VIRGINIA COMMONWEALTH UNIVERSITY

Statistical Analysis & Modelling

A4 – Multivariate Analysis

Using Python Colab

Submitted by

MONIKA SARADHA

#V01068784

Date of Submission: 07/05/2023

Table of Contents

1. Introduction

The datasets provided, Survey.csv and icecream.csv, provide useful information about consumer preferences and characteristics in two distinct domains. These datasets allow you to investigate and analyze consumer behavior, opinions, and choices in a variety of contexts. Both datasets provide opportunities for statistical techniques such as principal component analysis (PCA), factor analysis, cluster analysis, and multidimensional scaling to be applied. We can use these techniques to uncover underlying dimensions, group respondents based on their background variables, and analyze the relationships and similarities between variables or objects in datasets.

These datasets provide a means to investigate consumer preferences, comprehend market dynamics, and inform strategic decision-making in the domains of dairy products and ice cream consumption. These datasets can be analyzed to provide valuable insights into consumer behavior, assisting businesses in developing effective marketing strategies and meeting the changing needs of their target audience.

1.1. About the Data

Survey Dataset: This dataset provides insights into the preferences and factors influencing the decision-making process of Bangalore home buyers. It includes factors such as city, gender, age, occupation, monthly household income, desired house features, budget, and influencing factors. Analyzing this dataset provides useful information about potential homebuyers' demographics, income levels, and desired characteristics. It also reveals the significance of proximity to amenities, preferences for specific features, and the impact of price, builder reputation, and neighborhood profile. This dataset is a valuable resource for researchers, real estate professionals, and policymakers interested in understanding the Bangalore housing market and making informed decisions to meet the changing needs of homebuyers.

Ice-cream Dataset: The dataset provided provides information about the attributes and ratings of various brands in the dairy products market. Brand, price, availability, taste, flavor, consistency, and shelf life are all factors to consider. We can learn about consumer perceptions and preferences for various dairy brands by analyzing this dataset. The dataset allows users to evaluate and compare the performance of various brands based on a variety of criteria, providing useful information for market research, brand positioning, and decision-making in the highly competitive dairy industry.

1.2. Objective

The goal of this analysis is to apply various statistical techniques to the provided datasets (Survey.csv and icecream.csv), such as principal component analysis (PCA), factor analysis, cluster analysis, and multidimensional scaling.

- Principal Component Analysis (PCA) and Factor Analysis: Use PCA and factor analysis to identify the underlying dimensions or latent factors in the data. Extract the main components or factors that explain the most variance in the data and interpret their meaning in relation to the variables in the datasets.
- Cluster Analysis: Use cluster analysis to categorize respondents based on their demographics. Identify distinct segments or clusters within the data using appropriate clustering algorithms, characterizing respondents with similar characteristics or preferences.
- Multidimensional Scaling: Use multidimensional scaling to determine the similarity or dissimilarity of variables or objects in a dataset. Interpret multidimensional scaling results to gain insight into the relationships and proximity of variables or objects.

This way one can uncover patterns, structures, and relationships within the datasets, which will provide valuable insights for decision-making, market segmentation, and understanding consumer preferences in the given context.

1.3. Business Significance

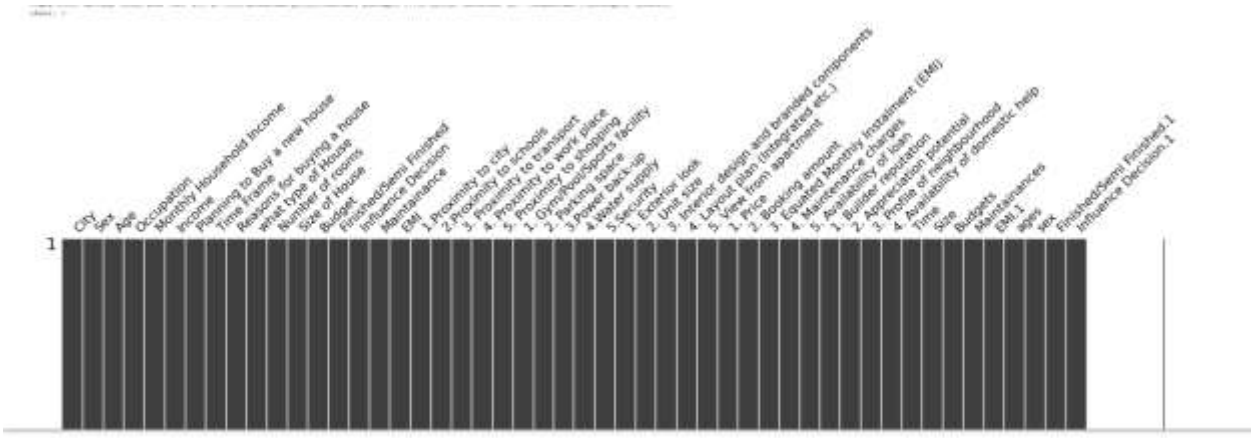
Survey Dataset: This dataset has significant business implications for various real estate stakeholders. Analyzing this dataset can provide valuable insights into the preferences and priorities of potential homebuyers in Bangalore for developers and builders. Understanding the desired features, budget ranges, and decision-making factors enables developers to align their construction and marketing strategies accordingly, ensuring that their projects cater to the needs and preferences of their target market. Real estate agents can use this data to provide personalized recommendations and guidance to clients, increasing customer satisfaction and conversion rates. Furthermore, policymakers and financial institutions can use this dataset to assess market trends, identify areas of potential growth, and tailor loan products and incentives to meet the population's housing needs. Overall, the dataset's commercial significance stems from its ability to inform

strategic decision-making, customer targeting, and market positioning in Bangalore's highly competitive real estate market.

Ice-cream Dataset: This dataset is extremely important for players in the dairy products industry. Companies can gain insights into consumer perceptions and preferences by analysing the attributes and ratings of various brands. Businesses can make informed decisions about product development, pricing strategies, and marketing campaigns by understanding the factors that influence consumer decisions, such as price, taste, flavour, consistency, and shelf life. By aligning their offerings with consumer preferences, companies can identify areas for improvement and develop competitive advantages. This dataset can also be used by market research teams to identify market trends, assess brand positioning, and make data-driven decisions to increase their market share. Overall, the commercial significance of this dataset stems from its ability to inform brand management strategies, product innovation, and marketing efforts in the competitive dairy products market.

2. Results

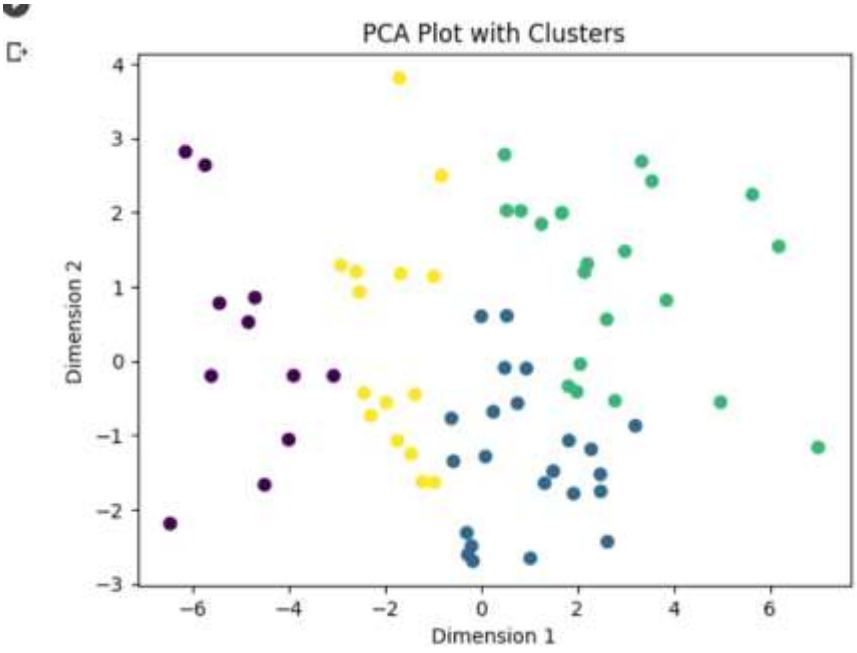
2.1. Data Preprocessing

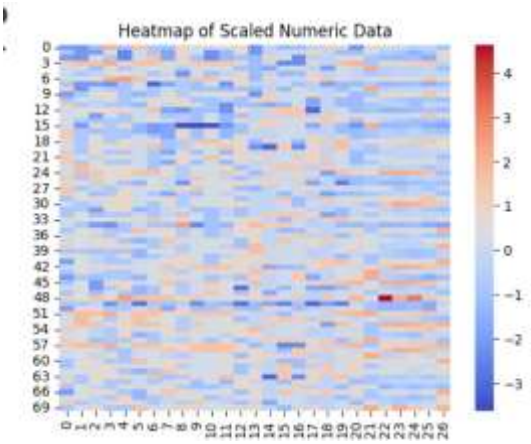


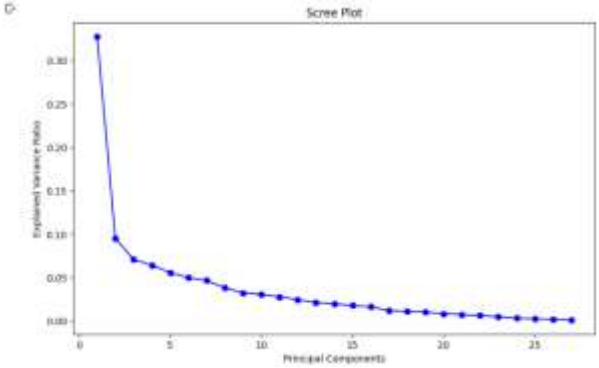
Inference:

The dataset was looked in for missing values and had null. Hence no processing steps was required.

2.2. Principal Component Analysis







Inference:

The dimensions in the dataset are represented by the following components based on the obtained Principal Component Analysis (PCA) output: Dim.1, Dim.2, Dim.3, and so on. These dimensions are derived from the dataset's variable analysis and represent orthogonal linear combinations of the original variables. Each dimension captures a different pattern or structure in the data and

represents a different amount of variance. Dim.1, for example, explains 31.6% of the variance in the data, Dim.2, 9.221% of the variance, Dim.3, 6.801% of the variance, and so on.

It is worth noting that the other dimensions (Dimensions 3–31) explain progressively less variance, implying that they contribute less to the overall structure of the data. These dimensions may capture additional patterns or noise that are less important in explaining the dataset's variation.

- Dimension 1: This dimension explains the highest variance in the data with an explained variance ratio of 0.3233. It has positive values for the first four data points and a negative value for the fifth data point.
- Dimension 2: This dimension explains the second-highest variance with an explained variance ratio of 0.0868. It has positive values for all the data points.

Based on the explained variance ratio, the dimensions in the dataset capture different amounts of variance. The values represent the proportion of variance explained by each dimension. Dimension 1 explains the highest amount of variance (32.33%), followed by Dimension 2 (8.68%), Dimension 3 (7.16%), and so on. The explained variance ratio provides insight into the significance of each dimension in capturing the variability in the dataset.

- **Dimension 1:** Strong positive loadings: '4. Proximity to work place', '5.Security', '1. Exterior look', '2. Unit size', '1. Builder reputation', '3. Profile of neighborhood', 'Maintainances'
  Strong negative loadings: '5. Proximity to shopping', '3.Power back-up', '4.Water supply', '2. Appreciation potential', 'Budgets', 'Maintainances'
- **Dimension 2:** Strong positive loadings: '2. Parking space', '1. Price', '4. Availability of domestic help', 'Maintainances'
  Strong negative loadings: '2. Booking amount', '3. Equated Monthly Instalment (EMI)', '4. Maintenance charges', '5. Availability of loan', '3. Profile of neighborhood', 'Budgets', 'Maintainances'
- **Dimension 3:** Strong positive loadings: 'ages'
  Strong negative loadings: None

The loadings indicate the correlation between the original variables and the principal components (dimensions). Positive loadings indicate a positive relationship, while negative loadings indicate a

negative relationship. By examining the loadings, we can infer which variables contribute more or less to each dimension.

2.3. Factor Analysis

Inference:

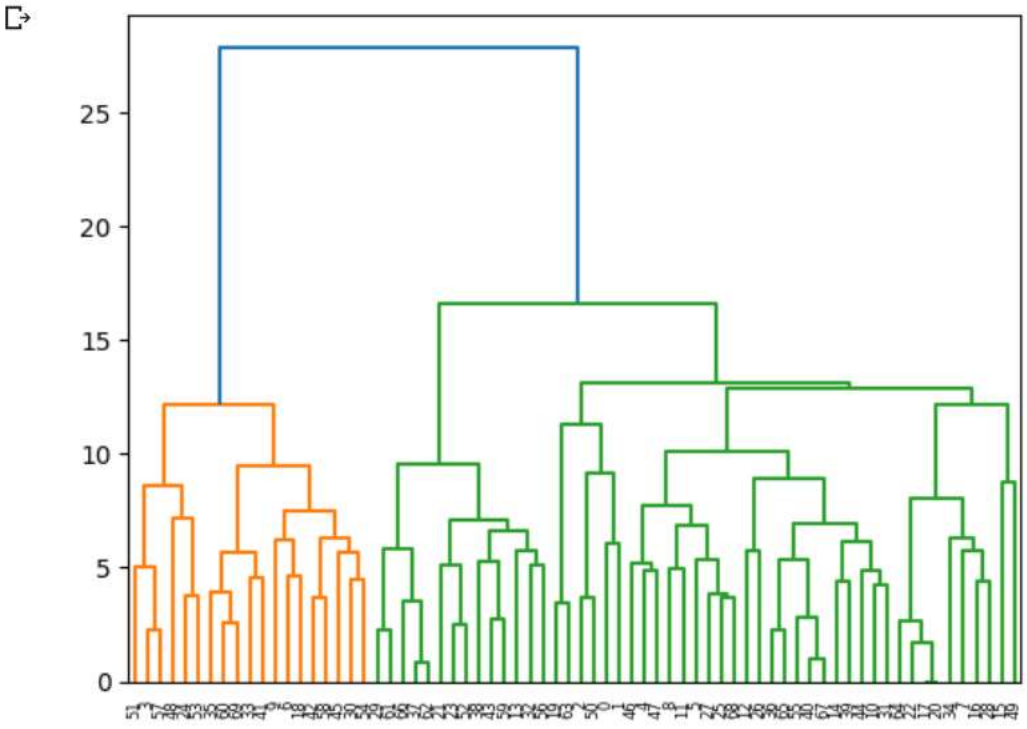
Based on the obtained factor loadings, we can make the following inferences for the Factor Analysis:

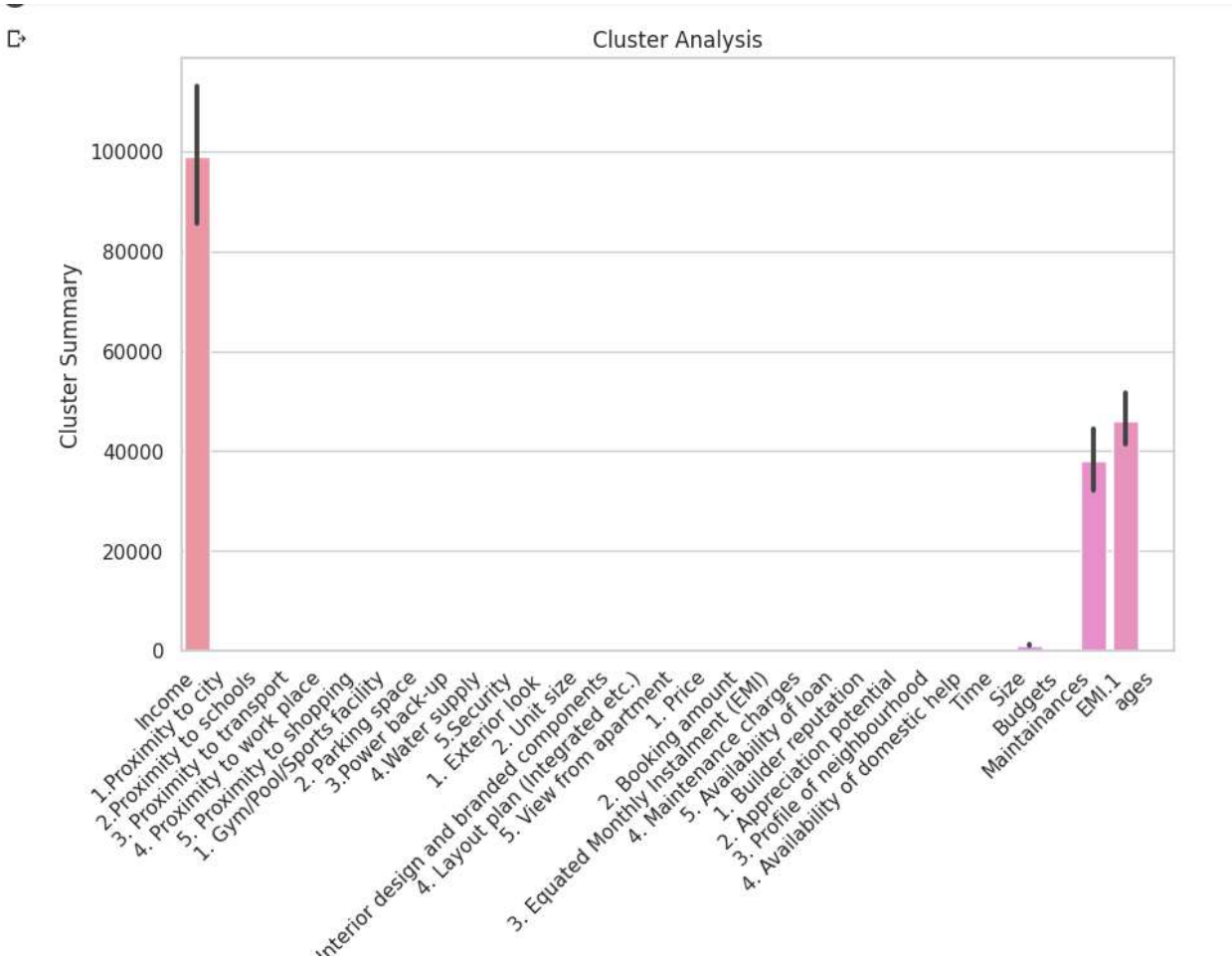
- Dimension 1: This dimension is positively influenced by variables related to proximity to work place, proximity to shopping, gym/pool/sports facility, parking space, power back-up, water supply, security, and exterior look. It suggests that these variables are correlated and contribute to the same underlying factor.
- Dimension 2: This dimension is positively influenced by variables related to unit size, interior design and branded components, view from apartment, booking amount, equated monthly installment (EMI), maintenance charges, and availability of loan. It indicates that these variables share a common factor.

- Dimension 3: This dimension is positively influenced by variables related to builder reputation, appreciation potential, profile of neighborhood, availability of domestic help, and size. These variables contribute to the same underlying factor.
- Dimension 4: This dimension is positively influenced by variables related to price, availability of loan, builder reputation, and profile of neighborhood. It suggests that these variables are correlated and represent a shared factor.

The factor loadings represent the strength and direction of the relationship between the original variables and the identified dimensions. Positive loadings indicate a positive relationship, while negative loadings indicate a negative relationship. The absolute values of the loadings represent the magnitude of the association.

2.4. Cluster Analysis





Inference:

Cluster 0:

- High proximity to work and shopping
- High amenities (gym/pool/sports facility, parking space)
- High scores for infrastructure (power back-up, water supply, security)
- High ratings for exterior look, unit size, and interior design
- High prices, booking amount, and maintenance charges
- High builder reputation, appreciation potential, and profile of the neighborhood
- High time, size, budgets, maintenances, EMI, and average age

Cluster 1:

- Moderate proximity to work and low proximity to shopping
- Moderate amenities and infrastructure scores
- Low ratings for exterior look, unit size, and interior design
- Moderate prices, booking amount, and maintenance charges
- Moderate builder reputation, appreciation potential, and profile of the neighborhood
- Moderate time, size, budgets, maintenances, EMI, and younger average age

Cluster 2:

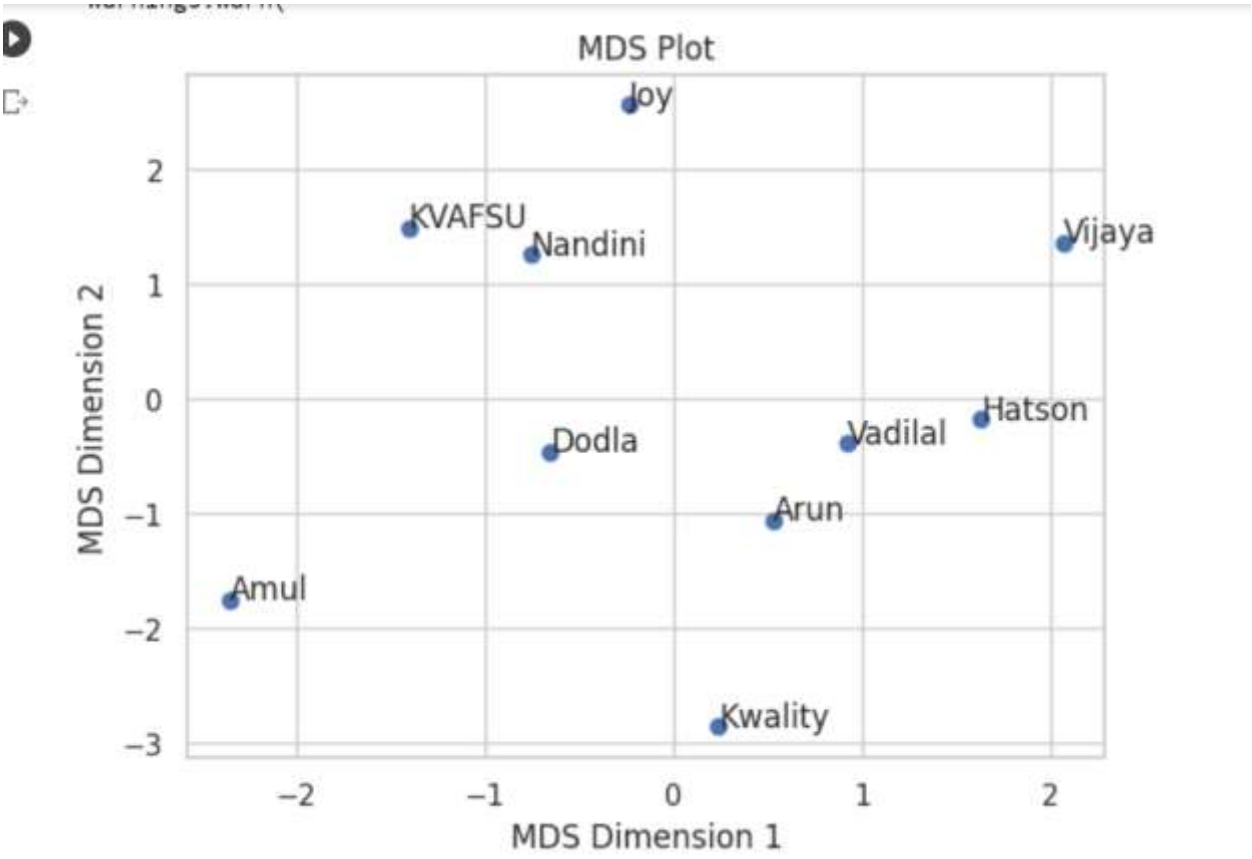
- Moderate proximity to work and very low proximity to shopping
- Low amenities and infrastructure scores
- Very low ratings for exterior look, unit size, and interior design
- Low prices, booking amount, and maintenance charges
- Low builder reputation, appreciation potential, and profile of the neighborhood
- Low time, size, budgets, maintainances, EMI, and younger average age

Cluster 3:

- Low proximity to work and moderate proximity to shopping
- High amenities (gym/pool/sports facility, security)
- Moderate infrastructure scores
- High ratings for exterior look, unit size, and interior design
- High prices, booking amount, and maintenance charges
- High builder reputation, appreciation potential, and profile of the neighborhood
- Low time, moderate size, high budgets, maintenances, EMI, and older average age

These clusters highlight key differences in respondents' preferences based on their background variables, including location preferences, amenities, infrastructure, pricing, and age.

2.5. Multidimensional Scaling



Inference:

Multidimensional scaling (MDS) is a technique used to visualize the similarity or dissimilarity between objects or cases based on a set of variables. By reducing the dimensionality of the data, MDS allows us to represent complex relationships in a lower-dimensional space, typically two or three dimensions. In the provided dataset, the MDS analysis has been performed, resulting in two dimensions: Dimension 1 and Dimension 2. Each item in the dataset is represented by its coordinates on these dimensions.

Based on the analysis, the variable "Taste" is strongly associated with Dimension 1, while the variable "Shelf life" is strongly associated with Dimension 2. This suggests that Dimension 1 captures the variability in taste preferences among the different ice cream brands, while Dimension

2 captures the variability in shelf-life ratings. Items that are closer to each other in the plot are more similar based on the variables used in the analysis, while items that are farther apart are more dissimilar.

- KVAFSU and Nandini are closely associated to Dimension 2 and similar to each other in terms of variables employed mostly with shelf-life.
- Hatson, Vadilal, Dodla and Arun are similarly associated with dimension 1 which is taste related variables.
- Arun is associated positively with both the dimensions of taste and shelf life.

3. Recommendation

3.1. Business Implications

- **Taste Differentiation:** According to the MDS analysis, taste is a significant factor in consumers' perceptions of ice cream brands. To differentiate themselves in the market, businesses should focus on developing unique and appealing flavours. This can be accomplished through continuous product innovation, consumer preference research, and feedback gathering to create flavors that cater to a wide range of tastes.
- **Shelf-Life Optimization:** The analysis emphasizes the importance of shelf life in the decision-making process of consumers. Brands with longer shelf lives, such as KVAFSU and Nandini, have an advantage. To extend the shelf life of their products, businesses must prioritize quality control measures, packaging innovation, and distribution strategies. Marketing efforts that communicate the longer shelf life to consumers can help build trust and loyalty.
- **Competitive Positioning:** The MDS plot provides information about the market positioning of ice cream brands based on taste and shelf life. Businesses can use this data to evaluate their competitive position in comparison to other brands and identify areas for improvement. Companies can improve their brand positioning strategies to attract target customers and gain a competitive advantage by leveraging their strengths and addressing any weaknesses.

3.2. Business Recommendations

- **Product Development:** Based on the MDS analysis, businesses should invest in R&D to introduce new and innovative flavors that cater to a variety of taste preferences. Consumer surveys, taste tests, and market trends can all provide useful information for flavor development. Updating the product portfolio on a regular basis with new and exciting flavors can help to attract a larger customer base and boost brand loyalty.
- **Quality Assurance and Packaging:** In order to increase shelf life and maintain product freshness, businesses should prioritize the implementation of robust quality assurance processes. This includes proper storage, temperature control monitoring, and optimizing

packaging materials. Investing in advanced packaging technologies that maintain product quality and extend shelf life can boost brand reputation and consumer satisfaction.

- **Marketing and communication:** Businesses should incorporate the MDS analysis findings into their marketing and communication strategies. To distinguish the brand from competitors, emphasize the distinct taste profiles and flavor varieties. Highlight the longer shelf life as a quality assurance measure that provides consumers with convenience. Engage with target audiences and raise brand awareness through various marketing channels such as social media, influencer collaborations, and interactive campaigns.
- **Customer Feedback and Continuous Improvement:** Seek customer feedback on a regular basis to better understand their preferences and gain insights into taste preferences and shelf-life expectations. To gather valuable feedback, conduct surveys, engage in social listening, and encourage customer reviews. This data can be used to drive product improvements, address quality concerns, and continuously improve the overall customer experience.

Businesses in the ice cream industry can improve their market positioning, meet customer expectations, and drive growth by offering appealing flavors, ensuring product quality, and differentiating themselves based on taste and shelf life by implementing these recommendations.

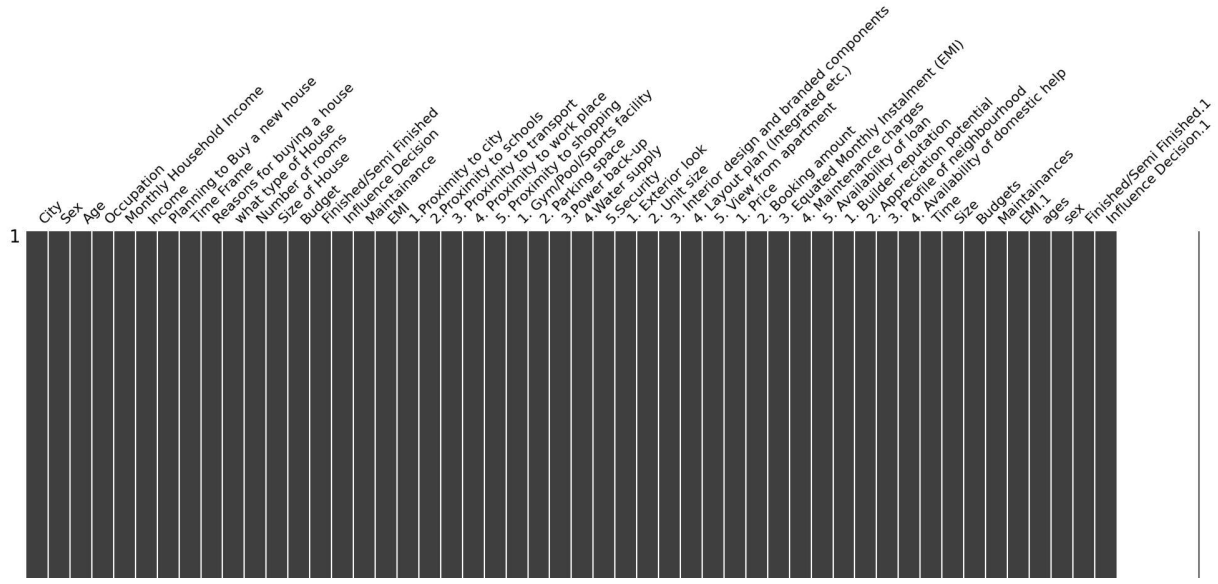
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.decomposition import PCA
!pip install factor-analyzer
from factor_analyzer import FactorAnalyzer
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import scipy.cluster.hierarchy as shc
from scipy.spatial.distance import pdist
from scipy.cluster.hierarchy import dendrogram
import plotly.express as px
```

```
Collecting factor-analyzer
  Downloading factor_analyzer-0.4.1.tar.gz (41 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 41.8/41.8 kB 2.9 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from factor-analyzer) (1.5.3)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from factor-analyzer) (1.10.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from factor-analyzer) (1.22.4)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (from factor-analyzer) (1.2.2)
Collecting pre-commit (from factor-analyzer)
  Downloading pre_commit-3.3.3-py2.py3-none-any.whl (202 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 202.8/202.8 kB 11.5 MB/s eta 0:00:00
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas->factor-analyzer) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->factor-analyzer) (2022.7.1)
Collecting cfgv>=2.0.0 (from pre-commit->factor-analyzer)
  Downloading cfgv-3.3.1-py2.py3-none-any.whl (7.3 kB)
Collecting identify>=1.0.0 (from pre-commit->factor-analyzer)
  Downloading identify-2.5.24-py2.py3-none-any.whl (98 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 98.8/98.8 kB 8.0 MB/s eta 0:00:00
Collecting nodeenv>=0.11.1 (from pre-commit->factor-analyzer)
  Downloading nodeenv-1.8.0-py2.py3-none-any.whl (22 kB)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from pre-commit->factor-analyzer) (6.0)
Collecting virtualenv>=20.10.0 (from pre-commit->factor-analyzer)
  Downloading virtualenv-20.23.1-py3-none-any.whl (3.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.3/3.3 MB 82.8 MB/s eta 0:00:00
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->factor-analyzer) (1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->factor-analyzer) (3.1
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from nodeenv>=0.11.1->pre-commit->factor-analyzer)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.1->pandas->factor-analyzer
Collecting distlib<1,>=0.3.6 (from virtualenv>=20.10.0->pre-commit->factor-analyzer)
  Downloading distlib-0.3.6-py2.py3-none-any.whl (468 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 468.5/468.5 kB 38.7 MB/s eta 0:00:00
Requirement already satisfied: filelock<4,>=3.12 in /usr/local/lib/python3.10/dist-packages (from virtualenv>=20.10.0->pre-commit->factc
Requirement already satisfied: platformdirs<4,>=3.5.1 in /usr/local/lib/python3.10/dist-packages (from virtualenv>=20.10.0->pre-commit->
Building wheels for collected packages: factor-analyzer
  Building wheel for factor-analyzer (pyproject.toml) ... done
  Created wheel for factor-analyzer: filename=factor_analyzer-0.4.1-py2.py3-none-any.whl size=42014 sha256=703bd1243f8e8ac86011572c99895
  Stored in directory: /root/.cache/pip/wheels/c5/94/da/41abe415f64706710726291086a814dd8b9e0dab1c491ef6ed
Successfully built factor-analyzer
Installing collected packages: distlib, virtualenv, nodeenv, identify, cfgv, pre-commit, factor-analyzer
Successfully installed cfgv-3.3.1 distlib-0.3.6 factor-analyzer-0.4.1 identify-2.5.24 nodeenv-1.8.0 pre-commit-3.3.3 virtualenv-20.23.1
```

```
survey = pd.read_csv("Survey.csv")
```

```
!pip install missingno
import missingno as msno
msno.matrix(survey)
```

```
Requirement already satisfied: missingno in /usr/local/lib/python3.10/dist-packages (0.5.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.22.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from missingno) (3.7.1)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from missingno) (1.10.1)
Requirement already satisfied: seaborn in /usr/local/lib/python3.10/dist-packages (from missingno) (0.12.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (1.
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (0.11.0
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (4
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (1
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (23.
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (8.4.0
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno) (3.
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->missingno)
Requirement already satisfied: pandas>=0.25 in /usr/local/lib/python3.10/dist-packages (from seaborn->missingno) (1.5.3)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.25->seaborn->missing
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib->
<Axes: >
```




```
survey.columns
```

```
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')
```

PCA

```
data = survey.iloc[:, 20:51].dropna()
numeric_data = data.select_dtypes(include=[np.number])
numeric_data_scaled = StandardScaler().fit_transform(numeric_data)

pca = PCA()
pca_result = pca.fit_transform(numeric_data_scaled)
explained_variance_ratio = pca.explained_variance_ratio_
pca_df = pd.DataFrame(pca_result, columns=['Dimension{}'.format(i+1) for i in range(pca_result.shape[1])])
print("Explained Variance Ratio:")
for i, ratio in enumerate(explained_variance_ratio):
    print("Dimension {}: {:.4f}".format(i+1, ratio))
```

```
Explained Variance Ratio:
Dimension 1: 0.3276
Dimension 2: 0.0949
```

```
Dimension 3: 0.0705
Dimension 4: 0.0640
Dimension 5: 0.0554
Dimension 6: 0.0496
Dimension 7: 0.0463
Dimension 8: 0.0379
Dimension 9: 0.0319
Dimension 10: 0.0304
Dimension 11: 0.0275
Dimension 12: 0.0242
Dimension 13: 0.0210
Dimension 14: 0.0196
Dimension 15: 0.0175
Dimension 16: 0.0159
Dimension 17: 0.0116
Dimension 18: 0.0110
Dimension 19: 0.0099
Dimension 20: 0.0080
Dimension 21: 0.0070
Dimension 22: 0.0059
Dimension 23: 0.0045
Dimension 24: 0.0028
Dimension 25: 0.0022
Dimension 26: 0.0017
Dimension 27: 0.0010
```

```
variable_names = numeric_data.columns
loadings = pca.components_
loadings_df = pd.DataFrame(loadings.T, columns=['Dimension{}'.format(i+1) for i in range(loadings.shape[0])], index=variable_names)
print("Loadings:")
print(loadings_df)
```

```
4. Layout plan (integrated etc.)          0.299510     0.163993
5. View from apartment                   -0.012088     0.172781
1. Price                                 -0.164617     0.183947
2. Booking amount                         0.008524    -0.165236
3. Equated Monthly Instalment (EMI)      -0.023443     0.039059
4. Maintenance charges                   -0.125185     0.063819
5. Availability of loan                   0.282898    -0.085808
1. Builder reputation                     0.412603    -0.186609
2. Appreciation potential                -0.088537    -0.016991
3. Profile of neighbourhood              -0.219837     0.018291
```

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
pca_result = pca.fit_transform(scaled_data)
pca_df = pd.DataFrame(pca_result, columns=['Dimension1', 'Dimension2'])

print("Principal Component Analysis:")
print(pca_df.head())
```

```
Principal Component Analysis:
   Dimension1  Dimension2
0    0.395665    3.344539
1    3.444209    3.048899
2    4.187523    3.464776
3   -6.162837    2.499158
4    0.595632   -0.695060
```

```
numeric_columns = survey.select_dtypes(include=[np.number])
numeric_data = numeric_columns.values
scaler = StandardScaler()
scaled_data = scaler.fit_transform(numeric_data)
pca = PCA()
pca.fit(scaled_data)

explained_variance_ratio = pca.explained_variance_ratio_
loadings = pca.components_

print("Explained Variance Ratio:")
for i, ratio in enumerate(explained_variance_ratio):
    print(f"Dimension {i+1}: {ratio:.4f}")
    print()
```

```
Explained Variance Ratio:
Dimension 1: 0.3233

Dimension 2: 0.0868

Dimension 3: 0.0716

Dimension 4: 0.0607

Dimension 5: 0.0563

Dimension 6: 0.0461

Dimension 7: 0.0438

Dimension 8: 0.0397

Dimension 9: 0.0353

Dimension 10: 0.0273

Dimension 11: 0.0245

Dimension 12: 0.0239

Dimension 13: 0.0198

Dimension 14: 0.0192

Dimension 15: 0.0175

Dimension 16: 0.0160

Dimension 17: 0.0147

Dimension 18: 0.0126

Dimension 19: 0.0097
```

```
        Dimension 20: 0.0095

        Dimension 21: 0.0072

        Dimension 22: 0.0067

        Dimension 23: 0.0063

        Dimension 24: 0.0062

        Dimension 25: 0.0044

        Dimension 26: 0.0031

        Dimension 27: 0.0022

        Dimension 28: 0.0019

        Dimension 29: 0.0016
```

```
# Visualization
plt.scatter(pca_df['Dimension1'], pca_df['Dimension2'], c=cluster_labels, cmap='viridis')
plt.xlabel('Dimension 1')
plt.ylabel('Dimension 2')
plt.title('PCA Plot with Clusters')
plt.show()

sns.heatmap(numeric_data_scaled, cmap='coolwarm', cbar=True)
plt.title('Heatmap of Scaled Numeric Data')
plt.show()
```

PCA Plot with Clusters

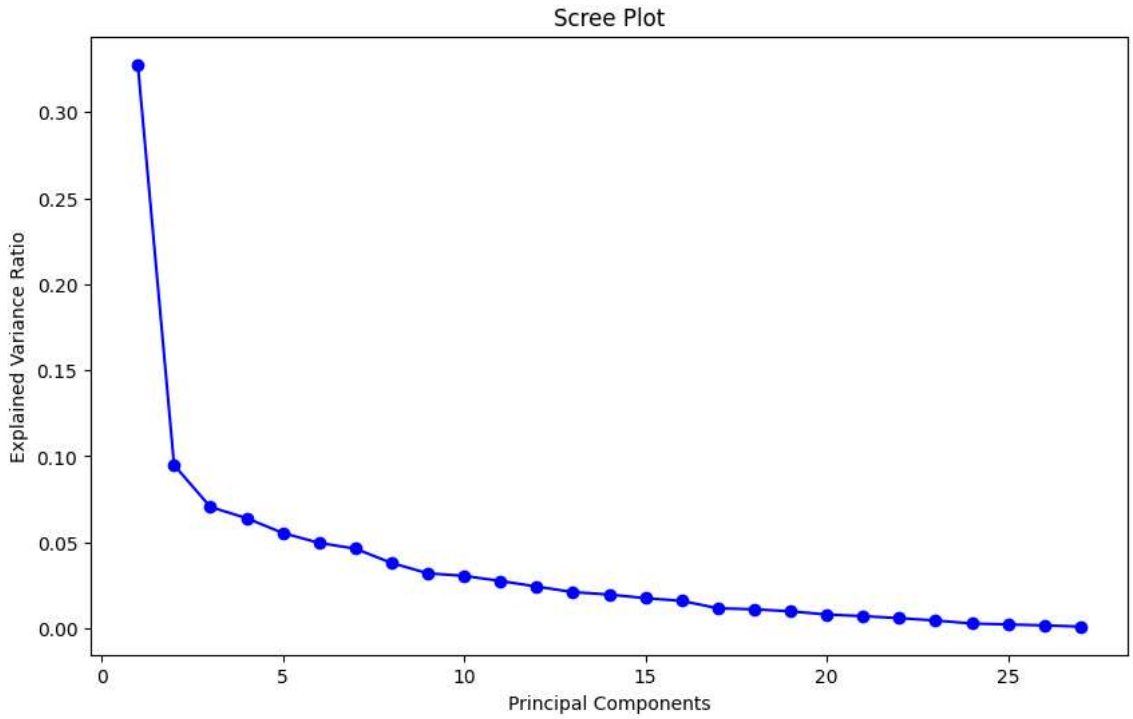
Factor Analysis

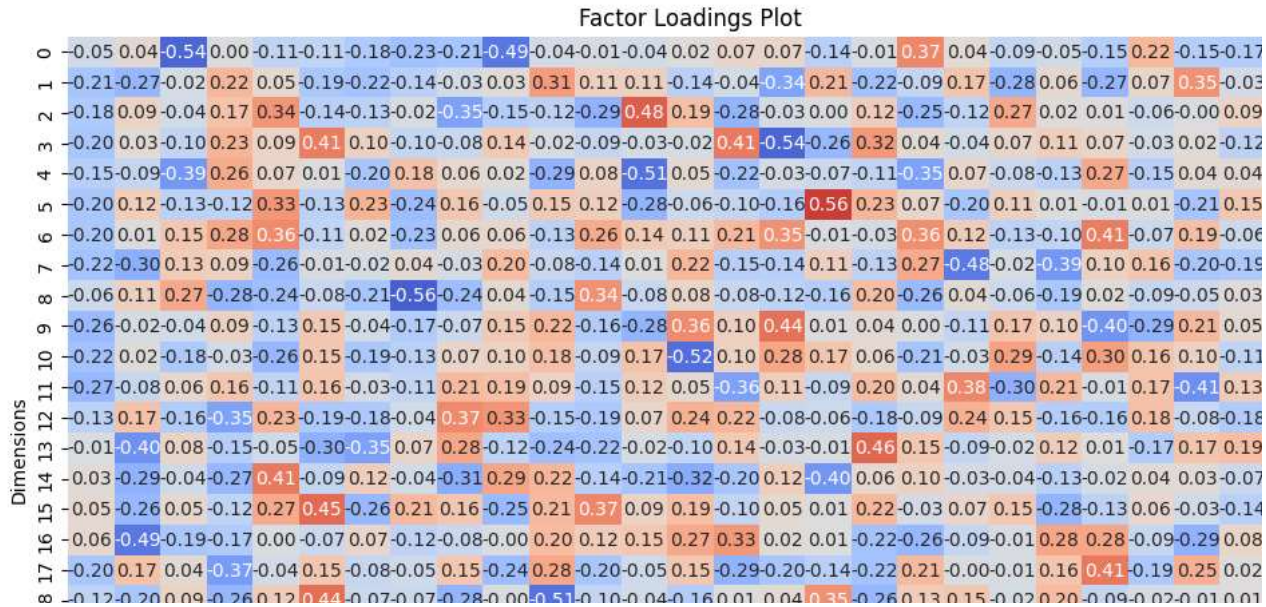
```
# Factor Analysis
factor_analyzer = FactorAnalyzer(n_factors=4, rotation="varimax")
factor_analyzer.fit(numeric_data_scaled)
factor_loadings = factor_analyzer.loadings_
communalities = factor_analyzer.get_communalities()
factor_scores = factor_analyzer.transform(numeric_data_scaled)

print("\nFactor Analysis - Loadings:")
print(factor_loadings)
```

```
Factor Analysis - Loadings:
[[ 0.21244413 -0.02548969 -0.05211908  0.5702185 ]
 [ 0.14830707  0.7049514   0.29088958  0.13435899]
 [ 0.25531624  0.45543691 -0.1909566   0.08322952]
 [ 0.27102902  0.52319381 -0.1525175   0.18406216]
 [ 0.16276873  0.44336957  0.00448474  0.54129574]
 [ 0.51009903  0.28814344 -0.15278718  0.08476692]
 [ 0.13633186  0.6478434  -0.14660039 -0.07888622]
 [ 0.2651974   0.631692    0.40902412 -0.10923306]
 [ 0.21433899 -0.00405345 -0.04301607 -0.3101373 ]
 [ 0.4492075   0.59763885 -0.01372466  0.08878531]
 [ 0.51173479  0.38706369 -0.02256002  0.19736827]
 [ 0.37478521  0.72242303  0.03608508 -0.0065669 ]
 [ 0.49817505 -0.00090909 -0.13055205  0.05840985]
 [-0.044179    0.07563258  0.54848482 -0.09408618]
 [-0.0039119  -0.10432144  0.39395568  0.00745916]
 [-0.14019166 -0.05459751  0.29327838 -0.03412517]
 [-0.07095006 -0.14078042  0.86001539  0.25469431]
 [ 0.69289274  0.1092861  -0.13432255 -0.1871625 ]
 [ 0.28203612  0.17598091  0.26260298 -0.11016062]
 [ 0.51334776  0.51168254 -0.22002309 -0.14471989]
 [ 0.1538711   0.74157034  0.11808503 -0.3761565 ]
 [ 0.00169223  0.09755872  0.35149559  0.02710705]
 [ 0.75405147  0.43585737  0.10093904 -0.0307936 ]
 [ 0.82996972  0.39116912  0.07858522 -0.06140399]
 [ 0.77143948  0.45025478  0.05773487  0.07226913]
 [ 0.78777     0.43627468 -0.02227957  0.06062078]
 [ 0.31974724  0.39228434 -0.10626731 -0.45303445]]
```

60 -

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

# Create a scree plot
plt.figure(figsize=(10, 6))
plt.plot(range(1, pca.n_components_ + 1), explained_variance_ratio, 'bo-')
plt.xlabel('Principal Components')
plt.ylabel('Explained Variance Ratio')
plt.title('Scree Plot')
plt.show()

# Create a factor loadings plot
plt.figure(figsize=(12, 8))
sns.heatmap(loadings.T, cmap='coolwarm', annot=True, fmt='.2f', cbar=False)
plt.xlabel('Original Variables')
plt.ylabel('Dimensions')
plt.title('Factor Loadings Plot')
plt.show()
```






```
# Hierarchical Clustering
distance_matrix = pdist(numeric_data_scaled, metric='euclidean')
linkage = shc.linkage(distance_matrix, method='ward')
dendrogram = shc.dendrogram(linkage)
```

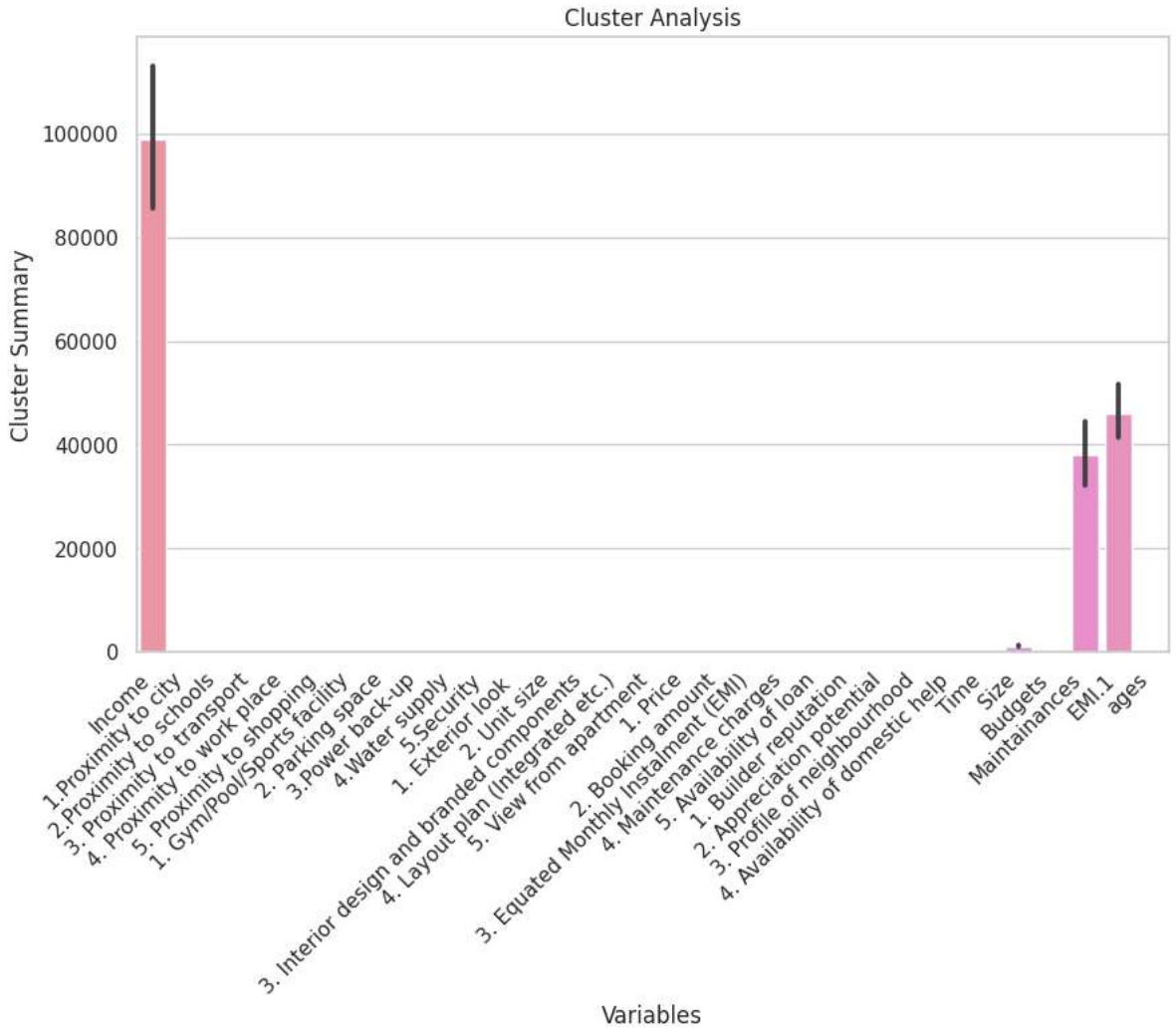

```
# Cluster Summary
numeric_data['Cluster'] = cluster_labels
cluster_summary = numeric_data.groupby('Cluster').mean()
print("\nCluster Summary:")
print(cluster_summary)
```

```
Cluster Summary:
         4. Proximity to work place  5. Proximity to shopping  \
Cluster
0                          4.181818                  3.454545
1                          3.913043                  2.869565
2                          3.904762                  1.904762
3                          3.400000                  2.666667

         1. Gym/Pool/Sports facility  2. Parking space  3.Power back-up  \
Cluster
0                           4.272727          4.181818         4.000000
1                           2.826087          3.521739         3.521739
2                           2.761905          3.095238         3.190476
3                           3.800000          3.666667         3.533333

         4.Water supply  5.Security  1. Exterior look   2. Unit size  \
Cluster
0              4.727273    4.545455           4.363636      3.636364
1              3.782609    3.652174           3.347826      3.130435
2              3.666667    3.095238           1.904762      3.571429
3              3.866667    3.933333           3.800000      3.400000

         3. Interior design and branded components  ...  \
Cluster                                             ...
0                                         4.727273  ...
1                                         3.956522  ...
2                                         3.095238  ...
3                                         4.066667  ...

         1. Builder reputation  2. Appreciation potential  \
Cluster
0                     4.909091                   4.272727
1                     3.956522                   4.086957
2                     4.142857                   3.952381
3                     4.733333                   4.533333

         3. Profile of neighbourhood  4. Availability of domestic help  \
Cluster
0                           4.545455                          4.181818
1                           3.608696                          3.086957
2                           3.380952                          2.380952
3                           4.333333                          3.533333

             Time         Size     Budgets  Maintainances         EMI.1  \
Cluster
0        9.000000  2109.090909  131.363636   81818.181818  77954.545455
1        9.391304   839.130435   42.065217   26521.739130  36195.652174
2        5.571429   700.000000   37.976190   20005.714286  29642.857143
3        5.400000  1413.333333   85.333333   48666.666667  61000.000000

              ages
Cluster
0        51.045455
1        39.173913
2        38.214286
3        55.866667
```

Cluster Analysis

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

sns.set(style="whitegrid")

fig, ax = plt.subplots(figsize=(10, 6))
sns.barplot(data=survey , ax=ax)
```

```
ax.set_xlabel('Variables')
ax.set_ylabel('Cluster Summary')
ax.set_title('Cluster Analysis')

# Rotate x-axis labels for better visibility
plt.xticks(rotation=45, ha='right')

# Show the plot
plt.show()
```




```
# Cluster Analysis
kmeans = KMeans(n_clusters=4, random_state=123)
kmeans.fit(numeric_data_scaled)
cluster_labels = kmeans.labels_
cluster_centers = kmeans.cluster_centers_
silhouette_avg = silhouette_score(numeric_data_scaled, cluster_labels)
print("\nCluster Analysis - Silhouette Score:", silhouette_avg)
print("Cluster Centers:")
print(cluster_centers)
```

```
Cluster Analysis - Silhouette Score: 0.09609440371511674
Cluster Centers:
[[ 3.62301222e-01  1.06200199e+00  9.14008851e-01  9.45093101e-01
   8.28416870e-01  1.21220439e+00  1.02668857e+00  9.78371179e-01
   3.56331589e-01  1.12487636e+00  1.11209642e+00  1.09585499e+00
   7.79812867e-01 -1.06679763e-01 -1.34875411e-01 -8.54211167e-01
  -2.12382848e-01  7.73359874e-01  1.66396427e-01  9.89792218e-01
   1.06537952e+00  3.37047055e-01  1.58812367e+00  1.66072309e+00
   1.68541092e+00  1.42767032e+00  5.22164514e-01]
 [ 7.50192269e-02  3.09859482e-01 -3.69883181e-01 -9.88471478e-03
   3.60181248e-02 -1.96337036e-01 -4.00533962e-02  1.44758528e-01
  -4.06384856e-01  1.44568481e-01  1.43749244e-02  1.85523738e-02
  -4.85970338e-01  3.57145292e-01  2.92484437e-01  4.03573802e-01
```

```
          8.17941266e-01 -4.95639355e-01 -1.38756440e-01 -3.29877709e-01
         -5.73220502e-02  4.15954392e-01 -4.50975335e-01 -5.45439172e-01
         -4.41579995e-01 -4.44316960e-01 -4.00718554e-01]
        [ 6.61674007e-02 -9.30643254e-01 -4.26844823e-01 -6.26930546e-01
         -5.12829491e-01 -3.69212413e-01 -7.05137567e-01 -1.03947497e+00
          2.58438295e-01 -9.50886674e-01 -6.85737134e-01 -1.05787182e+00
          4.92727552e-16 -4.28412380e-01 -6.09710761e-02 -3.29695538e-01
         -2.86903146e-01 -2.47405955e-01 -3.59814495e-01 -6.50713676e-01
         -7.81278314e-01 -3.54331519e-01 -6.74368689e-01 -6.46460634e-01
         -6.92220451e-01 -7.38068734e-01 -4.75319128e-01]
        [-4.73351405e-01  4.89812239e-02  4.94463805e-01  1.99791053e-01
          5.52277913e-02 -7.10023871e-02  2.95702851e-01  5.15829685e-01
          1.06264204e-16  2.84660342e-01  1.22453060e-01  6.48946576e-01
          1.73291748e-01  1.30386376e-01 -2.64207996e-01  4.69182112e-01
         -6.96764782e-01  5.39218107e-01  5.94476122e-01  6.90964007e-01
          4.00405136e-01 -3.88900448e-01  4.70987656e-01  5.23521351e-01
          4.10229948e-01  6.67623996e-01  8.96961252e-01]]
       /usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10
         warnings.warn(
```

```
import pandas as pd
from sklearn.manifold import MDS
import matplotlib.pyplot as plt
```

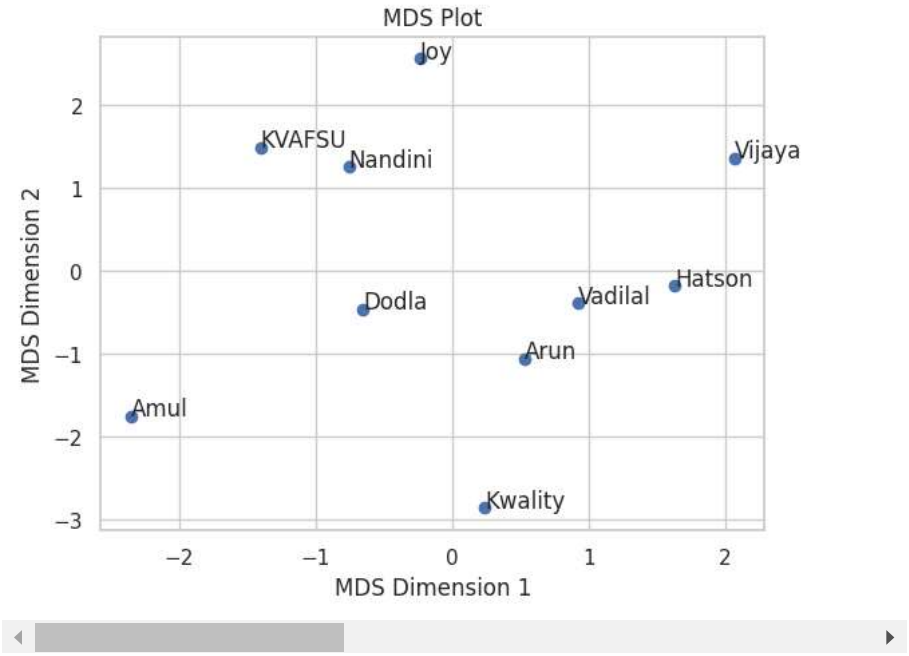
Multidimensional scaling

```
data = pd.read_csv("icecream.csv")
data.columns
```

```
Index(['Brand', 'Price', 'Availability', 'Taste', 'Flavour', 'Consistency',
       'Shelflife'],
      dtype='object')
```

```
columns = ['Price', 'Availability', 'Taste', 'Flavour', 'Consistency', 'Shelflife']
dissimilarity_data = data[columns]

dissimilarity_matrix = dissimilarity_data.to_numpy()

mds = MDS(n_components=2, dissimilarity='euclidean')
results = mds.fit_transform(dissimilarity_matrix)

plt.scatter(results[:, 0], results[:, 1])
plt.xlabel('MDS Dimension 1')
plt.ylabel('MDS Dimension 2')
plt.title('MDS Plot')

brands = data['Brand']
for i, brand in enumerate(brands):
    plt.annotate(brand, (results[i, 0], results[i, 1]))


plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/manifold/_mds.py:299: FutureWarning: The
  warnings.warn(
```



✓ 0s completed at 8:22 PM