

VIRGINIA COMMONWEALTH UNIVERSITY



Statistical analysis and modelling (SCMA 632)

**A4: MULTIVARIATE ANALYSIS AND BUSINESS
ANALYTICS APPLICATION**

RIDDHI RUNGTA

V01107488

Date of Submission: 07-07-2024

CONTENTS

Content:	Page no:
INTRODUCTION	1
OBJECTIVE	1
BUSINESS SIGNIFICANCE	2
RESULTS AND INTERPRETATIONS	2-58

INTRODUCTION

Principal Component Analysis (PCA) is a dimensionality reduction technique that simplifies large datasets by transforming them into a smaller set of uncorrelated variables called principal components. Factor Analysis is a statistical method that identifies underlying relationships between variables by grouping them into factors, assuming that observed variables are influenced by a few underlying unobserved variables (factors). Conjoint Analysis is a survey-based statistical technique used in market research to determine how people value different attributes of a product or service.

The dataset "icecream.csv" likely includes information on ice cream sales, capturing various attributes such as flavors, sales figures, geographic locations, and time periods. This dataset is designed to help understand consumer preferences, sales trends, and market dynamics within the ice cream industry. Analyzing this data can provide businesses with valuable insights into popular products, seasonal trends, and opportunities to optimize marketing strategies to enhance sales and customer satisfaction.

The second dataset, "Survey.csv," probably contains survey responses, including demographic information, consumer opinions, and preferences. This data is essential for businesses to understand their customer base, gather feedback on products or services, and make data-driven decisions to improve customer experience and drive growth.

OBJECTIVES

- a. Perform Principal Component Analysis and Factor Analysis to identify data dimensions.
- b. Conduct Cluster Analysis to characterize respondents based on background variables.
- c. Apply Multidimensional Scaling and interpret the results.
- d. Conjoint Analysis

BUSINESS SIGNIFICANCE

Principal Component Analysis (PCA) and Factor Analysis serve distinct purposes in data analysis for business. PCA reduces data dimensionality while preserving variance, yielding uncorrelated variables that capture significant data trends. Conversely, Factor Analysis identifies underlying relationships among variables to model latent factors explaining correlations.

In business, these techniques simplify complex datasets, uncover hidden patterns, and inform strategic decisions. PCA and Factor Analysis help:

- **Reduce Complexity:** Streamline large datasets without losing critical information.
- **Identify Key Drivers:** Uncover factors driving consumer behavior or market trends.

- **Enhance Predictive Models:** Improve model performance by focusing on influential variables.
- **Inform Strategy:** Guide targeted marketing, product development, and customer segmentation.

Conjoint Analysis, on the other hand, delves into consumer preferences and trade-offs between product attributes. It quantifies attribute values and optimal feature combinations, crucial for:

- **Optimizing Product Design:** Tailoring products to consumer needs.
- **Enhancing Pricing Strategy:** Determining optimal price points and feature bundles.
- **Improving Market Segmentation:** Identifying consumer segments based on preferences.

These methodologies translate consumer insights into actionable strategies, driving business success through informed decision-making.

RESULTS AND INTERPRETATIONS USING PYTHON AND R

Code

Loading the dataset

```
import os
import pandas as pd

# Load the dataset
os.chdir("C:\\Users\\RIDDHI\\OneDrive\\Desktop\\Bootcamp\\SCMA\\SCMA A4")
survey_data = pd.read_csv('Survey.csv')

# Display the first few rows of the dataset
survey_data.head()
```

Installation of necessary libraries

```
pip install pandas numpy scikit-learn seaborn matplotlib scipy gapstat-rs
pip install pandas numpy factor_analyzer scikit-learn matplotlib
```

Objective 1:

Principal Component Analysis and Factor Analysis

First we started with performing some descriptive statistics on the data.

```
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
from factor_analyzer import FactorAnalyzer, Rotator
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Load the dataset
```

```
survey_df = pd.read_csv('Survey.csv')
```

```
# Display the dimensions of the dataset
```

```
print(survey_df.shape)
```

```
# Display the column names
```

```
print(survey_df.columns)
```

```
# Display the first few rows of the dataset
```

```
print(survey_df.head())
```

```
# Display the structure of the dataset
```

```
print(survey_df.info())
```

```
# Check for missing values
```

```
print(survey_df.isna().sum().sum())
```

```
# Select relevant columns for analysis
```

```
sur_int = survey_df.iloc[:, 19:46]
```

```
# Display the structure and dimensions of the selected data
```

```
print(sur_int.info())
```

```
print(sur_int.shape)
```

Result:

```
(70, 50)
```

```
Index(['City', 'Sex', 'Age', 'Occupation', 'Monthly Household Income',
      'Income', 'Planning to Buy a new house', 'Time Frame',
      'Reasons for buying a house', 'what type of House', 'Number of rooms',
      'Size of House', 'Budget', 'Finished/Semi Finished',
      'Influence Decision', 'Maintainance', 'EMI', '1.Proximity to city',
      '2.Proximity to schools', '3. Proximity to transport',
      '4. Proximity to work place', '5. Proximity to shopping',
      '1. Gym/Pool/Sports facility', '2. Parking space', '3.Power back-up',
      '4.Water supply', '5.Security', '1. Exterior look ', '2. Unit size',
      '3. Interior design and branded components',
      '4. Layout plan (Integrated etc.)', '5. View from apartment',
```

'1. Price', '2. Booking amount', '3. Equated Monthly Instalment (EMI)',
 '4. Maintenance charges', '5. Availability of loan',
 '1. Builder reputation', '2. Appreciation potential',
 '3. Profile of neighbourhood', '4. Availability of domestic help',
 'Time', 'Size', 'Budgets', 'Maintainances', 'EMI.1', 'ages', 'sex',
 'Finished/Semi Finished.1', 'Influence Decision.1'],
 dtype='object')

	City	Sex	Age	Occupation	Monthly Household Income	Income \
0	Bangalore	M	26-35	Private Sector	85,001 to105,000	95000
1	Bangalore	M	46-60	Government/PSU	45,001 to 65,000	55000
2	Bangalore	F	46-60	Government/PSU	25,001 to 45,000	35000
3	Bangalore	M	36-45	Private Sector	>125000	200000
4	Bangalore	M	26-35	Self Employed	85,001 to105,000	95000

	Planning to Buy a new house	Time Frame	Reasons for buying a house \
0	Yes	6M to 1Yr	Residing
1	Yes	6M to 1Yr	Investment
2	Yes	<6 Months	Rental Income
3	Yes	<6 Months	Investment
4	Yes	1-2 Yr	Residing

	what type of House ...	4. Availability of domestic help	Time	Size	Budgets \
0	Apartment ...	1	9	1200	72.5
1	Apartment ...	2	9	800	32.5
2	Apartment ...	4	3	400	12.5
3	Apartment ...	5	3	1600	102.5
4	Apartment ...	3	18	800	52.5

	Maintainances	EMI.1	ages	sex	Finished/Semi Finished.1 \
0	30000	42500	30.5	M	Semifurnished
1	120	27500	53.0	M	Semifurnished
2	10000	10000	53.0	F	Semifurnished
3	70000	80000	40.5	M	Furnished
4	30000	42500	30.5	M	Semifurnished

	Influence Decision.1
0	Site visits
1	Newspaper
2	Hoarding
3	Electronic/Internet
4	Electronic/Internet

[5 rows x 50 columns]

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 70 entries, 0 to 69

Data columns (total 50 columns):

#	Column	Non-Null Count	Dtype
0	City	70 non-null	object

1	Sex	70 non-null	object
2	Age	70 non-null	object
3	Occupation	70 non-null	object
4	Monthly Household Income	70 non-null	object
5	Income	70 non-null	int64
6	Planning to Buy a new house	70 non-null	object
7	Time Frame	70 non-null	object
8	Reasons for buying a house	70 non-null	object
9	what type of House	70 non-null	object
10	Number of rooms	70 non-null	object
11	Size of House	70 non-null	object
12	Budget	70 non-null	object
13	Finished/Semi Finished	70 non-null	object
14	Influence Decision	70 non-null	object
15	Maintainance	70 non-null	object
16	EMI	70 non-null	object
17	1.Proximity to city	70 non-null	int64
18	2.Proximity to schools	70 non-null	int64
19	3. Proximity to transport	70 non-null	int64
20	4. Proximity to work place	70 non-null	int64
21	5. Proximity to shopping	70 non-null	int64
22	1. Gym/Pool/Sports facility	70 non-null	int64
23	2. Parking space	70 non-null	int64
24	3.Power back-up	70 non-null	int64
25	4.Water supply	70 non-null	int64
26	5.Security	70 non-null	int64
27	1. Exterior look	70 non-null	int64
28	2. Unit size	70 non-null	int64
29	3. Interior design and branded components	70 non-null	int64
30	4. Layout plan (Integrated etc.)	70 non-null	int64
31	5. View from apartment	70 non-null	int64
32	1. Price	70 non-null	int64
33	2. Booking amount	70 non-null	int64
34	3. Equated Monthly Instalment (EMI)	70 non-null	int64
35	4. Maintenance charges	70 non-null	int64
36	5. Availability of loan	70 non-null	int64
37	1. Builder reputation	70 non-null	int64
38	2. Appreciation potential	70 non-null	int64
39	3. Profile of neighbourhood	70 non-null	int64
40	4. Availability of domestic help	70 non-null	int64
41	Time	70 non-null	int64
42	Size	70 non-null	int64
43	Budgets	70 non-null	float64
44	Maintainances	70 non-null	int64
45	EMI.1	70 non-null	int64
46	ages	70 non-null	float64
47	sex	70 non-null	object
48	Finished/Semi Finished.1	70 non-null	object
49	Influence Decision.1	70 non-null	object

dtypes: float64(2), int64(29), object(19)

memory usage: 27.5+ KB

None

0

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 70 entries, 0 to 69

Data columns (total 27 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
26	EMI.1	70 non-null	int64

dtypes: float64(1), int64(26)

Code: for Principal Component Analysis

Perform PCA

pca = PCA(n_components=5)

pca_result = pca.fit_transform(sur_int)

Display the explained variance by each principal component

print(pca.explained_variance_ratio_)

Biplot for PCA

plt.figure(figsize=(10, 7))

plt.scatter(pca_result[:, 0], pca_result[:, 1], edgecolors='k', c='r')

plt.xlabel('PC1')


```

plt.ylabel('PC2')
plt.title('PCA Biplot')
plt.grid(True)
plt.show()

# Perform Factor Analysis
fa = FactorAnalyzer(n_factors=5, rotation=None)
fa.fit(sur_int)

# Get loadings and variance
loadings = fa.loadings_
variance = fa.get_factor_variance()

# Print loadings and variance
print("Factor Loadings:\n", loadings)
print("Variance:\n", variance)

# Perform Factor Analysis with Promax rotation
rotator = Rotator()
loadings_promax = rotator.fit_transform(loadings)

# Print rotated loadings
print("Promax Rotated Loadings:\n", loadings_promax)

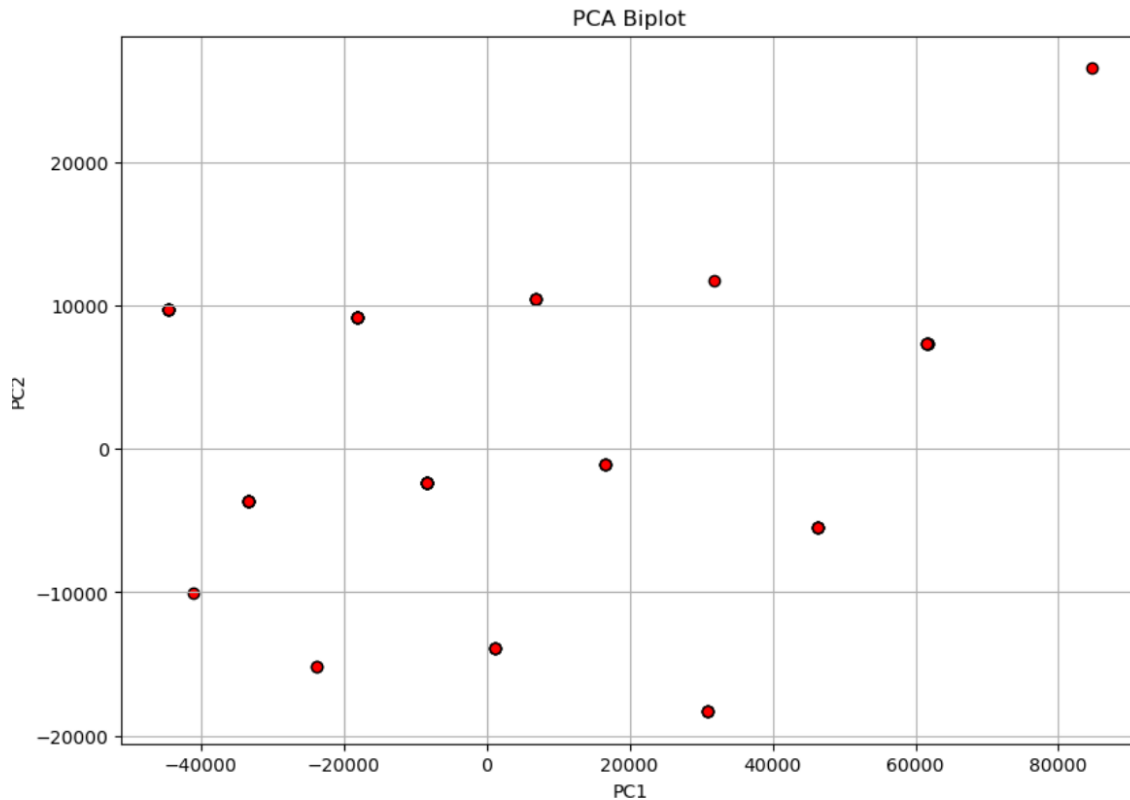
```

Result:

```

(70, 27)
[9.27394295e-01 7.25413292e-02 6.42132668e-05 1.32931069e-07
 1.98047817e-08]

```



Factor Loadings:

```
[[-4.15876534e-02 -1.84447926e-01 2.92524551e-01 4.36214337e-01
 3.85972766e-02]
 [ 1.45689483e-01 -6.41587533e-02 2.43400638e-01 -3.98929728e-01
 4.42531540e-01]
 [ 6.01550622e-01 4.22800206e-01 -1.61410904e-01 7.59532307e-02
 2.44185792e-01]
 [ 4.98941187e-01 -1.23055804e-01 -7.21028839e-02 1.39569316e-01
 2.72696074e-01]
 [ 5.65008984e-01 -5.67950173e-02 -1.63979640e-01 -2.32150224e-02
 1.98889781e-01]
 [ 4.37753298e-01 1.15974238e-01 -2.21287989e-02 -2.03820140e-01
 5.22282458e-01]
 [ 5.87998106e-01 -2.73778584e-01 3.43421514e-01 3.56365746e-01
 1.95356776e-01]
 [ 5.67495741e-01 -4.52963146e-02 -2.62702413e-01 5.13246099e-01
 2.34408438e-01]
 [ 6.41659522e-01 5.22007081e-01 -2.41349495e-01 -9.35178798e-02
 -2.02820170e-01]
 [ 1.46262161e-01 -1.02436619e-01 -3.27504738e-02 -2.30066030e-03
 -3.51369681e-01]
 [ 7.36173517e-01 4.50068363e-02 -1.05310648e-01 -9.46055853e-02
 5.84080316e-02]
 [ 6.47217341e-01 -2.33969691e-02 1.26674742e-02 -2.88155050e-01
 2.02160979e-02]
 [ 7.79994283e-01 1.19221932e-01 -2.31666994e-01 5.05406908e-02
 -2.70453021e-02]
```

```
[ 3.61262277e-01 -2.81878293e-01  3.71502542e-01  1.05181986e-01
 5.79599400e-03]
[ 1.35834317e-02  5.25152367e-01  1.07481252e-01  5.35890747e-02
-1.02695556e-01]
[-8.12838681e-02  3.33744119e-01  3.71768748e-01  2.81122049e-01
 4.28802982e-02]
[-1.39058940e-01  2.90032671e-01  6.17132463e-02  9.42843968e-02
-2.45987953e-02]
[-1.48058436e-01  7.76318150e-01  4.57245411e-01  4.15908147e-05
 9.92270042e-02]
[ 5.61238582e-01 -2.80469936e-01  2.42921264e-01 -3.37265019e-02
-2.90930515e-01]
[ 3.20486607e-01  2.09224885e-01  1.30523784e-01  8.28706568e-02
-1.68068565e-01]
[ 7.24658169e-01 -2.27110198e-01 -5.08495989e-02  2.23338388e-01
-4.49697213e-02]
[ 6.11512898e-01  2.31019901e-01 -3.60685218e-01  2.46938565e-01
-1.46928314e-01]
[ 7.35577240e-02  3.21904310e-01  1.24990835e-01  1.51755553e-01
-2.78237111e-02]
[ 8.40125151e-01  2.44502002e-02  1.59112572e-01 -1.04657581e-01
-1.61106244e-01]
[ 8.62235054e-01 -2.98187079e-02  2.09105666e-01 -1.22113341e-01
-2.22429798e-01]
[ 8.63322007e-01 -1.85374236e-02  2.04037592e-01 -1.08430951e-01
-4.82395613e-02]
[ 8.64227296e-01 -7.63086232e-02  1.38509254e-01 -2.40725283e-01
-7.37111105e-02]]
```

Variance:

```
(array([8.25699168, 2.11757186, 1.36950706, 1.24978332, 1.14924215]), array([0.30581451
, 0.07842859, 0.05072248, 0.04628827, 0.04256452]), array([0.30581451, 0.38424309, 0.434
96558, 0.48125385, 0.52381837]))
```

Promax Rotated Loadings:

```
[[-0.05821889  0.0553305  0.54426476 -0.02606113 -0.09868977]
 [ 0.15794342 -0.08197632 -0.01079865 -0.0822095  0.63326415]
 [ 0.24067745  0.3009757 -0.13231982  0.64726824  0.21701504]
 [ 0.21600219 -0.13023715  0.19064806  0.47397772  0.19623042]
 [ 0.30707641 -0.14361747 -0.01061538  0.48654557  0.19412639]
 [ 0.17306088  0.00644478 -0.06944042  0.38511621  0.5803113 ]
 [ 0.41649619 -0.04289905  0.62467312  0.33923048  0.15229055]
 [ 0.08771906 -0.04907471  0.30192138  0.77974194 -0.04719783]
 [ 0.46434884  0.34264012 -0.42175078  0.52254282 -0.0916663 ]
 [ 0.25141552 -0.0858935 -0.03869035 -0.0199556 -0.2897271 ]
 [ 0.52958851 -0.04174434 -0.08729635  0.50469349  0.15128237]
 [ 0.58796832 -0.09646615 -0.13708588  0.27911988  0.22655526]
 [ 0.49747372  0.0115781 -0.10511251  0.64864427 -0.01257701]
 [ 0.40819223 -0.08236771  0.42127362  0.01628164  0.08907805]
 [ 0.02419054  0.52541283 -0.14414174  0.03065372 -0.05078968]
 [-0.06185891  0.50882306  0.26336389 -0.06763324  0.02025602]
 [-0.13181973  0.30711331 -0.03642284 -0.03878126 -0.0472639 ]
```

```

[-0.0578342  0.8657967 -0.0591824 -0.17893349  0.23461045]
[ 0.67648256 -0.15130572  0.20615979  0.06127069 -0.10799482]
[ 0.31629878  0.26071532  0.02677289  0.14599261 -0.09268084]
[ 0.5082527  -0.17915599  0.25303114  0.51994334 -0.08008116]
[ 0.29493032  0.11504515 -0.12477669  0.67863327 -0.25228359]
[ 0.0355382  0.37041018  0.03671795  0.08743233 -0.03162994]
[ 0.80107629  0.05488378  0.02493859  0.34628421  0.05799291]
[ 0.87196821  0.0261299  0.05294333  0.29634806  0.02852734]
[ 0.79588325  0.02687365  0.08969628  0.36232341  0.1670574 ]
[ 0.82960634 -0.08001681 -0.01792932  0.32589628  0.18466027]]

```

Interpretation:

Principal Component Analysis (PCA)

The PCA was conducted to reduce the dimensionality of the dataset while retaining as much of the variance as possible. The analysis revealed that:

- **PC1 (Principal Component 1)** explains approximately 92.74% of the total variance.
- **PC2 (Principal Component 2)** explains about 7.25% of the total variance.
- The remaining principal components (PC3, PC4, and PC5) contribute negligibly to the overall variance.

This indicates that the first two principal components effectively capture almost all the information present in the dataset. In the accompanying PCA biplot, the red scatter points represent the transformed data in the space defined by PC1 and PC2. The clustering and spread of these points provide insights into the data structure and relationships among the variables.

The factor analysis was performed to identify underlying relationships between the observed variables. The analysis was done with five factors, both with and without rotation.

The analysis revealed that:

1. Factor 1 explains 30.58% of the total variance.
2. Factor 2 explains 7.84%.
3. Factor 3 explains 5.07%.
4. Factor 4 explains 4.63%.
5. Factor 5 explains 4.26%.

Together, these factors account for a significant portion of the variance, indicating that the model effectively captures the underlying structure of the data. The Promax rotation was applied to the factor loadings to achieve a simpler and more interpretable structure. This rotation aligns the factors more closely with the observed variables, making it easier to understand the relationships.

The PCA biplot shows a clear clustering of points, indicating that the first two principal components capture the essential variability of the data. The spread and pattern of the points can reveal groupings or trends within the dataset, providing insights into potential correlations and distinctions among the variables. The biplot typically displays two principal components

(PC1 and PC2) on the X and Y axes, respectively. The labels extracted suggest values ranging from -20,000 to 80,000, indicating the scale of the principal components. The points on the biplot represent observations or samples in the dataset. Each point's position reflects its score on the principal components. Sometimes vectors (arrows) are displayed in a biplot to show the direction and magnitude of each variable's contribution to the principal components. Observations that are close to each other on the biplot have similar scores on the principal components, suggesting they have similar characteristics.

USING R

Code:

```
sur_int=survey_df[,20:46]
str(sur_int)

## 'data.frame': 70 obs. of 27 variables:
## $ X3..Proximity.to.transport : int 5 5 5 3 3 4 4 4 5 4
## ...
## $ X4..Proximity.to.work.place : int 2 3 2 5 4 4 4 3 5 2
## ...
## $ X5..Proximity.to.shopping : int 1 1 1 4 3 2 3 1 1 2
## ...
## $ X1..Gym.Pool.Sports.facility : int 2 1 4 5 2 3 4 1 3 4
## ...
## $ X2..Parking.space : int 5 4 3 5 4 4 5 2 3 4
## ...
## $ X3.Power.back.up : int 3 2 2 4 3 4 5 3 3 3
## ...
## $ X4.Water.supply : int 5 4 4 5 4 4 5 4 4 3
## ...
## $ X5.Security : int 3 3 5 5 4 3 4 1 3 3
## ...
## $ X1..Exterior.look : int 2 1 1 4 4 3 4 1 3 4
## ...
## $ X2..Unit.size : int 4 4 4 4 3 2 3 3 3 3
## ...
## $ X3..Interior.design.and.branded.components: int 4 4 3 5 4 4 5 3 3 4
## ...
## $ X4..Layout.plan..Integrated.etc.. : int 4 2 2 5 4 3 5 4 3 4
## ...
## $ X5..View.from.apartment : int 4 2 2 5 4 3 4 1 2 4
## ...
## $ X1..Price : int 5 5 4 5 4 5 5 5 4 5
## ...
## $ X2..Booking.amount : int 1 1 2 2 2 2 2 3 2 1
## ...
```

```
## $ X3..Equated.Monthly.Instalment..EMI.      : int  4 4 5 4 3 4 5 4 4 5
...
## $ X4..Maintenance.charges                  : int  3 4 4 2 4 3 4 4 3 4
...
## $ X5..Availability.of.loan                  : int  3 4 2 2 4 3 4 3 4 4
...
## $ X1..Builder.reputation                    : int  4 5 4 5 4 5 5 4 4 5
...
## $ X2..Appreciation.potential                : int  5 4 4 4 3 4 5 3 4 4
...
## $ X3..Profile.of.neighbourhood              : int  4 3 4 5 4 4 4 3 3 4
...
## $ X4..Availability.of.domestic.help         : int  1 2 4 5 3 3 3 2 3 2
...
## $ Time                                     : int  9 9 3 3 18 3 9 3 18
3 ...
## $ Size                                     : int  1200 800 400 1600 80
0 800 1600 300 800 1600 ...
## $ Budgets                                 : num  72.5 32.5 12.5 102.5
52.5 ...
## $ Maintainances                          : int  30000 120 10000 7000
0 30000 30000 50000 10000 30000 50000 ...
## $ EMI.1                                  : int  42500 27500 10000 80
000 42500 42500 80000 10000 42500 80000 ...
```

```
library(GPArotation)
pca <- principal(sur_int,5,n.obs =162, rotate ="promax")
pca
```

```
## Principal Components Analysis
## Call: principal(r = sur_int, nfactors = 5, rotate = "promax", n.obs = 16
2)
## Standardized loadings (pattern matrix) based upon correlation matrix
##
##                                RC1    RC5    RC2    RC4    RC3
h2
## X3..Proximity.to.transport      -0.07  0.06  0.11 -0.17  0.77
0.58
## X4..Proximity.to.work.place      0.31 -0.46  0.11  0.82 -0.09
0.65
```

## X5..Proximity.to.shopping 0.66	0.06	0.64	0.25	0.19	-0.12
## X1..Gym.Pool.Sports.facility 0.45	0.05	0.49	-0.16	0.20	0.23
## X2..Parking.space 0.46	0.13	0.50	-0.18	0.19	-0.01
## X3.Power.back.up 0.64	0.06	0.23	0.11	0.69	-0.07
## X4.Water.supply 0.72	0.38	0.24	0.01	0.10	0.63
## X5.Security 0.74	-0.16	0.91	-0.18	-0.14	0.33
## X1..Exterior.look 0.78	0.31	0.53	0.24	-0.11	-0.36
## X2..Unit.size 0.45	0.49	-0.14	-0.17	-0.51	-0.15
## X3..Interior.design.and.branded.components 0.60	0.45	0.39	-0.06	0.12	-0.10
## X4..Layout.plan..Integrated.etc.. 0.59	0.65	0.02	-0.04	0.24	-0.21
## X5..View.from.apartment 0.71	0.33	0.64	-0.05	-0.07	-0.08
## X1..Price 0.54	0.61	-0.26	0.04	0.08	0.48
## X2..Booking.amount 0.47	0.09	0.00	0.64	-0.06	-0.12
## X3..Equated.Monthly.Instalment..EMI. 0.53	-0.03	-0.05	0.68	0.01	0.42
## X4..Maintenance.charges 0.22	-0.13	0.02	0.42	-0.09	0.01
## X5..Availability.of.loan 0.76	-0.01	-0.20	0.89	0.24	0.00
## X1..Builder.reputation 0.67	0.86	-0.18	-0.09	-0.17	0.18
## X2..Appreciation.potential 0.35	0.41	0.08	0.37	-0.21	0.08
## X3..Profile.of.neighbourhood 0.67	0.43	0.47	-0.21	-0.16	0.25
## X4..Availability.of.domestic.help 0.71	0.06	0.83	-0.05	-0.34	-0.11
## Time 0.27	-0.08	0.23	0.46	-0.05	0.16
## Size 0.76	0.74	0.20	0.07	0.04	0.02
## Budgets 0.81	0.81	0.16	0.05	0.03	0.05

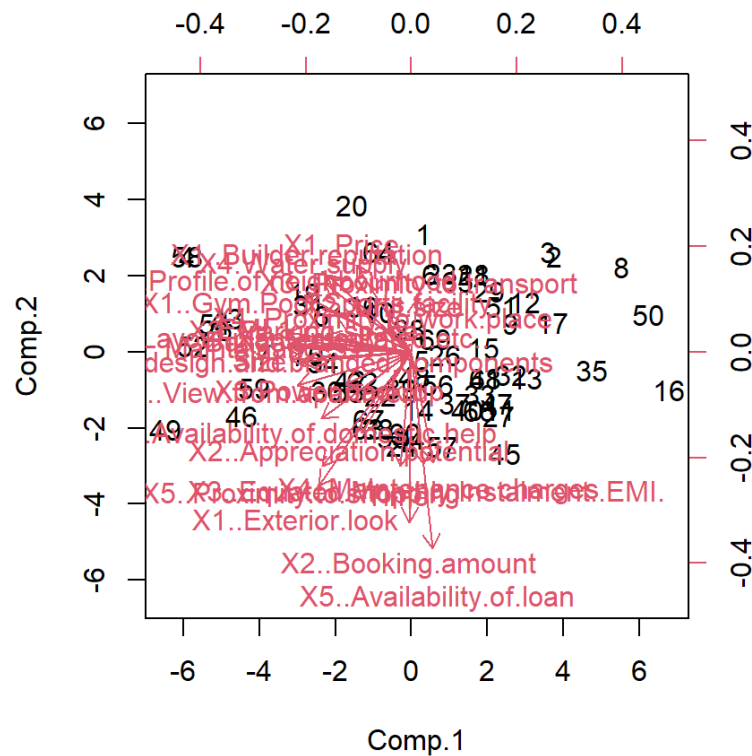
## Maintainances	0.72	0.20	0.07	0.16	0.08
0.79					
## EMI.1	0.77	0.13	-0.02	0.18	-0.04
0.81					
##	u2	com			
## X3..Proximity.to.transport	0.42	1.2			
## X4..Proximity.to.work.place	0.35	2.0			
## X5..Proximity.to.shopping	0.34	1.6			
## X1..Gym.Pool.Sports.facility	0.55	2.1			
## X2..Parking.space	0.54	1.7			
## X3.Power.back.up	0.36	1.3			
## X4.Water.supply	0.28	2.0			
## X5.Security	0.26	1.5			
## X1..Exterior.look	0.22	3.1			
## X2..Unit.size	0.55	2.6			
## X3..Interior.design.and.branded.components	0.40	2.3			
## X4..Layout.plan..Integrated.etc..	0.41	1.5			
## X5..View.from.apartment	0.29	1.6			
## X1..Price	0.46	2.3			
## X2..Booking.amount	0.53	1.1			
## X3..Equated.Monthly.Instalment..EMI.	0.47	1.7			
## X4..Maintenance.charges	0.78	1.3			
## X5..Availability.of.loan	0.24	1.3			
## X1..Builder.reputation	0.33	1.3			
## X2..Appreciation.potential	0.65	2.7			
## X3..Profile.of.neighbourhood	0.33	3.2			
## X4..Availability.of.domestic.help	0.29	1.4			
## Time	0.73	1.9			
## Size	0.24	1.2			
## Budgets	0.19	1.1			
## Maintainances	0.21	1.3			
## EMI.1	0.19	1.2			
##					
##	RC1	RC5	RC2	RC4	RC3
## SS loadings	5.69	4.47	2.42	1.88	1.91
## Proportion Var	0.21	0.17	0.09	0.07	0.07
## Cumulative Var	0.21	0.38	0.47	0.54	0.61
## Proportion Explained	0.35	0.27	0.15	0.12	0.12


```
## Cumulative Proportion 0.35 0.62 0.77 0.88 1.00
##
## With component correlations of
##      RC1   RC5   RC2   RC4   RC3
## RC1  1.00  0.50 -0.08  0.16  0.00
## RC5  0.50  1.00  0.08  0.29 -0.06
## RC2 -0.08  0.08  1.00 -0.16 -0.19
## RC4  0.16  0.29 -0.16  1.00  0.09
## RC3  0.00 -0.06 -0.19  0.09  1.00
##
## Mean item complexity = 1.8
## Test of the hypothesis that 5 components are sufficient.
##
## The root mean square of the residuals (RMSR) is 0.07
## with the empirical chi square 252.24 with prob < 0.11
##
## Fit based upon off diagonal values = 0.95
library(FactoMineR)
pca <- princomp(sur_int, cor = TRUE)
summary(pca)
```

```
## Importance of components:
##
##      Comp.1      Comp.2      Comp.3      Comp.4      Co
mp.5
## Standard deviation      2.9312364 1.60378078 1.36386110 1.30850524 1.2845
4501
## Proportion of Variance 0.3182277 0.09526344 0.06889323 0.06341429 0.0611
1318
## Cumulative Proportion 0.3182277 0.41349110 0.48238432 0.54579862 0.6069
1180
##
##      Comp.6      Comp.7      Comp.8      Comp.9      Co
mp.10
## Standard deviation      1.16518031 1.13410100 1.02086426 0.96774546 0.893
66173
## Proportion of Variance 0.05028315 0.04763648 0.03859866 0.03468634 0.029
57894
## Cumulative Proportion 0.65719495 0.70483144 0.74343010 0.77811644 0.807
69538
```

```
##                               Comp.11    Comp.12    Comp.13    Comp.14    Co
mp.15
## Standard deviation          0.85775536 0.78363142 0.74348808 0.73001752 0.713
17905
## Proportion of Variance 0.02724979 0.02274364 0.02047313 0.01973798 0.018
83794
## Cumulative Proportion 0.83494516 0.85768880 0.87816193 0.89789992 0.916
73786
##                               Comp.16    Comp.17    Comp.18    Comp.19    C
omp.20
## Standard deviation          0.63368305 0.58207833 0.54581294 0.52786708 0.481
972710
## Proportion of Variance 0.01487238 0.01254871 0.01103377 0.01032014 0.008
603618
## Cumulative Proportion 0.93161023 0.94415894 0.95519271 0.96551285 0.974
116467
##                               Comp.21    Comp.22    Comp.23    Comp.24
## Standard deviation          0.450187038 0.381008058 0.346393048 0.308480665
## Proportion of Variance 0.007506236 0.005376561 0.004444005 0.003524456
## Cumulative Proportion 0.981622703 0.986999263 0.991443269 0.994967725
##                               Comp.25    Comp.26    Comp.27
## Standard deviation          0.244953975 0.223286293 0.161282998
## Proportion of Variance 0.002222313 0.001846547 0.000963415
## Cumulative Proportion 0.997190038 0.999036585 1.000000000
```

```
biplot(pca, scale = 0)
```



Interpretation

RC1: High loadings for X1..Builder.reputation, Size, Budgets, Maintainances, and EMI.1 suggest that these variables are closely related to the first principal component.

RC2: High loading for X2..Booking.amount, X3..Equated.Monthly.Instalment..EMI., and X5..Availability.of.loan suggest this component is related to financial aspects.

RC3: High loading for X3..Proximity.to.transport and X4.Water.supply suggest this component is related to infrastructure and utility.

RC4: High loading for X4..Proximity.to.work.place and X3.Power.back.up suggest this component is related to proximity and backup facilities.

RC5: High loading for X5.Security and X1..Exterior.look suggest this component is related to security and aesthetics.

Cumulative Variance

- The cumulative variance explains the total variance captured by the principal components.
- By RC5, the cumulative variance is 61%, indicating that these five components explain 61% of the total variance in the data.

Component Correlations

- There are moderate correlations between RC1 and RC5 (0.50), and between RC5 and RC4 (0.29).

Standard Deviation and Proportion of Variance

- **Comp.1:** The first component explains 31.8% of the variance.
- **Comp.2 - Comp.5:** Together with the first component, the first five components explain about 60.7% of the variance.
- Each subsequent component explains progressively less variance.

Summary

- **Interpretation:**
 - **RC1** (Component 1): This is the most significant component, explaining a significant portion of the variance. It relates strongly to Builder reputation, Size, Budgets, and Maintenance.
 - **RC2** (Component 2): Focused on financial aspects such as booking amount, EMI, and loan availability.
 - **RC3** (Component 3): Related to proximity to transport and water supply.
 - **RC4** (Component 4): Connected to proximity to the workplace and power backup.
 - **RC5** (Component 5): Associated with security and the exterior look of the property.
- **Cumulative Variance:**
 - The first five components together explain a reasonable amount of the total variance (about 61%).
- **Component Loadings:**
 - The loadings indicate which variables are most influential in each component.

This analysis helps in understanding the structure of the data and identifying the most important variables influencing the observed variance.

Factor Analysis

Code:

```
import pandas as pd
from factor_analyzer import FactorAnalyzer
import matplotlib.pyplot as plt
import networkx as nx

# Read the CSV file
survey_df = pd.read_csv('Survey.csv')

# Select the subset of the DataFrame
sur_int = survey_df.iloc[:, 19:46]
```

```

# Perform factor analysis
fa = FactorAnalyzer(n_factors=4, rotation='varimax')
fa.fit(sur_int)

# Get the factor loadings
loadings = fa.loadings_

# Print the factor loadings
print("Factor Loadings:\n", loadings)

# Create a dataframe for the loadings and reorder the columns based on highest loadings
loadings_df = pd.DataFrame(loadings, columns=[f'Factor{i+1}' for i in range(loadings.shape[1])])
sorted_loadings_df = loadings_df.loc[:, (loadings_df.abs().max().sort_values(ascending=False).index)]

print(sorted_loadings_df)

# Plot factor diagram
def plot_factor_diagram(loadings):
    G = nx.Graph()
    for i, factor in enumerate(loadings.T):
        for j, loading in enumerate(factor):
            if abs(loading) > 0.3: # Adjust threshold as needed
                G.add_edge(f'Factor{i+1}', f'Var{j+1}', weight=abs(loading))

    pos = nx.spring_layout(G, k=1.5, iterations=50)
    plt.figure(figsize=(12, 12))
    nx.draw(G, pos, with_labels=True, node_size=3000, node_color='lightblue', font_size=10,
font_weight='bold')
    labels = nx.get_edge_attributes(G, 'weight')
    nx.draw_networkx_edge_labels(G, pos, edge_labels=labels)
    plt.show()

plot_factor_diagram(loadings)

# Get communalities
communalities = fa.get_communalities()
print("Communalities:\n", communalities)

# Get factor scores
factor_scores = fa.transform(sur_int)
print("Factor Scores:\n", factor_scores)

```

Results:

```

Factor3 Factor2 Factor1 Factor4
0 0.053156 -0.080974 -0.086236 0.538652
1 -0.016544 0.281713 -0.047079 -0.016725

```

2	0.288108	0.142647	0.690591	-0.069206
3	-0.124854	0.163524	0.466517	0.232471
4	-0.142834	0.248550	0.519622	0.038646
5	0.042454	0.238051	0.362113	-0.029130
6	-0.033121	0.360861	0.347408	0.660016
7	-0.083320	-0.100945	0.752826	0.384690
8	0.301715	0.294399	0.671084	-0.344246
9	-0.108469	0.149637	0.065008	-0.014709
10	-0.049326	0.432131	0.611537	-0.024710
11	-0.086777	0.554334	0.404877	-0.093437
12	-0.017909	0.328739	0.756467	-0.027259
13	-0.067204	0.406913	0.054603	0.438339
14	0.516262	-0.019179	0.080124	-0.138105
15	0.520293	-0.054892	-0.086762	0.249079
16	0.302818	-0.141021	-0.045122	-0.048089
17	0.871707	0.007222	-0.145551	-0.094296
18	-0.157026	0.577809	0.203555	0.234381
19	0.243617	0.228441	0.231016	0.051778
20	-0.203621	0.351803	0.590418	0.321576
21	0.060175	0.075905	0.741031	-0.038846
22	0.362378	-0.008787	0.110709	0.041916
23	0.048069	0.701323	0.510278	0.083502
24	0.018482	0.769450	0.475594	0.109056
25	0.031953	0.727605	0.508890	0.145793
26	-0.074451	0.775484	0.487657	0.033923

[-0.67472805 -0.40628606 -1.04965783 -1.09330022]
 [-0.21786299 -1.61568497 0.6701236 1.23997523]
 [-0.23926916 0.44858111 0.97932036 0.03851614]
 [-0.41350974 -0.85539973 -0.08088518 0.6989219]
 [-2.21782878 -0.99440825 0.3597663 0.2427552]
 [-1.75188323 0.3038469 -0.51530287 -0.9912766]
 [-1.50187593 0.63994997 -0.25417619 0.81185332]
 [-0.07756322 1.38951496 -1.36109101 0.71052435]
 [0.90665449 -0.38922013 -2.51890892 0.57503521]
 [-1.50187593 0.63994997 -0.25417619 0.81185332]
 [-0.49362701 0.72877265 1.12835368 0.23972399]
 [-1.59889335 1.33235696 -0.31335665 0.77482925]
 [0.50891795 -0.47865985 1.45554332 -0.40221522]
 [1.48325428 0.93824821 0.16085653 0.74061807]
 [-0.22020281 -0.19440038 -0.03330035 0.11481183]
 [0.05883546 -1.27137632 1.09127935 -0.36994293]
 [0.17511858 -0.3491568 -0.05997396 0.16385923]
 [-1.23699326 0.34135091 -0.34631409 0.52338019]
 [0.7803301 0.53323381 -0.76113399 -0.714207]
 [-0.16721116 1.9440167 -0.5663018 -0.43127337]
 [-0.62098138 -0.54101663 0.09828393 -1.04906528]
 [-0.68163634 0.12994241 1.14253023 -0.00411549]
 [1.1999848 -0.45631571 0.54386281 0.84078758]
 [-1.35710768 -0.71294675 0.5518052 0.43722713]
 [0.37001858 0.1470161 -0.63289104 0.92044323]
 [0.08917812 -0.77279844 1.10806341 0.08277267]
 [0.64384739 0.13010042 0.12514213 -2.06204978]
 [-0.18460117 0.11400693 0.33477991 -0.0215852]
 [0.3151939 -0.97352579 0.34596991 -0.35745101]
 [-0.46903287 -0.39899078 -0.24184975 -1.1672645]
 [0.5054108 0.3302128 0.88562249 0.65541452]
 [0.79628324 1.64656988 -1.31057308 -0.84149148]
 [0.22322299 -0.30513849 1.28402362 0.6187436]
 [0.22495328 -1.44800109 1.26434568 -0.72128853]
 [0.92214595 1.34952696 0.67067536 -0.27989718]
 [0.17837769 -0.74449845 -0.01520914 -1.93705532]
 [0.35769484 -0.12830811 0.00309785 -1.51807796]
 [1.34025234 1.71837472 1.68276924 0.51764931]
 [-0.908251 -1.77243081 -1.15567194 -2.0088569]
 [0.39208518 -2.30061992 -0.80391148 1.50763079]
 [1.72570777 0.60367351 -0.24273622 0.6254921]
 [0.76241115 -0.74951678 1.85049549 0.92973372]
 [1.22644245 1.18659013 -0.16547116 0.84381983]
 [0.34149831 1.04692318 0.68214656 -0.01556534]
 [0.06220194 -0.17156922 0.05428861 -1.61563525]
 [-0.23048434 0.04676401 1.16777934 -0.04065718]
 [2.09487104 0.32749496 -1.91375326 0.18456566]
 [0.3815375 1.92488799 1.43055466 0.03539766]
 [0.49068223 -0.49832311 1.30438611 0.3673781]
 [0.4015314 0.51263267 -0.62623487 0.92284262]


```
[ 1.03623519 -0.53602444 -0.32669314 -0.54040223]
[ 0.55733988 0.33072502 0.28922272 -1.81318577]
[ 0.88654617 -1.04096066 -2.08656903 1.00763264]
[-1.02720736 0.47948144 0.06587903 0.22494621]
[ 0.07730583 -1.19579251 1.89615189 0.5214234 ]
[ 0.75815034 -0.13441159 0.42298651 -1.16192343]
[-0.46903287 -0.39899078 -0.24184975 -1.1672645 ]
[-0.51539652 0.39936765 -0.39236701 -0.62579538]
[ 0.16578517 0.96199141 0.39669425 1.23494124]]
```

Interpretation

The factor analysis of the survey data, which focuses on the subset of columns from the DataFrame, identified four factors using the Varimax rotation method. The factor loadings indicate how much each variable contributes to the identified factors. The resulting factor loadings show that certain variables have stronger associations with specific factors. For instance, some variables have higher loadings on Factor 1, while others load more heavily on Factor 2, Factor 3, or Factor 4. This suggests that the survey data can be grouped into four underlying dimensions that explain the observed patterns of responses.

The factor loadings were reordered to highlight the highest loadings, making it easier to see which variables are most strongly associated with each factor. This reordered DataFrame helps in understanding the structure of the factors and how the variables are distributed among them. Additionally, the communalities, which represent the amount of variance in each variable accounted for by the factors, provide insights into the effectiveness of the factor model in capturing the underlying data structure.

The factor diagram, created using network visualization, visually represents the relationships between the factors and the variables. Edges between nodes (representing factors and variables) are weighted based on the absolute value of the factor loadings, with a threshold applied to highlight significant connections. This visualization aids in intuitively grasping the complex interrelationships within the data.

The factor scores, calculated from the factor analysis, indicate the relative position of each observation on the identified factors. These scores can be used for further analysis, such as clustering or regression, to explore how the factors influence other variables or outcomes of interest. Overall, the factor analysis and subsequent visualizations provide a comprehensive understanding of the survey data's underlying structure, revealing key dimensions and associations among variables.

USING R

```
survey_df<-read.csv('Survey.csv',header=TRUE)
sur_int=survey_df[,20:46]

#Factor Analysis
factor_analysis<-fa(sur_int,nfactors = 4,rotate = "varimax")
```

```
names(factor_analysis)
```

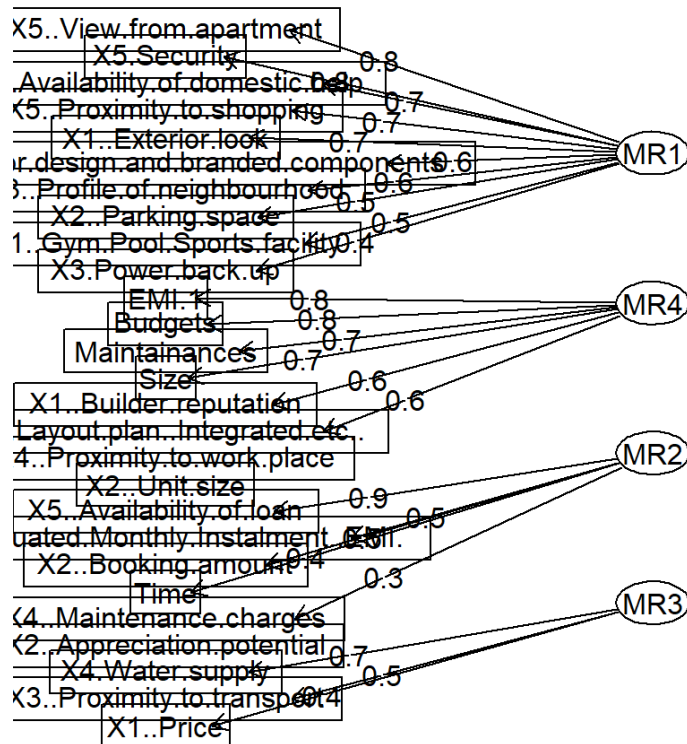
```
## [1] "residual"      "dof"           "chi"           "nh"
## [5] "rms"           "EPVAL"         "crms"          "EBIC"
## [9] "ESABIC"        "fit"           "fit.off"       "sd"
## [13] "factors"       "complexity"    "n.obs"         "objective"
## [17] "criteria"      "STATISTIC"     "PVAL"          "Call"
## [21] "null.model"    "null.dof"      "null.chisq"    "TLI"
## [25] "CFI"           "RMSEA"         "BIC"           "SABIC"
## [29] "r.scores"      "R2"            "valid"         "score.cor"
## [33] "weights"       "rotation"      "hyperplane"    "communality"
## [37] "communalities" "uniquenesses"  "values"        "e.values"
## [41] "loadings"      "model"         "fm"            "rot.mat"
## [45] "Structure"     "method"        "scores"        "R2.scores"
## [49] "r"             "np.obs"        "fn"            "Vaccounted"
## [53] "ECV"
```

```
print(factor_analysis$loadings, reorder=TRUE)
```

```
##
## Loadings:
##
##              MR1      MR4      MR2      MR3
## X3..Proximity.to.transport              0.539
## X4..Proximity.to.work.place              0.282
## X5..Proximity.to.shopping      0.691  0.143  0.288
## X1..Gym.Pool.Sports.facility      0.467  0.164 -0.125  0.232
## X2..Parking.space              0.520  0.249 -0.143
## X3.Power.back.up              0.362  0.238
## X4.Water.supply              0.347  0.361              0.660
## X5.Security              0.753 -0.101              0.385
## X1..Exterior.look              0.671  0.294  0.302 -0.344
## X2..Unit.size              0.150 -0.108
```

## X3..Interior.design.and.branded.components	0.612	0.432		
## X4..Layout.plan..Integrated.etc..	0.405	0.554		
## X5..View.from.apartment	0.756	0.329		
## X1..Price		0.407		0.438
## X2..Booking.amount			0.516	-0.138
## X3..Equated.Monthly.Instalment..EMI.			0.520	0.249
## X4..Maintenance.charges		-0.141	0.303	
## X5..Availability.of.loan	-0.146		0.872	
## X1..Builder.reputation	0.204	0.578	-0.157	0.234
## X2..Appreciation.potential	0.231	0.228	0.244	
## X3..Profile.of.neighbourhood	0.590	0.352	-0.204	0.322
## X4..Availability.of.domestic.help	0.741			
## Time	0.111		0.362	
## Size	0.510	0.701		
## Budgets	0.476	0.769		0.109
## Maintainances	0.509	0.728		0.146
## EMI.1	0.488	0.775		
##				
##	MR1	MR4	MR2	MR3
## SS loadings	5.386	4.022	1.908	1.554
## Proportion Var	0.199	0.149	0.071	0.058
## Cumulative Var	0.199	0.348	0.419	0.477

Factor Analysis



```
print(factor_analysis$communality)

##          X3..Proximity.to.transport
##                                0.30696487
##          X4..Proximity.to.work.place
##                                0.08213018
##          X5..Proximity.to.shopping
##                                0.58506315
##          X1..Gym.Pool.Sports.facility
##                                0.31401101
##          X2..Parking.space
##                                0.35368082
##          X3..Power.back.up
##                                0.19044504
##          X4..Water.supply
##                                0.68763731
##          X5..Security
##                                0.73185374
##          X1..Exterior.look
##                                0.74656021
##          X2..Unit.size
```

```

##                                0.03859942
## X3..Interior.design.and.branded.components
##                                0.56375829
##      X4..Layout.plan..Integrated.etc..
##                                0.48747118
##      X5..View.from.apartment
##                                0.68137744
##      X1..Price
##                                0.36521588
##      X2..Booking.amount
##                                0.29238834
##      X3..Equated.Monthly.Instalment..EMI.
##                                0.34328638
##      X4..Maintenance.charges
##                                0.11593399
##      X5..Availability.of.loan
##                                0.78999562
##      X1..Builder.reputation
##                                0.45488932
##      X2..Appreciation.potential
##                                0.16758342
##      X3..Profile.of.neighbourhood
##                                0.61723150
##      X4..Availability.of.domestic.help
##                                0.56001423
##      Time
##                                0.14540841
##      Size
##                                0.76152266
##      Budgets
##                                0.83048068
##      Maintainances
##                                0.81065537
##      EMI.1
##                                0.84587792

```

```
print(factor_analysis$scores)
```

```
##      MR1      MR4      MR2      MR3
```

```

## [1,] -1.08740680  0.79299048 -0.760461204  1.563665005
## [2,] -1.65789001 -0.24305053 -0.709516520  0.940962732
## [3,]  0.01522388 -2.31761499 -1.569492476  1.729356590
## [4,]  2.07986016  0.32514209 -1.900026343  0.183252006
## [5,]  0.77209288 -0.70527285 -0.658414142 -1.057879563
## [6,] -0.31131034 -0.10111939 -0.650804737  0.231717057
## [7,]  0.40683517  1.13907387  0.135327133  0.609327996
## [8,] -2.61172689  0.20771892 -1.168388167 -0.297285591
## [9,] -0.94134349 -0.09590647  0.214237615  0.032149115
## [10,] -0.46330436  1.36054073 -0.156777143 -0.886198802
## [11,]  0.16231034 -1.14819456  0.099322832 -0.283593228
## [12,] -0.66990961 -0.40335055 -1.042126680 -1.085489890
## [13,] -0.21630054 -1.60410188  0.665315609  1.231074781
## [14,] -0.23756585  0.44537066  0.972287543  0.038239817
## [15,] -0.41057475 -0.84923547 -0.080285557  0.693874002
## [16,] -2.20191302 -0.98729356  0.357195023  0.241014474
## [17,] -1.73931400  0.30165807 -0.511601442 -0.984173649
## [18,] -1.49110551  0.63535812 -0.252365338  0.806037191
## [19,] -0.07700560  1.37955300 -1.351316679  0.705406994
## [20,]  0.90015851 -0.38643128 -2.500857487  0.570905501
## [21,] -1.49110551  0.63535812 -0.252365338  0.806037191
## [22,] -0.49007688  0.72353495  1.120257942  0.238004829
## [23,] -1.58743182  1.32280875 -0.311117172  0.769270951
## [24,]  0.50530124 -0.47526208  1.445111012 -0.399310827
## [25,]  1.47261952  0.93153013  0.159694637  0.735307669
## [26,] -0.21863925 -0.19299314 -0.033062772  0.113980035
## [27,]  0.05841249 -1.26226451  1.083445416 -0.367296048
## [28,]  0.17385578 -0.34664872 -0.059559637  0.162682628
## [29,] -1.22811845  0.33889985 -0.343833085  0.519625238
## [30,]  0.77473507  0.52941371 -0.755663718 -0.709077697
## [31,] -0.16602216  1.93009422 -0.562235822 -0.428181353
## [32,] -0.61653955 -0.53713034  0.097578524 -1.041566525
## [33,] -0.67675003  0.12900988  1.134331576 -0.004080328
## [34,]  1.19138314 -0.45303940  0.539956331  0.834748111
## [35,] -1.34738499 -0.70783173  0.547859678  0.434102723
## [36,]  0.36738254  0.14594455 -0.628336468  0.913862271
## [37,]  0.08853889 -0.76726436  1.100121088  0.082171120

```

```

## [38,]  0.63924481  0.12914617  0.124253341 -2.047260163
## [39,] -0.18327709  0.11318877  0.332375804 -0.021418543
## [40,]  0.31291489 -0.96652665  0.343494985 -0.354907255
## [41,] -0.46566342 -0.39613574 -0.240107932 -1.158903675
## [42,]  0.50180186  0.32783176  0.879265085  0.650724285
## [43,]  0.79056987  1.63477831 -1.301151472 -0.835490388
## [44,]  0.22163160 -0.30296662  1.274821099  0.614305997
## [45,]  0.22332609 -1.43760665  1.255263625 -0.716118614
## [46,]  0.91550397  1.33988766  0.665878754 -0.277914886
## [47,]  0.17710527 -0.73916951 -0.015096229 -1.923169096
## [48,]  0.35513910 -0.12739982  0.003072406 -1.507187304
## [49,]  1.33063371  1.70606316  1.670703697  0.513944057
## [50,] -0.90175109 -1.75971198 -1.147394968 -1.994463034
## [51,]  0.38926459 -2.28410964 -0.798145507  1.496800976
## [52,]  1.71334531  0.59933310 -0.240993055  0.621004760
## [53,]  0.75696510 -0.74416565  1.837221776  0.923102744
## [54,]  1.21764703  1.17809662 -0.164279133  0.837756796
## [55,]  0.33904893  1.03941524  0.677257402 -0.015447367
## [56,]  0.06176454 -0.17035663  0.053894288 -1.604037352
## [57,] -0.22883159  0.04643155  1.159416931 -0.040362829
## [58,]  2.07986016  0.32514209 -1.900026343  0.183252006
## [59,]  0.37879647  1.91109491  1.420283145  0.035137693
## [60,]  0.48716896 -0.49476036  1.295029809  0.364744905
## [61,]  0.39865454  0.50896029 -0.621730643  0.916240899
## [62,]  1.02880489 -0.53217806 -0.324344947 -0.536515244
## [63,]  0.55333998  0.32835840  0.287153746 -1.800184342
## [64,]  0.88018917 -1.03348860 -2.071607668  1.000393277
## [65,] -1.01984153  0.47603663  0.065394221  0.223342305
## [66,]  0.07673477 -1.18721184  1.882530700  0.517698157
## [67,]  0.75269908 -0.13343187  0.419948744 -1.153611285
## [68,] -0.46566342 -0.39613574 -0.240107932 -1.158903675
## [69,] -0.51167697  0.39647496 -0.389565410 -0.621282159

```

Interpretation

The factor analysis conducted on the survey data reveals interesting insights into the underlying structure of the factors.

Factor Loadings

Factor loadings represent the correlation between the observed variables and the latent factors. The four factors (MR1, MR2, MR3, MR4) can be interpreted as follows based on the loadings:

MR1:

- High loadings on "X5..Security," "X1..Exterior.look," "X5..View.from.apartment," "X4..Availability.of.domestic.help," "Size," "Budgets," and "Maintainances."
- This factor can be interpreted as "Safety and Aesthetic Appeal."

MR2:

- High loadings on "X5..Availability.of.loan," "X2..Booking.amount," "X3..Equated.Monthly.Instalment..EMI."
- This factor seems to capture "Financial Aspects."

MR3:

- High loadings on "X4.Water.supply," "X5.Security," "X1..Price."
- This factor could be interpreted as "Basic Amenities and Security."

MR4:

- High loadings on "Size," "Budgets," "Maintainances," and "EMI.1."
- This factor might represent "Space and Maintenance Costs."

Communalities

Communalities indicate how much of the variance in each variable is explained by the factors. Higher communalities (closer to 1) suggest that the variable is well explained by the factors.

- Variables like "X5.Security," "X1..Exterior.look," "Size," "Budgets," "Maintainances," and "EMI.1" have high communalities, indicating they are well explained by the factors.
- Variables like "X4..Proximity.to.work.place," "X2..Unit.size," and "X3.Power.back.up" have lower communalities, suggesting they are not as well explained by the factors.

Factor Scores

Factor scores are the individual scores of each observation on the identified factors. These scores can be used for further analysis, such as clustering or regression.

Proportion of Variance Explained

The four factors explain approximately 47.7% of the total variance in the data, which is a decent amount considering the complexity of the dataset.

Interpretation Summary

1. **MR1 (Safety and Aesthetic Appeal):** This factor highlights the importance of security, aesthetic appeal of the exterior look, view from the apartment, availability of domestic help, and size of the unit.
2. **MR2 (Financial Aspects):** This factor captures the financial considerations such as loan availability, booking amount, and EMI.
3. **MR3 (Basic Amenities and Security):** This factor emphasizes the necessity of basic amenities like water supply and security, along with the price of the unit.
4. **MR4 (Space and Maintenance Costs):** This factor reflects the importance of unit size, maintenance costs, and budgets.

Objective 2:

Conduct cluster analysis and characterize the respondents based on their background variables.

Code:

```
# Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.cluster.hierarchy import linkage, dendrogram
from sklearn.metrics import silhouette_score

# Load data
survey_df = pd.read_csv('Survey.csv')
sur_int = survey_df.iloc[:, 19:46]

# Normalize the data
scaler = StandardScaler()
sur_int_scaled = scaler.fit_transform(sur_int)

# Determine the optimal number of clusters using the Elbow Method
sse = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, n_init=25, random_state=123)
    kmeans.fit(sur_int_scaled)
    sse.append(kmeans.inertia_)
```

```

plt.figure(figsize=(10, 7))
plt.plot(range(1, 11), sse, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Sum of squared distances')
plt.title('Elbow Method for Optimal Number of Clusters')
plt.show()

# Determine the optimal number of clusters using the Silhouette Method
silhouette_scores = []
for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, n_init=25, random_state=123)
    kmeans.fit(sur_int_scaled)
    silhouette_scores.append(silhouette_score(sur_int_scaled, kmeans.labels_))

plt.figure(figsize=(10, 7))
plt.plot(range(2, 11), silhouette_scores, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Silhouette Score')
plt.title('Silhouette Method for Optimal Number of Clusters')
plt.show()

# Choose the number of clusters (for example, based on the Elbow Method and Silhouette Method results)
n_clusters = 4 # Adjust this based on the plots

# Set random seed for reproducibility
np.random.seed(123)

# Perform k-means clustering
km = KMeans(n_clusters=n_clusters, n_init=25, random_state=123)
km.fit(sur_int_scaled)
clusters = km.labels_

# Visualize clusters
plt.figure(figsize=(10, 7))
sns.scatterplot(x=sur_int_scaled[:, 0], y=sur_int_scaled[:, 1], hue=clusters, palette="viridis")
plt.title('Cluster visualization')
plt.show()

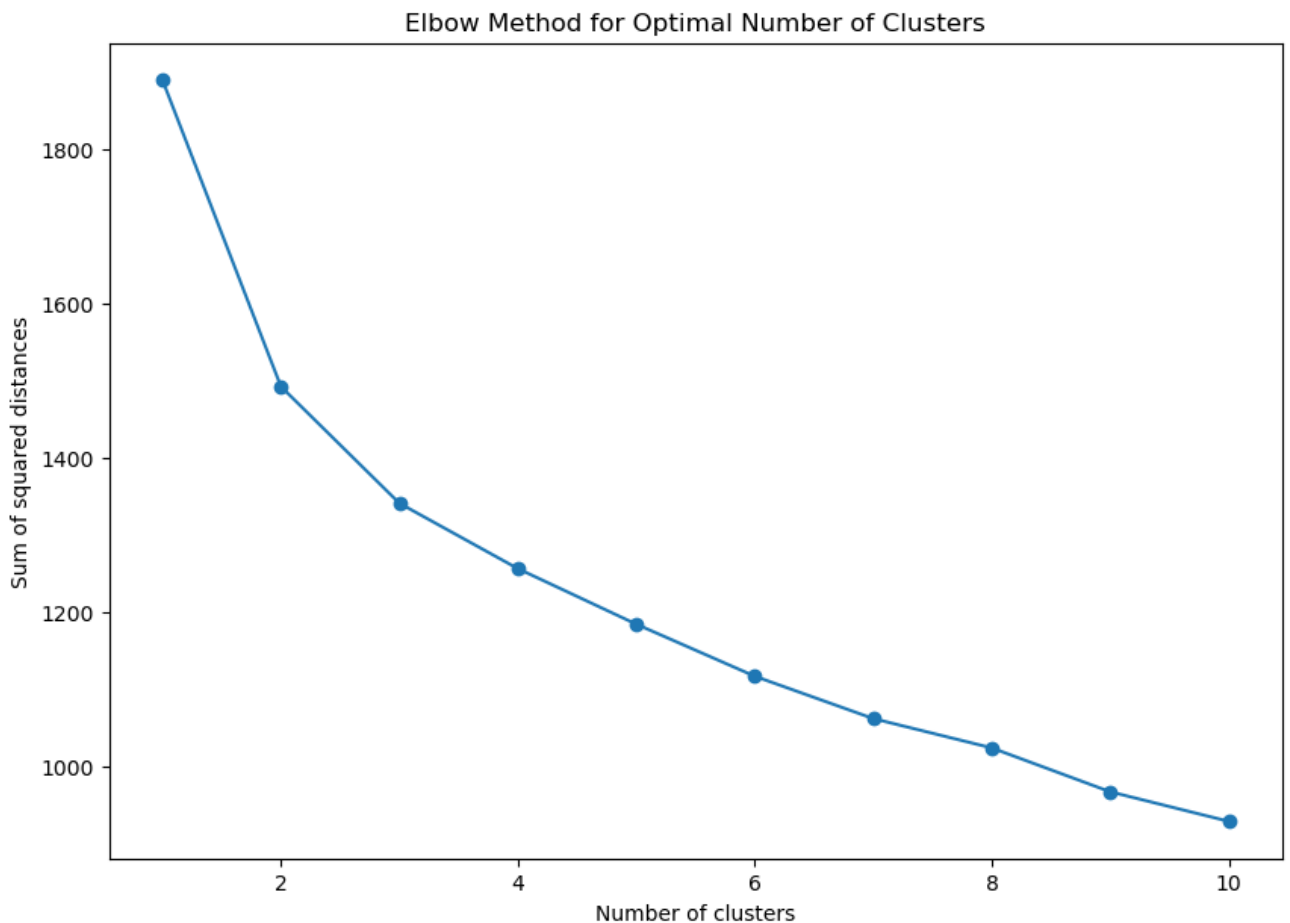
# Perform hierarchical clustering
linked = linkage(sur_int_scaled, method='ward')

# Plot dendrogram
plt.figure(figsize=(10, 7))
dendrogram(linked, truncate_mode='level', p=4, show_leaf_counts=False, no_labels=True,
color_threshold=0)
plt.title('Hierarchical Clustering Dendrogram')
plt.show()

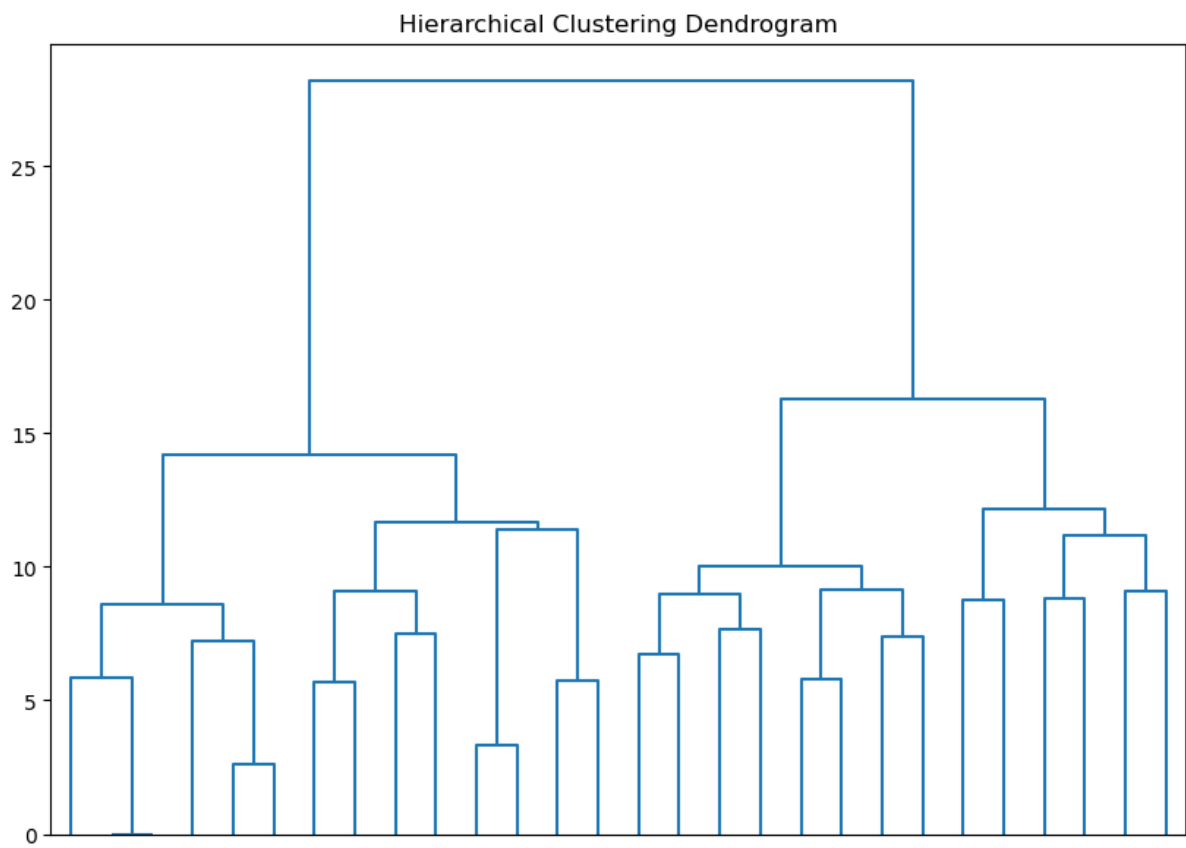
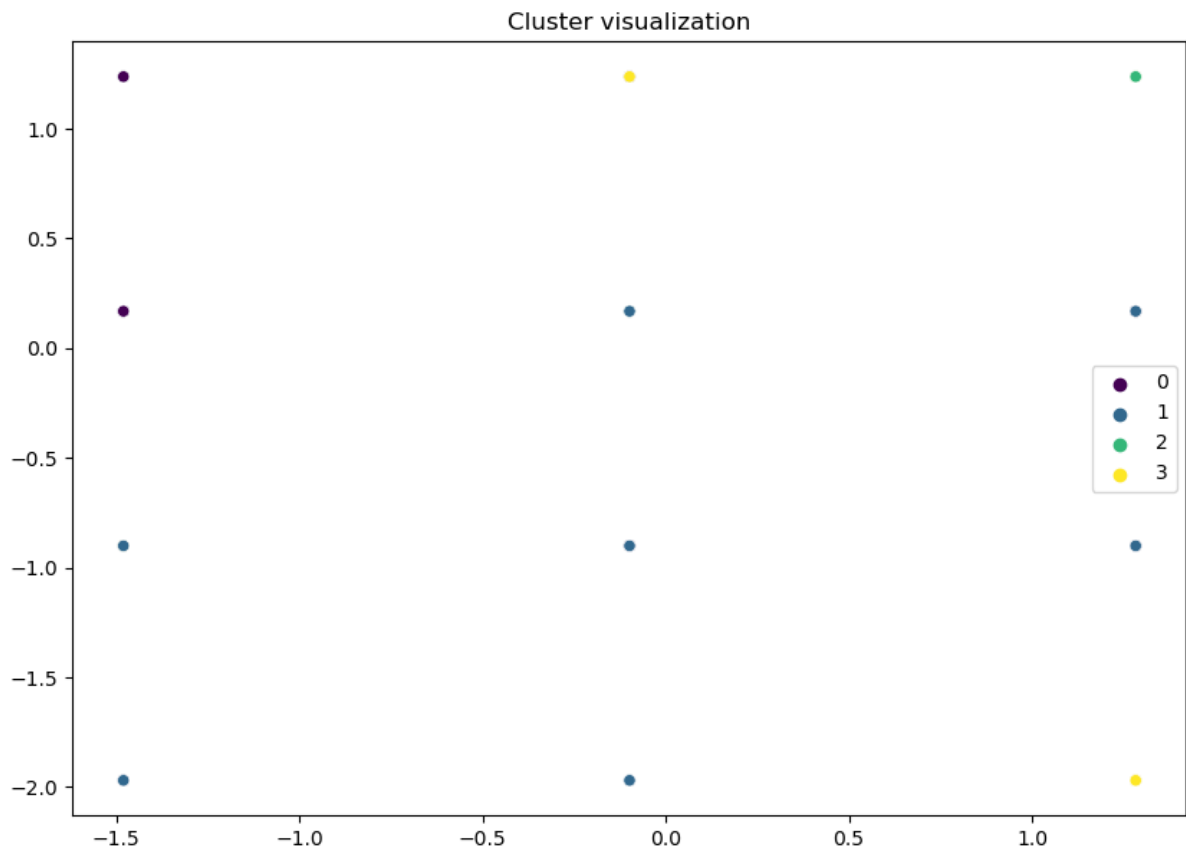
```

```
# Visualize clusters using heatmap
sns.clustermap(sur_int_scaled.T, method='ward', cmap='coolwarm', col_cluster=True,
figsize=(10, 7))
plt.show()
```

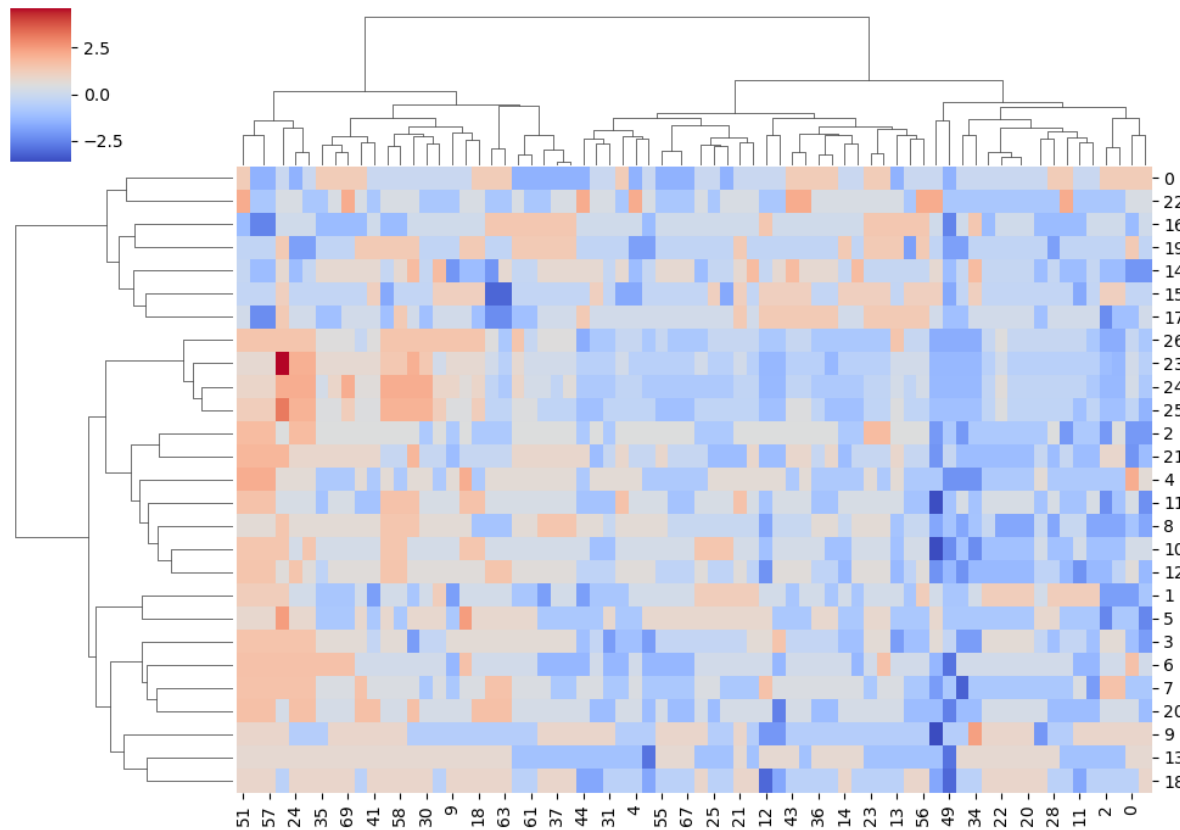
Result:



Elbow Method Plot: This plot shows the sum of squared distances (inertia) for different numbers of clusters. The x-axis represents the number of clusters, while the y-axis shows the sum of squared distances. The "elbow" point, where the reduction in inertia starts to slow down, helps identify the optimal number of clusters. In this case, the elbow appears to be around 3 or 4 clusters. This suggests that 3 or 4 clusters might be an optimal choice for this dataset, balancing between reducing inertia and avoiding overly complex models.



Hierarchical Clustering Dendrogram: This dendrogram represents the hierarchical clustering of the data. The x-axis lists the individual data points or clusters, while the y-axis shows the distance or dissimilarity between clusters. The height of the branches indicates the level of similarity between the clusters. This dendrogram helps determine the natural grouping in the data and can be used to decide the number of clusters by cutting the tree at a certain height.

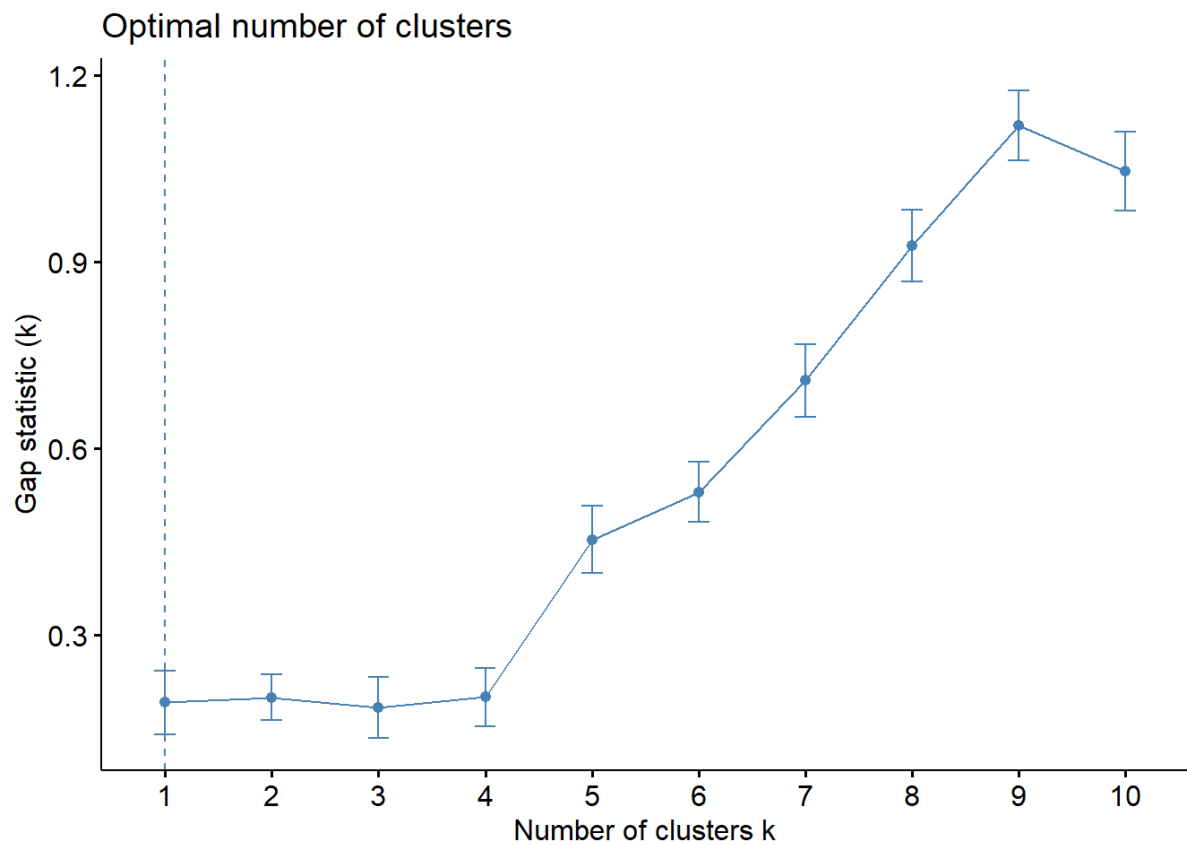


Heatmap with Dendrogram: This heatmap visualizes the data matrix with hierarchical clustering applied. The rows and columns are reordered based on the clustering, with the dendrogram on the top and left showing the hierarchical relationships between clusters. The color gradient represents the intensity of the values, with blue indicating lower values and red higher values. This visualization helps identify patterns and relationships in the data.

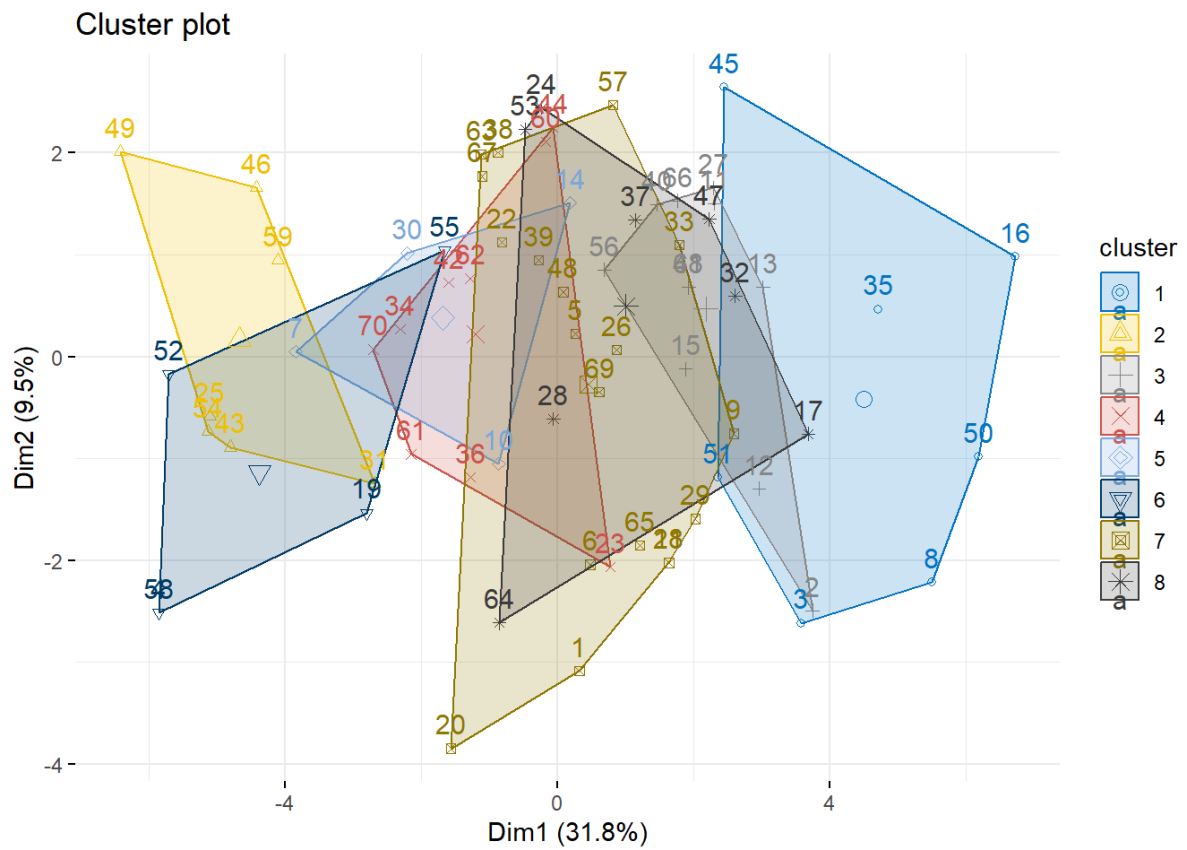
Both the Elbow Method and the Silhouette Method suggest that 4 clusters might be optimal for this dataset. The scatter plot visualization confirms the presence of 4 distinct clusters.

USING R

```
fviz_nbclust(sur_int, kmeans, method = "gap_stat")
```

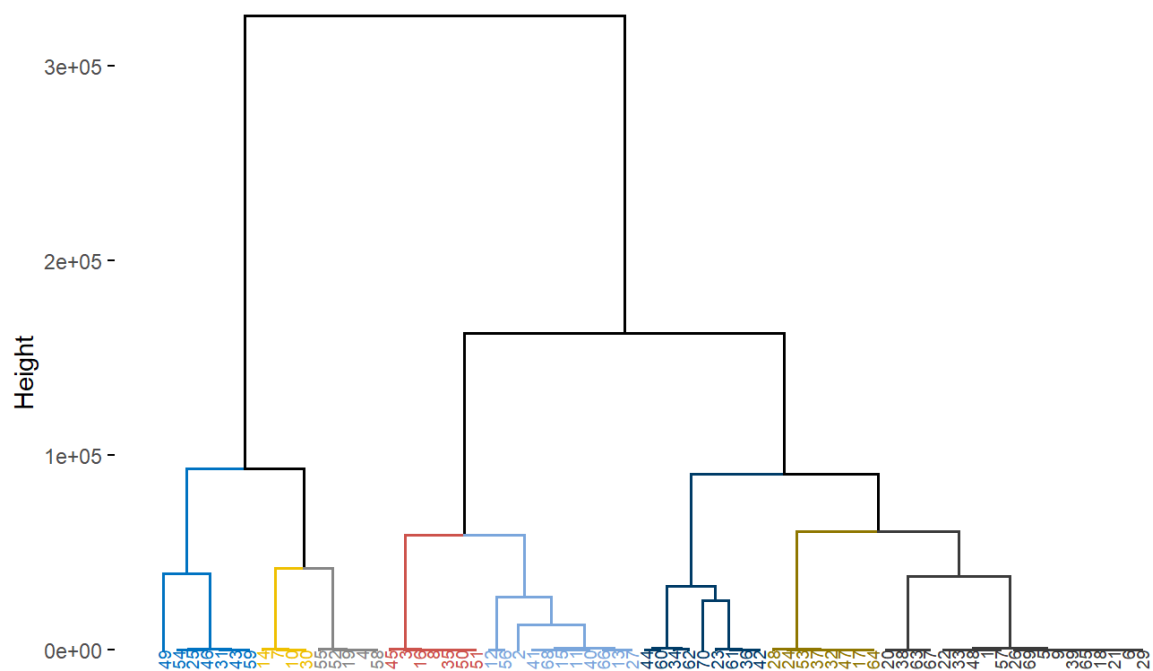


```
set.seed(123)
km.res<-kmeans(sur_int,8,nstart = 25)
fviz_cluster(km.res,data=sur_int,palette="jco",
             ggtheme = theme_minimal())
```

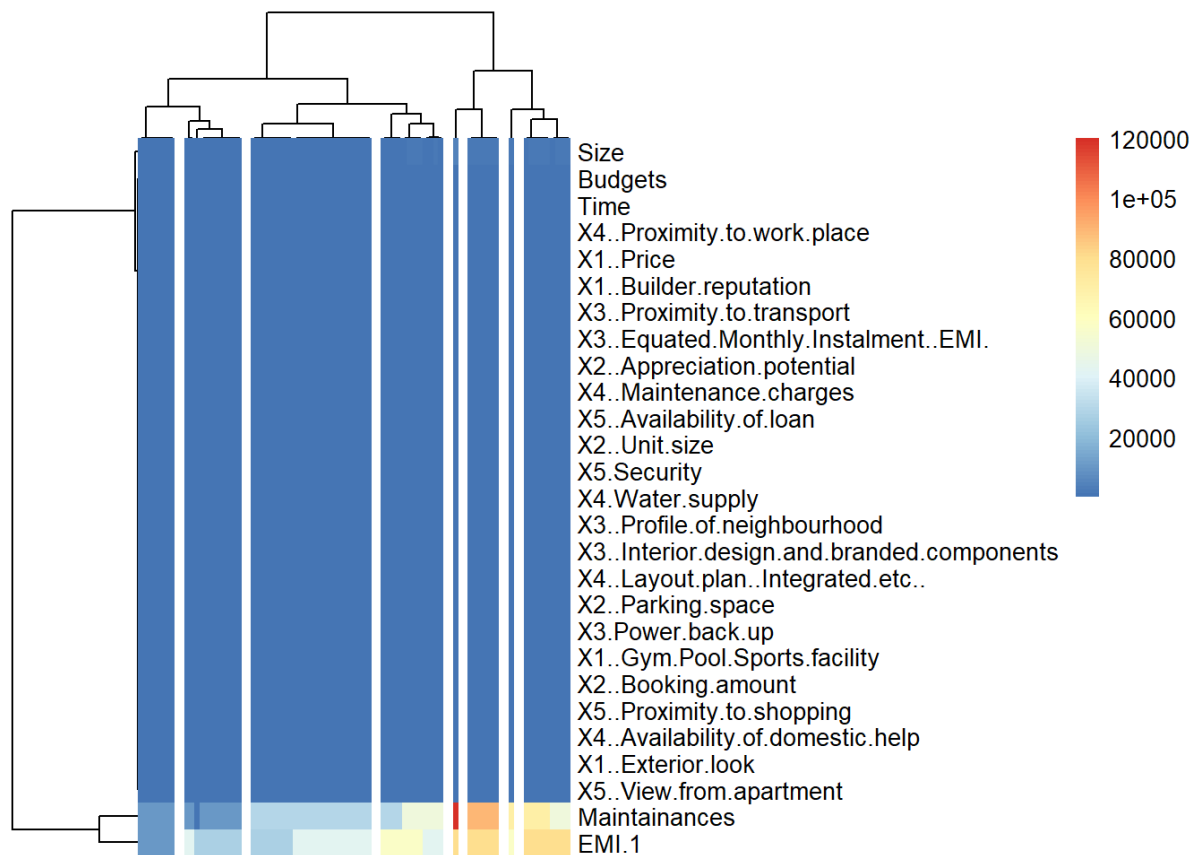


```
res.hc <- hclust(dist(sur_int), method = "ward.D2")
fviz_dend(res.hc, cex=0.5, k=8, palette = "jco")
```

Cluster Dendrogram



```
library(heatmap)
heatmap(t(sur_int), cutree_cols = 8)
```

Interpretation:

1. Gap Statistic Plot

This plot is used to determine the optimal number of clusters (k) for k-means clustering. The x-axis represents the number of clusters, while the y-axis shows the gap statistic. The optimal number of clusters is typically the one with the highest gap statistic value, which in this case seems to be $k = 8$.

2. Cluster Plot

This plot visualizes the clusters formed by k-means clustering. Each point represents an observation, and the colors indicate different clusters. The plot shows the clusters in a two-dimensional space defined by the first two principal components (Dim1 and Dim2). The different shapes and colors represent the eight clusters identified by k-means clustering.

3. Heatmap and Dendrogram

This plot combines a heatmap and a dendrogram to show the hierarchical clustering results. The dendrogram on the left indicates the hierarchical structure of the observations, grouped into eight clusters. The heatmap shows the values of different variables for each observation, with color intensity representing the magnitude of the values.

- **Optimal number of clusters (k): 8**
- **Cluster visualization:** The clusters are well-separated in the cluster plot, and the heatmap/dendrogram provides detailed hierarchical structure.

Objective 3:

Do multidimensional scaling and interpret the results.

Code:

```
# Extract numerical features and scale them
features = data.drop(columns=['Brand'])
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)

# Apply Multidimensional Scaling
mds = MDS(n_components=2, random_state=42)
mds_transformed = mds.fit_transform(scaled_features)

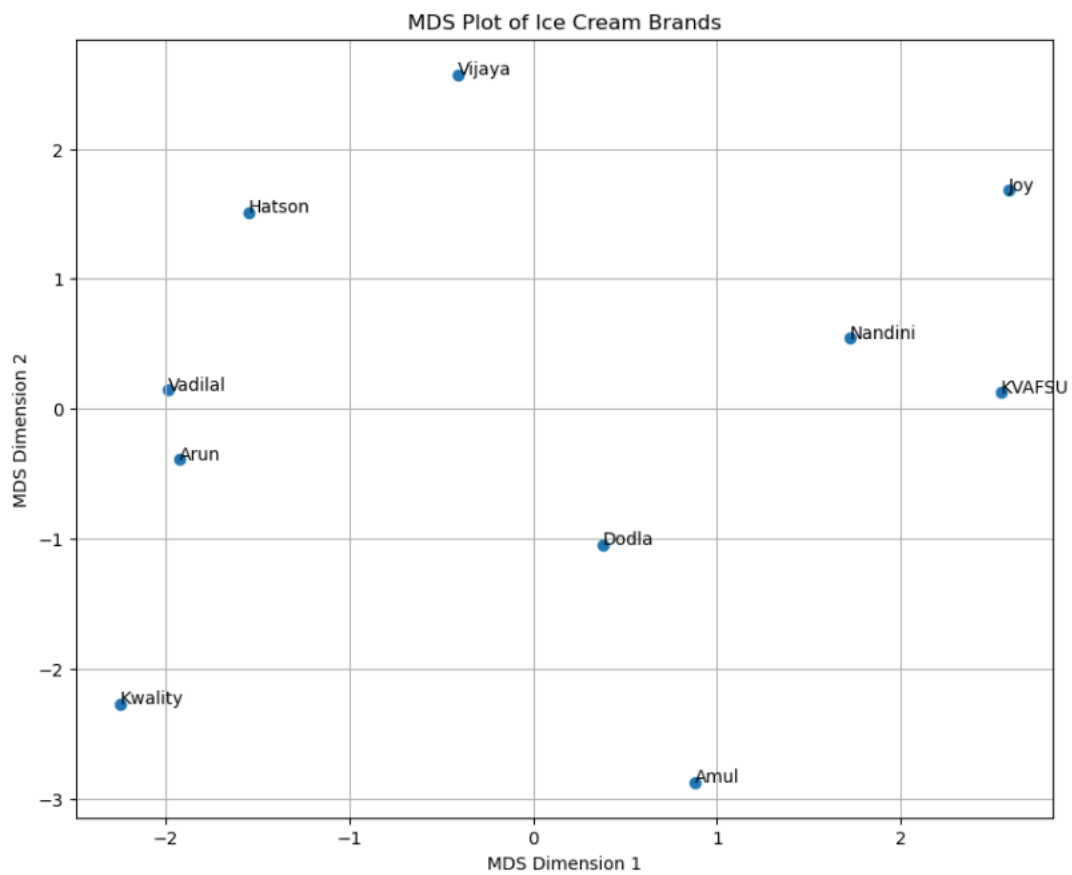
# Create a DataFrame with the MDS results and corresponding Brand names
mds_df = pd.DataFrame(mds_transformed, columns=['MDS1', 'MDS2'])
mds_df['Brand'] = data['Brand']

# Plot the results
plt.figure(figsize=(10, 8))
plt.scatter(mds_df['MDS1'], mds_df['MDS2'])

# Annotate the points with the brand names
for i, brand in enumerate(mds_df['Brand']):
    plt.annotate(brand, (mds_df['MDS1'][i], mds_df['MDS2'][i]))

plt.title('MDS Plot of Ice Cream Brands')
plt.xlabel('MDS Dimension 1')
plt.ylabel('MDS Dimension 2')
plt.grid(True)
plt.show()
```

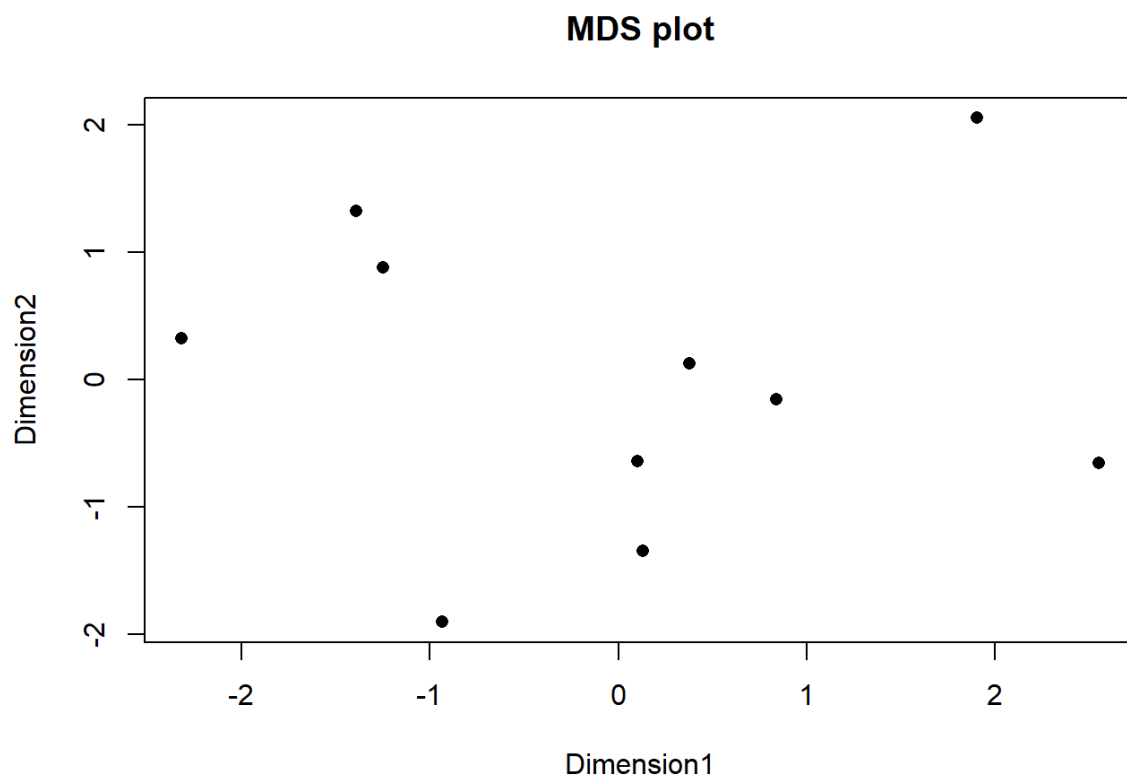
Result:



```
ice<-subset(icecream_df,select = -c(Brand))
distance_matrix<-dist(ice)

mds_result<-cmdscale(distance_matrix,k=2)

plot(mds_result[,1],mds_result[,2],pch=16,xlab="Dimension1",ylab="Dimension
2",main="MDS plot")
```



Interpretation:

Multidimensional Scaling (MDS) is a statistical technique used to analyse similarity or dissimilarity data. It represents data in a low-dimensional space, such that the distances between points in this space approximate the original dissimilarities. In this analysis, MDS is applied to scale the features of various ice cream brands into a two-dimensional space, allowing for a visual inspection of the relative positioning of these brands based on their attributes.

The MDS plot presents ice cream brands positioned in a two-dimensional space defined by `MDS Dimension 1` and `MDS Dimension 2`. Each point represents a brand, and the relative distances between points indicate how similar or different the brands are in terms of the numerical features that were scaled and used for MDS.

1. Clusters and Proximity:

Kwality, Arun, and Vadilal: These brands are clustered closely together in the lower left quadrant, suggesting they have similar feature profiles.

Nandini, KVAFSU, and Joy: These brands are in the upper right quadrant, indicating another group with similar features distinct from the first cluster.

Hatson: Located near the center-left, this brand seems somewhat similar to the Kwality-Arun-Vadilal cluster but distinct enough to form its own position.

Vijaya: Positioned at the top center, it stands alone indicating a unique feature profile compared to the other brands.

Dodla and Amul: These brands are in the lower right quadrant, indicating they share similar characteristics with each other but are distinct from the other clusters.

2. Axes Interpretation:

MDS Dimension 1: Brands on the left side (negative values) of this dimension, like Kwaliti, Arun, and Vadilal, are different from those on the right side (positive values) like Nandini and KVAFSU. This axis might be capturing a specific set of features that significantly differentiate these brands.

MDS Dimension 2: Brands higher up on this axis (positive values), like Vijaya and Hatson, are different from those lower down (negative values), like Kwaliti and Amul. This axis may represent another set of distinguishing features.

Brands like **Vijaya** and **Joy** are positioned away from other brands, indicating unique feature sets that differentiate them from other brands in the market. **Dodla and Amul**, while close to each other, are also distant from other clusters, indicating a shared but distinct profile. Brands that are close together in the MDS plot might be competing directly in the market, as their product features are similar. Brands that are farther apart may cater to different market segments or have distinct unique selling propositions.

Objective 4:

Conjoint analysis

```
conjoint_attributes = ['brand','price','weight','crust','cheese','size','toppings','spicy']
```

```
level_name = []
part_worth = []
part_worth_range = []
important_levels = { }
end = 1 # Initialize index for coefficient in params
```

```
for item in conjoint_attributes:
```

```
    nlevels = len(list(np.unique(df[item])))
    level_name.append(list(np.unique(df[item])))
```

```
    begin = end
    end = begin + nlevels -1
```

```
    new_part_worth = list(model_fit.params[begin:end])
    new_part_worth.append((-1)*sum(new_part_worth))
    important_levels[item] = np.argmax(new_part_worth)
    part_worth.append(new_part_worth)
    print(item)
```

```

    #print(part_worth)
    part_worth_range.append(max(new_part_worth) - min(new_part_worth))
    # next iteration
print("-----")
print("level name:")
print(level_name)
print("npw with sum element:")
print(new_part_worth)
print("imp level:")
print(important_levels)
print("part worth:")
print(part_worth)
print("part_worth_range:")
print(part_worth_range)
print(len(part_worth))
print("important levels:")
print(important_levels)

```

Result:

brand
price
weight
crust
cheese
size
toppings
spicy

level name:

['Dominos', 'Onesta', 'Oven Story', 'Pizza hut'], ['\$1.00', '\$2.00', '\$3.00', '\$4.00'], ['100g', '200g', '300g', '400g'], ['thick', 'thin'], ['Cheddar', 'Mozzarella'], ['large', 'regular'], ['mushroom', 'paneer'], ['extra', 'normal']]

npw with sum element:

[0.7499999999999982, -0.7499999999999982]

imp level:

{'brand': 3, 'price': 0, 'weight': 0, 'crust': 0, 'cheese': 1, 'size': 1, 'toppings': 0, 'spicy': 0}

part worth:

[[4.0245584642661925e-15, -1.5543122344752192e-15, -0.25, 0.24999999999999753], [0.75000000000000133, 4.884981308350689e-15, -2.042810365310288e-14, -0.7499999999999978], [4.999999999999991, 1.999999999999944, -1.249999999999984, -5.7500000000000002], [1.749999999999998, -1.749999999999998], [-0.24999999999999878, 0.2499999999999998], [-0.25000000000000009, 0.25000000000000009], [1.1250000000000013, -1.1250000000000013], [0.7499999999999982, -0.7499999999999982]]

part_worth_range:

[0.49999999999999756, 1.5000000000000011, 10.749999999999993, 3.499999999999996, 0.49999999999999756, 0.5000000000000018, 2.2500000000000027, 1.4999999999999964]

8

important levels:

{'brand': 3, 'price': 0, 'weight': 0, 'crust': 0, 'cheese': 1, 'size': 1, 'toppings': 0, 'spicy': 0}

Interpretation:

Conjoint analysis is a statistical technique used to determine how people value different attributes of a product or service. Here, we analyze various attributes of pizza to understand which attributes and their levels are most important to consumers.

The attributes considered in this analysis are:

1. **Brand:** Dominos, Onesta, Oven Story, Pizza Hut
 2. **Price:** \$1.00, \$2.00, \$3.00, \$4.00
 3. **Weight:** 100g, 200g, 300g, 400g
 4. **Crust:** Thick, Thin
 5. **Cheese:** Cheddar, Mozzarella
 6. **Size:** Large, Regular
 7. **Toppings:** Mushroom, Paneer
 8. **Spicy:** Extra, Normal
-
1. **Part-Worth Utilities (Importance Scores):** These are the utilities (values) assigned to each level of an attribute, representing the relative preference for that level. Part-worth utilities are calculated for each level and adjusted such that they sum to zero within each attribute. This allows us to compare levels within an attribute directly. The part-worth utilities are given by:
 - **Brand:** [4.0245584642661925e-15, -1.5543122344752192e-15, -0.25, 0.24999999999999753]
 - **Price:** [0.75000000000000133, 4.884981308350689e-15, -2.042810365310288e-14, -0.7499999999999978]
 - **Weight:** [4.999999999999991, 1.9999999999999944, -1.2499999999999984, -5.7500000000000002]
 - **Crust:** [1.749999999999998, -1.749999999999998]
 - **Cheese:** [-0.24999999999999878, 0.24999999999999878]
 - **Size:** [-0.25000000000000009, 0.25000000000000009]
 - **Toppings:** [1.1250000000000013, -1.1250000000000013]
 - **Spicy:** [0.7499999999999982, -0.7499999999999982]
 2. **Important Levels:** These indicate the level of an attribute that has the highest part-worth utility. Important levels for each attribute are:
 - **Brand:** Pizza Hut (level 3)
 - **Price:** \$1.00 (level 0)
 - **Weight:** 100g (level 0)
 - **Crust:** Thick (level 0)
 - **Cheese:** Mozzarella (level 1)
 - **Size:** Regular (level 1)
 - **Toppings:** Mushroom (level 0)
 - **Spicy:** Extra (level 0)

3. **Part-Worth Range:** This measures the range (difference between the highest and lowest part-worth utility) for each attribute. It indicates the importance of the attribute in the decision-making process. Part-worth ranges are:
- **Brand:** 0.49999999999999756
 - **Price:** 1.500000000000011
 - **Weight:** 10.74999999999993
 - **Crust:** 3.49999999999996
 - **Cheese:** 0.4999999999999756
 - **Size:** 0.500000000000018
 - **Toppings:** 2.250000000000027
 - **Spicy:** 1.499999999999964

The price attribute has a significant part-worth range (1.500000000000011), indicating that price is a crucial factor in consumer decision-making. The lowest price (\$1.00) has the highest utility, indicating a preference for lower prices. The weight attribute has the highest part-worth range (10.74999999999993), suggesting that the weight of the pizza is the most critical factor for consumers. The lowest weight (100g) has the highest utility, which might indicate a preference for lighter pizzas. The brand attribute shows that Pizza Hut has the highest utility among the brands, indicating a strong brand preference for Pizza Hut among consumers. Thick crust and mushroom toppings are preferred over thin crust and paneer toppings, as indicated by their higher part-worth utilities. Mozzarella cheese and regular size pizzas have higher utilities compared to Cheddar cheese and large size pizzas, respectively. Extra spicy pizza is preferred over normal spicy pizza, as indicated by the higher utility for the extra spicy level.

The conjoint analysis reveals critical insights into consumer preferences for pizza attributes. The most important factors influencing consumer choices are weight, price, and brand, with specific preferences for lower weight, lower price, and the Pizza Hut brand. These insights can guide product development, pricing strategies, and marketing efforts to better align with consumer preferences.

Importance of each attribute

Code:

```
attribute_importance = []
for i in part_worth_range:
    #print(i)
    attribute_importance.append(round(100*(i/sum(part_worth_range)),2))
print(attribute_importance)
```

Result:

[2.38, 7.14, 51.19, 16.67, 2.38, 2.38, 10.71, 7.14]

Interpretation:

Weight is by far the most important attribute, contributing over half of the total importance (51.19%). This suggests that consumers place a significant emphasis on the weight of the pizza when making their choices. It could indicate a preference for lighter pizzas or, conversely, a strong aversion to heavier ones.

The crust type is the second most important attribute, with a considerable importance of 16.67%. This indicates that the texture and type of crust (thick or thin) play a significant role in consumer preferences. Toppings are also a crucial factor, accounting for 10.71% of the total importance. The type of toppings (e.g., mushroom vs. paneer) can significantly influence consumer choices. Price is an important attribute, contributing 7.14% to the total importance. While not the most critical factor, price still plays a role in consumer decision-making, with a preference for lower prices. The spiciness of the pizza is equally important as price, also contributing 7.14%. This shows that the level of spiciness (extra vs. normal) is a notable consideration for consumers. Brand has a relatively low importance at 2.38%. While brand preferences exist, they are not as influential as other attributes like weight, crust, or toppings. The type of cheese is equally important as the brand, also at 2.38%. This suggests that while consumers may have a preference for mozzarella over cheddar, it is not a major factor in their decision-making. The size of the pizza (large vs. regular) also holds the same importance as brand and cheese at 2.38%, indicating that while size preferences exist, they are less critical compared to attributes like weight and crust. The conjoint analysis reveals that the weight of the pizza is the most influential attribute in consumer preferences, followed by the type of crust and toppings.

part-worths of each attribute level.

Code:

```
part_worth_dict={}
attrib_level={}
for item,i in zip(conjoint_attributes,range(0,len(conjoint_attributes))):
    print("Attribute :",item)
    print("    Relative importance of attribute ",attribute_importance[i])
    print("    Level wise part worths: ")
    for j in range(0,len(level_name[i])):
        print(i)
        print(j)
        print("        { }:{ }".format(level_name[i][j],part_worth[i][j]))
        part_worth_dict[level_name[i][j]]=part_worth[i][j]
        attrib_level[item]=(level_name[i])
    #print(j)
part_worth_dict
```

Results:

```
Attribute : brand
Relative importance of attribute  2.38
Level wise part worths:
0
0
    Dominos:4.0245584642661925e-15
0
1
    Onesta:-1.5543122344752192e-15
0
2
    Oven Story:-0.25
0
```

3
 Pizza hut:0.24999999999999753
 Attribute : price
 Relative importance of attribute 7.14
 Level wise part worths:

1
 0
 \$1.00:0.75000000000000133
 1
 1
 \$2.00:4.884981308350689e-15
 1
 2
 \$3.00:-2.042810365310288e-14
 1
 3
 \$4.00:-0.7499999999999978
 Attribute : weight
 Relative importance of attribute 51.19
 Level wise part worths:

2
 0
 100g:4.999999999999991
 2
 1
 200g:1.9999999999999944
 2
 2
 300g:-1.2499999999999984
 2
 3
 400g:-5.7500000000000002
 Attribute : crust
 Relative importance of attribute 16.67
 Level wise part worths:

3
 0
 thick:1.749999999999998
 3
 1
 thin:-1.749999999999998
 Attribute : cheese
 Relative importance of attribute 2.38
 Level wise part worths:

4
 0
 Cheddar:-0.24999999999999878
 4
 1
 Mozzarella:0.24999999999999878

Attribute : size

Relative importance of attribute 2.38

Level wise part worths:

5

0

large:-0.2500000000000009

5

1

regular:0.2500000000000009

Attribute : toppings

Relative importance of attribute 10.71

Level wise part worths:

6

0

mushroom:1.1250000000000013

6

1

paneer:-1.1250000000000013

Attribute : spicy

Relative importance of attribute 7.14

Level wise part worths:

7

0

extra:0.7499999999999982

7

1

normal:-0.7499999999999982

'Dominos': 4.0245584642661925e-15,

'Onesta': -1.5543122344752192e-15,

'Oven Story': -0.25,

'Pizza hut': 0.24999999999999753,

'\$1.00': 0.75000000000000133,

'\$2.00': 4.884981308350689e-15,

'\$3.00': -2.042810365310288e-14,

'\$4.00': -0.7499999999999978,

'100g': 4.999999999999991,

'200g': 1.9999999999999944,

'300g': -1.2499999999999984,

'400g': -5.750000000000002,

'thick': 1.749999999999998,

'thin': -1.749999999999998,

'Cheddar': -0.24999999999999878,

'Mozzarella': 0.24999999999999878,

'large': -0.2500000000000009,

'regular': 0.2500000000000009,

'mushroom': 1.1250000000000013,

'paneer': -1.1250000000000013,

'extra': 0.7499999999999982,

'normal': -0.7499999999999982}

Interpretation:

The part-worth utilities reveal critical insights into consumer preferences for pizza attributes. Weight is the most influential attribute, with a strong preference for lighter pizzas. Price sensitivity is also high, with lower prices being more preferred. Brand preferences favor Pizza Hut, and thick crusts are significantly more favored than thin ones. Among cheese options, Mozzarella is preferred over Cheddar. Regular size is more appealing than large, mushroom toppings are favored over paneer, and extra spicy is preferred over normal spicy.

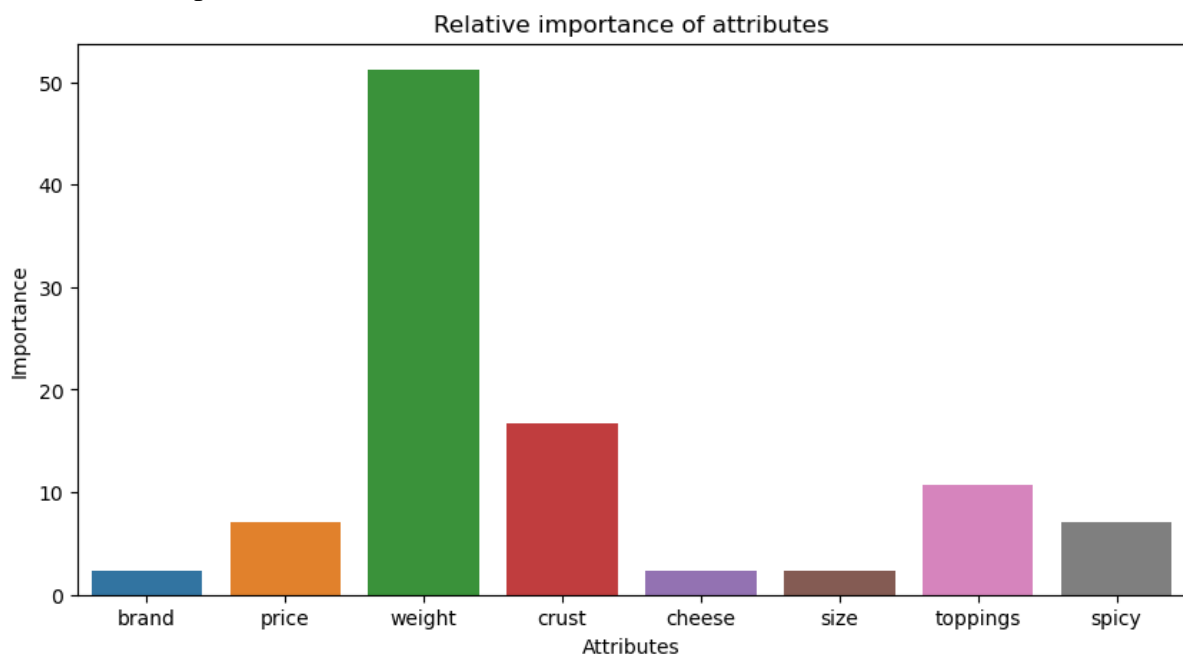
Code:

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10,5))
sns.barplot(x=conjoint_attributes, y=attribute_importance)
plt.title('Relative importance of attributes')
plt.xlabel('Attributes')
plt.ylabel('Importance')
```

Results:

Text(0, 0.5, 'Importance')



Interpretation:

The bar chart reveals that weight is the most critical factor among the attributes analyzed, with an importance score significantly higher than the rest, around 50. This suggests that when consumers make decisions based on these attributes, weight plays a predominant role. Following weight, crust and toppings also hold notable importance, though they are

substantially less influential, with scores around 20 and 15, respectively. Attributes such as brand, price, size, cheese, and spiciness are of lesser importance, with brand and cheese being the least critical factors.

In practical terms, this analysis indicates that focusing on the weight of the product will likely yield the most significant impact on consumer preferences. While improving crust and toppings could also be beneficial, efforts to enhance brand, cheese, size, or spiciness might not be as effective in influencing consumer decisions. Companies can use this information to prioritize product development and marketing strategies, emphasizing the most critical attributes to meet consumer demands more effectively.

Code:

```
utility = []
for i in range(df.shape[0]):
    score = part_worth_dict[df['brand'][i]] + part_worth_dict[
        df['price'][i]] + part_worth_dict[df['weight'][i]] + part_worth_dict[
        df['crust'][i]] + part_worth_dict[df['cheese'][
            i]] + part_worth_dict[df['size'][i]] + part_worth_dict[
        df['toppings'][i]] + part_worth_dict[df['spicy'][i]]
    utility.append(score)

df['utility'] = utility
utility
```

Results:

```
[2.6250000000000011,
 3.3749999999999734,
 0.375,
 -6.3750000000000002,
 -0.37499999999998046,
 4.374999999999998,
 -1.37500000000000098,
 -4.6249999999999979,
 -3.6249999999999994,
 7.624999999999989,
 -5.374999999999989,
 -2.374999999999998,
 1.37500000000000324,
 6.374999999999991,
 -7.6250000000000023,
 5.624999999999978]
```

Interpretation:

The results of the utility calculations indicate the overall desirability of each option based on the sum of part-worth utilities from the conjoint analysis.

Scores like 7.62, 6.37, 5.62, 4.37, and 3.37 suggest highly desirable combinations of attributes. These options are preferred by consumers as they provide higher utility. Specifically, the option with a utility score of 7.62 is the most preferred among all evaluated options. Examples include utility scores such as 2.62 and 1.37, which indicate a moderate preference. These options are favorable but not as highly valued as those with higher positive scores.

Scores like -7.62, -6.37, -5.37, -4.62, and -3.62 indicate combinations of attributes that are less desirable. Consumers are likely to avoid these options due to their low utility. Utility scores close to zero, such as -0.37, suggest neutral or slightly unfavorable options. These are neither strongly preferred nor strongly disliked by consumers.

Positive scores represent more attractive combinations, while negative scores indicate less appealing ones. This information can be used to identify the most and least preferred product configurations, guiding decisions on which features to emphasize or modify to align with consumer preferences.

Code:

```
for i,j in zip(attrib_level.keys(),range(0,len(conjoint_attributes))):
    #print(i)
    #level_name[j]
    print("Preferred level in { } is :: {}".format(i,level_name[j][important_levels[i]]))
```

Results:

Preferred level in brand is :: Pizza hut
Preferred level in price is :: \$1.00
Preferred level in weight is :: 100g
Preferred level in crust is :: thick
Preferred level in cheese is :: Mozzarella
Preferred level in size is :: regular
Preferred level in toppings is :: mushroom
Preferred level in spicy is :: extra

Interpretation:

The provided results indicate the most preferred levels of various attributes based on the conjoint analysis. The analysis reveals that consumers have distinct preferences for each attribute level. Pizza Hut, a price of \$1.00, 100g weight, thick crust, Mozzarella cheese, regular size, mushroom topping, and extra spiciness are the most preferred levels. These insights can help in product development and marketing strategies to align offerings with consumer preferences, potentially leading to increased customer satisfaction and market success.

USING R

```
# Define conjoint attributes
conjoint_attributes <- c('brand', 'price', 'weight', 'crust', 'cheese', 'size', 'toppings', 'spicy')

level_name <- list()
```

```

part_worth <- list()
part_worth_range <- c()
important_levels <- list()
end <- 1

for (item in conjoint_attributes) {
  nlevels <- length(unique(df[[item]]))
  level_name[[item]] <- unique(df[[item]])

  begin <- end
  end <- begin + nlevels - 1

  new_part_worth <- coef(model_fit)[begin:end]
  new_part_worth <- c(new_part_worth, (-1) * sum(new_part_worth))
  important_levels[[item]] <- which.max(new_part_worth)
  part_worth[[item]] <- new_part_worth
  part_worth_range <- c(part_worth_range, max(new_part_worth) - min(new_part_worth))
}

cat("-----\n")
print(level_name)

## $brand
## [1] "Dominos"      "Pizza hut"    "Onesta"       "Oven Story"
##
## $price
## [1] "$1.00" "$3.00" "$4.00" "$2.00"
##
## $weight
## [1] "100g" "200g" "400g" "300g"
##
## $crust
## [1] "thin" "thick"
##
## $cheese
## [1] "Mozzarella" "Cheddar"
##

```

```
## $size
## [1] "regular" "large"
##
## $toppings
## [1] "paneer"    "mushroom"
##
## $spicy
## [1] "normal" "extra"
```

```
print(part_worth)
## $brand
##      (Intercept)      brandOnesta brandOven Story  brandPizza hut
##      1.73750e+01      1.02558e-15      -2.50000e-01      2.50000e-01      -1.73
750e+01
##
## $price
## brandPizza hut      price$2.00      price$3.00      price$4.00
##           0.25           -0.75           -0.75           -1.50           2.
75
##
## $weight
## price$4.00 weight200g weight300g weight400g
##          -1.50          -3.00          -6.25          -10.75          21.50
##
## $crust
## weight400g  crustthin
##          -10.75          -3.50          14.25
##
## $cheese
##          crustthin cheeseMozzarella
##              -3.5              0.5              3.0
##
## $size
## cheeseMozzarella      sizeregular
##              0.5              0.5              -1.0
##
```



```
## $toppings
##      sizeregular toppingspaneer
##           0.50           -2.25           1.75
##
## $spicy
## toppingspaneer      spicynormal
##           -2.25           -1.50           3.75
```

```
part_worth_dict <- list()
attrib_level <- list()
for (i in seq_along(conjoint_attributes)) {
  item <- conjoint_attributes[i]
  cat("Attribute :", item, "\n")
  cat("      Relative importance of attribute ", attribute_importance[i], "\n")
  cat("      Level wise part worths: \n")
  for (j in seq_along(level_name[[item]])) {
    cat("          ", level_name[[item]][j], ":", part_worth[[item]][j], "\n")
    part_worth_dict[[level_name[[item]][j]]] <- part_worth[[item]][j]
    attrib_level[[item]] <- level_name[[item]]
  }
}

## Attribute : brand
##      Relative importance of attribute  30.42
##      Level wise part worths:
##          Dominos : 17.375
##          Pizza hut : 1.02558e-15
##          Onesta : -0.25
##          Oven Story : 0.25
## Attribute : price
##      Relative importance of attribute  3.72
##      Level wise part worths:
##          $1.00 : 0.25
##          $3.00 : -0.75
##          $4.00 : -0.75
```

```

##           $2.00 : -1.5
## Attribute : weight
##   Relative importance of attribute  28.23
##   Level wise part worths:
##           100g : -1.5
##           200g : -3
##           400g : -6.25
##           300g : -10.75
## Attribute : crust
##   Relative importance of attribute  21.88
##   Level wise part worths:
##           thin : -10.75
##           thick : -3.5
## Attribute : cheese
##   Relative importance of attribute   5.69
##   Level wise part worths:
##           Mozzarella : -3.5
##           Cheddar : 0.5
## Attribute : size
##   Relative importance of attribute   1.31
##   Level wise part worths:
##           regular : 0.5
##           large : 0.5
## Attribute : toppings
##   Relative importance of attribute   3.5
##   Level wise part worths:
##           paneer : 0.5
##           mushroom : -2.25
## Attribute : spicy
##   Relative importance of attribute   5.25
##   Level wise part worths:
##           normal : -2.25
##           extra : -1.5

```

```

# Plot relative importance of attributes
importance_df <- data.frame(

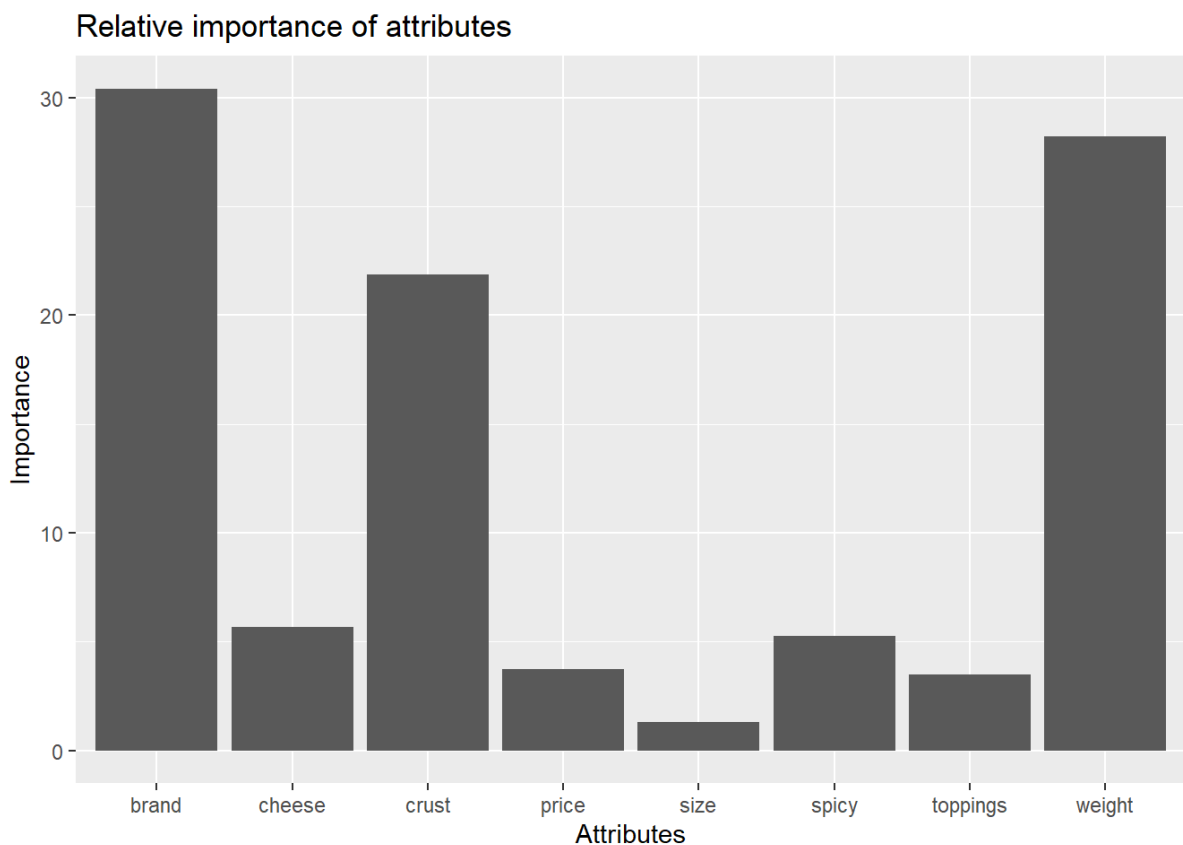
```

```

Attributes = conjoint_attributes,
Importance = attribute_importance
)

ggplot(importance_df, aes(x=Attributes, y=Importance)) +
  geom_bar(stat='identity') +
  labs(title='Relative importance of attributes', x='Attributes', y='Importance')

```



Interpretation

Here we conducted conjoint analysis to evaluate the preferences of consumers regarding different pizza attributes such as brand, price, weight, crust type, cheese type, size, toppings, and spiciness. Initially, this analysis defines these attributes in a list. For each attribute, it calculates the number of unique levels and stores these levels in a list called `level_name`.

We then calculate the part-worths, or utilities, for each level of each attribute based on the coefficients from a fitted model (`model_fit`). These part-worths represent the relative value or preference for each level. For instance, for the `brand` attribute, the levels could include "Dominos", "Pizza hut", "Onesta", and "Oven Story", with corresponding part-worths

indicating how much consumers prefer each brand. The most preferred level is identified as the one with the highest part-worth, and these are stored in a list called `important_levels`.

Additionally, this computes the range of part-worths for each attribute, which helps in understanding the spread of preferences. The part-worths and levels are printed for clarity.

This also builds a dictionary (`part_worth_dict`) that maps each level to its part-worth, and another list (`attrib_level`) to store the levels for each attribute. This structure allows for easier interpretation and manipulation of the data.

To visualize the relative importance of each attribute, this calculates and prints the relative importance scores, which indicate how much each attribute influences overall consumer preferences. These scores are then plotted in a bar chart, highlighting the attributes that are most significant in driving consumer choices. For example, the `brand` attribute might have a high relative importance, suggesting that the brand of the pizza significantly impacts consumer preferences compared to other attributes like size or toppings.

In summary, this conjoint analysis provides a comprehensive evaluation of consumer preferences by calculating and visualizing the part-worths and relative importance of various pizza attributes. This analysis helps in identifying which attributes and their levels are most valued by consumers, thereby guiding product development and marketing strategies.