ECON 613 Data and Multinomial Choices

Xin Lin

3/7/2022

```
# Load packages
library(tidyverse)
library(readxl)
library(magrittr)
library(janitor)
library(knitr)
library(kableExtra)
library(plm)
library(lmtest)

# abadon scientific notation in R
options(scipen = 999)

# set the starting number used to generate random sample
set.seed(999)

# import dataset
datjss <- read_csv("data/datjss.csv")
datsss <- read_csv("data/datsss.csv")
datstu <- read_csv("data/datstu_v2.csv",
    col_types = cols(score = col_number(),
        rankplace = col_number()))
```

Exercise 1 Basic Statistics

1. Calculate the number of students, schools, programs

```
# the number of students
n_students <- nrow(datstu)
n_students
```

```
## [1] 340823
```

```
# the number of schools
dat_school <- datsss %>%
  group_by(schoolcode) %>%
  summarise(n=n())
n_schools <- nrow(dat_school)
n_schools
```

```
## [1] 898
```

```
# the number of programs
n_programs <- length(unique(unlist(datstu[,11:16])))-1 # exclude 'NA'
n_programs
```

```
## [1] 32
```

2. Calculate the number of choices

```
# convert data from wide to long (for use in the following analysis as well)
school <- datstu %>%
  select(schoolcode1:schoolcode6, jssdistrict, score, rankplace, agey, male) %>%
  mutate(student_id = 1:n(),) %>%
  pivot_longer(cols = schoolcode1:schoolcode6,
               names_to = "schoolchoice",
               values_to = "school")

program <- datstu %>%
  select(choicepgm1:choicepgm6) %>%
  pivot_longer(cols = choicepgm1:choicepgm6,
               names_to = "programchoice",
               values_to = "program")

dat_long <- cbind.data.frame(school, program) %>%
  mutate(choice = paste(school, program))

# count the unique choices (school, program)
choices <- dat_long %>%
  group_by(choice) %>%
  count
n_choices <- nrow(choices) # count unique
n_choices
```

```
## [1] 3086
```

Answer:

	Counts
Students	340823
Schools	898
Programs	32
Choices	3086

3. Calculate the number of students applying to at least one senior high schools in the same district to home

```r
# create a new dataset containing only school and school district: schoolcode, sssdistrict
school_sssdistrict <- datsss %>%
  na.omit() %>%
  select(schoolcode, sssdistrict) %>%
  rename(school = schoolcode) %>%
  group_by_all() %>%
  unique()

# merge the above two datasets
# var: student_id, schoolcode1-6, jssdistrict, sssdistrict
dat_long <- left_join(dat_long, school_sssdistrict, by = "school")

# create a new variable: same_dist, equal to 1 if jssdistrict==sssdistrict; 0 o/w
school_same_dist <- dat_long %>%
  mutate(same_dist=ifelse(jssdistrict==sssdistrict, 1, 0)) %>%
  group_by(student_id) %>%
  summarise(n = sum(same_dist)) %>%
  filter(n>0)
n_same_dist <- nrow(school_same_dist)
n_same_dist
```

```
## [1] 250826
```

Answer: The number of students applying to at least one senior high schools in the same district to home is **250,826**.

4. Calculate the number of students each senior high school admitted

```r
admit <- datstu %>%
  filter(rankplace != 99 & !is.na(rankplace)) %>%
  select(V1, schoolcode1:schoolcode6, rankplace)

admit$schooladmit <- admit$schoolcode1
admit$schooladmit[admit$rankplace==2] <- admit$schoolcode2[admit$rankplace==2]
admit$schooladmit[admit$rankplace==3] <- admit$schoolcode3[admit$rankplace==3]
admit$schooladmit[admit$rankplace==4] <- admit$schoolcode4[admit$rankplace==4]
admit$schooladmit[admit$rankplace==5] <- admit$schoolcode5[admit$rankplace==5]
admit$schooladmit[admit$rankplace==6] <- admit$schoolcode6[admit$rankplace==6]

size <- admit %>%
  select(schooladmit) %>%
  group_by(schooladmit) %>%
  mutate(size = n())%>%
  rename(school = schooladmit) %>%
  distinct(school, .keep_all = TRUE) %>%
  ungroup() %>%
  arrange(school)
```

Answer: Here are the first ten entries

School	Size
10101	398
10102	248
10103	443
10104	220
10105	346
10106	395
10107	306
10108	318
10109	300
10110	535

5. Calculate the cutoff of senior high schools (the lowest score to be admitted)

```
cutoff <- datstu %>%
  filter(!is.na(score))%>%
  filter(rankplace != 99 & !is.na(rankplace))

cutoff$schooladmit <- cutoff$schoolcode1
cutoff$schooladmit[cutoff$rankplace==2] <- cutoff$schoolcode2[cutoff$rankplace==2]
cutoff$schooladmit[cutoff$rankplace==3] <- cutoff$schoolcode3[cutoff$rankplace==3]
cutoff$schooladmit[cutoff$rankplace==4] <- cutoff$schoolcode4[cutoff$rankplace==4]
cutoff$schooladmit[cutoff$rankplace==5] <- cutoff$schoolcode5[cutoff$rankplace==5]
cutoff$schooladmit[cutoff$rankplace==6] <- cutoff$schoolcode6[cutoff$rankplace==6]

cutoff <- cutoff %>%
  select(schooladmit, score)%>%
  group_by(schooladmit) %>%
  mutate(cutoff = min(score)) %>%
  distinct(cutoff, .keep_all = TRUE) %>%
  ungroup() %>%
  rename(school = schooladmit) %>%
  select(school, cutoff) %>%
  arrange(school)
```

Answer: Here are the first ten entries

School	Cutoff
10101	284
10102	343
10103	316
10104	245
10105	260
10106	293
10107	281
10108	248
10109	257
10110	343

6. Calculate the quality of senior high schools (the average score of students admitted)

```
quality <- datstu %>%
  filter(!is.na(score))%>%
  filter(rankplace != 99 & !is.na(rankplace))

quality$schooladmit <- quality$schoolcode1
quality$schooladmit[quality$rankplace==2] <- quality$schoolcode2[quality$rankplace==2]
quality$schooladmit[quality$rankplace==3] <- quality$schoolcode3[quality$rankplace==3]
quality$schooladmit[quality$rankplace==4] <- quality$schoolcode4[quality$rankplace==4]
quality$schooladmit[quality$rankplace==5] <- quality$schoolcode5[quality$rankplace==5]
quality$schooladmit[quality$rankplace==6] <- quality$schoolcode6[quality$rankplace==6]

quality <- quality %>%
  select(schooladmit, score) %>%
  group_by(schooladmit) %>%
  mutate(quality = mean(score)) %>%
  distinct(quality, .keep_all = TRUE) %>%
  ungroup() %>%
  rename(school = schooladmit) %>%
  select(school, quality) %>%
  arrange(school)
```

Answer: Here are the first ten entries

School	Quality
10101	320.2312
10102	394.1492
10103	353.8330
10104	296.9182
10105	351.2139
10106	340.1013
10107	311.9542
10108	303.9025
10109	281.8233
10110	408.0785

Exercise 2 Data

```
# create a dataset containing schools' longitude & latitude
district <- datsss %>%
  na.omit() %>%
  select(schoolcode, ssslong, sssslat) %>%
  group_by_all() %>%
  unique() %>%
  rename(school = schoolcode)

# merge the long dataset with district,
dat_long <- dat_long %>%
```

```
  left_join(district, by = "school") %>%
  left_join(cutoff, by = "school") %>%
  left_join(quality, by = "school") %>%
  left_join(size, by = "school")

# keep only the variable required
dat_school_program <- dat_long %>%
  select(choice, sssdistrict, ssslong, ssslat, cutoff, quality, size) %>%
  distinct(choice, .keep_all = TRUE) %>%
  arrange(choice)
```

Answer: Here are the first ten entries of the new school-level dataset

choice	sssdistrict	ssslong	ssslat	cutoff	quality	size
100101 General Arts	Wa Municipal	-2.28503	10.03062	198	238.1250	168
100101 Home Economics	Wa Municipal	-2.28503	10.03062	198	238.1250	168
100101 NA	Wa Municipal	-2.28503	10.03062	198	238.1250	168
100101 Technical	Wa Municipal	-2.28503	10.03062	198	238.1250	168
100102 Agriculture	Wa Municipal	-2.28503	10.03062	250	296.4956	450
100102 Business	Wa Municipal	-2.28503	10.03062	250	296.4956	450
100102 General Arts	Wa Municipal	-2.28503	10.03062	250	296.4956	450
100102 General Science	Wa Municipal	-2.28503	10.03062	250	296.4956	450
100102 Home Economics	Wa Municipal	-2.28503	10.03062	250	296.4956	450
100102 Visual Arts	Wa Municipal	-2.28503	10.03062	250	296.4956	450

Exercise 3 Distance

```
# merge the long data with datjss
dat_long <- dat_long %>%
  left_join(datjss, by = "jssdistrict") %>%
  select(-X1) %>%
  rename(jsslong = point_x,
         jsslat = point_y)

# calculate the distance
long_dif <- dat_long$ssslong - dat_long$jsslong
lat_dif <- dat_long$ssslat - dat_long$jsslat
dat_long$distance <- sqrt((69.172*long_dif*cos(dat_long$jsslat/57.3))^2 + (69.172*(lat_dif))^2)
```

Answer: Here are the first ten entries of distance for each of students' choices

student_id	choice	distance
1	50112 Home Economics	8.813579
1	50107 General Arts	8.813579
1	50202 Visual Arts	18.895053
1	50202 Visual Arts	18.895053
1	50702 Home Economics	17.179653
1	50901 General Arts	63.917746
2	70102 General Arts	0.000000
2	70602 Business	21.672792
2	70107 General Arts	0.000000
2	70105 General Arts	0.000000

Exercise 4 Dimensionality Reduction

1. Recode the schoolcode into its first three digits (substr)

```
# create a variable: scode_rev
dat_long <- dat_long %>%
  mutate(scode_rev = substr(school, 1, 3))
```

2. Recode the program variable into 4 categories: arts (general arts and visual arts), economics (business and home economics), science (general science) and others

```
# create the new variable: pgm_rev
dat_long <- dat_long %>%
  mutate(pgm_rev = ifelse(program %in% c("General Arts", "Visual Arts"), "arts",
                          ifelse(program %in% c("Business", "Home Economics"), "economics",
                                 ifelse(program %in% c("General Science"), "science", "others"))))
```

3. Create a new choice variable choice_rev

```
# create the new variable: choice_rev
dat_long <- dat_long %>%
  mutate(choice_rev = paste(scode_rev, pgm_rev))
```

4. Recalculate the cutoff and the quality for each recoded choice

```
# calculate the cutoff and the quality for each recoded choice
quality_cutoff <- dat_long %>%
  filter(rankplace != 99 & !is.na(rankplace)) %>%
  group_by(choice_rev) %>%
  summarise(cutoff = min(score),
            quality = round(mean(score), digits = 2))

# merge the quality and cutoff to the dat_long
```

```
dat <- dat_long %>%
  filter(!is.na(scode_rev), !is.na(pgm_rev)) %>%
  select(-cutoff, -quality, -size) %>% # drop school-level values
  left_join(quality_cutoff, by = "choice_rev") # add choice-level values
```

Answer: Here are the first ten entries of the cutoff and quality for each recoded choice and I merged it to the (school, program) datasets by 'choice_rev'

choice_rev	cutoff	quality
100 arts	185	268.60
100 economics	188	261.71
100 others	185	254.67
100 science	198	290.67
101 arts	194	328.94
101 economics	193	324.32
101 others	200	305.44
101 science	202	358.08
102 arts	199	323.40
102 economics	200	321.02

Exercise 5 First Model: Multinomial Logit Model

1. Propose a model specification. Write the Likelihood function

```
#### clean the dataset for exercise 5
# keep only 20,000 students' with the highest score
# keep only first choice
# keep useful variables
dat5 <- dat %>%
  filter(schoolchoice == "schoolcode1") %>%
  arrange(desc(score)) %>%
  filter(score >= score[20000]) %>% # 20445 obs
  select(student_id, choice_rev, quality, score)

# add quality of all other choices to the dataset
onerow_choice <- dat5[,2:3] %>%
  distinct(choice_rev, .keep_all = TRUE) %>%
  spread(choice_rev, quality)
q <- as.array(as.matrix(onerow_choice[1,], nrow=1))
dat_choice <- as.data.frame(matrix(rep(q,nrow(dat5)), byrow = TRUE, nrow = nrow(dat5)))
colnames(dat_choice) <- names(onerow_choice)
dat5 <- cbind(dat5, dat_choice)
```

```
#### write the likelihood function
# construct the variables
ni <- nrow(dat5)
nj <- ncol(dat5[,5:250])
X <- cbind(rep(1,nrow(dat5)), dat5[,4])
Y <- matrix(0, ni, nj)
for(i in 1:nj) {
```

```
  for (j in 1:nj) {
    if(dat5$choice_rev[j] == names(dat5)[i+4]) {
      Y[j,i] = 1
    }
  }
}

# likelihood function
mlike <- function(y, x, beta) {
  beta = mat.or.vec(2,nj)
  sum_exp = as.matrix(rowSums(exp(x %*% beta[,2:nj])))
  mat_pro = mat.or.vec(nrow(dat5), nj)
  mat_pro[, 1] = 1 / (1+sum_exp)
  for (i in 1:(nj-1)) {
    prob = exp(x %*% beta[, i+1]) / (1+sum_exp)
    mat_pro[, i+1] = prob
  }
  like = 0
  for (i in 1:nj) {
    like = like + colSums(as.matrix(y[,i]*log(mat_pro[,i])))
  }
  return(-like)
}
```

2. Estimate parameters and compute the marginal effect of the proposed model

```
# estimate the parameter
res1 <- optim(function(beta) mlike(Y, X, beta), par=runif(490), method="BFGS")
result_multi <- as.matrix(res1$par)
```

```
# compute the marginal effect
fnprob <- function(x, beta) {
  sum_exp = as.matrix(rowSums(exp(x %*% beta[,2:nj])))
  mat_pro = mat.or.vec(nrow(dat5), nj)
  mat_pro[, 1] = 1 / (1+sum_exp)
  for (i in 1:(nj-1)) {
    prob = exp(x %*% beta[, i+1]) / (1+sum_exp)
    mat_pro[, i+1] = prob
  }
  return(mat_pro)
}
b1 <- mat.or.vec(2, nj)
b1[1, 2:246] <- result_multi[1:245]
b1[2, 2:246] <- result_multi[246:490]
pij_m1 <- fnprob(X, b1)
mb <- c(0, b1[2, 2:246])
me_mlogit <- array(0, dim = c(nrow(X), 246))
for (i in 1:nrow(X)) {
  be <- sum(pij_m1[i,]*mb)
  me_mlogit[i,] <- pij_m1[i,]*(mb-be)
}
me_mlogit <- apply(me_mlogit, 2, mean)
```

Answer: Here are the first ten parameters I estimated (too lang to show all)

parameters of multinomial logit model
0.7991836
0.5751852
0.6348069
0.4980496
0.1381639
0.9969494
0.2988119
0.8755065
0.9988975
0.1841155

Exercise 6 Second Model: Conditional Logit Model

1. Propose a model specification. Write the Likelihood function

```
# construct the variables
dat6 <- dat5
choice_quality <- dat6 %>%
  group_by(choice_rev) %>%
  summarise(quality = unique(quality))
X2 <- as.matrix(cbind(rep(1, nj), choice_quality[,2]))

# likelihood function
clike <- function(y, x, beta) {
  beta = mat.or.vec(1,nj+1)
  beta[1] = 0
  mat_pro = mat.or.vec(1, nj)
  mat_beta = mat.or.vec(2, nj)
  mat_beta[1,] = beta[1:nj]
  mat_beta[2,] = beta[nj+1]
  sum_exp = sum(diag(exp(x %*% mat_beta)))
  mat_pro = diag(exp(x %*% mat_beta)) / sum_exp
  like = 0
  for (i in 1:nj) {
    like = like + colSums(as.matrix(y[,i]*log(mat_pro[i])))
  }
  return(-like)
}
```

2. Estimate parameters and compute the marginal effect of the proposed model

```
# estimate the parameter
res2 <- optim(function(beta) clike(Y, X2, beta), par=runif(246), method="BFGS")
result_con <- as.matrix(res2$par)
```

```r
# compute the marginal effect
fnprob2 <- function(x, beta) {
  beta[1] = 0
  mat_pro = mat.or.vec(1, nj)
  mat_beta = mat.or.vec(2, nj)
  mat_beta[1,] = beta[1:nj]
  mat_beta[2,] = beta[nj+1]
  sum_exp = sum(diag(exp(x %*% mat_beta)))
  mat_pro = diag(exp(x %*% mat_beta)) / sum_exp
  return(mat_pro)
}
pj <- fnprob2(X2, res2$par)
mid <- array(0, dim = c(nrow(X2), nj))
for (i in 1:nrow(X2)) {
  mid[i,i] <- 1
}
me_clogit <- array(0, dim = c(nrow(X2), nj))
for (j in 1:246) {
  for (k in 1:246) {
    me_clogit[j,k] <- pj[j] * (mid[j,k]-pj[k]) * res2$par[247]
  }
}
```

Answer: Here are the first ten parameters I estimated (too lang to show all)

parameters of conditional logit model
0.3668233
0.9024672
0.0558784
0.5534810
0.1418612
0.3731371
0.0607639
0.7417934
0.8786197
0.3104080

Exercise 7 Counterfactual Simulations

1. Explain and justify which model you think is appropriate to conduct this exercise

In the conditional logit model, quality changes when the choices changes; however, in the multinomial logit model, quality changes only when the observed student changes. Since we are interested in the effect of excluding choices where the program is "others", **conditional logit model** is more appropriate here.

2. Calculate choice probabilities under the appropriate model

```r
# drop obs whose choices contain 'others'
dat7 <- dat5 %>%
```

```
  filter(str_detect(choice_rev, 'others') == FALSE)

# calculate the choice probabilities (?)
```

3. Simulate how these choice probabilities change when these choices are excluded

```
# construct variables (same as exercise 6)
ni <- nrow(dat7)
nj <- ncol(dat7[,5:200])
Y2 <- as.matrix(0, ni, nj)
for (i in 1:nj) {
  for (j in 1:ni) {
    if(dat7$choice_rev[j] == names(dat7)[i+4]) {
      Y2[j,i] <- 1
    }
  }
}
choice_quality2 <- dat7 %>%
  group_by(choice_rev) %>%
  summarise(quality = unique(quality))
X3 <- as.matrix(cbind(rep(1, nj), choice_quality2[,2]))

# likelihood function
clike2 <- function(y, x, beta) {
  beta = mat.or.vec(1, nj+1)
  beta[1] = 0
  mat_pro = mat.or.vec(1, nj)
  mat_beta = mat.or.vec(2, nj)
  mat_beta[1,] = beta[1:nj]
  mat_beta[2,] = beta[nj+1]
  sum_exp = sum(diag(exp(x %*% mat_beta)))
  mat_pro = diag(exp(x %*% mat_beta)) / sum_exp
  like = 0
  for (i in 1:nj) {
    like = like + colSums(as.matrix(y[,i]*log(mat_pro[i])))
  }
  return(-like)
}

# find the parameters
res3 <- optim(function(beta) clike2(Y2, X3, beta), par=runif(196), method="BFGS")
result_con <- as.matrix(res3$par)
```