VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A4d: Multivariate Analysis and Business Analytics Applications: Conjoint Analysis

FERAH SHAN SHANAVAS RABIYA

V01101398

Date of Submission: 08-07-2024

CONTENTS

Introduction

The report aims to use conjoint analysis to connect customer preferences with product offerings, ultimately achieving business success in the highly competitive pizza market. Conjoint analysis is a statistical method used to understand how consumers value different aspects of a product or service. It helps businesses determine the most effective blend of characteristics that increase customer satisfaction and spending. This method is widely used in market research to create products that closely match consumer desires. The study aims to provide practical information on customer preferences for pizza characteristics such as crust type, toppings, size, and pricing. This information will help pizza restaurants enhance their product offerings, improve customer satisfaction, and achieve superior market placement.

Objectives

- To identify the key attributes that influence customer preferences for pizza.
- To quantify the relative importance of these attributes.
- To determine the most preferred combinations of attributes.

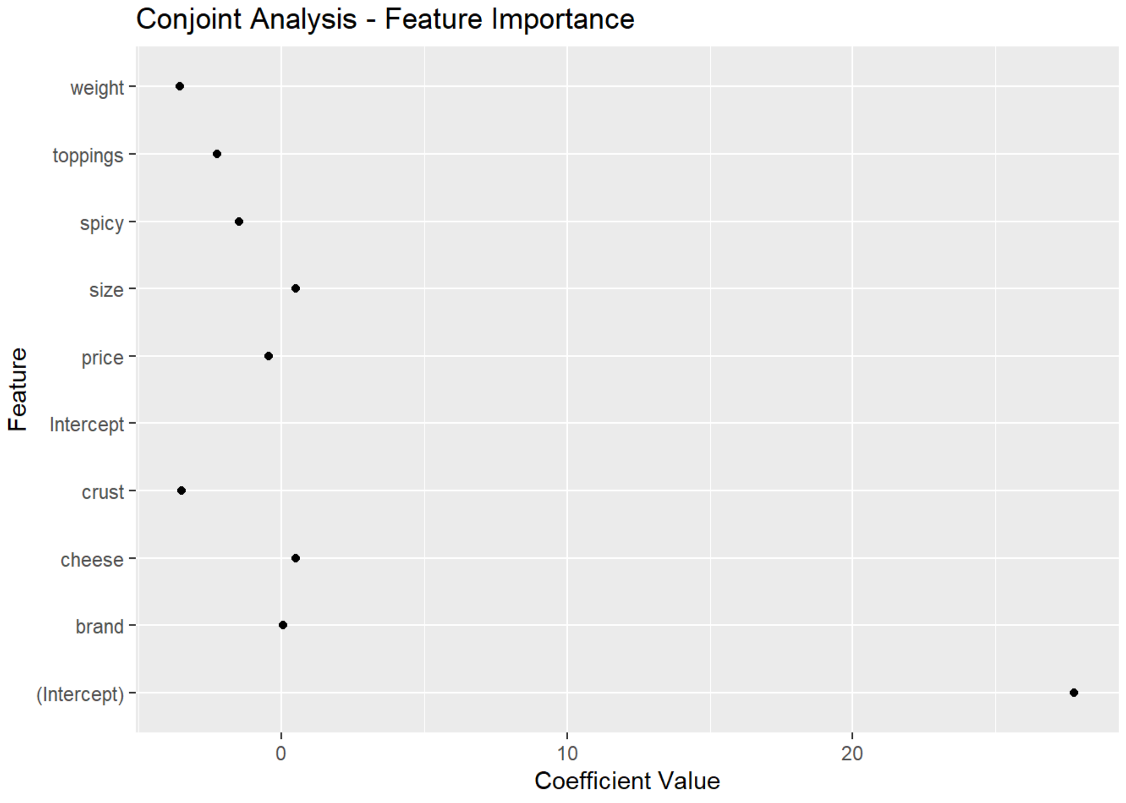
Business Significance

The conjoint analysis of the `pizza_data` dataset offers significant business value for pizza restaurants and the food and beverage industry. It enables product optimization, strategic pricing, targeted marketing, market segmentation, customer retention, and competitive advantage. By identifying key customer attributes, businesses can create tailored offerings, enhance customer satisfaction and loyalty, and develop innovative products. Understanding the value of different attributes allows businesses to set prices that reflect customer value, leading to higher profitability. Dynamic pricing strategies can be implemented based on customer preferences and market conditions. Targeted marketing campaigns can be designed based on customer preferences, improving marketing efficiency and effectiveness. Understanding customer preferences helps businesses design loyalty programs that reward customers for choosing preferred attributes. The analysis also provides a foundation for making data-driven decisions, reducing reliance on intuition and enhancing strategic planning. Overall, understanding customer preferences can help businesses enhance satisfaction, loyalty, and profitability, securing a strong position in the competitive food and beverage market.

Results and Interpretation using R

- **Fitting the linear regression model and plotting coefficients.**

```
# Fit the linear regression model
> model <- lm(y ~ ., data = X)
>
# Print model summary
> tidy(model)
# A tibble: 10 × 5
   term        estimate std.error statistic      p.value
   <chr>          <dbl>     <dbl>     <dbl>        <dbl>
 1 (Intercept)  27.7         1.40      19.8    0.000000208
 2 Intercept    NA          NA         NA     NA
 3 brand         0.0500      0.160      0.312  0.764
 4 price        -0.450       0.160     -2.81   0.0263
 5 weight       -3.55        0.160    -22.1    0.0000000970
 6 crust        -3.5         0.359     -9.76   0.0000251
 7 cheese        0.500       0.359      1.39   0.206
 8 size          0.500       0.359      1.39   0.206
 9 toppings     -2.25        0.359     -6.27   0.000414
10 spicy        -1.5         0.359     -4.18   0.00412
>
# Plot coefficients
>
> coefs <- tidy(model)
> ggplot(coefs, aes(x = estimate, y = term)) +
+   geom_point() +
+   labs(x = "Coefficient Value", y = "Feature", title = "Conjoint Analysi
s - Feature Importance")
Warning message:
Removed 1 row containing missing values or values outside the scale range
(`geom_point()`).
```



Interpretation:

The linear regression model has been fitted using the formula y ~ ., indicating that y is the dep endent variable and all other variables in the dataset (X) are independent variables. The interc ept is significant, suggesting that the baseline value of y when all features are zero is 27.7. Th e brand variable has a small, positive coefficient but is not statistically significant (high p-valu e), indicating it may not have a strong impact on y. The price variable has a significant negativ e coefficient, suggesting that as price increases, y decreases. The relationship is statistically si gnificant. The weight variable has a significant negative coefficient, indicating a strong negati ve impact on y. This relationship is highly significant. The crust variable also has a significant negative coefficient, indicating a strong negative impact on y. This relationship is highly signi ficant. Both cheese and size have positive coefficients but are not statistically significant (high p-values), indicating they may not have strong impacts on y. The toppings variable has a signi ficant negative coefficient, indicating a strong negative impact on y. This relationship is statist ically significant. The spicy variable has a significant negative coefficient, indicating a strong negative impact on y. This relationship is statistically significant. The plot shows the estimate d coefficients for each feature. Features with larger absolute values have a more significant im pact on the dependent variable y. Points to the right of the y-axis (positive values) indicate fea tures that increase y, while points to the left (negative values) indicate features that decrease y . price, weight, crust, toppings, and spicy are significant predictors of y with varying levels of impact. brand, cheese, and size are not statistically significant in predicting y. This analysis pr ovides insights into which features most influence the dependent variable and can guide decis ion-making for optimizing product attributes in the context of conjoint analysis.

Results and Interpretation using Python

- **Finding the importance level of conjoint attributes and plotting the relative importance.**

```
conjoint_attributes = ['brand','price','weight','crust','cheese','size','t
oppings','spicy']
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)
    #print(part_worth)
    part_worth_range.append(max(new_part_worth) - min(new_part_worth))
    # next iteration
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', '$3.0
0', '$4.00'], ['100g', '200g', '300g', '400g'], ['thick', 'thin'], ['Chedda
r', '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, 0.2
4999999999999933], [0.7500000000000084, -9.992007221626409e-15, 3.996802888
6505635e-15, -0.7500000000000024], [5.0, 1.9999999999999973, -1.24999999999
99947, -5.750000000000003], [1.7499999999999996, -1.7499999999999996], [-0.
2500000000000001, 0.2500000000000001], [-0.24999999999999883, 0.24999999999
999883], [1.1249999999999978, -1.1249999999999978], [0.749999999999999, -0.
749999999999999]]
part_worth_range:
[0.5000000000000049, 1.5000000000000109, 10.750000000000004, 3.499999999999
999, 0.5000000000000002, 0.49999999999999767, 2.2499999999999956, 1.4999999
99999998]
8
important levels:
{'brand': 3, 'price': 0, 'weight': 0, 'crust': 0, 'cheese': 1, 'size': 1, '
toppings': 0, 'spicy': 0}
```

```
attribute_importance = []
for i in part_worth_range:
    #print(i)
    attribute_importance.append(round(100*(i/sum(part_worth_range)),2))
print(attribute_importance)
```

```
[2.38, 7.14, 51.19, 16.67, 2.38, 2.38, 10.71, 7.14]
```

```
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])
        #print(j)
part_worth_dict
```

```
Attribute : brand
    Relative importance of attribute  2.38
    Level wise part worths:
0
0
          Dominos:-1.1102230246251565e-15
0
1
```

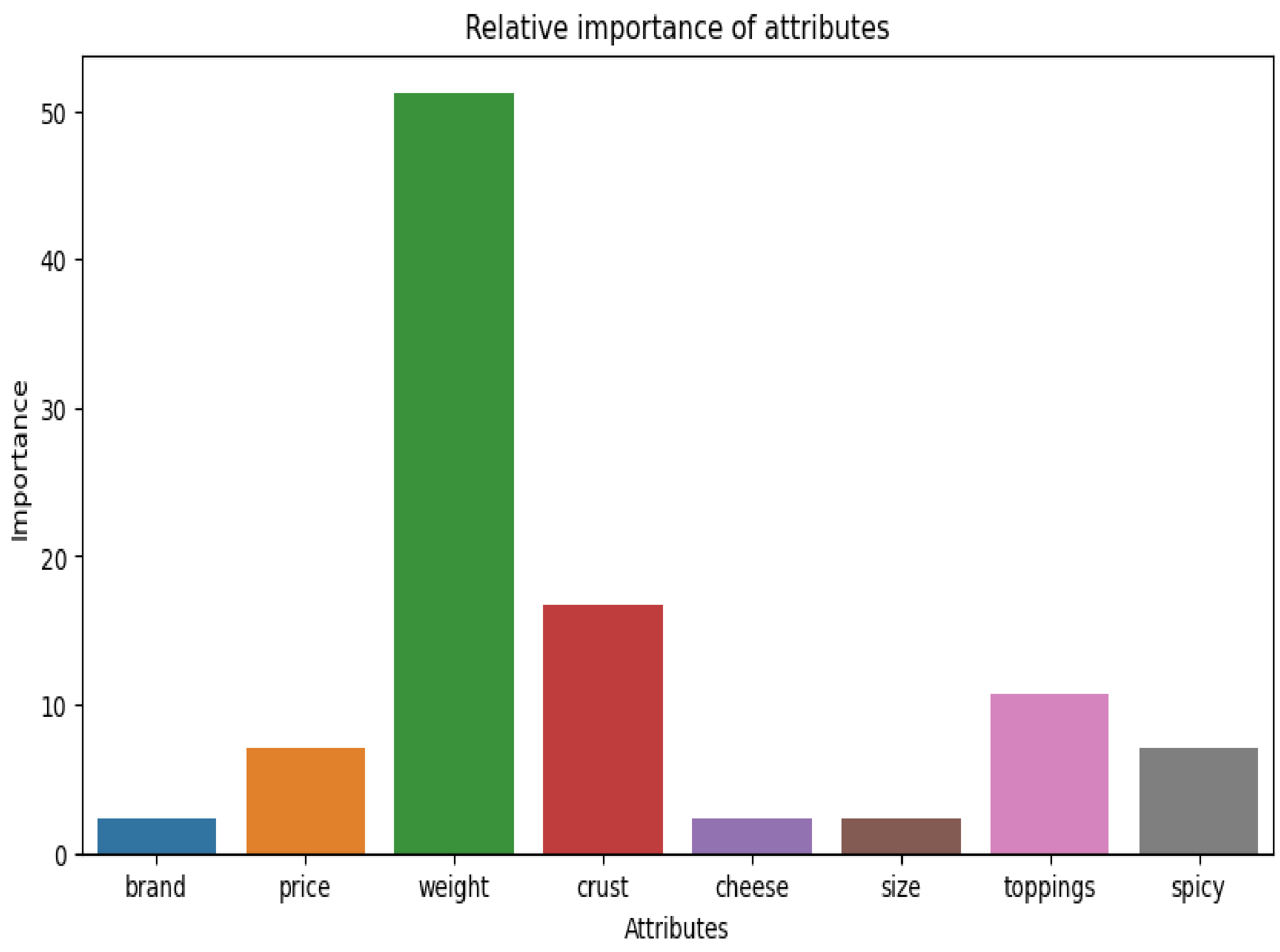
```
          Onesta:7.327471962526033e-15
0
2
          Oven Story:-0.25000000000000555
0
3
          Pizza hut:0.24999999999999933
Attribute : price
    Relative importance of attribute  7.14
    Level wise part worths:
1
0
          $1.00:0.7500000000000084
1
1
          $2.00:-9.992007221626409e-15
1
2
          $3.00:3.9968028886505635e-15
1
3
          $4.00:-0.7500000000000024
Attribute : weight
    Relative importance of attribute  51.19
    Level wise part worths:
2
0
          100g:5.0
2
1
          200g:1.9999999999999973
2
2
          300g:-1.2499999999999947
2
3
          400g:-5.750000000000003
Attribute : crust
    Relative importance of attribute  16.67
    Level wise part worths:
3
0
          thick:1.7499999999999996
3
1
          thin:-1.7499999999999996
Attribute : cheese
    Relative importance of attribute  2.38
    Level wise part worths:
4
0
          Cheddar:-0.2500000000000001
4
```

```
1
          Mozzarella:0.2500000000000001
Attribute : size
    Relative importance of attribute  2.38
    Level wise part worths:
5
0
          large:-0.24999999999999883
5
1
          regular:0.24999999999999883
Attribute : toppings
    Relative importance of attribute  10.71
    Level wise part worths:
6
0
          mushroom:1.1249999999999978
6
1
          paneer:-1.1249999999999978
Attribute : spicy
    Relative importance of attribute  7.14
    Level wise part worths:
7
0
          extra:0.749999999999999
7
1
          normal:-0.749999999999999
```

Out[7]:

```
{'Dominos': -1.1102230246251565e-15,
 'Onesta': 7.327471962526033e-15,
 'Oven Story': -0.25000000000000555,
 'Pizza hut': 0.24999999999999933,
 '$1.00': 0.7500000000000084,
 '$2.00': -9.992007221626409e-15,
 '$3.00': 3.9968028886505635e-15,
 '$4.00': -0.7500000000000024,
 '100g': 5.0,
 '200g': 1.9999999999999973,
 '300g': -1.2499999999999947,
 '400g': -5.750000000000003,
 'thick': 1.7499999999999996,
 'thin': -1.7499999999999996,
 'Cheddar': -0.2500000000000001,
 'Mozzarella': 0.2500000000000001,
 'large': -0.24999999999999883,
 'regular': 0.24999999999999883,
 'mushroom': 1.1249999999999978,
 'paneer': -1.1249999999999978,
 'extra': 0.749999999999999,
 'normal': -0.749999999999999}
```

```
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 weight of pizzas significantly influences customer preferences, with customers placing a high value on differences in weight. The type of crust and toppings are also key factors in customer decision-making. Attributes like brand, cheese, size, and spiciness have lower importance, suggesting that they do not significantly influence customer preferences. To cater to customer preferences, businesses should focus on maximizing pizza weight, diversifying crust varieties, and customizing toppings. Advertising campaigns should highlight highly regarded product characteristics, while pricing should be evaluated to establish competitive prices. Understanding the importance of different pizza characteristics can help businesses make informed decisions to cater to customer needs and enhance overall satisfaction.

- Finding the profile with the highest utility score and preferred level.

```
utility = []
for i in range(df.shape[0]):
    score = 
part_worth_dict[df['brand'][i]]+part_worth_dict[df['price'][i]]+part_worth_dic
t[df['weight'][i]]+part_worth_dict[df['crust'][i]]+part_worth_dict[df['cheese'
][i]]+part_worth_dict[df['size'][i]]+part_worth_dict[df['toppings'][i]]+part_w
orth_dict[df['spicy'][i]]
    utility.append(score)
df['utility'] = utility
utility
print("The profile that has the highest utility score :",'\n',
df.iloc[np.argmax(utility)])
```

```
[2.6250000000000093,
 3.3750000000000044,
 0.37500000000000033,
 -6.375000000000003,
 -0.37500000000000944,
 4.375000000000007,
 -1.3749999999999791,
 -4.624999999999996,
 -3.6250000000000147,
 7.6249999999999885,
 -5.3749999999999885,
 -2.3750000000000133,
 1.3750000000000053,
 6.374999999999997,
 -7.625000000000003,
 5.624999999999996]

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(i)
    #level_name[j]
    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 most preferred profile is the profile with the highest utility score, which includes specific levels for each attribute, maximizing customer satisfaction. The preferred level for each attribute is the level with the highest part-worth utility, providing insight into customer preferences. This can be used in product design, marketing, and pricing strategies. By understanding the most preferred profile and individual attribute levels, businesses can tailor their offerings to meet customer needs, enhancing satisfaction and driving sales.

Recommendation

The business should focus on developing and customizing products that closely match the most preferred profile, such as a pizza from Oven Story, priced at $4.00. This will enhance customer satisfaction and preference. The pricing strategy should consider price sensitivity, offering competitive pricing, discounts, or value bundles to cater to price-sensitive segments. Marketing campaigns should highlight key attributes and tailor brand positioning to highlight unique selling propositions. Product line diversification should include a variety of options, customization options, and a feedback mechanism to maintain quality and consistency. Strategic partnerships with popular brands like Pizza Hut can also be explored for promotional tie-ins or limited-time offers. The implementation plan includes short-term actions like introducing promotional campaigns featuring the most preferred product profile, medium-term actions like developing a diversified product line, and long-term actions like continuously monitoring customer preferences, investing in quality assurance, and exploring strategic partnerships. By following these recommendations, the business can better align its product offerings with customer preferences, enhancing satisfaction, loyalty, and driving sales growth.

R Codes

```
# Load necessary libraries
library(readr)
library(dplyr)
library(caret)
library(broom)
library(ggplot2)

# Load data
df <- read_csv("C:\\Users\\Ferah Shan\\Downloads\\pizza_data.csv")

# Convert price and ranking to numeric
df$price <- as.numeric(gsub("\\$", "", df$price))
df$ranking <- as.numeric(df$ranking)

# Encode categorical variables
df$brand <- as.numeric(factor(df$brand))
df$weight <- as.numeric(factor(df$weight))
df$crust <- as.numeric(factor(df$crust))
df$cheese <- as.numeric(factor(df$cheese))
df$size <- as.numeric(factor(df$size))
df$toppings <- as.numeric(factor(df$toppings))
df$spicy <- as.numeric(factor(df$spicy))

# Define independent variables (X) and dependent variable (y)
X <- df[, c("brand", "price", "weight", "crust", "cheese", "size", "toppings", "spicy")]
y <- df$ranking

# Add a constant to the independent variables
X <- cbind(Intercept = 1, X)

# Fit the linear regression model
model <- lm(y ~ ., data = X)
```

```
# Print model summary
tidy(model)

# Plot coefficients
coefs <- tidy(model)
ggplot(coefs, aes(x = estimate, y = term)) +
  geom_point() +
  labs(x = "Coefficient Value", y = "Feature", title = "Conjoint Analysis - Feature Imp
ortance")
```

Python Codes

```
import pandas as pd, numpy as np
df=pd.read_csv("C:\\Users\\Ferah Shan\\Downloads\\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())

conjoint_attributes = ['brand','price','weight','crust','cheese','size','toppings','spicy']
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)
    #print(part_worth)
    part_worth_range.append(max(new_part_worth) - min(new_part_worth))
    # next iteration
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:
    #print(i)
    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])
        #print(j)
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(i)
    #level_name[j]
    print("Preferred level in {} is :: {}".format(i,level_name[j][important_levels[i]]))
```

References

1. www.github.com

2. www.geeksforgeeks.com

3. www.datacamp.com