```r
##### Econ 690 Computational Methods #####

### Assignment 3 ###
# 2/7/2020
# by Bahar Zafer

###=======================================
# Exercise 1 Basic Description
###=======================================

install.packages("fastDummies")
library(fastDummies)
library(ggplot2)
library(AER)
data("Affairs")
dat = Affairs

# Finding correlations between affair and observed attributes in the data

# converting gender and children to numeric
dat1 = dat
dat1 = dummy_cols(dat1, remove_first_dummy = T )
# dummy variables, gender_male and children_yes are created.
dat1 = dat1 %>% remove_cols("gender", "children")

cor(dat1)

# Plotting the histogram of affairs
library(dplyr)
library(reshape2)

# by gender
g = dat %>%
  ggplot( aes(x=affairs, fill=gender)) +
  geom_histogram(color="#e9ecef", alpha=0.6, position = 'dodge') +
  theme_ipsum() +
  labs(fill="")

# by age groups
groups_age = vector()
for(i in 1:nrow(dat)) {
  if(dat$age[i]>40) groups_age[i] = "above 40"
  else   groups_age[i] = "under 40"
}
dat = cbind(dat, groups_age)

a = dat %>%
  ggplot( aes(x=affairs, fill=groups_age)) +
  geom_histogram(color="#e9ecef", alpha=0.6, position = 'dodge') +
  theme_ipsum() +
  labs(fill="")


# by religiousness
groups_relig = vector()
for(i in 1:nrow(dat)) {
  if(dat$religiousness[i]>3)
    groups_relg[i] = "religious"
  else   groups_relig[i] = "not religious"
}
dat = cbind(dat, groups_relig)

r = dat %>%
  ggplot( aes(x=affairs, fill=groups_relig)) +
  geom_histogram(alpha=0.6, position = 'dodge') +
  theme_ipsum() +
  labs(fill="")


```

```r
###=======================================
# Exercise 2 Linear Regression
###=======================================
regression1 = lm(affairs ~ education + gender_male + occupation + age + children_yes +
                   religiousness + yearsmarried + rating ,
                 data = dat1)


regression2 = lm(affairs ~ education*occupation +
                   education + gender_male + occupation + age + children_yes +
                   religiousness + yearsmarried + rating ,
                 data = dat1)


regression3 = lm(affairs ~ education*children_yes +
                   education + gender_male + occupation + age + children_yes +
                   religiousness + yearsmarried + rating ,
                 data = dat1)


###=======================================
# Exercise 3 Probit
###=======================================

install.packages("NORMT3")
library(NORMT3)

# recoding affairs as a dummy variable
dat1$affairs = as.numeric(dat1$affairs>0)

# creating a matrix of dependent and independent variables
xmat = cbind(1, dat1[-dat1[,"affairs"]])
xmat = as.matrix(Xmat)

ymat = as.matrix(dat1$affairs)

# Estimating the model using glm()
probitaffairs_glm = glm(formula = affairs ~ education + gender_male +
                          occupation + age + children_yes +
                          religiousness + yearsmarried + rating ,
                        family = binomial(link = "probit"),
                        data = dat1)

# extracting coefficients
betas = as.matrix(coef(probitaffairs_glm))

# likelihood function

affairs_probit = function(b, x, y){
                 L = vector()
                 for (i in 1:nrow(y)){
                   L[i] = erf(x[i,]%*%b)^y[i,]*
                     (1-erf(x[i,]%*%b))^(1-y[i,])
                 }
                 return(L)
}

# vector of likelihood for each observation
vec_L_i = affairs_probit(betas, xmat, ymat)

# likelihod of all sample
likelihood = prod(vec_L_i)


###=======================================
# Exercise 4 Logit
###=======================================
CDF_logit = function(xx, bb)  {
                 exp(xx%*%bb)/(1 + exp(xx%*%bb))
```

```
}

affairs_logit = function(b, x, y){
                        L = vector()
                        for (i in 1:nrow(y))  {
                          L[i] = CDF_logit(x[i,], b)^y[i,]*
                            (1-CDF_logit(x[i,], b))^(1-y[i,])
                        }
                        return(L)
                  }

# vector of likelihood for each observation in affairs
vec_logit_L = affairs_probit(betas, xmat, ymat)

# likelihod of all sample
MLE_logit = prod(vec_logit_L)


###=======================================
# Exercise 5 Multinomial Logit
###=======================================

# Conditional Logit Model
p_cond_logit = function(xx, bb)  {
                            sums = vector()
                            for(i in ncol(xx)) {
                            sums[i] = xx[,i]*bb
                            }
                        return(exp(xx*bb)/sum(sums))
}

affairs_cond_logit = function(b, x, y){
                            L = vector()
                            for (j in 1:ncol(x))  {

                            for (i in 1:nrow(y))  {
                              L[i] = p_cond_logit(x[i,j], b)^y[i,]*
                                (1-p_cond_logit(x[i,j], b))^(1-y[i,])
                            }
                            }
                            return(L)
}

vec_cond_logit_L = affairs_cond_logit(betas, xmat, ymat)

MLE_cond_logit = prod(vec_cond_logit_L)

# Multinomial Logit Model
p_mult_logit = function(xx, bb)  {
                                    sums = vector()
                                    for(i in ncol(xx)) {
                                      sums[i] = xx*bb[i]
                                    }
                                    return(exp(xx*bb)/sum(sums))
                                  }

affairs_mult_logit = function(b, x, y){
                            L = vector()
                            for (j in 1:nrow(b))  {
                            for (i in 1:nrow(y))  {
                            L[i] = p_cond_logit(x[i,], b[j])^y[i,]*
                            (1-p_cond_logit(x[i,], b[j]))^(1-y[i,])
                            }
                            }
                            return(L)
}

vec_mult_logit_L = affairs_mult_logit(betas, xmat, ymat)

```

```
MLE_mult_logit = prod(vec_mult_logit_L)

# Mixed logit model


```