Assignment 1

Yongyi Lin

2/28/2021

```
setwd("/Users/yongyilin/Econ613/Assignments/A1/dat")
stu = read.csv("datstu.csv", stringsAsFactors = FALSE)
jss = read.csv("datjss.csv", stringsAsFactors = FALSE)
sss = read.csv("datsss.csv", stringsAsFactors = FALSE)
```

Part 1

Exercise 1 Missing Data

Number of Students

```
length(unique(stu$X))
```

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

There are 340823 students.

Number of Schools

```
school <- stu %>%
  dplyr::select(starts_with("schoolcode")) %>%
  pivot_longer(
    cols = starts_with("schoolcode"),
    names_to = "rank",
    names_prefix = "schoolcode",
    values_to = "school",
    values_drop_na = TRUE
  ) %>%
  dplyr::select(-c(1)) %>%
  mutate_all(na_if,"")
sum(is.na(school)) # No missing strings.
```

```
## [1] 0
```

```
length(unique(school$school))
```

```
## [1] 640
```

There are 640 schools.

Number of Programs

```
program <- stu %>%
  dplyr::select(starts_with("choicepgm")) %>%
  pivot_longer(
    cols = starts_with("choicepgm"),
    names_to = "rank",
    names_prefix = "choicepgm",
    values_to = "program",
    values_drop_na = TRUE
  ) %>%
  dplyr::select(-c(1)) %>%
  mutate_all(na_if, "")
sum(is.na(program)) # There are 38454 missing strings.
```

```
## [1] 38454
```

```
program <- program[program != "NA"]
length(unique(program))
```

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

There are 32 programs.

Number of Choices

```
choice <- stu %>%
  dplyr::select(c(1, 5:16))
choice[is.na(choice)] <- ""
choice[choice == ""] <- "missing" # identify missing strings
choice$choice1 <- paste(choice$schoolcode1, "-", choice$choicepgm1)
choice$choice2 <- paste(choice$schoolcode2, "-", choice$choicepgm2)
choice$choice3 <- paste(choice$schoolcode3, "-", choice$choicepgm3)
choice$choice4 <- paste(choice$schoolcode4, "-", choice$choicepgm4)
choice$choice5 <- paste(choice$schoolcode5, "-", choice$choicepgm5)
choice$choice6 <- paste(choice$schoolcode6, "-", choice$choicepgm6)
choice <- dplyr::select(choice, c(X, choice1:choice6)) %>%
  pivot_longer(
    cols = starts_with("choice"),
    names_to = "rank",
    names_prefix = "choice",
    values_to = "choice",
  )
unique_choice <- dplyr::select(choice, "choice")
unique_choice <- unique_choice[!grepl("missing", unique_choice$choice), ] # drop the rows with missing
length(unique(unique_choice$choice))
```

```
## [1] 2773
```

There are 2773 choices.

Missing Test Score

```
summary(is.na(stu$score))
```

```
##    Mode   FALSE    TRUE
## logical  160936  179887
```

There are 179887 missing test scores

Apply to the same school (different programs)

```
for (i in 1:nrow(stu)) {
  stu$samesch[i] = length(
    unique(na.omit(unlist(stu[i, 5:10])))
  )
```

```r
}
sum(stu$samesch < 6-rowSums(is.na(stu[5:10])))
```

```
## [1] 120071
```

There are 120071 students applying to the same school.

Apply to less than 6 choices

```r
less_choice <- choice %>%
  separate(choice, c("schoolcode", "program"), sep = " - ") %>%
  mutate(empty_choice = case_when(
    schoolcode == "missing" ~ 1,
    program == "missing" ~ 1,
    TRUE ~ 0
  )) %>%
  filter(empty_choice == 1) %>%
  distinct(X)
nrow(less_choice)
```

```
## [1] 21001
```

There are 21001 students who applied to less than 6 choices.

Exercise 2 Data

```r
# Create a school-program level dataset
sch_prog <- stu %>%
  dplyr::select(-c(3:4))
sch_prog$choice1 <- paste(sch_prog$schoolcode1, "-", sch_prog$choicepgm1)
sch_prog$choice2 <- paste(sch_prog$schoolcode2, "-", sch_prog$choicepgm2)
sch_prog$choice3 <- paste(sch_prog$schoolcode3, "-", sch_prog$choicepgm3)
sch_prog$choice4 <- paste(sch_prog$schoolcode4, "-", sch_prog$choicepgm4)
sch_prog$choice5 <- paste(sch_prog$schoolcode5, "-", sch_prog$choicepgm5)
sch_prog$choice6 <- paste(sch_prog$schoolcode6, "-", sch_prog$choicepgm6)
sch_prog2 <- sch_prog %>%
  dplyr::select(c("X", "score", "jssdistrict":"choice6")) %>%
  pivot_longer(
    cols = starts_with("choice"),
    names_to = "rank",
    names_prefix = "choice",
    values_to = "choice",
  ) %>%
  separate(choice, c("schoolcode", "program"), sep = " - ") %>%
  filter(rankplace == rank) %>%
  drop_na()

# left join "sss"
sss2 <- sss %>%
  dplyr::select(-c(1:2)) %>%
  distinct() %>%
  drop_na()
sch_prog3 <- merge(x = sch_prog2, y = sss2, by = "schoolcode", all.x = TRUE)

# statistics
sch_prog_summary <- sch_prog3 %>%
  dplyr::select(c(1, 3:10)) %>%
  group_by(schoolcode, program) %>%
  dplyr::summarise(
    cutoff = min(score),
    quality = mean(score),
    size = n(), .groups = "drop"
  )
head(sch_prog_summary)
```

```
## # A tibble: 6 x 5
##   schoolcode program        cutoff quality  size
##   <chr>      <chr>           <int>   <dbl> <int>
## 1 100101     General Arts      198    244.    79
## 2 100101     Home Economics    199    229.    40
## 3 100101     Technical         201    235.    49
## 4 100102     Agriculture       273    293.    90
## 5 100102     Business          283    303.    90
## 6 100102     General Arts      291    311.    90
```

Exercise 3 Distance

```
# define a distance function
distance <- function(ssslong, jsslong, jsslat, ssslat) {
  dist <- sqrt((69.172*(ssslong-jsslong)*cos(jsslat/57.3))^2 + (69.172*(ssslat-jsslat))^2)
  return(dist)
}
jss2 <- jss %>%
  dplyr::select(-c("X"))

# left join jss
sch_prog_dist <- merge(x = sch_prog3, y = jss2, by = "jssdistrict", all.x = TRUE)

# calculate the distance
sch_prog_dist <- mutate(sch_prog_dist, distance = distance(ssslong, ssslat, point_x, point_y))

head(sch_prog_dist)
```

```
##                              jssdistrict schoolcode      X score rankplace
## 1 Abura/Asebu/Kwamankese (Abura Dunkwa)      31201 208121   301         1
## 2 Abura/Asebu/Kwamankese (Abura Dunkwa)      30201 231045   280         3
## 3 Abura/Asebu/Kwamankese (Abura Dunkwa)      31201 207954   337         2
## 4 Abura/Asebu/Kwamankese (Abura Dunkwa)    9040101 323945   227         2
## 5 Abura/Asebu/Kwamankese (Abura Dunkwa)      30402 181969   267         2
## 6 Abura/Asebu/Kwamankese (Abura Dunkwa)      30201 231151   254         4
##   samesch rank                   program                          sssdistrict
## 1       5    1               Visual Arts             Assin North (Assin Fosu)
## 2       6    3           General Science                 Edina/Komenda/Eguafo
## 3       3    2           General Science             Assin North (Assin Fosu)
## 4       6    2 Block Laying & Concreting Shama/Ahanta/East (Sekondi/Takoradi)
## 5       2    2                  Business Abura/Asebu/Kwamankese (Abura Dunkwa)
## 6       6    4               Visual Arts                 Edina/Komenda/Eguafo
##     ssslong   ssslat   point_x  point_y distance
## 1 -1.374617 5.777995 -1.197088 5.130001 660.4734
## 2 -1.437420 5.140793 -1.197088 5.130001 631.2719
## 3 -1.374617 5.777995 -1.197088 5.130001 660.4734
## 4 -1.623655 5.081101 -1.197088 5.130001 637.6075
## 5 -1.197088 5.130001 -1.197088 5.130001 618.8735
## 6 -1.437420 5.140793 -1.197088 5.130001 631.2719
```

Exercise 4 Descriptive Characteristics

mean and sd for ranked choice

```
rankedchoice0 <- sch_prog_dist %>%
  drop_na() %>%
  group_by(schoolcode, program, rankplace, rank) %>%
  dplyr::summarise(
    cutoff = min(score),
    quality = mean(score),
    dist = mean(distance)
  )
```

```
## `summarise()` regrouping output by 'schoolcode', 'program', 'rankplace' (override with `.groups` argu
```

```
rankedchoice <- rankedchoice0 %>%
  group_by(rankplace, rank) %>%
  dplyr::summarise(
    mean_cutoff = mean(cutoff),
    sd_cutoff = sd(cutoff),
    mean_quality = mean(quality),
    sd_quality = sd(quality),
    mean_dist = mean(dist),
    sd_dist = sd(dist)
  )
```

```
## `summarise()` regrouping output by 'rankplace' (override with `.groups` argument)
```

```
rankedchoice
```

```
## # A tibble: 6 x 8
## # Groups:   rankplace [6]
##   rankplace rank  mean_cutoff sd_cutoff mean_quality sd_quality mean_dist
##       <int> <chr>       <dbl>     <dbl>        <dbl>      <dbl>     <dbl>
## 1         1 1            264.      47.4         289.       44.5      747.
## 2         2 2            264.      47.5         284.       45.8      749.
## 3         3 3            262.      46.0         281.       43.1      751.
## 4         4 4            257.      43.7         276.       39.8      752.
## 5         5 5            230.      21.8         247.       22.3      778.
## 6         6 6            231.      23.5         249.       22.4      785.
## # ... with 1 more variable: sd_dist <dbl>
```

differentiate by quantile

```
quantile <- sch_prog_dist %>%
  drop_na()
quantile$quantile <- ntile(quantile$score, 4)
quantile_summary0 <- quantile %>%
  group_by(schoolcode, program, rankplace, rank, quantile) %>%
  dplyr::summarise(
    cutoff = min(score),
    quality = mean(score),
    dist = mean(distance)
  )
```

```
## `summarise()` regrouping output by 'schoolcode', 'program', 'rankplace', 'rank' (override with `.grou
```

```
quantile_summary <- quantile_summary0 %>%
  drop_na() %>%
  group_by(rankplace, rank, quantile) %>%
  dplyr::summarise(
    mean_cutoff = mean(cutoff),
    sd_cutoff = sd(cutoff),
    mean_quality = mean(quality),
    sd_quality = sd(quality),
    mean_dist = mean(dist),
    sd_dist = sd(dist)
  )
```

```
## `summarise()` regrouping output by 'rankplace', 'rank' (override with `.groups` argument)
```

```
quantile_summary
```

```
## # A tibble: 24 x 9
## # Groups:   rankplace, rank [6]
##    rankplace rank  quantile mean_cutoff sd_cutoff mean_quality sd_quality
##        <int> <chr>    <int>       <dbl>     <dbl>        <dbl>      <dbl>
##  1         1 1            1        229.     14.4          238.       9.79
##  2         1 1            2        265.      9.18         272.       7.17
##  3         1 1            3        299.     10.4          307.       8.67
##  4         1 1            4        344.     17.0          355.      19.1
##  5         2 2            1        228.     14.1          237.      10.3
##  6         2 2            2        266.      9.28         272.       7.78
##  7         2 2            3        299.     10.5          307.       9.30
##  8         2 2            4        344.     17.3          353.      18.9
##  9         3 3            1        228.     14.5          238.      10.2
## 10         3 3            2        265.      9.16         272.       7.38
## # ... with 14 more rows, and 2 more variables: mean_dist <dbl>, sd_dist <dbl>
```

Part 2

Exercise 5 Data Creation

```
set.seed(956336)
x1 <- runif(10000, 1, 3)
x2 <- rgamma(10000, shape = 3, scale = 2)
x3 <- rbernoulli(10000, p = 0.3)*1
epsilon <- rnorm(10000, mean = 2, sd = 1)
X <- cbind(x1, x2, x3)
y <- 0.5+1.2*x1-0.9*x2+0.1*x3+epsilon
y_bar <- mean(y)
ydum <- ifelse(y>y_bar, 1, 0)
dataset <- cbind(X, ydum)
dataset <- as.data.frame(dataset)
head(dataset)
```

```
##         x1       x2 x3 ydum
## 1 1.861351 3.980222  1    1
## 2 2.723714 3.176602  1    1
## 3 1.155744 3.088031  0    1
## 4 1.376748 6.270333  0    0
## 5 1.185392 4.813045  0    1
## 6 2.643856 5.074495  1    1
```

Exercise 6 OLS

```
cor(y, x1) # 0.2013141, which is significantly different from 1.2
```

```
## [1] 0.2013141
```

```
cons <- rep(1, 10000)
X <- cbind(cons,x1,x2,x3)
beta <- solve(t(X)%*%X)%*%t(X)%*%y
rownames(beta)[1] <- 'intercept'
colnames(beta)[1] <- 'est_beta'
beta
```

```
##              est_beta
## intercept  2.51624149
## x1         1.20342505
## x2        -0.90186227
## x3         0.07748936
```

```
sigma2 <- sum((y-X%*%beta)^2)/(nrow(X)-ncol(X))
var <- sigma2*solve(t(X)%*%X)
se_ols <- sqrt(diag(var))
se_ols
```

```
##        cons          x1          x2          x3
## 0.040143028 0.017248232 0.002878548 0.021660322
```

Exercise 7 Discrete Choice

```
set.seed(100)
start <- runif(4, 0.1, 0.5)
```

```r
# sum of log-likelihood function
ll <- function(f, y, x, beta) {
  pr <- f(x, beta)
  pr[pr>0.999999] = 0.999999
  pr[pr<0.000001] = 0.000001
  likelihood <- (pr^y)*((1-pr)^(1-y))
  return(sum(log(likelihood)))
}

# optimize the probit model
probit <- function(x, beta) {
  xbeta <- x %*% beta
  return(pnorm(xbeta))
}
pb_ll <- function(beta) {
  return(-ll(f = probit, y = ydum, x = X, beta))
}
pb <- optim(par = start, fn = pb_ll, method = "BFGS", hessian = T)

# optimize the logit model
logit <- function(x, beta) {
  xbeta <- x %*% beta
  return(exp(xbeta)/(1+exp(xbeta)))
}
lg_ll <- function(beta) {
  return(-ll(f = logit, y = ydum, x = X, beta))
}
lg <- optim(par = start, fn = lg_ll, method = "BFGS", hessian = T)

# optimize the linear model
linear <- function(x, beta) {
  return(sum((x %*% beta - ydum)^2))
}
ln <- optim(par = start, fn = linear, x = X, method = "BFGS", hessian = T)

# estimation outcome
estimation_compare <- cbind(pb$par, lg$par, ln$par)
colnames(estimation_compare) <- c("coef_probit", "coef_logit", "coef_linear")
rownames(estimation_compare) <- c("intercept", "x1", "x2", "x3")

estimation_compare
```

```
##           coef_probit coef_logit coef_linear
## intercept   3.0811622  5.5321135   0.8935953
## x1          1.1823903  2.1283942   0.1408550
## x2         -0.9177430 -1.6495055  -0.1039362
## x3          0.1549438  0.2925268   0.0236880
```

Exercise 8 Marginal Effects

```r
mfxboot <- function(modform,dist,data,boot=1000,digits=3){
  x <- glm(modform, family=binomial(link=dist),data)
  # get marginal effects
  pdf <- ifelse(dist=="probit",
                mean(dnorm(predict(x, type = "link"))),
                mean(dlogis(predict(x, type = "link"))))
  marginal.effects <- pdf*coef(x)
  # start bootstrap
  bootvals <- matrix(rep(NA,boot*length(coef(x))), nrow=boot)
  set.seed(1704)
  for(i in 1:boot){
    samp1 <- data[sample(1:dim(data)[1],replace=T,dim(data)[1]),]
    x1 <- glm(modform, family=binomial(link=dist),samp1)
    pdf1 <- ifelse(dist=="probit",
                   mean(dnorm(predict(x, type = "link"))),
                   mean(dlogis(predict(x, type = "link"))))
    bootvals[i,] <- pdf1*coef(x1)
  }
  res <- cbind(marginal.effects,apply(bootvals,2,sd),marginal.effects/apply(bootvals,2,sd))
  if(names(x$coefficients[1])=="(Intercept)"){
    res1 <- res[2:nrow(res),]
    res2 <- matrix(as.numeric(sprintf(paste("%.",paste(digits,"f",sep=""),sep=""),res1)),nrow=dim(res1)
    rownames(res2) <- rownames(res1)
  } else {
    res2 <- matrix(as.numeric(sprintf(paste("%.",paste(digits,"f",sep=""),sep=""))),nrow=dim(res)[1]))
    rownames(res2) <- rownames(res)
  }
  colnames(res2) <- c("marginal.effect","standard.error","z.ratio")
  return(res2)
}

mfx_probit <- mfxboot(ydum ~ x1 + x2 + x3, "probit", dataset)
mfx_logit <- mfxboot(ydum ~ x1 + x2 + x3, "logit", dataset)

mfx_probit
```

```
##    marginal.effect standard.error z.ratio
## x1           0.141          0.005  26.603
## x2          -0.110          0.002 -49.376
## x3           0.018          0.006   3.286
```

```
mfx_logit
```

```
##    marginal.effect standard.error z.ratio
## x1           0.141          0.005  26.310
## x2          -0.109          0.002 -47.899
## x3           0.019          0.006   3.421
```