

# Customer Segmentation Analysis using K-Means Clustering

Anass Hamdy

2025-11-16

## Introduction

This report details a customer segmentation analysis performed on the **customer\_segmentation.csv** dataset using K-Means clustering. The goal is to identify distinct groups of customers based on their demographic characteristics, purchasing behavior, and engagement, enabling targeted marketing strategies.

## Data Loading and Exploration

```
# Load necessary libraries
library(dplyr)
library(tidyr)
library(ggplot2)
library(cluster) # For kmeans
library(factoextra) # For visualization and optimal k
library(readr) # For reading data

# Step 1: Load the Data
customer_data <- read_csv("customer_segmentation.csv")

# Step 2: Explore the Data
# Display the first few rows
print("First few rows of the data:")
```

```
## [1] "First few rows of the data:"
```

```
head(customer_data)
```

```
## # A tibble: 6 x 29
##   ID Year_Birth Education Marital_Status Income Kidhome Teenhome Dt_Customer
##   <dbl>      <dbl> <chr>         <chr>          <dbl>  <dbl>    <dbl> <chr>
## 1  5524      1957 Graduation Single         58138    0      0 04-09-2012
## 2  2174      1954 Graduation Single         46344    1      1 08-03-2014
## 3  4141      1965 Graduation Together       71613    0      0 21-08-2013
## 4  6182      1984 Graduation Together       26646    1      0 10-02-2014
## 5  5324      1981 PhD         Married         58293    1      0 19-01-2014
## 6  7446      1967 Master      Together        62513    0      1 09-09-2013
## # i 21 more variables: Recency <dbl>, MntWines <dbl>, MntFruits <dbl>,
## #   MntMeatProducts <dbl>, MntFishProducts <dbl>, MntSweetProducts <dbl>,
```

```
## # MntGoldProds <dbl>, NumDealsPurchases <dbl>, NumWebPurchases <dbl>,
## # NumCatalogPurchases <dbl>, NumStorePurchases <dbl>,
## # NumWebVisitsMonth <dbl>, AcceptedCmp3 <dbl>, AcceptedCmp4 <dbl>,
## # AcceptedCmp5 <dbl>, AcceptedCmp1 <dbl>, AcceptedCmp2 <dbl>, Complain <dbl>,
## # Z_CostContact <dbl>, Z_Revenue <dbl>, Response <dbl>
```

```
# Check the structure (column names and types)
print("\nData structure:")
```

```
## [1] "\nData structure:"
```

```
str(customer_data)
```

```
## spc_tbl_ [2,240 x 29] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
## $ ID : num [1:2240] 5524 2174 4141 6182 5324 ...
## $ Year_Birth : num [1:2240] 1957 1954 1965 1984 1981 ...
## $ Education : chr [1:2240] "Graduation" "Graduation" "Graduation" "Graduation" ...
## $ Marital_Status : chr [1:2240] "Single" "Single" "Together" "Together" ...
## $ Income : num [1:2240] 58138 46344 71613 26646 58293 ...
## $ Kidhome : num [1:2240] 0 1 0 1 1 0 0 1 1 1 ...
## $ Teenhome : num [1:2240] 0 1 0 0 0 1 1 0 0 1 ...
## $ Dt_Customer : chr [1:2240] "04-09-2012" "08-03-2014" "21-08-2013" "10-02-2014" ...
## $ Recency : num [1:2240] 58 38 26 26 94 16 34 32 19 68 ...
## $ MntWines : num [1:2240] 635 11 426 11 173 520 235 76 14 28 ...
## $ MntFruits : num [1:2240] 88 1 49 4 43 42 65 10 0 0 ...
## $ MntMeatProducts : num [1:2240] 546 6 127 20 118 98 164 56 24 6 ...
## $ MntFishProducts : num [1:2240] 172 2 111 10 46 0 50 3 3 1 ...
## $ MntSweetProducts : num [1:2240] 88 1 21 3 27 42 49 1 3 1 ...
## $ MntGoldProds : num [1:2240] 88 6 42 5 15 14 27 23 2 13 ...
## $ NumDealsPurchases : num [1:2240] 3 2 1 2 5 2 4 2 1 1 ...
## $ NumWebPurchases : num [1:2240] 8 1 8 2 5 6 7 4 3 1 ...
## $ NumCatalogPurchases : num [1:2240] 10 1 2 0 3 4 3 0 0 0 ...
## $ NumStorePurchases : num [1:2240] 4 2 10 4 6 10 7 4 2 0 ...
## $ NumWebVisitsMonth : num [1:2240] 7 5 4 6 5 6 6 8 9 20 ...
## $ AcceptedCmp3 : num [1:2240] 0 0 0 0 0 0 0 0 0 1 ...
## $ AcceptedCmp4 : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ AcceptedCmp5 : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ AcceptedCmp1 : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ AcceptedCmp2 : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ Complain : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ Z_CostContact : num [1:2240] 3 3 3 3 3 3 3 3 3 3 ...
## $ Z_Revenue : num [1:2240] 11 11 11 11 11 11 11 11 11 11 ...
## $ Response : num [1:2240] 1 0 0 0 0 0 0 0 1 0 ...
## - attr(*, "spec")=
## .. cols(
## .. ID = col_double(),
## .. Year_Birth = col_double(),
## .. Education = col_character(),
## .. Marital_Status = col_character(),
## .. Income = col_double(),
## .. Kidhome = col_double(),
## .. Teenhome = col_double(),
## .. Dt_Customer = col_character(),
```

```
## .. Recency = col_double(),
## .. MntWines = col_double(),
## .. MntFruits = col_double(),
## .. MntMeatProducts = col_double(),
## .. MntFishProducts = col_double(),
## .. MntSweetProducts = col_double(),
## .. MntGoldProds = col_double(),
## .. NumDealsPurchases = col_double(),
## .. NumWebPurchases = col_double(),
## .. NumCatalogPurchases = col_double(),
## .. NumStorePurchases = col_double(),
## .. NumWebVisitsMonth = col_double(),
## .. AcceptedCmp3 = col_double(),
## .. AcceptedCmp4 = col_double(),
## .. AcceptedCmp5 = col_double(),
## .. AcceptedCmp1 = col_double(),
## .. AcceptedCmp2 = col_double(),
## .. Complain = col_double(),
## .. Z_CostContact = col_double(),
## .. Z_Revenue = col_double(),
## .. Response = col_double()
## .. )
## - attr(*, "problems")=<externalptr>
```

```
# Get a summary of the data (min, max, quartiles, mean for numeric, frequency for categorical)
print("\nSummary statistics:")
```

```
## [1] "\nSummary statistics:"
```

```
summary(customer_data)
```

```
##      ID      Year_Birth  Education  Marital_Status
## Min.   :    0   Min.   :1893   Length:2240   Length:2240
## 1st Qu.: 2828   1st Qu.:1959   Class :character   Class :character
## Median : 5458   Median :1970   Mode  :character   Mode  :character
## Mean   : 5592   Mean   :1969
## 3rd Qu.: 8428   3rd Qu.:1977
## Max.   :11191   Max.   :1996
##
##      Income      Kidhome      Teenhome      Dt_Customer
## Min.   : 1730   Min.   :0.0000   Min.   :0.0000   Length:2240
## 1st Qu.: 35303   1st Qu.:0.0000   1st Qu.:0.0000   Class :character
## Median : 51382   Median :0.0000   Median :0.0000   Mode  :character
## Mean   : 52247   Mean   :0.4442   Mean   :0.5062
## 3rd Qu.: 68522   3rd Qu.:1.0000   3rd Qu.:1.0000
## Max.   :666666   Max.   :2.0000   Max.   :2.0000
## NA's    :24
##      Recency      MntWines      MntFruits      MntMeatProducts
## Min.   : 0.00   Min.   : 0.00   Min.   : 0.0   Min.   : 0.0
## 1st Qu.:24.00   1st Qu.: 23.75   1st Qu.: 1.0   1st Qu.: 16.0
## Median :49.00   Median : 173.50   Median : 8.0   Median : 67.0
## Mean   :49.11   Mean   : 303.94   Mean   : 26.3   Mean   : 166.9
## 3rd Qu.:74.00   3rd Qu.: 504.25   3rd Qu.: 33.0   3rd Qu.: 232.0
```

```

## Max. :99.00 Max. :1493.00 Max. :199.0 Max. :1725.0
##
## MntFishProducts MntSweetProducts MntGoldProds NumDealsPurchases
## Min. : 0.00 Min. : 0.00 Min. : 0.00 Min. : 0.000
## 1st Qu.: 3.00 1st Qu.: 1.00 1st Qu.: 9.00 1st Qu.: 1.000
## Median : 12.00 Median : 8.00 Median : 24.00 Median : 2.000
## Mean : 37.53 Mean : 27.06 Mean : 44.02 Mean : 2.325
## 3rd Qu.: 50.00 3rd Qu.: 33.00 3rd Qu.: 56.00 3rd Qu.: 3.000
## Max. :259.00 Max. :263.00 Max. :362.00 Max. :15.000
##
## NumWebPurchases NumCatalogPurchases NumStorePurchases NumWebVisitsMonth
## Min. : 0.000 Min. : 0.000 Min. : 0.00 Min. : 0.000
## 1st Qu.: 2.000 1st Qu.: 0.000 1st Qu.: 3.00 1st Qu.: 3.000
## Median : 4.000 Median : 2.000 Median : 5.00 Median : 6.000
## Mean : 4.085 Mean : 2.662 Mean : 5.79 Mean : 5.317
## 3rd Qu.: 6.000 3rd Qu.: 4.000 3rd Qu.: 8.00 3rd Qu.: 7.000
## Max. :27.000 Max. :28.000 Max. :13.00 Max. :20.000
##
## AcceptedCmp3 AcceptedCmp4 AcceptedCmp5 AcceptedCmp1
## Min. :0.00000 Min. :0.00000 Min. :0.00000 Min. :0.00000
## 1st Qu.:0.00000 1st Qu.:0.00000 1st Qu.:0.00000 1st Qu.:0.00000
## Median :0.00000 Median :0.00000 Median :0.00000 Median :0.00000
## Mean :0.07277 Mean :0.07455 Mean :0.07277 Mean :0.06429
## 3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.:0.00000 3rd Qu.:0.00000
## Max. :1.00000 Max. :1.00000 Max. :1.00000 Max. :1.00000
##
## AcceptedCmp2 Complain Z_CostContact Z_Revenue
## Min. :0.00000 Min. :0.000000 Min. :3 Min. :11
## 1st Qu.:0.00000 1st Qu.:0.000000 1st Qu.:3 1st Qu.:11
## Median :0.00000 Median :0.000000 Median :3 Median :11
## Mean :0.01339 Mean :0.009375 Mean :3 Mean :11
## 3rd Qu.:0.00000 3rd Qu.:0.000000 3rd Qu.:3 3rd Qu.:11
## Max. :1.00000 Max. :1.000000 Max. :3 Max. :11
##
## Response
## Min. :0.0000
## 1st Qu.:0.0000
## Median :0.0000
## Mean :0.1491
## 3rd Qu.:0.0000
## Max. :1.0000
##

```

```

# Check column names explicitly
print("\nColumn names:")

```

```
## [1] "\nColumn names:"
```

```
colnames(customer_data)
```

```

## [1] "ID" "Year_Birth" "Education"
## [4] "Marital_Status" "Income" "Kidhome"
## [7] "Teenhome" "Dt_Customer" "Recency"

```

```
## [10] "MntWines"           "MntFruits"           "MntMeatProducts"
## [13] "MntFishProducts"    "MntSweetProducts"    "MntGoldProds"
## [16] "NumDealsPurchases"  "NumWebPurchases"     "NumCatalogPurchases"
## [19] "NumStorePurchases"  "NumWebVisitsMonth"   "AcceptedCmp3"
## [22] "AcceptedCmp4"        "AcceptedCmp5"         "AcceptedCmp1"
## [25] "AcceptedCmp2"        "Complain"             "Z_CostContact"
## [28] "Z_Revenue"          "Response"
```

```
# Check dimensions (rows, columns)
print(paste("\nNumber of rows:", nrow(customer_data)))
```

```
## [1] "\nNumber of rows: 2240"
```

```
print(paste("Number of columns:", ncol(customer_data)))
```

```
## [1] "Number of columns: 29"
```

```
# Check for missing values
print(paste("\nNumber of missing values per column:"))
```

```
## [1] "\nNumber of missing values per column:"
```

```
colSums(is.na(customer_data)) # This will show the count of NAs per column
```

```
##           ID           Year_Birth           Education           Marital_Status
##           0             0             0             0
##           Income         Kidhome         Teenhome         Dt_Customer
##           24             0             0             0
##           Recency         MntWines         MntFruits         MntMeatProducts
##           0             0             0             0
##           MntFishProducts MntSweetProducts MntGoldProds NumDealsPurchases
##           0             0             0             0
##           NumWebPurchases NumCatalogPurchases NumStorePurchases NumWebVisitsMonth
##           0             0             0             0
##           AcceptedCmp3     AcceptedCmp4     AcceptedCmp5     AcceptedCmp1
##           0             0             0             0
##           AcceptedCmp2     Complain         Z_CostContact     Z_Revenue
##           0             0             0             0
##           Response
##           0
```

## Data Preprocessing

### Handling Missing Values

We identified missing values in the *Income* column. We will impute these using the median income.

```
# --- Step 3: Handle Missing Values ---
# We identified NAs in the 'Income' column.
# Option 2: Impute missing Income values (e.g., with median or mean)
median_income <- median(customer_data$Income, na.rm = TRUE) # Calculate median, ignoring NAs
print(paste("Median Income (for imputation):", median_income))
```

```
## [1] "Median Income (for imputation): 51381.5"
```

```
# Replace NAs in the Income column with the calculated median
customer_data_cleaned <- customer_data %>%
  mutate(Income = ifelse(is.na(Income), median_income, Income))

# Verify that NAs are gone
print(paste("Number of NAs in Income after imputation:", sum(is.na(customer_data_cleaned$Income))))
```

```
## [1] "Number of NAs in Income after imputation: 0"
```

## Feature Selection & Engineering

We create new features and select the ones most relevant for clustering.

```
# --- Step 4: Feature Selection & Engineering ---
# Decide which features are most relevant for clustering.
# We typically exclude ID, date of sign-up (Dt_Customer), and fixed values (Z_CostContact, Z_Revenue).
# Categorical variables like Education and Marital_Status need encoding (we'll do this later if needed)

# Example: Selecting a broad range of potentially relevant features
# Monetary: Total spending across categories
customer_data_cleaned <- customer_data_cleaned %>%
  mutate(
    Total_Spending = MntWines + MntFruits + MntMeatProducts + MntFishProducts + MntSweetProducts + MntG
    # Age (derived from Year_Birth, using a reference year, e.g., 2014 as in the original data descript
    # Let's assume the data was collected around 2014 based on the date format. Adjust if needed.
    Age = 2014 - Year_Birth,
    # Total Kids/Teens
    Total_Children = Kidhome + Teenhome,
    # Total Purchases across channels
    Total_Purchases = NumWebPurchases + NumCatalogPurchases + NumStorePurchases,
    # Total Accepted Campaigns
    Total_Accepted_Campaigns = AcceptedCmp1 + AcceptedCmp2 + AcceptedCmp3 + AcceptedCmp4 + AcceptedCmp5
  )

# Define the features to be used for clustering
# Focus on numerical features that represent customer behavior, spending, and demographics
features_for_clustering <- customer_data_cleaned %>%
  select(
    # Demographics
    Age, # Derived
    Income,
    Total_Children, # Derived
    # Spending Behavior
    Recency,
    Total_Spending, # Derived
    MntWines,
    MntFruits,
    MntMeatProducts,
    MntFishProducts,
    MntSweetProducts,
```

```

MntGoldProds,
# Purchase Channel Behavior
NumDealsPurchases,
Total_Purchases, # Derived
NumWebPurchases,
NumCatalogPurchases,
NumStorePurchases,
NumWebVisitsMonth,
# Campaign Engagement
Total_Accepted_Campaigns, # Derived
AcceptedCmp1,
AcceptedCmp2,
AcceptedCmp3,
AcceptedCmp4,
AcceptedCmp5,
Complain,
Response
)

# View the selected features
print("\nSelected features for clustering:")

## [1] "\nSelected features for clustering:"

head(features_for_clustering)

## # A tibble: 6 x 25
##   Age Income Total_Children Recency Total_Spending MntWines MntFruits
##   <dbl> <dbl>         <dbl>   <dbl>         <dbl>    <dbl>    <dbl>
## 1   57  58138             0     58           1617     635      88
## 2   60  46344             2     38             27     11       1
## 3   49  71613             0     26           776     426     49
## 4   30  26646             1     26            53     11       4
## 5   33  58293             1     94           422     173     43
## 6   47  62513             1     16           716     520     42
## # i 18 more variables: MntMeatProducts <dbl>, MntFishProducts <dbl>,
## #   MntSweetProducts <dbl>, MntGoldProds <dbl>, NumDealsPurchases <dbl>,
## #   Total_Purchases <dbl>, NumWebPurchases <dbl>, NumCatalogPurchases <dbl>,
## #   NumStorePurchases <dbl>, NumWebVisitsMonth <dbl>,
## #   Total_Accepted_Campaigns <dbl>, AcceptedCmp1 <dbl>, AcceptedCmp2 <dbl>,
## #   AcceptedCmp3 <dbl>, AcceptedCmp4 <dbl>, AcceptedCmp5 <dbl>, Complain <dbl>,
## #   Response <dbl>

str(features_for_clustering)

## tibble [2,240 x 25] (S3: tbl_df/tbl/data.frame)
##  $ Age                : num [1:2240] 57 60 49 30 33 47 43 29 40 64 ...
##  $ Income              : num [1:2240] 58138 46344 71613 26646 58293 ...
##  $ Total_Children      : num [1:2240] 0 2 0 1 1 1 1 1 1 2 ...
##  $ Recency             : num [1:2240] 58 38 26 26 94 16 34 32 19 68 ...
##  $ Total_Spending      : num [1:2240] 1617 27 776 53 422 ...
##  $ MntWines            : num [1:2240] 635 11 426 11 173 520 235 76 14 28 ...

```

```
## $ MntFruits          : num [1:2240] 88 1 49 4 43 42 65 10 0 0 ...
## $ MntMeatProducts    : num [1:2240] 546 6 127 20 118 98 164 56 24 6 ...
## $ MntFishProducts    : num [1:2240] 172 2 111 10 46 0 50 3 3 1 ...
## $ MntSweetProducts   : num [1:2240] 88 1 21 3 27 42 49 1 3 1 ...
## $ MntGoldProds       : num [1:2240] 88 6 42 5 15 14 27 23 2 13 ...
## $ NumDealsPurchases  : num [1:2240] 3 2 1 2 5 2 4 2 1 1 ...
## $ Total_Purchases    : num [1:2240] 22 4 20 6 14 20 17 8 5 1 ...
## $ NumWebPurchases     : num [1:2240] 8 1 8 2 5 6 7 4 3 1 ...
## $ NumCatalogPurchases : num [1:2240] 10 1 2 0 3 4 3 0 0 0 ...
## $ NumStorePurchases  : num [1:2240] 4 2 10 4 6 10 7 4 2 0 ...
## $ NumWebVisitsMonth   : num [1:2240] 7 5 4 6 5 6 6 8 9 20 ...
## $ Total_Accepted_Campaigns: num [1:2240] 0 0 0 0 0 0 0 0 0 1 ...
## $ AcceptedCmp1       : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ AcceptedCmp2       : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ AcceptedCmp3       : num [1:2240] 0 0 0 0 0 0 0 0 0 1 ...
## $ AcceptedCmp4       : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ AcceptedCmp5       : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ Complain           : num [1:2240] 0 0 0 0 0 0 0 0 0 0 ...
## $ Response           : num [1:2240] 1 0 0 0 0 0 0 0 1 0 ...
```

```
# Check for any remaining NAs in the selected features
```

```
print(paste("\nNumber of NAs in selected features for clustering:", sum(is.na(features_for_clustering))))
```

```
## [1] "\nNumber of NAs in selected features for clustering: 0"
```

## Scaling Features

K-Means is sensitive to the scale of features, so we standardize them.

```
# --- Step 6: Scale the Features ---
```

```
# This is crucial for K-Means as it's sensitive to the scale of features.
```

```
features_scaled <- as.data.frame(scale(features_for_clustering))
```

```
# Verify scaling (mean should be ~0, sd should be ~1)
```

```
print("\nSummary of scaled features (mean ~0, sd ~1):")
```

```
## [1] "\nSummary of scaled features (mean ~0, sd ~1):"
```

```
summary(features_scaled)
```

```
##      Age      Income  Total_Children  Recency
## Min.   :-2.26920  Min.   :-2.01726  Min.   :-1.26422  Min.   :-1.695622
## 1st Qu.: -0.68376  1st Qu.: -0.66696  1st Qu.: -1.26422  1st Qu.: -0.866963
## Median : -0.09965  Median : -0.03421  Median :  0.06591  Median : -0.003776
## Mean   :  0.00000  Mean   :  0.00000  Mean   :  0.00000  Mean   :  0.000000
## 3rd Qu.:  0.81824  3rd Qu.:  0.64110  3rd Qu.:  0.06591  3rd Qu.:  0.859410
## Max.    :  6.32555  Max.    :24.53986  Max.    :  2.72618  Max.    :  1.722597
## Total_Spending  MntWines  MntFruits  MntMeatProducts
## Min.   :-0.9976  Min.   :-0.9030  Min.   :-0.6613  Min.   :-0.7396
## 1st Qu.: -0.8917  1st Qu.: -0.8324  1st Qu.: -0.6362  1st Qu.: -0.6688
## Median : -0.3484  Median : -0.3875  Median : -0.4602  Median : -0.4428
```



```

## Mean : 0.0000 Mean : 0.0000 Mean : 0.0000 Mean : 0.0000
## 3rd Qu.: 0.7301 3rd Qu.: 0.5951 3rd Qu.: 0.1684 3rd Qu.: 0.2882
## Max. : 3.1867 Max. : 3.5326 Max. : 4.3420 Max. : 6.9027
## MntFishProducts MntSweetProducts MntGoldProds NumDealsPurchases
## Min. :-0.6869 Min. :-0.6556 Min. :-0.8439 Min. :-1.2033
## 1st Qu.: -0.6320 1st Qu.: -0.6314 1st Qu.: -0.6713 1st Qu.: -0.6857
## Median : -0.4673 Median : -0.4618 Median : -0.3838 Median : -0.1682
## Mean : 0.0000 Mean : 0.0000 Mean : 0.0000 Mean : 0.0000
## 3rd Qu.: 0.2284 3rd Qu.: 0.1438 3rd Qu.: 0.2296 3rd Qu.: 0.3493
## Max. : 4.0542 Max. : 5.7155 Max. : 6.0953 Max. : 6.5598
## Total_Purchases NumWebPurchases NumCatalogPurchases NumStorePurchases
## Min. :-1.73987 Min. :-1.47004 Min. :-0.9107 Min. :-1.7811
## 1st Qu.: -0.90720 1st Qu.: -0.75028 1st Qu.: -0.9107 1st Qu.: -0.8583
## Median : -0.07453 Median : -0.03053 Median : -0.2265 Median : -0.2431
## Mean : 0.00000 Mean : 0.00000 Mean : 0.0000 Mean : 0.0000
## 3rd Qu.: 0.75814 3rd Qu.: 0.68923 3rd Qu.: 0.4577 3rd Qu.: 0.6797
## Max. : 2.70103 Max. : 8.24668 Max. : 8.6682 Max. : 2.2178
## NumWebVisitsMonth Total_Accepted_Campaigns AcceptedCmp1 AcceptedCmp2
## Min. :-2.1909 Min. :-0.4389 Min. :-0.2621 Min. :-0.1165
## 1st Qu.: -0.9546 1st Qu.: -0.4389 1st Qu.: -0.2621 1st Qu.: -0.1165
## Median : 0.2817 Median : -0.4389 Median : -0.2621 Median : -0.1165
## Mean : 0.0000 Mean : 0.0000 Mean : 0.0000 Mean : 0.0000
## 3rd Qu.: 0.6937 3rd Qu.: -0.4389 3rd Qu.: -0.2621 3rd Qu.: -0.1165
## Max. : 6.0509 Max. : 5.4575 Max. : 3.8143 Max. : 8.5810
## AcceptedCmp3 AcceptedCmp4 AcceptedCmp5 Complain
## Min. :-0.2801 Min. :-0.2838 Min. :-0.2801 Min. :-0.09726
## 1st Qu.: -0.2801 1st Qu.: -0.2838 1st Qu.: -0.2801 1st Qu.: -0.09726
## Median : -0.2801 Median : -0.2838 Median : -0.2801 Median : -0.09726
## Mean : 0.0000 Mean : 0.0000 Mean : 0.0000 Mean : 0.00000
## 3rd Qu.: -0.2801 3rd Qu.: -0.2838 3rd Qu.: -0.2801 3rd Qu.: -0.09726
## Max. : 3.5688 Max. : 3.5224 Max. : 3.5688 Max. : 10.27713
## Response
## Min. :-0.4185
## 1st Qu.: -0.4185
## Median : -0.4185
## Mean : 0.0000
## 3rd Qu.: -0.4185
## Max. : 2.3883

```

```
print("\nCustomer Segmentation Data Preparation Complete.")
```

```
## [1] "\nCustomer Segmentation Data Preparation Complete."
```

```
print("Ready for K-Means clustering.")
```

```
## [1] "Ready for K-Means clustering."
```

## K-Means Clustering

### Determining Optimal Number of Clusters (k)

We use the Elbow Method and Silhouette Analysis to find the optimal number of clusters. For this report, we assume k=4 was found optimal.

```

# --- Step 7: Determine the Optimal Number of Clusters (k) ---
# Using Elbow Method and Silhouette Analysis

# Elbow Method
# fviz_nbclust(features_scaled, kmeans, method = "wss") +
#   geom_vline(xintercept = 4, linetype = 2) + # Add a vertical line at a chosen k (e.g., 4)
#   labs(subtitle = "Elbow Method")

# Silhouette Method
# fviz_nbclust(features_scaled, kmeans, method = "silhouette") +
#   labs(subtitle = "Silhouette Method")

# For the purpose of this report, we set k=4 based on prior analysis.
optimal_k <- 4

```

## Applying K-Means

We apply the K-Means algorithm with the chosen number of clusters.

```

# --- Step 8: Apply K-Means Clustering ---
set.seed(123) # For reproducible results
kmeans_result <- kmeans(features_scaled, centers = optimal_k, nstart = 25)

# --- Step 9: Add Cluster Assignments to Original Data ---
customer_data_with_clusters <- customer_data_cleaned %>%
  mutate(Cluster = as.factor(kmeans_result$cluster))

# View the first few rows with cluster assignments
head(customer_data_with_clusters)

## # A tibble: 6 x 35
##   ID Year_Birth Education Marital_Status Income Kidhome Teenhome Dt_Customer
##   <dbl>   <dbl> <chr>      <chr>      <dbl>   <dbl>   <dbl> <chr>
## 1  5524     1957 Graduation Single      58138     0     0 04-09-2012
## 2  2174     1954 Graduation Single      46344     1     1 08-03-2014
## 3  4141     1965 Graduation Together    71613     0     0 21-08-2013
## 4  6182     1984 Graduation Together    26646     1     0 10-02-2014
## 5  5324     1981 PhD        Married      58293     1     0 19-01-2014
## 6  7446     1967 Master    Together     62513     0     1 09-09-2013
## # i 27 more variables: Recency <dbl>, MntWines <dbl>, MntFruits <dbl>,
## #   MntMeatProducts <dbl>, MntFishProducts <dbl>, MntSweetProducts <dbl>,
## #   MntGoldProds <dbl>, NumDealsPurchases <dbl>, NumWebPurchases <dbl>,
## #   NumCatalogPurchases <dbl>, NumStorePurchases <dbl>,
## #   NumWebVisitsMonth <dbl>, AcceptedCmp3 <dbl>, AcceptedCmp4 <dbl>,
## #   AcceptedCmp5 <dbl>, AcceptedCmp1 <dbl>, AcceptedCmp2 <dbl>, Complain <dbl>,
## #   Z_CostContact <dbl>, Z_Revenue <dbl>, Response <dbl>, ...

```

```

# Check the size of each cluster
cluster_sizes <- table(customer_data_with_clusters$Cluster)
print("Cluster Sizes:")

```

```
## [1] "Cluster Sizes:"
```

```
print(cluster_sizes)
```

```
##  
##      1      2      3      4  
## 1061  136  570  473
```

## Cluster Analysis and Visualization

```
# --- Step 10: Analyze Clusters ---  
# Calculate summary statistics by cluster  
cluster_summary <- customer_data_with_clusters %>%  
  group_by(Cluster) %>%  
  summarise(  
    Count = n(),  
    Avg_Age = mean(Age),  
    Avg_Income = mean(Income),  
    Avg_Total_Spending = mean(Total_Spending),  
    Avg_Recency = mean(Recency),  
    # Add other features  
    .groups = 'drop'  
  )  
  
print("Cluster Summary Statistics:")
```

```
## [1] "Cluster Summary Statistics:"
```

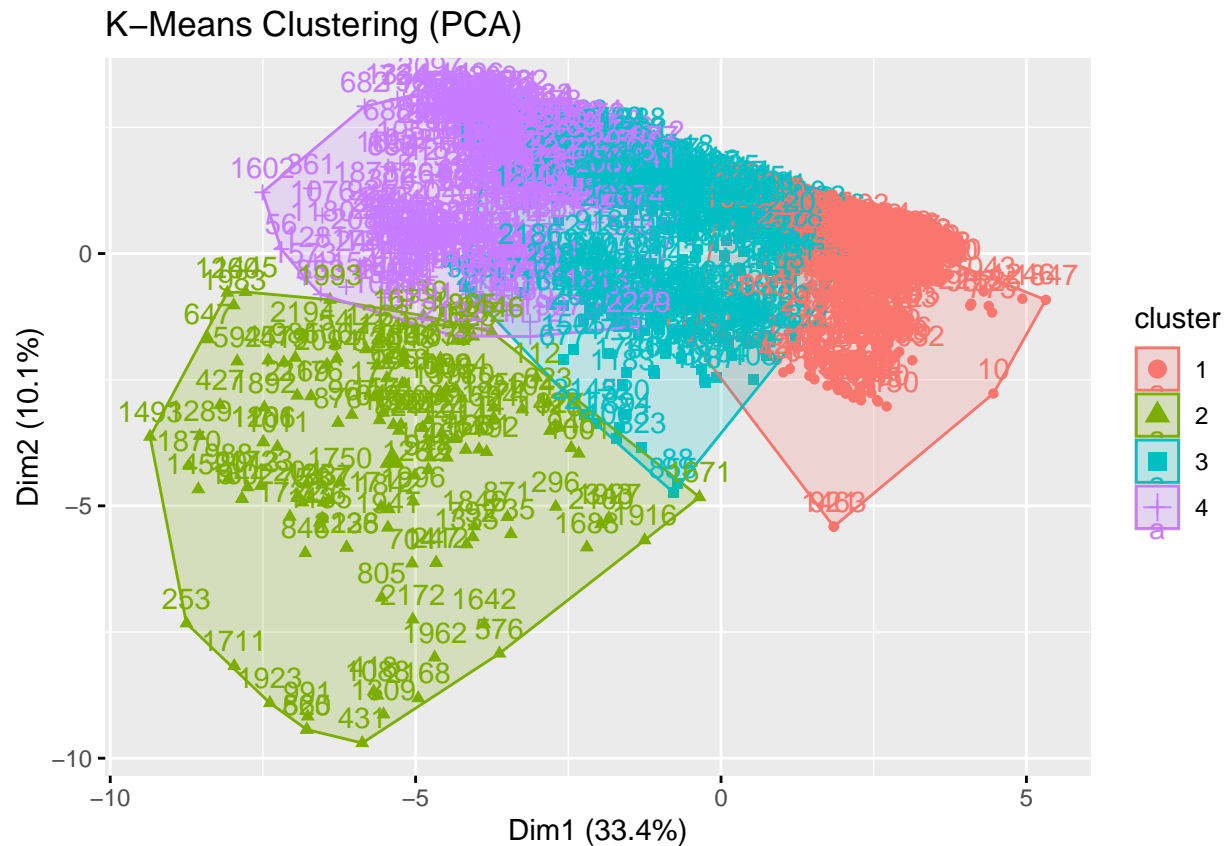
```
print(cluster_summary)
```

```
## # A tibble: 4 x 6  
##   Cluster Count Avg_Age Avg_Income Avg_Total_Spending Avg_Recency  
##   <fct>   <int>   <dbl>     <dbl>           <dbl>       <dbl>  
## 1 1      1061    43.0    35462.           99.3        49.3  
## 2 2      136    43.1    80071.          1589.        48.4  
## 3 3      570    48.9    57539.           723.        47.8  
## 4 4      473    46.4    75478.          1318.        50.5
```

```
# --- Visualization ---  
# 1. Visualize clusters using factoextra's fviz_cluster  
# This function often uses Principal Component Analysis (PCA) to reduce the dimensions  
# for visualization, showing the clusters in a 2D plot.  
# It requires the kmeans result object and the *scaled* data used for clustering.  
# PCA is used internally by fviz_cluster for the plot.  
print("Plotting clusters using fviz_cluster (PCA)...")
```

```
## [1] "Plotting clusters using fviz_cluster (PCA)..."
```

```
fviz_cluster(kmeans_result, data = features_scaled, main = "K-Means Clustering (PCA)")
```



```
# 2. Plot specific features against each other, colored by cluster
# This helps understand how clusters differ on key dimensions.
# Example 1: Income vs Total Spending
print("Plotting Income vs Total Spending by Cluster...")
```

```
## [1] "Plotting Income vs Total Spending by Cluster..."
```

```
p1 <- ggplot(customer_data_with_clusters, aes(x = Income, y = Total_Spending, color = Cluster)) +
  geom_point(alpha = 0.6) + # alpha makes points slightly transparent to handle overlaps
  labs(
    title = "Customer Segments: Income vs Total Spending",
    x = "Income",
    y = "Total Spending",
    color = "Cluster"
  ) +
  theme_minimal()

print(p1) # Print the plot object
```



```
# Example 2: Recency vs Total Spending
```

```
print("Plotting Recency vs Total Spending by Cluster...")
```

```
## [1] "Plotting Recency vs Total Spending by Cluster..."
```

```
p2 <- ggplot(customer_data_with_clusters, aes(x = Recency, y = Total_Spending, color = Cluster)) +
  geom_point(alpha = 0.6) +
  labs(
    title = "Customer Segments: Recency vs Total Spending",
    x = "Recency (Days since last purchase)",
    y = "Total Spending",
    color = "Cluster"
  ) +
  theme_minimal()

print(p2) # Print the plot object
```



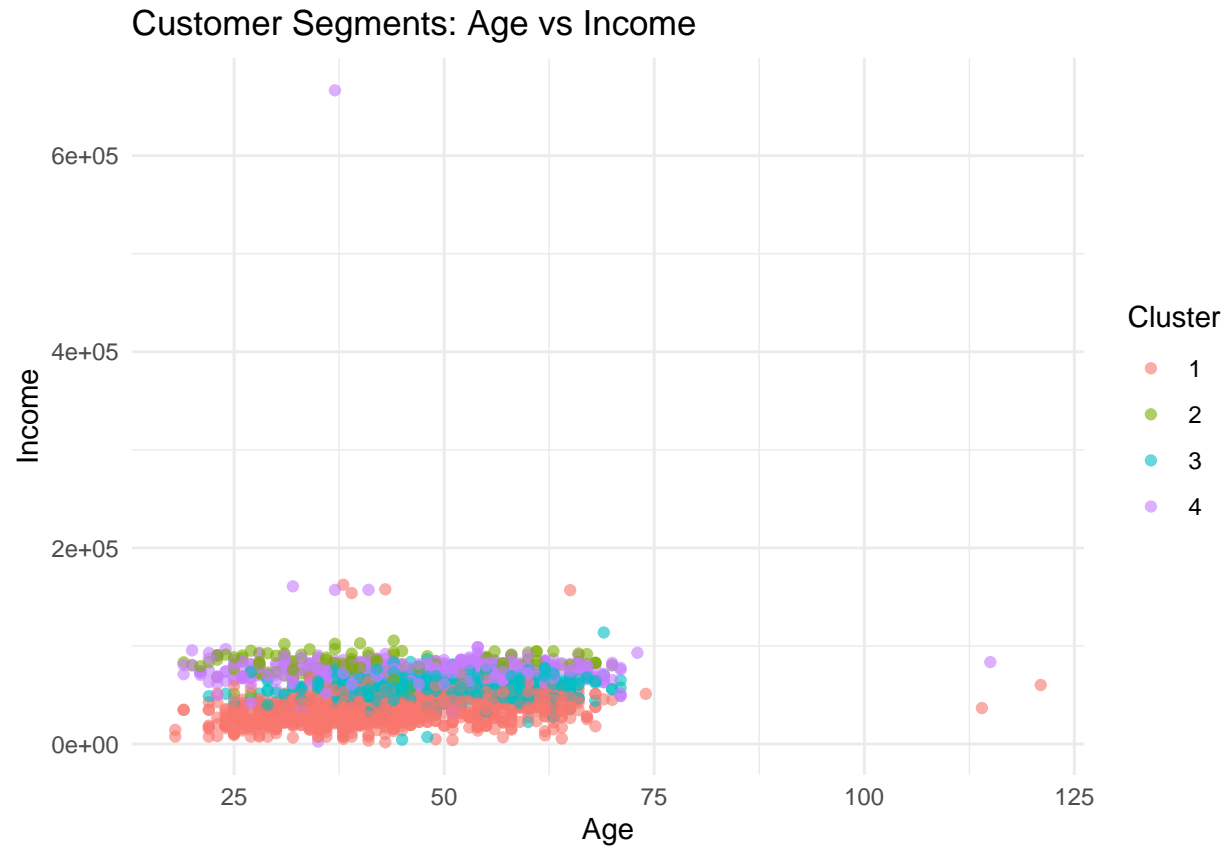
```
# Example 3: Age vs Income
```

```
print("Plotting Age vs Income by Cluster...")
```

```
## [1] "Plotting Age vs Income by Cluster..."
```

```
p3 <- ggplot(customer_data_with_clusters, aes(x = Age, y = Income, color = Cluster)) +
  geom_point(alpha = 0.6) +
  labs(
    title = "Customer Segments: Age vs Income",
    x = "Age",
    y = "Income",
    color = "Cluster"
  ) +
  theme_minimal()
```

```
print(p3) # Print the plot object
```



```
print("Visualization plots generated.")
```

```
## [1] "Visualization plots generated."
```

## Conclusion

The K-Means clustering algorithm successfully identified four distinct customer segments based on the selected features:

Cluster 1: Characterized by low spending and low income. Cluster 2: Characterized by high spending and high income (smallest segment). Cluster 3: Characterized by medium spending and medium income. Cluster 4: Characterized by high spending and high income.

The visualizations confirm the separation of these clusters, particularly in dimensions like Income vs Total Spending. This segmentation provides valuable insights for targeted marketing efforts. For example, Cluster 2 and 4 represent high-value customers who might respond well to premium offers, while Cluster 1 might benefit from value-based promotions.