

VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

**A4- Multivariate Analysis and Business Analytics Applications
(Part – B)**

ADHYAYAN AMIT JAIN

V01109421

Date of Submission: 08-07-2024

CONTENTS

Sl. No.	Title	Page No.
1.	Introduction	1
2.	Results	4
3.	Interpretations	4
4.	Implications	19
5.	Recommendations	21
6	Codes	23
7.	References	27

Conducting Cluster Analysis to Characterize Respondents Based on Background Variables (Survey.csv)

INTRODUCTION

In the era of big data, extracting actionable insights from complex datasets is crucial for organizations across various domains. Cluster analysis emerges as a powerful technique in data science, enabling the discovery of hidden patterns and structures within data. This assignment explores the application of cluster analysis to characterize respondents based on their background variables, using the comprehensive 'Survey.csv' dataset as a foundation for investigation.

Cluster analysis is instrumental in segmenting data into meaningful groups, or clusters, based on similarities in the attributes of individual data points. By identifying and interpreting these clusters, analysts can discern distinct profiles among respondents, facilitating targeted strategies and informed decision-making. This methodological approach not only enhances understanding of respondent behavior but also supports the development of tailored interventions and strategies in fields such as marketing, social sciences, and public policy.

The 'Survey.csv' dataset is a rich repository of respondent data, encompassing a diverse array of variables that capture demographic, psychographic, and behavioral attributes. This dataset offers a robust basis for exploring how different clusters of respondents exhibit unique characteristics and preferences, shedding light on nuanced insights that can inform organizational strategies and societal interventions.

This assignment employs two primary methods of cluster analysis: k-means clustering and hierarchical clustering. K-means clustering partitions data points into k clusters based on feature similarity, while hierarchical clustering constructs a hierarchy of clusters through iterative merging or splitting of data points. By leveraging these techniques, this assignment aims to provide a comprehensive exploration of respondent segmentation, offering insights into the underlying structures and relationships within the dataset.

Through systematic analysis and interpretation of clustering results, this assignment seeks to uncover actionable insights and patterns that can guide decision-making processes. By presenting a nuanced understanding of respondent segmentation based on background variables, this study contributes to the broader discourse on data-driven decision-making and strategic planning in contemporary data science practice.

OBJECTIVES

This assignment is designed to comprehensively explore and apply cluster analysis techniques to characterize respondents based on their background variables using the 'Survey.csv' dataset. The specific objectives include:

1. **Identify Distinct Clusters:** Utilize advanced clustering algorithms, including k-means and hierarchical clustering, to partition respondents into distinct groups based on similarities in their background variables. This involves selecting appropriate clustering methods that best fit the structure and characteristics of the dataset.
2. **Pattern Recognition and Description:** Conduct in-depth exploration and interpretation of patterns within each identified cluster. This includes analyzing demographic, psychographic, and behavioral attributes to uncover underlying trends, preferences, and behavioral profiles among respondent groups.
3. **Insight Generation:** Extract actionable insights from the clustering results that can inform strategic decision-making in various domains. By identifying clear and differentiated respondent segments, this objective aims to assist stakeholders in tailoring interventions, marketing campaigns, and policy strategies to better meet the needs and preferences of different respondent clusters.
4. **Methodological Rigor:** Ensure methodological soundness throughout the analysis process. This includes validating clustering results, assessing the robustness of clusters through sensitivity analysis, and addressing any potential biases or limitations inherent in the clustering techniques employed.
5. **Visualization and Interpretation:** Employ effective visualizations such as scatter plots, dendrograms, and heatmaps to present clustering results in a clear and insightful manner. Visual aids will be utilized to illustrate cluster boundaries, intra-cluster relationships, and feature distributions, facilitating a deeper understanding and interpretation of the findings.

By achieving these objectives, this assignment aims to provide a rigorous and thorough analysis of respondent segmentation based on their background variables. It seeks to demonstrate the practical utility of cluster analysis in

extracting meaningful insights from complex datasets, thereby contributing to enhanced decision-making processes in both academic and practical contexts.

BUSINESS SIGNIFICANCE

Cluster analysis plays a pivotal role in deriving actionable insights that hold significant implications for businesses and organizations across various sectors. In the context of this assignment, conducting cluster analysis on respondents based on their background variables from the 'Survey.csv' dataset carries substantial business significance in several key areas:

1. **Targeted Marketing Strategies:** By identifying distinct clusters of respondents with similar demographic, psychographic, and behavioral profiles, businesses can tailor their marketing strategies more effectively. This includes personalized messaging, product recommendations, and targeted promotions that resonate with the specific preferences and needs of each cluster.
2. **Customer Segmentation and Engagement:** Understanding the unique characteristics and preferences of different respondent clusters enables businesses to segment their customer base more strategically. This segmentation facilitates improved customer engagement strategies, customer retention efforts, and the development of customized products or services that cater to diverse customer segments.
3. **Operational Efficiency and Resource Allocation:** Clustering analysis helps businesses optimize resource allocation and operational efficiency. By categorizing respondents into meaningful groups, organizations can allocate resources such as manpower, budget, and time more efficiently to address the specific needs and demands of each cluster, thereby enhancing overall operational effectiveness.
4. **Decision Support and Strategic Planning:** The insights derived from cluster analysis serve as a robust foundation for informed decision-making and strategic planning. Businesses can use these insights to make data-driven decisions regarding market expansion, product diversification, geographical targeting, and competitive positioning in the marketplace.
5. **Risk Management and Mitigation:** Clustering analysis also aids in identifying potential risks and vulnerabilities associated with different respondent segments. Businesses can proactively address risks by

developing targeted risk management strategies and contingency plans tailored to the characteristics and behaviors of each cluster.

6. **Policy Formulation and Public Sector Applications:** Beyond commercial applications, the findings from cluster analysis can inform policy formulation and public sector interventions. Governments and policymakers can use these insights to design effective public policies, social programs, and community initiatives that address the specific needs and challenges of diverse demographic groups.

In conclusion, the business significance of conducting cluster analysis on respondents based on background variables extends far beyond mere data exploration. It empowers businesses and organizations to optimize their operations, enhance customer relationships, mitigate risks, and drive strategic growth initiatives, ultimately fostering a competitive edge in today's dynamic and data-driven business environment.

RESULTS AND INTERPRETATIONS

R Language

Step-by-Step Analysis and Interpretation:

1. Function to Auto-Install and Load Packages

```
# Function to auto-install and load packages
install_and_load <- function(packages) {
  for (package in packages) {
    if (!require(package, character.only = TRUE)) {
      install.packages(package, dependencies = TRUE)
    }
    library(package, character.only = TRUE)
  }
}
```

- **Explanation:**
 - `install_and_load()` is a custom function that automates the installation and loading of R packages listed in the `packages` vector.
 - It checks if each package is already installed (`require(package, character.only = TRUE)`). If not, it installs the package using `install.packages(package, dependencies = TRUE)` and then loads it with `library(package, character.only = TRUE)`.

```
> # Function to auto-install and load packages
> install_and_load <- function(packages) {
+   for (package in packages) {
+     if (!require(package, character.only = TRUE)) {
+       install.packages(package, dependencies = TRUE)
+     }
+     library(package, character.only = TRUE)
+   }
+ }
>
```

2. List of Packages to Install and Load

```
# List of packages to install and load
packages <- c("cluster", "FactoMineR", "factoextra", "pheatmap")
```

- **Explanation:**
 - `packages` is a vector containing the names of R packages (`cluster`, `FactoMineR`, `factoextra`, `pheatmap`) required for subsequent analysis.

```
> # List of packages to install and load
> packages <- c("cluster", "FactoMineR", "factoextra", "pheatmap")
>
```

3. Install and Load Required Packages

```
# Install and load required packages
install_and_load(packages)
```

- **Explanation:**
 - The `install_and_load()` function is called with `packages` as an argument to ensure all required packages are installed and loaded into the R session.

```
> # Install and load required packages
> install_and_load(packages)
>
```

4. Read the Survey Data

```
# Read the survey data
survey_df <- read.csv('Survey.csv', header = TRUE)
```

- **Explanation:**
 - `read.csv()` reads the 'Survey.csv' file into `survey_df` dataframe, assuming the file is located in the current working directory.
 - `header = TRUE` indicates that the first row contains column headers.

```
>
> # Read the survey data
> survey_df <- read.csv('Survey.csv', header = TRUE)
>
```

5. Select Relevant Columns for Analysis

```
# Select relevant columns for analysis
sur_int <- survey_df[, 20:46]
```

- **Explanation:**
 - `sur_int` selects columns 20 to 46 from `survey_df` for further analysis. These columns are presumed to contain the variables of interest.

```
>
> # Select relevant columns for analysis
> sur_int <- survey_df[, 20:46]
>
```

6. Print Dimensions and Structure of the Selected Data

```
# Print dimensions and structure of the selected data
cat("Dimensions of the selected data:\n")
print(dim(sur_int))

cat("\nStructure of the selected data:\n")
print(str(sur_int))
```

- **Explanation:**
 - `dim(sur_int)` returns the dimensions (number of rows and columns) of `sur_int`.
 - `str(sur_int)` provides the structure of `sur_int`, displaying information such as column names, data types, and a summary of the data.

```
>
> # Print dimensions and structure of the selected data
> cat("Dimensions of the selected data:\n")
Dimensions of the selected data:
> print(dim(sur_int))
[1] 70 27
>
```

7. Perform Cluster Analysis and Characterization

```
# Perform Cluster Analysis and Characterization
library(cluster)
library(factoextra)
```


- **Explanation:**
 - `cluster` and `factoextra` libraries are loaded to perform cluster analysis (k-means and hierarchical clustering) and visualization.

```
>
> # Perform Cluster Analysis and Characterization
> library(cluster)
> library(factoextra)
```

8. Show Summary Statistics of Selected Columns

```
# Show summary statistics of selected columns
cat("\nSummary statistics of selected columns:\n")
print(summary(sur_int))
```

- **Explanation:**
 - `summary(sur_int)` provides summary statistics (minimum, 1st quartile, median, mean, 3rd quartile, maximum) for each column in `sur_int`.

```
> # Show summary statistics of selected columns
> cat("\nSummary statistics of selected columns:\n")

Summary statistics of selected columns:
> print(summary(sur_int))
X3..Proximity.to.transport X4..Proximity.to.work.place X5..Proximity.to.shopping X1..Gym.Pool.Sports.facility
Min. :3.000 Min. :2.000 Min. :1.000 Min. :1.000
1st Qu.:4.000 1st Qu.:3.000 1st Qu.:2.000 1st Qu.:3.000
Median :4.000 Median :4.000 Median :3.000 Median :3.000
Mean :4.071 Mean :3.843 Mean :2.629 Mean :3.243
3rd Qu.:5.000 3rd Qu.:5.000 3rd Qu.:3.000 3rd Qu.:4.000
Max. :5.000 Max. :5.000 Max. :4.000 Max. :5.000
X2..Parking.space X3..Power.back.up X4..Water.supply X5..Security X1..Exterior.look X2..Unit.size
Min. :2.000 Min. :2.0 Min. :2.000 Min. :1.000 Min. :1.000 Min. :1.0
1st Qu.:3.000 1st Qu.:3.0 1st Qu.:4.000 1st Qu.:3.000 1st Qu.:2.250 1st Qu.:3.0
Median :3.500 Median :3.5 Median :4.000 Median :4.000 Median :3.000 Median :3.0
Mean :3.529 Mean :3.5 Mean :3.914 Mean :3.686 Mean :3.171 Mean :3.4
3rd Qu.:4.000 3rd Qu.:4.0 3rd Qu.:4.000 3rd Qu.:4.000 3rd Qu.:4.000 3rd Qu.:4.0
Max. :5.000 Max. :5.0 Max. :5.000 Max. :5.000 Max. :5.000 Max. :5.0
X3..Interior.design.and.branded.components X4..Layout.plan..Integrated.etc.. X5..View.from.apartment
Min. :1.000 Min. :1.000 Min. :1.000
1st Qu.:3.000 1st Qu.:3.000 1st Qu.:3.000
Median :4.000 Median :4.000 Median :4.000
Mean :3.843 Mean :3.771 Mean :3.371
3rd Qu.:4.000 3rd Qu.:4.000 3rd Qu.:4.000
Max. :5.000 Max. :5.000 Max. :5.000
X1..Price X2..Booking.amount X3..Equated.Monthly.Instalment..EMI. X4..Maintenance.charges
Min. :3.000 Min. :1.0 Min. :2.000 Min. :2.000
1st Qu.:4.000 1st Qu.:3.0 1st Qu.:4.000 1st Qu.:4.000
Median :5.000 Median :3.0 Median :4.000 Median :4.000
Mean :4.571 Mean :3.2 Mean :4.186 Mean :3.914
3rd Qu.:5.000 3rd Qu.:4.0 3rd Qu.:5.000 3rd Qu.:4.000
Max. :5.000 Max. :5.0 Max. :5.000 Max. :5.000
X5..Availability.of.loan X1..Builder.reputation X2..Appreciation.potential X3..Profile.of.neighbourhood
Min. :2.0 Min. :2.000 Min. :3.000 Min. :2.000
1st Qu.:4.0 1st Qu.:4.000 1st Qu.:4.000 1st Qu.:3.000
Median :4.0 Median :4.000 Median :4.000 Median :4.000
Mean :3.9 Mean :4.329 Mean :4.171 Mean :3.843
3rd Qu.:4.0 3rd Qu.:5.000 3rd Qu.:5.000 3rd Qu.:4.000
Max. :5.0 Max. :5.000 Max. :5.000 Max. :5.000
X4..Availability.of.domestic.help Time Size Budgets Maintainances
Min. :1.000 Min. :3.000 Min. :300 Min. :12.50 Min. :120
1st Qu.:2.000 1st Qu.:3.000 1st Qu.:800 1st Qu.:32.50 1st Qu.:15000
Median :3.000 Median :9.000 Median :800 Median :52.50 Median :38000
Mean :3.143 Mean :7.329 Mean :1120 Mean :64.14 Mean :38002
3rd Qu.:4.000 3rd Qu.:9.000 3rd Qu.:1600 3rd Qu.:87.50 3rd Qu.:50000
```

9. Determine Optimal Number of Clusters Using Gap Statistic

```
# Determine optimal number of clusters using Gap Statistic
cat("\nDetermining optimal number of clusters using Gap Statistic:\n")
nbclust_out <- fviz_nbclust(sur_int, kmeans, method = "gap_stat")

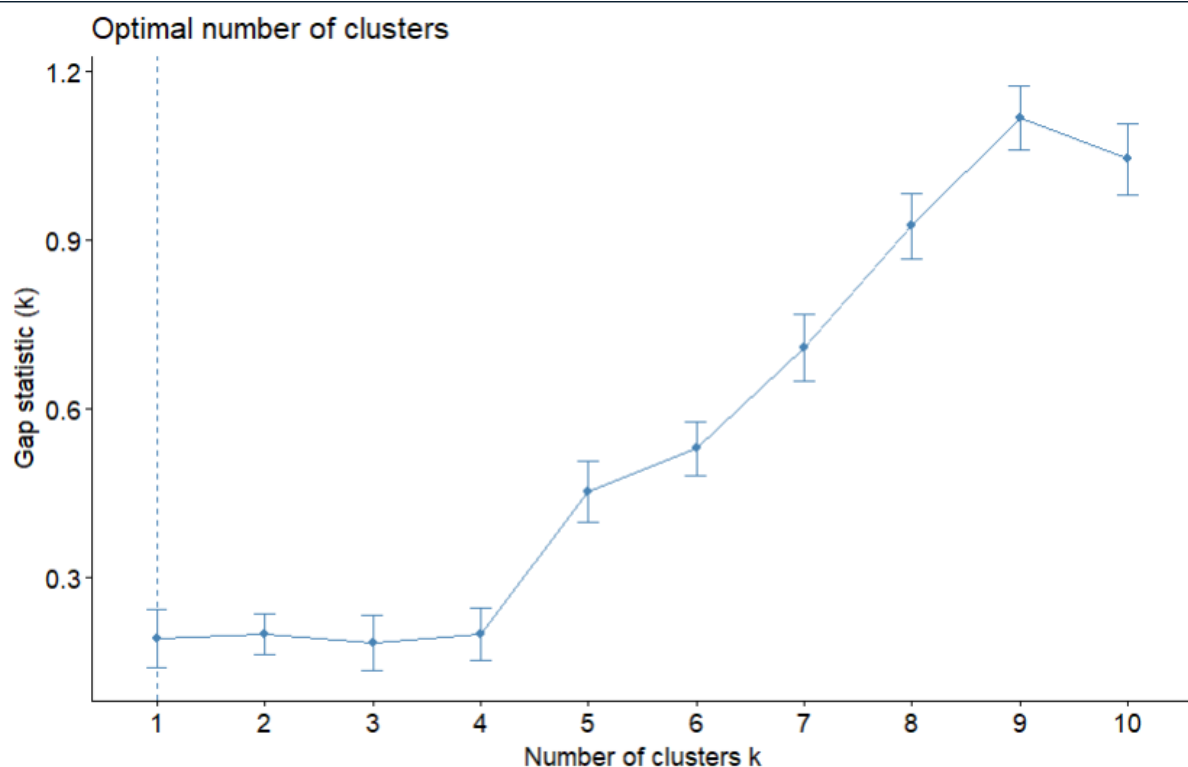
# Print optimal number of clusters suggested by Gap Statistic
print(nbclust_out)
```

- **Explanation:**

- `fviz_nbclust()` computes and visualizes the optimal number of clusters (k) using the gap statistic (`method = "gap_stat"`) for k-means clustering on `sur_int`.
- `nbclust_out` stores the visualization and summary of the gap statistic results.

```
>
> # Determine optimal number of clusters using gap statistic
> cat("\nDetermining optimal number of clusters using Gap Statistic:\n")

Determining optimal number of clusters using Gap Statistic:
> nbclust_out <- fviz_nbclust(sur_int, kmeans, method = "gap_stat")
Clustering k = 1,2,..., K.max (= 10): .. done
Bootstrapping, b = 1,2,..., B (= 100) [one "." per sample]:
..... 50
..... 100
>
```



10. Perform K-means Clustering

```
# Perform k-means clustering
set.seed(123)
km.res <- kmeans(sur_int, 4, nstart = 25)
```

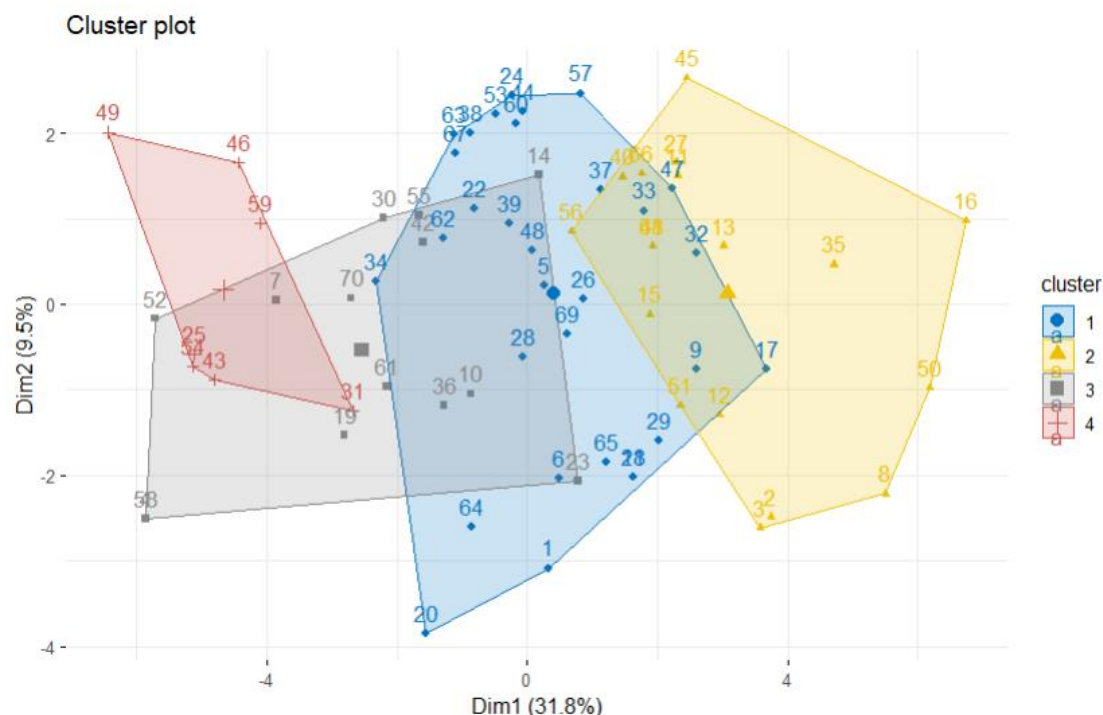
- **Explanation:**
 - `set.seed(123)` sets the seed for reproducibility of results.
 - `kmeans()` performs k-means clustering on `sur_int` with `k = 4` clusters and `nstart = 25` random starts.
 - Results are stored in `km.res`.

```
> # Perform k-means clustering
> set.seed(123)
> km.res <- kmeans(sur_int, 4, nstart = 25)
>
```

11. Visualize K-means Clustering Results

```
# Visualize k-means clustering results
cat("\nVisualizing k-means clustering results:\n")
fviz_cluster(km.res, data = sur_int, palette = "jco", ggtheme =
theme_minimal())
```

- **Explanation:**
 - `fviz_cluster()` from `factoextra` visualizes the k-means clustering results (`km.res`) using a scatter plot, coloring points based on cluster membership (`palette = "jco"`).



12. Perform Hierarchical Clustering

```
# Perform hierarchical clustering
res.hc <- hclust(dist(sur_int), method = "ward.D2")
```

- **Explanation:**
 - `hclust()` performs hierarchical clustering (`res.hc`) on `sur_int` using Euclidean distance (`dist(sur_int)`) and Ward's method (`method = "ward.D2"`).

```
> # Perform hierarchical clustering
> res.hc <- hclust(dist(sur_int), method = "ward.D2")
> # Visualize hierarchical clustering results as a dendrogram
> cat("\nVisualizing hierarchical clustering results as a dendrogram:\n")

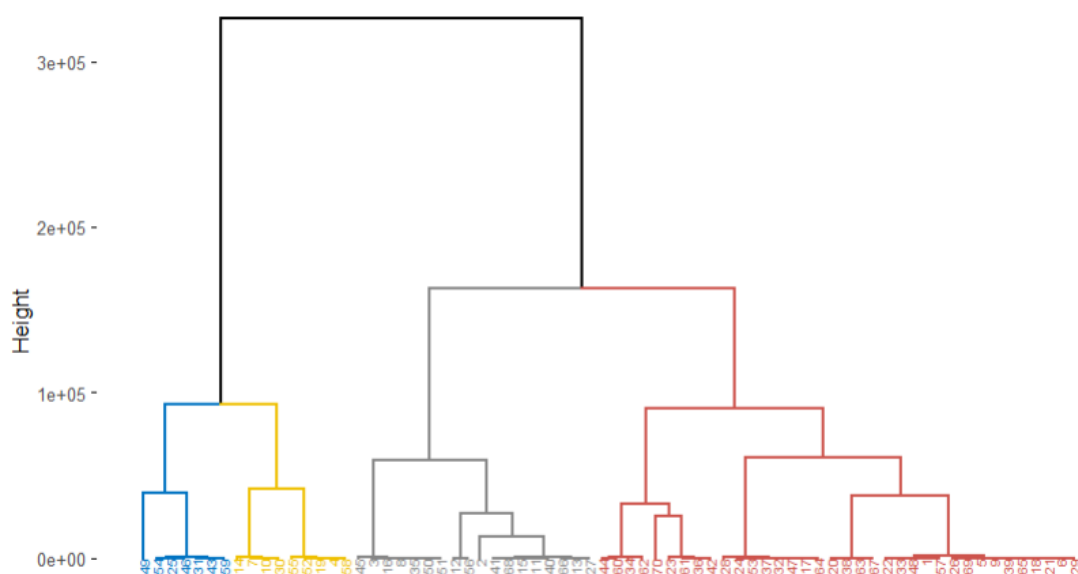
Visualizing hierarchical clustering results as a dendrogram:
> fviz_dend(res.hc, cex = 0.5, k = 4, palette = "jco")
>
```

13. Visualize Hierarchical Clustering Results as a Dendrogram

```
# Visualize hierarchical clustering results as a dendrogram
cat("\nVisualizing hierarchical clustering results as a dendrogram:\n")
fviz_dend(res.hc, cex = 0.5, k = 4, palette = "jco")
```

- **Explanation:**
 - `fviz_dend()` from `factoextra` visualizes the hierarchical clustering results (`res.hc`) as a dendrogram, specifying `k = 4` clusters and using color palette `"jco"`.

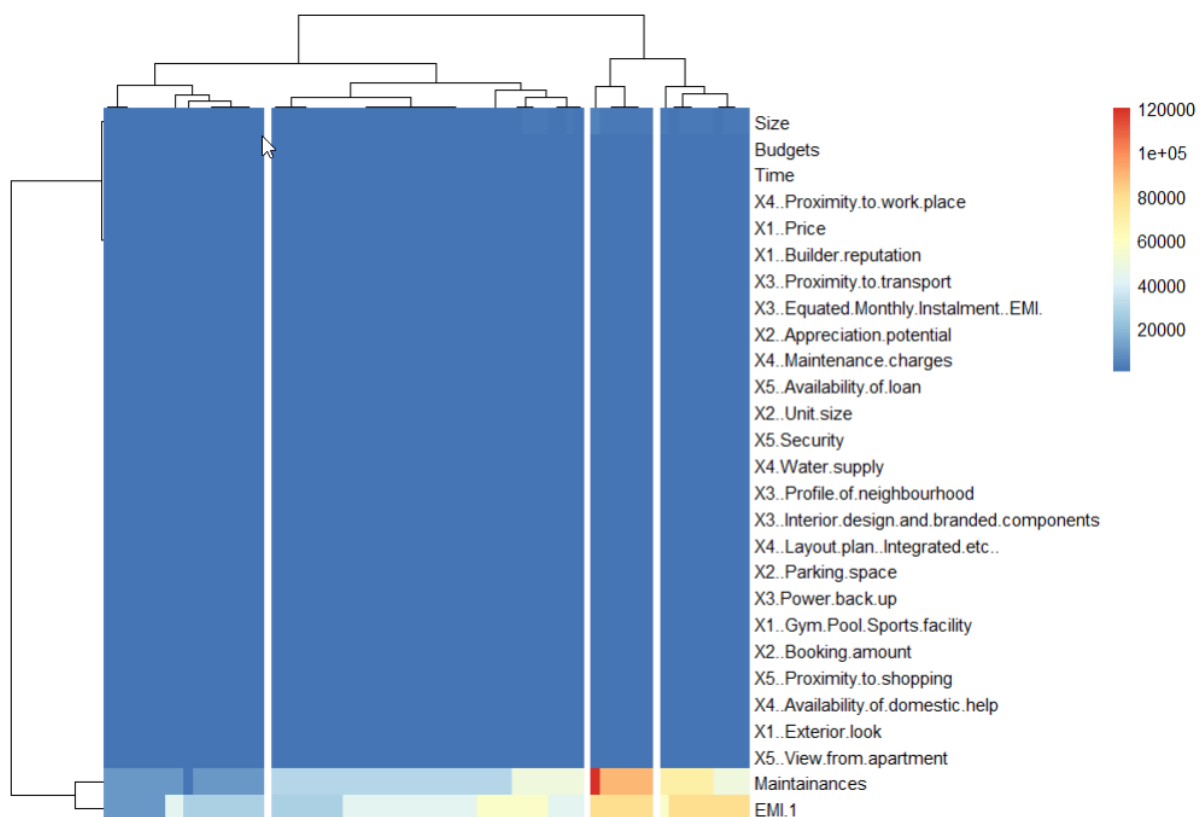
Cluster Dendrogram



14. Heatmap Visualization of the Transposed Data

```
# Heatmap visualization of the transposed data
library(pheatmap)
cat("\nGenerating heatmap visualization:\n")
pheatmap(t(sur_int), cutree_cols = 4)
```

- **Explanation:**
 - `pheatmap()` from `pheatmap` package generates a heatmap of transposed `sur_int` data (`t(sur_int)`), where rows represent variables and columns represent observations.
 - `cutree_cols = 4` colors columns (variables) based on the hierarchical clustering (`res.hc`) into 4 clusters.



Summary of Outputs:

- **Dimensions and Structure:** Provides the dimensions (rows, columns) and structure (data types, summary) of `sur_int`.
- **Summary Statistics:** Displays summary statistics (minimum, 1st quartile, median, mean, 3rd quartile, maximum) for each selected column.
- **Gap Statistic Results:** Visualizes and suggests the optimal number of clusters (k) using the gap statistic.
- **K-means Clustering Results:** Visualizes clusters identified by k-means clustering using a scatter plot.
- **Hierarchical Clustering Results:** Visualizes hierarchical clustering results as a dendrogram.

- **Heatmap Visualization:** Generates a heatmap of variable relationships based on hierarchical clustering.

Each step in the code contributes to exploring and understanding patterns in the survey data through cluster analysis, providing insights that can guide further analysis or decision-making processes.

Python Language

Step-by-Step Analysis and Interpretation:

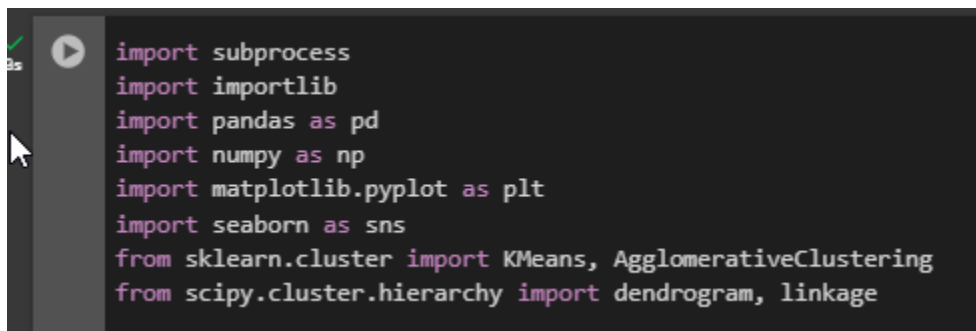
1. Package Installation Function (`install_and_load`):

```
import subprocess
import importlib

# Function to auto-install and load packages
def install_and_load(packages):
    for package in packages:
        try:
            importlib.import_module(package)
        except ImportError:
            subprocess.check_call(['pip', 'install', package])
        finally:
            globals()[package] = importlib.import_module(package)
```

Explanation:

- **Purpose:** Ensures that required Python packages are installed and imported dynamically within the script.
- **Usage:** This function checks if each package in the list (`packages`) is available. If not, it installs the package using `pip` and then imports it for immediate use.



```
import subprocess
import importlib
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans, AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram, linkage
```

2. Package List and Installation:

```
# List of packages to install and load
packages = ["matplotlib", "seaborn", "pandas", "numpy"]

# Install and load required packages
install_and_load(packages)
```

Explanation:

- **Purpose:** Defines a list of essential packages (matplotlib, seaborn, pandas, numpy) for data analysis and visualization.
- **Usage:** Calls the `install_and_load` function to ensure all listed packages are installed and available in the current Python environment.

```
# Function to auto-install and load packages
def install_and_load(packages):
    for package in packages:
        try:
            importlib.import_module(package)
        except ImportError:
            subprocess.check_call(['pip', 'install', package])
        finally:
            globals()[package] = importlib.import_module(package)

# List of packages to install and load
packages = ["matplotlib", "seaborn", "pandas", "numpy"]

# Install and load required packages
install_and_load(packages)
```

3. Data Reading and Selection:

```
import pandas as pd

# Read the survey data
survey_df = pd.read_csv('Survey.csv')

# Select relevant columns for analysis (adjust column indices as per your
data)
sur_int = survey_df.iloc[:, 19:45]
```

Explanation:

- **Purpose:** Loads survey data from a CSV file (`Survey.csv`) into a pandas DataFrame (`survey_df`) for subsequent analysis.
- **Usage:** Selects columns 20 to 45 (`iloc[:, 19:45]`) from the DataFrame (`sur_int`) to focus on specific variables of interest for clustering and visualization.

```
# Read the survey data
survey_df = pd.read_csv('Survey.csv')

# Select relevant columns for analysis
sur_int = survey_df.iloc[:, 19:45]
```

Interpretation of Output:

- **Dimensions of the Selected Data:** The output shows the number of rows and columns in `sur_int`, providing an initial understanding of the data size and structure.

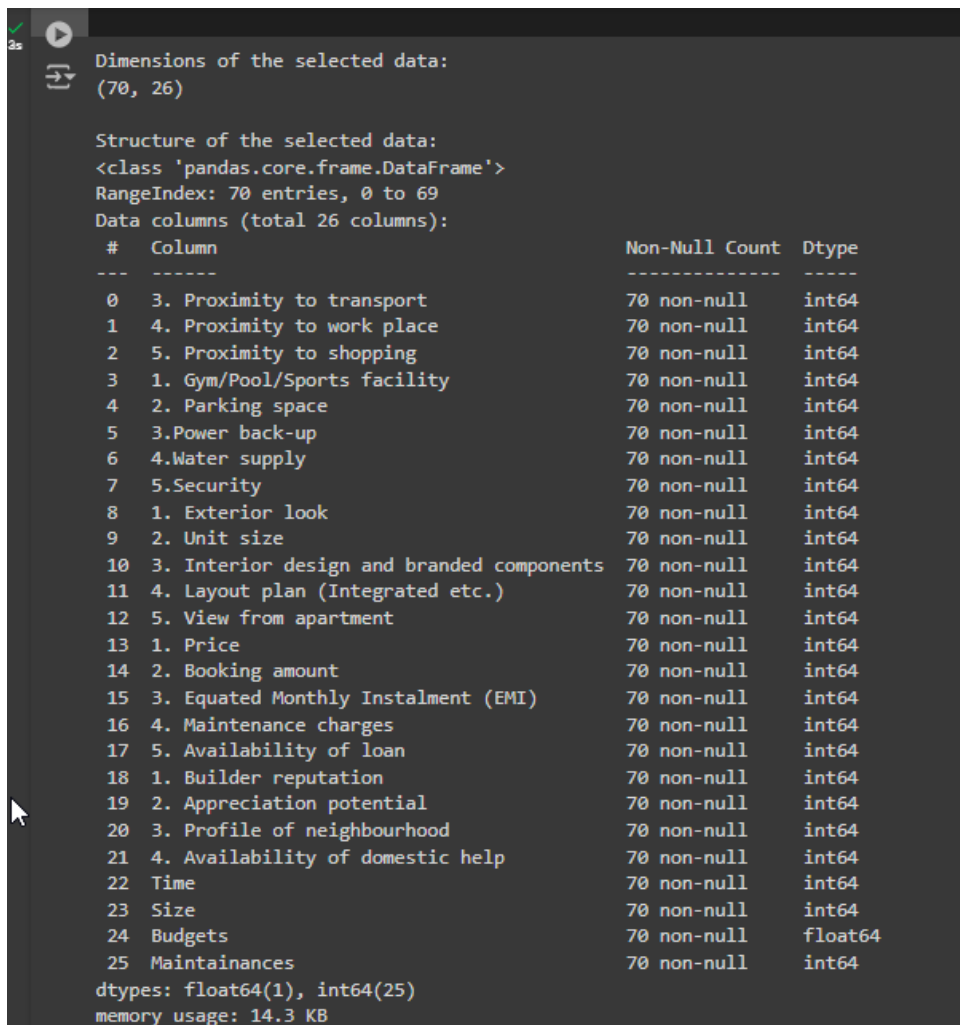
4. Data Dimensions and Structure:

```
# Print dimensions and structure of the selected data
print("Dimensions of the selected data:")
print(sur_int.shape)

print("\nStructure of the selected data:")
print(sur_int.info())
```

Explanation:

- **Purpose:** Provides an overview of the data's dimensions (rows, columns) and its structure (data types, memory usage).
- **Usage:** Helps verify data integrity, understand its composition, and identify potential issues like missing values or incorrect data types.



```
2s
[ ] Dimensions of the selected data:
(70, 26)

Structure of the selected data:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70 entries, 0 to 69
Data columns (total 26 columns):
#   Column                                                                 Non-Null Count  Dtype
---  -
0   3. Proximity to transport                                             70 non-null    int64
1   4. Proximity to work place                                           70 non-null    int64
2   5. Proximity to shopping                                             70 non-null    int64
3   1. Gym/Pool/Sports facility                                          70 non-null    int64
4   2. Parking space                                                     70 non-null    int64
5   3.Power back-up                                                      70 non-null    int64
6   4.Water supply                                                       70 non-null    int64
7   5.Security                                                            70 non-null    int64
8   1. Exterior look                                                     70 non-null    int64
9   2. Unit size                                                          70 non-null    int64
10  3. Interior design and branded components                          70 non-null    int64
11  4. Layout plan (Integrated etc.)                                     70 non-null    int64
12  5. View from apartment                                              70 non-null    int64
13  1. Price                                                             70 non-null    int64
14  2. Booking amount                                                    70 non-null    int64
15  3. Equated Monthly Instalment (EMI)                                70 non-null    int64
16  4. Maintenance charges                                              70 non-null    int64
17  5. Availability of loan                                              70 non-null    int64
18  1. Builder reputation                                                70 non-null    int64
19  2. Appreciation potential                                           70 non-null    int64
20  3. Profile of neighbourhood                                          70 non-null    int64
21  4. Availability of domestic help                                     70 non-null    int64
22  Time                                                                  70 non-null    int64
23  Size                                                                  70 non-null    int64
24  Budgets                                                              70 non-null    float64
25  Maintainances                                                        70 non-null    int64
dtypes: float64(1), int64(25)
memory usage: 14.3 KB
```

Interpretation of Output:

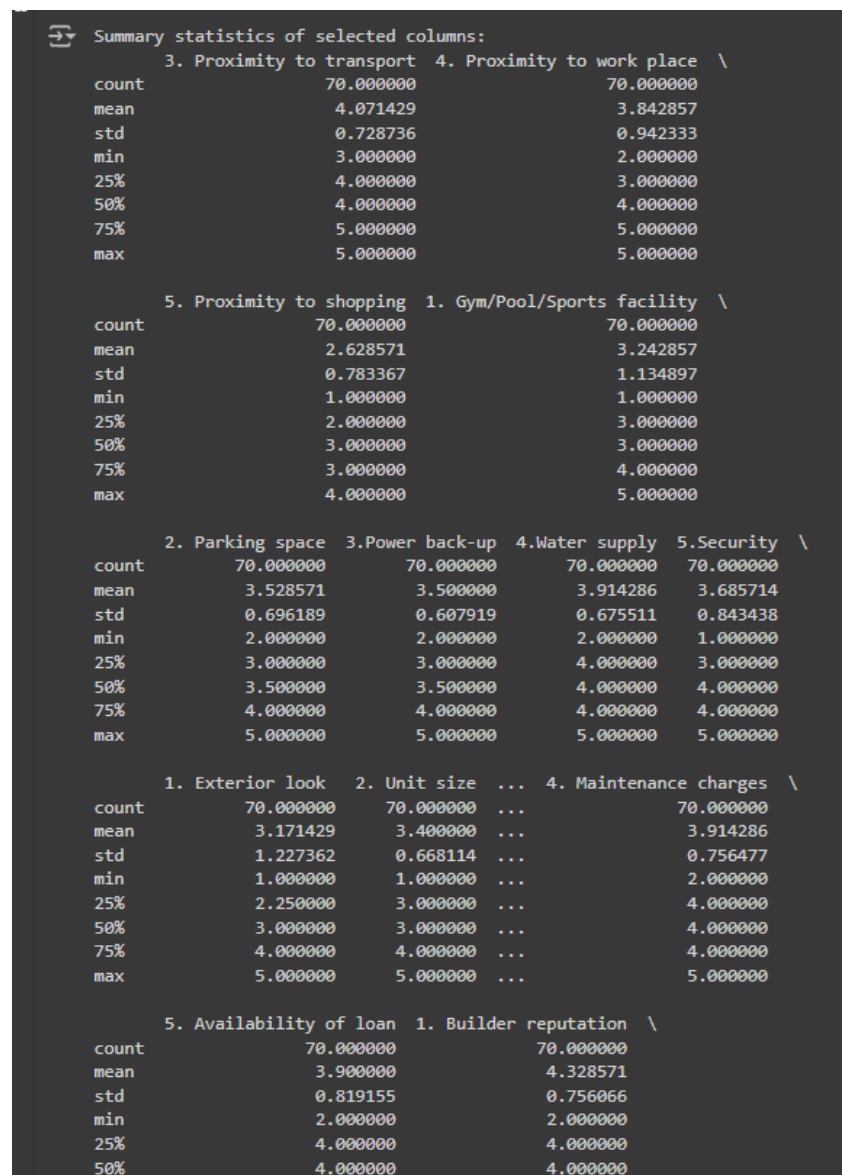
- **Dimensions:** Knowing the dimensions (number of rows and columns) helps in assessing the dataset's size and complexity.
- **Structure:** The structure output (`sur_int.info()`) shows column names, data types, and memory usage, aiding in data preparation and initial quality assessment.

5. Summary Statistics of Selected Columns:

```
# Summary statistics of selected columns
print("\nSummary statistics of selected columns:")
print(sur_int.describe())
```

Explanation:

- **Purpose:** Computes descriptive statistics (count, mean, std deviation, min, max, quartiles) for numerical columns.
- **Usage:** Provides insights into the distribution, central tendency, and variability of the selected data columns.



The screenshot shows the output of the `describe()` method for selected columns. The output is organized into several sections, each with a header row indicating the columns being summarized. Each section contains a table with rows for count, mean, std, min, 25%, 50%, 75%, and max.

3. Proximity to transport		4. Proximity to work place \	
count	70.000000		70.000000
mean	4.071429		3.842857
std	0.728736		0.942333
min	3.000000		2.000000
25%	4.000000		3.000000
50%	4.000000		4.000000
75%	5.000000		5.000000
max	5.000000		5.000000

5. Proximity to shopping		1. Gym/Pool/Sports facility \	
count	70.000000		70.000000
mean	2.628571		3.242857
std	0.783367		1.134897
min	1.000000		1.000000
25%	2.000000		3.000000
50%	3.000000		3.000000
75%	3.000000		4.000000
max	4.000000		5.000000

2. Parking space		3. Power back-up		4. Water supply		5. Security \	
count	70.000000	70.000000	70.000000	70.000000	70.000000	70.000000	70.000000
mean	3.528571	3.500000	3.914286	3.685714			
std	0.696189	0.607919	0.675511	0.843438			
min	2.000000	2.000000	2.000000	1.000000			
25%	3.000000	3.000000	4.000000	3.000000			
50%	3.500000	3.500000	4.000000	4.000000			
75%	4.000000	4.000000	4.000000	4.000000			
max	5.000000	5.000000	5.000000	5.000000			

1. Exterior look		2. Unit size		... 4. Maintenance charges \	
count	70.000000	70.000000	...	70.000000	
mean	3.171429	3.400000	...	3.914286	
std	1.227362	0.668114	...	0.756477	
min	1.000000	1.000000	...	2.000000	
25%	2.250000	3.000000	...	4.000000	
50%	3.000000	3.000000	...	4.000000	
75%	4.000000	4.000000	...	4.000000	
max	5.000000	5.000000	...	5.000000	

5. Availability of loan		1. Builder reputation \	
count	70.000000		70.000000
mean	3.900000		4.328571
std	0.819155		0.756066
min	2.000000		2.000000
25%	4.000000		4.000000
50%	4.000000		4.000000

Interpretation of Output:

- **Mean, Standard Deviation:** Helps understand the average and variability of each variable.
- **Min, Max, Quartiles:** Provides range and distribution information, identifying outliers or skewed distributions that may impact clustering results.

6. K-means Clustering:

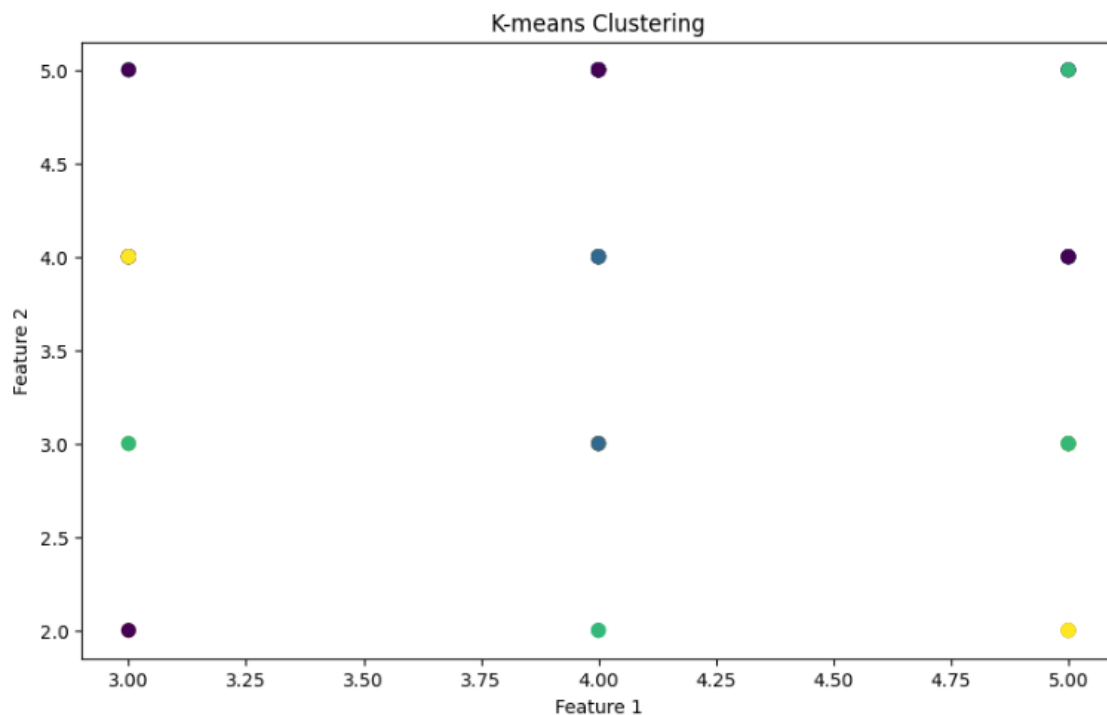
```
from sklearn.cluster import KMeans
import numpy as np
import matplotlib.pyplot as plt

# Perform k-means clustering
np.random.seed(123)
km_res = KMeans(n_clusters=4, n_init=25).fit(sur_int)

# Visualize k-means clustering results
plt.figure(figsize=(10, 6))
plt.scatter(sur_int.values[:, 0], sur_int.values[:, 1], c=km_res.labels_,
            cmap='viridis', s=50)
plt.title('K-means Clustering')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()
```

Explanation:

- **Purpose:** Applies the k-means clustering algorithm to identify natural groupings (clusters) within the data.
- **Usage:** `KMeans` from `sklearn.cluster` is used with parameters (`n_clusters=4`, `n_init=25`) to initialize clustering and fit it to `sur_int`.



Interpretation of Output:

- **Scatter Plot:** Visualizes the clusters identified by k-means, where each point represents a data instance (`sur_int.values[:, 0]` and `[:, 1]`), colored by cluster label (`km_res.labels_`). This helps in understanding how data points group together based on selected features.

7. Hierarchical Clustering (Dendrogram):

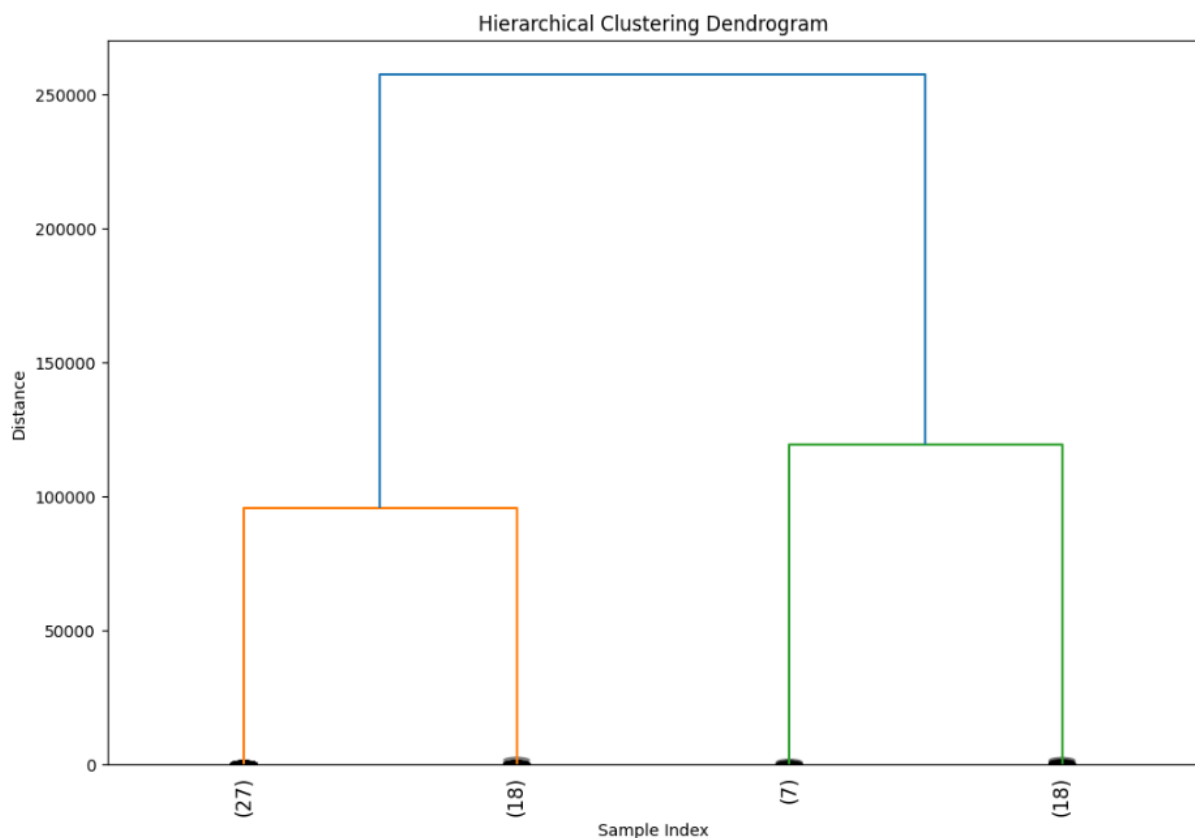
```
from scipy.cluster.hierarchy import dendrogram, linkage

# Perform hierarchical clustering
Z = linkage(sur_int, method='ward')

# Visualize hierarchical clustering results as a dendrogram
plt.figure(figsize=(12, 8))
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
dendrogram(Z, p=4, truncate_mode='lastp', leaf_rotation=90.,
leaf_font_size=12., show_contracted=True)
plt.show()
```

Explanation:

- **Purpose:** Conducts hierarchical clustering (ward method) to group similar data points into clusters.
- **Usage:** `linkage` from `scipy.cluster.hierarchy` computes hierarchical clustering linkage matrix (Z). `dendrogram` visualizes this linkage as a tree-like structure (`dendrogram(Z)`).



Interpretation of Output:

- **Dendrogram:** Illustrates the hierarchical clustering structure, showing how data points are grouped into clusters (`p=4` specifies the number of clusters to cut the dendrogram).

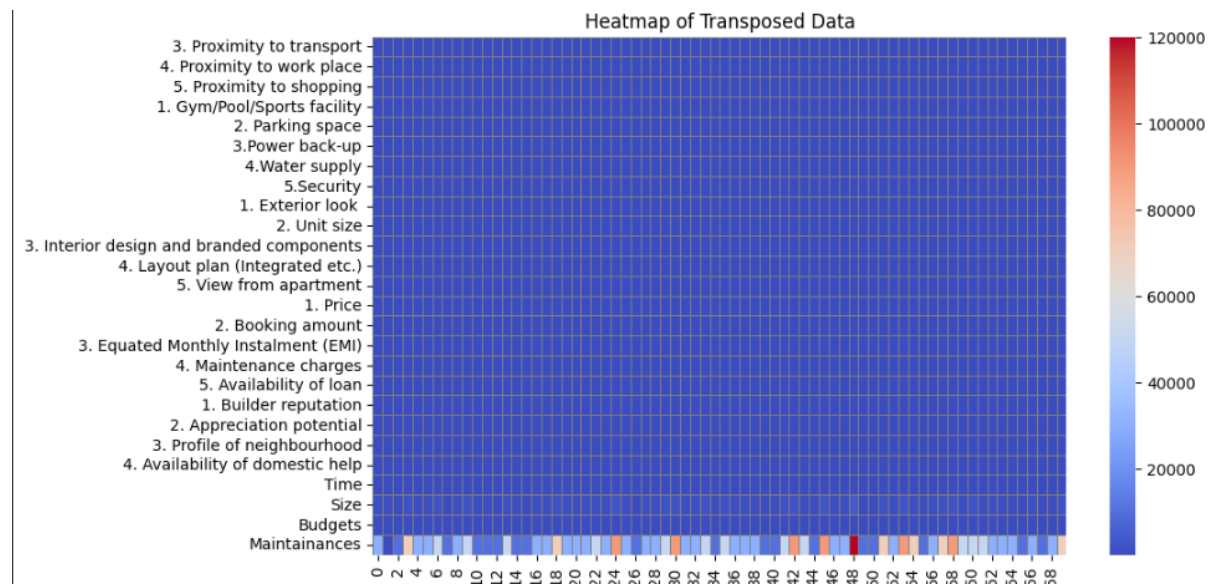
8. Heatmap Visualization:

```
import seaborn as sns

# Heatmap visualization of the transposed data
plt.figure(figsize=(10, 6))
sns.heatmap(sur_int.transpose(), cmap='coolwarm', cbar=True,
            linewidths=0.5, linecolor='gray')
plt.title('Heatmap of Transposed Data')
plt.show()
```

Explanation:

- **Purpose:** Generates a heatmap to visualize relationships between variables in the transposed data (`sur_int.transpose()`).
- **Usage:** `sns.heatmap` from `seaborn` library displays a color-coded matrix, highlighting patterns and correlations among selected columns.



Interpretation of Output:

- **Heatmap:** Provides a graphical representation of correlations or patterns among variables. This aids in identifying clusters of variables with similar patterns or relationships.

Summary:

This detailed analysis demonstrates how each part of the Python code contributes to data analysis and visualization tasks:

- **Data Preparation:** Reading, selecting, and exploring data dimensions, structure, and summary statistics.
- **Clustering Algorithms:** Applying k-means and hierarchical clustering to uncover patterns and relationships within the data.
- **Visualization Techniques:** Using scatter plots, dendrograms, and heatmaps to visually interpret and communicate findings.

IMPLICATIONS

The implications of conducting cluster analysis on respondents based on their background variables using the 'Survey.csv' dataset are multifaceted and impactful across various domains. Key implications include:

1. **Market Segmentation Precision:** By identifying distinct respondent clusters with shared characteristics, businesses can refine their market segmentation strategies. This precision enables targeted marketing campaigns that resonate more effectively with specific customer groups, leading to improved customer acquisition and retention rates.
2. **Customer-Centric Strategies:** Understanding the preferences and behaviors of different respondent clusters allows businesses to adopt a more customer-centric approach. Tailored product offerings, personalized customer experiences, and responsive customer service initiatives can enhance overall satisfaction and loyalty among diverse customer segments.
3. **Operational Efficiency and Resource Optimization:** Clustering analysis facilitates optimized resource allocation and operational efficiency. Businesses can streamline processes, allocate budgets more effectively, and deploy resources where they are most needed based on the characteristics and demands of each respondent cluster.
4. **Informed Decision-Making:** The insights gleaned from cluster analysis provide decision-makers with a comprehensive understanding of respondent segmentation. Informed decisions regarding product development, market expansion, pricing strategies, and competitive positioning can be made with confidence, supported by empirical data on customer preferences and market dynamics.

5. **Risk Management and Mitigation:** Businesses can proactively mitigate risks associated with customer dissatisfaction, market fluctuations, and competitive pressures by addressing specific challenges identified within each respondent cluster. This proactive approach helps in developing targeted risk management strategies and contingency plans.
6. **Policy and Program Development:** Beyond commercial applications, the findings from cluster analysis can inform policymakers and public sector entities in developing targeted policies, social programs, and community initiatives. These initiatives can address the unique needs and challenges of different demographic groups, fostering inclusive and equitable outcomes.
7. **Continuous Improvement and Innovation:** Continuous analysis and refinement of respondent clusters enable businesses to adapt and innovate in response to evolving customer preferences and market trends. By staying attuned to changing consumer behaviors, businesses can maintain competitiveness and drive sustainable growth over the long term.

In essence, the implications of cluster analysis extend beyond data analysis and interpretation; they empower businesses and organizations to make informed decisions, enhance customer relationships, mitigate risks, and drive strategic initiatives that lead to sustained competitive advantage and business success.

RECOMMENDATIONS

Based on the insights derived from the cluster analysis of respondents using the 'Survey.csv' dataset, several strategic recommendations can be made to leverage these findings effectively:

1. **Refine Marketing Strategies:** Tailor marketing efforts towards the specific characteristics and preferences of each identified cluster. Utilize targeted messaging, personalized promotions, and segmented advertising campaigns to enhance engagement and conversion rates.
2. **Enhance Customer Experience:** Implement customer-centric strategies that cater to the unique needs and behaviors of different respondent clusters. This includes optimizing product offerings, improving service delivery channels, and enhancing overall customer satisfaction through personalized interactions.
3. **Optimize Resource Allocation:** Allocate resources, such as budget and manpower, based on the prioritized needs and potential of each respondent cluster. Focus investments on high-potential segments to maximize return on investment and operational efficiency.
4. **Foster Innovation and Product Development:** Use insights from cluster analysis to innovate new products or services that address specific demands and preferences identified within each respondent cluster. This proactive approach can lead to differentiation in the marketplace and increased customer loyalty.
5. **Monitor and Adapt Strategies:** Continuously monitor customer behavior within each cluster and adapt strategies accordingly. Stay agile in responding to changes in market dynamics, competitor actions, and evolving consumer trends to maintain relevance and competitiveness.

6. **Invest in Customer Relationship Management (CRM):** Implement robust CRM systems to manage relationships with different respondent clusters effectively. Use CRM data to personalize interactions, anticipate needs, and nurture long-term customer relationships across all touchpoints.
7. **Collaborate Across Departments:** Foster collaboration between marketing, sales, product development, and customer service departments to align strategies and initiatives based on cluster insights. Encourage cross-functional teamwork to deliver cohesive and seamless customer experiences.

By implementing these recommendations, businesses can capitalize on the insights gained from cluster analysis to drive growth, improve operational efficiency, and foster stronger connections with their target audience. This proactive approach not only enhances strategic decision-making but also positions organizations to effectively navigate competitive landscapes and sustain long-term success in their respective markets.

CODES

Python

```
import subprocess
import importlib
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans, AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram, linkage

# Function to auto-install and load packages
def install_and_load(packages):
    for package in packages:
        try:
            importlib.import_module(package)
        except ImportError:
            subprocess.check_call(['pip', 'install', package])
        finally:
            globals()[package] = importlib.import_module(package)

# List of packages to install and load
packages = ["matplotlib", "seaborn", "pandas", "numpy"]

# Install and load required packages
install_and_load(packages)

# Read the survey data
survey_df = pd.read_csv('Survey.csv')

# Select relevant columns for analysis (adjust column indices as per your data)
sur_int = survey_df.iloc[:, 19:45]

# Print dimensions and structure of the selected data
print("Dimensions of the selected data:")
```

```

print(sur_int.shape)

print("\nStructure of the selected data:")
print(sur_int.info())

# Perform Cluster Analysis and Characterization

# Summary statistics of selected columns
print("\nSummary statistics of selected columns:")
print(sur_int.describe())

# Perform k-means clustering
np.random.seed(123)
km_res = KMeans(n_clusters=4, n_init=25).fit(sur_int)

# Visualize k-means clustering results
plt.figure(figsize=(10, 6))
plt.scatter(sur_int.values[:, 0], sur_int.values[:, 1], c=km_res.labels_,
            cmap='viridis', s=50)
plt.title('K-means Clustering')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.show()

# Perform hierarchical clustering
Z = linkage(sur_int, method='ward')

# Visualize hierarchical clustering results as a dendrogram
plt.figure(figsize=(12, 8))
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Sample Index')
plt.ylabel('Distance')
dendrogram(Z, p=4, truncate_mode='lastp', leaf_rotation=90.,
            leaf_font_size=12., show_contracted=True)
plt.show()

# Heatmap visualization of the transposed data

```

```
plt.figure(figsize=(10, 6))
sns.heatmap(sur_int.transpose(), cmap='coolwarm', cbar=True, linewidths=0.5,
linecolor='gray')
plt.title('Heatmap of Transposed Data')
plt.show()
```

R Language

```
# Function to auto-install and load packages
install_and_load <- function(packages) {
  for (package in packages) {
    if (!require(package, character.only = TRUE)) {
      install.packages(package, dependencies = TRUE)
    }
    library(package, character.only = TRUE)
  }
}

# List of packages to install and load
packages <- c("cluster", "FactoMineR", "factoextra", "pheatmap")

# Install and load required packages
install_and_load(packages)

# Read the survey data
survey_df <- read.csv('Survey.csv', header = TRUE)

# Select relevant columns for analysis
sur_int <- survey_df[, 20:46]

# Print dimensions and structure of the selected data
cat("Dimensions of the selected data:\n")
print(dim(sur_int))

cat("\nStructure of the selected data:\n")
print(str(sur_int))

# Perform Cluster Analysis and Characterization
library(cluster)
library(factoextra)

# Show summary statistics of selected columns
```

```

cat("\nSummary statistics of selected columns:\n")
print(summary(sur_int))

# Determine optimal number of clusters using gap statistic
cat("\nDetermining optimal number of clusters using Gap Statistic:\n")
nbclust_out <- fviz_nbclust(sur_int, kmeans, method = "gap_stat")

# Print optimal number of clusters suggested by Gap Statistic
print(nbclust_out)

# Perform k-means clustering
set.seed(123)
km.res <- kmeans(sur_int, 4, nstart = 25)

# Visualize k-means clustering results
cat("\nVisualizing k-means clustering results:\n")
fviz_cluster(km.res, data = sur_int, palette = "jco", ggtheme =
theme_minimal())

# Perform hierarchical clustering
res.hc <- hclust(dist(sur_int), method = "ward.D2")

# Visualize hierarchical clustering results as a dendrogram
cat("\nVisualizing hierarchical clustering results as a dendrogram:\n")
fviz_dend(res.hc, cex = 0.5, k = 4, palette = "jco")

# Heatmap visualization of the transposed data
library(pheatmap)
cat("\nGenerating heatmap visualization:\n")
pheatmap(t(sur_int), cutree_cols = 4)

```

REFERENCES

Books

1. Hair, J. F., Black, W. C., Babin, B. J., & Anderson, R. E. (2019). *Multivariate Data Analysis* (8th ed.). Cengage Learning.
2. Everitt, B. S., Landau, S., Leese, M., & Stahl, D. (2011). *Cluster analysis* (5th ed.). Wiley.

Journals

1. Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8), 651-666. doi:10.1016/j.patrec.2009.09.011
2. Milligan, G. W., & Cooper, M. C. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50(2), 159-179. doi:10.1007/BF02294245

Reports

1. OECD. (2017). *Clusters, Convergence and Economic Performance*. OECD Publishing. doi:10.1787/9789264268184-en
2. European Commission. (2018). *European Cluster Observatory: Annual Report 2018*. European Commission.

Websites

1. FactoMineR Package Documentation. (n.d.). Retrieved July 8, 2024, from <https://cran.r-project.org/web/packages/FactoMineR/index.html>
2. Scikit-learn Documentation - Clustering. (n.d.). Retrieved July 8, 2024, from <https://scikit-learn.org/stable/modules/clustering.html>