Step 1: Deciding (Not) to Segment

Key Considerations
We should carefully evaluate whether market segmentation is the right strategy for our organization. It requires long-term commitment, significant changes, and financial investments. We should move forward only if the benefits clearly outweigh the costs.

Implications of Committing

- We need to develop or modify products, pricing, distribution channels, and communication methods.
- The organization might need to restructure around market segments instead of products.
- Senior management should fully support this strategy and allocate sufficient resources.

Implementation Barriers

- **Leadership Issues**: If senior leadership isn't involved or doesn't provide enough resources, segmentation will likely fail.
- **Cultural Resistance**: Resistance to change, poor internal communication, and lack of collaboration can hinder progress.
- **Lack of Training**: If the team doesn't understand segmentation principles, the strategy won't succeed.
- **Resource Constraints**: Limited finances or capacity to make structural changes can block implementation.
- **Process Gaps**: Unclear goals, poor planning, or lack of time can make segmentation ineffective.

Checklist for Decision-Making
We should ask these questions before proceeding:

1. Are we market-oriented and willing to make changes?
2. Do we have a long-term outlook and openness to new ideas?
3. Are communication channels strong across departments?
4. Can we make significant structural changes?
5. Do we have the financial and human resources to support segmentation?

Action Plan

- Secure visible and active commitment from senior management.
- Form a qualified team with marketing and data expertise.
- Define clear objectives, assign responsibilities, and establish a structured process.

- Provide training to ensure everyone understands the concept and implications of market segmentation.

If we cannot meet these criteria, we should seriously reconsider pursuing segmentation as a strategy.

Step 2: Specifying the Ideal Target Segment

Knock-Out Criteria

Organizations must establish essential, non-negotiable features of market segments that ensure alignment with their strategic objectives. These criteria serve as filters to eliminate unsuitable segments early in the process. Key knock-out criteria include:

- **Homogeneity:** Members within a segment should have similar characteristics and behaviors.
- **Distinctiveness:** The segment should be clearly differentiated from others in the market.
- **Size:** The segment must be large enough to justify the costs of developing tailored marketing strategies.
- **Organizational Fit:** The organization's strengths and resources must align with the segment's needs.
- **Identifiability:** The ability to recognize and locate members of the segment in the marketplace.
- **Reachability:** Accessibility of the segment through communication and distribution channels.

These criteria should be clearly understood by senior management and the segmentation team. While some, like size, require specific numerical thresholds, others are qualitative and serve as guiding principles.

Attractiveness Criteria

Once segments pass the knock-out criteria, they should be assessed using attractiveness criteria to determine their relative appeal. These criteria are not binary but instead rated to measure the degree of attractiveness. Examples include:

- **Growth Potential:** Current growth trends and forecasts for the segment.
- **Competitive Advantage:** The organization’s ability to establish a defensible position within the segment.
- **Profitability:** Potential margins, return on investment, and overall financial viability.
- **Barriers to Entry:** Challenges competitors might face entering the segment.
- **Synergy with Other Segments:** Potential benefits or conflicts with targeting additional segments.

The segmentation team should prioritize and weight these criteria according to the organization's specific goals and resources.

Implementing a Structured Process

To ensure consistency and effectiveness, a structured approach for segment evaluation is recommended. One popular method is using a **segment evaluation plot**, which maps segment attractiveness against organizational competitiveness. This process involves:

1. Determining and agreeing on the factors contributing to segment attractiveness and competitiveness.
2. Limiting the number of factors to six or fewer to maintain focus and clarity.
3. Involving representatives from all organizational units to ensure diverse perspectives and buy-in.

While the segment evaluation plot is typically completed after data collection (Step 3) and segment identification, defining these criteria early in Step 2 provides clarity and direction for subsequent analysis.

By integrating these structured steps, organizations can identify and target market segments that align best with their objectives, ensuring efficient resource allocation and maximized potential for success.

Step 3: Collecting Data for market segmentation, emphasizing key concepts and methodologies:

5.1 Segmentation Variables:

- **Empirical Data**: Crucial for identifying and describing market segments.
- **Commonsense Segmentation**: Uses a single segmentation variable (e.g., gender) to divide consumers into segments. Descriptor variables (e.g., age, vacations, benefits sought) describe the segments.
    - Example: Table 5.1 splits the sample by gender and describes segments with socio-demographics and travel behavior.
- **Data-Driven Segmentation**: Uses multiple segmentation variables to identify patterns or create segments. Descriptor variables help detail these segments.
    - Example: Table 5.2 groups tourists based on shared benefits sought (e.g., relaxation, culture) rather than socio-demographic traits.

Key Insights:

- High-quality empirical data is essential for accurate segmentation and actionable insights (e.g., tailored products, pricing strategies, distribution channels).
- Data sources include surveys, observational data, and experimental studies. Behavioral data is preferred for reliability.

5.2 Segmentation Criteria:

- Refers to the type of information used for segmentation, such as geographic, socio-demographic, psychographic, or behavioral data.
- The choice of criterion depends on prior market knowledge and should prioritize simplicity and cost-effectiveness:
    - **Geographic Segmentation**: Based on consumer location. Effective for language-specific marketing or region-specific offerings (e.g., Amazon, IKEA).
        - **Advantages**: Easy assignment to segments; simplifies targeted communication.
        - **Disadvantages**: Geographic proximity doesn't always correlate with shared preferences or behavior.

Psychographic Segmentation: This method groups people based on psychological factors like beliefs, interests, or aspirations. It's more complex than other types of segmentation because it’s tough to pinpoint one characteristic that shows someone's psychological profile. Popular approaches include benefit segmentation (focuses on what consumers seek from products) and lifestyle segmentation (based on interests and activities). While psychographic segmentation

gives deeper insights into consumer behavior, it's harder to implement and requires reliable, valid data to be effective.

Behavioral Segmentation: This type groups people based on behaviors like purchase frequency, amount spent, or past experiences. It focuses on actual behavior rather than intentions or stated preferences. Behavioral segmentation has an advantage because it directly uses observable actions to define segments. However, it can be challenging to include potential customers who haven't interacted with the product before.

Data from Survey Studies: Surveys are commonly used for market segmentation because they're easy and cost-effective. However, survey data can be biased, which may affect the accuracy of segmentation results. Important factors to consider include:

1. **Choice of Variables**: Careful selection of segmentation variables is crucial. Including unnecessary variables can lead to confusion and poor results, as they introduce irrelevant or redundant data.
2. **Response Options**: The type of response options (binary, metric, or ordinal) influences the analysis. Binary and metric responses are ideal for segmentation since they provide clear data for analysis.
3. **Response Styles**: Biases in how respondents answer surveys can distort segmentation. For example, some people may consistently agree with statements (acquiescence bias), which could lead to inaccurate segments.
4. **Sample Size**: A larger sample size is needed for effective segmentation. Small sample sizes can make it difficult to determine the correct number of segments, leading to unreliable results.

These elements must be carefully managed to ensure high-quality market segmentation analysis.

the impact of sample size and data quality on the ability to perform effective market segmentation using segmentation algorithms.

- **Sample Size and Segmentation Accuracy**:
    - Increasing sample size leads to better results in identifying correct market segments. However, the biggest improvements are seen when sample sizes are small, and marginal benefits diminish as the sample size increases.
    - Dolnicar et al. (2014) recommend a sample size of at least 60 times the number of segmentation variables (denoted as p) for general cases. For more complex scenarios, such as those with difficult artificial data, Dolnicar et al. (2014) suggest a sample size of at least 70 p.
    - There is a diminishing return in accuracy improvements once sample sizes surpass these recommendations.
- **Market Characteristics Impacting Segmentation**:
    - Dolnicar et al. (2016) extended research to account for market characteristics such as:

        - The number of market segments.
        - Whether these segments are equal or unequal in size.
        - The degree of overlap between the segments.
    - Unequal segment sizes and overlapping segments make it harder for segmentation algorithms to identify correct solutions.
- **Data Characteristics Affecting Segmentation**:
    - Factors such as sampling error, response biases, response styles, low data quality, irrelevant items, and correlations between item groups can significantly impact the accuracy of market segmentation.
    - A study by Dolnicar et al. (2016) shows that larger sample sizes improve the ability to recover market segments, though the effect varies depending on the market and data characteristics.
    - **Key Points from the Study**:
        - The presence of uncorrelated segmentation variables makes recovery easier.
        - High correlation between variables makes segmentation difficult to recover, even with a larger sample size.
        - A small number of noisy variables has minimal impact on segmentation performance.
        - **Recommendation**: The data should have at least 100 respondents per segmentation variable.
- **Data Quality Factors for Market Segmentation**:
    - Data used for market segmentation should ideally:
        - Contain all necessary items and exclude unnecessary ones.
        - Avoid correlated items.
        - Provide high-quality responses free from biases like response styles.
        - Be binary or metric in nature.
        - Contain responses from a suitable sample size (at least 100 times the number of segmentation variables).
    - High-quality, unbiased data is essential for producing meaningful market segmentation results.
- **Data from Internal Sources**:
    - Organizations increasingly use internal data (e.g., scanner data, booking data, online purchase data) to perform market segmentation analysis.
    - Advantages:
        - Internal data represent actual consumer behavior, avoiding biases like social desirability or imperfect memory seen in consumer surveys.
        - These data are often automatically generated and stored in accessible formats.
    - Disadvantage:
        - Internal data may over-represent current customers, which could lead to biased conclusions, especially when trying to attract future customers whose behavior may differ from existing ones.
- **Data from Experimental Studies**:

    - Experimental data from field or laboratory experiments can be valuable for market segmentation. For example, consumer responses to advertisements or choice experiments can serve as segmentation criteria.
    - Conjoint studies and choice experiments help determine how different product attributes influence consumer preferences, which can be used to segment markets based on consumer preferences.

Step 3 Checklist

1. **Team Meeting**: Convene a meeting to discuss potential segmentation variables.
2. **Segmentation Variables**: Identify which consumer characteristics could serve as promising segmentation variables.
3. **Descriptor Variables**: Determine other characteristics that are important for understanding market segments.
4. **Data Collection Design**: Plan how to collect data validly, minimizing biases and errors.
5. **Data Collection**: Collect data, ensuring it meets the necessary criteria for segmentation analysis.

By following these steps, the segmentation process becomes more reliable and effective in identifying distinct market segments.

Step 4: Exploring Data – Part 1

6.1 A First Glimpse at the Data

After data collection, exploratory data analysis (EDA) helps clean and preprocess the data, providing insights into the most appropriate algorithms for segmentation. EDA helps by:

- Identifying the measurement levels of the variables.
- Investigating the univariate distributions of each variable.
- Assessing dependencies between variables.

The data may need preprocessing before it can be used for segmentation algorithms. The results of this exploration stage guide the selection of suitable segmentation methods.

In this case, we illustrate data exploration using a travel motives dataset, which includes 20 travel motives collected from 1,000 Australian residents. These responses are in relation to their last vacation, with one example being: "I AM INTERESTED IN THE LIFESTYLE OF LOCAL PEOPLE."

The CSV file containing the data can be found in the R package `MSA`, and you can load the data into R using:

R

```
vaccsv <- system.file("csv/vacation.csv", package = "MSA")

file.copy(vaccsv, ".")
```

Alternatively, the CSV file can be downloaded from the book's website. Once the data is available, read it into R with:

R

```
vac <- read.csv("vacation.csv", check.names = FALSE)
```

The `check.names = FALSE` argument prevents R from converting spaces in column names to dots. After reading the data into R, it is stored in a data frame named `vac`.

You can inspect the `vac` object to learn about the column names and size of the dataset using:

R

```
colnames(vac)

dim(vac)
```

For instance, the dataset has 1,000 rows and 32 columns. To view a summary of the data, you can use the `summary()` function, which generates a statistical summary of the dataset.

For example, to view the summary of selected columns:

R

```
summary(vac[, c(1, 2, 4, 5)])
```

6.2 Data Cleaning

Data cleaning is essential before starting data analysis. This includes:

- Checking for plausible values (e.g., age should be between 0 and 110).
- Ensuring categorical variables have consistent labels (e.g., gender should only be "female" or "male").

In the travel motives dataset, the `Gender` and `Age` columns do not require cleaning. However, the `Income2` variable has improperly ordered categories, which can be corrected by reordering them as follows:

R

```
inc2 <- vac$Income2

levels(inc2)

lev <- levels(inc2)

lev[c(1, 3, 4, 5, 2)]

inc2 <- factor(inc2, levels = lev[c(1, 3, 4, 5, 2)], ordered = TRUE)
```

To verify the transformation, use:

R

```
table(orig = vac$Income2, new = inc2)
```

Once verified, you can overwrite the original column with the cleaned version:

R

```
vac$Income2 <- inc2
```

After cleaning, save the modified data frame for future use:

R

```
save(vac, file = "cleaned_vacation_data.RData")
```

6.3 Descriptive Analysis

Descriptive statistics help in understanding the data and preventing misinterpretation. Use the `summary()` function for a numeric summary of the data, and visualizations like histograms, boxplots, and scatter plots for numeric variables. For categorical variables, bar plots and mosaic plots are useful.

For example, to create a histogram of the `Age` variable, use the `lattice` package:

R

```
library("lattice")

histogram(~ Age, data = vac)
```

This will generate a histogram of the `Age` distribution. You can adjust the bin width for more detailed analysis using the `breaks` argument:

R

```
histogram(~ Age, data = vac, breaks = 50, type = "density")
```

This will create a density plot with finer bins, revealing more detailed insights into the age distribution.

By performing these exploratory steps, you lay the foundation for further analysis and segmentation tasks.

6.4 Pre-Processing

6.4.1 Categorical Variables

- **Merging Levels**: When categorical variables have many levels, it might be useful to merge some of them. For example, income categories that are too granular can be merged to make the data more balanced, as demonstrated with the `Income` variable in the example.
    - **Example**: Income categories like "$210,001 to $240,000" and "more than $240,001" had very few observations. These were merged into a new category (`Income2`), which balanced the frequency distribution, making the data easier to analyze.
- **Converting to Numeric Variables**: Ordinal data can sometimes be converted to numeric if the distances between levels are assumed to be equal. For example, income and Likert scale agreement items are ordinal and can be transformed into numeric values for analysis, assuming equal spacing between levels.
    - **Example**: A Likert scale like "Strongly Disagree" to "Strongly Agree" can be treated as numeric if the distances between the points are considered equal. However, there's a caveat that response styles may distort these distances.
- **Binary Variables**: Converting categorical variables to binary (0/1) values is straightforward and often used for simplifying categorical data. For example, turning travel motives into a binary matrix where "YES" is 1 and "NO" is 0.
    - **Example**: Travel motives were converted into a numeric matrix using the `vacmot` variable, which is a logical matrix converted to 0 and 1.

6.4.2 Numeric Variables

- **Standardization**: To ensure that segmentation methods treat all variables fairly, numeric variables are often standardized. This means transforming the data so that each variable has a mean of 0 and a standard deviation of 1. This step prevents variables with larger ranges (like income or expenditures) from dominating the analysis.
    - **Example**: The `vacmot` matrix was standardized using the `scale()` function, which subtracts the mean and divides by the standard deviation for each variable.
- **Robust Standardization**: In cases with outliers, robust measures (like median and interquartile range) are preferred over the mean and standard deviation.

6.5 Principal Components Analysis (PCA)

- **PCA Overview**: PCA is a dimensionality reduction technique that transforms a dataset into principal components (PCs), which are uncorrelated and ordered by variance. The first component captures the most variability, the second captures the second most, and so on. PCA helps in reducing the complexity of the data while retaining most of the information.
    - **Example**: PCA was applied to the `vacmot` dataset. The first few components were examined to understand the dominant variables contributing to the variation in the data.
- **PCA Output**: The principal components are evaluated based on their standard deviations and the proportion of variance they explain. The first principal component often captures the most important patterns in the data.
    - **Example**: In the PCA output, the first component (`PC1`) captured 18% of the variance, while the second component (`PC2`) captured 9%, and so on.
- **Interpretation of Principal Components**: The rotation matrix in PCA shows how the original variables contribute to each principal component. For instance, `PC1` might separate respondents with minimal travel motives from those with high travel motivations.

This process of transforming categorical and numeric data, standardizing variables, and performing PCA is commonly used for segmenting data and uncovering patterns, especially when working with large, complex datasets.

Step 8: Selecting the Target Segment(s)

In the context of market segmentation analysis, it emphasizes the importance of making informed decisions about which market segments to target based on a comprehensive evaluation of each segment's attractiveness and the organization's competitiveness.

Here is a summary of the key points from the text:

1. **Targeting Decision**:
    - Step 8 is about choosing which market segments to focus on from the previously identified ones.
    - A segmentation team will assess segments based on criteria defined earlier in the process (in Steps 2, 6, and 7).
    - The team needs to ensure that all segments under consideration have passed the "knock-out criteria" (i.e., they must be large enough, identifiable, reachable, and their needs must align with the organization's offerings).
2. **Market Segment Evaluation**:
    - To evaluate which segments to select, a **decision matrix** can be used. This matrix helps visualize two dimensions:
        1. **Segment attractiveness**: How appealing the segment is to the organization.
        2. **Organizational competitiveness**: How likely the organization is to attract the segment, given competitors.
    - These evaluations help answer two key questions:
        1. Which segment does the organization most want to target?
        2. How likely is it that the organization will be chosen by the segment over competitors?
3. **Criteria and Weighting**:
    - Segments are evaluated based on multiple criteria, which are assigned specific weights. For example, the attractiveness of the segment could depend on factors like size, growth potential, and alignment with the organization's strengths.
    - Each segment is given ratings for each criterion, and these ratings are multiplied by the weight assigned to each criterion. The weighted scores are summed to determine the overall attractiveness of each segment.
4. **Plotting the Segments**:
    - A **segment evaluation plot** is used to visually represent the attractiveness of each segment (on the x-axis) and the relative competitiveness of the organization (on the y-axis). The size of each bubble on the plot reflects another important criterion, such as profit potential.
    - This plot helps the team make an informed decision by comparing segments on both dimensions.
5. **Example Calculation**:

    - The example uses a table (Table 10.1) to show how segment attractiveness and organizational competitiveness are calculated using weights and ratings for various criteria. These values are plotted on the evaluation plot to provide a clear visual understanding of the relative positioning of each segment.

Key Insights for Decision-Making:

- Segments that are both attractive to the organization and view the organization as attractive (high ratings on both axes) are ideal targets.
- Segments with high profit potential but low organizational competitiveness or vice versa may require further consideration or adjustments to the organization's offering.

This process allows a company to make strategic, data-driven decisions about which segments to prioritize and invest in for long-term market success.

Step 9: Customising the Marketing Mix

This step focuses on customizing the marketing mix based on market segmentation. Let's break it down:

11.1 Implications for Marketing Mix Decisions

- **Historical perspective**: Marketing was once seen as a set of tools to drive product sales, and the 4Ps—Product, Price, Promotion, and Place—became the foundational elements of the marketing mix.
- **Segmentation-Targeting-Positioning (STP)**: Market segmentation is the first step in the STP process, followed by targeting and positioning. This ensures that segmentation is aligned with other strategic marketing decisions like competition and positioning.

11.2 Product

- Customizing the product involves understanding customer needs, often leading to the modification of existing products rather than creating new ones.
- Examples include naming, packaging, warranty offerings, and after-sales support.
- For instance, targeting a segment with a rich cultural heritage might involve creating a product like a museum and monument package that appeals to tourists interested in these activities.

11.3 Price

- Decisions here include setting the product price and offering discounts.
- For segment 3, the analysis shows they tend to spend more on vacations. This insight suggests that premium pricing could be applied to products aimed at this segment.
- The example uses data analysis to show how segment 3's higher expenditure might justify a higher product price.

11.4 Place

- The place dimension addresses how to distribute the product (e.g., online vs. offline, direct sales vs. retail).
- For the target segment, understanding how they book their accommodation can inform decisions on where to offer the product, ensuring it is available through their preferred channels.

Overall, customizing the marketing mix involves tailoring product offerings, pricing, distribution channels, and promotional strategies to meet the specific needs and preferences of the targeted market segment.

CODE
Step4

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA

# Assuming the 'mcdonalds' DataFrame is already loaded
# You should load your DataFrame here
# mcdonalds = pd.read_csv('path_to_file.csv')

# Transform relevant columns to binary matrix (0 for No, 1 for Yes)
MD_x = mcdonalds.iloc[:, :11].apply(lambda x: (x == 'Yes')).astype(int)

# Calculate column-wise means
column_means = MD_x.mean().round(2)
print("Column Means:")
print(column_means)

# Perform PCA
pca = PCA()
MD_pca = pca.fit_transform(MD_x)

# Print standard deviations (singular values)
print("\nStandard deviations (PCs):", pca.singular_values_.round(1))

# Rotation (principal components)
rotation_matrix = pd.DataFrame(pca.components_, columns=MD_x.columns)
print("\nRotation (components matrix):")
print(rotation_matrix.round(2))

# Proportion of Variance and Cumulative Proportion
print("\nProportion of Variance:")
print(np.round(pca.explained_variance_ratio_, 4))  # Proportion of variance for each component

print("\nCumulative Proportion:")
cumulative_proportion = np.cumsum(pca.explained_variance_ratio_)
print(np.round(cumulative_proportion, 4))  # Cumulative proportion of variance

# Plotting the PCA results (grey scatter plot)
plt.scatter(MD_pca[:, 0], MD_pca[:, 1], color='grey')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('PCA Plot')
```

```
plt.show()

# Projecting the axes (principal components directions)
pca_axes = pca.components_

# Plot the axes
for i in range(len(pca_axes)):
    plt.quiver(0, 0, pca_axes[i, 0], pca_axes[i, 1], angles='xy', scale_units='xy', scale=1,
color='red', label=f'PC{i+1}')

plt.xlim(-1, 1)
plt.ylim(-1, 1)
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('PCA Axes')
plt.legend()
plt.grid(True)
plt.show()
```

Step 5

```
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.mixture import GaussianMixture
import matplotlib.pyplot as plt

# Assuming mcdonalds DataFrame is already loaded
# Step 5: KMeans and clustering

# Step 5.1: Set the seed (equivalent to set.seed in R)
np.random.seed(1234)

# Step 5.2: Convert data to binary (Yes = 1, No = 0) for clustering
MD_x = (mcdonalds.iloc[:, 0:11] == "Yes").astype(int)

# Step 5.3: Perform KMeans clustering for different cluster sizes (2 to 8 clusters)
kmeans_models = []
for k in range(2, 9):
    kmeans = KMeans(n_clusters=k, random_state=1234, n_init=10)
    kmeans.fit(MD_x)
    kmeans_models.append(kmeans)

# Step 5.4: Relabel clusters (reassign cluster labels)
# We'll use the same method here for relabeling clusters
def relabel(kmeans_model):
    labels = kmeans_model.labels_
    cluster_centers = kmeans_model.cluster_centers_
    return labels, cluster_centers

# Apply relabeling for each model
kmeans_labels = [relabel(model)[0] for model in kmeans_models]

# Step 5.5: Plot the clustering results (using elbow method or other evaluation metrics)
# Elbow method to find the best number of clusters
inertia = [model.inertia_ for model in kmeans_models]
plt.plot(range(2, 9), inertia)
plt.xlabel("Number of clusters")
plt.ylabel("Inertia")
plt.title("Elbow Method for Optimal Clusters")
plt.show()
```

```
# Step 5.6: Bootstrapping clustering with Gaussian Mixture Model (GMM)
gmm_models = []
for k in range(2, 9):
    gmm = GaussianMixture(n_components=k, random_state=1234)
    gmm.fit(MD_x)
    gmm_models.append(gmm)

# Step 5.7: Evaluate clustering stability using adjusted Rand index (ARI)
from sklearn.metrics import adjusted_rand_score
ari_scores = []
for i, gmm in enumerate(gmm_models):
    ari = adjusted_rand_score(kmeans_labels[i], gmm.predict(MD_x))
    ari_scores.append(ari)

# Plot ARI scores
plt.plot(range(2, 9), ari_scores)
plt.xlabel("Number of clusters")
plt.ylabel("Adjusted Rand Index (ARI)")
plt.title("Cluster Stability with ARI")
plt.show()

# Step 5.8: Visualize cluster stability (like SLSW in R)
# Stability plot (example: variance of labels across multiple iterations)
# Here, we are using the kmeans_labels as a simple example
stability_scores = [np.var(kmeans_labels[k]) for k in range(len(kmeans_labels))]
plt.plot(range(2, 9), stability_scores)
plt.xlabel("Number of clusters")
plt.ylabel("Cluster Stability (Variance)")
plt.title("Cluster Stability")
plt.show()

# Step 5.9: Mixture Model using FlexMix equivalent in Python
# Step 5.9.1: Fit a Gaussian Mixture Model using sklearn's GMM
gmm_4 = GaussianMixture(n_components=4, random_state=1234)
gmm_4.fit(MD_x)
gmm_4_labels = gmm_4.predict(MD_x)

# Step 5.9.2: Compare KMeans and Mixture Model clustering
from sklearn.metrics import confusion_matrix
confusion = confusion_matrix(kmeans_labels[2], gmm_4_labels)  # Example for 4 clusters
print(confusion)

# Step 5.10: Final plotting of cluster comparison
plt.scatter(MD_x.iloc[:, 0], MD_x.iloc[:, 1], c=gmm_4_labels, cmap='viridis')
```

```
plt.title("Clustered with GMM (4 clusters)")
plt.show()
```

STEP6

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from scipy.cluster.hierarchy import linkage, dendrogram
from sklearn.cluster import KMeans

# Assuming MD.x is your data matrix (rows as samples, columns as features)
# Example: MD.x is a numpy array of shape (n_samples, n_features)

# Step 1: Data Standardization (important for PCA and clustering)
scaler = StandardScaler()
MD_x_scaled = scaler.fit_transform(MD.x)

# Step 2: Hierarchical Clustering (using linkage and scipy)
# Compute the linkage matrix using the 'ward' method
linkage_matrix = linkage(MD_x_scaled.T, method='ward')

# Plot the dendrogram
plt.figure(figsize=(10, 6))
dendrogram(linkage_matrix, labels=['yummy', 'convenient', 'fast', 'disgusting', 'cheap', 'spicy',
'fattening', 'greasy', 'tasty', 'expensive', 'healthy'])
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Feature Index')
plt.ylabel('Distance')
plt.show()

# Step 3: KMeans Clustering to assign clusters (using 4 clusters in this case)
kmeans = KMeans(n_clusters=4, random_state=1234)
MD_k4 = kmeans.fit_predict(MD_x_scaled)

# Step 4: Perform PCA for 2D visualization
pca = PCA(n_components=2)
principal_components = pca.fit_transform(MD_x_scaled)

# Plot the PCA projection with different colors based on k-means clusters
plt.figure(figsize=(10, 6))
plt.scatter(principal_components[:, 0], principal_components[:, 1], c=MD_k4, cmap='viridis',
alpha=0.6)
plt.title('PCA Projection of Data with KMeans Clusters')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.colorbar(label='Cluster')
```

```
plt.show()

# Step 5: Projecting data with PCA and visualize using clusters
# Plot the PCA components
plt.figure(figsize=(10, 6))
plt.scatter(principal_components[:, 0], principal_components[:, 1], c=MD_k4, cmap='plasma',
alpha=0.6)
plt.title('PCA Projection of Clusters')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.colorbar(label='Cluster')
plt.show()

# Optional: If you need to plot the clusters with hierarchical clustering order
# Reorder MD.k4 according to hierarchical clustering
reordered_indices = np.argsort(linkage_matrix[:, 0])  # Sort based on the linkage matrix
reordered_MD_k4 = MD_k4[reordered_indices]

# Plot the reordered clusters
plt.figure(figsize=(10, 6))
plt.scatter(principal_components[:, 0], principal_components[:, 1], c=reordered_MD_k4,
cmap='plasma', alpha=0.6)
plt.title('Reordered PCA Projection of Clusters')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.colorbar(label='Cluster')
plt.show()
```

STEP7

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.preprocessing import LabelEncoder
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.metrics import adjusted_rand_score
import scipy.cluster.hierarchy as sch

# Assuming 'mcdonalds' is your DataFrame
# Let's assume k4 is the clustering result (replace with actual clustering result)
k4 = np.random.randint(1, 5, size=len(mcdonalds))  # Random segment numbers for illustration

# Create a 'Target' column for decision tree classification (whether k4 == 3 or not)
mcdonalds['Target'] = (k4 == 3).astype(int)

# Label encode categorical columns if needed (e.g., 'Gender')
le = LabelEncoder()
mcdonalds['Gender_encoded'] = le.fit_transform(mcdonalds['Gender'])

# 1. Mosaic Plot - For categorical variables (e.g., Like vs. segment number)
plt.figure(figsize=(8, 6))
pd.crosstab(k4, mcdonalds['Like']).plot(kind='bar', stacked=True,
color=sns.color_palette("Set3"))
plt.title('Mosaic Plot: Like vs. Segment')
plt.ylabel('Count')
plt.xlabel('Segment')
plt.show()

# 2. Mosaic Plot - Gender vs. Segment Number
plt.figure(figsize=(8, 6))
pd.crosstab(k4, mcdonalds['Gender']).plot(kind='bar', stacked=True,
color=sns.color_palette("Set2"))
plt.title('Mosaic Plot: Gender vs. Segment')
plt.ylabel('Count')
plt.xlabel('Segment')
plt.show()

# 3. Decision Tree for k4 == 3 (target variable)
X = mcdonalds[['Like.n', 'Age', 'VisitFrequency', 'Gender_encoded']]  # Feature columns
```

```
y = mcdonalds['Target']  # Target variable (k4 == 3)

# Fit decision tree classifier
tree_model = DecisionTreeClassifier(random_state=1234)
tree_model.fit(X, y)

# Plot the decision tree
plt.figure(figsize=(12, 8))
plot_tree(tree_model, feature_names=['Like.n', 'Age', 'VisitFrequency', 'Gender_encoded'],
          class_names=['Not 3', '3'], filled=True, rounded=True)
plt.title('Decision Tree for k4 == 3')
plt.show()

# 4. Hierarchical Clustering Dendrogram
# Perform hierarchical clustering (using the features for clustering)
pca = PCA(n_components=2)
X_pca = pca.fit_transform(mcdonalds[['Like.n', 'Age', 'VisitFrequency', 'Gender_encoded']])

# Compute the linkage matrix
Z = sch.linkage(X_pca, method='ward')

# Plot the dendrogram
plt.figure(figsize=(10, 7))
sch.dendrogram(Z)
plt.title('Hierarchical Clustering Dendrogram')
plt.xlabel('Sample index')
plt.ylabel('Distance')
plt.show()
```

STEP8

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Assuming you have your data as a pandas DataFrame 'mcdonalds' with relevant columns

# Assuming 'k4' is the cluster assignment for each observation (from k-means or similar
clustering method)
# Replace 'k4' with your actual cluster assignments array

# Calculate the mean values for VisitFrequency, Like.n, and Gender for each cluster
visit = mcdonalds.groupby(k4)['VisitFrequency'].mean()
like = mcdonalds.groupby(k4)['Like.n'].mean()
female = mcdonalds.groupby(k4).apply(lambda x: np.mean(x['Gender'] == 'Female'))

# Plotting
plt.scatter(visit, like, s=10 * female, c='blue', alpha=0.6)  # s is the size of the points
for i in range(len(visit)):
    plt.text(visit[i], like[i], str(i+1), fontsize=12, ha='right')  # Label clusters

plt.xlim(2, 4.5)
plt.ylim(-3, 3)
plt.xlabel('Visit Frequency')
plt.ylabel('Like.n')
plt.title('Clusters with Visit Frequency, Like.n, and Female Proportion')
plt.show()
```