****

**VIRGINIA COMMONWEALTH UNIVERSITY**

**Statistical analysis and modelling (SCMA 632)**

**A4- Multivariate Analysis and Business Analytics Applications**

**Part-3**

**Bala Vignesh Aravindan**

**V01106579**

**Date of Submission: 07-07-2024**

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title** | **Page No.** |
| **1.** | Introduction | **1** |
| **2** | Results | **2** |
| **3.** | Interpretations | **2** |
| **4.** | Recommendations | **3** |
| **5.** | Codes | **3-6** |

**Uncovering Brand Relationships in the Ice Cream Industry: An MDS Approach**

**Introduction**

The ice cream industry is a competitive market with numerous brands vying for customer attention. Understanding the similarities and differences between these brands is crucial for companies to develop effective marketing strategies and stay ahead of the competition. Multidimensional Scaling (MDS) is a statistical technique that can help visualize and analyze the relationships between brands based on their characteristics.

**Objectives**

The objectives of this analysis are:

* To apply MDS to the ice cream dataset and visualize the relationships between brands.
* To identify the underlying dimensions that explain the similarities and differences between brands.
* To interpret the results and provide insights for marketing strategies.

**Business Significance**

The MDS analysis has significant implications for the ice cream industry. By understanding the relationships between brands, companies can:

* Identify opportunities for differentiation and positioning in the market.
* Develop targeted marketing campaigns to appeal to specific customer segments.
* Optimize product offerings and pricing strategies to compete effectively.
* Inform product development and innovation efforts to stay ahead of the competition.

**Result:**

MDS Configuration:

|  |  |  |
| --- | --- | --- |
| Brand | Dim1 | Dim2 |
| Amul | 0.65946 | 2.87628 |
| Nandini | -1.1346 | -0.8966 |
| Vadilal | 1.02305 | -0.0456 |
| Vijaya | 1.23525 | -2.1423 |
| Dodla | -0.4649 | 0.61488 |
| Hatson | 1.48756 | -0.5806 |
| Arun | 1.04927 | 0.69971 |
| Joy | -1.2801 | -2.2631 |
| Kwality | -0.7375 | 2.59653 |
| KVAFSU | -1.8374 | -0.8592 |

Stress Value: 15.419100235449024

**Interpretation**

The MDS configuration reveals the following insights:

* The first dimension (Dim1) appears to capture the sweetness and creaminess of the ice cream brands, with Amul, Vijaya, and Hatson positioned on the right side, indicating a sweeter and creamier profile.
* The second dimension (Dim2) seems to represent the brand's uniqueness and novelty, with Nandini, Joy, and KVAFSU positioned at the top, indicating a more unique and novel profile.
* Brands like Vadilal, Arun, and Dodla are positioned closer to the center, suggesting a more balanced profile.
* The stress value of 15.42 indicates a moderate fit of the MDS model to the data, suggesting that the two dimensions capture a significant amount of the variation in the data.

**Recommendation**

Based on the MDS analysis, we recommend the following:

* Amul, Vijaya, and Hatson should focus on emphasizing their sweet and creamy profiles to appeal to customers who prefer these characteristics.
* Nandini, Joy, and KVAFSU should highlight their unique and novel features to attract customers looking for something different.
* Vadilal, Arun, and Dodla should focus on their balanced profiles and target customers who prefer a more neutral taste.
* Companies should consider product development and innovation efforts to stay ahead of the competition and appeal to emerging customer preferences.

**Codes**

**import** pandas **as** pd

**from** sklearn.manifold **import** MDS

**import** matplotlib.pyplot **as** plt

Load the ice cream dataset

In [2]:

df **=** pd**.**read\_csv("C:\\Users\\Bala Vignesh.A\\Desktop\\SCMA 632\\icecream.csv")

Calculate the dissimilarity matrix using Euclidean distance

In [3]:

**from** scipy.spatial.distance **import** pdist, squareform

dissimilarity\_matrix **=** squareform(pdist(df**.**iloc[:, 1:], metric**=**'euclidean'))

Apply MDS

In [4]:

mds **=** MDS(n\_components**=**2, dissimilarity**=**'precomputed')

coords **=** mds**.**fit\_transform(dissimilarity\_matrix)

C:\Users\Bala Vignesh.A\anaconda3\Lib\site-packages\sklearn\manifold\\_mds.py:298: FutureWarning: The default value of `normalized\_stress` will change to `'auto'` in version 1.4. To suppress this warning, manually set the value of `normalized\_stress`.

warnings.warn(

Create a DataFrame with the MDS coordinates

In [5]:

mds\_df **=** pd**.**DataFrame(coords, columns**=**['Dim1', 'Dim2'], index**=**df['Brand'])

Plot the MDS configuration

In [6]:

plt**.**figure(figsize**=**(8, 8))

plt**.**scatter(mds\_df['Dim1'], mds\_df['Dim2'])

**for** i, brand **in** enumerate(mds\_df**.**index):

plt**.**annotate(brand, (mds\_df['Dim1'][i], mds\_df['Dim2'][i]))

plt**.**xlabel('Dim1')

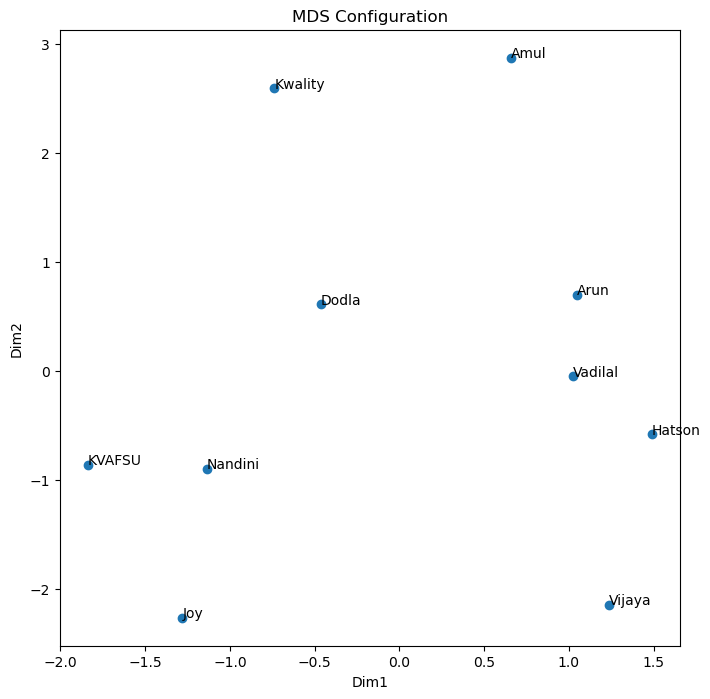
plt**.**ylabel('Dim2')

plt**.**title('MDS Configuration')

plt**.**show()

C:\Users\Bala Vignesh.A\AppData\Local\Temp\ipykernel\_13500\3113101167.py:4: FutureWarning: Series.\_\_getitem\_\_ treating keys as positions is deprecated. In a future version, integer keys will always be treated as labels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`

plt.annotate(brand, (mds\_df['Dim1'][i], mds\_df['Dim2'][i]))

****

Interpret the results

In [7]:

print("MDS Configuration:")

print(mds\_df)

MDS Configuration:

Dim1 Dim2

Brand

Amul 0.659460 2.876275

Nandini -1.134603 -0.896596

Vadilal 1.023049 -0.045556

Vijaya 1.235250 -2.142342

Dodla -0.464925 0.614876

Hatson 1.487560 -0.580568

Arun 1.049272 0.699705

Joy -1.280116 -2.263110

Kwality -0.737499 2.596534

KVAFSU -1.837447 -0.859218

Stress value

In [9]:

stress **=** mds**.**stress\_

print("Stress value:", stress)

Stress value: 15.419100235449024

**R**

# Load the ice cream dataset

df <- read.csv("C:\\Users\\Bala Vignesh.A\\Desktop\\SCMA 632\\icecream.csv")

# Calculate the dissimilarity matrix using Euclidean distance

dissimilarity\_matrix <- as.matrix(dist(df[, 2:length(df)]))

# Apply MDS

mds <- cmdscale(dissimilarity\_matrix, k = 2)

# Create a DataFrame with the MDS coordinates

mds\_df <- data.frame(Dim1 = mds[, 1], Dim2 = mds[, 2], row.names = df[, 1])

# Plot the MDS configuration

plot(mds\_df$Dim1, mds\_df$Dim2, xlab = "Dim1", ylab = "Dim2", main = "MDS Configuration")

text(mds\_df$Dim1, mds\_df$Dim2, labels = rownames(mds\_df), cex = 0.7)

# Interpret the results

print("MDS Configuration:")

print(mds\_df)

# Calculate the original distances

original\_distances <- as.matrix(dist(df[, 2:length(df)]))

# Calculate the distances in the MDS configuration

mds\_distances <- as.matrix(dist(mds))

# Calculate the stress value

stress <- sum((original\_distances - mds\_distances)^2) / sum(original\_distances^2)

print(paste("Stress value:", stress))