Data 609 - HW6

Avery Davidowitz

2023-04-25

Import

```
library(mltools)
library(factoextra)
library(class)
library(cluster)
```

EX 1

Use a data set such as the PlantGrowth in R to calculate three different distance metrics and discuss the results.

```
summary(PlantGrowth)
```

```
##      weight       group
##  Min.   :3.590   ctrl:10
##  1st Qu.:4.550   trt1:10
##  Median :5.155   trt2:10
##  Mean   :5.073
##  3rd Qu.:5.530
##  Max.   :6.310
```

```
head(PlantGrowth)
```

```
##   weight group
## 1   4.17  ctrl
## 2   5.58  ctrl
## 3   5.18  ctrl
## 4   6.11  ctrl
## 5   4.50  ctrl
## 6   4.61  ctrl
```

Since we have a combination of numeric and categorical data I will first hot encode the data using the mltools package before calculating any distance.

```
pg_dt <- data.table::as.data.table(PlantGrowth)
encoded_df <- mltools::one_hot(pg_dt)
```

The L1 norm (Manhattan distance) for any 2 observations is given by:

$$D_m(x, y) = \sum_{1}^{D} |x_i - y_i|$$

This distance is already coded in the base stats function "dist". For example, the manhattan distance between D(2, 21) is:

```
man_dist <- dist(rbind(encoded_df[2,], encoded_df[21,]), method="manhattan")
encoded_df[2,]
```

```
##    weight group_ctrl group_trt1 group_trt2
## 1:   5.58          1          0          0
```

```
encoded_df[21,]
```

```
##    weight group_ctrl group_trt1 group_trt2
## 1:   6.31          0          0          1
```

```
man_dist
```

```
##      1
## 2 2.73
```

The L2 norm (Euclidean distance) for any 2 observations is given by:

$$D_m(x, y) = \sqrt{\sum_{1}^{D} (x_i - y_i)^2}$$

This distance is already coded in the base stats function "dist". For example, the euclidean distance between D(11, 23) is:

```
euc_dist <- dist(rbind(encoded_df[11,], encoded_df[23,]), method="euclidean")
encoded_df[11,]
```

```
##    weight group_ctrl group_trt1 group_trt2
## 1:   4.81          0          1          0
```

```
encoded_df[23,]
```

```
##    weight group_ctrl group_trt1 group_trt2
## 1:   5.54          0          0          1
```

```
euc_dist
```

```
##          1
## 2 1.591509
```

In general any power of the Minkowski distance can be called with the "dist" function. For example:

```
mink_dist <- dist(rbind(encoded_df[1,], encoded_df[3,]), method = "minkowski", p = 4)
mink_dist
```

```
##      1
## 2 1.01
```

Since weight is a continuous variable distance metrics such as edit distance, Jaccard similarity and Hamming distance are of very limited value.

EX 2

Now use a higher dimensional data set mtcars, try the same three distance metrics in the previous question and discuss the results.

```
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

Since the data in mtcars is all numeric I don't have to do any encoding. Our distance metrics are much higher compared to the first data set since the max value from PlantGrowth was 6.310 and none of these distances are normalized.

```
man_dist <- dist(rbind(mtcars[2,], mtcars[21,]), method="manhattan")
man_dist
```

```
##               Mazda RX4 Wag
## Toyota Corona            65
```

```
euc_dist <- dist(rbind(mtcars[11,], mtcars[23,]), method="euclidean")
euc_dist
```

```
##             Merc 280C
## AMC Javelin  139.1182
```

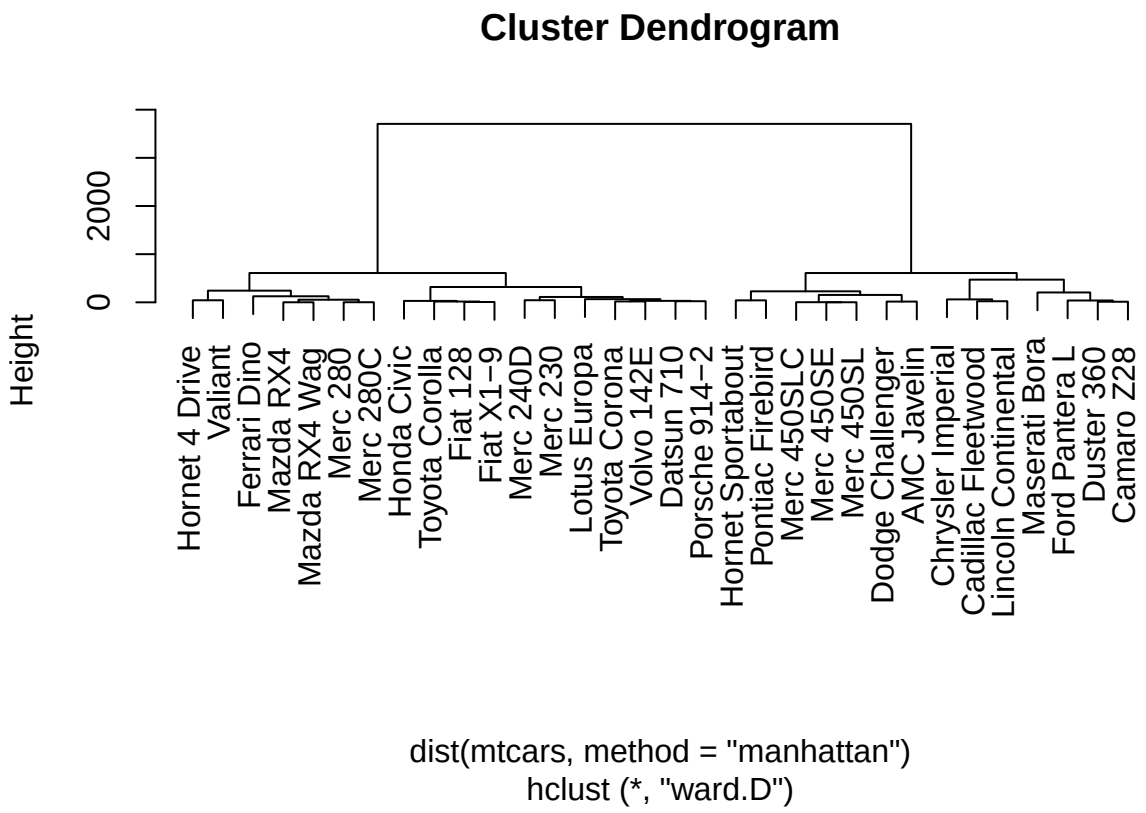
```
mink_dist <- dist(rbind(mtcars[1,], mtcars[3,]), method = "minkowski", p = 4)
mink_dist
```

```
##            Mazda RX4
## Datsun 710   52.1481
```

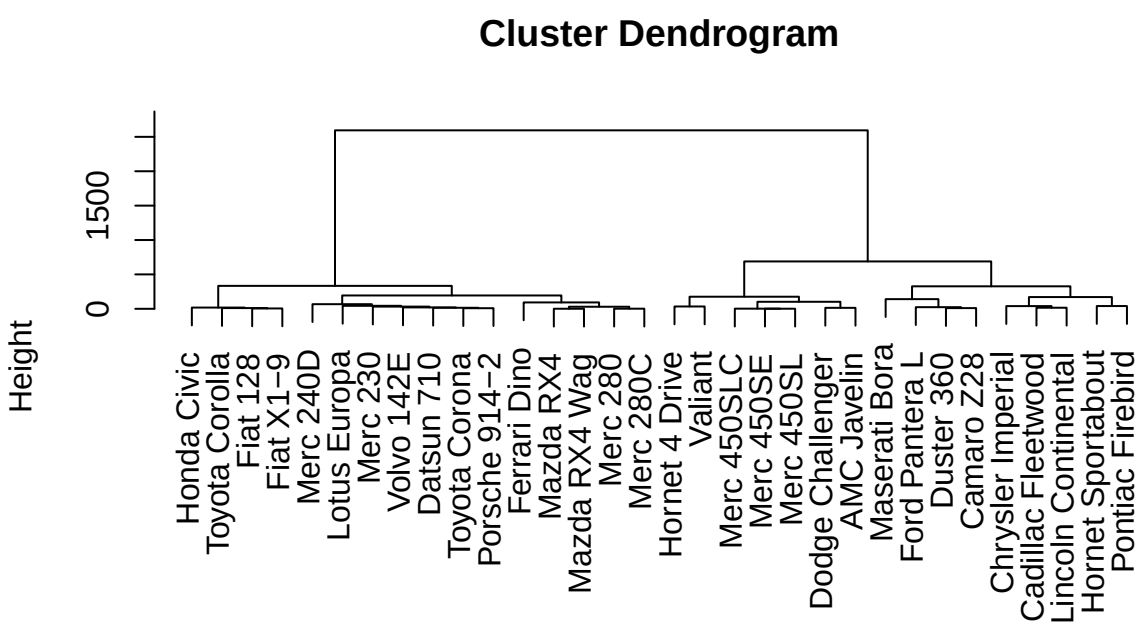
EX 3

Use the built-in data set mtcars to carry out hierarchy clustering using two different distance metrics and compare if they get the same results. Discuss the results.

```r
hc_man <- hclust(dist(mtcars, method="manhattan"), method = "ward.D")
hc_euc <- hclust(dist(mtcars, method="euclidean"), method = "ward.D")
plot(hc_man)
```

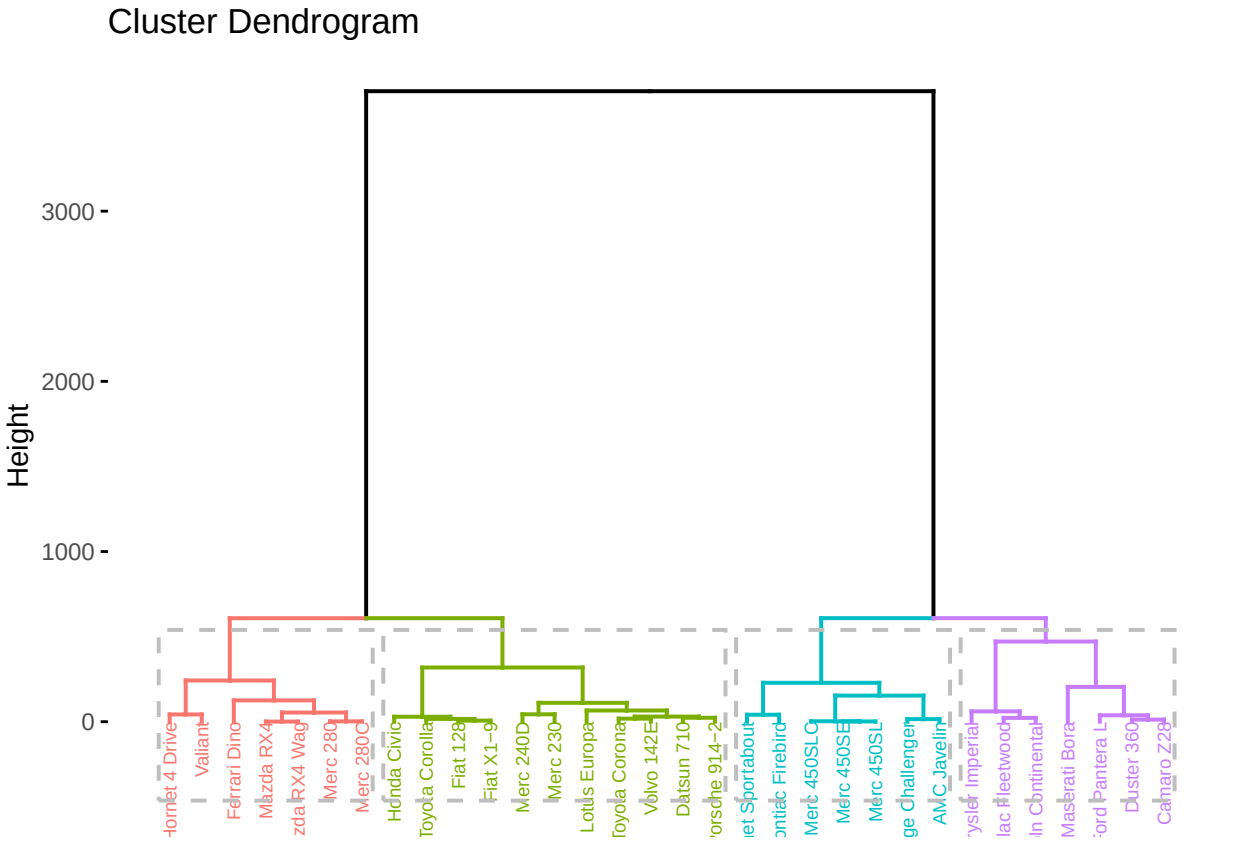



```r
plot(hc_euc)
```

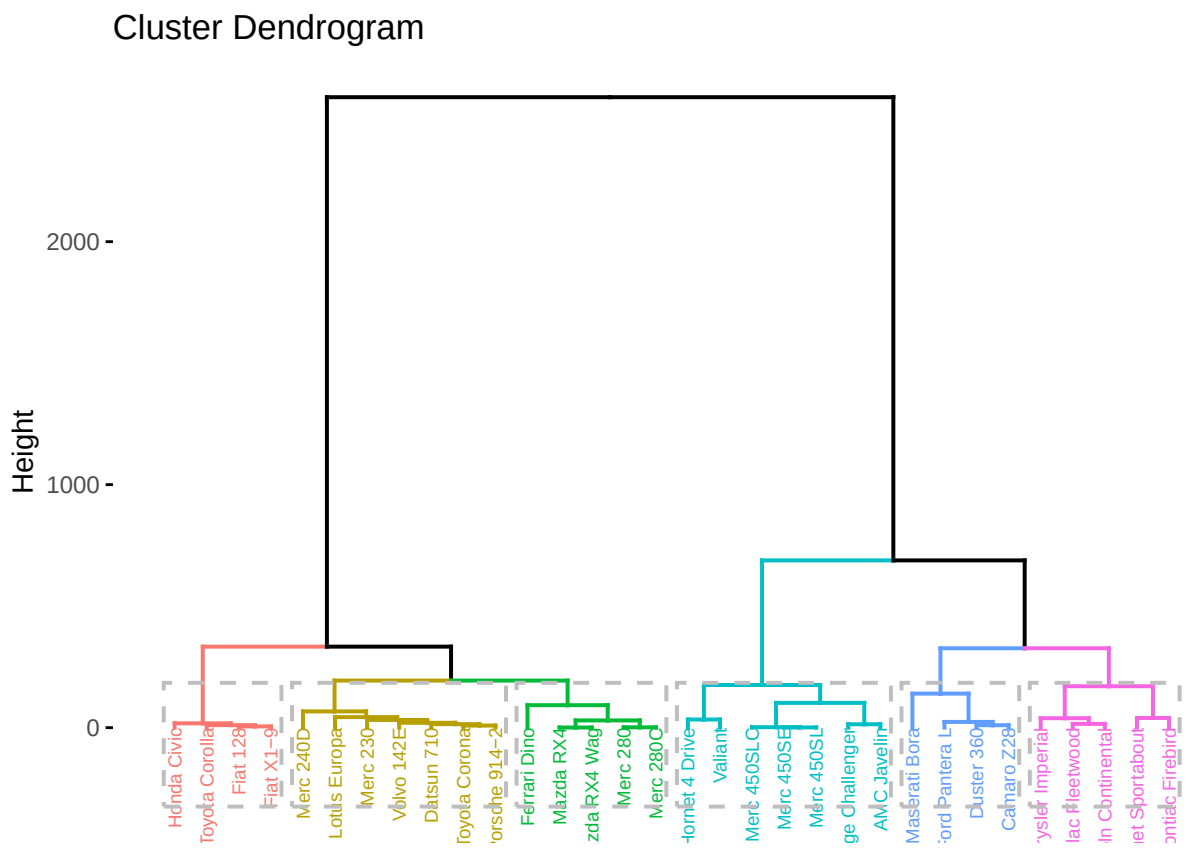


From the tree diagram it looks like the Manhattan distance split the data set up into fewer distinct groupings. We can use the "cutree" function to split our data set into specific number of groups based on how to tree looks. For the Manhattan I will use k=4 and for Euclidian k=6. We can then compare the distances' clustering.

```
fviz_dend(hc_man, k=4, rect = TRUE, cex = 0.5)
```




```
fviz_dend(hc_euc, k=6, rect = TRUE, cex = 0.5)
```



Ex 4

Load the well known Fisher's iris flower data set that consists of 150 samples for three species (50 samples each species). The four measures or features are the lengths and widths of sepals and petals. Use the kNN clustering to analyze this iris data set by selecting 120 samples for training and 30 samples for testing.

```
train_indices <- sample(nrow(iris), 120)
train_data <- iris[train_indices, 1:4]
train_labels <- iris[train_indices, 5]
test_data <- iris[-train_indices, 1:4]
test_labels <- iris[-train_indices, 5]

k <- 3
predicted_labels <- knn(train_data, test_data, train_labels, k)

accuracy <- sum(predicted_labels == test_labels) / length(test_labels)
cat("Accuracy:", round(accuracy, 2))
```

```
## Accuracy: 0.97
```

EX 5

Use the iris data set to carry out k-means clustering. Compare the restults to the actual classes and estimate the clustering accuracy.

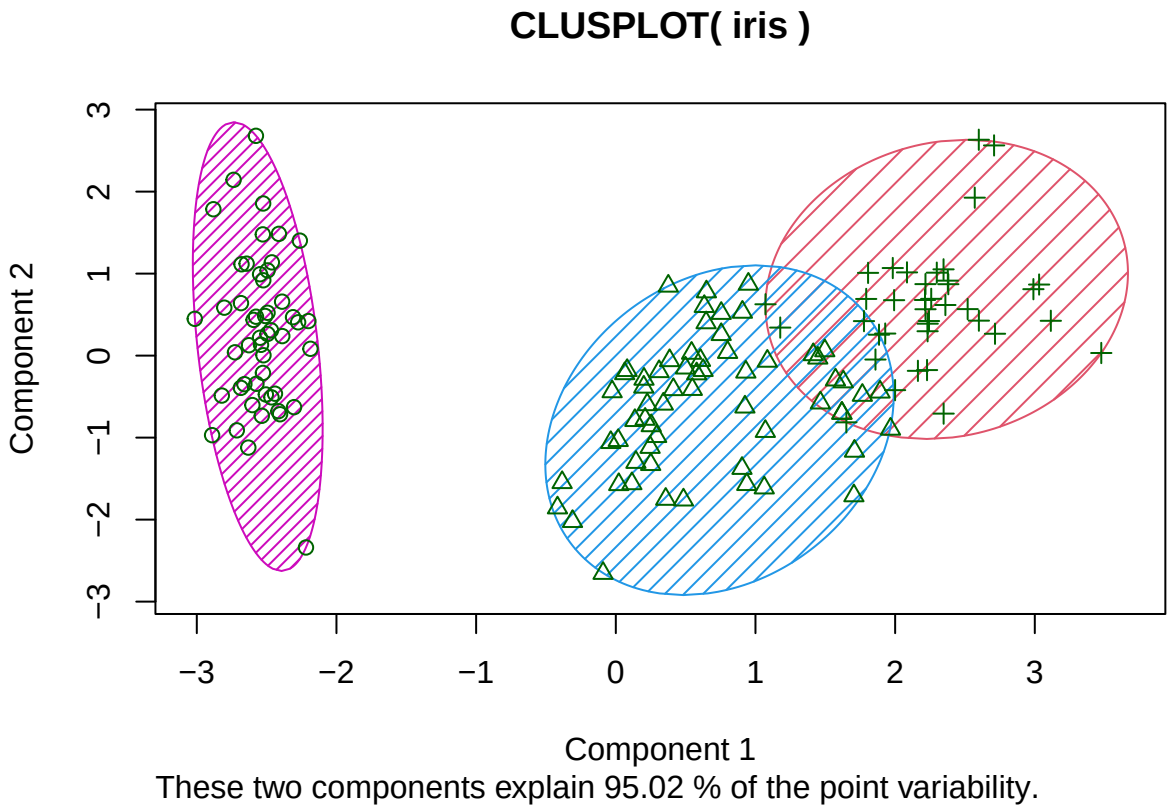
```
k <- 3
kmeans_results <- kmeans(iris[, 1:4], centers = k, nstart = 20)

table(kmeans_results$cluster, iris[, 5])
```

```
##
##     setosa versicolor virginica
##   1     50          0         0
##   2      0         48        14
##   3      0          2        36
```

```
actual_classes <- as.numeric(iris[, 5])
predicted_classes <- kmeans_results$cluster
num_correct <- sum(actual_classes == predicted_classes)
accuracy <- num_correct / nrow(iris)
cat("Clustering Accuracy:", round(accuracy, 2))
```

```
## Clustering Accuracy: 0.89
```

```
clusplot(iris, kmeans_results$cluster, color=T, shade=T, labels=0, lines=0)
```



While k means clustering predicts the class of setosa very well it struggles to differentiate between versicolor and virginica which have significant overlap. For this data set kNN performs much better.