Problem_3 - Riya

```
set.seed(15072)

library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(rpart)
library(randomForest)
```

```
## randomForest 4.7-1.2
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ lubridate 1.9.3     ✔ tibble    3.2.1
## ✔ purrr     1.0.2     ✔ tidyr     1.3.1
## ✔ readr     2.1.5
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ randomForest::combine() masks dplyr::combine()
## ✖ dplyr::filter()         masks stats::filter()
## ✖ dplyr::lag()            masks stats::lag()
## ✖ purrr::lift()           masks caret::lift()
## ✖ randomForest::margin()  masks ggplot2::margin()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts
to become errors
```

```r
library(readr)
library(ggplot2)
library(olsrr)
```

```
##
## Attaching package: 'olsrr'
##
## The following object is masked from 'package:datasets':
##
##     rivers
```

```r
library(rpart.plot)
library(gbm)
```

```
## Loaded gbm 2.2.2
## This version of gbm is no longer under development. Consider transitioning to gbm3, h
ttps://github.com/gbm-developers/gbm3
```

```r
library(Metrics)
```

```
##
## Attaching package: 'Metrics'
##
## The following objects are masked from 'package:caret':
##
##     precision, recall
```

a) Cluster the data using hierarchical clustering, using the “Ward D2” measure of cluster dissimilarity. Then, use the cutree function to cut the dendrogram tree to 3 clusters and extract cluster assignments. Report the cluster sizes and comment on the characteristics of each cluster.

```
df <- read.csv("autoSurvey2024.csv")

# convert categorical/binary variables to factors
df$gender <- factor(df$gender, levels = c(0, 1))
df$household <- factor(df$household, levels = c(0, 1))

# one-hot encode all variables so they're still included in the clustering
df_encoded <- model.matrix(~ . - 1, data = df)
df_encoded <- as.data.frame(df_encoded)

# standardize
df_scaled <- scale(df_encoded)

# euclidean distance matrix
distance_matrix <- dist(df_scaled, method = "euclidean")

# hierarchical clustering
hc_result <- hclust(distance_matrix, method = "ward.D2")

# cut dendrogram to form 3 clusters
cluster_assignments <- cutree(hc_result, k = 3)

# add cluster labels to the original data
df$cluster <- factor(cluster_assignments)

print(table(df$cluster))
```

```
##
##   1   2   3
## 639  49 105
```

```
print(aggregate(. ~ cluster, data = df, FUN = mean))
```

```
##   cluster driving_properties  interior technology   comfort reliability
## 1       1          0.7339593 0.2190923  0.4585290 0.3536776   0.4272300
## 2       2          0.5510204 0.1428571  0.3265306 0.4285714   0.4285714
## 3       3          0.4857143 0.4571429  0.3714286 0.5809524   0.2952381
##    handling     power consumption    sporty    safety gender household
## 1 0.3098592 0.5023474   0.2550861 0.4147105 0.3818466      1  1.575900
## 2 0.2857143 0.4081633   0.1224490 0.3469388 0.5102041      2  1.428571
## 3 0.1809524 0.4476190   0.3238095 0.3904762 0.4571429      1  1.761905
##        space
## 1 0.00312989
## 2 0.06122449
## 3 1.00000000
```

Hierarchical clustering using Ward.D2 revealed three distinct customer segments based on preferences for vehicle features. Cluster 1, the largest group with 639 customers, shows moderate interest in driving performance (driving_properties = 0.73, power = 0.50) and sportiness (0.41), but relatively low emphasis on comfort, safety, and space. This segment likely values dynamic driving and style, with a mix of household sizes. Cluster 2, comprising 49 customers, places the highest importance on safety (0.51) and reliability (0.43), with moderate comfort and technology scores. These customers appear to prioritize dependable, well-rounded vehicles, possibly with smaller households. Cluster 3, with 105 customers, stands out for its strong preference for interior features (0.46), comfort (0.58), and space (1.00), suggesting a group focused on practicality and spaciousness, likely from larger households. Compared to earlier models, this segmentation offers clearer differentiation between performance-oriented, safety-conscious, and space-seeking customer profiles.

b) Increase the number of clusters to 4 and analyze them. In what ways does the 4-cluster model differ from the 3-cluster model?

```
# cut dendrogram to form 3 clusters
cluster_assignments_2 <- cutree(hc_result, k = 4)

# add cluster labels to the original data
df$cluster_2 <- factor(cluster_assignments_2)

print(table(df$cluster_2))
```

```
##
##   1   2   3   4
## 484 155  49 105
```

```
print(aggregate(. ~ cluster_2, data = df, FUN = mean))
```

```
##   cluster_2 driving_properties  interior technology   comfort reliability
## 1         1          0.6859504 0.2334711  0.4421488 0.3615702   0.4380165
## 2         2          0.8838710 0.1741935  0.5096774 0.3290323   0.3935484
## 3         3          0.5510204 0.1428571  0.3265306 0.4285714   0.4285714
## 4         4          0.4857143 0.4571429  0.3714286 0.5809524   0.2952381
##     handling     power consumption    sporty    safety gender household
## 1 0.09504132 0.4442149   0.2252066 0.3822314 0.3636364      1  1.547521
## 2 0.98064516 0.6838710   0.3483871 0.5161290 0.4387097      1  1.664516
## 3 0.28571429 0.4081633   0.1224490 0.3469388 0.5102041      2  1.428571
## 4 0.18095238 0.4476190   0.3238095 0.3904762 0.4571429      1  1.761905
##        space cluster
## 1 0.00000000       1
## 2 0.01290323       1
## 3 0.06122449       2
## 4 1.00000000       3
```

Expanding to a four-cluster model reveals a more refined segmentation of customer preferences, improving upon the broader groupings in the three-cluster version. Cluster 1, now the largest with 484 customers, shows moderate interest in driving properties (0.69), technology (0.44), and power (0.44), but low emphasis on handling (0.10) and space (0.00), suggesting a balanced segment that values performance without prioritizing spaciousness or agility. Cluster 2, with 155 customers, isolates a distinctly sporty and performance-driven group, with the highest scores in driving properties (0.88), power (0.68), and handling (0.98), along with elevated interest in technology and sportiness—indicating a premium segment focused on dynamic driving and advanced features. Cluster 3, comprising 49 customers, emphasizes safety (0.51) and reliability (0.43), with moderate comfort and low consumption, reflecting practical buyers who prioritize dependability. Cluster 4, with 105 customers, stands out for its strong preference for interior features (0.46), comfort (0.58), and space (1.00), suggesting a group focused on spaciousness and comfort, likely from larger households. Compared to the three-cluster model, the four-cluster solution offers clearer separation as clusters 1 and 2 form from the original cluster 1.

c) Now cluster the data using the k-means algorithm, using 3 and 4 clusters. Set the seed to 15072 and max.iter to 100, and extract cluster assignments and compare them to those obtained

with hierarchical clustering.

```
# k-means w 3 clusters
set.seed(15072)
kmeans_3 <- kmeans(df_scaled, centers = 3, iter.max = 100)

# k-means w 4 clusters
set.seed(15072)
kmeans_4 <- kmeans(df_scaled, centers = 4, iter.max = 100)

# comparisons
df$kmeans3 <- factor(kmeans_3$cluster)
df$kmeans4 <- factor(kmeans_4$cluster)

cat("K-means (3 clusters):\n")
```

```
## K-means (3 clusters):
```

```
print(table(df$kmeans3))
```

```
##
##   1   2   3
## 289 259 245
```

```
cat("\nK-means (4 clusters):\n")
```

```
##
## K-means (4 clusters):
```

```
print(table(df$kmeans4))
```

```
##
##   1   2   3   4
## 177 203 202 211
```

```
cat("\nHierarchical clustering (3 clusters):\n")
```

```
##
## Hierarchical clustering (3 clusters):
```

```
print(table(df$cluster))  # assuming df$cluster already contains hierarchical labels
```

```
##
##   1   2   3
## 639  49 105
```

```
cat("\nHierarchical clustering (4 clusters):\n")
```

```
##
## Hierarchical clustering (4 clusters):
```

```
print(table(df$cluster_2))  # assuming df$cluster already contains hierarchical labels
```

```
##
##   1   2   3   4
## 484 155  49 105
```

```
cat("\nCross-tab: Hierarchical vs K-means (3 clusters):\n")
```

```
##
## Cross-tab: Hierarchical vs K-means (3 clusters):
```

```
print(table(Hierarchical = df$cluster, KMeans3 = df$kmeans3))
```

```
##             KMeans3
## Hierarchical   1   2   3
##            1 263 207 169
##            2   0  27  22
##            3  26  25  54
```

```
cat("\nCross-tab: Hierarchical vs K-means (4 clusters):\n")
```

```
##
## Cross-tab: Hierarchical vs K-means (4 clusters):
```

```
print(table(Hierarchical = df$cluster, KMeans4 = df$kmeans4))
```

```
##             KMeans4
## Hierarchical   1   2   3   4
##            1 152 157 133 197
##            2   0  27  22   0
##            3  25  19  47  14
```

To compare clustering methods, we applied both hierarchical clustering and k-means clustering to the standardized data frames. We made cross-tabulations to show that hierarchical cluster 1, which contains the majority of customers, is broadly distributed across all k-means clusters. It is particularly in the 3-cluster mapping, where it maps to k-means clusters 1 (263), 2 (207), and 3 (169). This tells us that k-means partitions the major hierarchical group into finer subgroups. Hierarchical cluster 2 aligned more cleanly with k-means clusters 2 and 3, while cluster 3 showed moderate overlap with all k-means groups. In the 4-cluster mapping, hierarchical cluster 1 again split across all k-means clusters, confirming that k-means offered more granular segmentation. Overall, k-means seems to give us more precise mappings by refining the broader hierarchical groupings by divvying up the largest cluster. This could help us uncover customer subtypes and improve targeting strategies.

d) Based on your responses in parts a, b, and c, in what ways are the clusters similar and different across the two clustering approaches (hierarchical and k-means)?

Across hierarchical clustering and k-means, several consistent patterns emerge, but the methods also reveal distinct segmentation strategies. Hierarchical clustering groups similar things together one by one after treating them each as individual entities, while k-means tries to find the points most similar to an establish central icon. Both approaches identify a cluster of customers who prioritize comfort, reliability, and safety, as well as a group focused on interior features and space, suggesting agreement on core consumer archetypes. However, hierarchical clustering tends to preserve broader groupings—especially in the 3-cluster model—while k-means more aggressively partitions large clusters into finer subgroups. For example, hierarchical cluster 1, which contains the majority of customers, is split across multiple k-means clusters, indicating that k-means detects latent variation within this dominant segment. In the 4-cluster comparison, k-means isolates a high-performance, tech-savvy group with elevated scores in driving properties, handling, and power—similar to the premium cluster identified in hierarchical clustering. Overall, while both methods capture similar thematic clusters, k-means offers sharper boundaries and more granular segmentation, especially within large, heterogeneous groups.