# Market Segmentation using Machine Learning:

Kanishka Arora

June 25, 2024

Indian Institute of Space Science and Technology

# Contents


**1 Market Segmentation** **1**
- 1.1 What is Marketing? . . . 1
- 1.2 What is Market Segmentation? . . . 3
- 1.3 What are the benefits of Market Segmentation? . . . 3
- 1.4 What are the costs of Execution of Market Segmentation? . . . 4

**2 Market Segmentation Analysis** **5**
- 2.1 Layers of Market Segmentation Analysis . . . 5
- 2.2 Approaches in Market Segmentation Analysis . . . 6
- 2.3 More on Data Driven Market Segmentation Approaches . . . 7

**3 Steps in Market Segmentation** **9**
- 3.1 Step 1 - Deciding Not to Segment . . . 9
- 3.2 Step 2 - Identifying the Ideal Target Segment . . . 11
  - 3.2.1 Segment Evaluation Criteria . . . 11
  - 3.2.2 Knock out Criteria . . . 11
  - 3.2.3 Attractiveness Criteria . . . 11
  - 3.2.4 Implementing a Structured Process . . . 11
- 3.3 Step 3 - Data Collection . . . 13
  - 3.3.1 Segmentation Criteria . . . 13
  - 3.3.2 Geographic Segmentation . . . 14
  - 3.3.3 Socio-Demographic Segmentation . . . 14
  - 3.3.4 Psychographic Segmentation . . . 15
  - 3.3.5 Behavioral Segmentation . . . 15
  - 3.3.6 Data from Survey Studies . . . 16
  - 3.3.7 Choice of Variables . . . 16
  - 3.3.8 Response Options . . . 17
  - 3.3.9 Response Styles . . . 17
  - 3.3.10 Sample Size . . . 18
  - 3.3.11 Data from Internal sources & Experimental studies . . . 18
- 3.4 Step 4 - Data Exploration . . . 20
  - 3.4.1 Loading and having a glance at the data . . . 20
  - 3.4.2 Data Cleaning . . . 20
  - 3.4.3 Descriptive Analysis . . . 21
  - 3.4.4 Pre-Processing . . . 25
  - 3.4.5 Principal Components Analysis . . . 26
- 3.5 Step 5 - Selecting the Targeting Segments . . . 27
  - 3.5.1 Target Decision . . . 27
  - 3.5.2 Market Segment Evaluation . . . 27
- 3.6 Step 6 - Customising the Marketing Mix . . . 29
  - 3.6.1 Product . . . 31

3.6.2 Price . . . 31
3.6.3 Place . . . 32
3.6.4 Promotion . . . 32

**A Code** **A.1**
A.1 Exploring the Data . . . A.1
A.2 Extracting Segments . . . A.3
A.2.1 Using K-Means . . . A.3
A.2.2 Using mixtures of distributions . . . A.10
A.2.3 Using Regression Models . . . A.13
A.3 Profiling Segments . . . A.17
A.4 Describing Segments . . . A.21
A.5 Github Repository . . . A.29

**Bibliography** **A.1**

# List of Tables

3.1 Data underlying the segment evaluation plot(Taken from the textbook) . 29

# List of Figures

1.1 SWOT Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

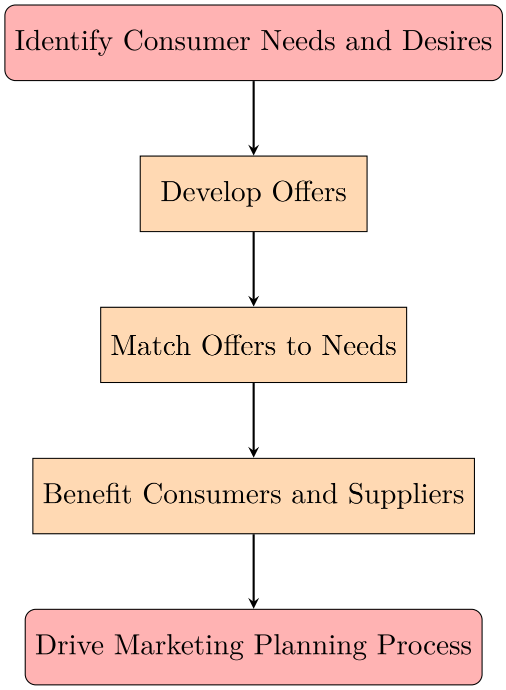
3.1 Histogram from *Market Segmentation Analysis* textbook . . . . . . . . . . 22
3.2 Box-and-Whisker plot of tourist age from *Market Segmentation Analysis* textbook . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23
3.3 Dot Chart of Yes percentages *Market Segmentation Analysis* textbook . . 24
3.4 Segment Evaluation Plot(Taken from the Textbook) . . . . . . . . . . . . 30

A.1 PCA of McDonald's Attributes . . . . . . . . . . . . . . . . . . . . . . . . A.3
A.2 Scree plot for K-Means clustering.png . . . . . . . . . . . . . . . . . . . . A.4
A.3 Global Stability Boxplot . . . . . . . . . . . . . . . . . . . . . . . . . . . A.6
A.4 Gorge plot of the four-segment k-means solution for the fast food dataset A.7
A.5 Segment Level Stability Across Solutions (SLSA) Plot . . . . . . . . . . . A.8
A.6 Segment Level Stability within solutions . . . . . . . . . . . . . . . . . . A.10
A.7 Information Criteria (AIC, BIC, ICL) . . . . . . . . . . . . . . . . . . . . A.12
A.8 Output . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . A.13
A.9 Output . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . A.14
A.10 Output . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . A.15
A.11 Regression coefficients for each component . . . . . . . . . . . . . . . . A.17
A.12 Segment Profile Plot . . . . . . . . . . . . . . . . . . . . . . . . . . . . . A.19
A.13 Segment Separation Plot . . . . . . . . . . . . . . . . . . . . . . . . . . . A.21
A.14 Shaded Mosaic Plot for Segment Membership and Like/Hate McDonald's A.23
A.15 Mosaic Plot for Gender Distribution across Segments . . . . . . . . . . . A.25
A.16 Box and Whisker Plot for Age . . . . . . . . . . . . . . . . . . . . . . . A.26
A.17 Decision Tree for segment 3 membership . . . . . . . . . . . . . . . . . . A.28

Chapter 1

# Market Segmentation

## 1.1 What is Marketing?




The above is a basic flowchart that explains the meaning of Marketing. It is broadly divided into two categories, which are,

1. **Strategic Marketing**
   It focuses largely on the long-term goal and direction of the organization, without paying much attention to the roadmap of short-term marketing, which leads to long-term goals. It is typically associated with consumer needs and desires, strengths and weaknesses within the organization, external threats, and opportunities for the organization.
   Once the above-listed factors have been studied and accounted for, we make two key decisions in the strategic marketing process i.e. which consumers to focus on (segmentation and targeting), and which image of the organization to create in the market (positioning).


**Fig. 1.1:** SWOT Analysis

2. **Tactical Marketing**
   The tactical marketing plan is invested in the short-term goals and actions required to bring forth the long-term desired results of the organization. Only when segmentation targeting and positioning have been done can we begin the tactical marketing process.
   Tactical marketing broadly covers four areas, namely

   - Product
   - Price
   - Place
   - Promotion

   In an organization, the combination of both good strategic marketing and good tactical marketing leads to the best possible outcome. But one must remember,

   So the foundation of growth for the organization's success lies in proper strategic marketing planning.

## 1.2 What is Market Segmentation?

Market segmentation is essential for marketing success, as it allows companies to target specific groups within a larger market. Smith first proposed this strategy in 1956, who defined it as dividing a heterogeneous market into smaller, homogeneous markets. Effective market segmentation means that consumers within a segment share similar characteristics important to management, while those in different segments are distinct in those characteristics.

Segmentation criteria can range from single attributes like age or gender to multiple factors like benefits sought or values held. This tailored approach helps organizations better meet the needs of each segment, leading to higher sales and stronger market positioning.

A **concentrated market strategy** focuses on one segment, making it ideal for resource-limited organizations facing competition. However, it carries the risk of relying on a single segment. A **differentiated market strategy** targets multiple segments with customized marketing efforts, suitable for mature markets. An undifferentiated market strategy uses the same product and marketing mix for the entire market, which can work for resource-rich organizations or new products.

## 1.3 What are the benefits of Market Segmentation?

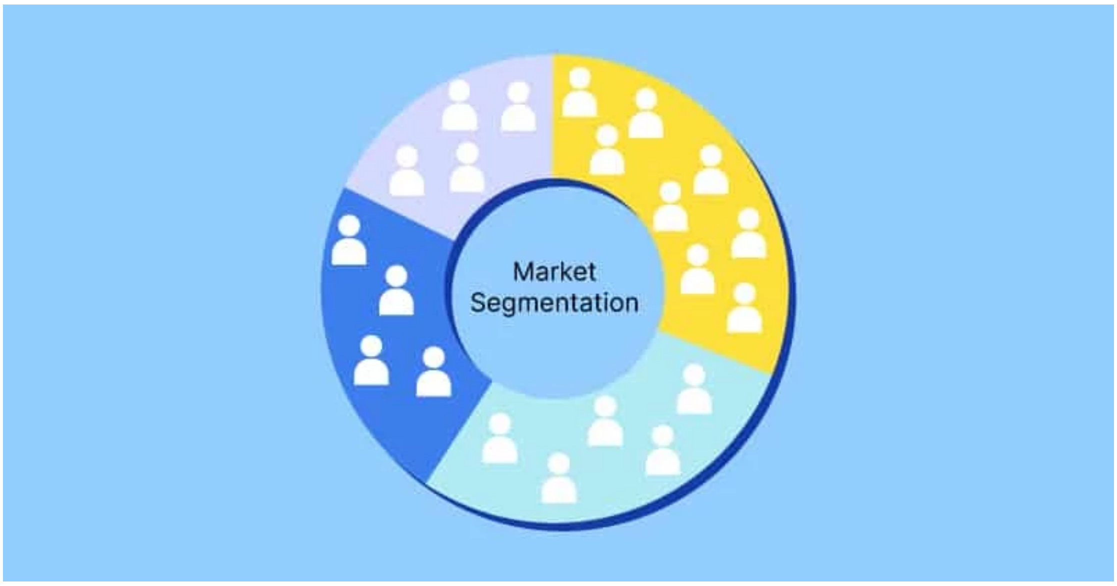
These are the following benefits of Market Segmentation,

- Market segmentation encourages organizations to assess their current position and future goals.
- It helps organizations identify their strengths relative to competitors.
- Forces reflection on consumer preferences and needs.
- Leads to critical insights and perspectives.
- Enhances understanding of consumer differences.
- Improves alignment between organizational strengths and consumer needs.
- Forms the basis for long-term competitive advantage.
- Facilitates market dominance in niche segments.
- Enables customization of products/services for specific consumer groups.
- Supports micro marketing and hyper-segmentation strategies.
- Enhances ROI by focusing marketing efforts more effectively.
- Essential for small organizations focusing on distinct consumer needs.
- Effective in targeting sales efforts towards specific consumer groups.
- Contributes to team building within organizations.
- Improves communication and information sharing across organizational units.

## 1.4 What are the costs of Execution of Market Segmentation?

Implementing market segmentation requires significant investment in terms of time, human resources, and finances. It involves thorough analysis, custom marketing strategies, and ongoing monitoring. Success promises a competitive advantage, but failure can lead to wasted resources and demoralization among the staff involved.

Hence the organization has to make a wise decision to move with the market segmentation analysis and strategy.



Chapter 2

# Market Segmentation Analysis

The process of grouping consumers into naturally existing or artificially created segments of consumers who share similar product preferences or characteristics.

## 2.1 Layers of Market Segmentation Analysis

Market segmentation involves three critical layers,

- **Technical Process**
  Led by data analysts, involves data collection, exploration, and statistical segment extraction to group consumers effectively. The grouping of consumers can always only be as good as the data provided by the segment extraction method.

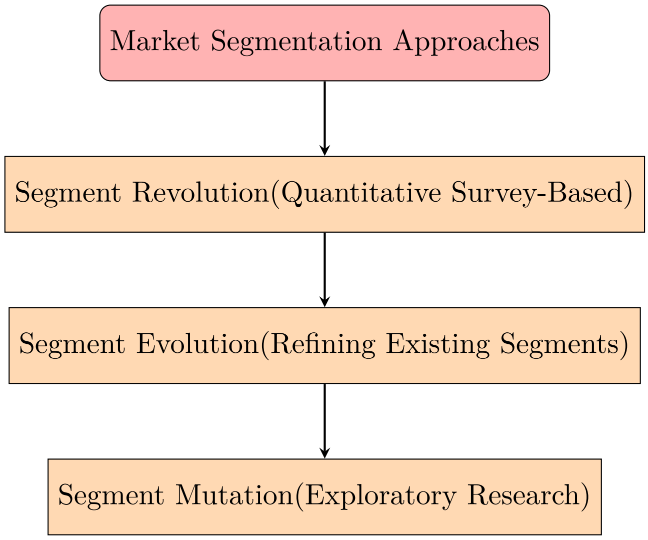
- **Analytical Tasks**
  Includes profiling and describing each segment to guide strategic marketing decisions and customize the marketing mix.

- **Implementation and Strategic Decision-making**
  Requires organizational commitment to long-term segmentation strategies based on market opportunities identified. User involvement is crucial throughout, from data collection to selecting target segments and developing tailored marketing plans.

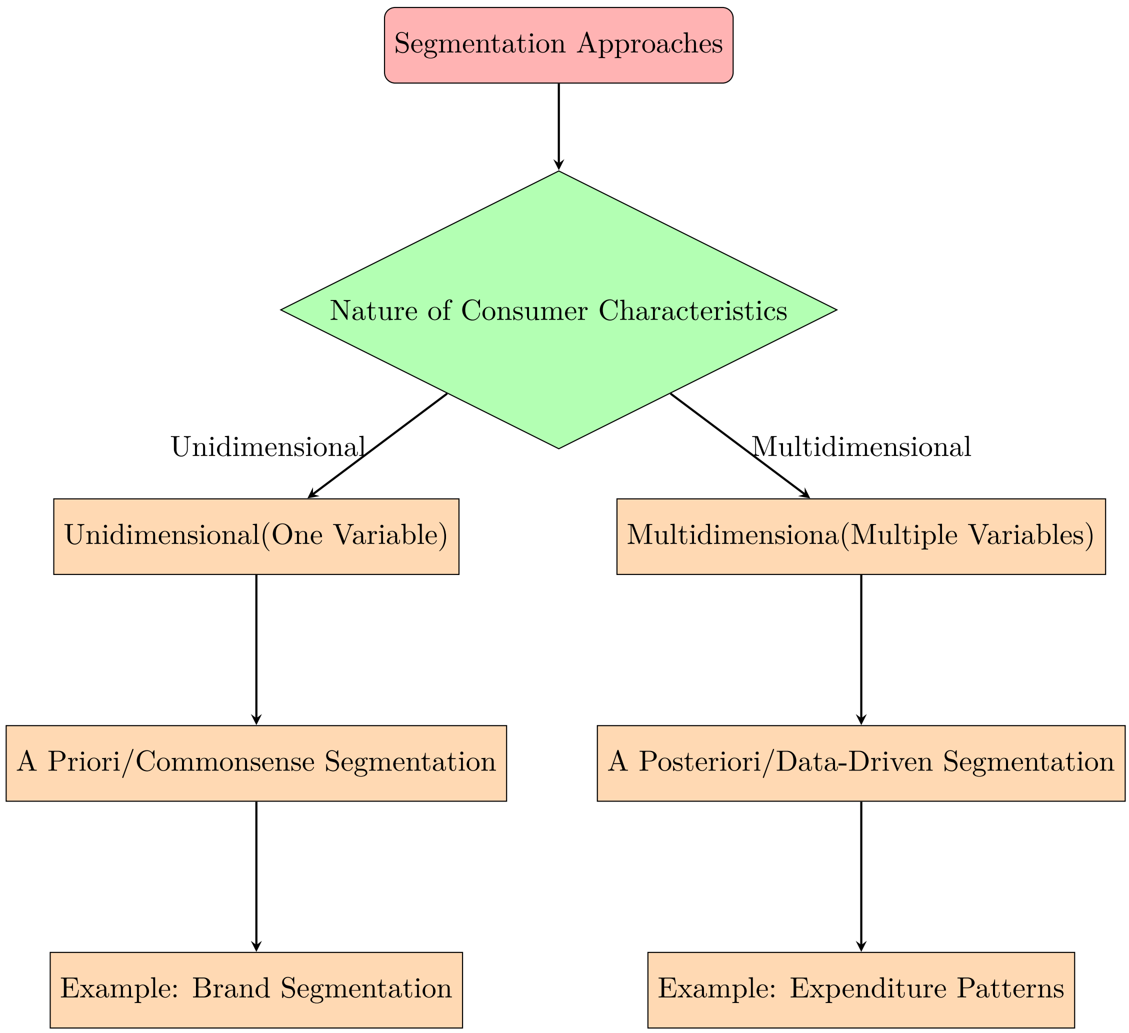
## 2.2 Approaches in Market Segmentation Analysis

Here are two approaches to Market Segmentation Analysis,

- **Based on Organizational Constraints**



- **Based on choice of Segmentation Variables**




## 2.3 More on Data Driven Market Segmentation Approaches

Over here data analysis created the solution for the market segmentation.

- **Rarity of Natural Segments:** Naturally occurring, distinct, and well-separated market segments are rare in real consumer data.
- **Conceptual Approaches:** There are three approaches to data-driven market segmentation: natural, reproducible, and constructive segmentation.

- **Natural Segmentation:** Assumes pre-existing segments in the data, aiming to uncover these distinct market segments.
- **Reproducible Segmentation:** Acknowledges some structure in the data, allowing for consistent and reliable segmentation solutions across different analyses.
- **Constructive Segmentation:** Involves creating artificial segments in the absence of natural data structure, which can still provide strategic marketing value.
- **Data Structure Analysis:** Essential to analyze data structure before segmentation to avoid methodological errors and misinterpretations. This can involve repeated segmentation with different algorithms.
- **Practical Implications:** Most data sets contain some exploitable structure, even if not in clear cluster form, making collaborative constructive segmentation useful for developing effective marketing strategies.

In the proceeding chapters we will discuss the step by step analysis of market segmentation.

Chapter 3

# Steps in Market Segmentation

In this chapter we will discuss the steps in Market Segmentation one by one.

## 3.1 Step 1 - Deciding Not to Segment

Market segmentation, a fundamental strategy in marketing, demands careful consideration due to its profound implications for organizations. It requires a steadfast, long-term commitment akin to a marriage, involving substantial investments in research, surveys, product development, and tailored marketing communications. Before embarking on segmentation, organizations must weigh these costs against potential sales increases to ensure profitability.

Implementing segmentation often necessitates organizational adjustments, such as restructuring around market segments rather than products, and potentially developing new products or adjusting pricing and distribution strategies. Strategic business units focused on segments can facilitate ongoing adaptation to evolving market needs.

Below are the following key implementation barriers,

- **Senior Management Involvement**
  Lack of leadership, proactive championing, commitment, and involvement in the segmentation process by senior executives.
- **Resource Allocation**
  Insufficient allocation of resources (financial and human) for conducting the segmentation analysis and implementing the strategy.
- **Organizational Culture**
  Lack of market or consumer orientation, resistance to change, inadequate communication and sharing of information across departments, short-term thinking, and office politics.
- **Training and Expertise**
  Lack of understanding and expertise in market segmentation among senior management and the segmentation team.
- **Formal Marketing Function**
  Absence of a formal marketing function or qualified marketing experts within the organization.

- **Data Management**
  Lack of qualified data managers and analysts to handle segmentation data effectively.
- **Objective Restrictions**
  Financial constraints or organizational structures that hinder the execution of segmentation strategies.
- **Process-related Issues**
  Unclear objectives, inadequate planning, lack of structured processes, unclear allocation of responsibilities, and time constraints during the segmentation process.
- **Acceptance of Management Techniques**
  Resistance to using management techniques (like segmentation analysis) that senior management does not fully understand.

Ultimately, the decision to pursue market segmentation should be made at the highest executive level and consistently communicated throughout the organization to foster sustained commitment and alignment across all operational units.

## 3.2 Step 2 - Identifying the Ideal Target Segment

### 3.2.1 Segment Evaluation Criteria

In this step we focus on defining segment evaluation criteria, comprising essential knock-out criteria and flexible attractiveness criteria. While knock-out criteria are non-negotiable and automatically disqualify segments lacking essential features, attractiveness criteria are selectively applied by the segmentation team to gauge the relative desirability of compliant segments. This structured approach, supported by a diverse array of proposed criteria detailed in the literature, facilitates a thorough evaluation process essential for identifying and prioritizing market segments effectively.

### 3.2.2 Knock out Criteria

Knock-out criteria are used to determine if market segments resulting from the market segmentation analysis qualify to be assessed using segment attractiveness criteria, which are the following,

- The segment must be homogeneous.
- The segment must be distinct.
- The segment must be large enough.
- The segment must match the strengths of the organization.
- Members of the segment must be identifiable.
- The segment must be reachable.

### 3.2.3 Attractiveness Criteria

Segments also require attractiveness criteria pertaining to any given situation, and it is not binary in nature. Attractiveness criteria are not assessed as either complying or non-complying, rather it is rated in a spectrum.

### 3.2.4 Implementing a Structured Process

- Structured processes are widely acknowledged as beneficial for assessing market segments.
- The most popular method is the segment evaluation plot, which plots segment attractiveness against organizational competitiveness.
- Segment attractiveness and organizational competitiveness values are determined by the segmentation team, with no universal criteria applicable to all organizations.

- Criteria for both segment attractiveness and organizational competitiveness must be negotiated and agreed upon, with a recommendation to use no more than six factors.
- A team approach is optimal, where a core team proposes initial solutions and an advisory committee, representing all organizational units, discusses and possibly modifies these proposals.
- Including representatives from all organizational units is crucial as each unit has different business perspectives and all units are stakeholders in the segmentation strategy.
- Segment attractiveness criteria should be selected early to ensure relevant data is captured during collection and to simplify target segment selection later.
- The segmentation team should finalize approximately six segment attractiveness criteria, each with a weighted importance relative to the others.
- Team members should distribute 100 points across the criteria to determine weights, negotiating until agreement is reached, with approval ideally sought from the advisory committee for a comprehensive perspective.

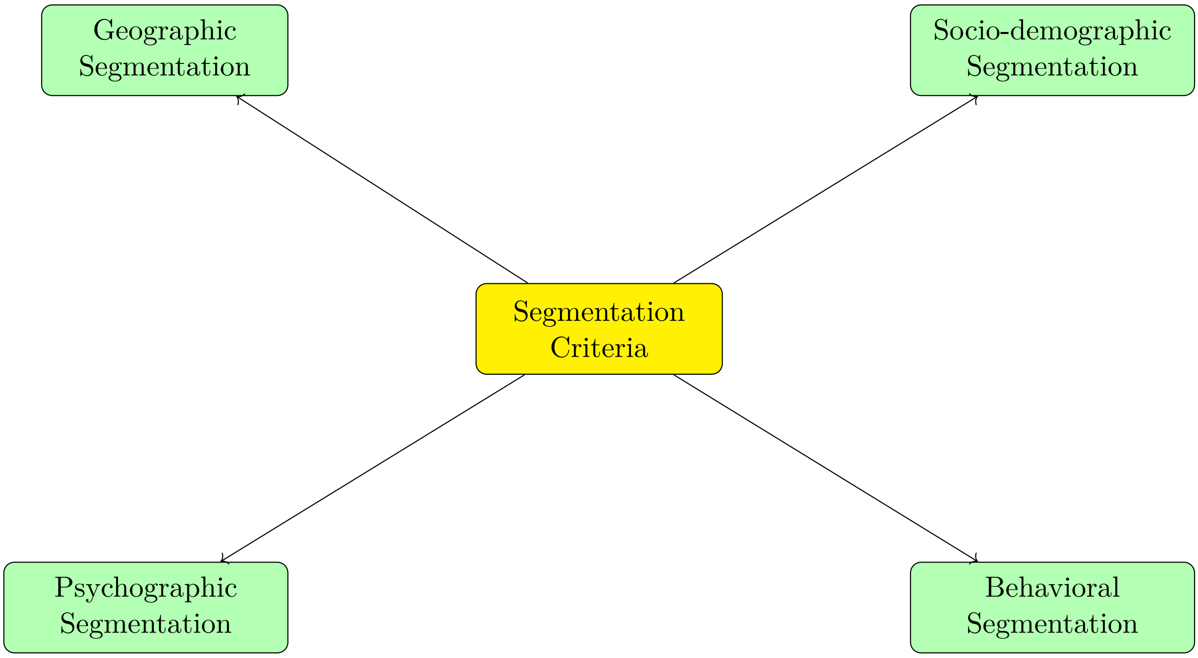
## 3.3 Step 3 - Data Collection

In this section we will discuss the collection of data for Segmentation.

- Empirical data is the foundation for both commonsense and data-driven market segmentation, used to identify, create, and describe market segments.
- Commonsense segmentation typically uses one single characteristic, such as gender, as the segmentation variable, with other characteristics serving as descriptor variables.
- Descriptor variables, such as socio-demographics and media behavior, are used to describe segments in detail, aiding in the development of effective marketing strategies.
- Data-driven market segmentation uses multiple segmentation variables to identify or create market segments that are useful to the organization.
- The quality of empirical data is critical for assigning individuals to the correct market segments and for accurately describing these segments.
- Empirical data for segmentation can come from surveys, observations (such as scanner data), and experimental studies, with data reflecting actual consumer behavior being preferable.
- Survey data, while common, can be unreliable for behaviors that are socially desirable, so exploring a range of data sources is recommended.

### 3.3.1 Segmentation Criteria

The term segmentation criterion relates to the nature of the information used for market segmentation.The most common segmentation criteria include geographic, socio-demographic, psychographic and bhevioural.




Below we will discuss the criteria individually,

### 3.3.2 Geographic Segmentation

- Geographic segmentation uses the consumer's location of residence as the primary criterion for forming market segments, making it straightforward for targeting communication messages and channels.
- This method is particularly useful for companies needing to cater to language differences and regional preferences.
- The key advantage is the ease of assigning each consumer to a geographic unit, allowing precise targeting through local media.
- The main disadvantage is that geographic location alone often fails to capture other relevant consumer characteristics, such as the benefits sought in products or services.
- Geographic segmentation has seen a revival in international market studies, although it faces challenges like ensuring meaningful segmentation variables across regions and avoiding cultural bias in survey responses.

### 3.3.3 Socio-Demographic Segmentation

- Typical socio-demographic segmentation criteria include age, gender, income, and education.

- Socio-demographic segments are particularly useful in industries like luxury goods (high income), cosmetics (gender-specific marketing), and baby products (gender-specific needs).
- In industries such as retirement villages and tourism resorts, age and family status (having children) play crucial roles in segmenting consumers.
- Like geographic segmentation, socio-demographic criteria allow easy determination of segment membership for each consumer.
- Socio-demographic factors can explain specific product preferences in some cases (e.g., family vacation choices), but often they do not fully account for consumer behavior variations.
- Studies suggest that demographics explain a small percentage (about 5%) of consumer behavior variance, with values, tastes, and preferences being more influential in buying decisions.

### 3.3.4 Psychographic Segmentation

- Psychographic segmentation categorizes people based on psychological criteria such as beliefs, interests, preferences, aspirations, and benefits sought in products.
- Benefit segmentation, focuses on identifying consumer benefits as a key psychographic approach.
- Lifestyle segmentation, another popular method, segments based on activities, opinions, and interests.
- Psychographic criteria are complex compared to geographic or socio-demographic criteria, often requiring multiple variables like travel motives or perceived risks in segmentation studies.
- Psychographic segmentation provides deeper insights into consumer behavior by focusing on underlying motivations. For example, tourists motivated by cultural exploration are likely to choose destinations rich in cultural experiences.

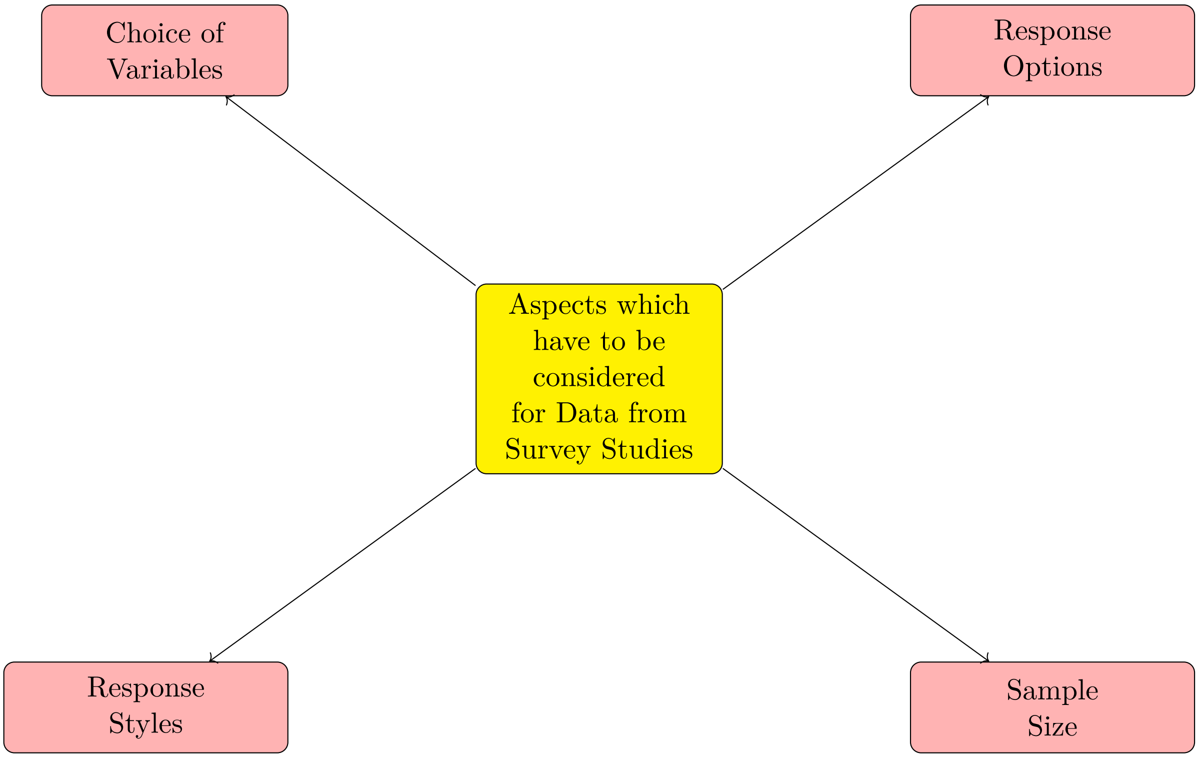
### 3.3.5 Behavioral Segmentation

- Behavioral segmentation extracts segments based on similarities in actual or reported behaviors, such as prior product experience, purchase frequency, amount spent per purchase occasion, and information search habits.
- Research indicates that behaviors reported by tourists are often more effective segmentation criteria than geographic variables.
- The primary advantage of behavioral approaches is their basis in actual behavior, rather than stated or intended behavior, which allows for segmentation by the most relevant consumer similarities.

- Examples include using actual consumer expenses and purchase data across product categories as segmentation variables and longitudinal brand choice behavior.
- Using behavioral data eliminates the need for developing valid measures of psychological constructs but accessing such data can be challenging, particularly when including potential customers who haven't yet purchased the product.

### 3.3.6 Data from Survey Studies

Most of the analyses in market segmentation are based on survey data as it is cheap and easy to collect but can contain a significant number of biases. A few things need to be discussed before we use the survey data for analysis,




### 3.3.7 Choice of Variables

The below points summarize information about the choice of variables for analysis of survey data.

- Careful selection of segmentation variables is crucial in both commonsense and data-driven segmentation to ensure the quality of market segmentation solutions.

- In data-driven segmentation, relevant variables capturing the segmentation criterion should be included, while unnecessary variables must be avoided to prevent respondent fatigue and maintain response quality.
- Unnecessary variables increase the complexity of segmentation problems without adding relevant information, complicating the extraction of optimal market segments for analytical techniques.
- Noisy or masking variables, which do not contribute relevant information, can hinder algorithms from identifying correct segmentation solutions.
- Developing a high-quality questionnaire involves both exploratory qualitative research to understand consumer beliefs and quantitative survey research to ensure all critical variables are included without redundancy.

### 3.3.8 Response Options

The below points summarize information about the Response Options of data.

- Survey response options determine the data scale for segmentation analysis, crucial for distance-based techniques.
- Binary responses (e.g., yes/no) are ideal for segmentation, represented as 0s and 1s with clear distances between options.
- Nominal variables (e.g., occupation choices) can be converted to binary for segmentation, simplifying analysis.
- Metric data (e.g., age, nights stayed) allow for precise measurement and are well-suited for segmentation.
- Ordinal data (e.g., Likert scales) lack defined distances between responses, complicating distance-based segmentation methods.
- Visual analogue scales, like slider scales, provide metric data suitable for nuanced responses in online surveys.

### 3.3.9 Response Styles

The below points summarize information about the Response Styles of data.

- Response bias in surveys occurs when respondents consistently answer based on factors unrelated to the specific item content, known as response styles.
- Common response styles include using extreme options (e.g., STRONGLY AGREE/DISAGREE) or agreeing with all statements, affecting segmentation accuracy.
- Segmentation algorithms may misinterpret data due to response styles, leading to incorrect market segment conclusions.

- Minimizing response styles is critical in market segmentation to accurately identify and target genuine consumer segments.
- Additional analysis or exclusion of respondents affected by response styles is necessary to mitigate biased segment interpretations.

### 3.3.10 Sample Size

The below points summarize information about the Sample Size of data.

- Market segmentation analysis requires sufficient sample sizes to accurately identify segments; inadequate samples lead to ambiguous results.
- Recommended sample sizes vary: Formann suggests at least 2p (or five times 2p) where p is the number of segmentation variables, while Qiu and Joe (2015) propose $10 \cdot p \cdot k$ for accurate segment identification.
- Empirical studies by Dolnicar et al. (2014) and others show that increasing sample size improves segment extraction algorithm accuracy, especially beneficial for smaller initial samples.
- Challenges such as unequal segment sizes, overlap, and data quality issues (e.g., response biases) complicate segment recovery, necessitating sufficient sample sizes.
- Dolnicar et al. (2016) recommend a guideline of at least 100 respondents per segmentation variable to ensure robust segmentation outcomes, emphasizing high-quality data collection.
- The impact of sample size on segment recovery varies with data characteristics; while some challenges can be mitigated with larger samples, others, like high correlation between variables, remain problematic even with increased sample size.

### 3.3.11 Data from Internal sources & Experimental studies

- Organizations increasingly rely on internal data sources such as scanner data from grocery stores, booking data from airline loyalty programs, and online purchase data for market segmentation.
- Internal data strengths include its reflection of actual consumer behavior rather than self-reported data, which is prone to memory imperfections and response biases (Niemi 1993; Fisher 1993; Paulhus 1991; Dolnicar and Grün 2007a,b, 2009).
- Such data is automatically generated and easily accessible if stored in accessible formats, requiring minimal additional effort for collection and processing.
- However, internal data may be biased towards existing customers, potentially missing insights into different consumption patterns of future customers.

- Experimental data, derived from field or laboratory experiments, also informs market segmentation, such as studies on consumer responses to advertisements or choice experiments evaluating preferences based on product attributes.
- Conjoint analyses and choice experiments provide insights into how specific product attributes influence consumer choices, which can be used as segmentation criteria.
- Experimental data offers controlled settings for studying consumer behavior, although translating experimental findings to real-world market segments requires careful consideration.

## 3.4 Step 4 - Data Exploration

### 3.4.1 Loading and having a glance at the data

After data collection, exploratory data analysis (EDA) is crucial for cleaning and pre-processing the data and selecting appropriate segmentation algorithms. EDA helps to identify measurement levels of variables, investigate univariate distributions, and assess dependencies between variables. This pre-processing ensures the data is compatible with segmentation algorithms and guides the selection of the most suitable segmentation methods based on EDA results.

This analysis can be done on languages such as python or R. In the textbook, Market Segmentation Analysis they have used R to explore the data of the Australian Vacation dataset, which can be found over here. Through this they have extracted important information from the dataset. Some of the following R commands were used,

```
R> vaccsv <- system.file("csv/vacation.csv",+   package = "MSA")
R> file.copy(vaccsv, ".")
R> vac <- read.csv("vacation.csv", check.names = FALSE)
R> colnames(vac)
R> dim(vac)
R> summary(vac[, c(1, 2, 4, 5)])
```

implementing these lines of code in R will give you the desired outputs with the dataset. Please explore!

### 3.4.2 Data Cleaning

- The first step in data analysis is to clean the data, ensuring all values are recorded correctly and categorical variables have consistent labels.
- Check the range of plausible values for metric variables, such as age, which should lie between 0 and 110 years, to identify any data entry errors.
- Verify that categorical variables contain only permissible values; for example, gender should typically have only two values: female and male.
- In the Australian travel motives data set, the variables Gender and Age required no cleaning, but the Income2 variable categories were not sorted.
- This sorting issue is due to R's `read.csv()` or `read.table()` functions converting non-numeric columns into factors, which are sorted alphabetically by default.
- To re-order categories in R, copy the column to a helper variable (`inc2`), store its levels (`lev`), find the correct order, and convert it into an ordered factor.

- Copy the `Income2` column to a helper variable and check its levels:

```
R> inc2 <- vac$Income2
R> levels(inc2)
[1] "<30k"    ">120k"   "30-60k"
```

- Store the levels in a variable and reorder them correctly:

```
R> lev <- levels(inc2)
R> lev[c(1, 3, 4, 5, 2)]
[1] "<30k"    "30-60k"  "60-90k"  "90-120k" ">120k"
```

- Transform the variable into an ordered factor:

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

- Double-check the transformation by cross-tabulating the original and new variables:

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

- Overwrite the original column with the correctly ordered version:

```
R> vac$Income2 <- inc2
```

- Ensure reproducibility by keeping all R code for data transformations and saving the cleaned data set using `save()` and `load()` functions.

### 3.4.3 Descriptive Analysis

- Being familiar with the data avoids misinterpretation of results from complex analyses.
- Descriptive numeric and graphic representations provide insights into the data.
- In R, the `summary()` command returns ranges, quartiles, and means for numeric variables, as well as frequency counts and missing values for categorical variables.
- Useful graphical methods for numeric data include histograms, boxplots, and scatter plots.
- Bar plots visualize frequency counts for categorical variables, while mosaic plots illustrate the association of multiple categorical variables.
- Histograms, created through binning, visualize the distribution of numeric variables by plotting the frequency of observations within specified value ranges.

#### 3.4.3.1 Histograms

- Use the `lattice` package in R for creating histograms by segments:

```
R> library("lattice")
```

- Construct a histogram for variable `Age` in the dataset `vac`:

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

- Specify the number of bins (`breaks`) to gain deeper insights:

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

- The finer bins provide more detailed information, revealing the distribution characteristics such as bi-modality.
- Use `type = "density"` to scale the y-axis by density estimates, allowing comparison with parametric distributions.
- This representation is typically preferred for histograms to visualize the probability density functions.

This is what the histogram looks like,

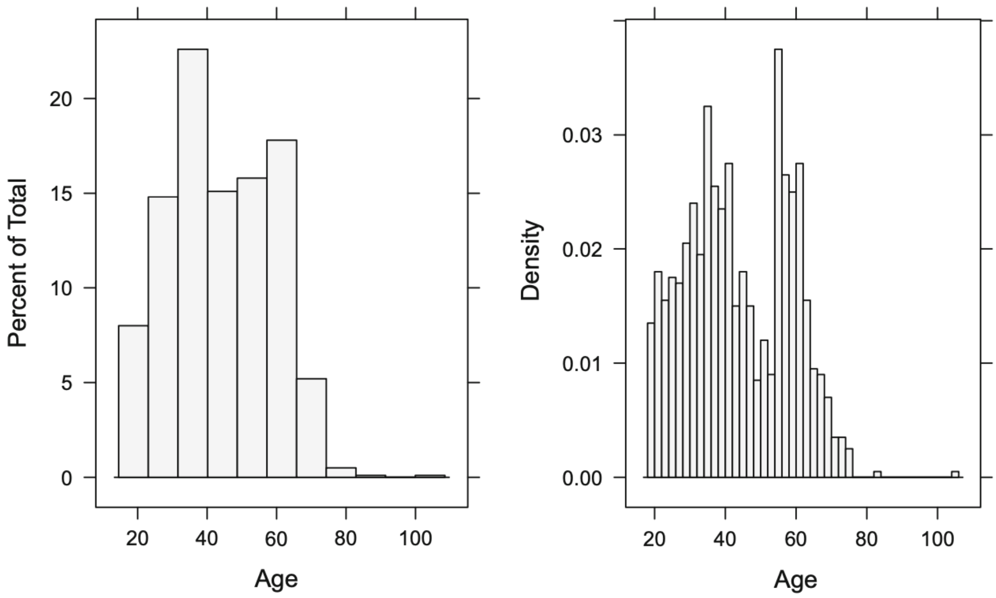



**Fig. 3.1:** Histogram from *Market Segmentation Analysis* textbook

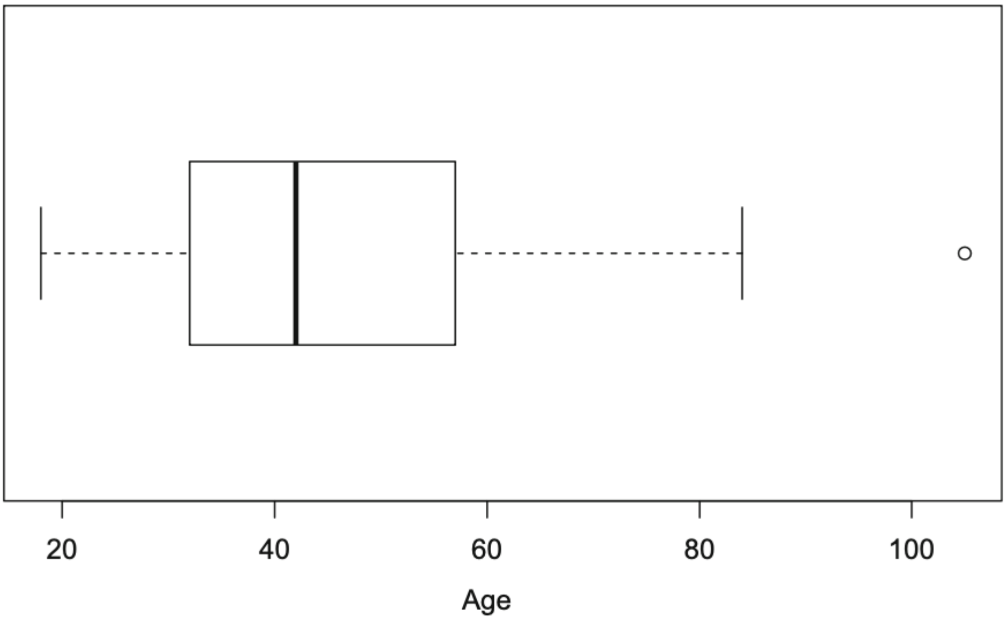



**Fig. 3.2:** Box-and-Whisker plot of tourist age from *Market Segmentation Analysis* textbook

#### 3.4.3.2 Box and Whisker plot

- **Box-and-Whisker Plot (Boxplot):**
  - Summarizes data using the minimum, first quartile, median, third quartile, and maximum.
  - Provides insights into the distributional properties assuming unimodality.
  - Highlights outliers beyond 1.5 times the size of the box as circles to prevent loss of outlier information.
- **Use in R:**
  - In R, create a horizontal boxplot for variable `Age` with `boxplot(vac$Age, horizontal = TRUE, xlab = "Age")`.
- **Dot Chart for Percentages:**
  - The dot chart visualizes the percentages of agreement with travel motives from columns 13 to 32 in the dataset.
  - Each dot on the chart represents the percentage of respondents indicating that a specific travel motive was important to them on their last vacation.
  - To compute these percentages in R, we use `100 * colMeans(vac[, 13:32] == "yes")`. This calculates the mean percentage of "yes" responses across all columns.

- The dot chart is sorted to display these percentages from lowest to highest, providing a clear view of the distribution of agreement levels across different motives.
- It serves as a powerful tool to quickly grasp which travel motives are universally popular among respondents and which ones are less universally appealing.
- Insights gained from the dot chart highlight the heterogeneity in the importance attributed to various travel motives among survey participants.
- This heterogeneity underscores the potential of these motives as effective segmentation variables in market analysis, as they reveal distinct preferences and behaviors among different segments of the population.

- **Insights from Dot Chart:**
  - Illustrates varying agreement levels with different travel motives.
  - Confirms heterogeneity in importance attributed to travel motives, suggesting suitability as segmentation variables.

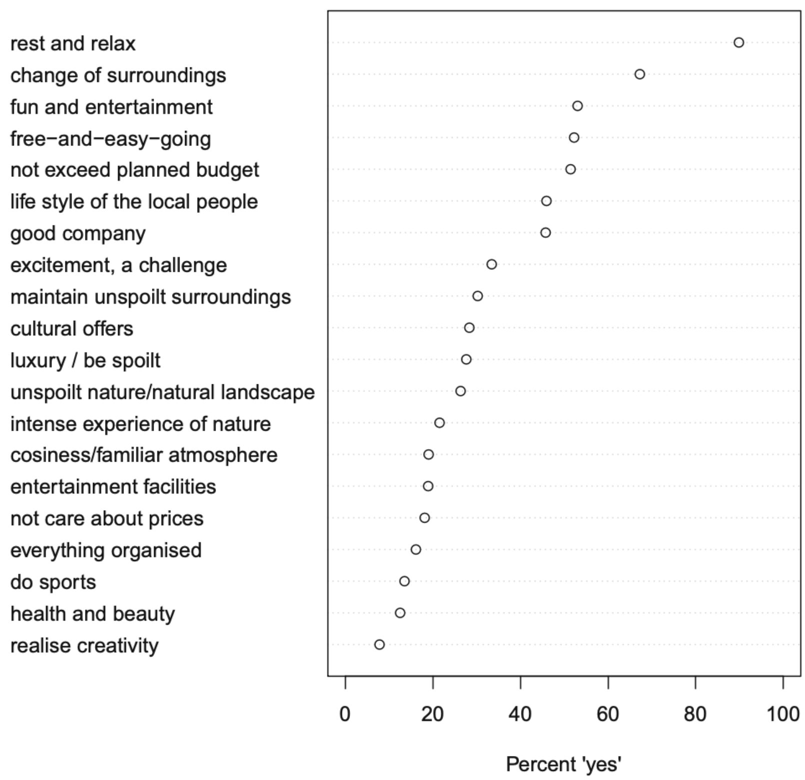
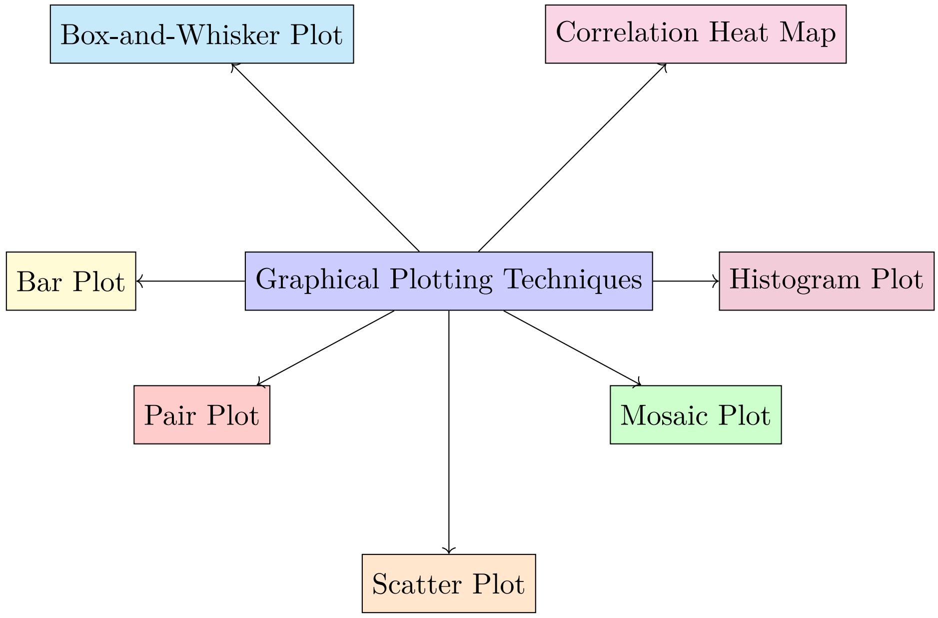



**Fig. 3.3:** Dot Chart of Yes percentages *Market Segmentation Analysis* textbook

So in short we have the following plotting devices to help us explain and portray the data set in a thorough way. I will be mentioning other plots as well, which have not been used in the Market Segmentation Analysis, but I have knowledge of them.




### 3.4.4 Pre-Processing

#### 3.4.4.1 Categorical Variables

- Merge levels of categorical variables to reduce complexity.
- Convert categorical variables to numeric where scale assumptions hold.
- Consider binary options over multi-category scales for simplicity.
- Example: Convert survey responses to 0/1 matrix in R (`vacmot <- (vac[, 13:32] == "yes") + 0`).
- Preprocessing alters data for compatibility with statistical methods.
- Reflect on survey response options and cultural influences.
- Assume numeric variables with comparable scales for analysis methods.
- Use R packages like `flexclust` for handling categorical data effectively.

#### 3.4.4.2 Numerical Variables

- Standardize numerical variables to put them on a common scale:

$$z_i = \frac{x_i - \bar{x}}{s}$$

where $\bar{x}$ is the mean and $s$ is the standard deviation.

- Use R function `scale()` for standardization:

```
vacmot.scaled <- scale(vacmot)
```

- Consider robust methods for location and spread if data has outliers:

$$\text{Median: } \tilde{x} \quad \text{Interquartile Range (IQR): IQR} = Q3 - Q1$$

- Distance-based methods assume variables are numeric and comparable.
- Ordinal data can be treated as numeric if distances between scale points are assumed equal.
- Likert scales may not have equal distances between categories due to response biases.
- Convert dichotomous ordinal or nominal variables to binary (0/1) for simplicity:

```
vacmot <- (vac[, 13:32] == "yes") + 0
```

- Use appropriate statistical methods after conversion to ensure validity of results.

### 3.4.5 Principal Components Analysis

- PCA transforms multivariate data into uncorrelated principal components that explain variance in descending order.
- It works from the covariance or correlation matrix of numeric variables, sensitive to data scaling.
- Principal components are used for dimensionality reduction and visualizing high-dimensional data.
- Interpret PCA results using standard deviations, explained variances, and cumulative proportions.
- Visualization typically involves the first few principal components, e.g., PC1 and PC2 in scatter plots.
- PCA assists in identifying patterns and relationships among variables, aiding in market segmentation and data reduction tasks.

## 3.5 Step 5 - Selecting the Targeting Segments

We had talked about this somewhat briefly in the first few chapters, and now we revisit again to discuss in more detail

### 3.5.1 Target Decision

- **Critical Decision**
  Selecting specific market segments for targeting is a crucial, long-term decision affecting the organization's future performance.
- **Segmentation Process**
  After conducting a global market segmentation analysis, segments are available for detailed inspection and evaluation.
- **Segment Profiling and Description**
  Segments are profiled by inspecting key characteristics and described in detail to ensure they are identifiable, reachable, and their needs align with the organization's capabilities.
- **Double-Check Knock-Out Criteria**
  Ensure all segments meet essential criteria: sufficient size, homogeneity, distinctiveness, identifiability, reachability, and alignment with the organization's capabilities.
- **Evaluating Segment Attractiveness**
  Assess the attractiveness of each segment based on factors such as potential profitability, growth potential, and strategic fit with the organization.
- **Assessing Organizational Competitiveness** Evaluate the organization's competitiveness in serving each segment, considering factors like market position, resources, and ability to meet segment needs effectively.
- **Key Targeting Questions**
  Determine which segments the organization most wants to target and commit to, and which segments are most likely to choose the organization over competitors.
- **Basis for Decision**
  The answers to the above questions form the basis for making the final target segment selection.

### 3.5.2 Market Segment Evaluation

- **Decision Matrix Usage**
  Decision matrices help visualize segment attractiveness and organizational competitiveness, aiding in target market selection. Various versions exist, including the Boston matrix, GE/McKinsey matrix, and directional policy matrix.

- **Segment Attractiveness and Competitiveness**
  The x-axis represents segment attractiveness ("How attractive is the segment to us?") and the y-axis represents relative organizational competitiveness ("How attractive are we to the segment?").
- **Criteria Weighting and Rating**
  Criteria for segment attractiveness and organizational competitiveness are weighted and rated. For example, Criterion 1 (weight 25%) for Segment 1 is rated 5 for attractiveness and 2 for competitiveness.
- **Weighted Calculation**
  Total values for each segment are calculated by multiplying the weights with the ratings and summing them up. Segment 1's attractiveness is calculated as $0.25 \times 5 + 0.35 \times 2 + 0.20 \times 10 + 0.10 \times 8 + 0.10 \times 9 = 5.65$.
- **Bubble Size Representation**
  The size of the bubbles in the plot represents another criterion, such as profit potential. For example, Segment 1 has a bubble size of 2.25.
- **Segment Evaluation Plot**
  The plot helps identify the most promising segments. Segment 8 is highly attractive to the organization and vice versa, but has low profit potential (bubble size 1.50). Segment 5 is highly attractive with high profit potential but less organizational appeal.
- **Selecting Target Segments**
  Based on the plot, segments 3 and 7 might be excluded despite their high profit potential due to low attractiveness. Segment 5 is highly attractive but has low compatibility. Segment 8 is highly compatible but has low profit potential.
- **Practical Application**
  Organizations can use the decision matrix to prioritize segments that align well with their strategic goals, balancing attractiveness and competitiveness.
- **Balancing Act**
  The goal is to find segments that balance attractiveness and competitiveness. A highly attractive segment may not always be the best choice if the organization cannot compete effectively in that segment.
- **Profit Potential**
  Bubble size in the plot often represents profit potential, which combines segment size and spending behavior, crucial for selecting target segments.
- **Strategy Alignment**
  The selected segments should align with the organization's strategic goals and capabilities, ensuring that resources are effectively utilized.
- **Visual Decision Aid**
  The segment evaluation plot serves as a visual aid to facilitate discussion and decision-making within the segmentation team, making it easier to compare and contrast different segments.

- **Hypothetical Example**
  In the example, Segment 8 is highly attractive to both the organization and the segment itself, making it a good target despite lower profit potential. Segment 5, while highly attractive, may require more effort to improve organizational competitiveness.

**Table 3.1:** Data underlying the segment evaluation plot(Taken from the textbook)

| | Weight | Seg 1 | Seg 2 | Seg 3 | Seg 4 | Seg 5 | Seg 6 | Seg 7 | Seg 8 |
|---|---|---|---|---|---|---|---|---|---|
| **How attractive is the segment to us? (segment attractiveness)** | | | | | | | | | |
| **Criterion 1** | 25% | 5 | 10 | 1 | 5 | 10 | 3 | 1 | 10 |
| **Criterion 2** | 35% | 2 | 1 | 2 | 6 | 9 | 4 | 2 | 10 |
| **Criterion 3** | 20% | 10 | 6 | 4 | 4 | 8 | 2 | 1 | 9 |
| **Criterion 4** | 10% | 8 | 4 | 2 | 7 | 10 | 8 | 3 | 10 |
| **Criterion 5** | 10% | 9 | 6 | 1 | 4 | 7 | 9 | 7 | 8 |
| **Total** | 100% | 5.65 | 5.05 | 2.05 | 5.25 | 8.95 | 4.25 | 2.15 | 9.60 |
| **How attractive are we to the segment? (relative organisational competitiveness)** | | | | | | | | | |
| **Criterion 1** | 25% | 2 | 10 | 10 | 10 | 1 | 5 | 2 | 9 |
| **Criterion 2** | 25% | 3 | 10 | 4 | 6 | 2 | 4 | 3 | 8 |
| **Criterion 3** | 25% | 4 | 10 | 8 | 7 | 3 | 3 | 1 | 10 |
| **Criterion 4** | 15% | 9 | 8 | 3 | 9 | 4 | 5 | 3 | 9 |
| **Criterion 5** | 10% | 1 | 8 | 6 | 2 | 1 | 4 | 4 | 8 |
| **Total** | 100% | 3.70 | 9.50 | 6.55 | 7.30 | 2.20 | 4.15 | 2.35 | 8.90 |
| **Size** | | 2.25 | 5.25 | 6.00 | 3.75 | 5.25 | 2.25 | 4.50 | 1.50 |

## 3.6 Step 6 - Customising the Marketing Mix

### Implementations for Marketing Mix Decisions

- **Evolution of Marketing**
  Initially viewed as a toolbox to assist in selling products, marketing combined various elements to achieve optimal sales results (Dolnicar and Ring 2014). Borden (1964) identified 12 marketing ingredients, but the common understanding now revolves around the 4Ps: Product, Price, Promotion, and Place.
- **Market Segmentation**
  It is integral to strategic marketing, closely linked with positioning and competition. The segmentation-targeting-positioning (STP) approach involves segment extraction, profiling, description, targeting, and positioning.

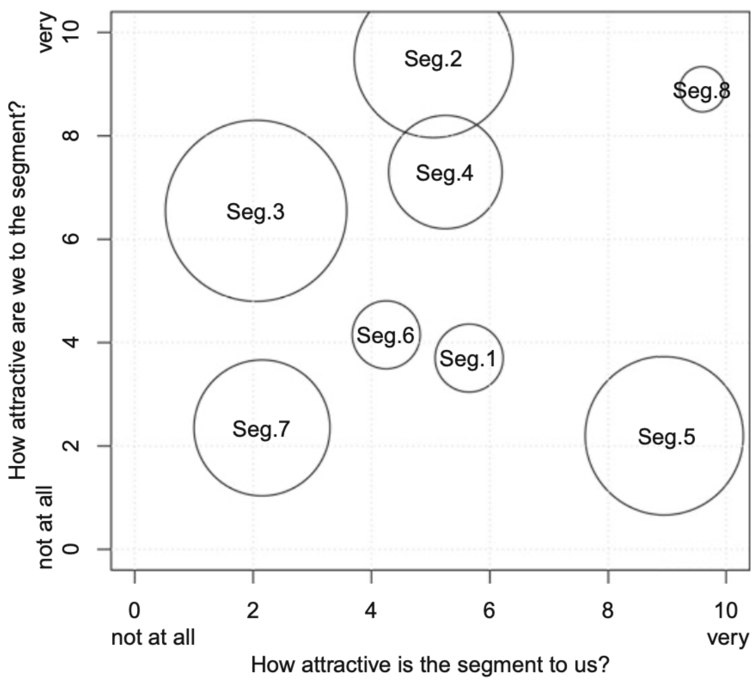
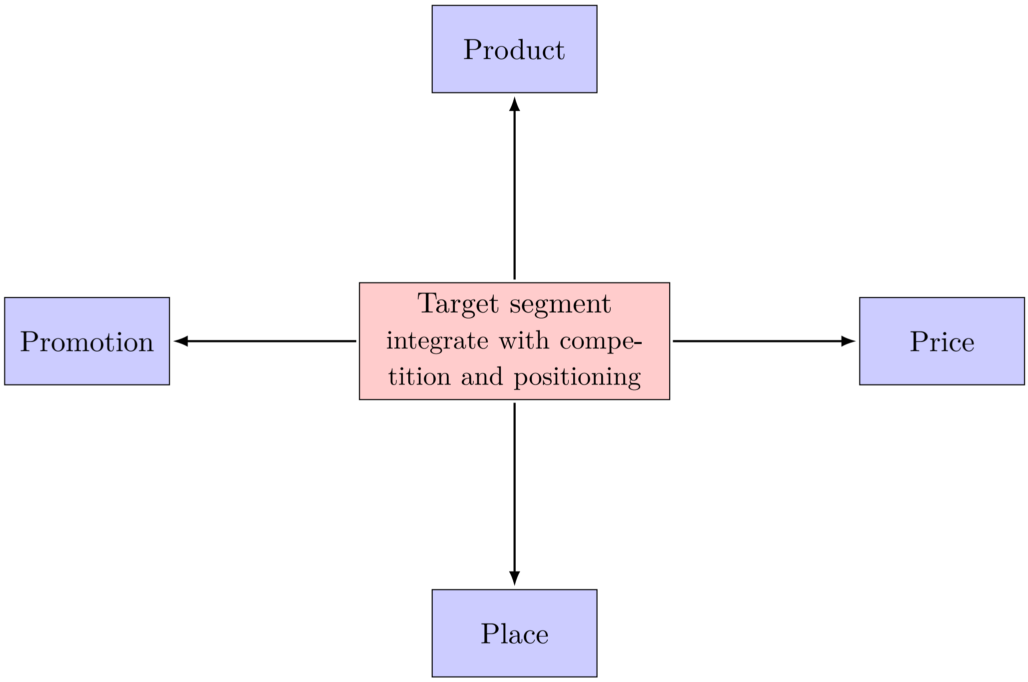



**Fig. 3.4:** Segment Evaluation Plot(Taken from the Textbook)

- **STP Approach**
  Ensures segmentation is integrated with other strategic decisions. The process may not always be linear, requiring adjustments between segmentation and targeting.
- **Impact on Marketing Mix**
  The selection of target segments influences the development of the marketing mix, traditionally comprising the 4Ps. Each aspect must be customized to align with the selected target segments.
- **Customizing Marketing Mix**
  To maximize benefits, the marketing mix should be tailored to the target segment. This may involve designing new products, adjusting prices, selecting appropriate distribution channels, and crafting targeted promotional strategies.
- **Segmentation Variables**
  The focus can be structured around one of the 4Ps, influencing the choice of segmentation variables. For pricing, variables like price sensitivity are relevant; for advertising, lifestyle and psychographic variables are useful; for distribution, store loyalty and patronage are valuable.
- **Holistic Approach**
  Typically, segmentation analysis is not limited to one of the 4Ps. Instead,

insights from a detailed target segment description guide the organization in developing or adjusting the entire marketing mix to cater to the chosen segment.




### 3.6.1 Product

- The product dimension of the marketing mix involves decisions related to specifying the product based on customer needs.
- This typically means modifying an existing product rather than designing a completely new one.
- Decisions include naming the product, packaging, offering warranties, and providing after-sales support services.
- An example of a product measure could be developing a "MUSEUMS, MONUMENTS  MUCH, MUCH MORE" product with an activities pass.
- Another example could be making gardens at the destination an attraction in their own right.

### 3.6.2 Price

- The price dimension of the marketing mix involves setting the price for a product and deciding on discounts to offer.
- It is important to understand the spending behavior of the target segment to make informed pricing decisions.

- Analysis of expenditure data can reveal if a segment has higher vacation expenditures per person per day.
- Higher expenditure segments may allow for premium pricing strategies rather than discounted prices.
- For instance, if a target segment shows higher spending, a premium price can be attached to a specialized product like "MUSEUMS, MONUMENTS MUCH, MUCH MORE".

### 3.6.3 Place

- The place dimension of the marketing mix involves decisions on how to distribute the product to customers.
- Key considerations include whether the product should be available online, offline, or both.
- Decisions need to be made on selling directly to customers or through wholesalers or retailers.
- Understanding the booking preferences of the target segment can help ensure that the product is available through preferred distribution channels.
- Visualizing booking behavior can provide insights into the preferred booking methods of the target segment.

### 3.6.4 Promotion

- The promotion dimension involves developing an advertising message and identifying effective communication methods.
- Tools in this category include public relations, personal selling, and sponsorship.
- It is important to determine the best information sources to reach the target segment.
- Comparing information sources and preferred TV stations of the target segment can help tailor the promotional strategy.
- Insights into preferred information sources can guide the design of specific information packs available both in tourist centers and online.
- Understanding TV channel preferences can help develop a media plan for maximum exposure to the target segment.

Appendix A

# Code

In this section, I have written the converted Python code for the Fast Food Case Study from the R code given in the Market Segmentation Analysis textbook.

## A.1 Exploring the Data

The dataset contains responses from 1453 Australian adults about their perceptions of McDonald's based on these attributes: YUMMY, CONVENIENT, SPICY, FATTENING, GREASY, FAST, CHEAP, TASTY, EXPENSIVE, HEALTHY, and DISGUSTING. Each attribute has a binary response: YES (McDonald's has this attribute) or NO (McDonald's does not).
Below is the Python code for performing Principal Component Analysis (PCA) on the McDonald's dataset:

```
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import entropy
from patsy import dmatrices
from scipy.stats import chi2_contingency
import matplotlib.patches as mpatches
from statsmodels.graphics.mosaicplot import mosaic
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
from adjustText import adjust_text

mcdonalds_data =
↪ pd.read_csv('/Users/kanishka.arora/Downloads/mcdonalds.csv')

# Convert YES/NO to binary
MD_x = mcdonalds_data.iloc[:, :11].applymap(lambda x: 1 if x == 'Yes'
↪ else 0)

# Principal Components Analysis (PCA)
MD_pca = PCA()
```

```python
MD_pca_fit = MD_pca.fit(MD_x)

# Transform data to principal components
MD_pca_proj = MD_pca.transform(MD_x)

# Plot PCA results with annotations
plt.figure(figsize=(12, 10))
plt.scatter(MD_pca_proj[:, 0], MD_pca_proj[:, 1], color='grey', alpha=0.6, s=100, edgecolor='k', label='Data Points')

# Collect all texts to adjust their positions
texts = []
for i, (pc1, pc2) in enumerate(zip(MD_pca.components_[0, :], MD_pca.components_[1, :])):
    plt.arrow(0, 0, pc1, pc2, color='red', alpha=0.75, head_width=0.05, head_length=0.1)
    text = plt.text(pc1 * 1.2, pc2 * 1.2, MD_x.columns[i], color='red', ha='center', va='center', fontsize=12, fontweight='bold')
    texts.append(text)

# Adjust text positions to avoid overlap
adjust_text(texts, arrowprops=dict(arrowstyle='->', color='blue'))

# Customize plot appearance
plt.xlabel('Principal Component 1', fontsize=14, fontweight='bold')
plt.ylabel('Principal Component 2', fontsize=14, fontweight='bold')
plt.title('PCA of McDonald\'s Attributes', fontsize=16, fontweight='bold')
plt.grid(True, linestyle='--', alpha=0.7)
plt.axhline(0, color='black', linewidth=0.8)
plt.axvline(0, color='black', linewidth=0.8)
plt.legend(loc='upper right')
plt.tight_layout()

plt.show()
```

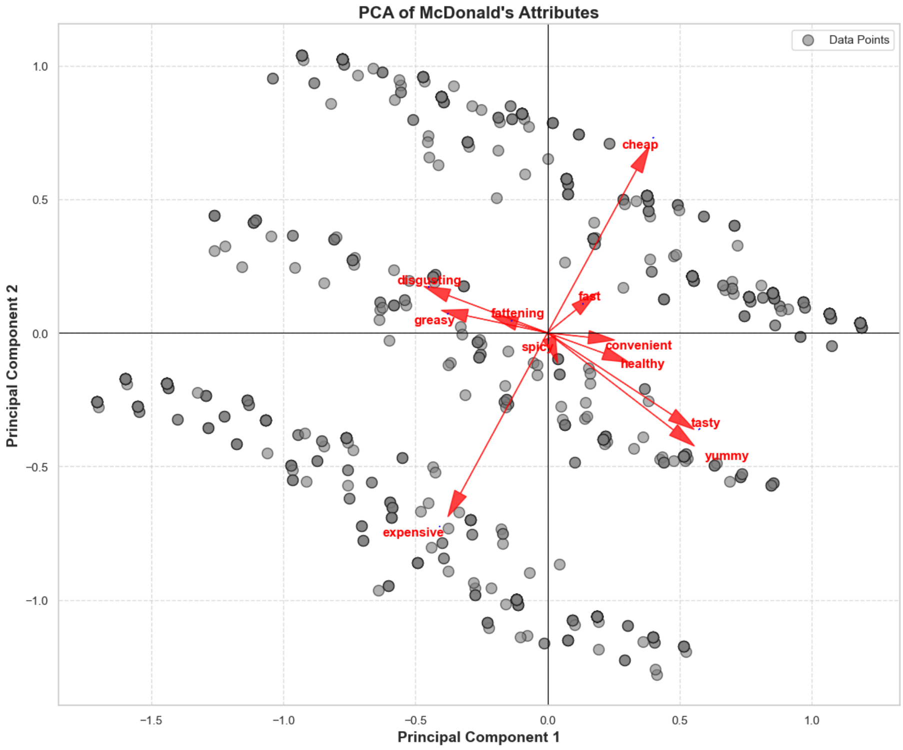



**Fig. A.1:** PCA of McDonald's Attributes

## A.2 Extracting Segments

### A.2.1 Using K-Means

```python
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
range_n_clusters = range(2, 9)
kmeans_models = {}
sum_of_distances = []
for n_clusters in range_n_clusters:
    kmeans = KMeans(n_clusters=n_clusters, n_init=10, random_state=1234)
    kmeans.fit(MD_x)
    kmeans_models[n_clusters] = kmeans
    sum_of_distances.append(kmeans.inertia_)
# Scree Plot
plt.figure(figsize=(12, 8))
bars = plt.bar(range(2, 9), sum_of_distances, color='skyblue',
    edgecolor='black')
for bar in bars:
    yval = bar.get_height()
```

```
    plt.text(bar.get_x() + bar.get_width()/2, yval + 0.05 * max(sum_of_distances), round(yval, 2), ha='center', va='bottom', fontsize=10, fontweight='bold')
plt.plot(range(2, 9), sum_of_distances, color='red', marker='o', linestyle='-', linewidth=2, markersize=6)
plt.xlabel('Number of Segments', fontsize=14, fontweight='bold')
plt.ylabel('Sum of Distances within Segments', fontsize=14, fontweight='bold')
plt.title('Scree Plot for K-means Clustering', fontsize=16, fontweight='bold')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.xticks(range(2, 9), fontsize=12)
plt.yticks(fontsize=12)
plt.tight_layout()
plt.show()
```

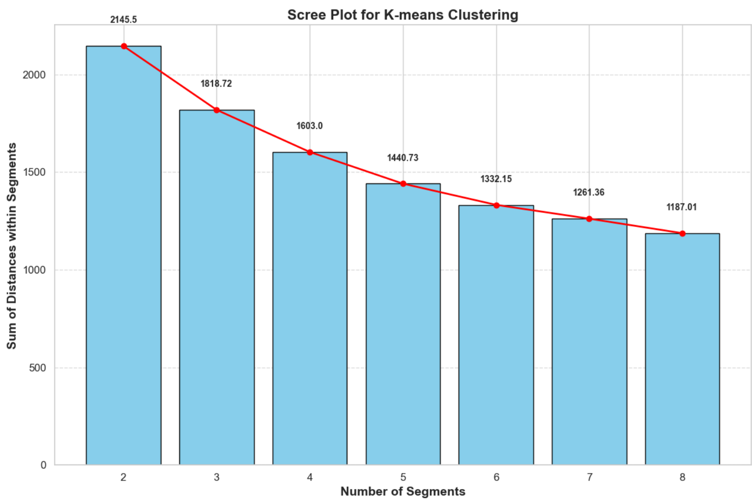



**Fig. A.2:** Scree plot for K-Means clustering.png

```
# Convert YES/NO to binary (1 for 'Yes', 0 for 'No')
MD_x = mcdonalds_data.iloc[:, :11].applymap(lambda x: 1 if x == 'Yes' else 0)

# Function to perform k-means clustering with random restarts
def kmeans_with_random_restarts(data, n_clusters, n_init=10):
    kmeans = KMeans(n_clusters=n_clusters, n_init=n_init, random_state=1234)
    kmeans.fit(data)
    return kmeans

# Function to calculate adjusted Rand index for stability
def calculate_stability(data, kmeans_model, n_boot=100, n_init=10):
```

```
    stability_scores = []
    for _ in range(n_boot):
        boot_sample = resample(data, n_samples=len(data), random_state=_)
        kmeans_boot = kmeans_with_random_restarts(boot_sample,
        ↪  kmeans_model.n_clusters, n_init)
        labels_true = kmeans_model.predict(data)
        labels_boot = kmeans_boot.predict(data)
        ari_score = adjusted_rand_score(labels_true, labels_boot)
        stability_scores.append(ari_score)
    return stability_scores

# Perform global stability analysis for each number of segments (2 to 8)
range_n_clusters = range(2, 9)
stability_results = {}
for n_clusters in range_n_clusters:
    kmeans_model = kmeans_with_random_restarts(MD_x, n_clusters)
    stability_scores = calculate_stability(MD_x, kmeans_model)
    stability_results[n_clusters] = stability_scores

# BoxPlot
plt.figure(figsize=(10, 6))
boxprops = dict(linestyle='-', linewidth=2, color='black')
medianprops = dict(linestyle='-', linewidth=2, color='orange')
whiskerprops = dict(linestyle='-', linewidth=1.5, color='black')
capprops = dict(linestyle='-', linewidth=1.5, color='black')
plt.boxplot(stability_results.values(), notch=False, patch_artist=False,
↪  boxprops=boxprops, medianprops=medianprops,
↪  whiskerprops=whiskerprops, capprops=capprops)
plt.xticks(range(1, len(range_n_clusters) + 1), range_n_clusters,
↪  fontsize=12)
plt.xlabel('Number of Segments', fontsize=12)
plt.ylabel('Adjusted Rand Index', fontsize=12)
plt.title('Global Stability Boxplot', fontsize=14)
plt.ylim(0.4, 1.0)
plt.grid(axis='y', linestyle='--', linewidth=0.5)
plt.show()
```

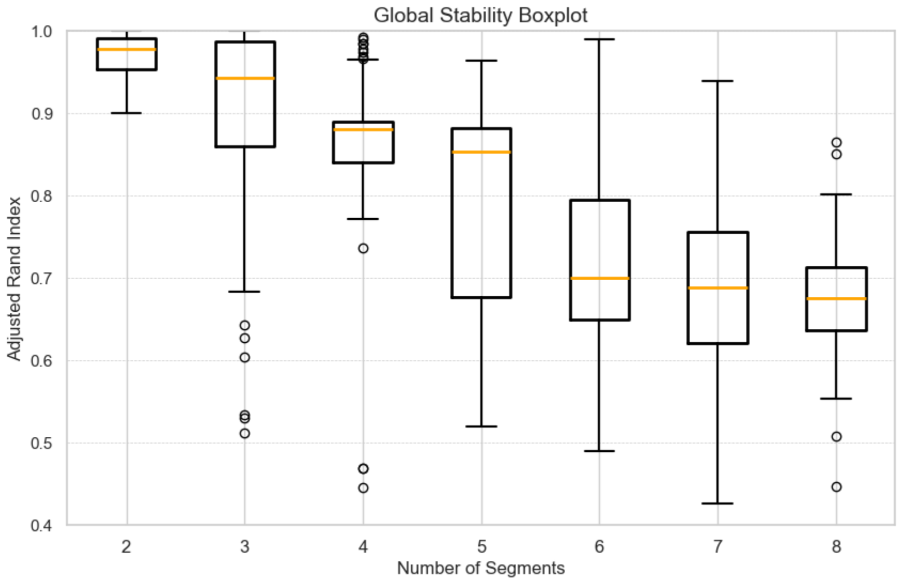



**Fig. A.3:** Global Stability Boxplot

```
# One-hot encode categorical variables
MD_x_numeric = pd.get_dummies(mcdonalds_data, drop_first=True)
scaler = StandardScaler()
MD_x_numeric_scaled = scaler.fit_transform(MD_x_numeric)
n_clusters = 4
kmeans_model = KMeans(n_clusters=n_clusters, n_init=10,
↪ random_state=1234)
kmeans_model.fit(MD_x_numeric_scaled)
labels = kmeans_model.labels_

# Calculate pairwise similarity within each cluster
def calculate_similarity_within_cluster(data, labels, cluster_label):
    cluster_data = data[labels == cluster_label]
    similarity = []
    for i in range(len(cluster_data)):
        for j in range(i + 1, len(cluster_data)):
            similarity.append(np.linalg.norm(cluster_data[i] -
            ↪ cluster_data[j]))
    return similarity
# Plot histograms for each cluster
fig, axes = plt.subplots(2, 2, figsize=(12, 10))
axes = axes.flatten()
fig.suptitle('Gorge plot of the four-segment k-means solution for the
↪ fast food dataset', fontsize=16)
for cluster_label in range(n_clusters):
    similarity = calculate_similarity_within_cluster(MD_x_numeric_scaled,
    ↪ labels, cluster_label)
```

```
    axes[cluster_label].hist(similarity, bins=10, edgecolor='black', density=True)
    axes[cluster_label].set_title(f'Cluster {cluster_label + 1}')
    axes[cluster_label].set_xlabel('Similarity')
    axes[cluster_label].set_ylabel('Percent of Total')
plt.tight_layout(rect=[0, 0, 1, 0.96])  # Adjust layout to make room for the suptitle
plt.show()
```

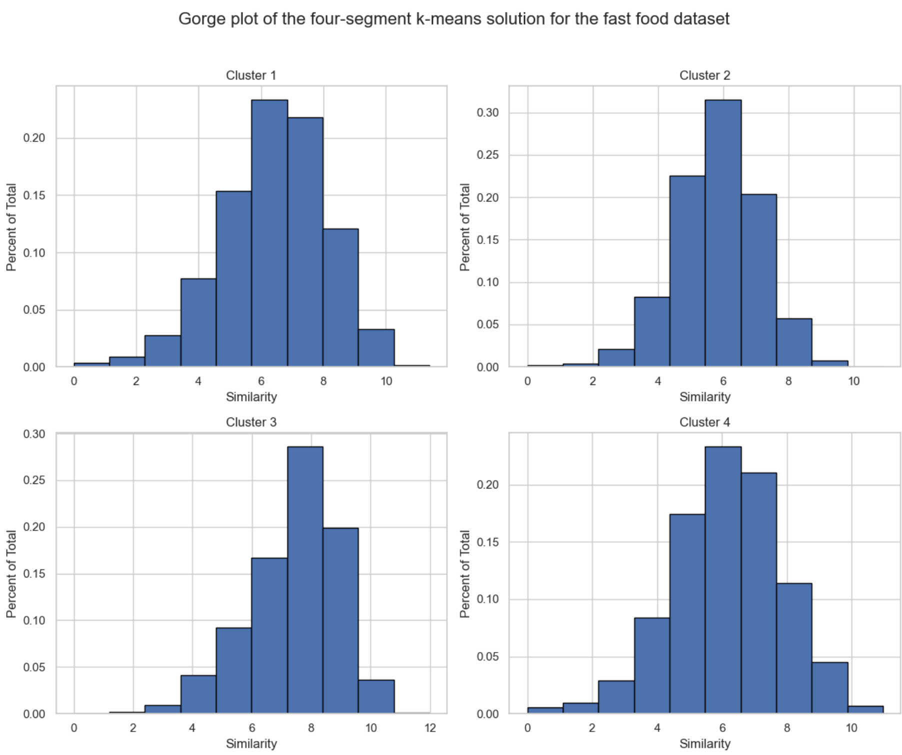



**Fig. A.4:** Gorge plot of the four-segment k-means solution for the fast food dataset

```
MD_x_numeric = pd.get_dummies(mcdonalds_data, drop_first=True)

# Standardize the data
scaler = StandardScaler()
MD_x_numeric_scaled = scaler.fit_transform(MD_x_numeric)
num_segments = range(2, 9)

# Fit KMeans models for each segment
kmeans_models = {}
for segment in num_segments:
    kmeans_models[segment] = KMeans(n_clusters=segment, n_init=10, random_state=1234)
    kmeans_models[segment].fit(MD_x_numeric_scaled)
```

```

# Calculate segment stability
segment_stability = []
for segment in num_segments:
    labels_segment = kmeans_models[segment].labels_
    segment_stability.append(labels_segment)

# SLSA Plot
plt.figure(figsize=(12, 8))
for i, segment in enumerate(num_segments):
    stability_scores = [np.mean(segment_stability[i] == labels) for
    ↪  labels in segment_stability]
    plt.plot(num_segments, stability_scores, marker='o', label=f'Segment
    ↪  {segment}')
plt.xlabel('Number of Segments', fontsize=14, fontweight='bold')
plt.ylabel('Segment Level Stability', fontsize=14, fontweight='bold')
plt.title('Segment Level Stability Across Solutions (SLSA) Plot',
↪  fontsize=16, fontweight='bold')
plt.xticks(num_segments, fontsize=12)
plt.yticks(fontsize=12)
plt.legend(fontsize=12)
plt.grid(True, linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```

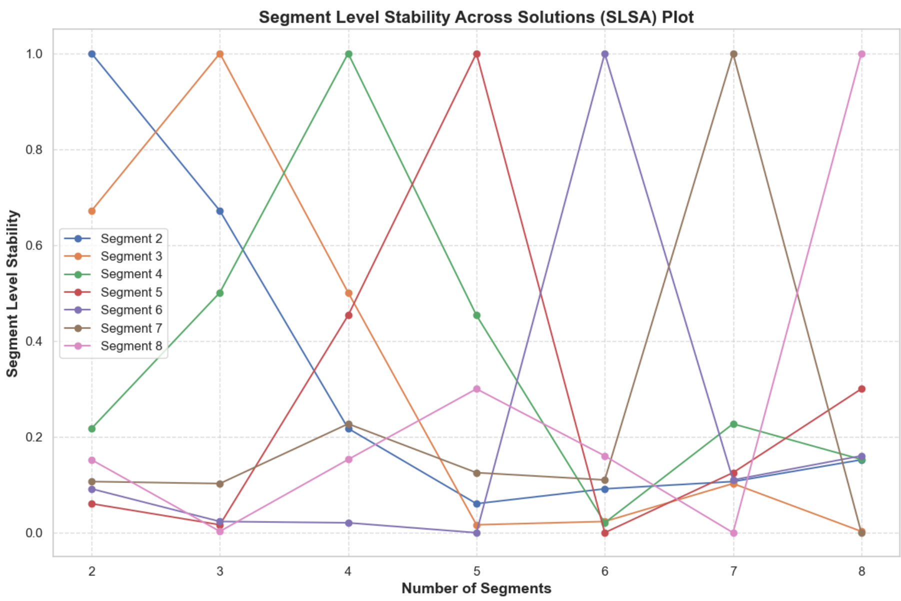



**Fig. A.5:** Segment Level Stability Across Solutions (SLSA) Plot

```
MD_x_numeric = pd.get_dummies(mcdonalds_data, drop_first=True)
scaler = StandardScaler()
```

```
MD_x_numeric_scaled = scaler.fit_transform(MD_x_numeric)
segment_solutions = ["2", "3", "4", "5"]

# Fit KMeans models for each segment and store labels and similarities
segment_labels = {}
segment_similarities = {}
for segment in segment_solutions:
    kmeans = KMeans(n_clusters=int(segment), n_init=10, random_state=1234)
    kmeans.fit(MD_x_numeric_scaled)
    segment_labels[segment] = kmeans.labels_
    segment_similarities[segment] = kmeans.transform(MD_x_numeric_scaled).min(axis=1)

# Calculate segment stability values
segment_stability_values = []
for segment in segment_solutions:
    similarities = segment_similarities[segment]
    normalized_similarities = similarities / np.max(similarities)
    segment_stability_values.append(normalized_similarities)

# Plot Segment Level Stability within Solutions
plt.figure(figsize=(12, 8))
plt.boxplot(segment_stability_values, whis=1.5)
plt.xlabel("Segment Number", fontsize=14, fontweight='bold')
plt.ylabel("Segment Stability", fontsize=14, fontweight='bold')
plt.xticks(range(1, len(segment_solutions) + 1), segment_solutions, fontsize=12)
plt.ylim(0, 1)
plt.title("Segment Level Stability within Solutions", fontsize=16, fontweight='bold')
plt.grid(True, linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```

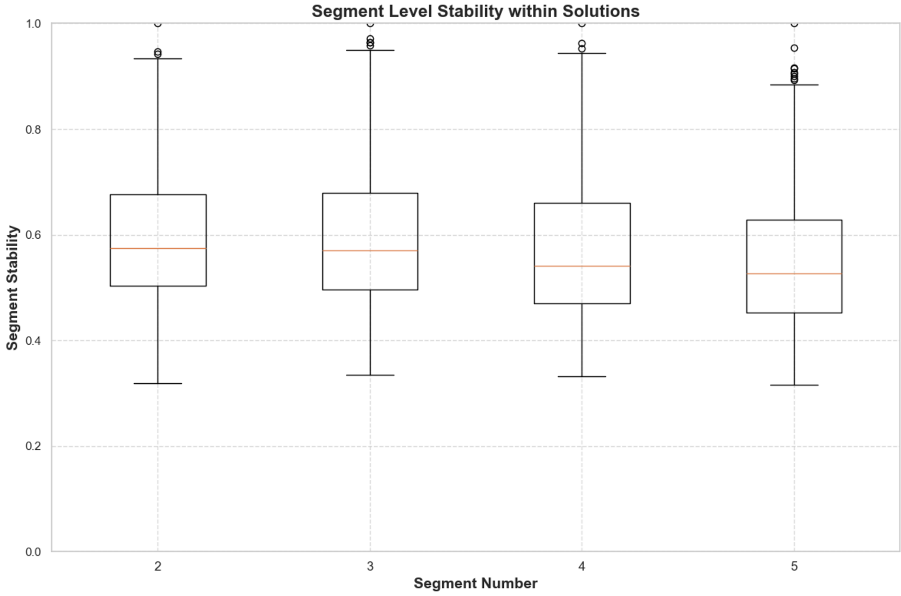



**Fig. A.6:** Segment Level Stability within solutions

### A.2.2 Using mixtures of distributions

```
# Convert non-numeric columns to numeric using one-hot encoding
MD_x_numeric = pd.get_dummies(MD_x)
np.random.seed(1234)
k_values = range(2, 9)
MD_m28 = []

for k in k_values:
    model = KMeans(n_clusters=k, random_state=1234)
    model.fit(MD_x_numeric.values)
    iter_val = model.n_iter_
    converged = model.n_iter_ < model.max_iter
    log_likelihood = -model.inertia_
    n_samples, _ = MD_x_numeric.shape
    aic = -2 * log_likelihood + 2 * k
    bic = -2 * log_likelihood + np.log(n_samples) * k
    labels = model.labels_
    counts = np.bincount(labels)
    probs = counts / float(counts.sum())
    class_entropy = entropy(probs)
    icl = bic - class_entropy

    MD_m28.append((iter_val, converged, k, k, log_likelihood, aic, bic,
    ↪  icl))

```

```
MD_m28 = pd.DataFrame(MD_m28, columns=['iter', 'converged', 'k', 'k0', 'logLik', 'AIC', 'BIC', 'ICL'])

print(MD_m28)
num_segments = MD_m28["k"]
AIC_values = MD_m28["AIC"]
BIC_values = MD_m28["BIC"]
ICL_values = MD_m28["ICL"]

# Set Seaborn style for better aesthetics
sns.set(style="whitegrid")
plt.figure(figsize=(12, 8))
sns.lineplot(x=num_segments, y=AIC_values, marker='o', label='AIC (Akaike Information Criterion)', linestyle='-', color='b')
sns.lineplot(x=num_segments, y=BIC_values, marker='o', label='BIC (Bayesian Information Criterion)', linestyle='-', color='r')
sns.lineplot(x=num_segments, y=ICL_values, marker='o', label='ICL (Integrated Complete Likelihood)', linestyle='-', color='g')
plt.xlabel('Number of Segments', fontsize=14)
plt.ylabel('Value of Information Criteria', fontsize=14)
plt.title('Information Criteria (AIC, BIC, ICL)', fontsize=16, fontweight='bold')
plt.legend(title='Criteria', fontsize=12, title_fontsize='13')
plt.xticks(num_segments)
plt.grid(True)
plt.tight_layout()
plt.show()
```

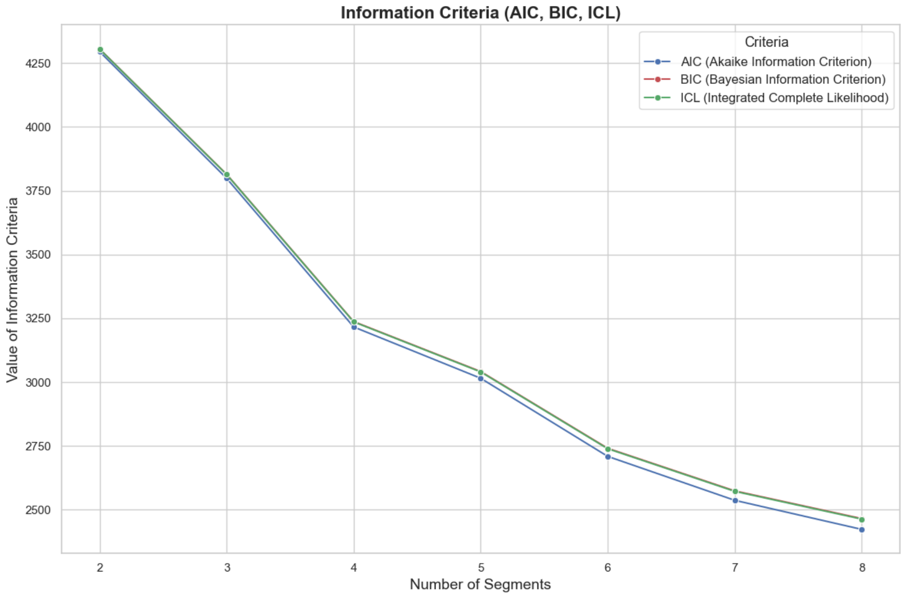



**Fig. A.7:** Information Criteria (AIC, BIC, ICL)

```
# Perform KMeans clustering
k_values = range(2, 9)
kmeans_models = {}
np.random.seed(1234)
for k in k_values:
    model = KMeans(n_clusters=k, random_state=1234)
    model.fit(MD_x_numeric)
    kmeans_models[k] = model

# Select the four-segment solution from k-means
MD_k4 = kmeans_models[4]

# Create a function to generate a mixture model
def fit_gmm(X, n_components, random_state=None):
    gmm = GaussianMixture(n_components=n_components,
    ↪  random_state=random_state)
    gmm.fit(X)
    return gmm

# Fit a Gaussian Mixture Model (GMM) with 4 components
MD_m4 = fit_gmm(MD_x_numeric, 4, random_state=1234)

# Cross-tabulation of k-means and GMM clusters
kmeans_labels = MD_k4.labels_
gmm_labels = MD_m4.predict(MD_x_numeric)
cross_tab = pd.crosstab(kmeans_labels, gmm_labels, rownames=['kmeans'],
↪  colnames=['mixture'])
```

```
print(cross_tab)

# Initialize GMM with k-means labels
MD_m4a = GaussianMixture(n_components=4, random_state=1234)
MD_m4a.fit(MD_x_numeric.values, y=kmeans_labels)
gmm_labels_a = MD_m4a.predict(MD_x_numeric)

# Cross-tabulation of k-means and GMM clusters with initialized labels
cross_tab_a = pd.crosstab(kmeans_labels, gmm_labels_a,
↪  rownames=['kmeans'], colnames=['mixture'])
print(cross_tab_a)

# Comparing the log-likelihood values
logLik_m4 = MD_m4.score(MD_x_numeric) * len(MD_x_numeric)
logLik_m4a = MD_m4a.score(MD_x_numeric) * len(MD_x_numeric)
print(f'Log Likelihood (random init): {logLik_m4}')
print(f'Log Likelihood (k-means init): {logLik_m4a}')
```

```
mixture     0    1    2    3
kmeans
0         508    0    4   37
1           0  215   11    4
2          44    3  267    8
3           0   36   15  301
mixture     0    1    2    3
kmeans
0         508    0    4   37
1           0  215   11    4
2          44    3  267    8
3           0   36   15  301
Log Likelihood (random init): 16082.886182864011
Log Likelihood (k-means init): 16082.886182864011
```

**Fig. A.8:** Output

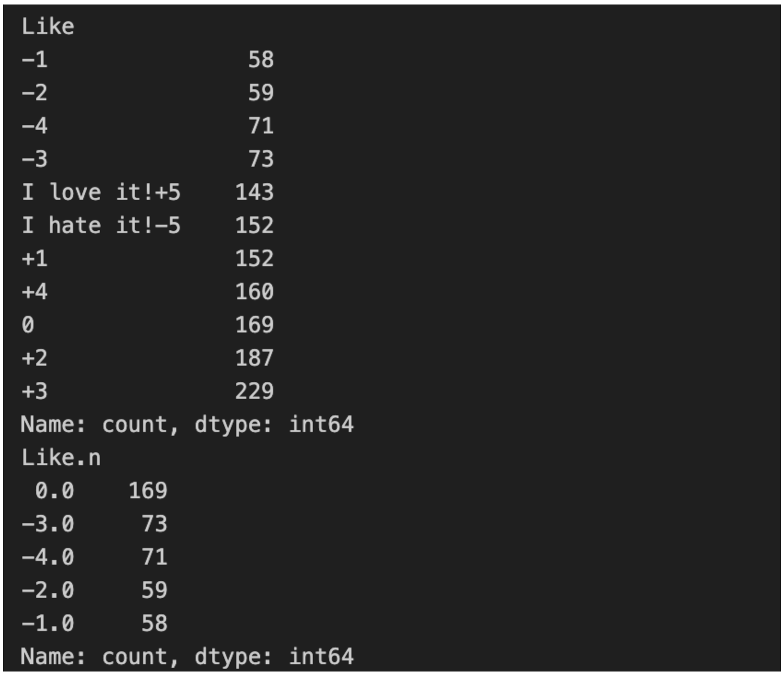
### A.2.3 Using Regression Models

```
like_counts = pd.value_counts(mcdonalds_data['Like'])
reversed_counts = like_counts.iloc[::-1]
print(reversed_counts)
```

```python
# Define a mapping of string values to numeric codes
like_mapping = {
    'I HATE IT!-5': -5,
    '-4': -4,
    '-3': -3,
    '-2': -2,
    '-1': -1,
    '0': 0,
    '1': 1,
    '2': 2,
    '3': 3,
    '4': 4,
    'I LOVE IT!+5': 5
}

# Map 'Like' to numeric values using the defined mapping
mcdonalds_data['Like.n'] = mcdonalds_data['Like'].map(like_mapping)

# Count the occurrences of each numeric value in 'Like.n'
like_n_counts = mcdonalds_data['Like.n'].value_counts()
print(like_n_counts)
```



```
Like
-1               58
-2               59
-4               71
-3               73
I love it!+5    143
I hate it!-5    152
+1              152
+4              160
0               169
+2              187
+3              229
Name: count, dtype: int64
Like.n
 0.0    169
-3.0     73
-4.0     71
-2.0     59
-1.0     58
Name: count, dtype: int64
```

**Fig. A.9:** Output

```python
# Applying the mapping to create 'Like_n' column
mcdonalds_data['Like_n'] = mcdonalds_data['Like'].map(like_mapping)
```

```

# Handle missing values (if any) - remove rows with missing values
mcdonalds_data.dropna(subset=['Like_n'] +
↪   mcdonalds_data.columns[1:-1].tolist(), inplace=True)

print(mcdonalds_data.head())
```

```
   yummy convenient spicy fattening greasy fast cheap tasty expensive healthy  \
0     No        Yes    No       Yes     No  Yes   Yes    No       Yes      No
10    No        Yes    No       Yes     No  Yes   Yes    No        No      No
12    No        Yes    No       Yes     No  Yes   Yes    No        No      No
14    No        Yes    No       Yes     No  Yes    No    No       Yes      No
16   Yes        Yes    No       Yes    Yes  Yes   Yes   Yes        No      No

   disgusting Like  Age      VisitFrequency  Gender  Like_n  Cluster
0          No   -3   61  Every three months  Female      -3        2
10        Yes   -2   53  Every three months  Female      -2        2
12         No    0   65  Every three months    Male       0        2
14         No   -3   67        Once a month    Male      -3        2
16         No    0   34        Once a month  Female       0        2
```

**Fig. A.10:** Output

```
import seaborn as sns
from patsy import dmatrices
from sklearn.mixture import GaussianMixture
import statsmodels.api as sm

# Cleaning column names
print("Original column names:", mcdonalds_data.columns)
mcdonalds_data.columns = [col.replace(' ', '_').replace('.', '_') for col
↪   in mcdonalds_data.columns]
mcdonalds_data = mcdonalds_data.loc[:,
↪   ~mcdonalds_data.columns.duplicated()]
print("Updated column names:", mcdonalds_data.columns)
independent_vars = mcdonalds_data.columns[1:-1].tolist()
dependent_var = 'Like_n'
formula_str = dependent_var + ' ~ ' + ' + '.join(independent_vars)

y, X = dmatrices(formula_str, data=mcdonalds_data,
↪   return_type='dataframe')

# Fit Gaussian Mixture Model
np.random.seed(1234)
n_components = 2  # Adjust number of components if needed
```

```
model = GaussianMixture(n_components=n_components, n_init=10,
↪ random_state=0)
model.fit(X)

# Predict the cluster labels
labels = model.predict(X)
mcdonalds_data['Cluster'] = labels

# Fit linear regression models for each cluster and store coefficients
↪ and standard errors
coefficients = []
standard_errors = []
for cluster in range(n_components):
    cluster_data = mcdonalds_data[mcdonalds_data['Cluster'] == cluster]
    y_cluster, X_cluster = dmatrices(formula_str, data=cluster_data,
    ↪ return_type='dataframe')
    model = sm.OLS(y_cluster, X_cluster).fit()
    coefficients.append(model.params)
    standard_errors.append(model.bse)

# Convert coefficients to DataFrame for plotting
coefficients_df = pd.DataFrame(coefficients).T
coefficients_df.columns = [f'Comp. {i + 1}' for i in range(n_components)]
coefficients_df = coefficients_df.reset_index()

# Convert standard errors to DataFrame for error bars
errors_df = pd.DataFrame(standard_errors).T
errors_df.columns = [f'Comp. {i + 1}' for i in range(n_components)]
errors_df = errors_df.reset_index()

# Plotting the coefficients with error bars
fig, ax = plt.subplots(figsize=(10, 8))

# Plot each component's coefficients with error bars
colors = sns.color_palette("Paired", n_components)
for i in range(n_components):
    ax.barh(coefficients_df['index'], coefficients_df[f'Comp. {i + 1}'],
    ↪ xerr=errors_df[f'Comp. {i + 1}'],
            label=f'Comp. {i + 1}', alpha=0.6, color=colors[i],
            ↪ edgecolor='black')

# Beautifying the plot
ax.set_xlabel('Coefficient Value', fontsize=14)
ax.set_ylabel('Variable', fontsize=14)
ax.set_title('Regression Coefficients for Each Component', fontsize=16,
↪ fontweight='bold')
ax.legend(title='Components', fontsize=12)
ax.grid(True, which='both', linestyle='--', linewidth=0.5)
ax.axvline(x=0, color='grey', linestyle='--', linewidth=0.8)
```

```
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

plt.show()
```

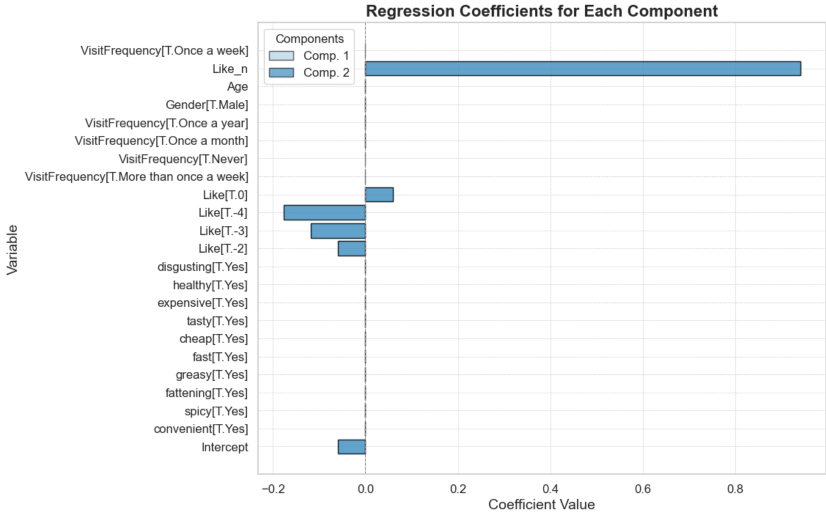



**Fig. A.11:** Regression coefficients for each component

## A.3 Profiling Segments

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


# Setting seed for reproducibility
np.random.seed(42)

# Define attributes and create simulated data
attributes = ['disgusting', 'expensive', 'greasy', 'healthy', 'spicy',
↪  'fast', 'convenient', 'fattening', 'cheap', 'tasty', 'yummy']
data = np.random.rand(100, len(attributes))  # Replace with your actual
↪  data
df = pd.DataFrame(data, columns=attributes)

# Simulated k-means results (replace with actual segment percentages)
segment_data = {
    'Cluster 1': np.random.rand(len(attributes)),
    'Cluster 2': np.random.rand(len(attributes)),
    'Cluster 3': np.random.rand(len(attributes)),
    'Cluster 4': np.random.rand(len(attributes))
}
```

```python
MD_k4 = pd.DataFrame(segment_data, index=attributes)
linked = linkage(df.T, method='ward')
order = leaves_list(linked)
ordered_attributes = [attributes[i] for i in order]

ordered_MD_k4 = MD_k4.loc[ordered_attributes]
fig, axs = plt.subplots(2, 2, figsize=(14, 10), sharex=True, sharey=True)
axs = axs.flatten()

# Sample size and percentage of each cluster (replace with actual numbers)
cluster_info = {'Cluster 1': (470, 32), 'Cluster 2': (257, 18), 'Cluster 3': (324, 22), 'Cluster 4': (402, 28)}
colors = ['blue', 'green', 'red', 'pink']  # Different colors for each segment

for i, (cluster, (size, percent)) in enumerate(cluster_info.items()):
    bars = ordered_MD_k4[cluster]
    marker_threshold = 0.25  # Highlighting threshold (change according to the data)
    ax = axs[i]

    # Plot the bars and highlight marker variables
    for j, (attribute, value) in enumerate(bars.items()):
        ax.barh(j, value, color=colors[i], edgecolor='none', alpha=0.6)
        # Highlight the marker variables
        if abs(value - ordered_MD_k4.mean(axis=1)[attribute]) > marker_threshold:
            ax.plot(value, j, 'ro')

    # Add horizontal lines indicating the overall sample percentage
    sample_means = df.mean(axis=0)
    for k, mean in enumerate(sample_means[ordered_attributes]):
        ax.plot([mean, mean], [k - 0.4, k + 0.4], 'k--', lw=0.7)
    ax.set_yticks(range(len(ordered_attributes)))
    ax.set_yticklabels(ordered_attributes)
    ax.set_xlim(0, 1)
    ax.set_title(f"{cluster}: {size} ({percent}%)")

plt.subplots_adjust(hspace=0.3, wspace=0.3)
fig.suptitle('Segment Profile Plot', fontsize=16)
plt.show()
```


**Fig. A.12:** Segment Profile Plot

```
from scipy.cluster.hierarchy import linkage, leaves_list
np.random.seed(42)
attributes = ['disgusting', 'expensive', 'greasy', 'healthy', 'spicy',
↪   'fast', 'convenient', 'fattening', 'cheap', 'tasty', 'yummy']
data = np.random.rand(100, len(attributes))  # Replace with your actual
↪   data
segments = np.random.choice([1, 2, 3, 4], size=100)  # Simulated segment
↪   membership

# Convert to DataFrame for easier manipulation
df = pd.DataFrame(data, columns=attributes)
df['Segment'] = segments

# Perform PCA
pca = PCA(n_components=2)
pca_result = pca.fit_transform(df[attributes])

# Create a DataFrame with PCA results
df_pca = pd.DataFrame(pca_result, columns=['principal component 1',
↪   'principal component 2'])
df_pca['Segment'] = df['Segment']

# Plotting
fig, ax = plt.subplots(figsize=(12, 10))
colors = {1: 'tomato', 2: 'mediumseagreen', 3: 'royalblue', 4: 'orchid'}
```

```
markers = {1: 'o', 2: '^', 3: 's', 4: 'x'}
for segment in sorted(df_pca['Segment'].unique()):
    subset = df_pca[df_pca['Segment'] == segment]
    ax.scatter(subset['principal component 1'], subset['principal component 2'],
               c=colors[segment], label=f'Segment {segment}', marker=markers[segment], alpha=0.7, s=60)

for segment in sorted(df_pca['Segment'].unique()):
    center = df_pca[df_pca['Segment'] == segment][['principal component 1', 'principal component 2']].mean()
    ax.scatter(center['principal component 1'], center['principal component 2'],
               c='black', edgecolor='white', s=100, zorder=5)  # Reduced size
    # Label as "1", "2", etc. slightly offset from the center
    ax.text(center['principal component 1'], center['principal component 2'],
            str(segment), color='white', ha='center', va='center', fontsize=12, weight='bold', zorder=6)


loadings = pca.components_.T * np.sqrt(pca.explained_variance_)
texts = []
for i, attr in enumerate(attributes):
    ax.arrow(0, 0, loadings[i, 0], loadings[i, 1], color='red', alpha=0.7, head_width=0.02, head_length=0.03)
    # Adjust label positions to avoid overlap
    texts.append(ax.text(loadings[i, 0] * 1.3, loadings[i, 1] * 1.3, attr, color='red', ha='center', va='center', fontsize=10))

# Use adjust_text to prevent overlap
adjust_text(texts, ax=ax, only_move={'text': 'xy'}, arrowprops=dict(arrowstyle="-", color='red', alpha=0.7))
ax.set_xlabel('Principal Component 1', fontsize=14)
ax.set_ylabel('Principal Component 2', fontsize=14)
ax.legend(loc='upper right', fontsize=12, markerscale=1.2)  # Adjusted marker scale
ax.grid(True)
plt.title('Segment Separation Plot using components 1 and 2', fontsize=16)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.tight_layout()

# Save the plot for inclusion in LaTeX
plt.savefig('Segment_Separation_Plot.png')
plt.close()
```

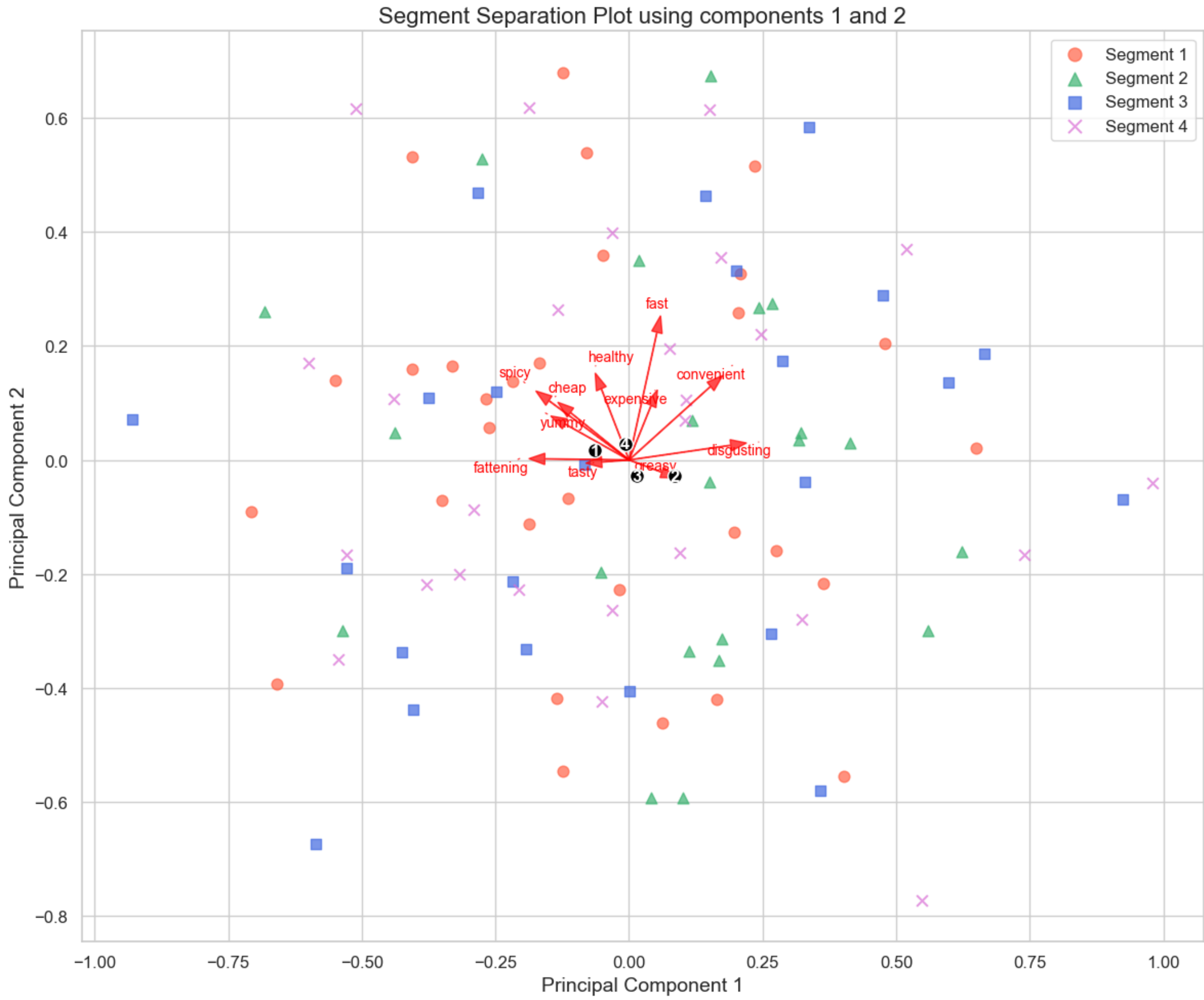



**Fig. A.13:** Segment Separation Plot

## A.4 Describing Segments

```python
from statsmodels.graphics.mosaicplot import mosaic
from scipy.stats import chi2_contingency
import matplotlib.patches as mpatches

# Sample data for demonstration (replace with actual data)
np.random.seed(42)
segments = np.random.choice([1, 2, 3, 4], size=100)  # Simulated segment membership
```

```
like_hate = np.random.choice(['I hate it!', '-5', '-4', '-3', '-2', '-1',
↪  '0', '+1', '+2', '+3', '+4', 'I love it! +5'], size=100)  # Simulated
↪  like/hate variable

# Convert to DataFrame for easier manipulation
df = pd.DataFrame({'Segment': segments, 'Like': like_hate})

# Create a contingency table
contingency_table = pd.crosstab(df['Segment'], df['Like'])

# Perform chi-squared test to get expected frequencies
chi2, p, dof, expected = chi2_contingency(contingency_table)

# Calculate standardized residuals
residuals = (contingency_table - expected) / np.sqrt(expected)

# Define color mapping for standardized residuals
def residual_color(value):
    if value < -4:
        return '#D73027'
    elif value < -2:
        return '#FC8D59'
    elif value < 0:
        return '#FEE08B'
    elif value < 2
    return '#D9EF8B'
    elif value < 4:
        return '#91CF60'
    else:
        return '#1A9850'

# Function to format the properties of each cell
def props(key):
    segment, like_hate = key
    value = residuals.loc[int(segment), like_hate]
    return {'color': residual_color(value)}

# Generate the mosaic plot with shading for standardized residuals
plt.figure(figsize=(24, 18))  # Increase the figure size for better
↪  readability
mosaic(contingency_table.stack(), gap=0.01, properties=props)
plt.xlabel('Segment Number', fontsize=18, labelpad=20)
plt.ylabel('Like/Hate McDonald\'s', fontsize=18, labelpad=20)
plt.title('Shaded Mosaic Plot for Segment Membership and I LIKE IT for
↪  the Fast Food Data Set', fontsize=22, pad=30)
plt.xticks(rotation=45, ha='right', fontsize=12)
plt.yticks(rotation=0, va='center', fontsize=12)

# Adding a color legend
```

```python
legend_labels = ['<-4', '-4:-2', '-2:0', '0:2', '2:4', '>4']
colors = ['#D73027', '#FC8D59', '#FEE08B', '#D9EF8B', '#91CF60', '#1A9850']
patches = [mpatches.Patch(color=colors[i], label=legend_labels[i]) for i in range(len(legend_labels))]
plt.legend(handles=patches, title="Standardized Residuals", loc="center left", bbox_to_anchor=(1, 0.5), fontsize=14, title_fontsize=16)
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.gca().set_facecolor('#f0f0f0')

plt.show()
```

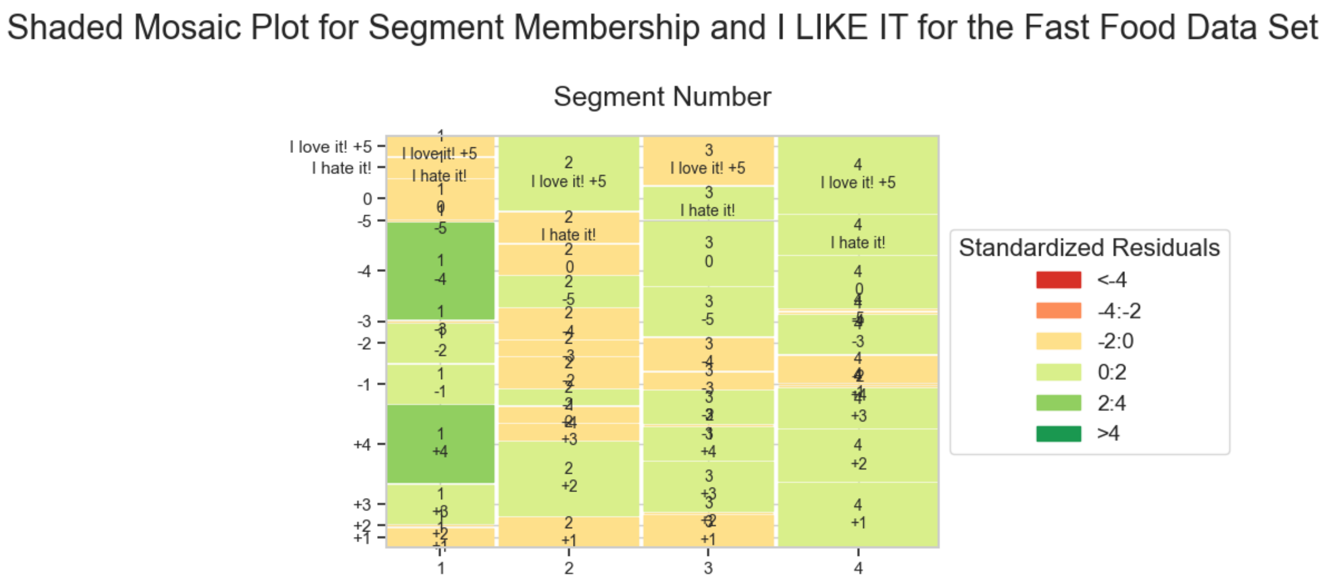



**Fig. A.14:** Shaded Mosaic Plot for Segment Membership and Like/Hate McDonald's

```python
# Sample data for demonstration (replace with actual data)
np.random.seed(42)
segments = np.random.choice([1, 2, 3, 4], size=100)  # Simulated segment membership
like_hate = np.random.choice(['I hate it!', '-5', '-4', '-3', '-2', '-1', '0', '+1', '+2', '+3', '+4', 'I love it! +5'], size=100)  # Simulated like/hate variable
gender = np.random.choice(['Male', 'Female'], size=100)  # Simulated gender variable
age = np.random.randint(18, 70, size=100)  # Simulated age variable

# Convert to DataFrame for easier manipulation
df = pd.DataFrame({'Segment': segments, 'Like': like_hate, 'Gender': gender, 'Age': age})

```

```
# Create a contingency table
gender_contingency_table = pd.crosstab(df['Segment'], df['Gender'])

# Perform chi-squared test to get expected frequencies
chi2, p, dof, expected = chi2_contingency(gender_contingency_table)

# Calculate standardized residuals
gender_residuals = (gender_contingency_table - expected) /
↪ np.sqrt(expected)

# Define color mapping for standardized residuals
def gender_residual_color(value):
    if value < -4:
        return '#D73027'
    elif value < -2:
        return '#FC8D59'
    elif value < 0:
        return '#FEE08B'
    elif value < 2:
        return '#D9EF8B'
    elif value < 4:
        return '#91CF60'
    else:
        return '#1A9850'

# Function to format the properties of each cell
def gender_props(key):
    segment, gender = key
    value = gender_residuals.loc[int(segment), gender]
    return {'color': gender_residual_color(value)}

# Generate the mosaic plot with shading for standardized residuals
plt.figure(figsize=(16, 12))
mosaic(gender_contingency_table.stack(), gap=0.01,
↪ properties=gender_props)
plt.xlabel('Segment Number', fontsize=18, labelpad=20)
plt.ylabel('Gender', fontsize=18, labelpad=20)
plt.title('Mosaic Plot for Gender Distribution across Segments',
↪ fontsize=22, pad=30)
plt.xticks(rotation=45, ha='right', fontsize=14)
plt.yticks(rotation=0, va='center', fontsize=14)

# Adding a color legend
legend_labels = ['<-4', '-4:-2', '-2:0', '0:2', '2:4', '>4']
colors = ['#D73027', '#FC8D59', '#FEE08B', '#D9EF8B', '#91CF60',
↪ '#1A9850']
patches = [mpatches.Patch(color=colors[i], label=legend_labels[i]) for i
↪ in range(len(legend_labels))]
```

```
plt.legend(handles=patches, title="Standardized Residuals", loc="center left", bbox_to_anchor=(1, 0.5), fontsize=14, title_fontsize=16)

# Adding grid lines and a background color
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.gca().set_facecolor('#f0f0f0')

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

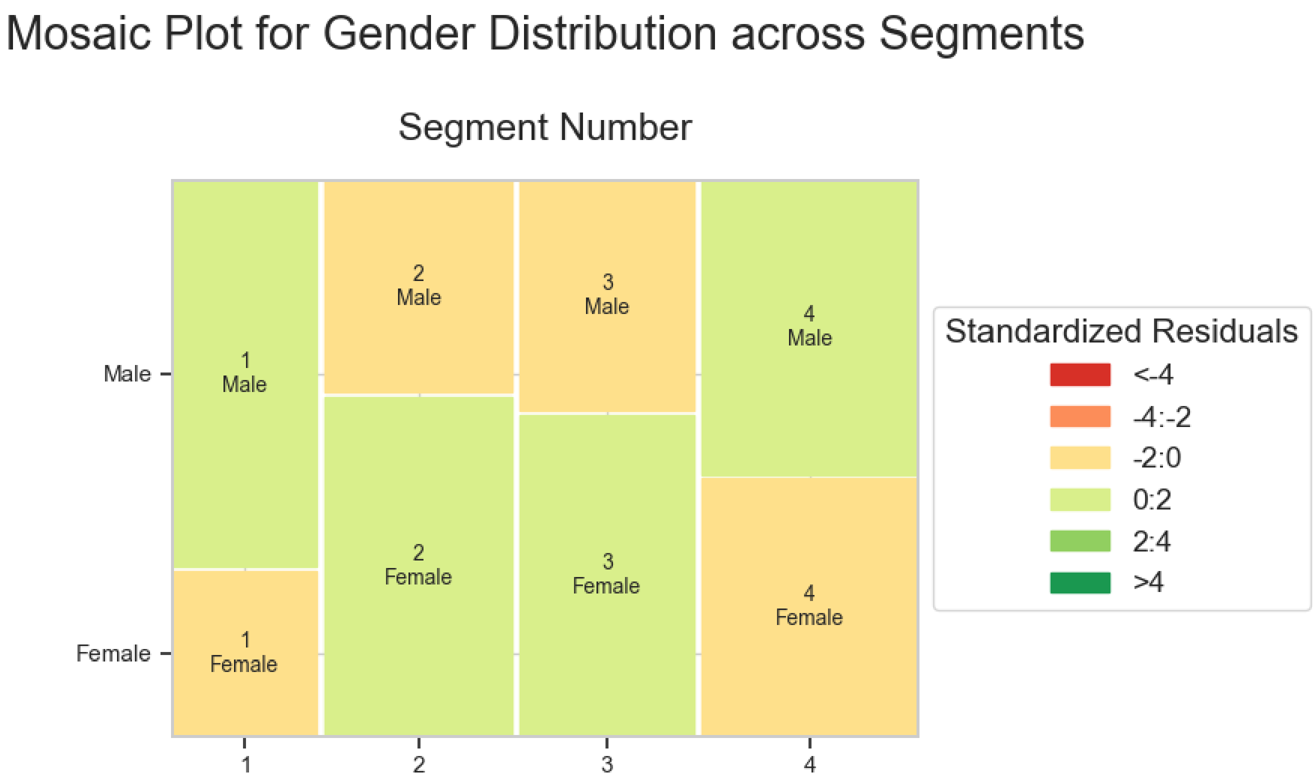



**Fig. A.15:** Mosaic Plot for Gender Distribution across Segments

```
plt.figure(figsize=(16, 10))
sns.boxplot(x='Segment', y='Age', data=df, notch=True)
plt.xlabel('Segment Number', fontsize=18, labelpad=20)
plt.ylabel('Age', fontsize=18, labelpad=20)
plt.title('Box-and-Whisker Plot for Age Distribution across Segments', fontsize=22, pad=30)
plt.xticks(rotation=45, ha='right', fontsize=12)
plt.yticks(fontsize=12)
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.gca().set_facecolor('#f0f0f0')
```

```
plt.show()
```

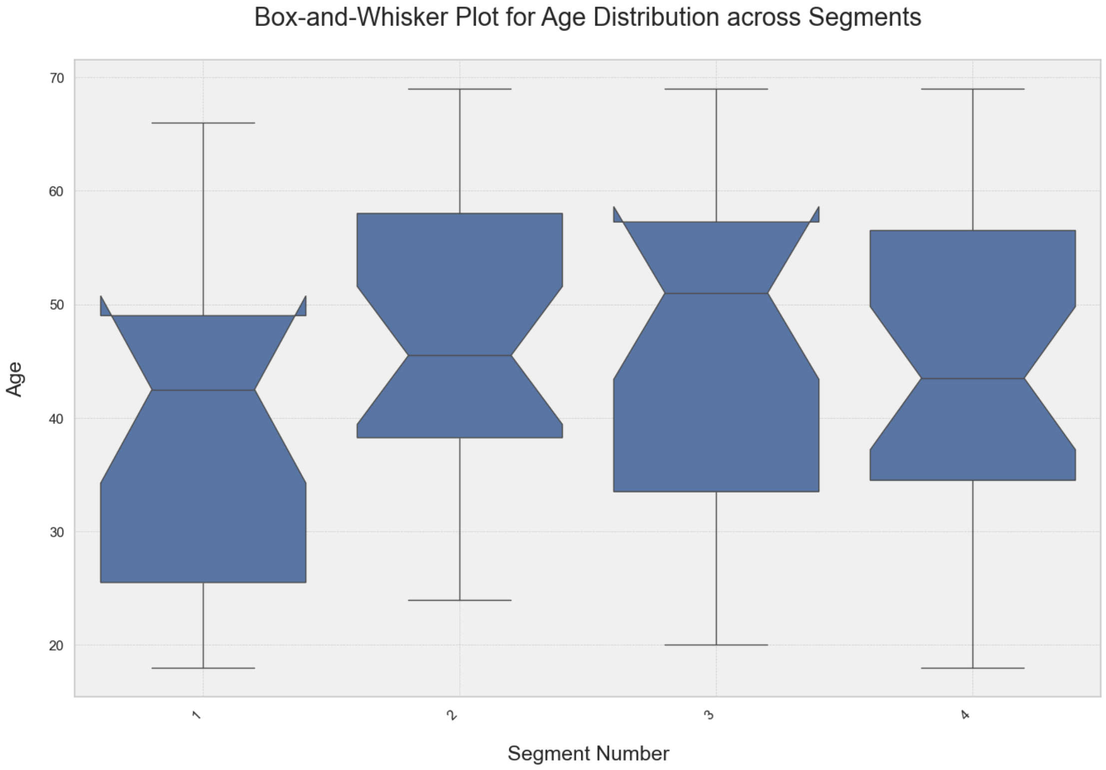



**Fig. A.16:** Box and Whisker Plot for Age

```
np.random.seed(42)
segments = np.random.choice([1, 2, 3, 4], size=100)  # Simulated segment
↪  membership
like_hate = np.random.choice(['I hate it!', '-5', '-4', '-3', '-2', '-1',
↪  '0', '+1', '+2', '+3', '+4', 'I love it! +5'], size=100)  # Simulated
↪  like/hate variable
gender = np.random.choice(['Male', 'Female'], size=100)  # Simulated
↪  gender variable
age = np.random.randint(18, 70, size=100)  # Simulated age variable
visit_frequency = np.random.choice(['Once a week', 'Once a month', 'More
↪  than once a month', 'Less than once a month'], size=100)

# Convert categorical variables to numerical codes
like_hate_map = {'I hate it!': -5, '-5': -5, '-4': -4, '-3': -3, '-2':
↪  -2, '-1': -1, '0': 0, '+1': 1, '+2': 2, '+3': 3, '+4': 4, 'I love it!
↪  +5': 5}
visit_frequency_map = {'Once a week': 1, 'More than once a month': 2,
↪  'Once a month': 3, 'Less than once a month': 4}
gender_map = {'Male': 0, 'Female': 1}
```

```python

# Apply mappings
df = pd.DataFrame({
    'Segment': segments,
    'Like': like_hate,
    'Gender': gender,
    'Age': age,
    'VisitFrequency': visit_frequency
})
df['Like.n'] = df['Like'].map(like_hate_map)
df['VisitFrequency.n'] = df['VisitFrequency'].map(visit_frequency_map)
df['Gender.n'] = df['Gender'].map(gender_map)
df['Segment3'] = (df['Segment'] == 3).astype(int)
X = df[['Like.n', 'Age', 'VisitFrequency.n', 'Gender.n']]
y = df['Segment3']

# Fit the decision tree classifier
tree = DecisionTreeClassifier()
tree.fit(X, y)

# Plot the decision tree
plt.figure(figsize=(20, 10))
plot_tree(tree, feature_names=['Like.n', 'Age', 'VisitFrequency.n',
↪   'Gender.n'], class_names=['Not Segment 3', 'Segment 3'], filled=True)
plt.title('Decision Tree for Segment 3 Membership Prediction')
plt.show()
```

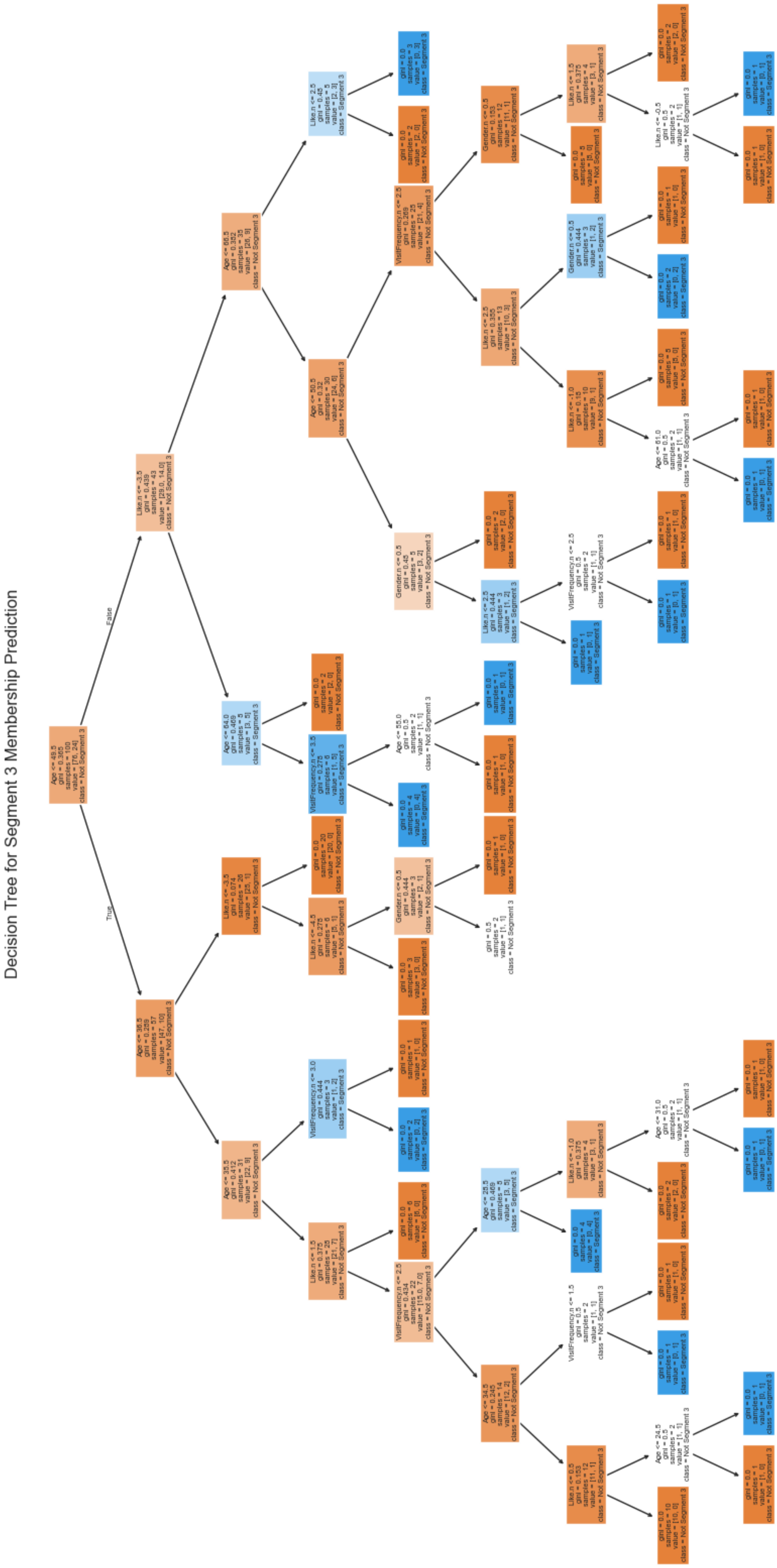



**Fig. A.17:** Decision Tree for segment 3 membership

## A.5 Github Repository

The GitHub link for the jupyter notebook can be found at here.

Thank you for reading this report. Best wishes.