```r
# Load required libraries
library(tidyverse)      # For data manipulation and visualization
library(cluster)        # For clustering algorithms
library(factoextra)     # For cluster visualization
library(NbClust)        # For determining optimal number of clusters
library(corrplot)       # For correlation visualization
library(scales)         # For formatting scales in plots
library(gridExtra)      # For arranging multiple plots
library(dendextend)     # For dendrogram visualization
library(ggrepel)        # For improved text labeling in plots
library(reshape2)       # For data reshaping

# Set seed for reproducibility
set.seed(123)

# This is crucial for clustering as it ensures all variables contribute equally
cluster_data_scaled <- scale(cluster_data)

# Silhouette Method
# Higher average silhouette width indicates better clustering
silhouette_scores <- function(k) {
  km <- kmeans(cluster_data_scaled, centers = k, nstart = 25)
  ss <- silhouette(km$cluster, dist(cluster_data_scaled))
  mean(ss[, 3])
}

sil_values <- sapply(2:10, silhouette_scores)

silhouette_plot <- ggplot(data.frame(k = 2:10, sil = sil_values), aes(x = k, y = sil)) +
  geom_line() +
  geom_point() +
  labs(title = "Silhouette Method for Optimal k",
       x = "Number of Clusters (k)",
       y = "Average Silhouette Width") +
  theme_minimal()

print(silhouette_plot)

# Gap Statistic Method
# The optimal number of clusters is where the gap statistic is maximized
gap_stat <- clusGap(cluster_data_scaled, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
print(gap_stat)
fviz_gap_stat(gap_stat)

# Based on the above methods, determine the optimal number of clusters
# For this example, let's say the optimal number is 4
# (You should adjust this based on the actual results)
optimal_k <- 5

# Perform k-means clustering with the optimal number of clusters
kmeans_result <- kmeans(cluster_data_scaled, centers = optimal_k, nstart = 25)

# Add cluster assignments to the original dataset
smart_fresh$cluster_kmeans <- as.factor(kmeans_result$cluster)

# Visualize the clusters using PCA for dimensionality reduction
pca_result <- prcomp(cluster_data_scaled)
pca_data <- as.data.frame(pca_result$x[, 1:2])
pca_data$cluster <- kmeans_result$cluster

# Plot the first two principal components
kmeans_pca_plot <- ggplot(pca_data, aes(x = PC1, y = PC2, color = as.factor(cluster))) +
  geom_point(alpha = 0.7) +
  labs(title = "K-means Clusters Visualized with PCA",
       x = "Principal Component 1",
       y = "Principal Component 2",
       color = "Cluster") +
  theme_minimal() +
  scale_color_brewer(palette = "Set1")

print(kmeans_pca_plot)

# 1. Radar chart of key variables by cluster
# First, prepare data for radar chart
radar_vars <- c("Spend_Wine", "Spend_OrganicFood", "Spend_Meat",
                "Spend_WellnessProducts", "Spend_Treats", "Spend_LuxuryGoods",
                "Purchases_Online", "Purchases_Store",
                "Online_Preference", "Promo_Response_Rate")

# Scale values for radar chart (0-1 scale)
radar_data <- smart_fresh %>%
  group_by(cluster) %>%
  summarize(across(all_of(radar_vars), ~mean(., na.rm = TRUE))) %>%
  ungroup()

# Scale each variable to 0-1 range
radar_data_scaled <- radar_data
for (var in radar_vars) {
  min_val <- min(radar_data[[var]])
  max_val <- max(radar_data[[var]])
  radar_data_scaled[[var]] <- (radar_data[[var]] - min_val) / (max_val - min_val)
}

# Convert to long format for plotting
radar_long <- radar_data_scaled %>%
  pivot_longer(cols = all_of(radar_vars),
               names_to = "variable",
               values_to = "value")

# Create radar chart using ggplot2
# First, calculate the angles for each variable
radar_long$angle <- (as.numeric(factor(radar_long$variable)) - 1) *
  (2 * pi / length(unique(radar_long$variable)))
radar_long$hjust <- ifelse(sin(radar_long$angle) > 0, 0, 1)
radar_long$vjust <- ifelse(cos(radar_long$angle) < 0, 1, 0)

# Create the radar chart
ggplot(radar_long, aes(x = variable, y = value, group = cluster, color =
as.factor(cluster))) +
  geom_polygon(aes(fill = as.factor(cluster)), alpha = 0.1) +
  geom_line() +
  geom_point() +
  coord_polar() +
  scale_y_continuous(limits = c(0, 1)) +
  labs(title = "Cluster Profiles Across Key Variables",
       x = "",
       y = "",
       color = "Cluster",
       fill = "Cluster") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 0))
```