Econometrics II TA Session #3

Hiroki Kato

1 Empirical Application of Binary Model: Titanic Survivors

Brief Background. "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.

Data. We use an open data about Titanic survivors [1]. Although this dataset 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 passeger's age.
- `fare`: a continuous variable representing how much passeger paid.
- `sex`: a string variable representing passenger's sex.

Using `sex`, we will make a binary variable, called `female`, taking 1 if passeger is female. Intead of `sex`, we use `female` variable in regression.

Moreover, we split data into two subsets: the *training* data and the *test* data. The training data is randomly drawn from the original data. The sample size of this data is two thirs of total observations. We use the training data (*in-sample*) to estimate and evaluate model fitness. The test data consists of observations which the training data does not include. We use the test data (*out-of-sample*) to evaluate model prediction.

```
dt <- read.csv(
  file = "./data/titanic.csv",
  header = TRUE,  sep = ",", row.names = NULL,  stringsAsFactors = FALSE)

dt$female <- ifelse(dt$sex == "female", 1, 0)
dt <- subset(dt, !is.na(survived)&!is.na(age)&!is.na(fare)&!is.na(female))
dt <- dt[,c("survived", "age", "fare", "female")]

set.seed(120511)
```

[1] data source: http://biostat.mc.vanderbilt.edu/DataSets.

```r
train_id <- sample(1:nrow(dt), size = (2/3)*nrow(dt), replace = FALSE)
train_dt <- dt[train_id,]
test_dt <- dt[-train_id,]

head(dt)
```

```
##   survived   age     fare female
## 1        1 29.00 211.3375      1
## 2        1  0.92 151.5500      0
## 3        0  2.00 151.5500      1
## 4        0 30.00 151.5500      0
## 5        0 25.00 151.5500      1
## 6        1 48.00  26.5500      0
```

Model. In a binary model, a dependent (outcome) variable y_i takes only two values, i.e., $y_i \in \{0, 1\}$. A binary variable is sometimes called a *dummy* variable. In this application, the outcome variable is `survived`. Explanatory variables are `female`, `age`, and `fare`. The regression function is

$$
\begin{aligned}
&E[survived|female, age, fare] \\
=&\mathbb{P}[survived = 1|female, age, fare] = G(\beta_0 + \beta_1 female + \beta_2 age + \beta_3 fare).
\end{aligned}
$$

The function $G(\cdot)$ is arbitrary function. In practice, we often use following three specifications:

- Linear probability model (LPM): $G(\mathbf{x}_i\beta) = \mathbf{x}_i\beta$.
- Probit model: $G(\mathbf{x}_i\beta) = \Phi(\mathbf{x}_i\beta)$ where $\Phi(\cdot)$ is the standard Gaussian cumulative function.
- Logit model: $G(\mathbf{x}_i\beta) = 1/(1 + \exp(-\mathbf{x}_i\beta))$.

1.1 Linear Probability Model

The linear probability model specifys that $G(a)$ is linear in a, that is,

$$\mathbb{P}[survived = 1|female, age, fare] = \beta_0 + \beta_1 female + \beta_2 age + \beta_3 fare.$$

This model can be estimated using the OLS method. In `R`, we can use the OLS method, running `lm()` function.

```r
model <- survived ~ factor(female) + age + fare
LPM <- lm(model, data = train_dt)
```

However, `lm()` function does not deal with heteroskedasticity problem. To resolve it, we need to claculate heteroskedasticity-robust standard errors using the White method.

$$\hat{V}(\hat{\beta}) = \left(\frac{1}{n}\sum_i \mathbf{x}_i'\mathbf{x}_i\right)^{-1} \left(\frac{1}{n}\sum_i \hat{u}_i^2\mathbf{x}_i'\mathbf{x}_i\right) \left(\frac{1}{n}\sum_i \mathbf{x}_i'\mathbf{x}_i\right)^{-1}$$

```r
# heteroskedasticity-robust standard errors
train_dt$"(Intercept)" <- 1
X <- as.matrix(train_dt[,c("(Intercept)", "female", "age", "fare")])
u <- diag(LPM$residuals^2)

XX <- t(X) %*% X
avgXX <- XX * nrow(X)^{-1}
inv_avgXX <- solve(avgXX)

uXX <- t(X) %*% u %*% X
avguXX <- uXX * nrow(X)^{-1}

vcov_b <- (inv_avgXX %*% avguXX %*% inv_avgXX) * nrow(X)^{-1}
rse_b <- sqrt(diag(vcov_b))

label <- c("(Intercept)", "factor(female)1", "age", "fare")
names(rse_b) <- label

# homoskedasticity-based standard errors
se_b <- sqrt(diag(vcov(LPM)))

print("The Variance of OLS"); vcov(LPM)
```

```
## [1] "The Variance of OLS"

##                   (Intercept) factor(female)1           age          fare
## (Intercept)      1.505787e-03   -3.905773e-04 -3.676396e-05 -5.951346e-07
## factor(female)1 -3.905773e-04    1.089299e-03  2.569835e-06 -2.154400e-06
## age             -3.676396e-05    2.569835e-06  1.264948e-06 -6.274261e-08
## fare            -5.951346e-07   -2.154400e-06 -6.274261e-08  9.167801e-08
```

```r
print("The Robust variance of OLS"); vcov_b
```

```
## [1] "The Robust variance of OLS"

##               (Intercept)        female           age          fare
## (Intercept)  1.810499e-03 -3.968956e-04 -4.601203e-05  8.979498e-07
## female      -3.968956e-04  1.239665e-03  4.975911e-06 -4.566026e-06
## age         -4.601203e-05  4.975911e-06  1.476806e-06 -7.956793e-08
## fare         8.979498e-07 -4.566026e-06 -7.956793e-08  7.846876e-08
```

```r
print("The Robust se using White method"); rse_b
```

```
## [1] "The Robust se using White method"

##     (Intercept) factor(female)1             age            fare
##    0.0425499596    0.0352088828    0.0012152389    0.0002801228
```

Using the package `lmtest` and `sandwich` is the easiest way to calculate heteroskedasticity-robust standard errors.

```
library(lmtest) #use function `coeftest`
library(sandwich) #use function `vcovHC`
coeftest(LPM, vcov = vcovHC(LPM, type = "HC0"))[, "Std. Error"]
```

```
##     (Intercept) factor(female)1             age            fare
##    0.0425499596    0.0352088828    0.0012152389    0.0002801228
```

Finally, we summarize results of linear probability model in table 1. We will discuss interpretation of results and goodness-of-fit of LPM later.

```
library(stargazer)
stargazer(
  LPM, LPM,
  se = list(se_b, rse_b),
  t.auto = FALSE, p.auto = FALSE,
  report = "vcs", keep.stat = c("n"),
  covariate.labels = c("Female = 1"),
  add.lines = list(
    c("Standard errors", "Homoskedasticity-based", "Heteroskedasticity-robust")),
  title = "Results of Linear Probability Model", label = "LPM",
  type = "latex", header = FALSE, font.size = "small",
  omit.table.layout = "n", table.placement = "h"
)
```

1.2 Probit and Logit Model

Unlike LPM, the probit and logit model must be estimated using the ML method. The probability of observing y_i is

$$p_\beta(y_i|\mathbf{x}_i) = \mathbb{P}(y_i = 1|x_i)^{y_i}[1 - \mathbb{P}(y_i = 1|x_i)]^{1-y_i} = G(\mathbf{x}_i\beta)^{y_i}(1 - G(\mathbf{x}_i\beta))^{1-y_i}.$$

Taking logalithm yields

$$\log p_\beta(y_i|\mathbf{x}_i) = y_i \log(G(\mathbf{x}_i\beta)) + (1 - y_i)\log(1 - G(\mathbf{x}_i\beta)).$$

The log-likelihood is

$$M_n(\beta) = \sum_{i=1}^n \log p_\beta(y_i|\mathbf{x}_i).$$

The MLE $\hat{\beta}$ holds that the score, which is the first-order derivatives with respect to β, is equal to 0. That is $\nabla_\beta M_n(\hat{\beta}) = 0$. For both logit and probit model, the Hessian matrix, $\nabla^2_{\beta\beta'} M_n(\beta)$, is always negative definite. This implies that log-likelihood function based on both models is grobally concave, and ensures that the MLE maximizes the log-likelihood

Table 1: Results of Linear Probability Model

	Dependent variable:	
	survived	
	(1)	(2)
Female = 1	0.512	0.512
	(0.033)	(0.035)
age	−0.003	−0.003
	(0.001)	(0.001)
fare	0.001	0.001
	(0.0003)	(0.0003)
Constant	0.245	0.245
	(0.039)	(0.043)
Standard errors	Homoskedasticity-based	Heteroskedasticity-robust
Observations	696	696

function. The first-order condition of the probit model is

$$\nabla_\beta M_n(\hat{\beta}) = \sum_{i=1}^{n} \left(y_i - \Phi(\mathbf{x}_i\hat{\beta}) \right) \frac{\phi(\mathbf{x}_i\hat{\beta})}{\Phi(\mathbf{x}_i\hat{\beta})(1-\phi(\mathbf{x}_i\hat{\beta}))} = 0.$$

The first-order condition of the logit model is

$$\nabla_\beta M_n(\hat{\beta}) = \sum_{i=1}^{n} \left(y_i - G(\mathbf{x}_i\hat{\beta}) \right) \mathbf{x}_i' = 0.$$

Since it is hard for us to solve this condition analytically, we obtain estimators using numerical procedure.

The asymptotic distribution of $\hat{\beta}$ is $\hat{\beta} \xrightarrow{d} N(\beta, \Sigma_\beta)$ where

$$\Sigma_\beta = -\left(\sum_i E[E[\nabla^2_{\beta\beta'} \log p_\beta(y_i|\mathbf{x}_i)|\mathbf{x}_i]] \right)^{-1}.$$

In practice, we replace $E[E[\nabla^2_{\beta\beta'} \log p_\beta(y_i|\mathbf{x}_i)|\mathbf{x}_i]]$ by

$$\frac{1}{n}\sum_i \nabla^2_{\beta\beta'} \log p_{\hat{\beta}}(y_i|\mathbf{x}_i),$$

that is,

$$\hat{\Sigma}_\beta = \left(\sum_i \nabla^2_{\beta\beta'} (-\log p_{\hat{\beta}}(y_i|\mathbf{x}_i)) \right)^{-1}.$$

In R, the function `nlm()` provides the Newton-Raphson algorithm to minimize the function [2]. To run this function, we need to define the log-likelihood function (`LnLik`) beforehand. Moreover, since we need to give initial values in augments, we use coefficients estimated by OLS. Alternatively, we often use `glm()` function. Using this function, we do not need to define the log-likelihood function and initial values. Since estimates of `glm()` are approximate to estiamtes of `nlm()`, we can use this command safely. In this application, we use `nlm` function to minimize the log-likelihood function.

```
Y <- train_dt$survived
female <- train_dt$female
age <- train_dt$age
fare <- train_dt$fare

# log-likelihood
LnLik <- function(b, model = c("probit", "logit")) {

  xb <- b[1]+ b[2]*female + b[3]*age + b[4]*fare

  if (model == "probit") {
    L <- pnorm(xb)
  } else {
    L <- 1/(1 + exp(-xb))
  }

  LL_i <- Y * log(L) + (1 - Y) * log(1 - L)
  LL <- -sum(LL_i)

  return(LL)
}

#Newton-Raphson
init <- c(0.169, 0.520, -0.0002, 0.001)
probit <- nlm(LnLik, 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
```

[2] `optim()` function is an another way to minimize the function. Especially, the function `optim(method = "BFGS")` provides the Quasi-Newton algorithm which carries on the spirit of Newton method.

```r
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 = train_dt, family = binomial("probit"))

#result
print("The MLE of probit model using nlm"); b_probit
```

```
## [1] "The MLE of probit model using nlm"

##     (Intercept) factor(female)1             age            fare
##    -0.740010404     1.440663450    -0.009316882     0.006302940
```

```r
print("The Variance of probit model using nlm"); vcov_probit
```

```
## [1] "The Variance of probit model using nlm"

##                   (Intercept) factor(female)1           age          fare
## (Intercept)      1.764185e-02   -4.735516e-03 -4.149486e-04 -2.453847e-05
## factor(female)1 -4.735516e-03    1.255295e-02  8.495496e-06 -5.592007e-06
## age             -4.149486e-04    8.495496e-06  1.512962e-05 -9.929199e-07
## fare            -2.453847e-05   -5.592007e-06 -9.929199e-07  1.737151e-06
```

```r
print("The se of probit model using nlm"); se_probit
```

```
## [1] "The se of probit model using nlm"

##     (Intercept) factor(female)1             age            fare
##     0.132822608     0.112039969     0.003889681     0.001318010
```

```r
print("The coefficients of probit using glm"); coef(probit_glm)
```

```
## [1] "The coefficients of probit using glm"

##     (Intercept) factor(female)1             age            fare
##    -0.740094134     1.440662013    -0.009314690     0.006303577
```

```r
print("The se of probit using glm"); sqrt(diag(vcov(probit_glm)))
```

```
## [1] "The se of probit using glm"

##     (Intercept) factor(female)1             age            fare
##     0.134738833     0.112061942     0.003966673     0.001326048
```

Using `LogLik`, we can also estimate logit model by Newton-Raphson algorithm. To compare result, we also use `glm()` function.

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

label <- c("(Intercept)", "factor(female)1", "age", "fare")
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 = train_dt, family = binomial("logit"))

#result
print("The MLE of logit model"); b_logit
```

```
## [1] "The MLE of logit model"
```

```
##     (Intercept) factor(female)1             age            fare
##     -1.19071868      2.36579523     -0.01665811      0.01049121
```

```r
print("The Variance of logit model"); vcov_logit
```

```
## [1] "The Variance of logit model"
```

```
##                   (Intercept) factor(female)1           age          fare
## (Intercept)      5.347251e-02   -1.306856e-02 -1.260674e-03 -7.166131e-05
## factor(female)1 -1.306856e-02    3.678907e-02 -4.389835e-05 -2.773805e-06
## age             -1.260674e-03   -4.389835e-05  4.703086e-05 -3.343743e-06
## fare            -7.166131e-05   -2.773805e-06 -3.343743e-06  5.199195e-06
```

```r
print("The se of logit model"); se_logit
```

```
## [1] "The se of logit model"
```

```
##     (Intercept) factor(female)1             age            fare
##     0.231241234     0.191804780     0.006857905     0.002280174
```

```r
print("The coefficients of logit using glm"); coef(logit_glm)
```

```
## [1] "The coefficients of logit using glm"
```

```
##     (Intercept) factor(female)1             age            fare
##     -1.19080405      2.36579304     -0.01665588      0.01049185
```

```r
print("The se of logit using glm"); sqrt(diag(vcov(logit_glm)))
```

```
## [1] "The se of logit using glm"
```

```
##     (Intercept) factor(female)1             age            fare
##     0.231133819     0.191810415     0.006862245     0.002272391
```

As a result, table 2 summarizes results of probit model and logit model. Standard errors are in parentheses. We will discuss interpretation of results and goodness-of-fit later.

```r
stargazer(
  probit_glm, logit_glm,
  coef = list(b_probit, b_logit), se = list(se_probit, se_logit),
  t.auto = FALSE, p.auto = FALSE,
  report = "vcs", keep.stat = c("n"),
  covariate.labels = c("Female = 1"),
  add.lines = list(
    c("Log-Likelihood", round(LL_probit, 3), round(LL_logit, 3))),
  title = "Results of Probit and Logit model",
  label = "probit_logit",
  type = "latex", header = FALSE, font.size = "small",
  table.placement = "h", omit.table.layout = "n"
)
```

Table 2: Results of Probit and Logit model

	Dependent variable:	
	survived	
	probit	*logistic*
	(1)	(2)
Female = 1	1.441	2.366
	(0.112)	(0.192)
age	−0.009	−0.017
	(0.004)	(0.007)
fare	0.006	0.010
	(0.001)	(0.002)
Constant	−0.740	−1.191
	(0.133)	(0.231)
Log-Likelihood	-351.507	-351.873
Observations	696	696

1.3 Interpretaions

In the linear probability model, interepretations 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 interepret results because the partial effect is not constant across other covariates. As an illustration, the partial effect of continuous variable `age` is

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

The partial effect of dummy variable `female` is

$$\begin{aligned} &\mathbb{P}[survived = 1|female = 1, age, fare] - \mathbb{P}[survived = 1|female = 0, age, fare] \\ &= \begin{cases} \beta_1 & \text{if LPM} \\ \Phi(\beta_0 + \beta_1 + \beta_2 age + \beta_3 fare) - \Phi(\beta_0 + \beta_2 age + \beta_3 fare) & \text{if Probit} \\ \Lambda(\beta_0 + \beta_1 + \beta_2 age + \beta_3 fare) - \Lambda(\beta_0 + \beta_2 age + \beta_3 fare) & \text{if Logit} \end{cases}, \end{aligned}$$

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

Table 3 shows results of linear probability model, probit model, and logit model. Qualitatively, all specifications shows same trend. The survival probability of females is greater than of male. The survival probability is decreaseing in age. Quantitatively, LPM shows that the survival probability of female is about 50% point higher than of male. Moreover, the survival probability of 0-year-old baby is about 0.3% point less than of 100-year-old elderly. This implies that the survival probability is not largely changed by age. To evaluate probit and logit model quantitatively, consider 'average' person with respect to `age` and `fare`. Average age is about 30, and average fare is about 37. Then, the survival probability of female is calculated as follows:

```
#probit
cval_p <- b_probit[1] + 30*b_probit[3] + 37*b_probit[4]
female_p <- pnorm(cval_probit + b_probit[2]) - pnorm(cval_probit)
#logit
cval_l <- b_logit[1] + 30*b_logit[3] + 37*b_logit[4]
female_l <- 1/(1 + exp(-(cval_l + b_logit[2]))) - 1/(1 + exp(-cval_l))
# result
print("Probit: Diff of prob. b/w average female and male"); female_p
```

```
## [1] "Probit: Diff of prob. b/w average female and male"
```

```
## (Intercept)
##    0.527715
```

```
print("Logit: Diff of prob. b/w average female and male"); female_l
```

```
## [1] "Logit: Diff of prob. b/w average female and male"
```

```
## (Intercept)
##     0.52958
```

As a result, in terms of the difference of survival probability between females and males the probit and logit model obtain similar result to LPM. In the same way, we can calculate the partial effect of age in the probit and logit model, but we skip this. If you have an interest, please try yourself. Overall, crews obeyed the code of "women and children first", but the survival probability of children is not largely different from of adult.

1.4 Model Fitness

There are two measurements of goodness-of-fit. First, the *percent correctly predicted* reports the percentage of unit whose predicted y_i matches the actual y_i. The predicted y_i takes one if $G(\mathbf{x}_i\hat{\beta}) > 0.5$, and takes zero if $G(\mathbf{x}_i\hat{\beta}) \leq 0.5$. We calculate this index, using the training data and the test data.

```
# In-sample
in_Y <- train_dt$survived
in_X <- as.matrix(train_dt[,c("(Intercept)", "female", "age", "fare")])

in_Xb_lpm <- in_X %*% matrix(coef(LPM), ncol = 1)
in_Xb_probit <- in_X %*% matrix(b_probit, ncol = 1)
in_Xb_logit <- in_X %*% matrix(b_logit, ncol = 1)

in_hatY_lpm <- ifelse(in_Xb_lpm > 0.5, 1, 0)
in_hatY_probit <- ifelse(pnorm(in_Xb_probit) > 0.5, 1, 0)
in_hatY_logit <- ifelse(1/(1 + exp(-in_Xb_logit)) > 0.5, 1, 0)

in_pcp_lpm <- round(sum(in_Y == in_hatY_lpm)/nrow(in_X), 4)
in_pcp_probit <- round(sum(in_Y == in_hatY_probit)/nrow(in_X), 4)
in_pcp_logit <- round(sum(in_Y == in_hatY_logit)/nrow(in_X), 4)

# Out-of-sample
out_Y <- test_dt$survived
test_dt$"(Intercept)" <- 1
out_X <- as.matrix(test_dt[,c("(Intercept)", "female", "age", "fare")])

out_Xb_lpm <- out_X %*% matrix(coef(LPM), ncol = 1)
out_Xb_probit <- out_X %*% matrix(b_probit, ncol = 1)
out_Xb_logit <- out_X %*% matrix(b_logit, ncol = 1)

out_hatY_lpm <- ifelse(out_Xb_lpm > 0.5, 1, 0)
```

```r
out_hatY_probit <- ifelse(pnorm(out_Xb_probit) > 0.5, 1, 0)
out_hatY_logit <- ifelse(1/(1 + exp(-out_Xb_logit)) > 0.5, 1, 0)

out_pcp_lpm <- round(sum(out_Y == out_hatY_lpm)/nrow(out_X), 4)
out_pcp_probit <- round(sum(out_Y == out_hatY_probit)/nrow(out_X), 4)
out_pcp_logit <- round(sum(out_Y == out_hatY_logit)/nrow(out_X), 4)
```

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

```r
Y2 <- in_Y^2

hatu_lpm <- (in_Y - in_Xb_lpm)^2
hatu_probit <- (in_Y - pnorm(in_Xb_probit))^2
hatu_logit <- (in_Y - 1/(1 + exp(-in_Xb_logit)))^2

pr2_lpm <- round(1 - sum(hatu_lpm)/sum(Y2), 4)
pr2_probit <- round(1 - sum(hatu_probit)/sum(Y2), 4)
pr2_logit <- round(1 - sum(hatu_logit)/sum(Y2), 4)
```

Table 3 summarizes two measurements of model fitness. There is little difference among LPM, probit model, and logit model.

```r
stargazer(
  LPM, probit_glm, logit_glm,
  coef = list(coef(LPM), b_probit, b_logit),
  se = list(rse_b, se_probit, se_logit),
  t.auto = FALSE, p.auto = FALSE,
  omit = c("Constant"), covariate.labels = c("Female = 1"),
  report = "vcs", keep.stat = c("n"),
  add.lines = list(
    c("Percent correctly predicted (in-sample)",
      in_pcp_lpm, in_pcp_probit, in_pcp_logit),
    c("Percent correctly predicted (out-of-sample)",
      out_pcp_lpm, out_pcp_probit, out_pcp_logit),
    c("Pseudo R-squared", pr2_lpm, pr2_probit, pr2_logit)
  ),
  omit.table.layout = "n", table.placement = "t",
  title = "Titanic Survivors: LPM, Probit, and Logit",
  label = "titanic",
  type = "latex", header = FALSE
)
```

Table 3: Titanic Survivors: LPM, Probit, and Logit

	Dependent variable:		
	survived		
	OLS	*probit*	*logistic*
	(1)	(2)	(3)
Female = 1	0.512	1.441	2.366
	(0.035)	(0.112)	(0.192)
age	−0.003	−0.009	−0.017
	(0.001)	(0.004)	(0.007)
fare	0.001	0.006	0.010
	(0.0003)	(0.001)	(0.002)
Percent correctly predicted (in-sample)	0.7802	0.7744	0.7744
Percent correctly predicted (out-of-sample)	0.7794	0.7765	0.7765
Pseudo R-squared	0.5869	0.5873	0.5869
Observations	696	696	696