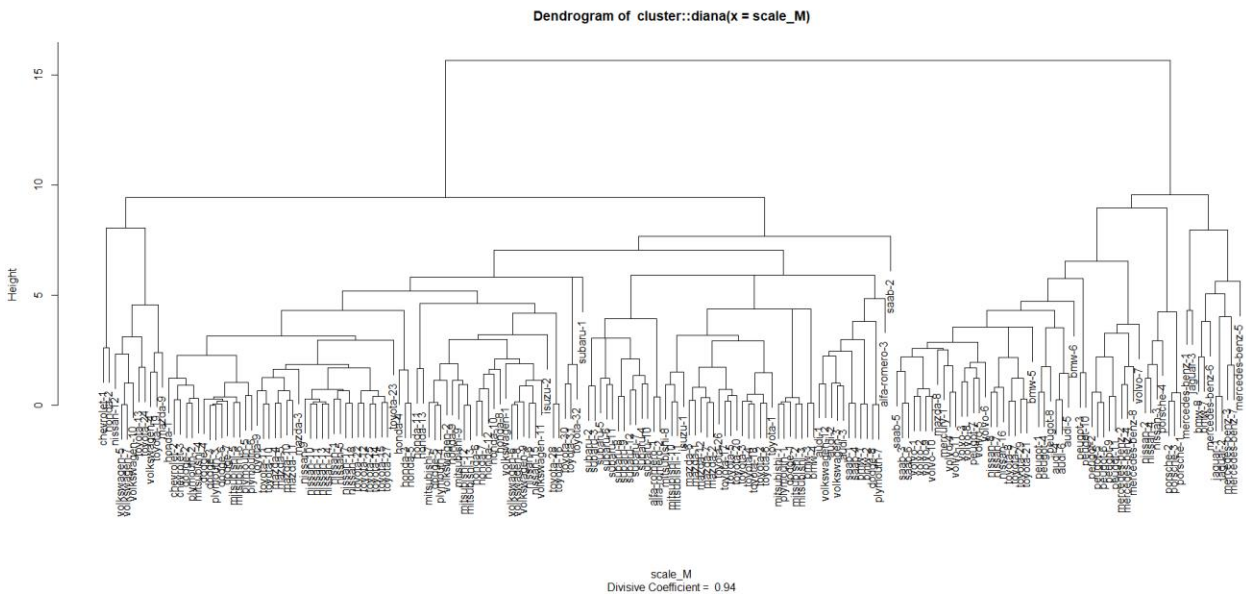
Diana does not support any method so we will use this algo with default value.

```r
diana_M <- cluster::diana(scale_M)
plot(diana_M)
```









2.3.1. Cut tree

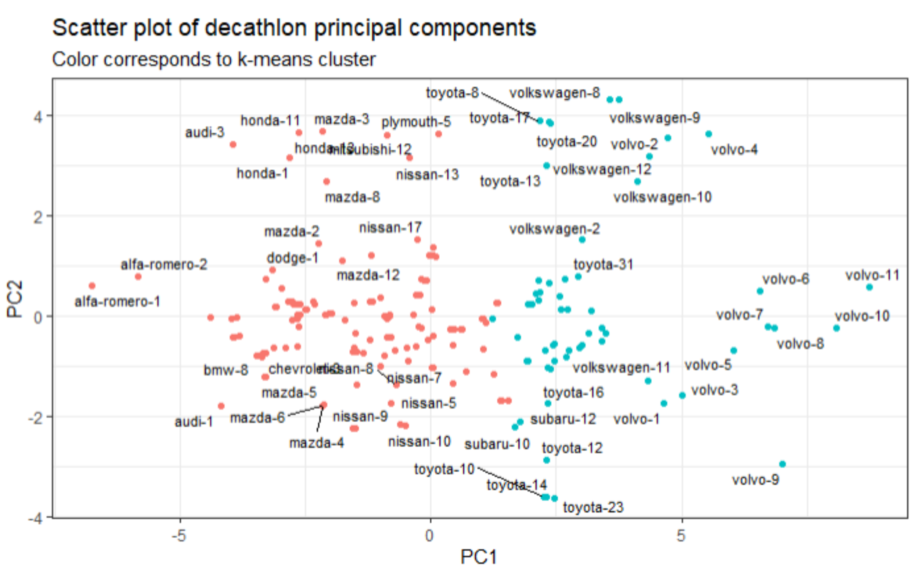
```{r}
cutree(  tree = diana_M,  k = 4)
```

```
  [1] 1 1 2 2 2 1 2 2 2 2 2 2 2 2 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 2 1 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [77] 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 3 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
[153] 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 4 3 4 4 4 4 4 4 4 4 4
```

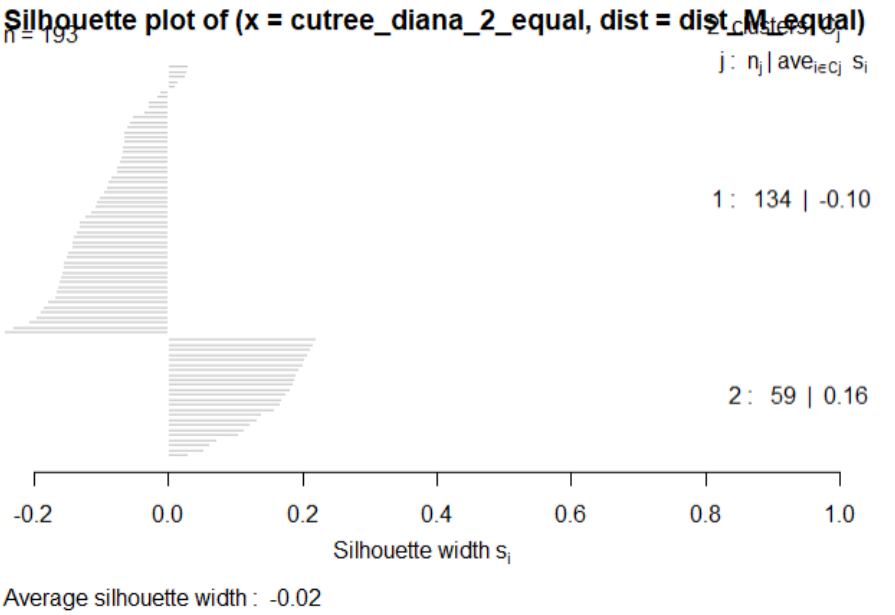
```{r}
cutree_diana_2_equal <- cutree(
  tree = diana_M,
  k = 2
)
```

```{r}
prcomp_diana_equal_2 <- data.frame(
  prcomp(
    x = scale_M,
    center = FALSE,
    scale. = FALSE
  )$x[,1:2],
  Name = rownames(Dataset),
  Cluster = as.character(cutree_diana_2_equal),
  stringsAsFactors = FALSE
)
```

```{r}
require(ggplot2)
ggplot(prcomp_diana_equal_2) +
  aes(x = PC1,y = PC2,color = Cluster,fill = Cluster,label = Name,group = Cluster) +
  geom_point() +
  ggrepel::geom_text_repel(color = "black",size = 3) +
  ggtitle("Scatter plot of decathlon principal components","Color corresponds to k-means cluster") +
  theme_bw() +
  theme(legend.position = "none")
```



```r
silhouette_diana_2_equal <- cluster::silhouette(
  x = cutree_diana_2_equal,
  dist = dist_M_equal
)
plot(
  x = silhouette_diana_2_equal
)
```



Diana also has more number of observations on left side.

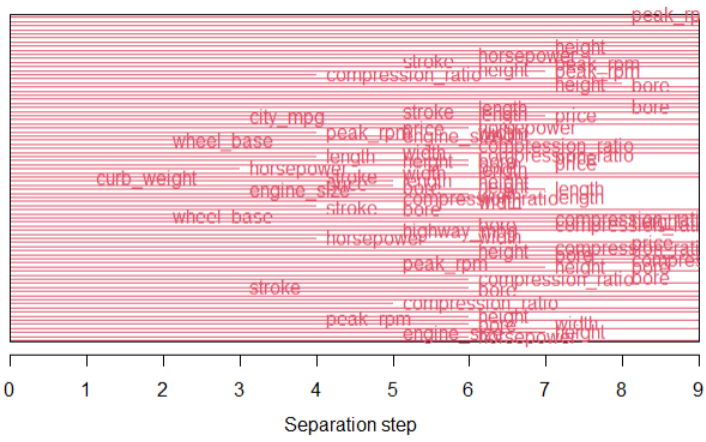
2.4. Using cluster::mona()

For mona I have converted the data observations into binary .

```r
binary_M <- scale_M
for(j in 1:ncol(binary_M)) binary_M[,j] <- as.numeric(
  binary_M[,j] >median(binary_M[,j])
)
mona_M <- cluster::mona(binary_M)
print(mona_M)
```

```
mona(x, ..) fit;  x of dimension 193x14
Order of objects:
  [1] chevrolet-1      honda-1          chevrolet-2      dodge-2          plymouth-2       chevrolet-3      mitsubishi-4     dodge-4          dodge-3
 [10] plymouth-3       dodge-6          dodge-7          mitsubishi-5     mitsubishi-6     plymouth-5       plymouth-6       nissan-1         nissan-5
 [19] nissan-17        nissan-18        nissan-9         nissan-10        nissan-11        nissan-13        nissan-14        toyota-28        toyota-16
 [28] honda-2          toyota-13        toyota-24        honda-4          honda-3          honda-5          nissan-12        honda-11         honda-13
 [37] toyota-9         toyota-11        toyota-10        toyota-30        toyota-31        subaru-3         toyota-23        mazda-4          mazda-5
 [46] mazda-10         mazda-11         mazda-3          toyota-22        toyota-12        toyota-25        toyota-14        toyota-15        toyota-27
 [55] subaru-1         subaru-2         subaru-11        honda-6          honda-7          honda-8          honda-12         mitsubishi-7     dodge-5
 [64] plymouth-4       mitsubishi-9     mitsubishi-12    mitsubishi-13    volkswagen-2     volkswagen-1     honda-9          isuzu-2          subaru-7
 [73] mitsubishi-8     mitsubishi-10    mitsubishi-11    volkswagen-5     volkswagen-7     volkswagen-10    volkswagen-6     volkswagen-8     volkswagen-9
 [82] subaru-5         subaru-6         subaru-8         mazda-6          mazda-1          toyota-17        toyota-26        toyota-20        mazda-12
 [91] mazda-7          volkswagen-11    audi-1           nissan-6         nissan-15        bmw-1            bmw-2            toyota-32        subaru-9
[100] subaru-12        honda-10         isuzu-1          subaru-4         subaru-10        alfa-romero-1    alfa-romero-2    nissan-2         nissan-4
[109] mercedes-benz-1  porsche-4        porsche-2        porsche-3        porsche-1        alfa-romero-3    mitsubishi-1     plymouth-1       dodge-1
[118] mitsubishi-2     mitsubishi-3     volkswagen-4     toyota-19        volkswagen-12    toyota-5         mazda-2          mazda-9          peugot-2
[127] peugot-5         peugot-6         dodge-8          plymouth-7       peugot-9         peugot-11        toyota-3         toyota-4         toyota-18
[136] toyota-2         toyota-6         toyota-1         jaguar-1         jaguar-2         volvo-7          mercedes-benz-2  mercedes-benz-4  mercedes-benz-8
[145] mercedes-benz-5  mercedes-benz-3  mercedes-benz-7  audi-2           bmw-8            volkswagen-3     audi-3           audi-4           audi-6
[154] audi-5           bmw-6            bmw-7            toyota-8         toyota-7         toyota-29        toyota-21        saab-2           saab-1
[163] saab-4           saab-5           volvo-1          volvo-2          volvo-9          volvo-10         volvo-3          volvo-8          saab-3
[172] saab-6           nissan-8         nissan-7         nissan-16        peugot-7         volvo-5          volvo-6          nissan-3         bmw-3
[181] bmw-4            bmw-5            mazda-8          mercury-1        volvo-4          volvo-11         mercedes-benz-6  jaguar-3         peugot-3
[190] peugot-10        peugot-1         peugot-4         peugot-8
Variable used:
  [1] peak_rpm          NULL             NULL             NULL             NULL             NULL             NULL             NULL
  [9] NULL              NULL             NULL             NULL             NULL             NULL             NULL             NULL
 [17] NULL              NULL             NULL             height           NULL             NULL             NULL             NULL
 [25] horsepower        NULL             stroke           NULL             NULL             peak_rpm         NULL             NULL
 [33] height            peak_rpm         NULL             compression_ratio NULL            NULL             NULL             NULL
 [41] bore              height           NULL             NULL             NULL             NULL             NULL             NULL
 [49] NULL              NULL             NULL             NULL             NULL             bore             NULL             length
 [57] stroke            NULL             length           price            city_mpg         NULL             NULL             NULL
 [65] NULL              NULL             horsepower       price            width            peak_rpm         height           engine_size
 [73] NULL              NULL             wheel_base       NULL             NULL             compression_ratio NULL            NULL
 [81] width             NULL             compression_ratio length          price            bore             height           price
 [89] bore              NULL             horsepower       length           width            NULL             stroke           NULL
 [97] curb_weight       length           height           price            bore             height           length           engine_size
[105] NULL              width            NULL             length           compression_ratio width           NULL             NULL
[113] stroke            bore             NULL             NULL             NULL             NULL             wheel_base       NULL
[121] compression_ratio height           bore             compression_ratio NULL            NULL             NULL             highway_mpg
[129] NULL              width            NULL             horsepower       NULL             NULL             price            NULL
[137] NULL              compression_ratio NULL            height           bore             NULL             NULL             NULL
[145] compression_ratio NULL             peak_rpm         bore             height           NULL             NULL             NULL
[153] NULL              bore             NULL             compression_ratio NULL            NULL             NULL             stroke
[161] bore              NULL             NULL             NULL             NULL             NULL             NULL             NULL
[169] NULL              compression_ratio NULL            NULL             NULL             NULL             NULL             NULL
[177] NULL              height           peak_rpm         NULL             width            bore             NULL             NULL
[185] NULL              NULL             height           engine_size      NULL             horsepower       NULL             NULL
Separation step:
  [1] 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 7 0 0 0 0 6 0 5 0 0 7 0 0 6 7 0 4 0 0 0 0 8 7 0 0 0 0 0 0 0 0 0 0 0 0 8 0 6 5 0 6 7 3 0 0 0 0 0 6 5 6 4 6 5 0 0 2 0
 [77] 0 6 0 0 5 0 6 4 7 6 5 7 6 0 3 6 5 0 4 0 1 5 6 4 5 6 7 3 0 6 0 7 5 6 0 0 4 5 0 0 0 0 2 0 7 8 6 7 0 0 0 5 0 6 0 4 0 0 8 0 0 7 0 6 7 0 0 0 8 0 5 8 7 0 0 0
[153] 0 8 0 6 0 0 0 3 6 0 0 0 0 0 0 0 0 5 0 0 0 0 0 0 0 6 4 0 7 6 0 0 0 0 7 5 0 6 0 0 0

Available components:
[1] "data"      "hasNA"     "order"     "variable"  "step"      "order.lab" "call"
```

```{r}
plot(mona_M)
```



2.5. Conclusions

For hierarchical clustering, I used hclust(), cluster::agnes(), cluster::diana(), cluster::mona() to find outlier and partition the data into two equal proportion.

For hclust created a matrix of different combination of distance matrices and methods to find out optimal combination for outlier detection and partition in equal proportion. As Manhattan and Ward.D has highest score, used in partition equal group whereas Canberra & Single has lowest score, used in outlier detection. silhouette plots help us to identify the optimal tree cut to find equal proportions groups. In this case we used tree cut 2 as a smaller number of observations are in left side of the plot, that means less number of observations are wrongly predicted in a group. I am considering this model as best model for partition the equal proportion group. The dendrogram plot by using single method used for displaying the outlier. In our case we identified couple of outlier but the graph was not that clear. So rejecting this model to detect the outlier.

For agnes I have used ward method to identify equal number of observations in groups. Again, used silhouette plot to identify optimal tree cut but in this case there are lot of observations in left side of plot. That is why rejected this model for equal group partition. The dendrogram plot created by using single method gave us a clear tree diagram and able to identify the outlier. So accepting this model for outlier detection.

Diana does not support any specific method , that is why used default Diana algo . In this scenario found lot of observations in left side of the silhouette plot. That is why not considering this model for equal group partition.

Mona use only binary observation in calculation that is why have to convert the dataset into binary format. There are nothing much information out of Mona statistics, that is why rejecting this model for both equal group partition and outlier detection.

Below are the final models :

- Equal proportion partition :
    - Hclust() : Manhattan and Ward.D , 2 cluster tree cut
- Outlier Detection :
    - Agnes() : Manhattan and Single