# ECON-613 HW #3

*Peter Kim*

*2019-03-18*

## Exercise1

```
#Reading Data
data(margarine)

#Subsetting Data
mar_choicePrice = margarine$choicePrice %>% as.data.frame()
mar_demos = margarine$demos %>% as.data.frame()

#Average and Spread by Columns
avg_col = mar_choicePrice[, 3:12] %>% colMeans()
spread_col = mar_choicePrice[, 3:12] %>% apply(MARGIN = 2, FUN = sd)

#Dataframe of Average and Spread by Columns
avg_spread_col_df = data.frame(
  avg_col,
  spread_col
)
colnames(avg_spread_col_df) = c("average", "spread")

#Table of Average and Spread by Columns
kable(avg_spread_col_df, digits = 4,
      caption = "Table of Average and Spread by Column")
```

Table 1: Table of Average and Spread by Column

| | average | spread |
|---|---|---|
| PPk_Stk | 0.5184 | 0.1505 |
| PBB_Stk | 0.5432 | 0.1203 |
| PFl_Stk | 1.0150 | 0.0429 |
| PHse_Stk | 0.4371 | 0.1188 |
| PGen_Stk | 0.3453 | 0.0352 |
| PImp_Stk | 0.7808 | 0.1146 |
| PSS_Tub | 0.8251 | 0.0612 |
| PPk_Tub | 1.0774 | 0.0297 |
| PFl_Tub | 1.1894 | 0.0141 |
| PHse_Tub | 0.5687 | 0.0725 |

```
#Identifying Unique Brands
brands = sub("_.*", "", colnames(mar_choicePrice))[3:12] %>% unique()

#Indices of Each Brand
index1 = grepl(brands[1], colnames(mar_choicePrice)) %>% which()
index2 = grepl(brands[2], colnames(mar_choicePrice)) %>% which()
```

```
index3 = grepl(brands[3], colnames(mar_choicePrice)) %>% which()
index4 = grepl(brands[4], colnames(mar_choicePrice)) %>% which()
index5 = grepl(brands[5], colnames(mar_choicePrice)) %>% which()
index6 = grepl(brands[6], colnames(mar_choicePrice)) %>% which()
index7 = grepl(brands[7], colnames(mar_choicePrice)) %>% which()

#Average and Spread by Brand
b1 = mar_choicePrice[, index1] %>% gather(key) %>% .[, 2]
b2 = mar_choicePrice[, index2]
b3 = mar_choicePrice[, index3] %>% gather(key) %>% .[, 2]
b4 = mar_choicePrice[, index4] %>% gather(key) %>% .[, 2]
b5 = mar_choicePrice[, index5]
b6 = mar_choicePrice[, index6]
b7 = mar_choicePrice[, index7]

#Table of Average and Spread by Brand
brand_avg_sd_df = data.frame(
  c(brands[1:7]),
  c(mean(b1), mean(b2), mean(b3), mean(b4), mean(b5), mean(b6), mean(b7)),
  c(sd(b1), sd(b2), sd(b3), sd(b4), sd(b5), sd(b6), sd(b7))
)
colnames(brand_avg_sd_df) = c("Brand", "Mean", "Sd")

kable(brand_avg_sd_df, digits = 4,
      caption = "Table of Average and Spread by Brand")
```

Table 2: Table of Average and Spread by Brand

| Brand | Mean | Sd |
|---|---|---|
| PPk | 0.7979 | 0.2998 |
| PBB | 0.5432 | 0.1203 |
| PFl | 1.1022 | 0.0928 |
| PHse | 0.5029 | 0.1184 |
| PGen | 0.3453 | 0.0352 |
| PImp | 0.7808 | 0.1146 |
| PSS | 0.8251 | 0.0612 |

```
#Average and Spread by Stick and Tubs
stick_avg_spread = select(mar_choicePrice, contains("Stk")) %>%
  gather(key = "stk", value = vals) %>%
  summarise(mean = mean(vals), var = sd(vals)^2)
rownames(stick_avg_spread) = "Stick"

tub_avg_spread = select(mar_choicePrice, contains("Tub")) %>%
  gather(key = "tub", value = vals) %>%
  summarise(mean = mean(vals), var = var(vals))
rownames(tub_avg_spread) = "Tub"

avg_spread_char_df = rbind(stick_avg_spread, tub_avg_spread)

kable(avg_spread_char_df, digits = 4,
      caption = "Table of Average and Spread (Var) by Stick and Tubs")
```

Table 3: Table of Average and Spread (Var) by Stick and Tubs

| | mean | var |
|---|---|---|
| Stick | 0.6066 | 0.0622 |
| Tub | 0.9151 | 0.0599 |

```
#Market Share
choice_total = table(mar_choicePrice$choice) %>% sum()
market_share = (table(mar_choicePrice$choice) / choice_total) %>%
  as.data.frame()

#Removing Var1
market_share = market_share %>%
  select(Freq)

#Providing Appropriate Colnames and Rownames
colnames(market_share) = "Market Share"
rownames(market_share) = colnames(mar_choicePrice)[3:12]

#Table of Market Share
kable(market_share, digits = 4,
      caption = "Table of Market Share Based on Products")
```

Table 4: Table of Market Share Based on Products

| | Market Share |
|---|---|
| PPk_Stk | 0.3951 |
| PBB_Stk | 0.1564 |
| PFl_Stk | 0.0544 |
| PHse_Stk | 0.1327 |
| PGen_Stk | 0.0705 |
| PImp_Stk | 0.0166 |
| PSS_Tub | 0.0714 |
| PPk_Tub | 0.0454 |
| PFl_Tub | 0.0503 |
| PHse_Tub | 0.0074 |

```
#Finding Indices of Sticks and Tubs
stk_idx = colnames(mar_choicePrice[3:12]) %>%
  ends_with(match = "stk", ignore.case = TRUE)
tub_idx = colnames(mar_choicePrice[3:12]) %>%
  ends_with(match = "tub", ignore.case = TRUE)

#Calculating the Total Numbers of Sticks and Tubs
stk_tot = table(mar_choicePrice$choice)[stk_idx] %>% sum()
tub_tot = table(mar_choicePrice$choice)[tub_idx] %>% sum()

#Calculating Market Shares Based on Sticks and Tubs
market_share_char = data.frame(
  c(stk_tot / dim(mar_choicePrice)[1], tub_tot / dim(mar_choicePrice)[1])
)
```

```
colnames(market_share_char) = "Market Share"
rownames(market_share_char) = c("Sticks", "Tubs")

#Table of Market Shares Based on Sticks and Tubs
kable(market_share_char, digits = 4,
      caption = "Table of Market Shares Based on Sticks vs. Tubs")
```

Table 5: Table of Market Shares Based on Sticks vs. Tubs

| | Market Share |
|---|---|
| Sticks | 0.8255 |
| Tubs | 0.1745 |

```
#Defining Table of Choices
tb = table(mar_choicePrice$choice)

#Calculating Market Share of Each Brand
mark1 = tb[index1 - 2] %>% sum() / 4470
mark2 = tb[index2 - 2] %>% sum() / 4470
mark3 = tb[index3 - 2] %>% sum() / 4470
mark4 = tb[index4 - 2] %>% sum() / 4470
mark5 = tb[index5 - 2] %>% sum() / 4470
mark6 = tb[index6 - 2] %>% sum() / 4470
mark7 = tb[index7 - 2] %>% sum() / 4470

#Dataframe of Frequency of Each Brand
brand_df = data.frame(
  c(brands[1:7]),
  c(mark1, mark2, mark3, mark4, mark5, mark6, mark7)
)
colnames(brand_df) = c("Brands", "Marketshare")

kable(brand_df, digits = 4, caption = "Market Share by Brand")
```

Table 6: Market Share by Brand

| Brands | Marketshare |
|---|---|
| PPk | 0.4405 |
| PBB | 0.1564 |
| PFl | 0.1047 |
| PHse | 0.1400 |
| PGen | 0.0705 |
| PImp | 0.0166 |
| PSS | 0.0714 |

```
#Counting Choices Based on Income
choices_Income_df = left_join(x = mar_choicePrice, y = mar_demos,
                              by = "hhid") %>% select(choice, Income) %>%
  group_by(choice, Income) %>%
  tally() %>%
  arrange(Income) %>%
```

```
  spread(key = Income, value = n)

#Replacing NA's with 0
choices_Income_df[is.na(choices_Income_df)] = 0

#Calculating Market Share by Income
Income_market_df = data.frame(
  colnames(mar_choicePrice[, 3:12]),
  apply(choices_Income_df[, 2:15], 2, function(x) x / sum(x))
)
colnames(Income_market_df) =
  c("Brands", mar_demos$Income %>% unique() %>% sort())

kable(Income_market_df[, 1:6], digits = 4,
      caption = "Mapping Between Observed Attributes and Choices")
```

Table 7: Mapping Between Observed Attributes and Choices

| Brands | 2.5 | 7.5 | 12.5 | 17.5 | 22.5 |
|---|---|---|---|---|---|
| PPk_Stk | 0.38 | 0.3966 | 0.3960 | 0.4697 | 0.3464 |
| PBB_Stk | 0.08 | 0.1831 | 0.2141 | 0.1477 | 0.1459 |
| PFl_Stk | 0.00 | 0.0441 | 0.0828 | 0.0399 | 0.0403 |
| PHse_Stk | 0.04 | 0.1153 | 0.0889 | 0.1640 | 0.1827 |
| PGen_Stk | 0.12 | 0.0644 | 0.0465 | 0.0310 | 0.1459 |
| PImp_Stk | 0.00 | 0.0068 | 0.0182 | 0.0074 | 0.0024 |
| PSS_Tub | 0.32 | 0.0915 | 0.0808 | 0.0798 | 0.0486 |
| PPk_Tub | 0.02 | 0.0203 | 0.0162 | 0.0281 | 0.0427 |
| PFl_Tub | 0.04 | 0.0746 | 0.0505 | 0.0295 | 0.0356 |
| PHse_Tub | 0.00 | 0.0034 | 0.0061 | 0.0030 | 0.0095 |

```
kable(Income_market_df[, 7:15], digits = 4,
      caption = "Mapping Between Observed Attributes and Choices")
```

Table 8: Mapping Between Observed Attributes and Choices

| 27.5 | 32.5 | 37.5 | 42.5 | 47.5 | 55 | 67.5 | 87.5 | 130 |
|---|---|---|---|---|---|---|---|---|
| 0.4097 | 0.3807 | 0.4731 | 0.4125 | 0.4415 | 0.2338 | 0.3725 | 0.2432 | 0.1923 |
| 0.1975 | 0.1530 | 0.1219 | 0.1089 | 0.1170 | 0.1493 | 0.0784 | 0.2703 | 0.0385 |
| 0.0189 | 0.0510 | 0.0609 | 0.1089 | 0.1223 | 0.0547 | 0.0196 | 0.0811 | 0.1154 |
| 0.1408 | 0.1166 | 0.1039 | 0.0759 | 0.0851 | 0.1592 | 0.1569 | 0.0270 | 0.3077 |
| 0.0378 | 0.0984 | 0.0824 | 0.0198 | 0.0372 | 0.0348 | 0.1176 | 0.0000 | 0.0769 |
| 0.0126 | 0.0073 | 0.0036 | 0.0660 | 0.0904 | 0.0149 | 0.0392 | 0.0270 | 0.0769 |
| 0.0504 | 0.0893 | 0.0538 | 0.0891 | 0.0319 | 0.0597 | 0.1373 | 0.0270 | 0.0000 |
| 0.0525 | 0.0346 | 0.0502 | 0.0693 | 0.0479 | 0.2090 | 0.0588 | 0.0000 | 0.0000 |
| 0.0714 | 0.0601 | 0.0323 | 0.0462 | 0.0106 | 0.0846 | 0.0000 | 0.3243 | 0.1923 |
| 0.0084 | 0.0091 | 0.0179 | 0.0033 | 0.0160 | 0.0000 | 0.0196 | 0.0000 | 0.0000 |

### Comment

For mapping between observed attributes and choices, I wanted to know which product has been purchased the most based on each income category.

## Exercise2

For this exercise, I propose to use a conditional logit model.

According to the class slide, for conditional logit model,

$$p_{ij} = \frac{exp(x_{ij}\beta)}{\sum_{l=1}^{m} exp(x_{il}\beta)}$$

According to page 496 of textbook,

$$L(\beta \mid x_{ij}, y_{ij}) = \Pi_{i=1}^{n} \Pi_{j=1}^{m} p_{ij}^{y_{ij}}$$

But before we calculate the likelihood, we have to calculate the indicator variable $y_{ij}$ for $i \in \{1, ..., 4470\}$ and $j \in \{1, ..., 10\}$. $y$ represents column "choice" in the dataset.

$$y_{ij} = \begin{cases} 1 & y = j \\ 0 & y \neq j \end{cases}$$

```
#Allocating Memory for Matrix y
y = matrix(0, nrow = dim(mar_choicePrice)[1],
           ncol = dim(mar_choicePrice[, 3:12])[2])

#Calculating y
for( j in seq(1, 10) ){
  for( i in 1:dim(mar_choicePrice)[1] ){

    if(mar_choicePrice$choice[i] == j){
        y[i, j] = 1
    }

  }
}
```

Now, we implement the conditional logit likelihood.

```
#Condtional Logit Likelihood Function
cond_logit_lik = function(beta){

  #Intercept Matrix
  alpha = matrix( rep( c(0, beta[1:9]), each = n ), nrow = n, ncol = 10)

  #Non-Intercept Matrix
  xbeta = as.matrix( mar_choicePrice[, 3:12] ) * beta[10]

  #Calculating XB
  XB = alpha + xbeta
```

```
  #Calculating Numerator
  N = exp(XB)

  #Calculating Denominator
  D = exp(XB) %>% rowSums()

  #Calculating Probability p
  p_ij = N / D

  #Calculating Likelihood
  likelihood = prod( p_ij^(y) )

  return(likelihood)

}
```

The log likelihood of the conditional logit model is the following. Note that negative of the log likelihood value is returned.

```
#Conditional Logit Log Likelihood
cond_logit_log_lik = function(beta){

  #Intercept Matrix
  alpha = matrix( rep( c(0, beta[1:9]), each = n ), nrow = n, ncol = 10)

  #Non-Intercept Matrix
  xbeta = as.matrix( mar_choicePrice[, 3:12] ) * beta[10]

  #Calculating XB
  XB = alpha + xbeta

  #Calculating Numerator
  N = exp(XB)

  #Calculating Denominator
  D = exp(XB) %>% rowSums()

  #Calculating Probability p
  p_ij = N / D

  #Calculating log Likelihood
  log_lik = sum( y * log( p_ij ) )

  return(-log_lik)

}
```

Since the likelihood and loglikelihood functions are defined above, we now optimize the model and present the results. Note that $\hat{\alpha}$'s represent intercepts.

```
#Dimension Calculation for Following Intercept Matrix
n = nrow( mar_choicePrice )

#Initializing Parameters
b = rep(-1.8, 10)
```

```r
#Optimizing Log Likelihood Function
opt = optim(par = b, fn = cond_logit_log_lik)

cond_opt_df = data.frame(
  opt$par
  )
colnames(cond_opt_df) = c("Optimized Coefficient")
rownames(cond_opt_df) = c("$\\hat{\\alpha}_{1}$", "$\\hat{\\alpha}_{2}$",
                      "$\\hat{\\alpha}_{3}$", "$\\hat{\\alpha}_{4}$",
                      "$\\hat{\\alpha}_{5}$", "$\\hat{\\alpha}_{6}$",
                      "$\\hat{\\alpha}_{7}$", "$\\hat{\\alpha}_{8}$",
                      "$\\hat{\\alpha}_{9}$", "$\\hat{\\beta}_{price}$")

kable(cond_opt_df, digits = 4, caption = "Price Coefficient Under Conditional Logit")
```

Table 9: Price Coefficient Under Conditional Logit

| | Optimized Coefficient |
|---|---|
| $\hat{\alpha}_1$ | -0.8600 |
| $\hat{\alpha}_2$ | 1.2275 |
| $\hat{\alpha}_3$ | -1.6640 |
| $\hat{\alpha}_4$ | -2.7915 |
| $\hat{\alpha}_5$ | -1.8773 |
| $\hat{\alpha}_6$ | 0.1990 |
| $\hat{\alpha}_7$ | 1.3123 |
| $\hat{\alpha}_8$ | 1.9900 |
| $\hat{\alpha}_9$ | -3.8600 |
| $\hat{\beta}_{price}$ | -6.3206 |

### Interpretation

According to the table, $\hat{\beta}_{price} = -6.3206$. This means that the likelihood of product being purchased decreases as the price increases.

## Exercise3

For this exercise, I propose to use multinomial logit model.

According to page 494 of the textbook, for multinomial logit model,

$$p_{ij} = \frac{exp(\alpha_j + \beta_{Ij} I_i)}{\sum_{k=1}^{m} exp(\alpha_k + \beta_{Ik} I_i)}$$

where $I$ denotes income.

According to page 496 of textbook,

$$L(\beta \mid x_{ij}, y_{ij}) = \Pi_{i=1}^{n} \Pi_{j=1}^{m} p_{ij}^{y_{ij}}$$

Since we have calculated y in exercise 2, we proceed to implementing the multinomial logit likelihood. We also merge choicePrice data frame and demos data frame here.

```
#Merging Data to Put Choice and Income in One Data Frame
merged_data = left_join(x = mar_choicePrice, y = mar_demos, by = "hhid")

#Choice and Income Data Frame
choice_I_df = merged_data %>%
  select(choice, Income) %>%
  arrange(Income)

#Defining Variable I
I = choice_I_df$Income
```

```
#Multinomial Logit Likelihood
multi_logit_lik = function(beta){

  #Intercept Matrix
  alpha = matrix( rep( beta[1:10], each = n ), nrow = n, ncol = 10 )

  #Non-Intercept Matrix
  xbeta = matrix( rep( beta[11:20], each = n), nrow = n, ncol = 10 ) * I

  #Calculating XB
  XB = alpha + xbeta

  #Calculating Numerator
  N = exp(XB)

  #Calculating Denominator
  D = exp(XB) %>% rowSums()

  #Calculating Probability p
  p_ij = N / D

  #Calculating Likelihood
  likelihood = prod( p_ij^(y) )

  return(likelihood)

}
```

We now calculate the log likelihood.

```
#Multinomial Logit Log Likelihood
multi_logit_log_lik = function(betas){

  #Intercept Matrix
  alpha = matrix( rep( betas[1:10], each = n), nrow = n, ncol = 10 )

  #Non-Intercept Matrix
  xbeta = matrix( rep( betas[11:20], each = n), nrow = n, ncol = 10 ) * I

  #Calculating XB
  XB = alpha + xbeta
```

```
  #Calculating Numerator
  N = exp(XB)

  #Calculating Denominator
  D = exp(XB) %>% rowSums()

  #Calculating Probability p
  p_ij = N / D

  #Calculating log Likelihood
  log_logit = sum( y*log( p_ij ) )

  return(-log_logit)

}
```

Since the likelihood and loglikelihood functions are defined above, we now optimize the model.

```
betas = c(0, -0.5, -2.5, -1.5, -1.5, -4, -1.5, -3, -2.5, -4,
          rep(0, 10))

I = choice_I_df$Income

#Optimizing Log Likelihood Function
multi_opt = optim(par = betas, fn = multi_logit_log_lik)

multi_opt_df = data.frame(
  multi_opt$par[1:10],
  multi_opt$par[11:20]
  )
colnames(multi_opt_df) = c("Optimized $\\hat{\\alpha}$", "Optimzed $\\hat{\\beta}$")
rownames(multi_opt_df) = c("Reference", seq(2, 10))

kable(multi_opt_df, digits = 4, caption = "Coefficients Under Multinomial Logit")
```

Table 10: Coefficients Under Multinomial Logit

| | Optimized $\hat{\alpha}$ | Optimzed $\hat{\beta}$ |
|---|---|---|
| Reference | -0.1616 | 0.0034 |
| 2 | -0.9018 | -0.0005 |
| 3 | -1.5586 | -0.0175 |
| 4 | -1.3626 | 0.0058 |
| 5 | -2.1882 | 0.0163 |
| 6 | -3.4089 | 0.0012 |
| 7 | -1.8481 | 0.0025 |
| 8 | -2.0928 | -0.0082 |
| 9 | -2.0756 | -0.0066 |
| 10 | -4.1647 | 0.0020 |

### Interpretation

$\beta_{Income_2}$: It is more likely for an individual to choose choice 1 than choice 2 if his or her income increases.

$\beta_{Income_3}$: It is more likely for an individual to choose choice 1 than choice 3 if his or her income increases.

$\beta_{Income_4}$: It is more likely for an individual to choose choice 4 than choice 1 if his or her income increases.

$\beta_{Income_5}$: It is more likely for an individual to choose choice 5 than choice 1 if his or her income increases.

$\beta_{Income_6}$: It is more likely for an individual to choose choice 6 than choice 1 if his or her income increases.

$\beta_{Income_7}$: It is more likely for an individual to choose choice 7 than choice 1 if his or her income increases.

$\beta_{Income_8}$: It is more likely for an individual to choose choice 1 than choice 8 if his or her income increases.

$\beta_{Income_9}$: It is more likely for an individual to choose choice 1 than choice 9 if his or her income increases.

$\beta_{Income_{10}}$: It is more likely for an individual to choose choice 10 than choice 1 if his or her income increases.

## Exercise4

For conditional logit, the marginal effect is computed as the following.

```
#Conditional Logit Log Likelihood
cond_logit_log_lik2 = function(beta){

  #Intercept Matrix
  alpha = matrix( rep( c(0, beta[1:9]), each = n ), nrow = n, ncol = 10)

  #Non-Intercept Matrix
  xbeta = as.matrix( mar_choicePrice[, 3:12] ) * beta[10]

  #Calculating XB
  XB = alpha + xbeta

  #Calculating Numerator
  N = exp(XB)

  #Calculating Denominator
  D = exp(XB) %>% rowSums()

  #Calculating Probability p
  p_ij = N / D

  #Calculating log Likelihood
  log_lik = sum( y * log( p_ij ) )
```

```
  return(p_ij)

}

#Saving P from Conditional Logit Log Likelihood
P = cond_logit_log_lik2(cond_opt_df$`Optimized Coefficient`)

#Saving Price Beta from Estimated Coefficient
beta = cond_opt_df$`Optimized Coefficient`[10]

#Allocating Memory for Conditional Marginal
conditional_marginal = matrix(NA, nrow = 10, ncol = 10)

#Calculating Average Marginal Effect
for( j in 1:10 ){
  for( k in 1:10 ){

    delta = ifelse(j == k, 1, 0)
    conditional_marginal[j, k] = mean(P[, j] * (delta - P[, k]) * beta)

  }
}

cond_marg_df = conditional_marginal %>% as.data.frame()

kable(cond_marg_df[,1:5], digits = 4, caption =
        "Average Marginal Effect Under Conditional")
```

Table 11: Average Marginal Effect Under Conditional

| V1 | V2 | V3 | V4 | V5 |
|---|---|---|---|---|
| -1.2343 | 0.3009 | 0.1235 | 0.2806 | 0.1530 |
| 0.3009 | -0.7510 | 0.0599 | 0.1340 | 0.0756 |
| 0.1235 | 0.0599 | -0.3437 | 0.0515 | 0.0315 |
| 0.2806 | 0.1340 | 0.0515 | -0.6809 | 0.0625 |
| 0.1530 | 0.0756 | 0.0315 | 0.0625 | -0.4186 |
| 0.0263 | 0.0125 | 0.0053 | 0.0115 | 0.0063 |
| 0.1498 | 0.0718 | 0.0304 | 0.0619 | 0.0377 |
| 0.0949 | 0.0461 | 0.0200 | 0.0376 | 0.0244 |
| 0.0892 | 0.0434 | 0.0186 | 0.0355 | 0.0233 |
| 0.0161 | 0.0070 | 0.0031 | 0.0057 | 0.0043 |

```
kable(cond_marg_df[,6:10], digits = 4, caption =
        "Average Marginal Effect Under Conditional")
```

Table 12: Average Marginal Effect Under Conditional

| V6 | V7 | V8 | V9 | V10 |
|---|---|---|---|---|
| 0.0263 | 0.1498 | 0.0949 | 0.0892 | 0.0161 |
| 0.0125 | 0.0718 | 0.0461 | 0.0434 | 0.0070 |
| 0.0053 | 0.0304 | 0.0200 | 0.0186 | 0.0031 |
| 0.0115 | 0.0619 | 0.0376 | 0.0355 | 0.0057 |

| V6 | V7 | V8 | V9 | V10 |
|---|---|---|---|---|
| 0.0063 | 0.0377 | 0.0244 | 0.0233 | 0.0043 |
| -0.0760 | 0.0061 | 0.0038 | 0.0036 | 0.0006 |
| 0.0061 | -0.4092 | 0.0249 | 0.0227 | 0.0041 |
| 0.0038 | 0.0249 | -0.2702 | 0.0157 | 0.0028 |
| 0.0036 | 0.0227 | 0.0157 | -0.2545 | 0.0026 |
| 0.0006 | 0.0041 | 0.0028 | 0.0026 | -0.0461 |

## Interpretation

Each increase in price decreases the probability of choosing jth choice but increases the probability of choosing non-jth choice.

For multinomial logit, the marginal effect is calculated as the following.

```
#Multinomial Logit Log Likelihood
multi_logit_log_lik2 = function(betas){

  #Intercept Matrix
  alpha = matrix( rep( betas[1:10], each = n), nrow = n, ncol = 10 )

  #Non-Intercept Matrix
  xbeta = matrix( rep( betas[11:20], each = n), nrow = n, ncol = 10 ) * I

  #Calculating XB
  XB = alpha + xbeta

  #Calculating Numerator
  N = exp(XB)

  #Calculating Denominator
  D = exp(XB) %>% rowSums()

  #Calculating Probability p
  p_ij = N / D

  #Calculating log Likelihood
  log_logit = sum( y*log( p_ij ) )

  return(p_ij)

}

#Probability from Multinomial
multi_P = multi_logit_log_lik2(multi_opt$par)

#Betas
multi_beta = multi_opt$par[11:20]

#Calculating Beta-Bar
beta_bar = multi_P %*% (multi_beta)

#Allocating Memory for Multinomial Marginal Effect
```

```r
multi_marginal = rep(NA, 10)

#Calculating Multinomial Marginal Effect
for(j in 1:10){
  multi_marginal[j] = mean(multi_P[, j] * (multi_opt$par[11:20] - beta_bar))
}

multi_mar_df = multi_marginal %>% as.data.frame()

kable(multi_mar_df, digits = 4, col.names = "Average Marginal Effect",
      caption = "Average Marginal Effect Under Multinomial Logit")
```

Table 13: Average Marginal Effect Under Multinomial Logit

| Average Marginal Effect |
|---:|
| -8e-04 |
| -3e-04 |
| -1e-04 |
| -3e-04 |
| -2e-04 |
| 0e+00 |
| -1e-04 |
| -1e-04 |
| -1e-04 |
| 0e+00 |

### Interpretaion

Each one-unit increase in income changes the probability of selecting jth choice by the given magnitude.

## Exercise5

The following is the mixed logit log likelihood function.

```r
mixed_logit_log_lik = function(beta){

  #Intercept Matrix
  alpha = matrix( rep( beta[1:10], each = n ), nrow = n, ncol = 10)

  #Non-Intercept Matrix
  xbeta_I = matrix( rep( beta[11:20], each = n), nrow = n, ncol = 10 ) * I
  xbeta_p = as.matrix( mar_choicePrice[, 3:12] ) * beta[21]

  #Calculating XB
  XB = alpha + xbeta_p + xbeta_I

  #Calculating Numerator
  N = exp(XB)

  #Calculating Denominator
```

```
  D = exp(XB) %>% rowSums()

  #Calculating Probability p
  p_ij = N / D

  #Calculating log Likelihood
  log_lik = sum( y * log( p_ij ) )

  return(-log_lik)
}
```

Here, we optimize the log likelihood function.

```
#Initial Values
betas = c(0, -1, 1, -2, -3, -2, 0.5, 1, 2, -4,
          rep(0, 10), -6)

#Defining I
I = choice_I_df$Income

#Optimizing Log Likelihood Function
opt1 = optim(par = betas, fn = mixed_logit_log_lik)

opt_df = data.frame(
  c(opt1$par[1:10], opt1$par[11:20], opt1$par[21])
  )
colnames(opt_df) = c("Optimzed $\\hat{\\beta}^{f}$")
rownames(opt_df) = c("Intercept Reference",
                     paste("Intercept", 2:10, sep = ""),
                     "Income Reference",
                     paste("Income", 2:10, sep = ""),
                           "Price")

kable(opt_df, digits = 4, caption = "Coefficients Under Mixed Logit")
```

Table 14: Coefficients Under Mixed Logit

| | Optimzed $\hat{\beta}^f$ |
|---|---|
| Intercept Reference | -0.0657 |
| Intercept2 | -1.0966 |
| Intercept3 | 1.2344 |
| Intercept4 | -1.6447 |
| Intercept5 | -2.9809 |
| Intercept6 | -2.0062 |
| Intercept7 | 0.4405 |
| Intercept8 | 1.3248 |
| Intercept9 | 2.1570 |
| Intercept10 | -4.0409 |
| Income Reference | 0.0011 |
| Income2 | 0.0036 |
| Income3 | -0.0095 |
| Income4 | -0.0036 |
| Income5 | 0.0032 |
| Income6 | 0.0053 |

| | Optimzed $\hat{\beta}^f$ |
|---|---|
| Income7 | -0.0066 |
| Income8 | -0.0049 |
| Income9 | -0.0007 |
| Income10 | 0.0106 |
| Price | -6.3491 |

Here, we also implement mixed logit log likelihood. But, this time, we compute the mixed logit log likelihood after removing choice 10.

```
#Dataframe After Removing Choice 1
choicePrice2 = merged_data %>%
  filter(choice != 10)

n2 = nrow(choicePrice2)

#Allocating Memory for Matrix y
y2 = matrix(0, nrow = dim(choicePrice2)[1],
            ncol = dim(choicePrice2[, 3:11])[2])

#Calculating y
for( j in seq(1, 10) ){
  for( i in 1:dim(choicePrice2)[1] ){

    if(choicePrice2$choice[i] == j){
        y2[i, j] = 1
    }

  }
}

#New Mixed Logit Log Likelihood
mixed_logit_log_lik2 = function(beta){

  #Intercept Matrix
  alpha = matrix( rep( beta[1:9], each = n2 ), nrow = n2, ncol = 9)

  #Non-Intercept Matrix
  xbeta_I = matrix( rep( beta[11:19], each = n2), nrow = n2, ncol = 9 ) *
    choicePrice2$Income
  xbeta_p = as.matrix( choicePrice2[, 3:11] ) * beta[21]

  #Calculating XB
  XB = alpha + xbeta_p + xbeta_I

  #Calculating Numerator
  N = exp(XB)

  #Calculating Denominator
  D = exp(XB) %>% rowSums()

  #Calculating Probability p
  p_ij = N / D
```

```
  #Calculating log Likelihood
  log_lik = sum( y2 * log( p_ij ) )

  return(-log_lik)
}
```

We optimize the removed mixed logit log likelihood.

```
#Initial Values
betas = c(0, -1, 1, -2, -3, -2, 0.5, 1, 2, -4,
          rep(0, 10), -6)

#Defining I
I = choice_I_df$Income

#Optimizing Log Likelihood Function
opt2 = optim(par = betas, fn = mixed_logit_log_lik2)

opt2_df = data.frame(
  c(opt2$par[1:9], opt2$par[11:19], opt2$par[21])
  )
colnames(opt2_df) = c("Optimzed $\\hat{\\beta}^{r}$")
rownames(opt2_df) = c("Intercept Reference",
                      paste("Intercept", 2:9, sep = ""),
                      "Income Reference",
                      paste("Income", 2:9, sep = ""),
                            "Price")

kable(opt2_df, digits = 4, caption = "Coefficients Under Removed Mixed Logit")
```

Table 15: Coefficients Under Removed Mixed Logit

| | Optimzed $\hat{\beta}^r$ |
|---|---|
| Intercept Reference | -0.1349 |
| Intercept2 | -0.8298 |
| Intercept3 | 1.1196 |
| Intercept4 | -1.7036 |
| Intercept5 | -2.8424 |
| Intercept6 | -1.5126 |
| Intercept7 | 0.4236 |
| Intercept8 | 0.9796 |
| Intercept9 | 2.0424 |
| Income Reference | 0.0067 |
| Income2 | 0.0001 |
| Income3 | 0.0053 |
| Income4 | 0.0020 |
| Income5 | 0.0030 |
| Income6 | 0.0078 |
| Income7 | -0.0033 |
| Income8 | 0.0148 |
| Income9 | 0.0093 |
| Price | -6.4081 |

We compute test statistics here. Note that the deviance follows a $\chi^2$ distribution.

```
#Calculating Test Statistic
MTT = -2* (mixed_logit_log_lik2(opt1$par) -
             mixed_logit_log_lik2(opt2$par))

#Table of Test Statistics
kable(data.frame(MTT), digits = 4, caption = "Test Stiatistics",
      col.names = "MTT")
```

Table 16: Test Stiatistics

| MTT |
|---|
| -83.6055 |

```
#Acceptance / Rejection
pchisq(MTT, df = length(opt2$par[c(1:9, 11:19, 21)]), lower.tail = F)
```

```
## [1] 1
```

## Conclusion

Since the test shows p-value of 1, IIA holds.