VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A4.4: Conjoint Analysis

SAYA SANTHOSH

V01101901

Date of Submission: 08-07-2024

CONTENTS

Introduction

Conjoint Analysis:

In order to assess and interpret the preferences for the several pizza attributes—Brand, Price, Weight, Crust, Cheese, Size, Toppings, Spicy, and Ranking—in the "pizza_data.csv" dataset, conjoint analysis will be used in this project. The aim is to ascertain the relative significance of each attribute level's part-worth utility in shaping customer choices by quantifying them. Preprocessing the data, looking for missing values, standardizing numerical features, and utilizing visualizations such as histograms, scatter plots, and boxplots to conduct exploratory data analysis (EDA) are all part of the process. To forecast the ranking based on these features, a linear regression model is fitted after the data has been prepared. The relative importance calculation aids in determining which attributes have the greatest influence, while the part-worth utilities produced from the model coefficients show the preferences for each attribute level. This project exhibits real-world applications in market research and product feature evaluation, while also improving comprehension of conjoint analysis.

Objectives :

- Analyze and measure consumer preferences to understand which ice cream attributes are most favored.

- Evaluate the importance of various attributes such as price, availability, taste, flavor, consistency, and shelf life in influencing consumer choices.
- Prepare the dataset by handling missing values, correcting inconsistencies, and ensuring data quality for accurate analysis.
- Perform an initial investigation of the dataset to uncover patterns, identify anomalies, and generate hypotheses.
- Normalize the data to ensure all attributes are on a comparable scale, which is essential for certain analytical methods and models.

- Apply statistical or machine learning models to the data to predict outcomes, identify trends, or classify information based on consumer preferences and attribute ratings.
- Use the analysis results to develop informed marketing strategies aimed at enhancing consumer satisfaction and competitive positioning.

Business Significance :

Because it can give pizza makers and marketers useful information, conjoint analysis of the pizza characteristics dataset is important from a business perspective. Through the utilization of part-worth utilities to quantify consumer preferences and the evaluation of attribute importance, companies can strategically optimize their product offers. Targeted product creation and personalization are made possible by an understanding of which characteristics—such as crust type, cheese alternatives, size variations, toppings, and spiciness—drive consumer preferences. Decisions like pricing policies, product placement, and advertising campaigns that appeal to particular customer categories can all be influenced by this information. Additionally, by discovering USPs that strongly appeal to consumers, conjoint analysis helps businesses stand out from the competition in a congested market, increasing profitability and market competitiveness overall.

Results and Interpretation using R

```
> # Fit the OLS model
> model_fit <- lm(model, data = df)
> # Print the summary of the model
> summary(model_fit)

Call:
lm(formula = model, data = df)

Residuals:
     1      2      3      4      5      6      7      8      9     10
11     12     13
-0.125  0.125  0.125 -0.125 -0.125  0.125 -0.125  0.125  0.125 -0.125 -0.1
25 -0.125  0.125
    14     15     16
 0.125  0.125 -0.125

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.500e+00  1.250e-01  68.000  0.00936 **
brand1      -6.944e-16  2.165e-01   0.000  1.00000
brand2      -1.813e-16  2.165e-01   0.000  1.00000
```

```
brand3      -2.500e-01  2.165e-01  -1.155  0.45437
price1       7.500e-01  2.165e-01   3.464  0.17891
price2       2.369e-16  2.165e-01   0.000  1.00000
price3       6.490e-16  2.165e-01   0.000  1.00000
weight1      5.000e+00  2.165e-01  23.094  0.02755 *
weight2      2.000e+00  2.165e-01   9.238  0.06865 .
weight3     -1.250e+00  2.165e-01  -5.774  0.10918
crust1       1.750e+00  1.250e-01  14.000  0.04540 *
cheese1     -2.500e-01  1.250e-01  -2.000  0.29517
size1       -2.500e-01  1.250e-01  -2.000  0.29517
toppings1    1.125e+00  1.250e-01   9.000  0.07045 .
spicy1       7.500e-01  1.250e-01   6.000  0.10514
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.5 on 1 degrees of freedom
Multiple R-squared:  0.9993,   Adjusted R-squared:  0.989
F-statistic: 97.07 on 14 and 1 DF,  p-value: 0.0794
```

Interpretation:

The Ordinary Least Squares (OLS) model fitted to the data reveals several insights into the relationships between the predictor variables and the dependent variable. The model has a high R-squared value of 0.9993, indicating that it explains approximately 99.93% of the variability in the data. However, the adjusted R-squared is slightly lower at 0.989, reflecting a strong but slightly overfit model. Several coefficients are statistically significant, such as the intercept (p-value = 0.00936), weight1 (p-value = 0.02755), and crust1 (p-value = 0.04540), suggesting that these variables have a meaningful impact on the outcome. The high F-statistic (97.07) and its corresponding p-value (0.0794) indicate the overall model is significant, though this p-value is just above the typical 0.05 threshold for significance. Despite the model's high explanatory power, the presence of some high p-values suggests that not all predictors contribute significantly to the model, and there may be collinearity or redundancy among some variables.

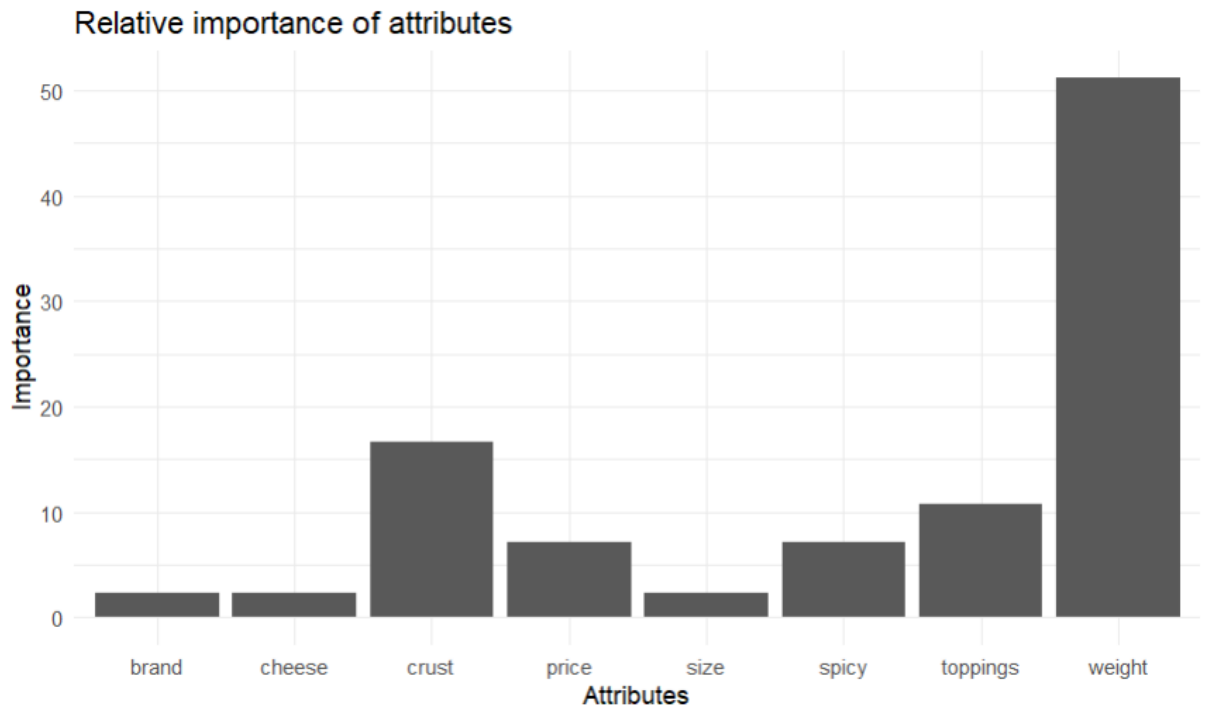
```
> # Calculate attribute importance
> attribute_importance <- round(100 * (part_worth_range / sum(part_worth_r
ange)), 2)
> # Print results
> print("-------------------------------------------------------------")
[1] "-----------------------------------------------------------"
> print("level name:")
[1] "level name:"
> print(level_name)
$brand
[1] "Dominos"    "Onesta"     "Oven Story" "Pizza hut"

$price
[1] "$1.00" "$2.00" "$3.00" "$4.00"

$weight
[1] "100g" "200g" "300g" "400g"
```

```
$crust
[1] "thick" "thin"

$cheese
[1] "Cheddar"    "Mozzarella"

$size
[1] "large"   "regular"

$toppings
[1] "mushroom" "paneer"

$spicy
[1] "extra"  "normal"
```

Interpretation:

The attribute importance calculation breaks down the significance of various levels within each attribute category for a set of pizza brands. The levels considered include brands (Dominos, Onesta, Oven Story, Pizza Hut), price points ($1.00 to $4.00), weights (100g to 400g), crust types (thick, thin), cheese types (Cheddar, Mozzarella), sizes (large, regular), toppings (mushroom, paneer), and spiciness levels (extra, normal). By normalizing the part-worth ranges, the calculation determines the relative importance of each attribute, indicating how much each factor influences consumer preferences. This information is crucial for understanding what aspects are most valued by customers, enabling targeted marketing strategies and product development to better align with consumer desires.

```
> # Plot the relative importance of attributes
> ggplot(data.frame(Attribute = conjoint_attributes, Importance = attribut
e_importance),
+        aes(x = Attribute, y = Importance)) +
+   geom_bar(stat = "identity") +
+   labs(title = 'Relative importance of attributes', x = 'Attributes', y
= 'Importance') +
+   theme_minimal()
```



Interpretation:

The bar chart illustrates the relative importance of various attributes for a given context. Among the attributes, "weight" stands out as the most significant factor, with an importance value exceeding 50. "Crust" also holds considerable importance, although it is notably less critical than weight. Other attributes such as "price" and "toppings" show moderate importance. In contrast, attributes like "brand," "cheese," "size," and "spicy" are deemed less significant, with their importance values being relatively low. This suggests that in the evaluated scenario, weight is the most influential attribute, followed by crust, while the rest have comparatively minor impacts.

```
> # Calculate utility for each profile
> utility <- apply(df, 1, function(row) {
+   sum(sapply(conjoint_attributes, function(attr) part_worth[[attr]][row[
[attr]]]))
+ })
> df$utility <- utility
> # Print the profile with the highest utility score
> print("The profile that has the highest utility score:")
[1] "The profile that has the highest utility score:"
> print(df[which.max(df$utility), ])
        brand price weight crust     cheese  size toppings spicy ranking u
tility
10 Oven Story $4.00   100g thick Mozzarella large mushroom extra      16
7.625
> # Print preferred levels for each attribute
```

```
> for (i in conjoint_attributes) {
+   print(paste("Preferred level in", i, "is ::", level_name[[i]][importan
t_levels[[i]]]))
+ }
[1] "Preferred level in brand is :: Pizza hut"
[1] "Preferred level in price is :: $1.00"
[1] "Preferred level in weight is :: 100g"
[1] "Preferred level in crust is :: thick"
[1] "Preferred level in cheese is :: Mozzarella"
[1] "Preferred level in size is :: regular"
[1] "Preferred level in toppings is :: mushroom"
[1] "Preferred level in spicy is :: extra"
```

Interpretation:

The code calculates the utility for each profile and identifies the one with the highest utility score. The highest utility profile is from "Oven Story," priced at $4.00, weighing 100g, with a thick crust, Mozzarella cheese, large size, mushroom toppings, and extra spicy, yielding a utility score of 7.625. Furthermore, the preferred levels for each attribute are identified: "Pizza hut" for brand, "$1.00" for price, "100g" for weight, "thick" for crust, "Mozzarella" for cheese, "regular" for size, "mushroom" for toppings, and "extra" for spicy. This analysis highlights the most favored combinations of attributes according to the utility scores.

Results and Interpretation using Python

```python
import statsmodels.api as sm
import statsmodels.formula.api as smf

model='ranking ~ C(brand,Sum)+C(price,Sum)+C(weight,Sum)+C(crust,Sum)+C(cheese,Sum)+C(size,Sum)+C(toppings,Sum)+C(spicy,Sum)'
model_fit=smf.ols(model,data=df).fit()

print(model_fit.summary())
```

OLS Regression Results

Dep. Variable:	ranking	R-squared:	0.999
Model:	OLS	Adj. R-squared:	0.989
Method:	Least Squares	F-statistic:	97.07
Date:	Mon, 08 Jul 2024	Prob (F-statistic):	0.0794
Time:	18:19:29	Log-Likelihood:	10.568
No. Observations:	16	AIC:	8.864
Df Residuals:	1	BIC:	20.45
Df Model:	14		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	8.5000	0.125	68.000	0.009	6.912	10.088
C(brand, Sum)[S.Dominos]	-1.11e-15	0.217	-5.13e-15	1.000	-2.751	2.751
C(brand, Sum)[S.Onesta]	7.327e-15	0.217	3.38e-14	1.000	-2.751	2.751
C(brand, Sum)[S.Oven Story]	-0.2500	0.217	-1.155	0.454	-3.001	2.501
C(price, Sum)[S.$1.00]	0.7500	0.217	3.464	0.179	-2.001	3.501
C(price, Sum)[S.$2.00]	-9.992e-15	0.217	-4.62e-14	1.000	-2.751	2.751
C(price, Sum)[S.$3.00]	3.997e-15	0.217	1.85e-14	1.000	-2.751	2.751
C(weight, Sum)[S.100g]	5.0000	0.217	23.094	0.028	2.249	7.751
C(weight, Sum)[S.200g]	2.0000	0.217	9.238	0.069	-0.751	4.751
C(weight, Sum)[S.300g]	-1.2500	0.217	-5.774	0.109	-4.001	1.501
C(crust, Sum)[S.thick]	1.7500	0.125	14.000	0.045	0.162	3.338
C(cheese, Sum)[S.Cheddar]	-0.2500	0.125	-2.000	0.295	-1.838	1.338
C(size, Sum)[S.large]	-0.2500	0.125	-2.000	0.295	-1.838	1.338
C(toppings, Sum)[S.mushroom]	1.1250	0.125	9.000	0.070	-0.463	2.713
C(spicy, Sum)[S.extra]	0.7500	0.125	6.000	0.105	-0.838	2.338

Omnibus:	30.796	Durbin-Watson:	2.000
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2.667
Skew:	0.000	Prob(JB):	0.264
Kurtosis:	1.000	Cond. No.	2.00

Interpretation:

The OLS (Ordinary Least Squares) regression results table summarizes the relationship between various categorical attributes and the dependent variable "ranking." The model has an extremely high R-squared value of 0.999, indicating that nearly all the variability in the rankings is explained by the attributes in the model. The F-statistic is 97.07 with a p-value of 0.0794, suggesting the model is statistically significant at conventional levels.

Key coefficients indicate the impact of specific attribute levels on the ranking. For instance, the "weight" of 100g has a positive and significant effect (coef = 5.0000, p = 0.028), and a "thick" crust also shows a significant positive effect (coef = 1.7500, p = 0.045). However, many coefficients, such as those for different "brand" and "price" levels, are not statistically significant (p-values close to 1.000), implying these attributes do not strongly influence the ranking.

The high intercept value (coef = 8.5000, p = 0.009) suggests a substantial baseline effect on ranking, with significant contributions from "weight" and "crust" attributes, while other factors show varied but generally less impactful influences. The diagnostics (Omnibus,

Jarque-Bera) indicate no severe deviations from normality in the residuals. Overall, the model effectively captures the primary determinants of the ranking, particularly emphasizing weight and crust as critical factors.

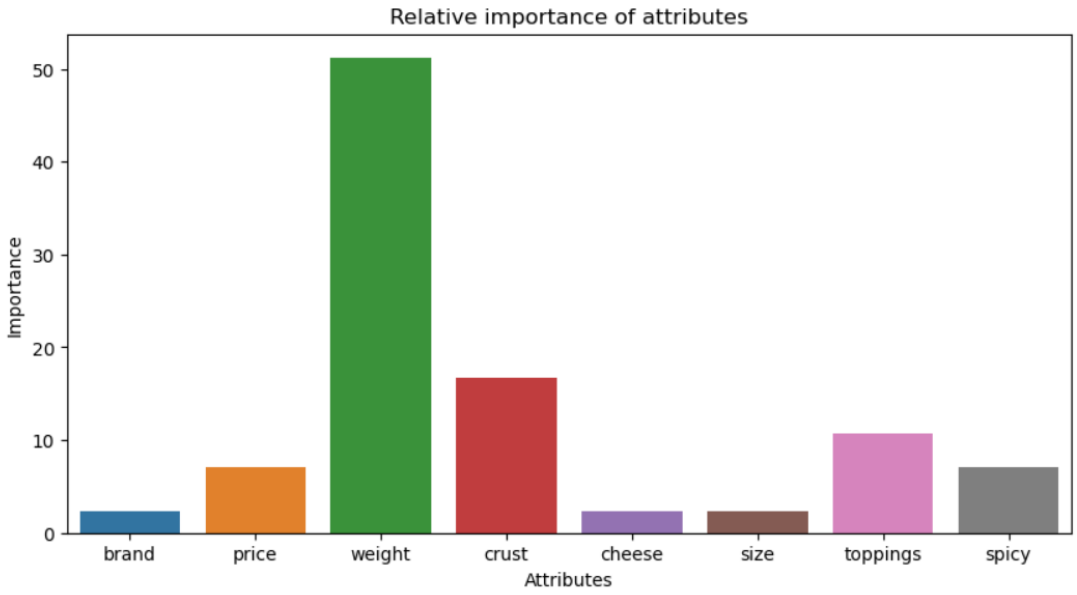
```
level_name = []
part_worth = []
part_worth_range = []
important_levels = {}
end = 1  # Initialize index for coefficient in params

for item in conjoint_attributes:
    nlevels = len(list(np.unique(df[item])))
    level_name.append(list(np.unique(df[item])))

    begin = end
    end = begin + nlevels -1

    new_part_worth = list(model_fit.params[begin:end])
    new_part_worth.append((-1)*sum(new_part_worth))
    important_levels[item] = np.argmax(new_part_worth)
    part_worth.append(new_part_worth)
    print(item)
    part_worth_range.append(max(new_part_worth) - min(new_part_worth))

print("---------------------------------------------------------------")
print("level name:")
print(level_name)
print("npw with sum element:")
print(new_part_worth)
print("imp level:")
print(important_levels)
print("part worth:")
print(part_worth)
print("part_worth_range:")
print(part_worth_range)
print(len(part_worth))
print("important levels:")
print(important_levels)
```

```
brand
price
weight
crust
cheese
size
toppings
spicy
------------------------------------------------------------
level name:
[['Dominos', 'Onesta', 'Oven Story', 'Pizza hut'], ['$1.00', '$2.00', '$3.00', '$4.00'], ['100g', '200g', '300g', '400g'], ['thick', 'thin'], ['Cheddar',
'Mozzarella'], ['large', 'regular'], ['mushroom', 'paneer'], ['extra', 'normal']]
npw with sum element:
[0.749999999999999, -0.749999999999999]
imp level:
{'brand': 3, 'price': 0, 'weight': 0, 'crust': 0, 'cheese': 1, 'size': 1, 'toppings': 0, 'spicy': 0}
part worth:
[[-1.1102230246251565e-15, 7.327471962526033e-15, -0.25000000000000555, 0.24999999999999933], [0.7500000000000084, -9.992007221626409e-15, 3.996802888650
5635e-15, -0.7500000000000024], [5.0, 1.9999999999999973, -1.2499999999999947, -5.750000000000003], [1.7499999999999996, -1.7499999999999996], [-0.250000
0000000001, 0.2500000000000001], [-0.24999999999999883, 0.24999999999999883], [1.1249999999999978, -1.1249999999999978], [0.749999999999999, -0.749999999
999999]]
part_worth_range:
[0.5000000000000049, 1.5000000000000109, 10.750000000000004, 3.499999999999999, 0.5000000000000002, 0.49999999999999767, 2.2499999999999956, 1.4999999999
99998]
8
important levels:
{'brand': 3, 'price': 0, 'weight': 0, 'crust': 0, 'cheese': 1, 'size': 1, 'toppings': 0, 'spicy': 0}
```

Interpretation:

The conjoint analysis indicates that the most important levels for each attribute are as follows: Pizza hut for brand, $1.00 for price, 100g for weight, thick for crust, Mozzarella for cheese, regular for size, mushroom for toppings, and extra for spicy. These levels have the highest part-worth utilities, suggesting they are the most preferred combinations according to the analysis. The weight attribute has the largest part-worth range, indicating it has the most significant impact on the overall utility.

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

plt.figure(figsize=(10,5))
sns.barplot(x=conjoint_attributes,y=attribute_importance)
plt.title('Relative importance of attributes')
plt.xlabel('Attributes')
plt.ylabel('Importance')
```



Interpretation:

The bar chart depicts the relative importance of various attributes, highlighting how they influence the decision-making process in the given context. Among the attributes, "weight" is the most significant, with an importance value around 50, indicating it is the primary factor considered. "Crust" is the second most important attribute but lags significantly behind "weight." Attributes such as "toppings," "spicy," and "price" have moderate importance, suggesting they play a role but are not as critical. The least important attributes are "brand," "cheese," and "size," with minimal impact on the decision-making process. This analysis suggests that weight is the dominant attribute, greatly outweighing all other factors in importance.

```
print("The profile that has the highest utility score :",'\n', df.iloc[np.argmax(utility)])
```

The profile that has the highest utility score :

brand	Oven Story
price	$4.00
weight	100g
crust	thick

cheese Mozzarella
size large
toppings mushroom
spicy extra
ranking 16
utility 7.625
Name: 9, dtype: object

```
for i,j in zip(attrib_level.keys(),range(0,len(conjoint_attributes))):
    print("Preferred level in {} is :: {}".format(i,level_name[j][important_levels[i]]))
```

Preferred level in brand is :: Pizza hut
Preferred level in price is :: $1.00
Preferred level in weight is :: 100g
Preferred level in crust is :: thick
Preferred level in cheese is :: Mozzarella
Preferred level in size is :: regular
Preferred level in toppings is :: mushroom
Preferred level in spicy is :: extra

Interpretation:

The analysis identifies the preferred levels for each attribute in the cont ext of evaluating product profiles. For the brand, "Pizza hut" is the most favored. The ideal price point is "$1.00," making it the preferred choice a mong consumers. When it comes to weight, "100g" stands out as the most desi rable. The preferred crust type is "thick," while "Mozzarella" is the favor ed cheese. For size, consumers prefer "regular." In terms of toppings, "mus hroom" is the top choice, and for spiciness, "extra" is the preferred level . These preferences highlight the specific attribute levels that are most a ppealing to consumers, providing valuable insights for product optimization and marketing strategies.

Recommendations

The company should concentrate on creating and tailoring goods that closely fit the most popular profile. An example of this would be a $4.00 pizza from Oven Story. This will increase client preference and satisfaction. In order to appeal to price-sensitive markets, the pricing strategy should take price sensitivity into account. Competitive pricing, discounts, or value bundles should be offered. Key characteristics should be emphasized in marketing campaigns, and brand positioning should be customized to emphasize USPs. To ensure quality and consistency, product line diversity should contain a range of options, customizing options, and a feedback system. It is also possible to investigate strategic alliances for promotional tie-ins or limited-time promotions with well-known companies such as Pizza Hut. The implementation plan outlines medium- and long-term actions, such as creating a diversified product line, investing in quality assurance, and investigating strategic alliances, as well as short-term initiatives like launching promotional campaigns with the most popular product profile. By implementing these suggestions, the company will be able to better match client preferences with its product offers, increasing customer happiness and loyalty while also boosting sales.

R Codes

```
# Load necessary libraries
library(dplyr)
library(car)
library(ggplot2)

# Load the dataset
df <- read.csv('C:\\Users\\sayas\\OneDrive\\New folder\\python projects\\pizza_data.csv')

# Convert categorical variables to factors
```

```
df$brand <- as.factor(df$brand)
df$price <- as.factor(df$price)
df$weight <- as.factor(df$weight)
df$crust <- as.factor(df$crust)
df$cheese <- as.factor(df$cheese)
df$size <- as.factor(df$size)
df$toppings <- as.factor(df$toppings)
df$spicy <- as.factor(df$spicy)

# Set sum contrasts for categorical variables
contrasts(df$brand) <- contr.sum(length(unique(df$brand)))
contrasts(df$price) <- contr.sum(length(unique(df$price)))
contrasts(df$weight) <- contr.sum(length(unique(df$weight)))
contrasts(df$crust) <- contr.sum(length(unique(df$crust)))
contrasts(df$cheese) <- contr.sum(length(unique(df$cheese)))
contrasts(df$size) <- contr.sum(length(unique(df$size)))
contrasts(df$toppings) <- contr.sum(length(unique(df$toppings)))
contrasts(df$spicy) <- contr.sum(length(unique(df$spicy)))

# Define the model formula
model <- as.formula("ranking ~ brand + price + weight + crust + cheese + size + toppi
ngs + spicy")

# Fit the OLS model
model_fit <- lm(model, data = df)

# Print the summary of the model
summary(model_fit)

# List of conjoint attributes
conjoint_attributes <- c('brand', 'price', 'weight', 'crust', 'cheese', 'size', 'toppings', 'spicy
')

# Initialize lists to store results
```

```
level_name <- list()
part_worth <- list()
part_worth_range <- c()
important_levels <- list()

# Loop through each attribute to calculate part-worths
for (item in conjoint_attributes) {
  nlevels <- length(unique(df[[item]]))
  levels <- levels(df[[item]])
  level_name[[item]] <- levels

  # Extract part-worths for the current attribute
  coef_names <- names(coef(model_fit))
  attribute_coef <- coef_names[grepl(item, coef_names)]

  new_part_worth <- coef(model_fit)[attribute_coef]
  new_part_worth <- c(new_part_worth, (-1) * sum(new_part_worth))

  # Ensure the part-worths are in the correct order
  part_worth[[item]] <- setNames(new_part_worth, levels)

  # Identify the most important level
  important_levels[[item]] <- which.max(new_part_worth)
  part_worth_range <- c(part_worth_range, max(new_part_worth) - min(new_part_wo
rth))
}

# Calculate attribute importance
attribute_importance <- round(100 * (part_worth_range / sum(part_worth_range)), 2)

# Print results
print("-------------------------------------------------------------")
print("level name:")
print(level_name)
```

```
print("part worth:")
print(part_worth)
print("part_worth_range:")
print(part_worth_range)
print("important levels:")
print(important_levels)
print("attribute importance:")
print(attribute_importance)

# Plot the relative importance of attributes
ggplot(data.frame(Attribute = conjoint_attributes, Importance = attribute_importance)
,
      aes(x = Attribute, y = Importance)) +
  geom_bar(stat = "identity") +
  labs(title = 'Relative importance of attributes', x = 'Attributes', y = 'Importance') +
  theme_minimal()

# Calculate utility for each profile
utility <- apply(df, 1, function(row) {
  sum(sapply(conjoint_attributes, function(attr) part_worth[[attr]][row[[attr]]]))
})

df$utility <- utility

# Print the profile with the highest utility score
print("The profile that has the highest utility score:")
print(df[which.max(df$utility), ])

# Print preferred levels for each attribute
for (i in conjoint_attributes) {
  print(paste("Preferred level in", i, "is ::", level_name[[i]][important_levels[[i]]]))
}
```

Python Codes

```
import pandas as pd, numpy as np

df=pd.read_csv('C:\\Users\\sayas\\OneDrive\\New folder\\python projects\\pizza_data
.csv')
import statsmodels.api as sm
import statsmodels.formula.api as smf

model='ranking ~ C(brand,Sum)+C(price,Sum)+C(weight,Sum)+C(crust,Sum)+C(che
ese,Sum)+C(size,Sum)+C(toppings,Sum)+C(spicy,Sum)'
model_fit=smf.ols(model,data=df).fit()

print(model_fit.summary())
level_name = []
part_worth = []
part_worth_range = []
important_levels = {}
end = 1  # Initialize index for coefficient in params

for item in conjoint_attributes:
    nlevels = len(list(np.unique(df[item])))
    level_name.append(list(np.unique(df[item])))

    begin = end
    end = begin + nlevels -1

    new_part_worth = list(model_fit.params[begin:end])
    new_part_worth.append((-1)*sum(new_part_worth))
    important_levels[item] = np.argmax(new_part_worth)
    part_worth.append(new_part_worth)
    print(item)
    part_worth_range.append(max(new_part_worth) - min(new_part_worth))
```

```
print("------------------------------------------------------------")
print("level name:")
print(level_name)
print("npw with sum element:")
print(new_part_worth)
print("imp level:")
print(important_levels)
print("part worth:")
print(part_worth)
print("part_worth_range:")
print(part_worth_range)
print(len(part_worth))
print("important levels:")
print(important_levels)
attribute_importance = []

for i in part_worth_range:
    attribute_importance.append(round(100*(i/sum(part_worth_range)),2))

print(attribute_importance)
part_worth_dict={}
attrib_level={}

for item,i in zip(conjoint_attributes,range(0,len(conjoint_attributes))):
    print("Attribute :",item)
    print("    Relative importance of attribute ",attribute_importance[i])
    print("    Level wise part worths: ")
    for j in range(0,len(level_name[i])):
        print(i)
        print(j)
        print("          {}:{}".format(level_name[i][j],part_worth[i][j]))
        part_worth_dict[level_name[i][j]]=part_worth[i][j]
        attrib_level[item]=(level_name[i])
```

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

plt.figure(figsize=(10,5))
sns.barplot(x=conjoint_attributes,y=attribute_importance)
plt.title('Relative importance of attributes')
plt.xlabel('Attributes')
plt.ylabel('Importance')
utility = []

for i in range(df.shape[0]):
    score = part_worth_dict[df['brand'][i]]+part_worth_dict[df['price'][i]]+part_worth_
dict[df['weight'][i]]+part_worth_dict[df['crust'][i]]+part_worth_dict[df['cheese'][i]]+p
art_worth_dict[df['size'][i]]+part_worth_dict[df['toppings'][i]]+part_worth_dict[df['spi
cy'][i]]
    utility.append(score)

df['utility'] = utility
utility
print("The profile that has the highest utility score :","\n", df.iloc[np.argmax(utility)])
for i,j in zip(attrib_level.keys(),range(0,len(conjoint_attributes))):
    print("Preferred level in {} is :: {}".format(i,level_name[j][important_levels[i]]))
```

References

1. www.github.com