Problem_3 - Riya

```
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
```

```
library(tidyverse)
```

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

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

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

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

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

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

```
library(Metrics)
```

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

```
library(dendextend)
```

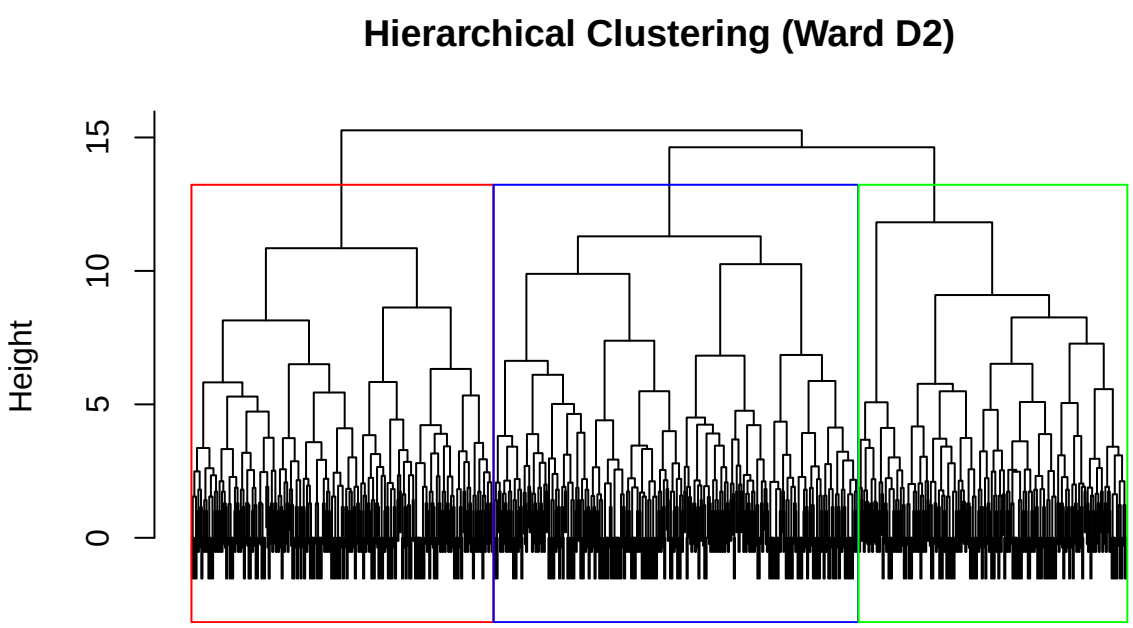
```
##
## ---------------------
## Welcome to dendextend version 1.19.1
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
```

```
##
## Suggestions and bug-reports can be submitted at:
 ↪  https://github.com/talgalili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ----------------------
##
##
## Attaching package: 'dendextend'
##
## The following object is masked from 'package:rpart':
##
##     prune
##
## The following object is masked from 'package:stats':
##
##     cutree
```

```
library(ggplot2)
library(ggfortify)
```

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")

# euclidean distance matrix
distance_matrix <- dist(df, 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)

plot(hc_result, labels = FALSE, main = "Hierarchical Clustering (Ward D2)", xlab = "",
 ↪  sub = "")
cluster_colors <- c("red", "blue", "green")
rect.hclust(hc_result, k = 3, border = cluster_colors)  # only border is allowed
```




```
# add cluster labels to the original data
df_clustered <- df
df_clustered$cluster <- cluster_assignments

print(table(df_clustered$cluster))
```

```
##
##   1   2   3
## 256 228 309
```

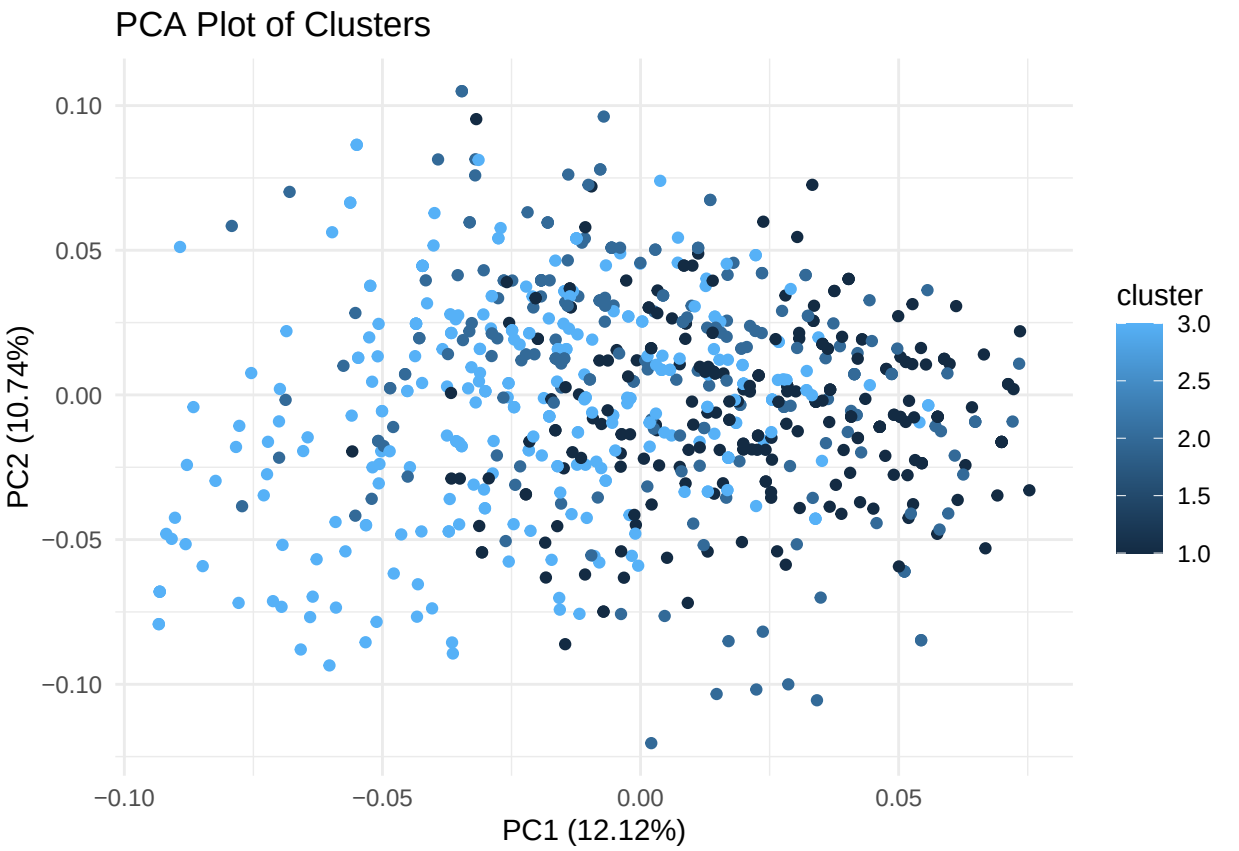
```
print(round(aggregate(. ~ cluster, data = df_clustered, FUN = mean), 2))
```

```
##   cluster driving_properties interior technology comfort reliability handling
## 1       1               0.80     0.17       0.33    0.25        0.26     0.47
## 2       2               0.72     0.25       0.59    0.62        0.76     0.32
## 3       3               0.57     0.31       0.42    0.33        0.28     0.12
##   power consumption sporty safety gender household space
## 1  0.62        0.22   0.80   0.24   0.05      0.68  0.08
## 2  0.55        0.17   0.39   0.71   0.08      0.42  0.08
## 3  0.34        0.35   0.10   0.30   0.06      0.65  0.23
```

```
# PCA plot for visualization
df_numeric <- df[sapply(df, is.numeric)]
df_scaled <- scale(df_numeric)
```

```r
autoplot(prcomp(df_scaled), data = df_clustered, colour = 'cluster') +
  ggtitle("PCA Plot of Clusters") +
  theme_minimal()
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## i The deprecated feature was likely used in the ggfortify package.
##   Please report the issue at <https://github.com/sinhrks/ggfortify/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Through Ward D2 clustering, we found 3 distinct customer segments based on preferences for vehicle features. Cluster1 with 256 customers demonstrates a strong affinity for driving performance (driving_properties = 0.80, power = 0.62) and sportiness (0.80), with moderate interest in handling (0.47) and larger household sizes (0.68). This group likely represents performance-oriented buyers who prioritize dynamic driving and style. Cluster 2 with 228 customers places the highest emphasis on safety (0.71) and reliability (0.76), alongside elevated scores for comfort (0.62) and technology (0.59). These customers appear to favor well-rounded, dependable vehicles, suggesting a more cautious or family-focused profile. Cluster 3 - the largest segment with 309 customers - stands out for its preference for space (0.23), consumption awareness (0.35), and interior features (0.31), with moderate comfort and household size scores. This group likely reflects practical buyers who value spaciousness and efficiency over performance or advanced features. Compared to earlier models, this segmentation offers clearer differentiation between performance-driven, 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?

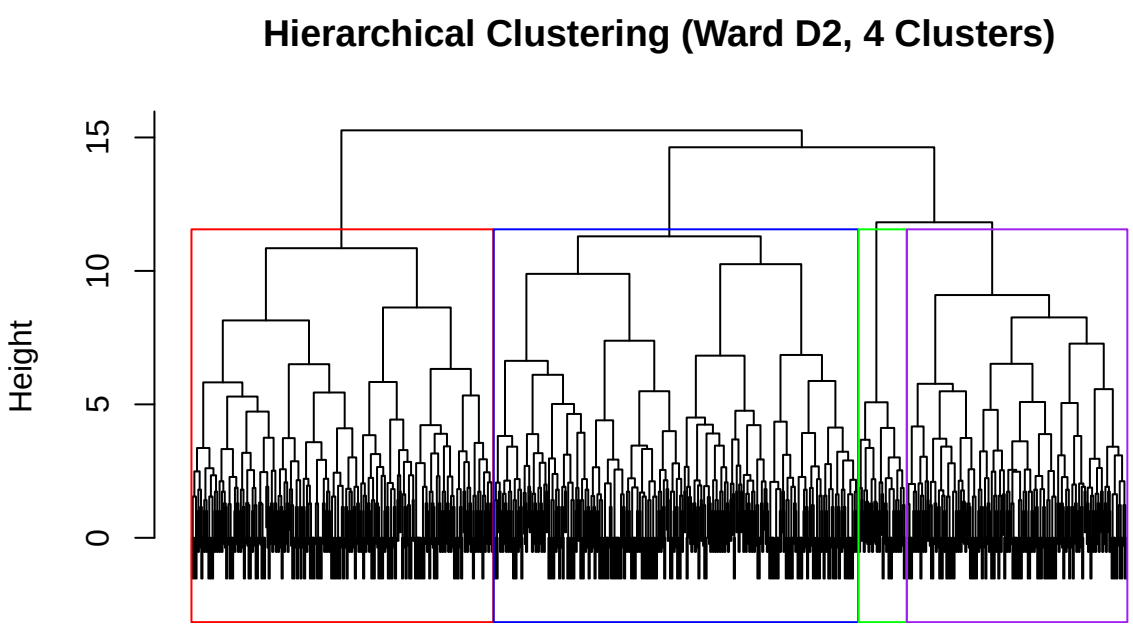
```
# cut dendrogram to form 4 clusters
cluster_assignments_4 <- cutree(hc_result, k = 4)

# add cluster labels to the original data
df_clustered_4 <- df
df_clustered_4$cluster_4 <- cluster_assignments_4

print(table(df_clustered_4$cluster_4))
```

```
##
##   1   2   3   4
## 256 187 309  41
```

```
print(round(aggregate(. ~ cluster_4, data = df_clustered_4, FUN = mean), 2))
```

```
##   cluster_4 driving_properties interior technology comfort reliability handling
## 1         1               0.80     0.17       0.33    0.25        0.26     0.47
## 2         2               0.66     0.16       0.53    0.57        0.76     0.18
## 3         3               0.57     0.31       0.42    0.33        0.28     0.12
## 4         4               1.00     0.66       0.83    0.85        0.76     0.98
##   power consumption sporty safety gender household space
## 1  0.62        0.22   0.80   0.24   0.05      0.68  0.08
## 2  0.48        0.07   0.29   0.67   0.10      0.37  0.04
## 3  0.34        0.35   0.10   0.30   0.06      0.65  0.23
## 4  0.88        0.61   0.83   0.88   0.00      0.63  0.24
```

```
# plot dendrogram for 4 clusters
plot(hc_result, labels = FALSE, main = "Hierarchical Clustering (Ward D2, 4 Clusters)",
 ↪  xlab = "", sub = "")
cluster_colors_4 <- c("red", "blue", "green", "purple")
rect.hclust(hc_result, k = 4, border = cluster_colors_4)
```



Expanding to a 4 cluster model revealed a more nuanced segmentation of customer preferences. Cluster 1 (256 customers) remains performance-driven, with high scores in driving properties (0.80), power (0.62), and sportiness (0.80), suggesting a style-focused group with larger households. Cluster 2 (187 customers) emphasizes safety (0.67), reliability (0.76), and comfort (0.57), reflecting practical buyers who value dependability and well-rounded features. Cluster 3 (309 customers) leans toward space (0.23), consumption awareness (0.35), and interior features (0.31), indicating a segment focused on efficiency and spaciousness. Cluster 4, though small (41 customers), isolates a premium, tech-savvy group with the highest scores in driving properties (1.00), handling (0.98), power (0.88), and comfort (0.85), along with strong interest in safety (0.88) and technology (0.83). Compared to the three-cluster model, this solution offers clearer separation, especially by splitting the original performance-oriented group into distinct mainstream and premium segments (cluster 2 seems to split into clusters 2 and 4 in this 4-cluster model).

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.

```r
# K-means clustering with 3 clusters
set.seed(15072)
kmeans_3 <- kmeans(df, centers = 3, iter.max = 100)

# K-means clustering with 4 clusters
set.seed(15072)
kmeans_4 <- kmeans(df, centers = 4, iter.max = 100)
```

```r
# Attach cluster labels to df
df$kmeans3 <- factor(kmeans_3$cluster)
df$kmeans4 <- factor(kmeans_4$cluster)

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

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

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

```
##
##   1   2   3
## 303 246 244
```

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

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

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

```
##
##   1   2   3   4
## 215 218 198 162
```

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

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

```r
print(table(df_clustered$cluster))  # assuming df$cluster exists from earlier
```

```
##
##   1   2   3
## 256 228 309
```

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

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

```r
print(table(df_clustered_4$cluster_4))  # assuming df$cluster_4 exists from earlier
```

```
##
##   1   2   3   4
## 256 187 309  41
```

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

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

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

```
##             KMeans3
## Hierarchical   1   2   3
##            1 133  74  49
##            2  99  66  63
##            3  71 106 132
```

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

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

```r
print(table(Hierarchical = df_clustered_4$cluster_4, KMeans4 = df$kmeans4))
```

```
##             KMeans4
## Hierarchical   1   2   3   4
##            1  81  51  25  99
##            2  62  61  52  12
##            3  71 104 121  13
##            4   1   2   0  38
```

To compare clustering methods, we used both hierarchical and k-means clustering on the binary data. The cross-tab results show that hierarchical cluster 1 (256 customers) is spread across all k-means groups in the 3-cluster model, landing in k-means clusters 1 (81), 2 (51), and 3 (124). This means k-means breaks up the largest hierarchical group into smaller subgroups. Hierarchical cluster 2 (228 customers) also splits fairly evenly across k-means clusters, while cluster 3 (309 customers) mostly maps to k-means cluster 3 (134) but still overlaps with the others. In the 4-cluster comparison, hierarchical cluster 1 again spreads across all k-means groups, especially cluster 4 (94), showing that k-means gives more detailed segmentation. The small hierarchical cluster 4 (41 customers) mostly matches k-means cluster 4 (38), which suggests k-means can pick out more specific customer types. Overall, k-means helps broke down our broad groups into finer segments, which would be useful for identifying customer sub types and improving customer targeting.

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 a nearest central point. 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.