Econometrics II TA Session *

Wang Xin

Oct 30, 2023

1 Empirical Application of Binary Model: Titanic Survivors

1.1 Background and Data

"Women and children first" is a behavioral norm, which women and children are saved first in a life-threatening situation. This code was made famous by the sinking of the Titanic in 1912. An empirical application investigates characteristics of survivors of Titanic to answer whether crews obeyed the code or not.

We use an open data about Titanic survivors [1]. Although this data set contains many variables, we use only four variables: `survived`, `age`, `fare`, and `sex`. We summarize descriptions of variables as follows:

- `survived`: a binary variable taking 1 if a passenger survived.
- `age`: a continuous variable representing passenger's age.
- `fare`,: a continuous variable representing how much passenger paid.
- `sex`: a string variable representing passenger's sex.

Instead of `sex`, we generate a dummy variable `female`, taking 1 if passenger is female, in regression.

```
dt <- read.csv(
  file = "./titanic.csv",
  header = TRUE, sep = ",", row.names = NULL,
     stringsAsFactors = FALSE)
# set a gender dummy
dt$female <- ifelse(dt$sex == "female", 1, 0)
# delete the missing data
```

*The codes are cited from document by Hiroki Kato.

[1] data source: https://www.kaggle.com/c/titanic/data

```
dt <- subset(dt, !is.na(survived)&!is.na(age)&!is.na(fare)&!
   is.na(female))
summary(dt)
    survived           sex                 age
 Min.   :0.0000   Length:714         Min.   : 0.42
 1st Qu.:0.0000   Class :character   1st Qu.:20.12
 Median :0.0000   Mode  :character   Median :28.00
 Mean   :0.4062                      Mean   :29.70
 3rd Qu.:1.0000                      3rd Qu.:38.00
 Max.   :1.0000                      Max.   :80.00
      fare            female
 Min.   :  0.00   Min.   :0.0000
 1st Qu.:  8.05   1st Qu.:0.0000
 Median : 15.74   Median :0.0000
 Mean   : 34.69   Mean   :0.3655
 3rd Qu.: 33.38   3rd Qu.:1.0000
 Max.   :512.33   Max.   :1.0000
```

In this binary model, the outcome variable is `survived`. Explanatory variables are `age`, `fare`, and `sex`. The probability function should be

$$P[survived = 1|female, age, fare] = F(\beta_0 + \beta_1 female + \beta_2 age + \beta_3 fare).$$

1.2 Probit and Logit Model

Under probit or logit model, the MLE is widely used. Then our first step is to construct criterion function.

The log-likelihood function of observation y_i, conditionally on x_i is

$$M_n(\beta) = \sum_{i=1}^{n} \left(y_i \log\left(F(\mathbf{x}_i\boldsymbol{\beta})\right) + (1 - y_i)\log\left(1 - F(\mathbf{x}_i\boldsymbol{\beta})\right)\right).$$

As a reminder, the probit model is

$$F(\mathbf{x}_i\boldsymbol{\beta}) = \Phi(\mathbf{x}_i\boldsymbol{\beta}),$$

the logit model is

$$F(\mathbf{x}_i\boldsymbol{\beta}) == \frac{exp\left(\mathbf{x}_i\boldsymbol{\beta}\right)}{1 + exp\left(\mathbf{x}_i\boldsymbol{\beta}\right)}$$

In R, the function `nlm()` provides the Newton-Raphson algorithm to minimize the function. To run this function, we need to define the log-likelihood function (`M_n`) beforehand. Moreover, since we need to give initial values in augments, we use coefficients estimated by OLS. Alternatively, we often use `glm()` (generalized linear model) function. It can unify various other statistical models, including linear regression, logistic regression and Poisson regression. Using this function, we do not need to define the log-likelihood function and initial values, and estimates of `glm()` are approximate to estimtaes of `nlm()`. In this case, both functions are displayed.

Let's try the probit model first.

```
Y <- dt$survived
female <- dt$female; age <- dt$age; fare <- dt$fare
dt$"(Intercept)" <- 1
# log-likelihood
M_n <- function(beta, model = c("probit", "logit")) {
  y <- beta[1]+ beta[2]*female + beta[3]*age + beta[4]*fare
  if (model == "probit") {
    L <- pnorm(y) # pnorm() generates a cdf of normal dist.
  } else {
    L <- 1/(1 + exp(-y))
  }
  LL_i <- Y * log(L) + (1 - Y) * log(1 - L)
  LL <- -sum(LL_i)
  return(LL)
}

# probit model
# Newton-Raphson
init <- c(0,0,0,0)
probit <- nlm(M_n, init, model = "probit", hessian = TRUE)

label <- c("(Intercept)", "factor(female)1", "age", "fare")
names(probit$estimate) <- label
colnames(probit$hessian) <- label; rownames(probit$hessian)
   <- label

b_probit <- probit$estimate
vcov_probit <- solve(probit$hessian)
se_probit <- sqrt(diag(vcov_probit))
LL_probit <- -probit$minimum
# glm function
model <- survived ~ factor(female) + age + fare
probit_glm <- glm(model, data = dt, family = binomial("
   probit"))
```

The logit model is also shown below.

```
# logit model
# Newton-Raphson
logit <- nlm(M_n, init, model = "logit", hessian = TRUE)

names(logit$estimate) <- label
```

```
colnames(logit$hessian) <- label; rownames(logit$hessian) <-
    label

b_logit <- logit$estimate
vcov_logit <- solve(logit$hessian)
se_logit <- sqrt(diag(vcov_logit))
LL_logit <- -logit$minimum
# glm function
logit_glm <- glm(model, data = dt, family = binomial("logit"
   ))
```

The results are summarized below. Both probit and logit models estimated by `glm()` can be summarized directly by `R`, while the models estimated by `nlm()` can not. Therefore we need to find the the standard deviation and P-value. The results of these two estimates are summarized in Table 1.

```
> round(b_probit,4)
    (Intercept) factor(female)1             age            fare
        -0.8639          1.4340         -0.0058          0.0073
> summary(probit_glm)
Call:
glm(formula = model, family = binomial("probit"), data = dt)
Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)     -0.864022   0.133186  -6.487 8.74e-11 ***
factor(female)1  1.433964   0.111149  12.901  < 2e-16 ***
age             -0.005843   0.003747  -1.560    0.119
fare             0.007347   0.001515   4.850 1.23e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 ''. 0.1 ''  1
...(Other info. is omitted)

> round(b_logit,4)
    (Intercept) factor(female)1             age            fare
        -1.4127          2.3476         -0.0106          0.0128
> summary(logit_glm)
Call:
glm(formula = model, family = binomial("logit"), data = dt)
Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.412758   0.230874  -6.119 9.41e-10 ***
factor(female)1  2.347599   0.189956  12.359  < 2e-16 ***
age             -0.010570   0.006498  -1.627    0.104
fare             0.012773   0.002696   4.738 2.16e-06 ***
---
```

```
...(Other info. is omitted)
```

```
# z-value
z_probit <- b_probit/se_probit
z_logit <- b_logit/se_logit
# Pr(>|z|)
p_probit <- pnorm(abs(z_probit), lower = FALSE)*2
p_logit <- pnorm(abs(z_logit), lower = FALSE)*2
```

Table 1: nlm Results of Probit and Logit model

	probit	*logistic*
Female = 1	1.434***	2.348***
	(0.111)	(0.190)
	t = 12.917	t = 12.357
	p = 0.000	p = 0.000
age	−0.006	−0.011
	(0.004)	(0.006)
	t = −1.588	t = −1.628
	p = 0.113	p = 0.104
fare	0.007***	0.013***
	(0.001)	(0.003)
	t = 5.089	t = 4.717
	p = 0.00000	p = 0.00001
Constant	−0.864***	−1.413***
	(0.132)	(0.231)
	t = −6.563	t = −6.115
	p = 0.000	p = 0.000
Log-Likelihood	-357.678	-358.035
Percent correctly predicted	0.7759	0.7773
Pseudo R-squared	0.5981	0.5978
Observations	714	714

1.3 Quick interpretation

In the linear probability model (which is rarely used in discrete choice model), interpretations of coefficients are straight-forward. The coefficient β_1 is the change in survival probability given a one-unit increase in continuous variable x. In the case of discrete variable, the coefficient β_1 is the difference in survival probability between two groups. However, when we use the probit or logit model, it is hard for us to interpret results because the partial effect is not constant across other covariates. As an illustration, the partial effect of continuous variable `age` is

$$\partial_{age} P[survived = 1|female, age, fare] = \begin{cases} \phi(\mathbf{x}_i\boldsymbol{\beta})\beta_2 & \text{if probit} \\ \frac{exp(-\mathbf{x}_i\boldsymbol{\beta})}{(1+exp(-\mathbf{x}_i\boldsymbol{\beta}))^2}\beta_2 & \text{if logit} \end{cases}$$

The partial effect of dummy variable `female` is

$$P[survived = 1|female = 1, age, fare] - P[survived = 1|female = 0, age, fare]$$

$$= \begin{cases} \Phi(\beta_0 + \beta_1 female + \beta_2 age + \beta_3 fare) - \Phi(\beta_0 + \beta_2 age + \beta_3 fare) & \text{if probit} \\ \Lambda(\beta_0 + \beta_1 female + \beta_2 age + \beta_3 fare) - \Lambda(\beta_0 + \beta_2 age + \beta_3 fare) & \text{if logit} \end{cases}$$

where $\Lambda(a) = 1/(1 + exp(-a))$.

To interpret probit and logit model roughly, consider 'average' person with respect to age and fare. Average age is about 29.7, and average fare is about 34.7. Then, the survival probability of female is calculated as follows:

```
# probit
m_p <- b_probit[1] + 29.7*b_probit[3] + 34.7*b_probit[4]
female_p <- pnorm(m_p + b_probit[2]) - pnorm(m_p)
# logit
m_l <- b_logit[1] + 29.7*b_logit[3] + 34.7*b_logit[4]
female_l <- 1/(1 + exp(-(m_l + b_logit[2]))) - 1/(1 + exp(-m
   _l))

> print("Probit: Diff of prob. between avg female and male")
   ; female_p
[1] "Probit: Diff of prob. between avg female and male"
(Intercept)
  0.5256639
> print("Logit: Diff of prob. between avg female and male");
    female_l
[1] "Logit: Diff of prob. between avg female and male"
(Intercept)
  0.5265183
```

Above, this method only provides a quick explanation of the model. A more precise interpretation named *average marginal effect* (AME) are also achievable by user-defined function. We will see it if possible.

The coefficients I used to calculate the magnitude of effect shown above are from `nlm()`, and they vary little between probit and logit model. The difference between coefficients from probit and logit model is due to the different distribution functions used. After transformation, they actually show similar partial effects.

1.4 Model Fitness

There are two measurements of goodness-of-fit. First, the *percent correctly predicted* reports the percentage of unit whose predicted $\hat{y}_i$ matches the actual y_i. The predicted $\hat{y}_i$ takes one if $F(\mathbf{x}_i\hat{\boldsymbol{\beta}}) > 0.5$, and takes zero if $F(\mathbf{x}_i\hat{\boldsymbol{\beta}}) \leq 0.5$.

```
# model fitness
X <- as.matrix(dt[,c("(Intercept)", "female", "age", "fare")
  ])
Xb_probit <- X %*% matrix(b_probit, ncol = 1)
Xb_logit <- X %*% matrix(b_logit, ncol = 1)

hatY_probit <- ifelse(pnorm(Xb_probit) > 0.5, 1, 0)
hatY_logit <- ifelse(1/(1 + exp(-Xb_logit)) > 0.5, 1, 0)
# percent correctly predicted
pcp_probit <- round(sum(Y == hatY_probit)/nrow(X), 4)
pcp_logit <- round(sum(Y == hatY_logit)/nrow(X), 4)
```

Second measurement is the *pseudo R-squared*. The pseudo R-squared is obtained by$1 - \sum_i \hat{u}_i^2 / \sum_i \hat{y}_i^2$, where$\hat{u}_i = \hat{y}_i - F(\mathbf{x}_i\hat{\boldsymbol{\beta}})$.

```
# pseudo R-squared
Y2 <- Y^2
hatu_probit <- (Y - pnorm(Xb_probit))^2
hatu_logit <- (Y - 1/(1 + exp(-Xb_logit)))^2

pr2_probit <- round(1 - sum(hatu_probit)/sum(Y2), 4)
pr2_logit <- round(1 - sum(hatu_logit)/sum(Y2), 4)
```

2 Empirical Application of Multinomial Model: Gender Discrimination in Job Position

2.1 Background and Data

Recently, many developed countries move toward women's social advancement, for example, an increase of number of board member. In this application, we explore whether the gender discrimination existed in the U.S. bank industry. Our hypothesis is that women are less likely to be given a higher position than male.

We use a built-in data set called `BankWages` in the library `AER`. This data set contains the following variables:

- `job`: three job position. The rank of position is `custodial` < `admin` < `manage`
- `education`: years of education
- `gender`: a dummy variable of female

We begin with splitting data into two subsets: the training data and the test data. The training data, which is used for estimation and model fitness, is randomly drawn from the original data. The sample size of this subset is two thirds of total observations of the original one ($N = 316$). The test data, which is used for model prediction, consists of observations which the training data does not include ($N = 158$).

To use the multinomial logit model in R, we need to transform outcome variable into the form factor, which is special variable form in R. The variable form factor is similar to dummy variables. For example, `factor(dt$job, levels = c("admin", "custodial", "manage"))` transforms the variable form `job` from the form `character` as explanatory variables, R automatically makes two dummy variables of `custodial` and `manage`.

```
library(AER)
data(BankWages)
dt <- BankWages
dt$job <- as.character(dt$job)
dt$job <- factor(dt$job, levels = c("admin", "custodial", "
   manage"))
dt <- dt[,c("job", "education", "gender")]
# split data into training set and test set
set.seed(120511)
train_id <- sample(1:nrow(dt), size = (2/3)*nrow(dt),
   replace = FALSE)
train_dt <- dt[train_id,]; test_dt <- dt[-train_id,]
summary(train_dt)
        job         education        gender
 admin    :240   Min.   : 8.00   male  :178
 custodial: 18   1st Qu.:12.00   female:138
 manage   : 58   Median :12.00
                 Mean   :13.52
                 3rd Qu.:15.00
                 Max.   :21.00
```

2.2 Model

The outcome variable y_i takes three values $0, 1, 2$. Note that the labelling of the choices is arbitrary. Then, the multinomial logit model has the following response probabilities

$$P_{ij} = P(y_i = j|\mathbf{x}_i) = \begin{cases} \frac{exp(\mathbf{x}_i\boldsymbol{\beta}_j)}{1+\sum_{k=1}^{2} exp(\mathbf{x}_i\boldsymbol{\beta}_k)} & \text{if } j = 1, 2 \\ \\ \frac{1}{1+\sum_{k=1}^{2} exp(\mathbf{x}_i\boldsymbol{\beta}_k)} & \text{if } j = 0 \end{cases},$$

and $\boldsymbol{\beta}_0 = 0$

The log-likelihood function is

$$M_n(\beta_1, \beta_2) = \sum_{i=1}^{n} \sum_{j=0}^{2} d_{ij} \log(P_{ij}).$$

where d_{ij} is a dummy variable taking 1 if $y_i = j$.

In R, some packages provide the multinomial logit model. In this application, we use the multinom function in the library `nnet`.

```
library(nnet)
est_mlogit <- multinom(job ~ education + gender, data =
  train_dt)
```

Table 2: Multinomial Logit Model of Job Position

	Dependent variable:	
	custodial	manage
	(1)	(2)
Education	-0.547^{***}	1.322^{***}
	(0.116)	(0.229)
Female = 1	-10.507	-0.891^{*}
	(31.352)	(0.524)
Constant	4.634^{***}	-21.448^{***}
	(1.269)	(3.605)
Observations	316	
Percent correctly predicted (in-sample)	0.839	
Percent correctly predicted (out-of-sample)	0.88	
Log-likelihood	-102.964	
Pseudo R-squared	0.523	

2.3 Interpretaions and Model Fitness

Table 2 summarizes the result of multinomial logit model. The coefficient represents the change of $log(P_{ij}/P_{i0})$ in corresponding covariates because the response probabilities yields

$$\frac{P_{ij}}{P_{i0}} = exp(\mathbf{x}_i \boldsymbol{\beta}_j) \Leftrightarrow \log\left(\frac{P_{ij}}{P_{i0}}\right) = \mathbf{x}_i \boldsymbol{\beta}_j$$

For example, `eduction` decreases the log-odds between `custodial` and `admin` by -0.547. This implies that those who received higher education are more likely to obtain the position `admin`. Highly-educated workers are also more likely to obtain the position `manage`. Moreover, a `female` dummy decrease the log-odds between `manage` and `admin` by -0.891, which implies that females are less likely to obtain higher position `manage`. From this result, we conclude that the U.S. bank discouraged females to assign higher job position. Again, we still use pseudo R-squared and percent correctly predicted to evaluate model fitness and prediction. The preudo R-sqaured is calculated by $1 - L_1/L_0$ where L_1 is the value of log-likelihood for estimated model and L_0 is the value of log-likelihood in the model with only an intercept. Note that `nnet:::logLik.multinom()` returns the value of log-likelihood.

```
loglik1 <- as.numeric(nnet:::logLik.multinom(est_mlogit))
est_mlogit0 <- multinom(job ~ 1, data = train_dt)
loglik0 <- as.numeric(nnet:::logLik.multinom(est_mlogit0))
# pseudo R-sqaured
pr2 <- round(1 - loglik1/loglik0, 3)
```

The second index is the percent correctly predicted. The predicted outcome is the outcome with the highest estimated probability. Using the training data (in-sample) and the test data (out-of-sample), we calculate this index.

```
# in-sample prediction
inpred <- predict(est_mlogit, newdata = train_dt, "probs")
inpred <- colnames(inpred)[apply(inpred, 1, which.max)]
inpcp <- round(sum(inpred == train_dt$job)/length(inpred),
  3)
# out-of-sample prediction
outpred <- predict(est_mlogit, newdata = test_dt, "probs")
outpred <- colnames(outpred)[apply(outpred, 1, which.max)]
outpcp <- round(sum(outpred == test_dt$job)/length(outpred),
   3)
```

The fitness data along with coefficient results are summarized in Table 2. our model is good in terms of fitness and prediction because the percent correctly predicted is high (83.9% of in-sample data and 88.0% of out-of-sample data), and the pseudo R-sqaured is 0.523.