CMDA-3654

Homework 9

Michael La Vance

Due as a .pdf upload

Problem 1: [50 pts] k-means clustering

Consider the Hotdog dataset shown in `hotdogs.csv`. Load in the data but ignore the first column as we'll pretend we know nothing other than the `Sodium` and `Calorie` content of the hotdogs.

a. Carry out K-means clustering with 2, 3, 4, 5 clusters. Don't forget to scaled the data first using:

```
hotdogs <- read.csv(file = "hotdogs.csv")
hotdogs.scaled <- scale(hotdogs[, 2:3])
```

```
hotdogs <- read.csv(file = "hotdogs.csv")
hotdogs.scaled <- scale(hotdogs[, 2:3])

km2 <- kmeans(hotdogs.scaled, centers = 2)
km3 <- kmeans(hotdogs.scaled, centers = 3)
km4 <- kmeans(hotdogs.scaled, centers = 4)
km5 <- kmeans(hotdogs.scaled, centers = 5)
```

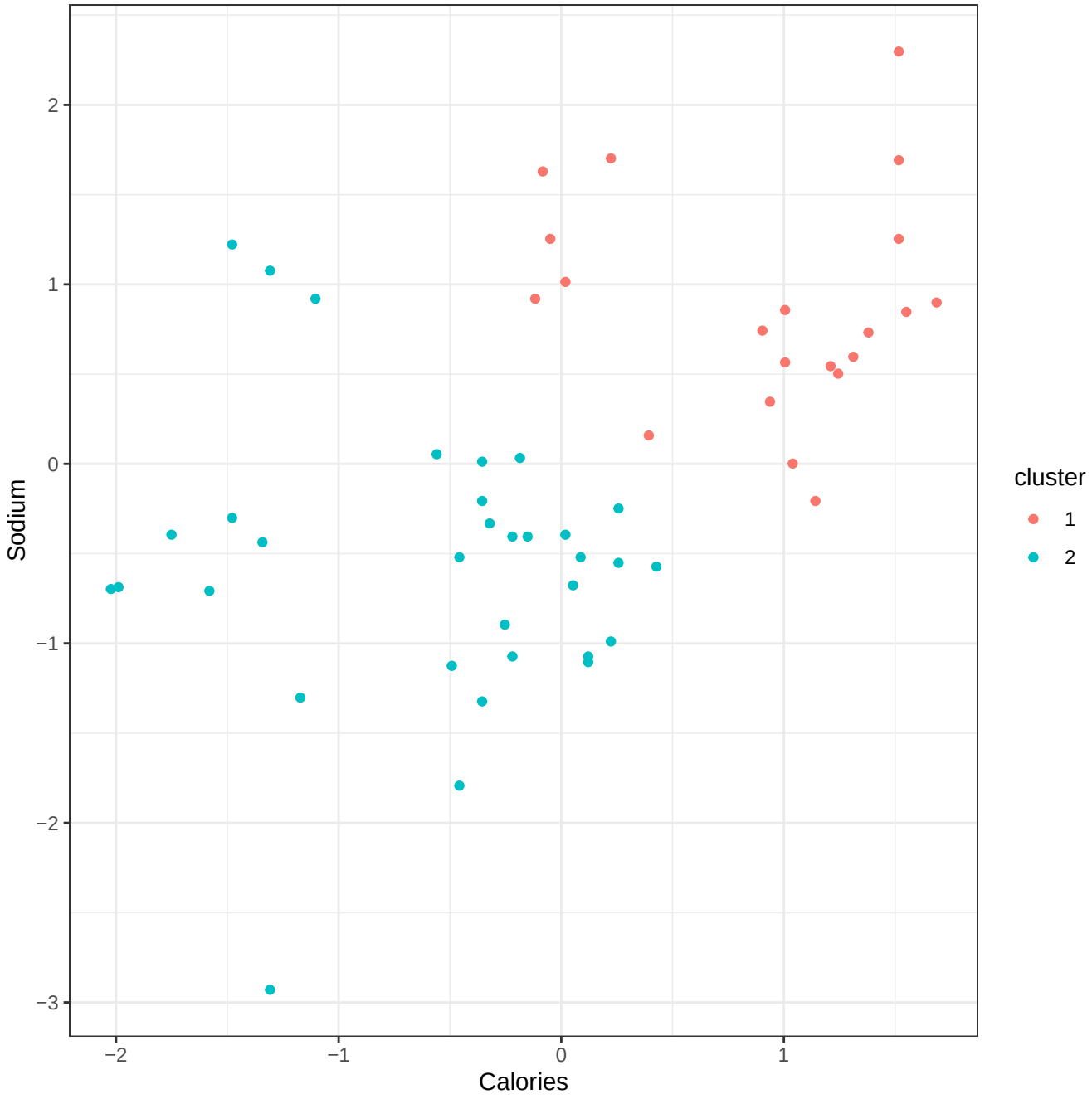
b. Plot the clusters from part (a) using `ggplot()` and assigning different colors and plot characters to the clusters found using `kmeans()`.

Hint: If `km.result <- kmeans(....)`, then the cluster assignments are in `km.result$cluster`, then simply add this to a new data.frame:

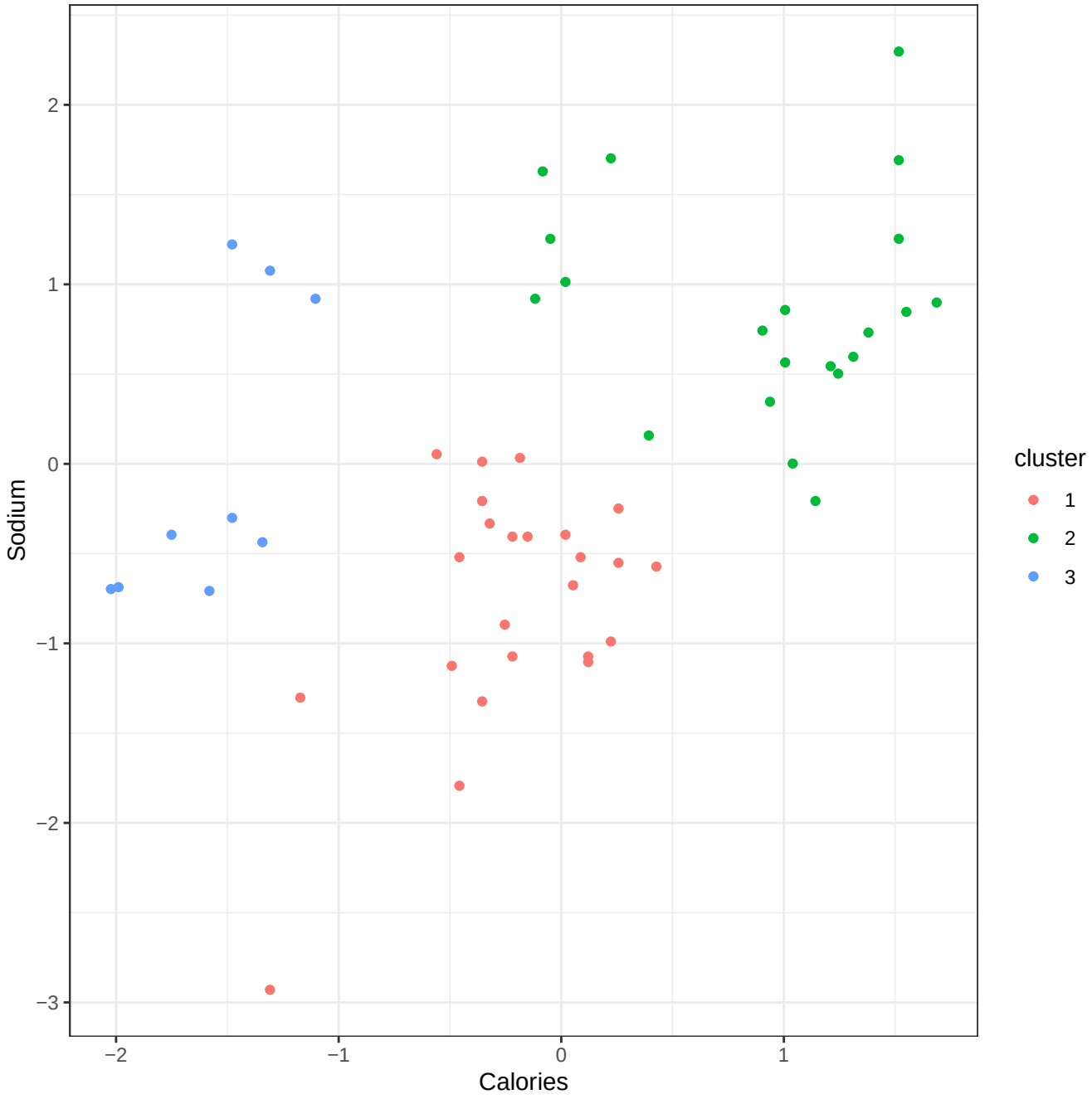
```
hotdogs2.scaled <- cbind(hotdogs.scaled, "cluster" = as.factor(km.result$cluster))
```

and then use `ggplot()` accordingly to make the plot.

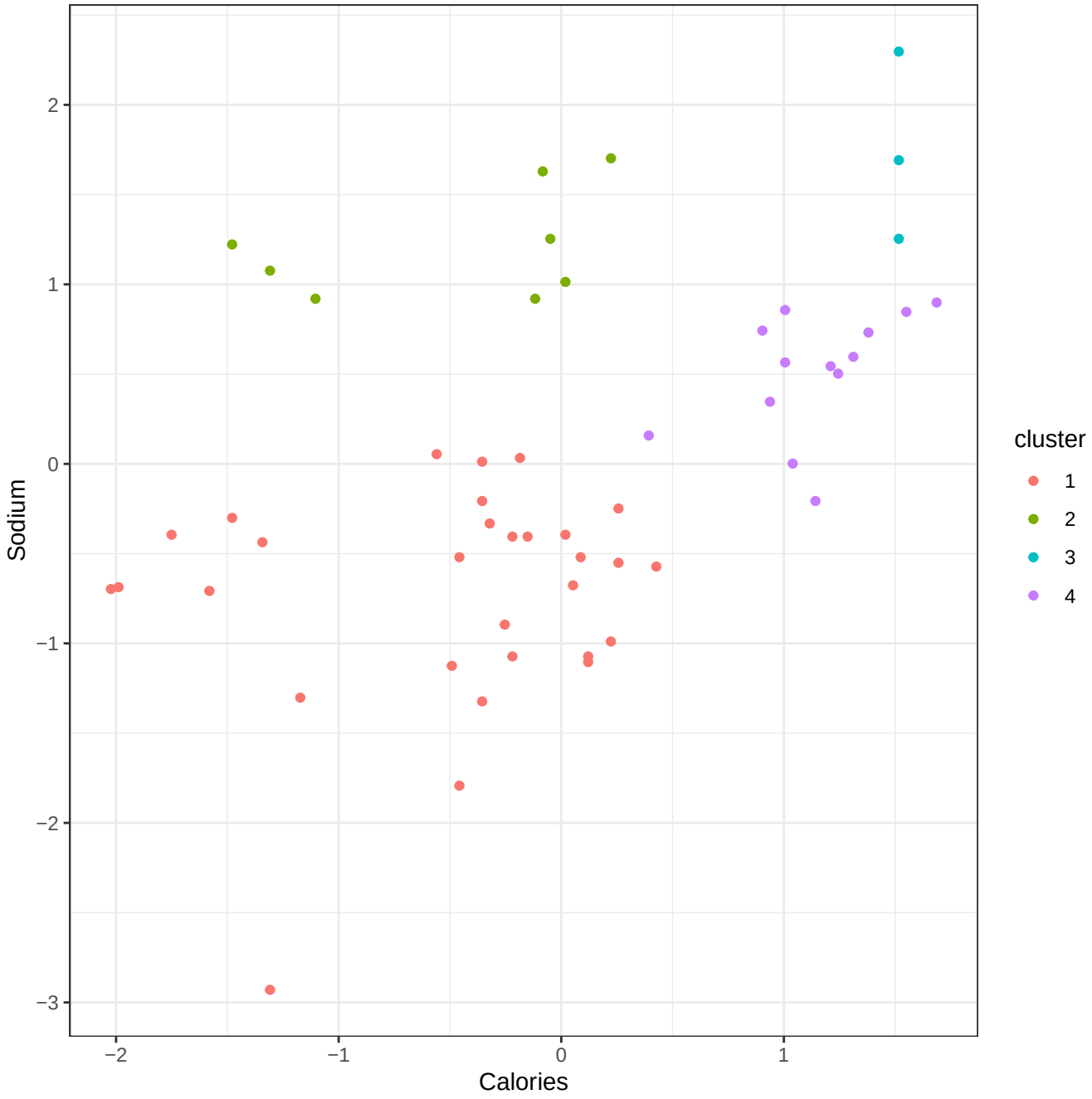
```
library(ggplot2)


hotdogs2.scaled <- cbind(hotdogs.scaled, "cluster" = as.factor(km2$cluster))
hotdogs3.scaled <- cbind(hotdogs.scaled, "cluster" = as.factor(km3$cluster))
hotdogs4.scaled <- cbind(hotdogs.scaled, "cluster" = as.factor(km4$cluster))
hotdogs5.scaled <- cbind(hotdogs.scaled, "cluster" = as.factor(km5$cluster))


hd2 <- as.data.frame(hotdogs2.scaled)
hd3 <- as.data.frame(hotdogs3.scaled)
hd4 <- as.data.frame(hotdogs4.scaled)
hd5 <- as.data.frame(hotdogs5.scaled)


hd2$cluster <- as.factor(hd2$cluster)
hd3$cluster <- as.factor(hd3$cluster)
hd4$cluster <- as.factor(hd4$cluster)
hd5$cluster <- as.factor(hd5$cluster)

ggplot( data = hd2, aes(x = Calories, y = Sodium)) + geom_point(aes(color = cluster)) + theme_bw()
```



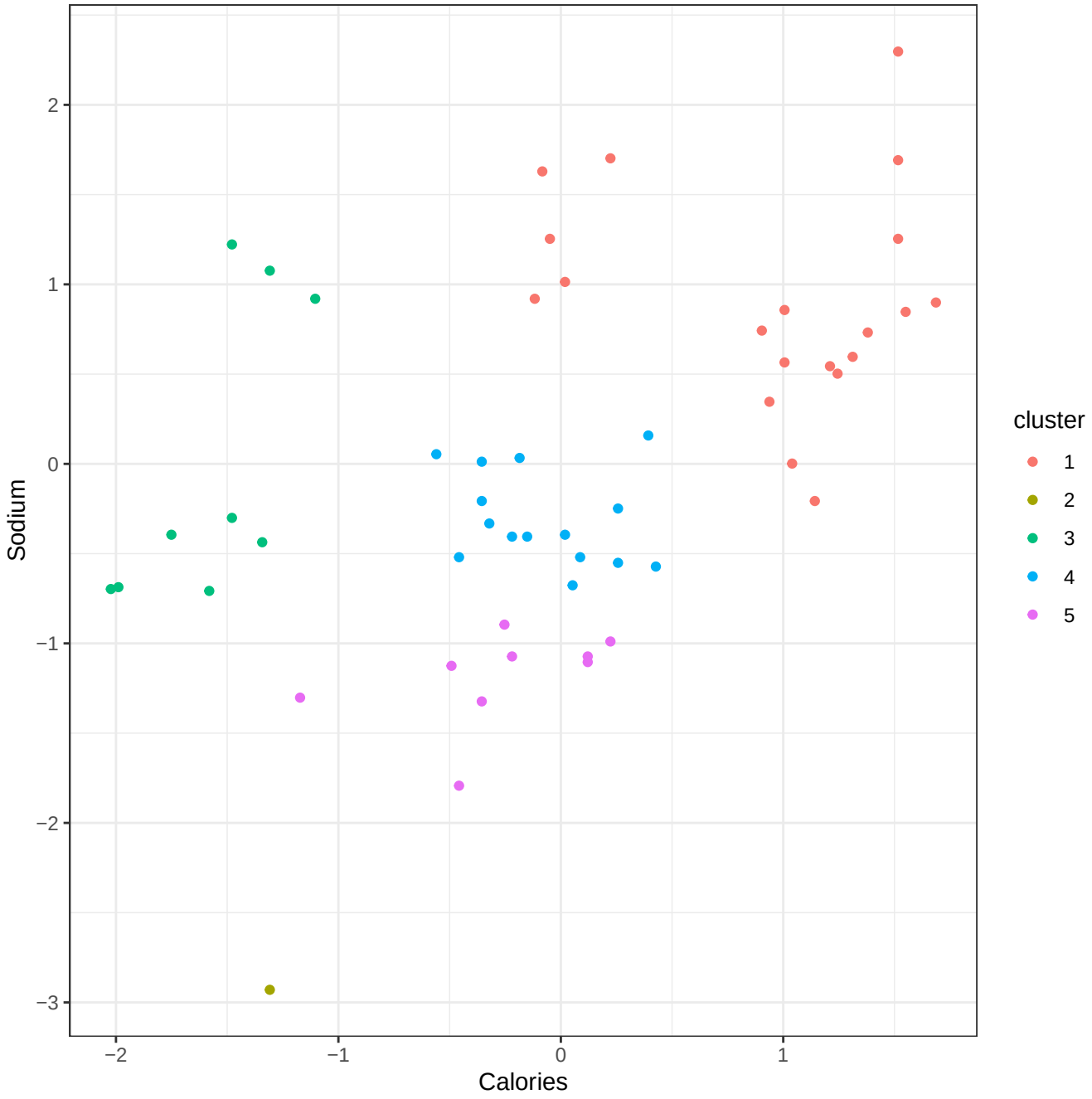

```
ggplot( data = hd3, aes(x = Calories, y = Sodium)) + geom_point(aes(color = cluster)) + theme_bw()
```




```
ggplot( data = hd4, aes(x = Calories, y = Sodium)) + geom_point(aes(color = cluster)) + theme_bw()
```



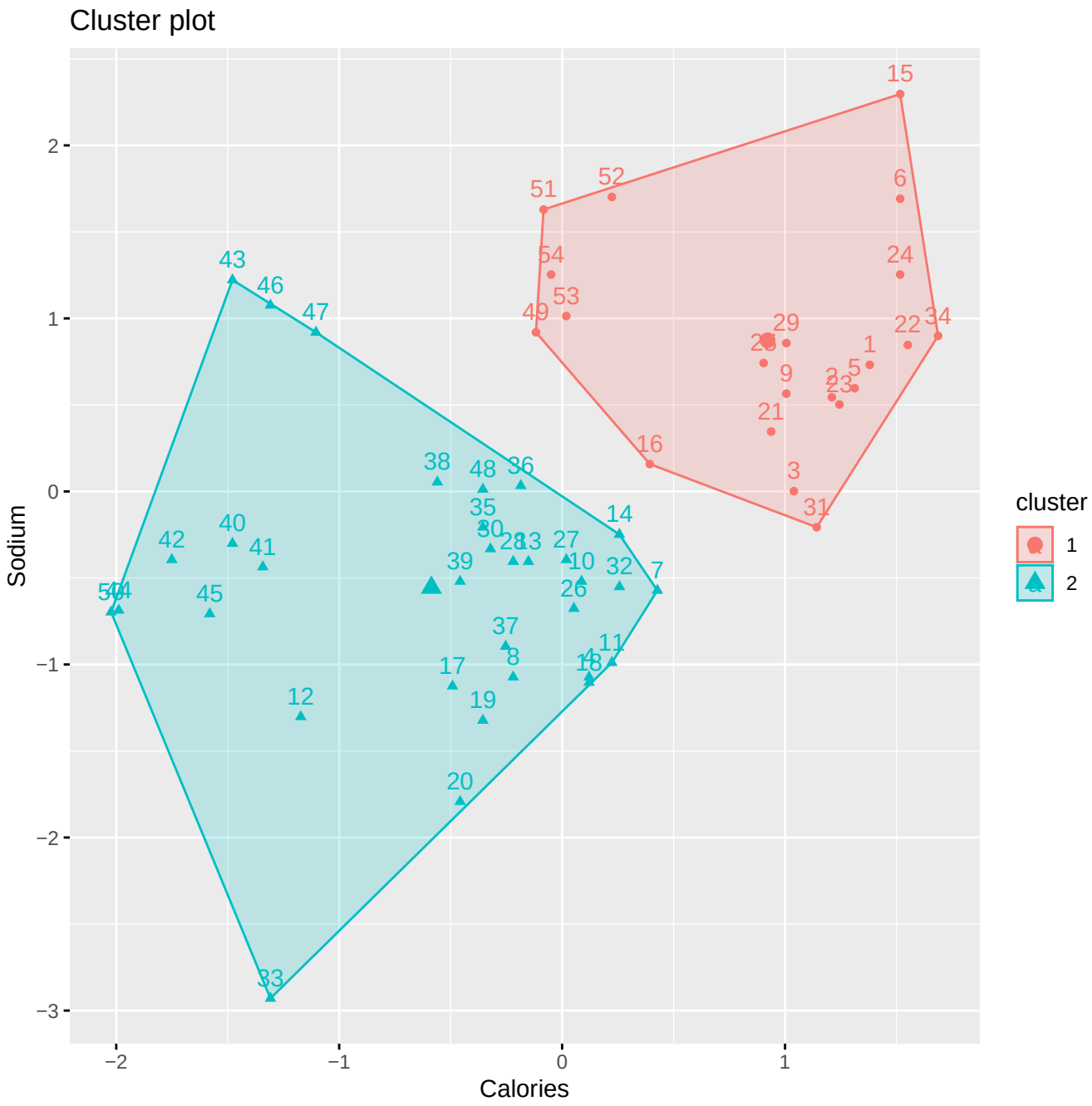

```
ggplot( data = hd5, aes(x = Calories, y = Sodium)) + geom_point(aes(color = cluster)) + theme_bw()
```



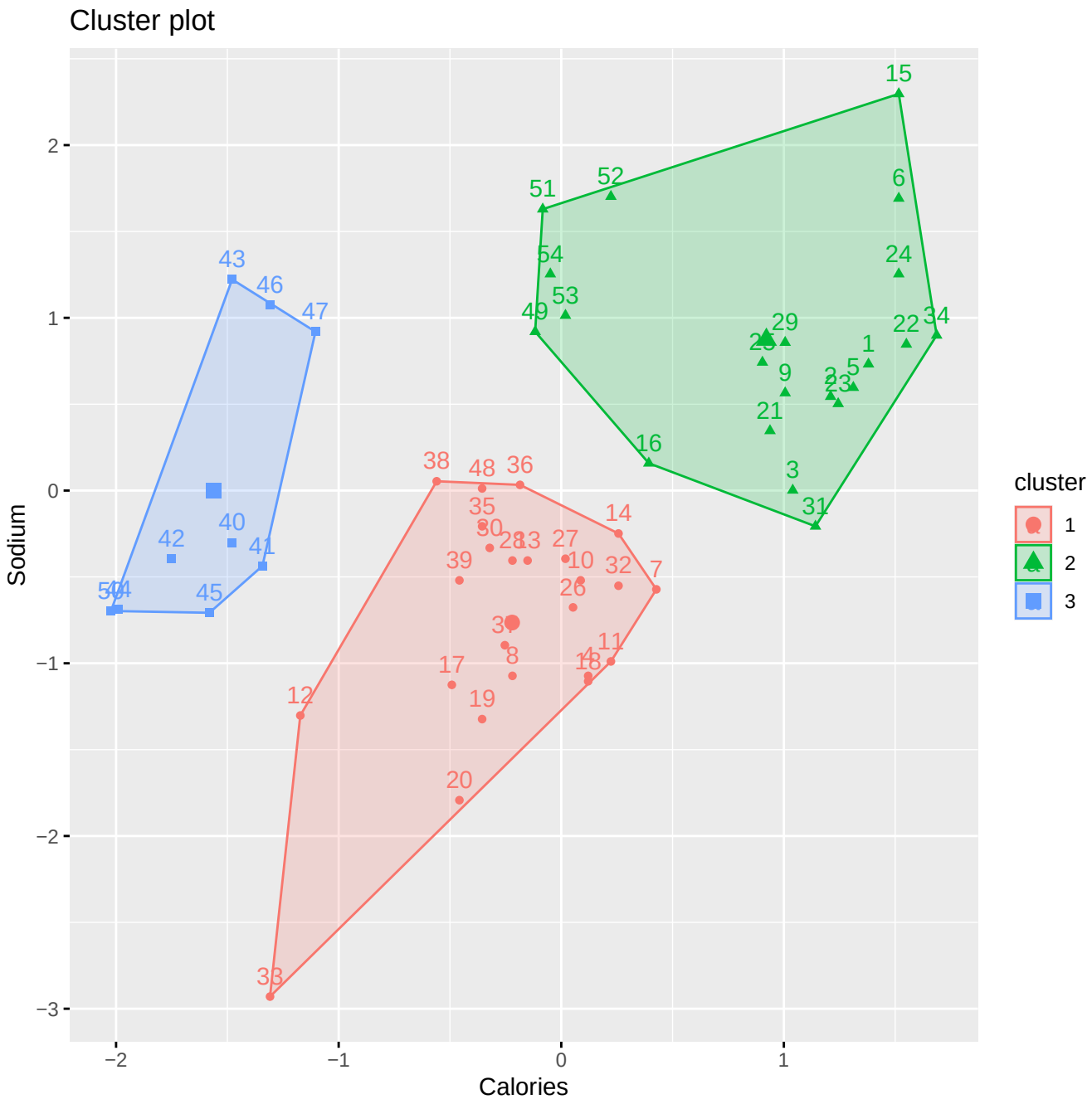
c. Install and eanble the following R libraries: `cluster`, `NbClust`, `factoextra`. Then use the `fviz_cluster()` function to vizualize the clusters you made in part (a). Here is an example of how to use it.

```
fviz_cluster(km.result, data = hotdogs.scaled)
```

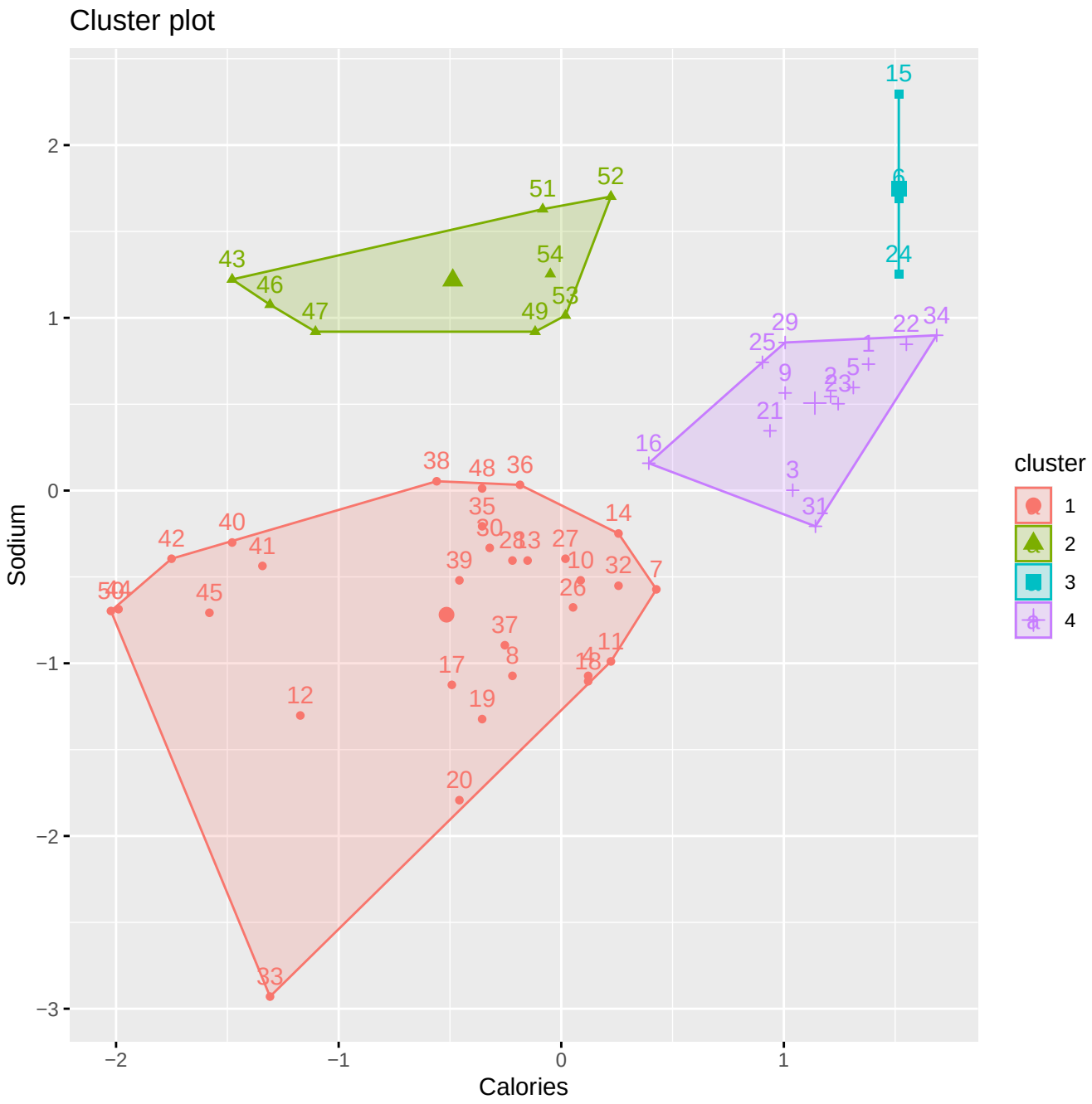
```
library("NbClust")
fviz_cluster(km2, data = hotdogs.scaled)
```



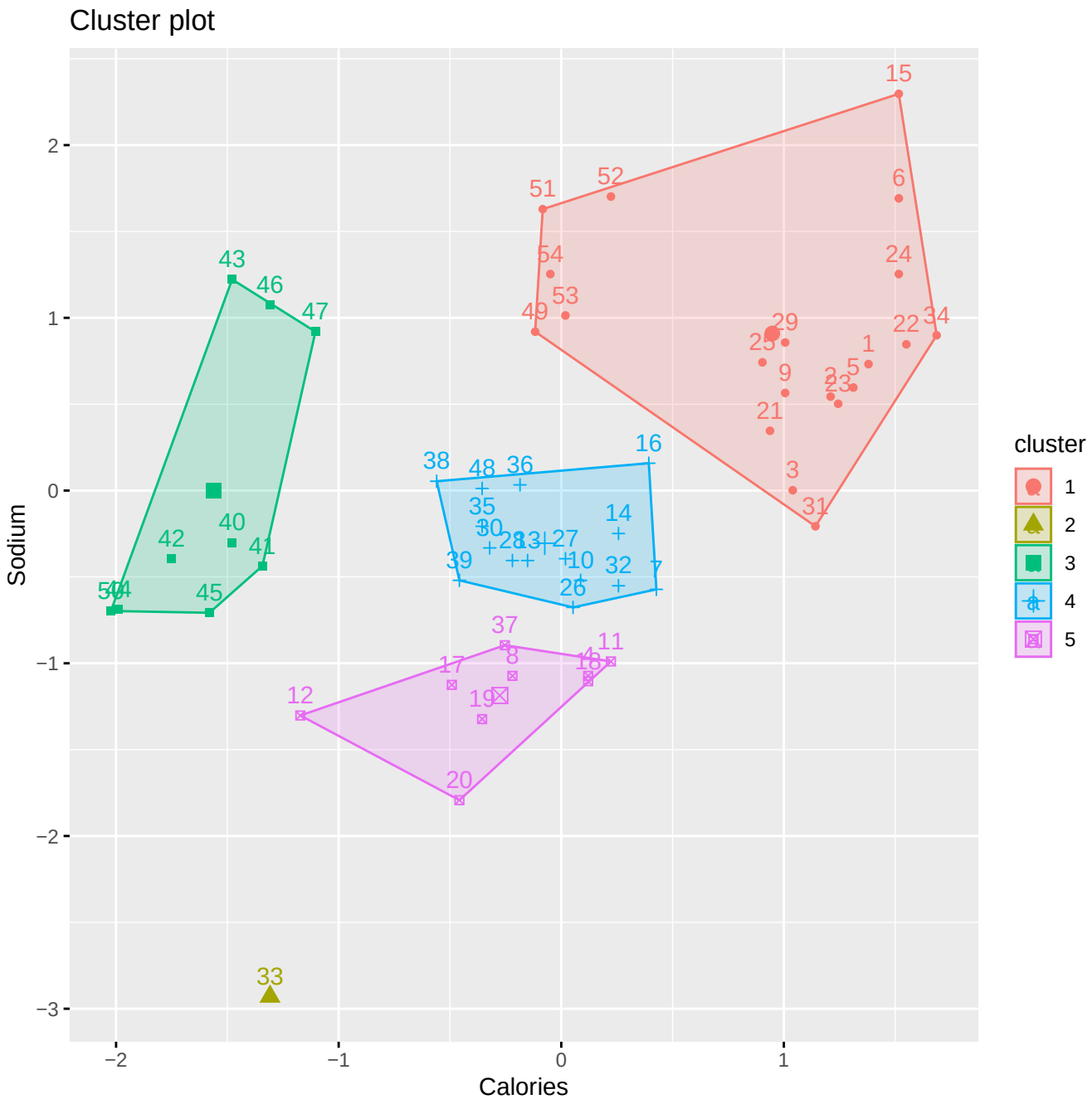

```
fviz_cluster(km3, data = hotdogs.scaled)
```




```
fviz_cluster(km4, data = hotdogs.scaled)
```




```
fviz_cluster(km5, data = hotdogs.scaled)
```



Determining the optimal number of clusters.

Recall that the basic idea behind partitioning methods, such as k-means clustering, is to define clusters such that the total within-cluster variation or total within-cluster sum of squares is minimized: That is,

$$minimize\left(\sum_{i=1}^{k} W(C_k)\right)$$

There are a number of methods that we can use to determine the optimal number of clusters that should be used. One such method is called the Elbow Method which plots the total within-cluster sum of squares versus number of clusters.

We can obtain the total within-cluster sum of squares using `km.result$tot.withinss`.

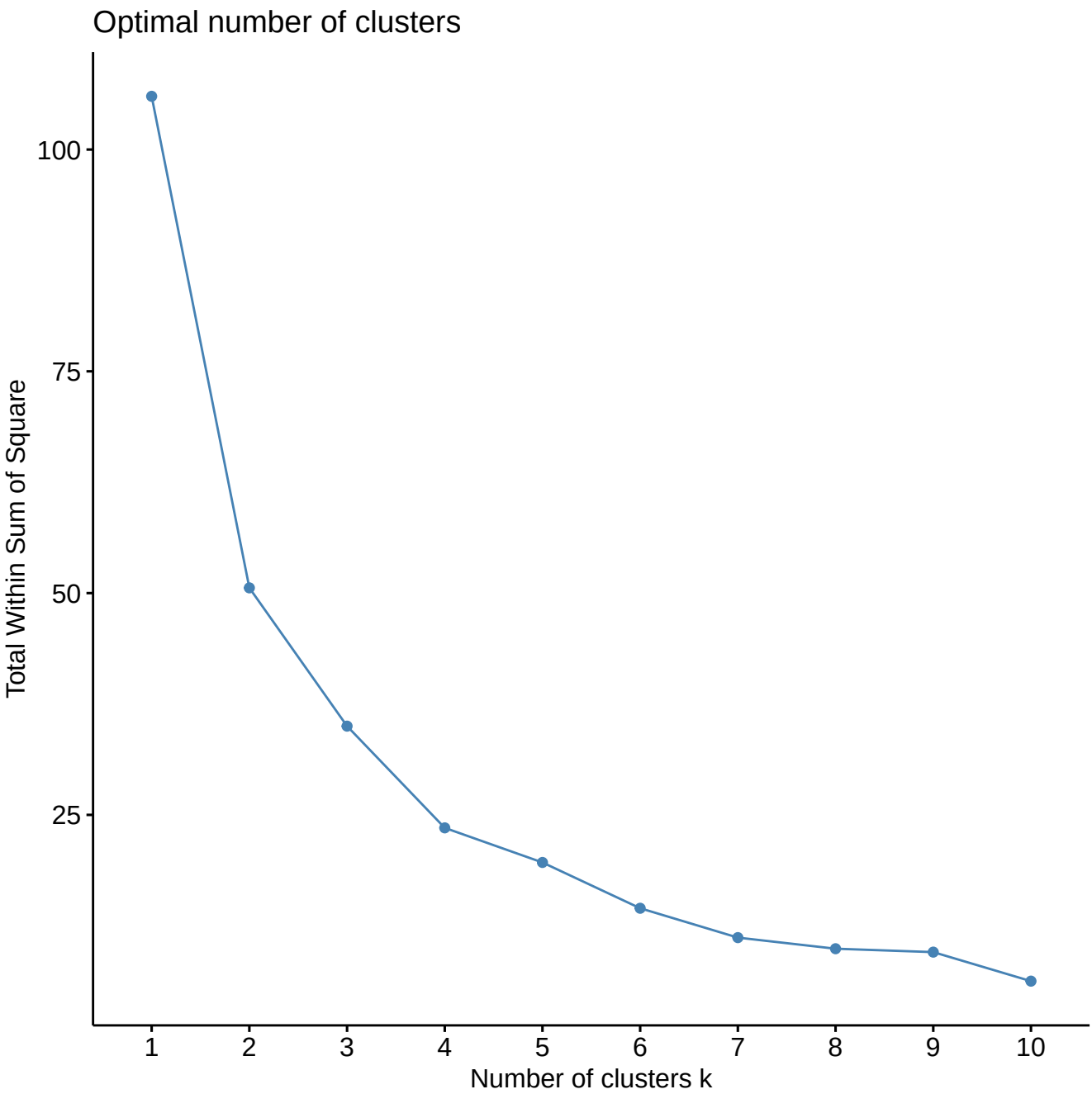
The elbow method suggest that you find the "elbow" of this plot, where the total within-cluster sum of squares essentially stops reducing signficantly as the number of clusters grows.

Thankfully I can save you from this the long way by telling you about a function that can do this automatically.

d. Use the function `fviz_nbclust(hotdogs.scaled, kmeans, method = "wss")` to produce a plot using the Elbow method. Using this plot, determine how many clusters you think should be used.

Discussion: There are other methods for determining the optimal number of clusters that are more sophisticated such as the average silhouette method and the gap statistic method, but these are beyond the scope of this course.

```
fviz_nbclust(hotdogs.scaled, kmeans, method = "wss")
```



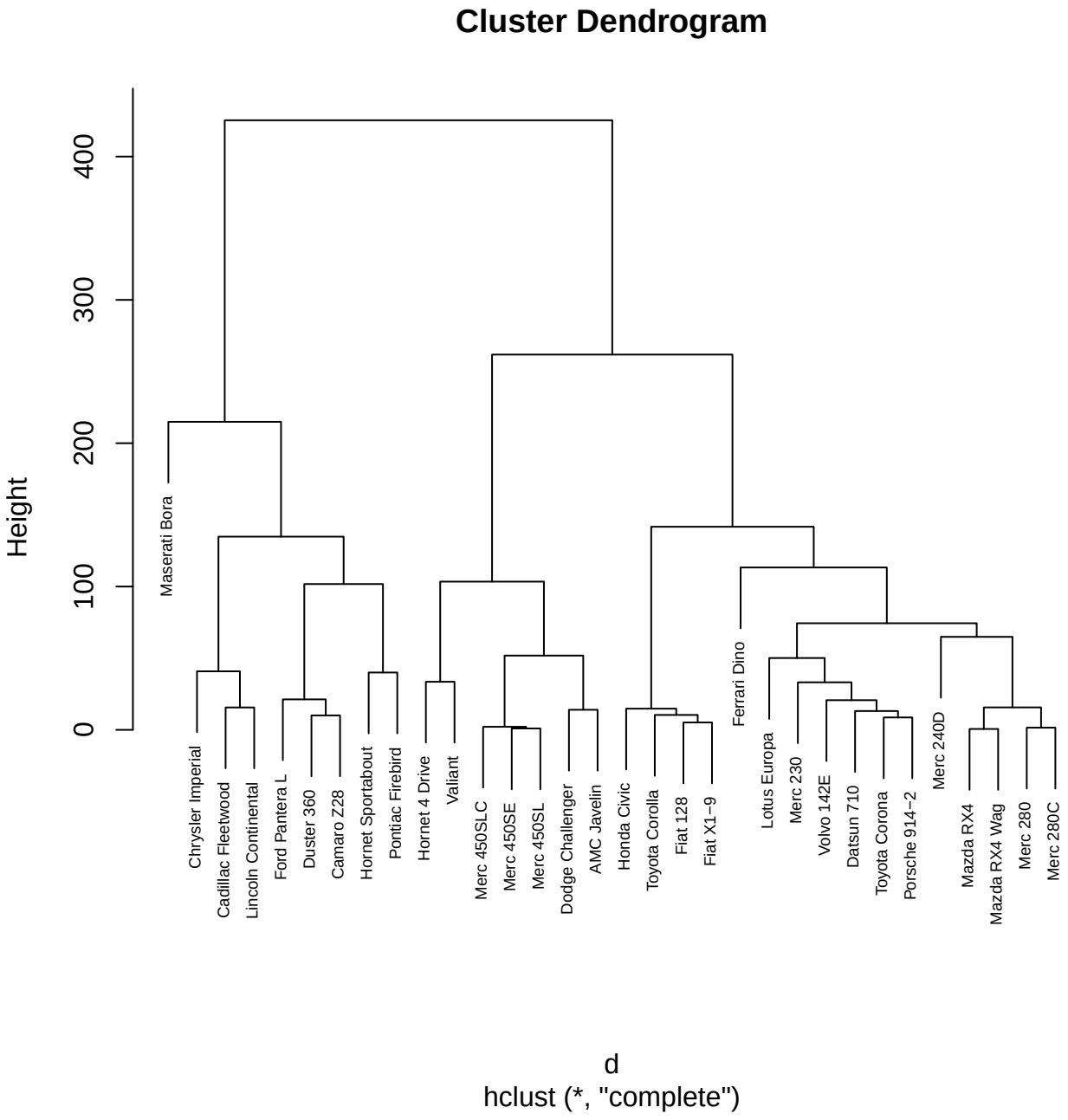
I think either 4 or 6 clusters should be used. In the graph above, the difference between 4 and 5 is very small, but 5 and 6 is relatively large.

Problem 2: [50 pts] Hierarchical Clustering.

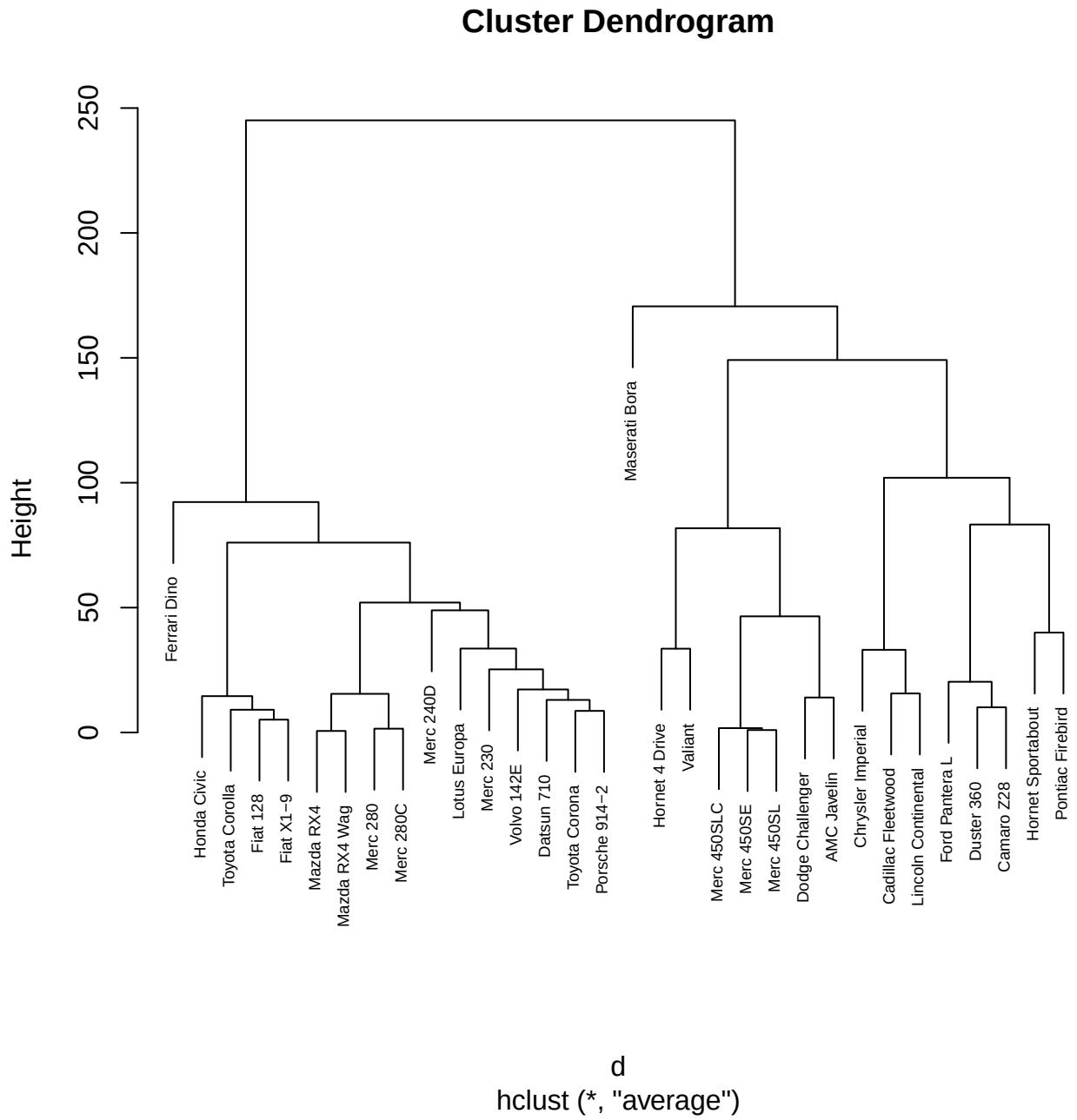
Consider the `mtcars` dataset.

a. Using Euclidean distance as the dissimilarity measure, perform hierarchical clustering on the data, with (i) Complete Linkage, (ii) Average Linkage, and (iii) Single Linkage. Don't forget that you need to scale the data and compute the distances before using the `hclust()` function.

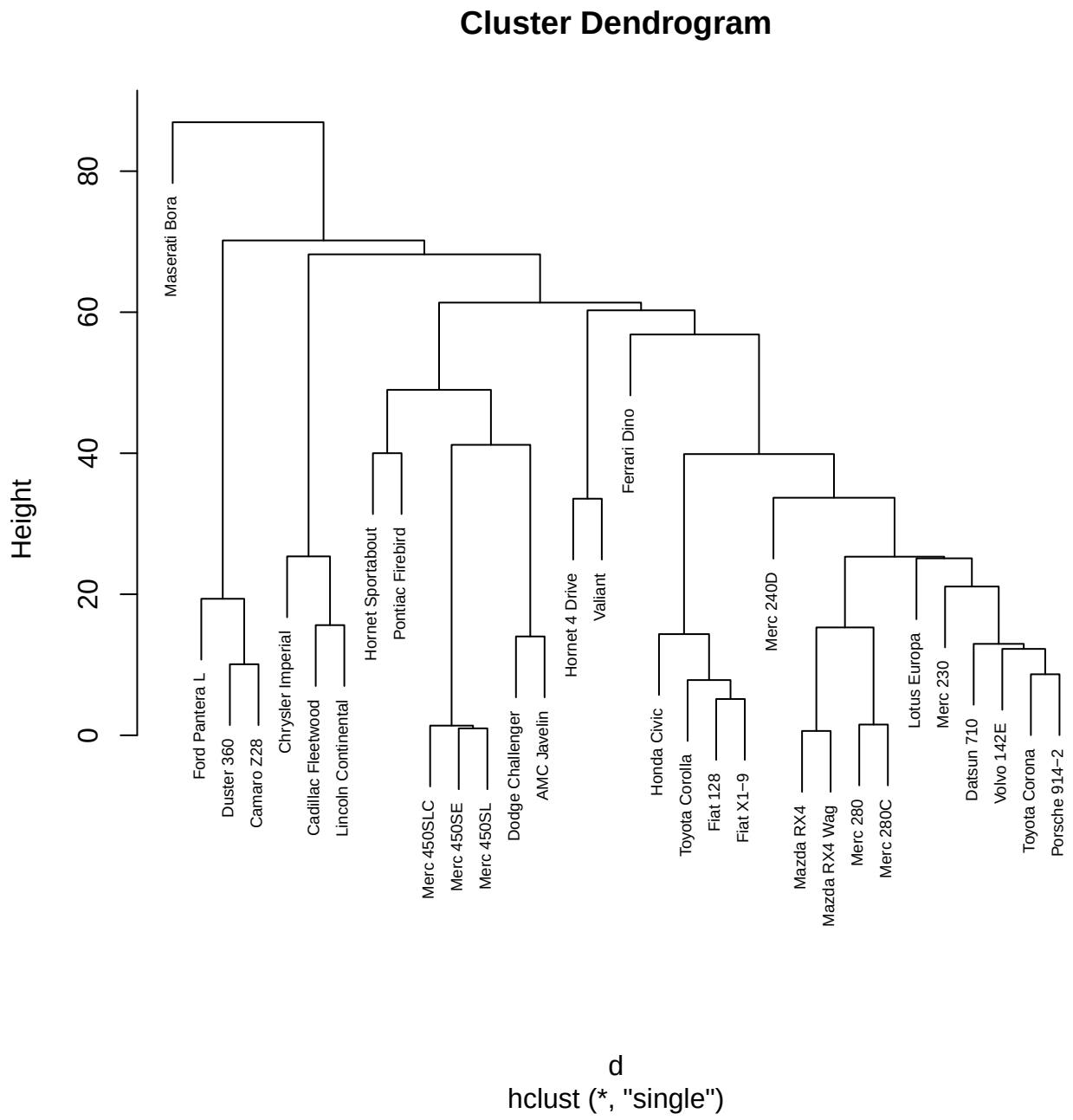
```
data(mtcars)
cars <- na.omit(mtcars)
d <- dist(cars, method = "euclidean")

hcl.com <- hclust(d, method = "complete")

plot(hcl.com, cex = 0.6)
```



```
hcl.avg <- hclust(d, method = "average")

plot(hcl.avg, cex = 0.6)
```




```
hcl.single <- hclust(d, method = "single")

plot(hcl.single, cex = 0.6)
```



b. For all three methods in (a), cut the hierarchical clustering tree at 4 clusters and report the two-way table of the car name and the cluster it belongs to.

```
library(ggdendro)

cutree(hcl.com, k = 4)
```

```
          Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
                  1                   1                   1                   2
  Hornet Sportabout             Valiant          Duster 360           Merc 240D
                  3                   2                   3                   1
          Merc 230            Merc 280           Merc 280C          Merc 450SE
                  1                   1                   1                   2
         Merc 450SL         Merc 450SLC  Cadillac Fleetwood Lincoln Continental
```

```
                2                 2                 3                 3
  Chrysler Imperial          Fiat 128        Honda Civic      Toyota Corolla
                3                 1                 1                 1
      Toyota Corona    Dodge Challenger        AMC Javelin         Camaro Z28
                1                 2                 2                 3
   Pontiac Firebird          Fiat X1-9      Porsche 914-2       Lotus Europa
                3                 1                 1                 1
     Ford Pantera L       Ferrari Dino      Maserati Bora         Volvo 142E
                3                 1                 4                 1
```

```
cutree(hcl.avg, k = 4)
```

```
          Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
                1                 1                 1                 2
  Hornet Sportabout           Valiant         Duster 360          Merc 240D
                3                 2                 3                 1
           Merc 230          Merc 280          Merc 280C         Merc 450SE
                1                 1                 1                 2
        Merc 450SL        Merc 450SLC  Cadillac Fleetwood Lincoln Continental
                2                 2                 3                 3
  Chrysler Imperial          Fiat 128        Honda Civic      Toyota Corolla
                3                 1                 1                 1
      Toyota Corona    Dodge Challenger        AMC Javelin         Camaro Z28
                1                 2                 2                 3
   Pontiac Firebird          Fiat X1-9      Porsche 914-2       Lotus Europa
                3                 1                 1                 1
     Ford Pantera L       Ferrari Dino      Maserati Bora         Volvo 142E
                3                 1                 4                 1
```

```
test <- cutree(hcl.single, k = 4)
```

c. We can plot the dendrograms easily enough by doing the following:

```
plot(hcl.com, labels = rownames(mtcars),
     main = "Cluster Dendrogram (Complete Linkage)")
```

Where the above would plot the dendrogram for the Complete Linkage case. Provide this plot and repeat the above for the other 2 cases from part (a). Alternatively we can use the following library that makes use of `ggplot2`, called `ggdendro`.

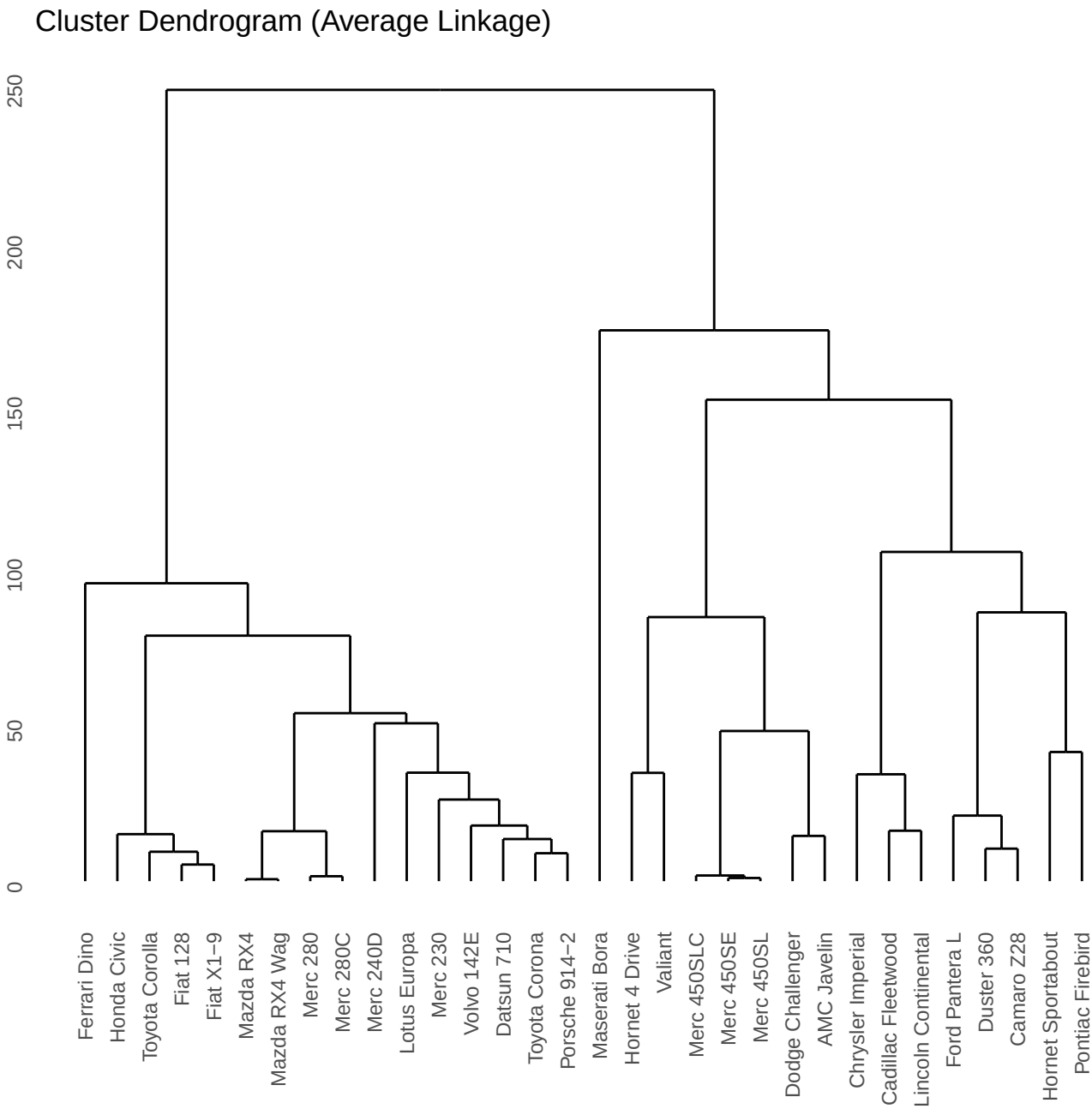
```
# Vertical
ggdendrogram(hcl.com) + labs(title = "Cluster Dendrogram (Complete Linkage)")
# or horizontal with a different theme
ggdendrogram(hcl.com, rotate = T, theme_dendro = F) +
  labs(title = "Cluster Dendrogram (Complete Linkage)")
```

Some alternative tools for plotting & customizing dendrograms can be found here: http://www.sthda.com/english/wiki/beautiful-dendrogram-visualizations-in-r-5-must-known-methods-unsupervised-machine-learning

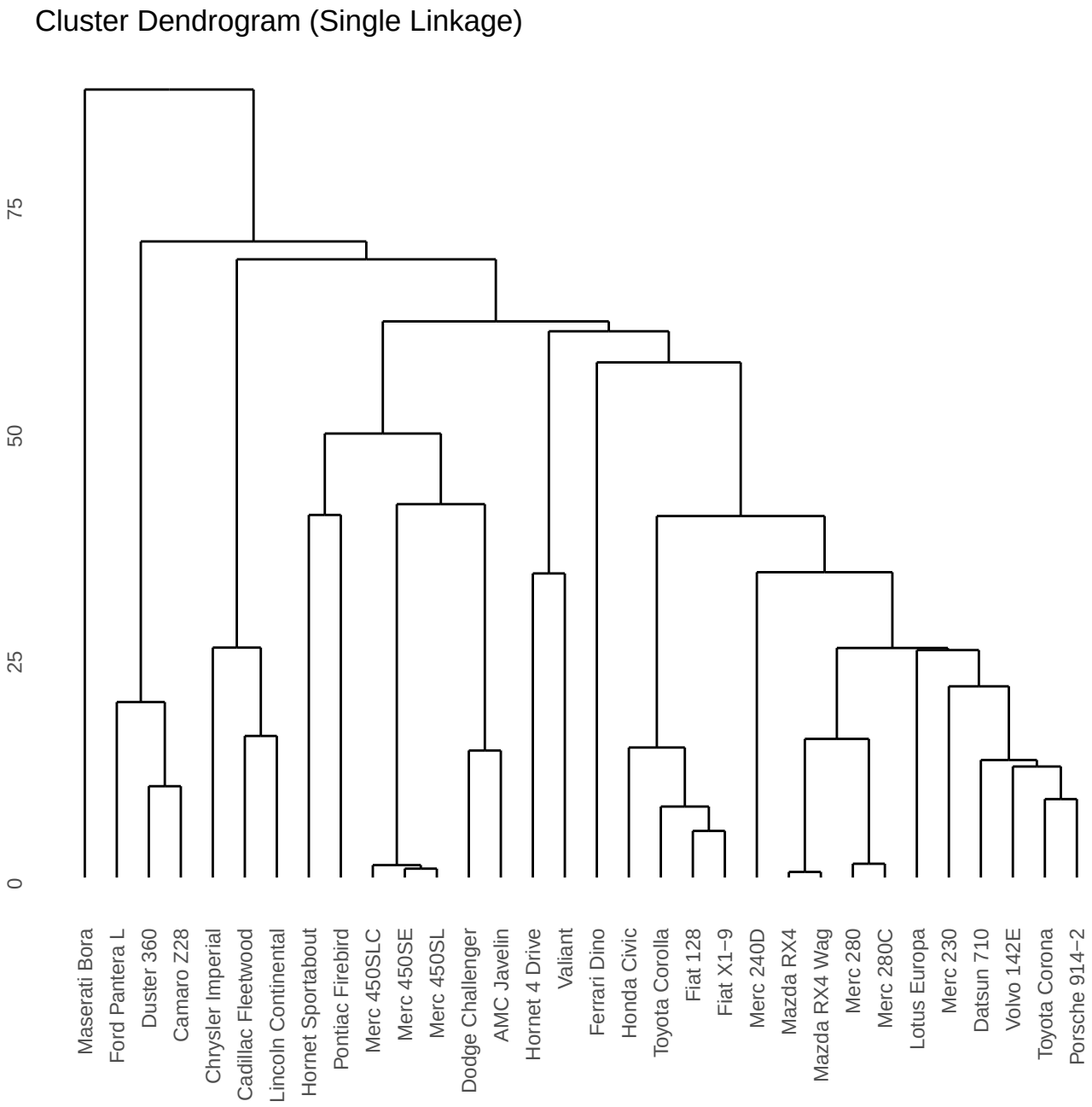
Section 5 has some really cool examples of how to colorize clusters, etc.

Another awesome example can be found here: https://www.r-graph-gallery.com/340-custom-your-dendrogram-with-dendextend/

```
ggdendrogram(hcl.avg) + labs(title = "Cluster Dendrogram (Average Linkage)")
```




```
ggdendrogram(hcl.single) + labs(title = "Cluster Dendrogram (Single Linkage)")
```



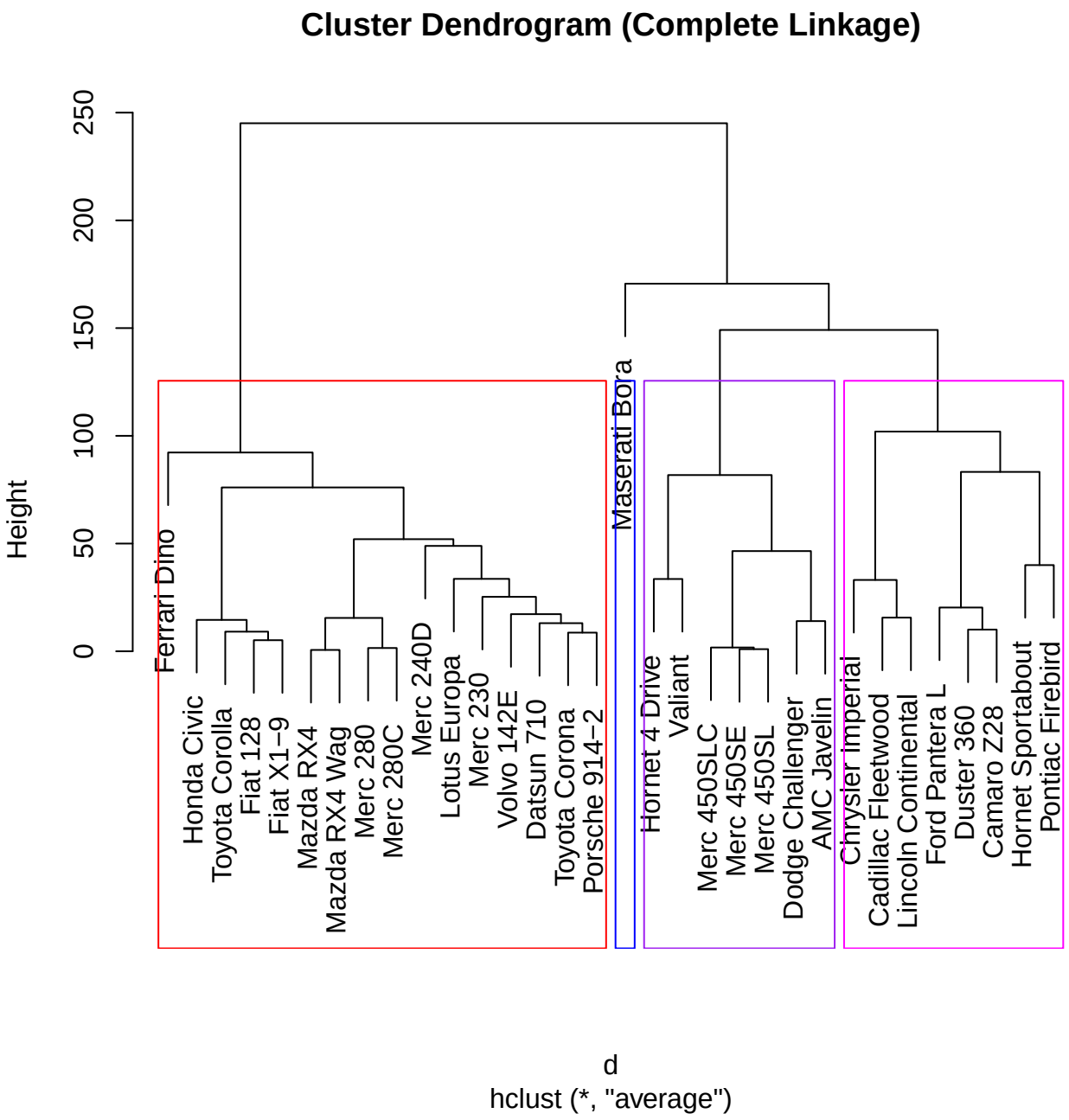
d. Use the elbow method to determine the optimal number of clusters that you should use. This works the same basic way as in problem 1, but the call is slightly different because it needs to use the `hcut()` function (named without the ()) as an option as seen below.

```
fviz_nbclust(mtcars.scaled, hcut, method = "wss")
```

Using four clusters appears to be the best number of clusters for this data set.

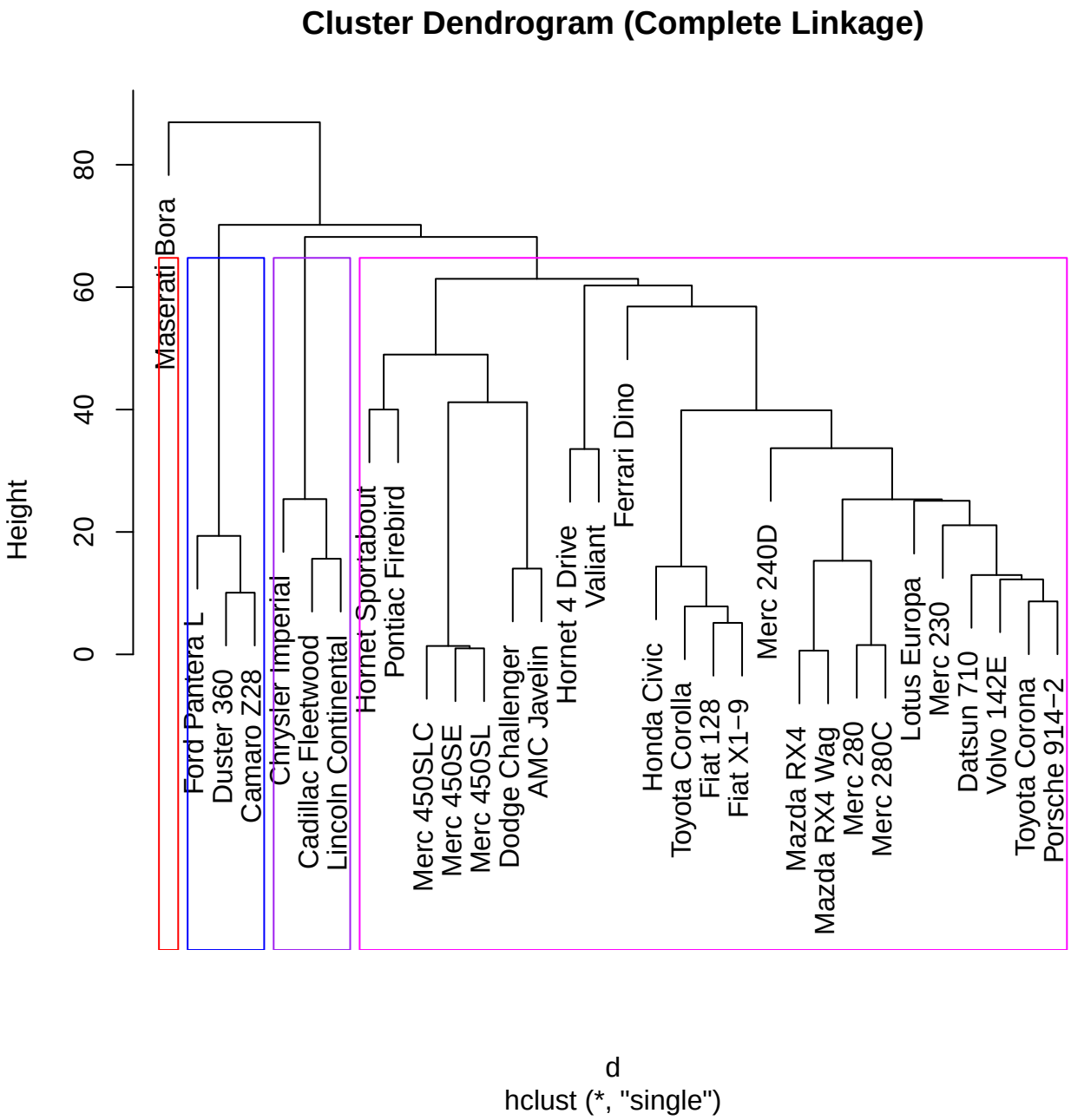
e. Add colored rectangles around the clusters you have determined in part (d) to the dendrogram in part (c). You can simply run the following line after the plot in (c) if you used the first method (other methods required other functions).

```
rect.hclust(mtcars.hclust,
            k = 100, # replace with whatever you decided based upon (d)
            border = border = c("red","blue","purple","magenta")
            )

plot(hcl.avg, labels = rownames(mtcars),
     main = "Cluster Dendrogram (Complete Linkage)")
rect.hclust(hcl.avg,
            k = 4, # replace with whatever you decided based upon (d)
            border = c("red","blue","purple","magenta")
            )
```




```
plot(hcl.single, labels = rownames(mtcars),
     main = "Cluster Dendrogram (Complete Linkage)")
rect.hclust(hcl.single,
            k = 4, # replace with whatever you decided based upon (d)
```

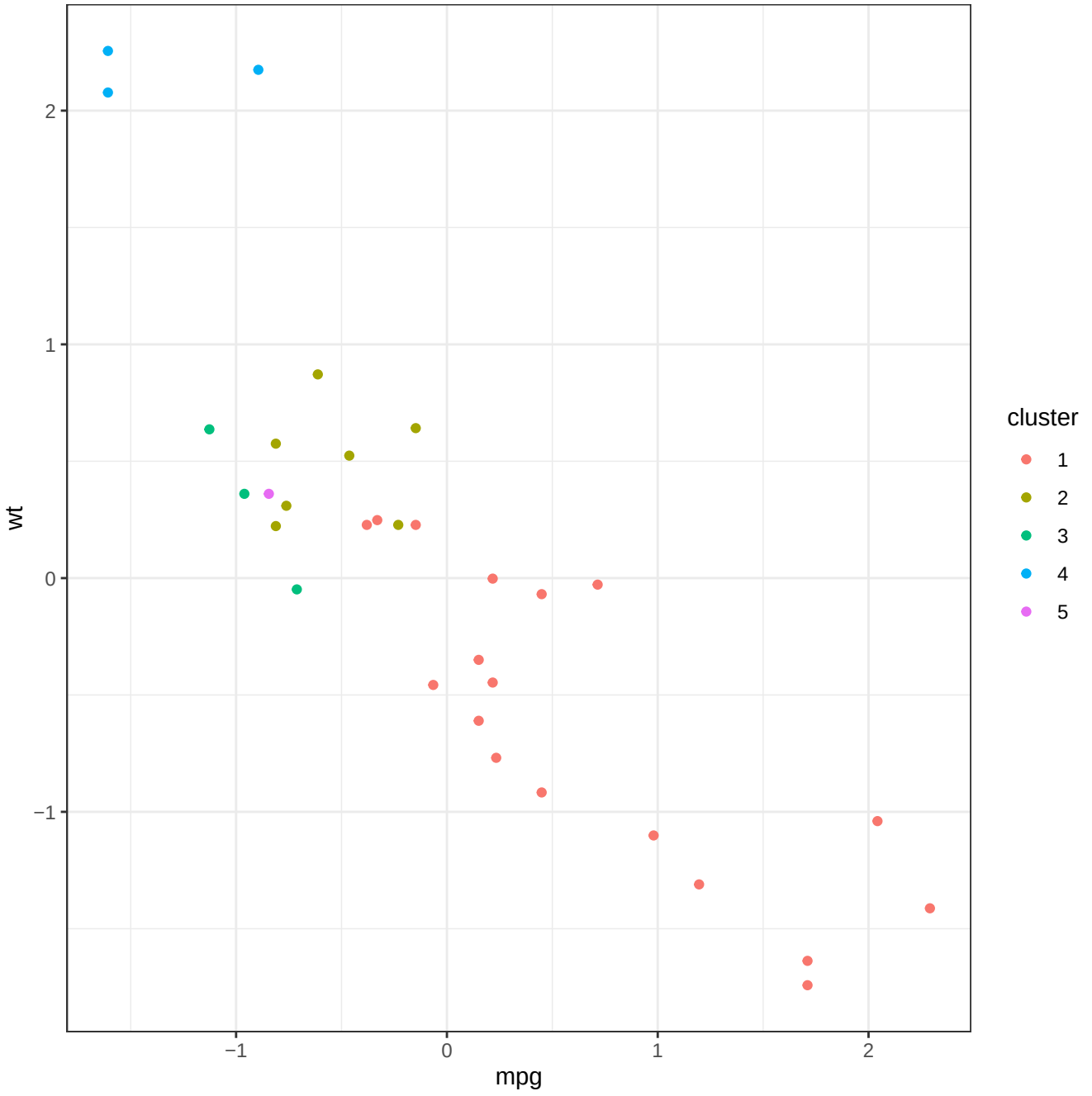
```
border = c("red","blue","purple","magenta")
)
```



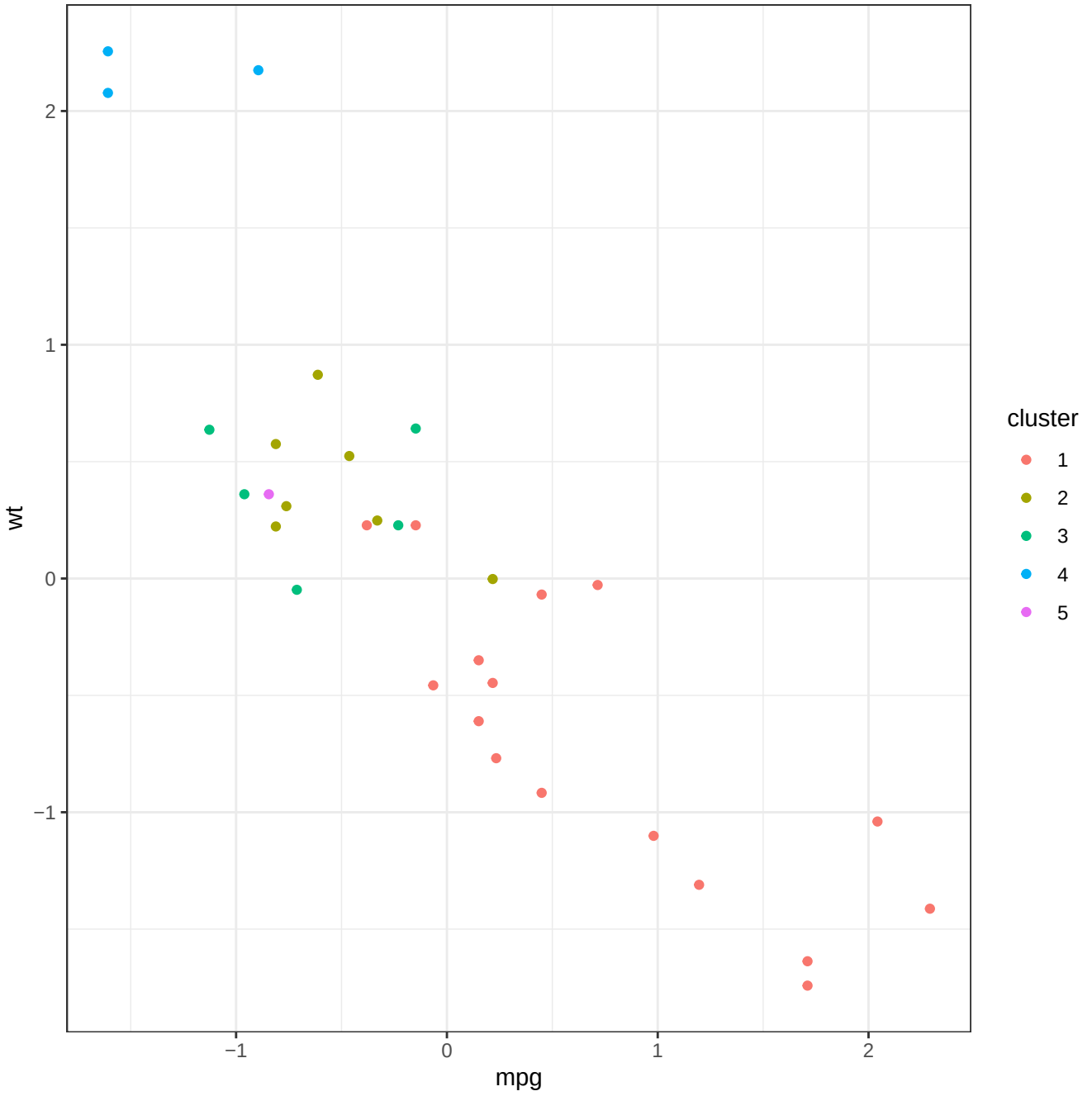
f. Use `cutree()` to obtain the cluster assignments using your decision in (d). Recode this as a factor object. Plot the `mpg` versus `wt` and use the different colors according to your cluster assignment.

```
hcl.single.cut <- cutree(hcl.single, k = 5) %>% as.data.frame()

cars.scaled.single <- cbind(mtcars.scaled, "cluster" = as.factor(hcl.single.cut$.))


single.scaled <- as.data.frame(cars.scaled.single)

single.scaled$cluster <- as.factor(single.scaled$cluster)
```

```
ggplot( data = single.scaled, aes(x = mpg, y = wt)) + geom_point(aes(color = cluster)) + theme_bw()
```




```
hcl.avg.cut <- cutree(hcl.avg, k = 5) %>% as.data.frame()

cars.scaled.avg <- cbind(mtcars.scaled, "cluster" = as.factor(hcl.avg.cut$.))


avg.scaled <- as.data.frame(cars.scaled.avg)

avg.scaled$cluster <- as.factor(avg.scaled$cluster)

ggplot( data = avg.scaled, aes(x = mpg, y = wt)) + geom_point(aes(color = cluster)) + theme_bw()
```



Discussion: While you can see some clustering that makes sense, some of the clusters seem to intermixed with each other. Remember, the clustering was determined not using only the `mpg` and `wt` variables alone, but using all of the information from all of the variables in the dataset. So the clusters are formed in multidimensional space. This can be difficult to visualize when the number of dimensions is bigger than 3.

One solution is to rotate the multidimensional variable coordinate system into the principal component coordinate system as we will show in the next problem. This means that if most of the variation is in the first 2 principal components, it might be possible to more easily see the clusters in higher dimensional space using the lower dimensional representation.

Problem 3: [15 pts Extra Credit] PCA + Hierarchical Clustering.

Consider the `mtcars` dataset once again.

a. Use PCA to rotate the observations from the `mtcars` dataset into a new coordinate system using the principal components. Remember that you have to do either `scale. = TRUE` if using `prcomp()` or `cor = TRUE` if using `princomp()`. We want the component scores which will either be `pca.result$x` if you used `prcomp()` or `pca.result$scores` if you use `princomp()`.

b. For the principal component "variables" for the cars (which are the component scores), use the hierarchical clustering techniques that you learned in Problem 2. Use this to determine the optimal number of clusters, show the dendrogram and put a box around these clusters. **Use Complete Linkage only.** How does this compare with your answer result from the previous problem when you used complete linkage on the original variables?

c. Using `k = 4` for the number of clusters, plot `PC2` versus `PC1` and use different colors to show the clusters. You will need to use `cutree()` to get the cluster assignments. Can you see the clusters a bit better than you did compared to plotting `mpg` versus `wt` in the previous problem?
