# Smartwatch

#Candidate number: 055445

# When you open RStudio:

# Source Editor (top-left): where you write your R code in files (scripts).

# Console (bottom-left): where code is executed.

# Environment/History (top-right): to inspect variables or your command history.

# Files/Plots/Packages/Help (bottom-right): for navigating files, viewing plots, etc.

# Create or Open RScript:

# File > New File > R Script.

# Type your code in the new script window.

# Run the code:

# Place the cursor beside the code you want to run and type on alt + enter

# To check the current working directory

getwd() # Prints the path of the current working directory

# To set a new working directory (replace 'your/path/here' with the desired path)

setwd("C:/Users/dhara/OneDrive/Desktop/Marketing analytics")

# Example: If the working directory is set to "C:/Users/Documents"

# setwd("C:/Users/Documents")

################## INTEL SMARTWATCH ###################

# Note:

# - Replace the blanks with appropriate functions, objects, or parameters.

# - Try to reason through each step of the script to understand what it is doing.

# Install necessary packages if not already installed

install.packages("readxl")

install.packages("tidyverse")

install.packages("cluster")

install.packages("openxlsx")

# Load the required libraries

library(readxl) # For reading Excel files

library(tidyverse) # For data manipulation and visualisation

library(cluster) # For clustering methods

library(openxlsx) # For exporting data to Excel

################## PREPARATION ###################

# IMPORTING DATA FROM EXCEL

# "file.choose()" lets the user select a file interactively.

Smartwatch <- read\_excel(file.choose())

# View the imported dataset (opens in a separate viewer)

View(Smartwatch) # Hint: What did you name your dataset?

# INITIAL DATA EXPLORATION

# Display column names of the dataset

names(Smartwatch) # Hint: What object holds your data?

# Display basic summary statistics (e.g., min, max, mean, etc.) for each variable

summary(Smartwatch) # Hint: Same as above

# Explore the structure of the dataset (e.g., column types, number of rows)

str(Smartwatch) # Hint: Same as above

# REMOVE ID COLUMN (Assuming the first column is an ID column)

# Create a new dataframe excluding the first column

df <- Smartwatch # Hint: Which column number corresponds to the ID?

names(df)

# STANDARDISE DATA

# "scale()" standardises the data to have a mean of 0 and standard deviation of 1.

dfz <- scale(df) # Hint: What data should be scaled?

# View the standardised data

View(dfz)

################### SEGMENTATION STEP ###################

# CALCULATE EUCLIDEAN DISTANCE

# "dist()" calculates pairwise distances between rows of a dataset.

distance <- dist(dfz, method = 'euclidean')

# CLUSTER DENDROGRAM

# Perform hierarchical clustering

# ?hclust

hc.w <- hclust(distance, method = 'ward.D') # Hint: Using Hint: Ward.D linkage

# Plot the dendrogram to visualise the clustering

plot(hc.w, main = "Cluster Dendrogram", xlab = "Obs", ylab = "Height")

# DETERMINE THE OPTIMAL NUMBER OF CLUSTERS

# Use the elbow method to decide the number of clusters.

x <- c(1:10)

sort\_height <- sort(hc.w$height, decreasing = TRUE) #Hint: What we are trying to sort?

y <- sort\_height[1:10]

# Plot elbow plot

plot(x, y, type = "b", main = "Elbow Plot", xlab = "No of clusters", ylab = "Height") # Hint: "Elbow Plot"

lines(x, y, col = "green")

# After cluster=?, adding more clusters isn’t significantly improving the fit within the clusters.

# We can consider ? as the optimal number of clusters.

# Display ? clusters on dendrogram

plot(hc.w, main = "Cluster Dendrogram", xlab = "Obs", ylab = "Height")

rect.hclust(hc.w, k = 4, border = 2:5) # Here, we choose ? clusters

# CUT DENDROGRAM INTO 4 CLUSTERS

# "cutree()" assigns each observation to a cluster.

cluster <- cutree(hc.w, k = 4) # Here, we choose 4 clusters

# Create a frequency table to see the size of each cluster

table(cluster)

# Add cluster assignments back to the original data

df\_final <- cbind(df , cluster) # Hint: Combine the original data with clusters.

# Check the updated dataset

View(df\_final)

################### DESCRIPTION STEP ###################

# CALCULATE SEGMENT SIZES

proportions <- table(df\_final$cluster) / nrow(df\_final) # Hint: What is your final dataset?

percentages <- proportions \* 100

# Display segment sizes in percentages

print(percentages) # Hint: What variable holds the percentages?

# EXPLORE MEAN VALUES OF VARIABLES IN EACH CLUSTER

segments<-

df\_final %>%

group\_by(cluster) %>%

summarise(across(where(is.numeric), mean, .names = "{col}\_mean"))

# Display the calculated means

segments

view(segments)

# SAVE MEAN TABLE TO EXCEl

setwd("C:/Users/dhara/OneDrive/Desktop/Marketing analytics")

write.xlsx(segments, 'Smartwatch.xlsx') # Hint: What variable holds the summarised data?

# Please go to your working directory to open the Excel file and explore

# You can choose the "Files" tab in the bottom-right corner panel to check if you see the Excel file.