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 = var)

#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 Product Brand")
```

Table 1: Table of Average and Spread by Product Brand

	average	spread
PPk_Stk	0.5184	0.0227
PBB_Stk	0.5432	0.0145
PFl_Stk	1.0150	0.0018
PHse_Stk	0.4371	0.0141
PGen_Stk	0.3453	0.0012
PImp_Stk	0.7808	0.0131
PSS_Tub	0.8251	0.0037
PPk_Tub	1.0774	0.0009
PFl_Tub	1.1894	0.0002
PHse_Tub	0.5687	0.0052

```
#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"
```

```r
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 2: Table of Average and Spread (Var) by Stick and Tubs

	mean	var
Stick	0.6066	0.0622
Tub	0.9151	0.0599

```r
#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 3: 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

```r
#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]) %>%
```

```r
  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 4: Table of Market Shares Based on Sticks vs. Tubs

	Market Share
Sticks	0.8255
Tubs	0.1745

```r
#Creating Mode Function to Map Between Observed Attributed and Choices
getmode = function(v) {
   uniq = unique(v)
   return( uniq[which.max(tabulate(match(v, uniq)))] )
}

#Creating getcolNames Function to Convert Results to Names
getcolNames = function(m) {

  names = colnames(mar_choicePrice[3:12])[m] %>% unlist()

  return(names)
}

#Finding Mode of Choices Based on Income Categories
map_df = left_join(x = mar_choicePrice, y = mar_demos, by = "hhid") %>%
  select(choice, Income) %>%
  group_by(Income) %>%
  summarise( mode = getmode(choice) )

#Adding a Column of Product Mostly Purchased
map_df = map_df %>%
  mutate(
    map_chr(map_df$mode, getcolNames)
  )
colnames(map_df) = c("Income", "Mode", "Product Mostly Purchased")

kable(map_df, digits = 4,
      caption = "Mapping Between Observed Attributes and Choices")
```

Table 5: Mapping Between Observed Attributes and Choices

Income	Mode	Product Mostly Purchased
2.5	1	PPk_Stk
7.5	1	PPk_Stk
12.5	1	PPk_Stk
17.5	1	PPk_Stk
22.5	1	PPk_Stk
27.5	1	PPk_Stk
32.5	1	PPk_Stk
37.5	1	PPk_Stk
42.5	1	PPk_Stk
47.5	1	PPk_Stk
55.0	1	PPk_Stk
67.5	1	PPk_Stk
87.5	9	PFl_Tub
130.0	4	PHse_Stk

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] ){
```

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

  }
}
```

Now, we implement the conditional logit likelihood.

```r
#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.

```r
#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)

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

opt_df = data.frame(
  opt$par
  )
colnames(opt_df) = c("Optimized Coefficient")
rownames(opt_df) = c("$\\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(opt_df, digits = 4, caption = "Price Coefficient Under Conditional Logit")
```

Table 6: 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
opt = optim(par = betas, fn = multi_logit_log_lik)

opt_df = data.frame(
  opt$par[1:10],
  opt$par[11:20]
  )
colnames(opt_df) = c("Optimized $\\hat{\\alpha}$", "Optimzed $\\hat{\\beta}$")
```

```r
rownames(opt_df) = c("Reference", seq(2, 10))

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

Table 7: 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

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

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

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

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

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

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

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

β_{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

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.

```r
#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 8: 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
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
```

```r
  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.

```r
#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 9: Coefficients Under Removed Mixed Logit

	Optimzed $\hat{\beta}^{r}$
Intercept Reference	-0.1349
Intercept2	-0.8298

	Optimzed $\hat{\beta}^r$
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 χ^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 10: 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.