Oct 12, 2019 Binary Probit Scratch using R

Name: Jikhan Jeong
Ref: Econometric Modelling with Time Series

This is for scratch of binary probit regression (Done) Now, I will move to ordered probit and mixed logit

1)**chunk n: Ctrl + Alt + I**
2)**knit: Ctrl + Shift + k**
3)**run: Ctrl + Enter**

```
setwd('C:/Users/jikhan.jeong/Documents/R/Econ_Modelling_R/New folder')
```

```
getwd()
```

```
## [1] "C:/Users/jikhan.jeong/Documents/R/Econ_Modelling_R"
```

```
rm (list = ls(all=TRUE))
graphics.off()
```

Load required functions - inv, figure, seqa

```
source("C:/Users/jikhan.jeong/Documents/R/Econ_Modelling_R/New folder/EMTSUtil.R")
```

Unrestricted Probit negative log-likelihood function prom: the function pnorm returns the integral from $-\infty$ to q of the pdf of the normal distribution where q is a Z-score.
Simply, this is a cumulative normal distribution one tail probability in 50%

```
pnorm(0) # 0.5 probability
```

```
## [1] 0.5
```

one tail probabiltiy in 97.5%

```
pnorm(1.96)
```

```
## [1] 0.9750021
```

The unrestricted Log likelihood contains all covaritate

%*% : matrix multiplication

example A <- matrix (c(1,3,4, 5,8,9, 1,3,3), 3,3) B <- matrix (c(2,4,5, 8,9,2, 3,4,5), 3,3) C = A %*% B

- a Bernoulli distribution

$$f(y;\theta) = \phi_t^{y_t}(1-\phi_t)^{1-y_t}$$

The cumulative normal distribution = pnorm in r

$$\phi_t = \phi(\beta \times X) = Pr(y=1) = \int_{-\infty}^{\frac{\beta \times X}{\sigma}} \frac{1}{\sqrt{2\pi}} \exp[-\frac{s^2}{2} ds], \quad \sigma = 1$$

$$1 - \phi_t = Pr(y=0)$$

*The log-likelihood function for a sample N observations is

$$\ln L_n(\theta) = \frac{1}{n} \sum_{1}^{n} [y_t \ln \phi_t + (1-y_t) \ln(1-\phi_t)]$$

```
lprobit <- function (b,y,x) {
  f  <- pnorm(x %*% b) # culumative pdf from -oo the values
  lf <- -mean( y*log(f) + (1 - y)*log(1 - f) ) # negative log-likelihood
  return (lf)
}
```

Restricted Probit negative log-likelihood function

without covariate x

```
l0probit <- function(b,y) {
  f  <- pnorm(b)
  lf <- -mean( y*log(f) + (1 - y)*log(1 - f) )
  return(lf)
}
```

input the data from usmoney data

```
usmoney <- as.matrix(read.table("C:/Users/jikhan.jeong/Documents/R/Econ_Modelling_R/New folder/usmone
y.dat"))
target  <- usmoney[,2]
bin     <- usmoney[,4]
fomc    <- usmoney[,8]
spread6 <- usmoney[,20]
inf     <- usmoney[,23]
gdp     <- usmoney[,28]
```

Reverse the spread so it is the Federal funds rate less 6-month Treasury bill rate
put - sign on spread6

```
spread <- -spread6
```

Redefine the target rate based on the consolidated series constructed in Hamilton and Jorda (2002)

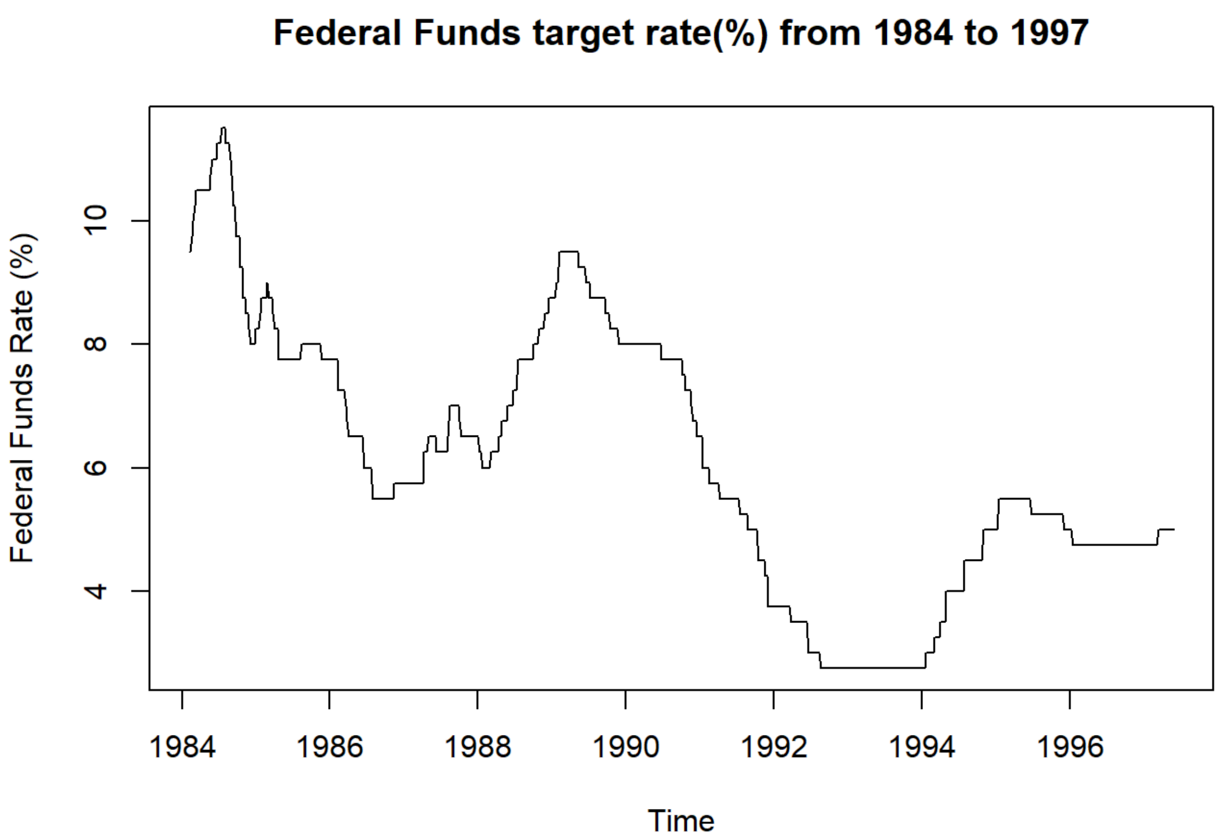
```
target_adj <- cumsum( c(target[1], bin[2:length(bin)] ))
length(target_adj)
```

```
## [1] 693
```

to do Sep 14, 2019

what is seqa waht is cbind what is optim

```
figure()
plot(seqa(1984+5/52,1/52,length(target_adj)),target_adj, type="l",
       main="Federal Funds target rate(%) from 1984 to 1997",
       ylab = "Federal Funds Rate (%)",
       xlab = "Time")
```



Choose data based on fomc days

```
ind       <- fomc == 1
data      <- cbind(bin, spread, inf, gdp)
data_fomc <- data[ind,]
```

Dependent and independent variables **cat** Use cat to print information to an end-user from a function.

```
y <- as.numeric(data_fomc[,1] > 0.0)
t <- length(y)
x <- cbind(rep(1, t), data_fomc[,2:4])
```

```
# class(y)
# class(x) # x contains constant = 1
# df <-cbind(y,x)
# write.csv(df, 'binary_probit.csv') # for comparing stata
```

#Estimate model by OLS (ie ignore that the data are binary)

$$\sigma = \sqrt{E(X-\mu)^2} = \sqrt{E[X^2] - (E[X]^2)}$$

```
reg <- lm(y ~ x - 1) # -1 no constant
summary(reg)
```

```
##
## Call:
## lm(formula = y ~ x - 1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.44738 -0.14136 -0.08095  0.05764  0.89689
##
## Coefficients:
##         Estimate Std. Error t value Pr(>|t|)
## x       -0.01300    0.15843  -0.082   0.9348
## xspread  0.27794    0.05856   4.746 6.78e-06 ***
## xinf     4.02155    3.99133   1.008   0.3160
## xgdp     0.02957    0.01463   2.021   0.0458 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2612 on 102 degrees of freedom
## Multiple R-squared:  0.3038, Adjusted R-squared:  0.2765
## F-statistic: 11.13 on 4 and 102 DF,  p-value: 1.569e-07
```

```
b <- reg$coef
u <- reg$residuals # error
s <- sqrt( mean(u^2) ) # squaredeviation of X
s
```

```
## [1] 0.2562725
```

#Estimation the unrestricted probit regression model by MLE

- I don't know why the estimated value is different from that of glm probit
- -> The reason is optimization method,

optim : General-purpose optimization : "optimization method is a big matter"

- optim(par, fn, gr, method =)
- par = Initial values for the parameters to be optimized over.
- fn = a function to be minimized, this is the reason why using negative LL in the above
- *"BFGS"* is a quasi-Newton method

- *"Nelder-Mead"* Nelder and Mead (1965), that uses only function values and is robust but relatively slow. It will work reasonably well for non-differentiable functions.

```
# theta0 <- b/s assumes s=1

theta0 <- b/s

estResults <- optim(theta0, lprobit, x=x, y=y, method="Nelder-Mead", hessian=T)

theta1 <- estResults$par
l1 <- estResults$val
h <- estResults$hessian

cov <- (1/t)*inv(h)
cov
```

```
##                   x     xspread       xinf        xgdp
## x         2.9252439 -0.72162750  -65.491296 -0.20939868
## xspread  -0.7216275  0.43443500   17.399733  0.03572998
## xinf    -65.4912960 17.39973257 1539.882076  3.96594841
## xgdp     -0.2093987  0.03572998    3.965948  0.02803314
```

```
variance  = diag(cov)

standard.error = sqrt(variance) # standard.error of each variables
standard.error
```

```
##         x    xspread       xinf       xgdp
##  1.7103344  0.6591168 39.2413312  0.1674310
```

```
l1 <- -l1

cat('\nUnrestricted log-likelihood function =     ',l1)
```

```
##
## Unrestricted log-likelihood function =      -0.1918418
```

```
cat('\nT x unrestricted log-likelihood function = ',t*l1)
```

```
##
## T x unrestricted log-likelihood function =  -20.33524
```

```
cat('\n\nUnrestricted parameter estimates')
```

```
##
##
## Unrestricted parameter estimates
```

```
cat('₩n',theta1)
```

```
##
##  -3.858994 2.328775 58.54271 0.2770339
```

```
# cat('₩n₩ covariance matrix',cov)
```

z-value calcuation

- in here, I just check whether z-value for constant is correct or not. ref: http://logisticregressionanalysis.com/1577-what-are-z-values-in-logistic-regression/ (http://logisticregressionanalysis.com/1577-what-are-z-values-in-logistic-regression/)

```
z.test.costant =  -3.858994/ 1.7103344
z.test.costant
```

```
## [1] -2.25628
```

p-value calcuation

- ref: https://www.cyclismo.org/tutorial/R/pValues.html (https://www.cyclismo.org/tutorial/R/pValues.html)

```
2*pnorm(-abs(z.test.costant ))
```

```
## [1] 0.02405308
```

Yes, all the coefficient, standard error, and z-score and p-value are correct.

The result of stata and glm r function is similar and

this is difference with the above unrestricted probit regression. Why?

```
. *(5 variables, 106 observations pasted into data editor)

. probit y spread inf gdp

Iteration 0:   log likelihood = -33.121267
Iteration 1:   log likelihood = -21.521847
Iteration 2:   log likelihood = -20.350626
Iteration 3:   log likelihood = -20.335236
Iteration 4:   log likelihood = -20.335235

Probit regression                               Number of obs     =        106
                                                LR chi2(3)        =      25.57
                                                Prob > chi2       =     0.0000
Log likelihood = -20.335235                     Pseudo R2         =     0.3860

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      spread |   2.328515   .6590706     3.53   0.000      1.03676     3.62027
         inf |   58.54633    39.2389     1.49   0.136     -18.3605    135.4532
         gdp |   .2771467   .1674272     1.66   0.098    -.0510046    .6052981
       _cons |  -3.859327   1.710247    -2.26   0.024     -7.21135   -.5073043
------------------------------------------------------------------------------
```

stata result

```
myprobit <- glm(y ~ x -1, family = binomial(link = "probit"))
summary(myprobit)
```

```
## 
## Call:
## glm(formula = y ~ x - 1, family = binomial(link = "probit"))
## 
## Deviance Residuals: 
##      Min        1Q    Median        3Q       Max  
## -1.77023  -0.36705  -0.20615  -0.04034   2.54922  
## 
## Coefficients:
##         Estimate Std. Error z value Pr(>|z|)    
## x        -3.8593     1.6807  -2.296 0.021663 *  
## xspread   2.3285     0.6605   3.525 0.000423 ***
## xinf     58.5456    38.9475   1.503 0.132789    
## xgdp      0.2771     0.1608   1.724 0.084707 .  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 146.95  on 106  degrees of freedom
## Residual deviance:  40.67  on 102  degrees of freedom
## AIC: 48.67
## 
## Number of Fisher Scoring iterations: 7
```

Estimate the restricted probit regression model by MLE

```
theta0 <- qnorm(mean(y),0,1) # quantitle, mean, sd
estResults <- optim(theta0, l0probit, y=y, method="BFGS", hessian=T)

theta <- estResults$par
l0 = estResults$val
h <- estResults$hessian

l0 <- -l0
cat('₩n₩nRestricted log-likelihood function =     ',-l0)
```

```
##
##
## Restricted log-likelihood function =      0.3124648
```

```
cat('₩nT x restricted log-likelihood function = ',-t*l0)
```

```
##
## T x restricted log-likelihood function =  33.12127
```

```
cat('₩n₩nRestricted parameter estimates')
```

```
##
##
## Restricted parameter estimates
```

```
cat('₩n', theta)
```

```
##
##  -1.314496
```

Likelihood ratio test

```
lr <- -2*t*(l0 - l1)

cat('₩n₩nLR Statistic        = ',lr)
```

```
##
##
## LR Statistic        =  25.57206
```

```
cat('₩np-value             = ',1-pchisq(lr,ncol(x)-1))
```

```
##
## p-value              =  1.172204e-05
```

```
cat('₩n')
```

Wald test

```
r <- matrix(c(0,   1,   0,   0,
              0,   0,   1,   0,
              0,   0,   0,   1), byrow=T, nrow=3)

q <- rbind(0,0,0)

wd <- t( (r %*% theta1 - q) ) %*% inv(r %*% cov %*% t(r)) %*% (r %*% theta1 - q)

cat('₩nWald Statistic       = ',wd)
```

```
##
## Wald Statistic       =  15.97866
```

```
cat('₩np-value              = ',(1-pchisq(wd,ncol(x)-1)))
```

```
##
## p-value              =  0.001145466
```

```
cat('₩n')
```

LM test of the joint restrictions

```
u  <- y - mean(y)
b  <- lm(u ~ x - 1)$coef
e  <- u - x %*% b
r2 <- 1 - (t(e) %*% e)/(t(u) %*% u)
lm <- t*r2

cat('₩nRegression estimates = ', b)
```

```
##
## Regression estimates =  -0.1073382 0.2779435 4.021546 0.02957328
```

```
cat('₩nSample size (t)      = ', t )
```

```
##
## Sample size (t)      =  106
```

```
cat('WnR2                   = ', r2 )
```

```
##
## R2                   =  0.2313221
```

```
cat('WnLM Statistic         = ',lm)
```

```
##
## LM Statistic         =  24.52014
```

```
cat('Wnp-value              = ',1-pchisq(lm,ncol(x)-1))
```

```
##
## p-value              =  1.945167e-05
```

```
```