Team Members :

Kaamna Kandpal (Team Leader)

Shubhman Kumar

Sai Karthik Reddy N S

Step1 : Deciding to Segment

Implications of Committing to Market Segmentation

Before investing time and resources in a market segmentation analysis, it is important to understand the implications of pursuing a market segmentation strategy. The key implication is that the organisation needs to commit to the segmentation strategy on the long term. Potentially required changes include the development of new products, the modification of existing products, changes in pricing and distribution channels used to sell the product, as well as all communications with the market. major implications of such a long-term organisational commitment, the decision to investigate the potential of a market segmentation strategy must be made at the highest executive level, and must be systematically and continuously communicated and reinforced at all organisational levels and across all organisational units.

Implementation Barriers

The first group of barriers relates to senior management. Lack of leadership, pro-active championing, commitment and involvement in the market segmentation process by senior leadership undermines the success of market segmentation. Senior management can also prevent market segmentation to be successfully implemented by not making enough resources available, either for the initial market segmentation analysis itself, or for the long-term implementation of a market segmentation strategy.

A second group of barriers relates to organisational culture. Lack of market or consumer orientation, resistance to change and new ideas, lack of creative thinking, bad communication and lack of sharing of information and insights across organisational units, short-term thinking, unwillingness to make changes and office politics have been identified as preventing the successful implementation of market segmentation.

Another potential problem is lack of training. If senior management and the team tasked with segmentation do not understand the very foundations of market segmentation, or if they are unaware of the consequences of pursuing such a strategy, the attempt of introducing market segmentation is likely to fail.

Step 2: Specifying the Ideal Target Segment

Segment Evaluation Criteria

The third layer of market segmentation analysis depends primarily on user input. It is important to understand that – for a market segmentation analysis to produce results that are useful to an organisation – user input cannot be limited to either a briefing at the start of the process, or the development of a marketing mix at the end. After having committed to investigating the value of a segmentation strategy in Step 1, the organisation has to make a major contribution to market segmentation analysis in Step 2. While this contribution is conceptual in nature, it guides many of the following steps, most critically Step 3 (data collection) and Step 8. One set of evaluation criteria can be referred to as knock-out criteria. These criteria are the essential, non-negotiable features of segments that the organisation would consider targeting. The second set of evaluation criteria can be referred to as attractiveness criteria.

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.

- **The segment must be homogeneous;** members of the segment must be similar to one another.
- **The segment must be distinct;** members of the segment must be distinctly different from members of other segments.
- **The segment must be large enough;** the segment must contain enough consumers to make it worthwhile to spend extra money on customising the marketing mix for them.
- **The segment must be matching the strengths of the organisation;** the organisation must have the capability to satisfy segment members' needs.
- **Members of the segment must be identifiable;** it must be possible to spot them in the marketplace.
- **The segment must be reachable;** there has to be a way to get in touch with members of the segment in order to make the customised marketing mix accessible to them.

Attractiveness Criteria

Attractiveness criteria are not binary in nature. Segments are not assessed as either complying or not complying with attractiveness criteria. Rather, each market segment is rated; it can be more or less attractive with respect to a specific criterion.

Implementing a Structured Process

The most popular structured approach for evaluating market segments in view of selecting them as target markets is the use of a segment evaluation plot. The segment attractiveness and organisational competitiveness values are determined by the segmentation team. This is necessary because there is no standard set of criteria that could be used by all organisations.

Back to the segment evaluation plot. Obviously, the segment evaluation plot cannot be completed in Step 2 of the market segmentation analysis because – at this point – no segments are available to assess yet. But there is a huge benefit in selecting the attractiveness criteria for market segments at this early stage in the process: knowing precisely what it is about market segments that matters to the organisation ensures that all of this information is captured when collecting data (Step 3). It also makes the task of selecting a target segment in Step 8 much easier because the groundwork is laid before the actual segments.

At the end of this step, the market segmentation team should have a list of approximately six segment attractiveness criteria. Each of these criteria should have a weight attached to it to indicate how important it is to the organisation compared to the other criteria.

Step 3: Collecting Data

This step provide data collection process for market segmentation purposes, covering both primary research (survey studies, experimental studies) and secondary data sources (internal sources).

This study looks at data about Australian travel motives. When checking the data, it was found that the variables for Gender and Age were already clean and didn't need fixing. However, the variable for Income2 had categories that weren't in order. This happened because of how the data was read into the program R. R often converts non-number information into categories called factors, and these factors are usually sorted alphabetically.

To fix the order of the income categories, the study copied the column to a new variable called inc2. Then, it found the correct order for the categories and made them into an ordered list. Before replacing the original data, the study double-checked by comparing the original and the reordered versions.

The comparison showed that all the values matched up, meaning no mistakes were made during the reordering. So, it was safe to replace the original column with the reordered one. This process ensures the data is sorted correctly, making it easier to analyze. Before diving into analyzing data, it's crucial to clean it up first. This means

making sure all the recorded values are correct and that labels for different categories are consistent.

For things like age, we know there's a reasonable range of values, like between 0 and 110 years. If we find any values outside this range, it could mean there were mistakes during data collection or entry.

Similarly, for categorical variables like gender, we expect to see only two options: female and male. If we see any other values, they're not supposed to be there and need fixing during the cleaning process.

This section talks about the importance of reproducibility in data analysis, especially in cleaning and exploring data using R code. It emphasizes using functions like `save()` and `load()` to ensure reproducibility. In descriptive analysis, it mentions using tools like `summary()` for numeric data and graphical methods like histograms, boxplots, and scatter plots. It explains how to create histograms in R using the `histogram()` function from the lattice package, emphasizing the importance of binning and exploring different bin widths.

STEP 4: Exploring Data

Data Exploration helps:

- Identify the measurement levels of the variables .
- Investigate the univariate distributions of each of the variables .
- Assess dependency structures between variables .

Data need to be pre-processed and prepared so it can be used as input for different segmentation algorithms. Results from the data exploration stage provide insights into the suitability of different segmentation methods for extracting market segments.

Data Cleaning : Before analysing data, ensure it's clean by verifying accurate values and consistent labels. Check for plausible ranges in metric variables and permissible values in categorical ones, correcting any discrepancies during the cleaning process.

Descriptive Analysis : Understanding data prevents misinterpretation in complex analyses. Descriptive statistics and graphical representations, such as histograms and bar plots, provide insights into numeric and categorical variables, aiding in data interpretation and visualization.

Pre- Processing : Pre-processing of categorical variables commonly involves either merging levels to simplify categories or converting them into numeric forms, depending on the context. This helps manage complexity and enhance analysis, as seen in the example of income categories simplification.

The range of values in segmentation variables impacts their influence in distance-based segmentation methods. Standardizing variables, such as transforming them to a common scale, helps balance their influence on segmentation results, particularly when variables have different ranges of values, ensuring fair weighting across all variables.

PCA (Principal Component Analysis) : PCA transforms multivariate data into uncorrelated principal components, ordered by importance. It maintains relative positions of observations while offering a new perspective on the data. Typically used for dimensionality reduction in visualization, focusing on the first few principal components capturing most variation.

The summary covers the importance of pre-processing data before collection, favoring binary answer options for simplicity and reliability. It explains converting variables to binary format for ease of analysis and discusses standardizing variables to balance their influence.

This excerpt discusses data exploration techniques, focusing on standardization and principal components analysis (PCA). It explains how to standardize data using R's scale() function and introduces PCA as a method to transform multivariate data into uncorrelated components, allowing for visualization and dimensionality reduction. It also provides a code example of performing PCA using the prcomp() function in R.

This is a summary of a Principal Components Analysis (PCA) conducted on a dataset with 20 variables. The analysis identifies the importance of each principal component and shows a rotation matrix indicating how the original variables contribute to each component. The summary discusses the significance of the first three principal components and their interpretation in terms of the original variables. Additionally, it mentions how to obtain further information using the summary function.

This passage discusses the interpretation of principal component analysis (PCA) results, focusing on the standard deviation, proportion of variance explained, and cumulative proportion of explained variance for each principal component. It highlights that the initial principal components explain a significant portion of the variance, indicating the importance of all original variables. In the context of Australian travel motives data, it mentions using the second and third principal components for creating a perceptual map due to their differentiated loading pattern of motives. It concludes with R code for plotting the data in two-dimensional space using these principal components.

This section introduces principal components 2 and 3 for the Australian travel motives dataset, highlighting the simultaneous occurrence of interests in the lifestyle of local people and cultural offers, as well as the contrast between nature-oriented motives and luxury, excitement, and indifference towards prices. It discusses the use of principal

components analysis for reducing the number of segmentation variables in consumer data, but warns about the limitations and problems associated with this approach, which will be detailed further in Section.

This section discusses Step 4 of exploring data for market segmentation. It advises against using a subset of principal components for segmentation but suggests using principal components analysis to identify highly correlated variables. It provides a checklist for tasks like exploring data inconsistencies, cleaning data, checking segmentation variable numbers, and ensuring variables are uncorrelated before passing the data to Step 5 for segmentation extraction. References include studies on response styles in marketing research and market segmentation reviews.

This passage discusses the importance of profiling in market segmentation, particularly in data-driven approaches. It emphasizes the need to identify key characteristics of market segments to interpret segmentation results correctly and make strategic marketing decisions.

STEP 5: Extracting Segments

Grouping Consumers:

- Market segmentation analysis is exploratory, driven by unstructured consumer data.
- Results depend on the chosen extraction algorithm and assumptions made about segment structure.
- No single algorithm is universally superior; each has its own tendencies and limitations.
- Common extraction methods include distance-based and model-based approaches.
- Some methods incorporate variable selection during segmentation.
- Treatment of binary segmentation variables depends on the analysis objective: symmetrically or asymmetrically.
- Distance-based methods can accommodate asymmetry in binary variables, extracting segments based on shared attributes.

Distance Based Methods:

<u>Distance Measures:</u>

Distance measures are crucial in market segmentation analysis, with Euclidean and Manhattan distances being common choices.

Euclidean Distance: Direct "straight-line" distance between points, suitable for equally scaled dimensions.

Manhattan Distance: Considers movement along grid-like streets, useful for varied scales or dimensions.

Asymmetric binary distance highlights shared uncommon activities, treating 0s and 1s differently. Understanding distance measures is vital, for instance, larger numerical values in dimensions can dominate distance calculations. Selecting the right measure ensures segmentation outcomes accurately reflect data patterns.

Hierarchical Methods:

Hierarchical clustering methods mimic human-like grouping of data into segments, ranging from one large segment to individual segments for each consumer, with divisive and agglomerative approaches representing two ends of the spectrum.

Divisive Method: Starts with the entire dataset and splits it iteratively into smaller segments until each consumer forms their own segment.

Agglomerative Method: Begins with each consumer as a singleton segment, gradually merging the closest segments until the dataset forms one large segment.

Both methods result in a sequence of nested partitions, ranging from one segment to n segments. The Lance-Williams framework unifies agglomerative clustering algorithms, ensuring deterministic outcomes with no randomness in the process.

Partitioning Methods :

Hierarchical clustering is ideal for small datasets but impractical for larger ones due to dendrogram complexity and memory limitations. For datasets over 1000 observations, single partition clustering is preferred, as it computes distances only between observations and segment centers, reducing computational load. Partitioning algorithms efficiently optimize for a specific number of segments, bypassing dendrogram construction and heuristic cutting.

- **k-Means**
- **k-Centroid Clustering**
- **self-organising maps**
- **Neural Networks**

Model-Based Methods:

Traditional vs. Modern Approaches: While distance-based methods have a long history in market segmentation, recent attention has shifted towards model-based methods, notably mixture methodologies.

Impact of Model-based Methods: Wedel and Kamakura predict that mixture models will have a significant influence on both academia and practice, alongside conjoint analysis, in shaping market segmentation analysis.

Pragmatic Perspective: Model-based methods are viewed as additional tools for segment extraction, offering a distinct approach to exploring data compared to traditional distance based methods.

Assumptions of Model-based Methods: Unlike distance-based approaches, model-based methods assume segment characteristics and sizes, refining them based on empirical data rather than relying on similarities or distances.

Finite Mixture Models: Model-based methods, such as finite mixture models, establish a fixed number of segments and adjust segment characteristics based on the data, providing a structured yet flexible approach to segmentation analysis.

Finite Mixtures of Distributions :

Model-based clustering fits a distribution to the dependent variable only, while finite mixtures of distributions use the same segmentation variables without incorporating additional consumer information.

- **Normal Distributions**
- **Binary Distributions**

Finite Mixtures of Regressions :

Finite mixtures of distributions resemble distance-based clustering methods but offer varied outcomes, while finite mixtures of regression models provide a distinct approach to market segmentation analysis.

Extensions and Variations:

Finite mixture models provide versatility by accommodating diverse data types and mitigating response style effects. They strike a balance between continuous distribution and distinct segment modelling, allowing for both while permitting variation within segments. These models are instrumental in clustering time series data and capturing changes in consumer behaviour over time, especially in tracking brand choices and shifts in customer value systems. They incorporate descriptor variables to capture variations in segment sizes, with concomitant variables enabling their implementation using packages like flexmix.

Algorithms with Integrated Variable Selection:

Algorithms often assume all segmentation variables contribute equally, but redundant or noisy variables may exist. Steinley and Brusco's filtering approach retains only relevant variables above a threshold, while for binary data, biclustering and VSBD help select suitable variables during segment extraction. Factor-cluster analysis compresses variables into factors before segmentation, offering a streamlined approach.

Bi-clustering Algorithms :

- Bi-clustering handles both consumers and variables together, useful for various data types.
- Originally developed for genetic data challenges, bi-clustering electively manages noisy variables.
- Popular algorithms vary in defining bi-clusters, focusing on identifying consumer groups with shared variables.
- Crucial for tasks like market segmentation, bi-clustering aims to find large consumer groups with common activities.

It's advantage and where to use :

Bi-clustering is advantageous for market segmentation with numerous variables, avoiding suboptimal consumer groupings common in standard techniques. It doesn't require data transformation, unlike methods like principal components analysis, which may introduce bias by altering segmentation variables. Particularly adept at capturing niche markets by identifying identical patterns among consumer groups and variables.

Control parameters can be adjusted to yield smaller, more homogeneous segments for niche market analysis or larger, less homogeneous segments for broader insights. Bi-clustering selects groups of similar consumers, leaving ungrouped individuals who don't fit into any segment.

Variable Selection Procedure for Clustering Binary Data:

- The Variable Selection Procedure for Clustering Binary Data Sets (VSBD) aims to address irrelevant variables in clustering by identifying and removing masking variables.
- VSBD utilizes the k-means algorithm and assesses the within-cluster sum-of-squares criterion to identify the best subset of variables.
- Additional variables are incrementally added based on their impact on the within-cluster sumof-squares until a predefined threshold is reached.
- The number of segments (k) needs to be specified beforehand, with the procedure recommending the Ratkowsky and Lance index for selecting k.

Variable Reduction: Factor-Cluster Analysis

- Factor-cluster analysis involves a two-step process in market segmentation, where segmentation variables are first factor analysed, and then factor scores are utilized to extract market segments.
- This approach is justified when empirical data stems from validated psychological test batteries, like IQ tests, designed with variables loading onto factors.

- However, using factor scores should be simultaneous with group extraction or provided separately, rather than derived in a data-driven manner.
- In practice, factor-cluster analysis is often employed when the number of original segmentation variables is excessive.
- A rule of thumb suggests a sample size should be at least 100 times the number of segmentation variables, posing challenges as many studies use fewer consumers than recommended.

Running factor-cluster analysis to address the issue of an excessive number of segmentation variables relative to sample size lacks conceptual justification and entails significant drawbacks : -

- Factor analysing data leads to a substantial loss of information .
- Factor analysis transforms data.
- Factors-cluster results are more difficult to interpret.

Data Structure Analysis:

- Market segmentation is exploratory, making traditional validation targeting a clear optimality criterion impractical.
- Validation in segmentation typically focuses on assessing reliability or stability across repeated calculations.
- Stability-based data structure analysis involves modifying the data or algorithm to assess the consistency of segmentation solutions.
- Data structure analysis offers insights into the presence of distinct market segments and guides methodological decisions.
- Approaches to data structure analysis include cluster indices, gorge plots, global stability analysis, and segment level stability analysis.

Data structure analysis in market segmentation is crucial for assessing the reliability and stability of segmentation solutions. Traditional validation methods, which target clear optimality criteria, are often impractical due to the exploratory nature of segmentation. Instead, stability-based data structure analysis focuses on assessing the consistency of segmentation solutions across repeated calculations. This involves modifying the data or algorithm to evaluate the reliability of the results. By providing insights into the presence of distinct market segments and guiding methodological decisions, data structure analysis plays a vital role in segmentation. Various approaches, such as cluster indices, gorge plots, global stability analysis, and segment level stability analysis, are used to conduct data structure analysis and gain a deeper understanding of the underlying data properties.

We discuss four different approaches to data structure analysis:

- cluster indices

- gorge plots
- global stability analysis
- segment level stability analysis

Cluster indices, gorge plots, global stability, and segment-level stability methods assess the reliability of market segmentation solutions by measuring cluster quality, visualizing cluster stability, and evaluating consistency across data variations and algorithm modifications.

Step 6: Profiling segments

Profiling is essential for data-driven market segmentation to understand the defining characteristics of resulting segments. Traditional segmentation may have predefined profiles, but data-driven segmentation requires analysis to uncover segment characteristics. It distinguishes between data-driven segmentation and commonsense segmentation, emphasizing the need for profiling in data-driven approaches. Traditional approaches to profiling involve presenting segment characteristics in tables, which can be complex and difficult to interpret. Visualizations, such as segment profile plots and segment separation plots, are introduced as effective alternatives to traditional tabular presentations. These visualizations make it easier to interpret segmentation results and assess segment separation. For example, consider a dataset containing customer information such as age, income, and spending habits. After performing segmentation analysis, we might identify three distinct segments:

Segment 1: Young adults with moderate income and high spending on technology products.

Segment 2: Middle-aged individuals with high income and moderate spending on luxury items.

Segment 3: Seniors with low to moderate income and conservative spending habits.

Profiling these segments helps us understand the specific needs, preferences, and behaviours of each group. This insight can then inform marketing campaigns tailored to resonate with each segment effectively.

This passage discusses the challenges marketing managers face in interpreting market segmentation results, as well as the traditional and graphical approaches to segment profiling. It highlights the difficulties in presenting segmentation results effectively and provides insights into how data-driven segmentation solutions are typically presented to users. Additionally, it introduces the Australian vacation motives data set and discusses the presentation of segmentation variables by segment for analysis.

The passage discusses the challenges in presenting market segmentation results to marketing managers, highlighting common complaints and ineffective methods. It also

introduces traditional and graphical statistical approaches to segment profiling, emphasizing the need for clarity and simplicity in presenting complex data. Additionally, it provides insights into how data-driven segmentation solutions are typically presented to users and the difficulties in interpreting them. Finally, it explains how to analyze segmentation variables to identify the defining characteristics of market segments.

Table 8.2 presents six segments derived from the Australian travel motives dataset using the neural gas algorithm, with percentages indicating the agreement to various travel motives. Interpreting the segments reveals distinct characteristics, such as Segment 2 being motivated by relaxation and budget-consciousness. Comparing segments requires analyzing numerous percentages, making the process complex and time-consuming. Additionally, multiple segmentation solutions further compound the analysis, requiring extensive comparisons.

The passage discusses the challenges of comparing multiple segmentation solutions, highlighting the daunting task of analyzing numerous pairs of numbers. It also critiques the use of statistical significance tests due to the nature of segment creation. Furthermore, it emphasizes the importance of visualizations in presenting market segmentation solutions, noting their effectiveness in providing insights and facilitating interpretation. Several examples of prior use of visualizations in segmentation analysis are cited, underscoring their utility in assessing segment profiles and selecting the most suitable solution.

The passage explains the process of creating segment profile plots to understand the defining characteristics of market segments. It suggests rearranging variables for better visualization and discusses hierarchical clustering to order variables. The segment profile plot displays segment centroids and compares them to the overall sample, with marker variables highlighted in colour for easier interpretation.

The passage explains the effectiveness of segment profile plots compared to traditional tables in interpreting market segmentation results. It discusses an eye tracking study showing that participants processed information more efficiently with graphical representations. Additionally, it highlights the importance of investing time in creating well-designed visualizations for better interpretation, which can ultimately lead to more informed strategic decisions. Furthermore, it introduces segment separation plots as a tool to visualize the overlap of segments in the data space.

Step 7: Describing Segments

Describing segments involves using additional information about segment members, such as demographic, psychographic, socio-economic variables, media exposure, and product attitudes. This step helps in gaining detailed insight into the nature of segments

and is essential for developing a customized marketing mix tailored to each segment's characteristics.

For instance, in a data-driven market segmentation analysis using the Australian travel motives dataset, profiling involves investigating differences between segments based on travel motives, while segment description includes additional variables like age, gender, past travel behaviour, media use, and expenditure patterns during vacations.

We use Visualisations to Describe Market Segments- It comprises distinct approaches of variables:

1- Nominal and Ordinal Descriptor Variables -variables (such as gender, level of education, country of origin).
2- 2- Metric Descriptor Variables - (such as age, number of nights at the tourist destinations, money spent on accommodation).

For testing for Segment Differences in Descriptor Variables: We conduct statistical tests (e.g., ANOVA) to determine if there are significant differences in variables like income or usage frequency between segments.

Predicting Segments from Descriptor Variables: involves predictive techniques such as

1- Binary Logistic Regression-Binary logistic regression is a statistical method used to predict the likelihood of a binary outcome, such as whether a customer belongs to a particular segment or not.
2- Multinomial Logistic Regression-Multinomial logistic regression extends binary logistic regression to predict outcomes with more than two categories, making it suitable for predicting market segments with multiple categories.
3- Tree-Based Methods- such as decision trees or random forests, are predictive modeling techniques that use a tree-like structure to represent and predict outcomes based on input variables.

Step 7 of market segmentation analysis, which involves describing segments using additional information beyond the variables used for segmentation. It compares this step to getting to know a potential spouse before marriage. The chapter highlights the importance of using descriptor variables such as demographics, psychographics, and past behaviour to provide detailed insight into market segments.

Step 7 of market segmentation, which involves describing segments using additional information beyond the variables used for segmentation. It emphasizes the importance of understanding descriptor variables such as demographics and behaviour to develop a customized marketing mix. The chapter discusses the use of visualizations to simplify the interpretation of differences between market segments and avoid over-interpretation of insignificant differences.

Discusses the use of visualizations to describe market segments, focusing on cross-tabulations of descriptor variables. It demonstrates how to generate cross-tabulations in R using the Australian travel motives dataset. Visualizations such as stacked bar charts and mosaic plots are used to illustrate differences between segments, providing insights into demographic characteristics across market segments.

Step 7, demonstrates visual methods for comparing segment membership and gender using the Australian travel motives dataset. It contrasts stacked bar charts with mosaic plots, showing how mosaic plots provide a clearer representation of proportions across segments. Mosaic plots can handle tables with multiple descriptor variables and incorporate inferential statistics to aid interpretation.

Chapter 9.2 explains the use of mosaic plots to visualize differences between market segments based on descriptor variables. It describes how mosaic plots incorporate inferential statistics to highlight differences in observed and expected frequencies, using color-coded shading to represent standardized differences. The example provided shows that gender distribution does not significantly differ across the six market segments in the Australian travel motives dataset.

Step 7 illustrates the use of mosaic plots to visualize the association between segment membership and income in the Australian travel motives dataset. It shows how different income levels are distributed across segments, highlighting those members of segment 4, interested in cultural offers, tend to earn higher incomes. Additionally, it demonstrates a strong association between travel motives and the stated moral obligation to protect the environment.

Chapter 9.2 showcases the use of mosaic plots to visualize the relationship between segment membership and moral obligation to protect the environment in the Australian travel motives dataset. It demonstrates how different segments vary in their levels of stated moral obligation, with segments motivated by nature showing a stronger association with high moral obligation.

Chapter 9.2.2 demonstrates the visualization of differences between market segments using metric descriptor variables. It explores methods such as conditional plots and parallel box-and-whisker plots to compare age distribution and moral obligation scores across segments in the Australian travel motives dataset. These visualizations aid in understanding segment differences, although statistical testing may be necessary to confirm observed disparities.

Chapter 9.2.2 showcases visualizations for describing market segments using metric descriptor variables. It presents histograms of age distribution and parallel box-and-whisker plots of moral obligation scores across segments in the Australian travel motives dataset. These visualizations help identify differences between segments, with statistical testing providing further confirmation of disparities.

Chapter 9.2.2 provides visual representations of metric descriptor variables in the Australian travel motives dataset. It includes histograms displaying the distribution of moral obligation scores across segments and parallel box-and-whisker plots illustrating age differences among segments. These visuals aid in understanding the characteristics of each segment.

Chapter 9.2.2 further explores visual representations of metric descriptor variables, such as moral obligation, in the Australian travel motives dataset. It includes a modified version of the segment level stability across solutions (SLSA) plot, where node colors represent moral obligation levels across segments. This analysis helps identify consistent patterns in segment characteristics across different segmentation solutions.

Chapter 9.3 delves into testing for differences in descriptor variables across market segments. It highlights the use of statistical tests, such as the $\chi 2$-test, to assess associations between nominal or ordinal variables and segment membership. Additionally, it introduces a modified segment level stability across solutions (SLSA) plot, where node colours represent mean moral obligation values across segments.

Chapter 9.3 explores testing for differences in descriptor variables across market segments. It discusses the use of statistical tests, such as the $\chi 2$-test and Analysis of Variance (ANOVA), to assess associations and significant differences in variables like gender distribution and moral obligation across segments. The chapter emphasizes interpreting p-values and visualizations like mosaic plots and parallel boxplots to understand segment differences.

An analysis of variance (ANOVA) was conducted to compare moral obligation across different market segments. The results show significant differences among segments, indicating that at least two segments differ in their mean moral obligation to protect the environment.

Pairwise t-tests were then performed to identify which segments differ from each other in terms of moral obligation. Pairwise t-tests were performed to compare moral obligation across different segments. The p-values were adjusted using the Holm method to control for multiple testing. The results indicate significant differences between some segments, with segments 5 and 6 feeling more morally obliged to protect the environment compared to segments 1, 2, 3, and 4. Adjustment of p-values is crucial to maintain the correct error rate when conducting multiple tests. Additionally, Tukey's honest significant differences can be plotted as an alternative to pairwise t-tests.

Tukey's honest significant differences were plotted to compare moral obligation between segments. Segments 1, 2, 3, and 4 show no significant differences among each other, nor do segments 5 and 6. However, segments 5 and 6 exhibit significantly higher moral obligation compared to other segments, except for a non-significant difference between segments 4 and 5.

Step 8: The Targeting Decision

It's a phase of project where several segments are selected for targeting. Here segments are referred to as the places where the project is functioning as per requirements and generating good revenue to the company at the same time.

Step 8 helps to finalise the market segments where the project can be deployed.

Where multiple segments are extracted in step 5 where consumers behaviours and features are known by the organisation.

A segment consists similarities between consumers, difference between other segments and number, size of segments among people.

Example – Riots and group of people requires same thing but for different usage.

At step 6 it is categorised and given a proper profile so it can be described in step 7.

Step 8 is for finalising the target segments which can be 1 or more.

Step 2 consists a process called segment attractiveness criteria which helps to determine which segment can be consistent.

In both steps 6 and 7 declare that weather the segment have needs of satisfying the organisation or not.

Step - 8.1

Finalise the market segments

The first step in step 8 is to ensure that all market segments under consideration to be selected as target markets. They should qualify knock out criteria.

Once this is done,

> The attractiveness of this segments and relatively organisational competitiveness for those segments should be evaluated.

Segmentation team should have several questions which can be classified into two categories:

1. Which market segments are our target and what segments are we committed to?
2. Other organizations offering same product and what is the priority of product from each segments? How likely the other organisations would be chosen? Reason why the each segments commit to us? Whats unique that we provide the other organisations don't?

Step – 8.2

Market segment evaluation

To evaluate the market segment we use decision matrix to visualize relative segment attractiveness and relative organizational competitiveness. **It makes easier for organisation to evaluate alternativeness market segments and select one or a small number of segments for targeting .**

As the matrix is 2Dimensional one of the dimensions is segment attractiveness and relative organisational competitiveness .

The dimensions define ,

How attractive is the segment to us?

How attractive are we to the segment?

We can plot it like a standard clustering plot where we can use scatter plot to represent those clusters where each cluster can be appeared into circles.

Size of circle can determine which can be turnover and loyalty if satisfied.

Segmentation team needs to assign a value of attractiveness for each segments so they can be a crucial asset for the organisation.

In plotting segment evaluation plot the each location of the segment is assigned by calculating the weight of segment attractiveness criterion with value of segment attractiveness criterion for each market segment.

Plot location of each market segment = **weight of segment attractiveness * segment attractiveness criterion**

Same procedure is followed for relative organizational competitiveness.

Then data for segment evaluation plot is obtained

- for how attractive is the segment to us?
- How attractive we are to the segment?

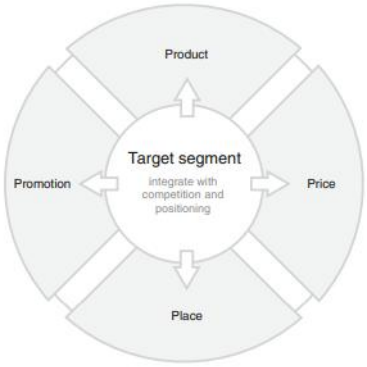
Then a bubble plot is plotted where the bubble size defines the attractiveness of the segment.

Big bubble says its highly attractive and smaller one says its less attractive.

Step 9 :Customising the market mix

Implications for marketing mix decisions

Segmentation targeting positioning approach is useful because it ensures that segmentation is not as independent from other strategic decisions. This process is a sequential process starts with marketing segmentation followed by targeting and positioning.



To maximise benefits in marketing strategy its important to customise the marketing mix to the target segment.

The customising market mix revolves around the 4 P's:

Product:

Product is a key decision and organisation develop the product according view of customer needs. Other marketing mix decisions that fall under the product dimension are naming the product, packaging it, offering or not offering warranties, and after sales support services.

Price :

Typical decisions an organisation needs to make when developing the price dimension of the marketing mix include setting the price for a product, and deciding on discounts to be offered.

Place :

The key decision relating to the place dimension of the marketing mix is how to distribute the product to the customers. This includes answering questions such as should the product be made available for purchase online or offline only or both; should the manufacturer sell directly to customers; or should a wholesaler or a retailer or both be used.

Promotion :

Typical promotion decisions that need to be made when designing a marketing mix include developing an advertising message that will resonate with the target market, and identifying the most effective way of communicating this message. Other tools in the promotion category of the marketing mix include public relations, personal selling, and sponsorship.

Code conversion

Step 4

```
Import pandas as pd

df=pd.read_excel('McDonalds.xlsx')

print(df.columns)

df.head()
```

```
MD_X=df.iloc[:,0:11].vlaues

MD_X=(MD_X=='YES').astype(int)

Result=np.round(np.mean(MD_X,axis=0),2)

Print(Result)
```

```
From sklearn.decomposition import PCA

Pca=PCA()

MD_PCA=pca.fit_transform(MD_X)

print("Explained variance ratio:", pca.explained_variance_ratio_)
print("Singular values:", pca.singular_values_)

print(np.round(MD_pca,1))
```

```
import matplotlib.pyplot as plt

from mpl_toolkits.mplot3d import Axes3D

# 3D plot

plt.subplot(1, 2, 2, projection='3d')

plt.scatter(MD_pca[:, 0], MD_pca[:, 1], MD_pca[:, 2], color='grey')

plt.title('PCA Plot (3D)')

plt.xlabel('PC1')
```

```
plt.ylabel('PC2')
plt.zlabel('PC3')
plt.show()
#not accurately as shown in the figure but explains same thing
```

Step 5

```
from sklearn.cluster import KMeans
wcss=[]
for k in range(2, 9):
    # Perform K-means clustering
    kmeans = KMeans(n_clusters=k, n_init=10, random_state=0)
    cluster_assignments = kmeans.fit_predict(MD_x)
    wcss.append(kmeans.intertia_)
    # Save clustering results
    cluster_results[k] = cluster_assignments
#relabeling clusters based on segmentation criteria
relabel_mapping = {}
for k, assignments in cluster_results.items():
    unique_labels = np.unique(assignments)
    relabel_mapping[k] = {old_label: new_label for new_label, old_label in
enumerate(unique_labels)}
```

```
#plotting wcss to know number of optimal segments
plt.plot(range(2, 9), wcss, marker='o')
plt.title('Within-Cluster Sum of Squares vs Number of Clusters')
plt.xlabel('Number of Clusters')
plt.ylabel('Within-Cluster Sum of Squares (WCSS)')
plt.xticks(range(2, 9))
```

```
plt.grid(True)
plt.show()
```

```
from sklearn.utils import resample
# Assuming MD_x is wer data matrix
bootstrap_results = {}

# Range of cluster numbers
for k in range(2, 9):
    bootstrap_cluster_assignments = []
    for _ in range(100):  # Number of bootstrap samples
        # Resample the data with replacement
        bootstrap_sample = resample(MD_x)
  kmeans = KMeans(n_clusters=k, n_init=10, random_state=0)
        cluster_assignments = kmeans.fit_predict(bootstrap_sample)
        # Save clustering results
        bootstrap_cluster_assignments.append(cluster_assignments)
    # Save bootstrap results
    bootstrap_results[k] = bootstrap_cluster_assignments
```

```
from sklearn.metrics import adjusted_rand_score

adjusted_rand_indices = {}
for k, cluster_assignments_list in bootstrap_results.items():
    aris = [] #Adjusted Rand Indices
    for cluster_assignments in cluster_assignments_list:
```

```
        # Compute adjusted Rand index
        true_labels = np.random.choice(range(2), len(cluster_assignments))  # Random labels for demonstration
        aris.append(adjusted_rand_score(true_labels, cluster_assignments))
    adjusted_rand_indices[k] = aris

# Plot adjusted Rand index against the number of segments
plt.errorbar(range(2, 9), [np.mean(adjusted_rand_indices[k]) for k in range(2, 9)], yerr=[np.std(adjusted_rand_indices[k]) for k in range(2, 9)], fmt='-o')
plt.title('Adjusted Rand Index vs Number of Segments')
plt.xlabel('Number of Segments')
plt.ylabel('Adjusted Rand Index')
plt.xticks(range(2, 9))
plt.grid(True)
plt.show()
```

```
# MD_kmeans_4  having 4 custer_result
MD_kmeans_4 = cluster_results[4]

# Create a histogram of cluster assignments
plt.hist(MD_kmeans_4, bins=range(5), align='left', edgecolor='black')
plt.xlabel('Cluster Assignment')
plt.ylabel('Frequency')
plt.xlim(0, 4)  # Set x-axis limit from 0 to 4
plt.xticks(range(5))  # Set x-ticks from 0 to 4
plt.title('Histogram of Cluster Assignments for 4 Clusters')
plt.show()
```

```
from sklearn.metrics import silhouette_samples, silhouette_score

# Calculate silhouette scores
silhouette_scores = silhouette_samples(MD_x, MD_kmeans_4)

# overall silhouette score
overall_silhouette_score = silhouette_score(MD_x, MD_kmeans_4)

# Create a bar plot of silhouette scores
fig, ax = plt.subplots()
y_lower = 10
for i in range(4):  # Number of clusters
    # Aggregate the silhouette scores  and sort them
    cluster_silhouette_scores = silhouette_scores[MD_kmeans_4 == i]
    cluster_silhouette_scores.sort()

    # Calculate the number of samples in the current cluster
    size_cluster_i = cluster_silhouette_scores.shape[0]
    y_upper = y_lower + size_cluster_i



    color = plt.cm.viridis(float(i) / 4)  # Color map for different clusters
    ax.fill_betweenx(np.arange(y_lower, y_upper),
                     0, cluster_silhouette_scores,
                     facecolor=color, edgecolor=color, alpha=0.7)

    # Label the silhouette plots with their cluster numbers at the middle
    ax.text(-0.05, y_lower + 0.5 * size_cluster_i, str(i))
```

```
    # Compute the new y_lower for next plot
    y_lower = y_upper + 10  # 10 for the 0 samples

# Set labels, title, and legend
ax.set_xlabel("Silhouette coefficient values")
ax.set_ylabel("Cluster label")
ax.set_title("Silhouette plot for K-means clustering")
ax.axvline(x=overall_silhouette_score, color="red", linestyle="--")  # Add a vertical line for the average silhouette score
ax.set_yticks([])  # Clear the yaxis labels / ticks
ax.set_xticks([-0.1, 0, 0.2, 0.4, 0.6, 0.8, 1])

# Display the average silhouette score
ax.text(0.7, 2, "Average silhouette score: {:.2f}".format(overall_silhouette_score))

plt.show()
```

```
#In Python, there isn't a direct equivalent for the stability measure provided by slswFlexclust() in the flexclust package. However, we can compute stability measures using resampling methods and metrics such as the Jaccard index, Adjusted Rand Index, or others.
from sklearn.metrics import jaccard_score
MD_kmeans_4 = cluster_results[4]
def compute_jaccard_index(cluster_assignments1, cluster_assignments2):
    return jaccard_score(cluster_assignments1, cluster_assignments2)
segment_stability = []
for i in range(4):  # Number of clusters
    # Select data points belonging to the current segment
    segment_indices = np.where(MD_kmeans_4 == i)[0]
    segment_assignments = MD_kmeans_4[segment_indices]
```

```
    # Compute Jaccard index between original segment and resampled segment
    jaccard_indices = []
    for _ in range(100):  # Number of resampling iterations
        # Resample the segment assignments
        resampled_segment_assignments = resample(segment_assignments)

        # Compute Jaccard index
        jaccard_index = compute_jaccard_index(segment_assignments,
resampled_segment_assignments)
        jaccard_indices.append(jaccard_index)

    # Average Jaccard index across resampling iterations
    average_jaccard_index = np.mean(jaccard_indices)
    segment_stability.append(average_jaccard_index)

# Plot segment stability
plt.plot(range(1, 5), segment_stability, marker='o')
plt.xlabel('Segment Number')
plt.ylabel('Segment Stability (Jaccard Index)')
plt.title('Segment Stability for K-means Clustering (4 clusters)')
plt.xticks(range(1, 5))
plt.ylim(0, 1)
plt.grid(True)
plt.show()
```

```
#Flexmix function can be used in python as gaussianmixture function
# Defining a function to perform stepwise mixture model selection
def stepFlexmix(data, components_range, nrep, verbose=False):
```

```
    best_model = None
    best_bic = np.inf
        for n_components in components_range:
        for _ in range(nrep):
            # Fit a Gaussian Mixture Model with diagonal covariance structure
            gmm = GaussianMixture(n_components=n_components, covariance_type='diag',
random_state=np.random.randint(1000))
            gmm.fit(data)
        # Calculate the Bayesian Information Criterion (BIC)
            bic = gmm.bic(data)


            # Update the best model and BIC if necessary
            if bic < best_bic:
                best_bic = bic
                best_model = gmm


        if verbose:
            print(f"Number of components: {n_components}, BIC: {best_bic}")


    return best_model
# Perform stepwise mixture model selection
best_gmm_model = stepFlexmix(MD_x, components_range=range(2, 9), nrep=10,
verbose=False)
# Print the best model
print(best_gmm_model)
```

```
from sklearn.mixture import GaussianMixture


# Define a function to calculate ICL
```

```
def calculate_icl(gmm, data):
    return gmm.lower_bound_
# Perform stepwise mixture model selection
components_range = range(2, 9)
nrep = 10
aic_values = []
bic_values = []
icl_values = []

for n_components in components_range:
    best_bic = np.inf
    best_model = None

    for _ in range(nrep):
        # Fit a Gaussian Mixture Model with diagonal covariance structure
        gmm = GaussianMixture(n_components=n_components, covariance_type='diag',
random_state=np.random.randint(1000))
        gmm.fit(MD_x)

        # Calculate AIC and BIC
        aic = gmm.aic(MD_x)
        bic = gmm.bic(MD_x)

        # Update the best model and BIC if necessary
        if bic < best_bic:
            best_bic = bic
            best_model = gmm

    aic_values.append(aic)
```

```
    bic_values.append(best_bic)
    icl_values.append(calculate_icl(best_model, MD_x))

# Plot AIC, BIC, and ICL against the number of components
plt.plot(components_range, aic_values, label='AIC', marker='o')
plt.plot(components_range, bic_values, label='BIC', marker='o')
plt.plot(components_range, icl_values, label='ICL', marker='o')
plt.xlabel('Number of Components')
plt.ylabel('Value of Information Criteria')
plt.title('Information Criteria for Gaussian Mixture Models')
plt.legend()
plt.grid(True)
plt.show()
```

```
best_gmm_model_4 = GaussianMixture(n_components=4, covariance_type='diag',
random_state=1234)
best_gmm_model_4.fit(MD_x)
gmm_cluster_assignments = best_gmm_model_4.predict(MD_x)
# Create a contingency table comparing the cluster assignments from K-means and
GMM
contingency_table = np.zeros((4, 4))
for i in range(4):
    for j in range(4):
        contingency_table[i, j] = np.sum((MD_kmeans_4 == i) &
(gmm_cluster_assignments == j))
# Display the contingency table
print(contingency_table)
```

#In Python, we can't directly replicate the functionality of flexmix from R, as flexmix provides finite mixture modeling with a wide variety of distributions and clustering methods. However , we can use the clustering assignments from K-means as inputs to train a new model using the flexmix-like behavior.

```
MD_df['cluster'] = MD_kmeans_4

MD_x = MD_df.drop(columns=['cluster']).values

# Fit a Gaussian Mixture Model with diagonal covariance structure using the K-means cluster assignments

gmm_with_kmeans_clusters = GaussianMixture(n_components=4, covariance_type='diag', random_state=1234)

gmm_with_kmeans_clusters.fit(MD_x)

# Get cluster assignments from the trained GMM

gmm_cluster_assignments = gmm_with_kmeans_clusters.predict(MD_x)

# Create a contingency table comparing the cluster assignments from K-means and GMM with K-means clusters

contingency_table = np.zeros((4, 4))

for i in range(4):

    for j in range(4):

        contingency_table[i, j] = np.sum((MD_kmeans_4 == i) & (gmm_cluster_assignments == j))

# Display the contingency table

print(contingency_table)
```

```
# Compute log-likelihood for the GMM trained with K-means cluster assignments

log_likelihood_kmeans = gmm_with_kmeans_clusters.score(MD_x)

# Compute log-likelihood for the GMM trained directly with the flexmix-like behavior

log_likelihood_flexmix_like = best_gmm_model_4.score(MD_x)


print("Log-likelihood for the GMM trained with K-means cluster assignments:", log_likelihood_kmeans)
```

```
print("Log-likelihood for the GMM trained directly with the flexmix-like behavior:",
log_likelihood_flexmix_like)
```

```
# Create a table of counts for the 'Like' variable
like_counts = pd.value_counts(df['Like'])
# Reverse the order of the table
reversed_like_counts = like_counts[::-1]
# Print the reversed table
print(reversed_like_counts)
# Create a new column 'Like.n' by subtracting each value of 'Like' from 6
df['Like.n'] = 6 - df['Like'].astype(int)

# Display the table of counts for the 'Like.n' variable
like_n_counts = df['Like.n'].value_counts()
print(like_n_counts)
```

```
import patsy

# Get the names of the first 11 columns of the DataFrame
predictor_names = df.columns[:11]

# Construct the formula string
formula_str = "Like_n ~ " + " + ".join(predictor_names)

# Convert the formula string to a formula object
formula = patsy.Formula(formula_str)

# Display the formula object
```

```
print(formula)

_____# Set random seed

np.random.seed(1234)


# Define the number of components (k)

k = 2


# Define the number of repetitions

nrep = 10


# Perform stepwise finite mixture modeling

best_bic = np.inf

best_model = None


for _ in range(nrep):

    # Fit a Gaussian Mixture Model

    gmm = GaussianMixture(n_components=k, covariance_type='full',
random_state=np.random.randint(1000))

    gmm.fit(df)


    # Calculate the Bayesian Information Criterion (BIC)

    bic = gmm.bic(df)


    # Update the best model and BIC if necessary

    if bic < best_bic:

        best_bic = bic

        best_model = gmm


# Display the best model
```

print(best_model)

#In Python, there isn't a direct equivalent of the refit() function from the flexmix package in R. However, we can refit the Gaussian Mixture Model (GMM) using the same parameters obtained from the stepwise finite mixture modeling.

Refit the best model obtained from stepwise finite mixture modeling

MD_ref2 = best_model

Summary of the refitted model

print(MD_ref2)

Get cluster assignments for each data point

cluster_assignments = MD_ref2.predict(df)

Plot the data points colored by their cluster assignments

plt.figure(figsize=(8, 6))

sns.scatterplot(x='x_variable_name', y='y_variable_name', hue=cluster_assignments, data=df, palette='viridis')

plt.title('Cluster Assignments from Refitted Gaussian Mixture Model')

plt.xlabel('X Variable')

plt.ylabel('Y Variable')

plt.legend(title='Cluster')

plt.grid(True)

plt.show()

Step 6

#In Python, we can perform hierarchical clustering using the scipy library.

from scipy.cluster.hierarchy import linkage, dendrogram

Perform hierarchical clustering

```
Z = linkage(MD_x.T, method='complete', metric='euclidean')


# Plot the dendrogram

plt.figure(figsize=(10, 6))

dendrogram(Z, labels=range(1, MD_x.shape[1] + 1))

plt.title('Hierarchical Clustering Dendrogram')

plt.xlabel('Data Points')

plt.ylabel('Distance')

plt.show()
```

summary : This code helps to generate clusters using hierarchy clustering method.

After getting linkage i.e clusters we are plotting dendogram which helps to know optimal number of clusters.

```
# Reverse the order of the variables after hierarchical clustering

MD_vclust_order_rev = MD_vclust_order[::-1]


# Get the unique cluster labels from Hierarchical clustering

cluster_labels = np.unique(z.labels_)


# Create a bar chart

plt.figure(figsize=(10, 6))

for cluster_label in cluster_labels:

    cluster_counts = np.sum(MD_k4 == cluster_label, axis=0)

    plt.bar(range(len(cluster_counts)), cluster_counts, label=f'Cluster
{cluster_label}', alpha=0.7)


# Shade the bars based on the order of variables after hierarchical clustering

for i, order in enumerate(MD_vclust_order_rev):

    plt.axvspan(i - 0.5, i + 0.5, color='lightgrey', alpha=0.5)
```

plt.title('Cluster Assignments from K-means with Shading based on Hierarchical Clustering Order')

plt.xlabel('Variables')

plt.ylabel('Counts')

plt.xticks(range(len(MD_vclust_order_rev)), MD_vclust_order_rev)

plt.legend()

plt.show()

summary:The ordering of the segmentation variables identified by hierarchical clustering is then used to create the segment profile plot. Marker variables are highlighted.

Define colors for each cluster

colors = ['b', 'g', 'r', 'c']

Create a scatter plot

plt.figure(figsize=(10, 6))

for cluster_label, color in zip(np.unique(MD_k4), colors):

    cluster_indices = MD_k4 == cluster_label

    plt.scatter(MD_pca[cluster_indices, 0], MD_pca[cluster_indices, 1], c=color, label=f'Cluster {cluster_label}', alpha=0.7)

Plot projection axes from PCA

projection_axes = projAxes(MD_pca) # We need to implement this function or use a library that provides it

plt.quiver(*MD_pca.mean(axis=0), *projection_axes[:, 0], color='k', scale=3, label='PC1')

plt.quiver(*MD_pca.mean(axis=0), *projection_axes[:, 1], color='k', scale=3, label='PC2')

plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')

plt.title('Cluster Assignments from K-means on PCA')

plt.legend()

plt.grid(True)

plt.show()

summary: Another visualisation that can help managers grasp the essence of market segments is the segment separation plot . The segment separation plot can be customised with additional arguments. Segment separation plot helps to understand the distance between clusters that is different between clusters from a plot . Each segment in PCA should be defined and understood by domain expert.

Step 7

#In Python, we can create a mosaic plot to visualize the relationship between the cluster assignments obtained from K-means clustering (k4) and the Like variable.

from statsmodels.graphics.mosaicplot import mosaic

Create a DataFrame with cluster assignments and the 'Like' variable

data = pd.DataFrame({'Cluster': k4, 'Like': df['Like']})

Create a mosaic plot

plt.figure(figsize=(10, 6))

mosaic(data, ['Cluster', 'Like'], title='Mosaic Plot of Cluster Assignments and Like Variable')

plt.xlabel('Segment Number')

plt.show()

---

contingency_table = pd.crosstab(k4, df['Gender'])

Create a mosaic plot

plt.figure(figsize=(10, 6))

```
mosaic(contingency_table.stack(), title='Mosaic Plot of Cluster Assignments and Gender', labelizer=lambda k: f'Cluster {k[0]} - Gender {k[1]}', gap=0.01)

plt.show()
```

```
# Create a list to hold the age values for each cluster

age_by_cluster = [df[df['k4'] == cluster]['Age'] for cluster in np.unique(k4)]

# Create a boxplot

plt.figure(figsize=(10, 6))

plt.boxplot(age_by_cluster, labels=np.unique(k4), patch_artist=True, varwidth=True, notch=True)

plt.xlabel('Cluster')

plt.ylabel('Age')

plt.title('Boxplot of Age by Cluster')

plt.show()
```

```
#In Python, we can use the sklearn library to create a decision tree model similar to the ctree function in R's partykit package.

from sklearn.tree import DecisionTreeClassifier

from sklearn.preprocessing import LabelEncoder

from sklearn.tree import plot_tree


# Encode the binary response variable

le = LabelEncoder()

y = le.fit_transform(k4 == 3)


# Encode categorical variables

X = df[['Like.n', 'Age', 'VisitFrequency', 'Gender']]

X_encoded = pd.get_dummies(X, drop_first=True)
```

```
# Create and fit the decision tree model
tree = DecisionTreeClassifier()
tree.fit(X_encoded, y)
# Plot the decision tree
plt.figure(figsize=(15, 10))
plot_tree(tree, feature_names=X_encoded.columns, class_names=['Not Cluster 3', 'Cluster 3'], filled=True)
plt.show()
```

Step 8

```
visit = df.groupby(k4)['VisitFrequency'].mean()
# Print the result
print(visit)
```

summary: Grouping ‘k4’ column and getting average values of ‘visitfrequency ‘ column from each class in k4 column’s

---

```
# Calculate the mean value of the 'Like.n' variable for each cluster
like = df.groupby(k4)['Like.n'].mean()
# Print the result
print(like)
```

summary: summary: Grouping ‘k4’ column and getting average value of ‘like.n ‘ column from each class in k4 column’s

---

```
# Calculate the proportion of females in each cluster
female = df['Gender'].eq('Female').groupby(k4).mean()
# Print the result
print(female)
```

summary : To know the proportion of females in the gender column tge code helps to know the proportion.

#In Python, we can create a scatter plot to visualize the relationship between the mean visit frequency (visit) and the mean Like.n score (like) for each cluster obtained from K-means clustering.

```
# Create a scatter plot

plt.figure(figsize=(10, 6))

plt.scatter(visit, like, s=10 * female, c=np.arange(1, len(visit) + 1))


# Set the x-axis and y-axis limits

plt.xlim(2, 4.5)

plt.ylim(-3, 3)


# Add text labels for each cluster

for i, txt in enumerate(range(1, len(visit) + 1)):

   plt.text(visit[i], like[i], txt)


# Add labels and title

plt.xlabel('Mean Visit Frequency')

plt.ylab

el('Mean Like.n Score')

plt.title('Relationship between Mean Visit Frequency and Mean Like.n Score by Cluster')


# Show the plot

plt.show()
```

summary: The code implies plotting mean visit frequency and mean like.n score into a scatter plot with colour separated according to kmeans clustering as scatters in the scatter plot are separated from colours where each colour represents a cluster and its properties.
