HW3\_*3*.R

USER

Tue Dec 11 00:37:15 2018

library(lmtest)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(forecast)  
rm(list=ls())  
setwd("C:\\Users\\USER\\Documents\\Github\\Econometrics\\시계열 분석\\HW3")  
rv.data<-read.csv("RV\_IV\_data.csv", header=TRUE)  
head(rv.data)

## Date VKOSPI VIX SNP.RV KOSPI.RV  
## 1 2003-01-06 33.04 24.91 0.008186903 0.01243283  
## 2 2003-01-07 33.98 25.13 0.010548387 0.01506374  
## 3 2003-01-08 33.17 25.53 0.011272095 0.01632494  
## 4 2003-01-09 34.02 24.25 0.010671167 0.01859559  
## 5 2003-01-10 38.05 24.32 0.012709980 0.03435511  
## 6 2003-01-13 37.56 24.90 0.011114250 0.01659640

#============================#  
# SNP.RV = Y, VIX = X #  
#============================#  
# MAKE DATA  
y.t = rv.data$SNP.RV[1:nrow(rv.data)] # SNP.RV  
head(y.t); length(y.t)

## [1] 0.008186903 0.010548387 0.011272095 0.010671167 0.012709980 0.011114250

## [1] 3558

x.t = rv.data$VIX[1:(nrow(rv.data))] # VIX  
head(x.t); length(x.t)

## [1] 24.91 25.13 25.53 24.25 24.32 24.90

## [1] 3558

# AR 모형 BIC : p=8  
AR.bic = c()  
for( p in 1:10){  
 AR.fit = arima(y.t, order = c(p,0,0))  
 AR.bic[p] = AIC(AR.fit,k = log(length(y.t)))  
}  
par(mfrow=c(1,2))  
plot(AR.bic, type = "b", pch =19, main = "BIC", ylab = "", xlab = "p")  
abline(v = which.min(AR.bic), col=2, lty = 2) # AR(8)  
  
y\_t = y.t[9:length(y.t)]  
y\_t\_1 = y.t[8:(length(y.t)-1)]  
y\_t\_2 = y.t[7:(length(y.t)-2)]  
y\_t\_3 = y.t[6:(length(y.t)-3)]  
y\_t\_4 = y.t[5:(length(y.t)-4)]  
y\_t\_5 = y.t[4:(length(y.t)-5)]  
y\_t\_6 = y.t[3:(length(y.t)-6)]  
y\_t\_7 = y.t[2:(length(y.t)-7)]  
y\_t\_8 = y.t[1:(length(y.t)-8)]  
  
x\_t = x.t[9:length(x.t)]  
x\_t\_1 = x.t[8:(length(x.t)-1)]  
x\_t\_2 = x.t[7:(length(x.t)-2)]  
x\_t\_3 = x.t[6:(length(x.t)-3)]  
x\_t\_4 = x.t[5:(length(x.t)-4)]  
x\_t\_5 = x.t[4:(length(x.t)-5)]  
x\_t\_6 = x.t[3:(length(x.t)-6)]  
x\_t\_7 = x.t[2:(length(x.t)-7)]  
x\_t\_8 = x.t[1:(length(x.t)-8)]  
  
y.data = cbind(y\_t\_1 = y\_t\_1, y\_t\_2= y\_t\_2, y\_t\_3 = y\_t\_3, y\_t\_4 = y\_t\_4,  
 y\_t\_5 = y\_t\_5, y\_t\_6= y\_t\_6, y\_t\_7 = y\_t\_7, y\_t\_8 = y\_t\_8)  
x.data = cbind(x\_t\_1 = x\_t\_1, x\_t\_2= x\_t\_2, x\_t\_3 = x\_t\_3, x\_t\_4 = x\_t\_4,  
 x\_t\_5 = x\_t\_5, x\_t\_6= x\_t\_6, x\_t\_7 = x\_t\_7, x\_t\_8 = x\_t\_8)  
  
# ADL 모형 BIC : p=6, q=6  
ADL.bic = matrix(0, ncol = 8, nrow = 8)  
for( p in 1:8){  
 for(q in 1:8){   
 ADL.fit = lm(y\_t ~ y.data[,1:p] + x.data[,1:q])   
 ADL.bic[p,q] = AIC(ADL.fit,k = log(length(y.t)))  
 }  
}  
colnames(ADL.bic) = c("q=1", "q=2", "q=3", "q=4", "q=5", "q=6", "q=7", "q=8")  
row.names(ADL.bic) = c("p=1", "p=2", "p=3", "p=4", "p=5", "p=6", "p=7", "p=8")  
ADL.bic; min(ADL.bic) # ADL(6,6)

## q=1 q=2 q=3 q=4 q=5 q=6 q=7  
## p=1 -30730.66 -30764.09 -30790.13 -30792.10 -30806.08 -30823.50 -30825.73  
## p=2 -30836.12 -30943.35 -30972.21 -30966.69 -30964.98 -30972.28 -30966.61  
## p=3 -30827.97 -30946.21 -31002.40 -30998.35 -30993.98 -30996.00 -30989.24  
## p=4 -30822.93 -30942.19 -31013.12 -31021.53 -31018.83 -31017.89 -31009.99  
## p=5 -30814.76 -30935.23 -31006.56 -31019.42 -31024.30 -31024.98 -31016.84  
## p=6 -30809.18 -30933.90 -31010.01 -31023.47 -31037.82 -31057.77 -31050.08  
## p=7 -30803.53 -30926.69 -31001.90 -31015.31 -31029.70 -31052.63 -31046.82  
## p=8 -30795.44 -30920.92 -30997.84 -31012.95 -31029.58 -31053.31 -31051.16  
## q=8  
## p=1 -30836.17  
## p=2 -30967.10  
## p=3 -30986.51  
## p=4 -31005.71  
## p=5 -31010.98  
## p=6 -31042.91  
## p=7 -31040.05  
## p=8 -31051.37

## [1] -31057.77

## 1-step ahead forecasting  
AR8.fore = c(); ADL66.fore = c()  
for(i in 3106:(length(y.t)-1)){  
 train.data = data.frame(y.t = y.t[1:i], x.t = x.t[1:i])  
   
 y\_t = train.data[9:nrow(train.data),1]  
 y\_t\_1 = train.data[8:(nrow(train.data)-1),1]  
 y\_t\_2 = train.data[7:(nrow(train.data)-2),1]  
 y\_t\_3 = train.data[6:(nrow(train.data)-3),1]  
 y\_t\_4 = train.data[5:(nrow(train.data)-4),1]  
 y\_t\_5 = train.data[4:(nrow(train.data)-5),1]  
 y\_t\_6 = train.data[3:(nrow(train.data)-6),1]  
 y\_t\_7 = train.data[2:(nrow(train.data)-7),1]  
 y\_t\_8 = train.data[1:(nrow(train.data)-8),1]  
   
 x\_t = train.data[9:nrow(train.data),2]  
 x\_t\_1 = train.data[8:(nrow(train.data)-1),2]  
 x\_t\_2 = train.data[7:(nrow(train.data)-2),2]  
 x\_t\_3 = train.data[6:(nrow(train.data)-3),2]  
 x\_t\_4 = train.data[5:(nrow(train.data)-4),2]  
 x\_t\_5 = train.data[4:(nrow(train.data)-5),2]  
 x\_t\_6 = train.data[3:(nrow(train.data)-6),2]  
 x\_t\_7 = train.data[2:(nrow(train.data)-7),2]  
 x\_t\_8 = train.data[1:(nrow(train.data)-8),2]  
   
 # AR(8)  
 AR8.fit = lm(y\_t ~ y\_t\_1 + y\_t\_2 + y\_t\_3 + y\_t\_4 +   
 y\_t\_5 + y\_t\_6 + y\_t\_7 + y\_t\_8)  
 AR8.fore[i-3105] = sum(AR8.fit$coef\*c(1, y\_t[length(y\_t)],   
 y\_t\_1[length(y\_t)],   
 y\_t\_2[length(y\_t)],   
 y\_t\_3[length(y\_t)],  
 y\_t\_4[length(y\_t)],  
 y\_t\_5[length(y\_t)],  
 y\_t\_6[length(y\_t)],  
 y\_t\_7[length(y\_t)]))   
 # ADL(6,6)  
 ADL.fit1 = lm(y\_t ~   
 y\_t\_1 + y\_t\_2 +y\_t\_3 + y\_t\_4 + y\_t\_5 + y\_t\_6 +   
 x\_t + x\_t\_1 + x\_t\_2 + x\_t\_3+ x\_t\_4 + x\_t\_5 );  
 ADL66.fore[i-3105] = sum(ADL.fit1$coef\*c(1, y\_t[length(y\_t)],   
 y\_t\_1[length(y\_t)],   
 y\_t\_2[length(y\_t)],   
 y\_t\_3[length(y\_t)],  
 y\_t\_4[length(y\_t)],  
 y\_t\_5[length(y\_t)],  
 x\_t[length(y\_t)],   
 x\_t\_1[length(y\_t)],   
 x\_t\_2[length(y\_t)],   
 x\_t\_3[length(y\_t)],  
 x\_t\_4[length(y\_t)],  
 x\_t\_5[length(y\_t)] ))  
   
}  
  
# 1. VKOSPI가 KOSPI 5분 실현변동성을 GRANGER CAUSE하는가  
grangertest(y.t~x.t)

## Granger causality test  
##   
## Model 1: y.t ~ Lags(y.t, 1:1) + Lags(x.t, 1:1)  
## Model 2: y.t ~ Lags(y.t, 1:1)  
## Res.Df Df F Pr(>F)   
## 1 3554   
## 2 3555 -1 1000.6 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

# 2. 예측력 비교  
# AR(8) VS ADL(6,6)   
# AR8.fore VS ADL66.fore  
# MAE  
AR8.MAE = mean(abs(AR8.fore - y.t[3107:length(y.t)]))  
ADL66.MAE = mean(abs(ADL66.fore - y.t[3107:length(y.t)]))  
AR8.MAE < ADL66.MAE # AR(8) 모형이 MAE 값이 더 작음

## [1] TRUE

# MSE  
AR8.MSE = mean((AR8.fore - y.t[3107:length(y.t)])^2)  
ADL66.MSE = mean((ADL66.fore - y.t[3107:length(y.t)])^2)  
AR8.MSE < ADL66.MSE # AR(8) 모형이 MSE 값이 더 작음

## [1] TRUE

result = matrix(c(AR8.MAE, ADL66.MAE, AR8.MSE, ADL66.MSE), nrow = 2, byrow = T)  
row.names(result) = c("MAE", "MSE")  
colnames(result) = c("AR8", "ADL(6,6)")  
result # AR(8) 모형이 더 잘 예측

## AR8 ADL(6,6)  
## MAE 1.293314e-03 1.357345e-03  
## MSE 3.976575e-06 4.643699e-06

