# A1_Xinwei

Xinwei Liu

3/1/2021

## Part 1

```
datjss <- read.csv("C:/Users/DELL/Desktop/lxw/datjss.csv")
datsss <- read.csv("C:/Users/DELL/Desktop/lxw/datsss.csv")
datstu <- read.csv("C:/Users/DELL/Desktop/lxw/datstu.csv")
```

### Exercise 1

```
# Number of students
length(datstu$X)
```

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

```
# Number of schools
numberofschools<-unique(datsss$schoolcode)
numberofschools<- data.frame(numberofschools)
nrow(numberofschools)
```

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

```
# Number of programs
length(unique(unlist(datstu[11:16])))
```

```
## [1] 33
```

```
# Number of choices (school,program)
a<-data.frame(datstu$schoolcode1,datstu$choicepgm1)
b<-data.frame(datstu$schoolcode2,datstu$choicepgm2)
c<-data.frame(datstu$schoolcode3,datstu$choicepgm3)
d<-data.frame(datstu$schoolcode4,datstu$choicepgm4)
e<-data.frame(datstu$schoolcode5,datstu$choicepgm5)
f<-data.frame(datstu$schoolcode6,datstu$choicepgm6)

colnames(a)[colnames(a) == "datstu.schoolcode1"] <- "schoolcode"
colnames(a)[colnames(a) == "datstu.choicepgm1"] <- "choicepgm"
colnames(b)[colnames(b) == "datstu.schoolcode2"] <- "schoolcode"
```

```r
colnames(b)[colnames(b) == "datstu.choicepgm2"] <- "choicepgm"
colnames(c)[colnames(c) == "datstu.schoolcode3"] <- "schoolcode"
colnames(c)[colnames(c) == "datstu.choicepgm3"] <- "choicepgm"
colnames(d)[colnames(d) == "datstu.schoolcode4"] <- "schoolcode"
colnames(d)[colnames(d) == "datstu.choicepgm4"] <- "choicepgm"
colnames(e)[colnames(e) == "datstu.schoolcode5"] <- "schoolcode"
colnames(e)[colnames(e) == "datstu.choicepgm5"] <- "choicepgm"
colnames(f)[colnames(f) == "datstu.schoolcode6"] <- "schoolcode"
colnames(f)[colnames(f) == "datstu.choicepgm6"] <- "choicepgm"

datachoice<-bind_rows(a,b,c,d,e,f)
numberofchoices <- unique(datachoice)
datachoice<-datachoice[!(is.na(numberofchoices$schoolcode)),]
nrow(numberofchoices)
```

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

```r
# Missing Test score
sum(is.na(datstu$score))
```

```
## [1] 179887
```

```r
# Apply to the same school (different programs)
frame <- data.frame(datstu)
frame$n = abs(frame$schoolcode1 - frame$schoolcode2) + abs(frame$schoolcode2 -
          frame$schoolcode3) + abs(frame$schoolcode3 - frame$schoolcode4) +
          abs(frame$schoolcode4 - frame$schoolcode5) + abs(frame$schoolcode5 - frame$schoolcode6)
length(which(frame$n==0))
```

```
## [1] 174
```

```r
## Apply to less than 6 choices
complete6choices <- sum(complete.cases(frame[,5:10]))
numberofstudents <- complete6choices
numberofstudents
```

```
## [1] 323089
```

### Exercise 2

```r
datsss2<-datsss
datsss2<-datsss2[!(is.na(datsss2$ssslong)),]
datsss2<-datsss2[,-1]
datsss2<-as.data.frame(unique(datsss2))

data2<-merge(x=numberofchoices,y=datsss2,by="schoolcode",all.x = TRUE, all.y = FALSE)

datstu_clear<-datstu
datstu_clear<-datstu_clear[!(is.na(datstu_clear$rankplace)),] #delete observations with missing value i
```

```r
for (i in 1:nrow(datstu_clear)){
  if (datstu_clear$rankplace[i]==1){
    datstu_clear$schoolcode[i]<-datstu_clear$schoolcode1[i]
    datstu_clear$choicepgm[i]<-datstu_clear$choicepgm1[i]
  }
  if (datstu_clear$rankplace[i]==2){
    datstu_clear$schoolcode[i]<-datstu_clear$schoolcode2[i]
    datstu_clear$choicepgm[i]<-datstu_clear$choicepgm2[i]
  }
  if (datstu_clear$rankplace[i]==3){
    datstu_clear$schoolcode[i]<-datstu_clear$schoolcode3[i]
    datstu_clear$choicepgm[i]<-datstu_clear$choicepgm3[i]
  }
  if (datstu_clear$rankplace[i]==4){
    datstu_clear$schoolcode[i]<-datstu_clear$schoolcode4[i]
    datstu_clear$choicepgm[i]<-datstu_clear$choicepgm4[i]
  }
  if (datstu_clear$rankplace[i]==5){
    datstu_clear$schoolcode[i]<-datstu_clear$schoolcode5[i]
    datstu_clear$choicepgm[i]<-datstu_clear$choicepgm5[i]
  }
  if (datstu_clear$rankplace[i]==6){
    datstu_clear$schoolcode[i]<-datstu_clear$schoolcode6[i]
    datstu_clear$choicepgm[i]<-datstu_clear$choicepgm6[i]
  }
}

datstu_clear<-datstu_clear[!(datstu_clear$rankplace == 99),]

data2_final<-datstu_clear %>%
  group_by(schoolcode,choicepgm) %>%
  summarise(cutoff=min(score),quality = mean(score),size = n())
```

```
## ‘summarise()‘ regrouping output by ’schoolcode’ (override with ‘.groups‘ argument)
```

```r
data2_final<-merge(x=data2,y=data2_final,by= c("schoolcode", "choicepgm"))
data2_final[1:20,]
```

```
##    schoolcode       choicepgm                                   schoolname
## 1      100101    General Arts          WA SENIOR HIGH/TECHNICAL SCHOOL, WA
## 2      100101  Home Economics          WA SENIOR HIGH/TECHNICAL SCHOOL, WA
## 3      100101       Technical          WA SENIOR HIGH/TECHNICAL SCHOOL, WA
## 4      100102     Agriculture                    WA SENIOR HIGH SCHOOL, WA
## 5      100102        Business                    WA SENIOR HIGH SCHOOL, WA
## 6      100102    General Arts                    WA SENIOR HIGH SCHOOL, WA
## 7      100102 General Science                    WA SENIOR HIGH SCHOOL, WA
## 8      100102  Home Economics                    WA SENIOR HIGH SCHOOL, WA
## 9      100102     Visual Arts                    WA SENIOR HIGH SCHOOL, WA
## 10     100104    General Arts LASSIE-TUOLO SNR SENIOR HIGH. SCHOOL, LASSIE
## 11     100104 General Science LASSIE-TUOLO SNR SENIOR HIGH. SCHOOL, LASSIE
## 12     100104  Home Economics LASSIE-TUOLO SNR SENIOR HIGH. SCHOOL, LASSIE
## 13     100105        Business              ISLAMIC SENIOR HIGH. SCHOOL, WA
## 14     100105    General Arts              ISLAMIC SENIOR HIGH. SCHOOL, WA
## 15     100105  Home Economics              ISLAMIC SENIOR HIGH. SCHOOL, WA
## 16     100106     Agriculture      T. I. AHMADIYYA SENIOR HIGH. SCHOOL, WA
## 17     100106        Business      T. I. AHMADIYYA SENIOR HIGH. SCHOOL, WA
## 18     100106    General Arts      T. I. AHMADIYYA SENIOR HIGH. SCHOOL, WA
## 19     100201        Business            NANDOM SENIOR HIGH SCHOOL, NANDOM
## 20     100201    General Arts            NANDOM SENIOR HIGH SCHOOL, NANDOM
##      sssdistrict   ssslong   ssslat cutoff  quality size
## 1  Wa Municipal -2.285030 10.03062    198 244.3924   79
## 2  Wa Municipal -2.285030 10.03062    199 229.4500   40
## 3  Wa Municipal -2.285030 10.03062    201 235.1020   49
## 4  Wa Municipal -2.285030 10.03062    273 292.5556   90
## 5  Wa Municipal -2.285030 10.03062    283 303.3444   90
## 6  Wa Municipal -2.285030 10.03062    291 311.1111   90
## 7  Wa Municipal -2.285030 10.03062    273 298.4333   90
## 8  Wa Municipal -2.285030 10.03062    262 278.8667   45
## 9  Wa Municipal -2.285030 10.03062    250 275.2000   45
## 10 Wa Municipal -2.285030 10.03062    319 337.4444   45
## 11 Wa Municipal -2.285030 10.03062    313 334.0000   45
## 12 Wa Municipal -2.285030 10.03062    282 309.3556   45
## 13 Wa Municipal -2.285030 10.03062    251 268.0125   80
## 14 Wa Municipal -2.285030 10.03062    258 274.7375   80
## 15 Wa Municipal -2.285030 10.03062    242 258.1625   80
## 16 Wa Municipal -2.285030 10.03062    223 240.6250   40
## 17 Wa Municipal -2.285030 10.03062    238 253.5000   40
## 18 Wa Municipal -2.285030 10.03062    248 268.9750   40
## 19        Lawra -2.800941 10.54640    288 314.2750   80
## 20        Lawra -2.800941 10.54640    319 339.0250   40
```

**Exercise 3**

```r
data3<-merge(x=datstu_clear,y=data2_final,by= c("schoolcode", "choicepgm"))
data3<-merge(x=data3,y=datjss,by="jssdistrict",all.x = TRUE, all.y = FALSE)

colnames(data3)[colnames(data3) == "point_x"] <- "jsslong"
colnames(data3)[colnames(data3) == "point_y"] <- "jsslat"

data3$distance<-0

for (i in 1:nrow(data3)){
  data3$distance[i]<-sqrt((69.172 * (data3$ssslong[i]-data3$jsslong[i])*cos(data3$jsslat[i]/57.3))^2 +
}
data3$distance[1:20]
```

```
##  [1] 16.574446  0.000000  0.000000  0.000000  0.000000 14.034819  0.000000
##  [8]  0.000000  0.000000  7.719788 46.461827  0.000000  0.000000  0.000000
## [15]  7.719788 39.536292  0.000000  0.000000  0.000000  0.000000
```

**Exercise 4**

```r
# Group by ranked choice
data3<-data3[!(is.na(data3$score)),]
data3<-data3[!(is.na(data3$distance)),]

data3 %>%
  group_by(rankplace) %>%
  summarise(cutoff=min(score),quality = mean(score),distance=mean(distance))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 6 x 4
##   rankplace cutoff quality distance
##       <int>  <int>   <dbl>    <dbl>
## 1         1    165    312.     35.7
## 2         2    173    301.     34.3
## 3         3    190    288.     28.8
## 4         4    185    276.     23.0
## 5         5    198    253.     32.5
## 6         6    158    251.     32.0
```

```r
# Group by quantile
library(cutr)

data3$quantile <- smart_cut(data3$score, 4, "g", output = "numeric")

data3$quantile <- replace(data3$quantile, data3$quantile==1, "0%-25%")
data3$quantile <- replace(data3$quantile, data3$quantile==2, "25%-50%")
data3$quantile <- replace(data3$quantile, data3$quantile==3, "50%-75%")
data3$quantile <- replace(data3$quantile, data3$quantile==4, "75%-100%")

data3 %>%
  group_by(quantile) %>%
  summarise(cutoff=min(score),quality = mean(score),distance=mean(distance))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 4 x 4
##   quantile cutoff quality distance
##   <chr>     <int>   <dbl>    <dbl>
## 1 0%-25%      158    236.     26.3
## 2 25%-50%     255    271.     29.1
## 3 50%-75%     288    307.     31.7
## 4 75%-100%    329    365.     38.5
```

## Part 2

### Exercise 5

```
obs <- 10000
X1 <- runif(obs, max = 3, min = 1)
X2 <- rgamma(obs,3,scale = 2)
X3 <- rbinom(obs,1,0.3)
eps <- rnorm(obs, mean = 2, sd = 1)
Y <- 0.5 + 1.2*X1 - 0.9*X2 + 0.1*X3 + eps
ydum <- ifelse(Y > mean(Y),1,0)
mydata <- data.frame(cbind(Y,ydum,X1,X2,X3,eps))
mydata[1:20,]
```

```
##              Y ydum       X1        X2 X3          eps
## 1  -1.03725628    0 2.464190  7.150324  1  1.841006909
## 2   0.04193952    1 1.343390  4.924866  0  2.362250640
## 3   1.54868597    1 1.134259  2.078834  0  1.558526244
## 4  -3.53589999    0 2.553223  7.049873  0 -0.754882006
## 5   3.68698095    1 2.246745  2.224293  0  2.492750762
## 6  -4.49406098    0 2.376988 10.210120  0  1.342662089
## 7  -1.74986746    0 2.398528  6.060867  1  0.226679282
## 8  -1.32948205    0 2.751889  7.690726  0  1.789903894
## 9   0.78146127    1 1.145160  1.223055  0  0.008019416
## 10 -2.87941681    0 1.283462  8.537855  1  2.664498310
## 11 -2.26877678    0 2.415499  9.477503  1  2.762377038
## 12 -2.02347547    0 1.338043  6.246592  0  1.492805808
## 13 -0.42696157    0 2.003420  5.508591  1  1.526666338
## 14 -5.27442122    0 2.079506 10.834579  0  1.481292396
## 15 -1.79453932    0 2.081677  7.972937  0  2.383091180
## 16  0.72141075    1 1.905727  5.790358  0  3.145860830
## 17  0.51542248    1 1.842389  4.333362  1  1.604581493
## 18 -0.69165599    0 1.259008  4.439332  0  1.292933091
## 19 -2.10129647    0 1.382544  6.757983  1  1.721836238
## 20  0.21694813    1 2.903178  5.062066  1  0.688992991
```

## Exercise 6

```
# Correlation between x1 and y, which is about 0.20 and is very different from 1.2.
cor(Y,X1)
```

```
## [1] 0.1928094
```

```
# Regression of Y on X
cons <- matrix(1,10000,1)
X <- matrix(c(X1,X2,X3),10000,3)
X <- cbind(cons,X)

r1<- solve(t(X) %*% X) %*% t(X)%*% Y
r1
```

```
##             [,1]
## [1,]  2.54366291
## [2,]  1.17719261
## [3,] -0.89885910
## [4,]  0.08465306
```

```
# Calculation of standard error
se <- Y - r1[1]-r1[2]*X1 - r1[3]*X2 - r1[4]*X3
se[1:20]
```

```
##  [1] -0.1392651  0.3436082 -0.4616389 -2.7483558  0.4977925 -0.6584367
##  [7] -1.7538474 -0.1997700 -2.0109208  0.6557139  0.7783394 -0.5274666
## [13] -0.4622416 -0.5273036  0.3778096  1.1390563 -0.3866583 -0.7270800
## [19] -0.2826577 -1.2788844
```

### Exercise 7

```
# Optimizing Probit
X <- cbind(1,X1,X2,X3)
y <- as.matrix(Y)
probit.llike <- function(b., y. = ydum, X. = X){
  phi <- pnorm(X.%*%b.)
  phi[phi==1] <- 0.9999 # avoid NaN of log function
  phi[phi==0] <- 0.0001
  f <- sum(y.*log(phi))+sum((1-y.)*log(1-phi))
  f <- -f
  return(f)
}

result.p <- optim(par = c(0,0,0,0), probit.llike)
result.p$par
```

```
## [1]  2.98630678  1.21732085 -0.91596882  0.02728636
```

```
# Optimizing Logit
logit.llike <- function(b., y. = ydum,X. = X){
  gamma <- plogis(X%*%b.)
  f <- sum(y.*log(gamma))+sum((1-y.)*log(1-gamma))
  f <- -f
  return(f)
}
result.l <- optim(par = c(0,0,0,0), logit.llike)
result.l$par
```

```
## [1]  5.37356520  2.17110971 -1.64247065  0.05705816
```

```
# Optimizing Linear: same with OLS

result.lp <- lm(ydum~X1+X2+X3)
summary(result.lp)
```

```
##
## Call:
```

```
## lm(formula = ydum ~ X1 + X2 + X3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.92143 -0.26961  0.06018  0.25194  1.71871
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  0.8783269  0.0133501   65.792   <2e-16 ***
## X1           0.1494790  0.0057293   26.090   <2e-16 ***
## X2          -0.1039301  0.0009666 -107.519   <2e-16 ***
## X3           0.0065142  0.0072436    0.899    0.369
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3332 on 9996 degrees of freedom
## Multiple R-squared:  0.549,  Adjusted R-squared:  0.5489
## F-statistic:  4056 on 3 and 9996 DF,  p-value: < 2.2e-16
```

### Exercise 8

```
# Compute Marginal Effect of X of probit

probit.ME <- function(df){
  result <- glm(ydum ~ X1 + X2 + X3, family=binomial(link = "probit"),df)
  ME <- mean(dnorm(X%*%coef(result)))*coef(result)
  return(ME)
}
probit.ME(mydata)
```

```
##  (Intercept)           X1           X2           X3
##  0.359008865  0.146424541 -0.110147744  0.003311964
```

```
# Compute Marginal Effect of X of Logit
logit.ME <- function(df){
  result <- glm(ydum ~ X1 + X2 + X3, family=binomial(link = "logit"),df)
  ME <- mean(dlogis(X%*%coef(result)))*coef(result)
  return(ME)
}
logit.ME(mydata)
```

```
## (Intercept)          X1          X2          X3
##  0.36022085  0.14552408 -0.11009777  0.00383085
```

```
# Compute the Standard Error
jacobian <- function(fun,par){
  d <- 1e-8
  par. <- matrix(par,length(par),length(par))
  J <- (apply(par. + diag(d,length(par)),2,fun)-apply(par.,2,fun))/d
  return(J)
}
```

```r
# Compute the Standard Error of ME (Probit)
result.p.glm <- glm(ydum ~ X1 + X2 + X3, family = binomial(link = "probit"), data = mydata)

J <- jacobian(function(result) mean(dnorm(X%*%result))*result, coef(result.p.glm)) # "jacobian" defined
cov_matrixv <- vcov(result.p.glm)
se.p <- sqrt(diag(J%*%cov_matrixv%*%t(J)))
se.p
```

```
## [1] 0.0094530062 0.0043103344 0.0004301401 0.0056443212
```

```r
# Compute the Standard Error of ME (Logit)
result.l.glm <- glm(ydum ~ X1 + X2 + X3, family = binomial(link = "logit"), data = mydata)

J <- jacobian(function(result) mean(dlogis(X%*%result))*result, coef(result.l.glm))
cov_matrixv <- vcov(result.l.glm)
se.l <- sqrt(diag(J%*%cov_matrixv%*%t(J)))
se.l
```

```
## [1] 0.0094551613 0.0043092886 0.0004422671 0.0056511187
```