FA_Assignment9_Duo_Zhou

Duo Zhou

8/18/2020

High Frequency Trading

Fit decomposition model to hft2_trades_train.csv

```
datapath <- "C:/Users/zd000/Desktop/MSCA/Financial Analytics/Assignments/week9/"
da=read.csv(paste(datapath,"hft2_trades_train.csv",sep="/"),header=T)
head(da)
```

```
##      timestamp  price size side  vix
## 1 1.441892e+15 193775    4    A 2660
## 2 1.441892e+15 193775   44    B 2660
## 3 1.441892e+15 193850    1    B 2655
## 4 1.441892e+15 193775    5    A 2655
## 5 1.441892e+15 193775   24    B 2655
## 6 1.441892e+15 193800    2    A 2655
```

Calculate pch, the changes of the futures price in # of ticks. Calculate S, the absolute change of the futures price in # of ticks.

```
tick <- 25
da$pch <- c(0, da$price[2:nrow(da)] - da$price[2:nrow(da) - 1] ) / tick
da$S <- abs(da$pch)
head(da[,c('pch','S')],10)
```

```
##    pch S
## 1    0 0
## 2    0 0
## 3    3 3
## 4   -3 3
## 5    0 0
## 6    1 1
## 7    0 0
## 8    0 0
## 9    0 0
## 10   1 1
```

Create the components of decomposition A, D and S.

```r
nl<-nrow(da)
idx=c(1:nl)[da$pch > 0]
jdx=c(1:nl)[da$pch < 0]
A=rep(0,nl); A[idx]=1; A[jdx]=1
D=rep(0,nl); D[idx]=1; D[jdx]=-1
S=da$S
head(cbind(A,D,S))
```

```
##      A  D S
## [1,] 0  0 0
## [2,] 0  0 0
## [3,] 1  1 3
## [4,] 1 -1 3
## [5,] 0  0 0
## [6,] 1  1 1
```

Create lagged variables

```r
Ai=A[2:nl]; Aim1=A[1:nl-1]
Di=D[2:nl]; Dim1=D[1:nl-1]
Si=S[2:nl]; Sim1=S[1:nl-1]
```

Fit logistic regression models to the components. A model

```r
m1=glm(Ai~Aim1,family="binomial")
summary(m1)
```

```
##
## Call:
## glm(formula = Ai ~ Aim1, family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3995  -1.3315   0.9705   0.9705   1.0307
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.35527    0.02330  15.245  < 2e-16 ***
## Aim1         0.15300    0.03003   5.095 3.48e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 26069  on 19492  degrees of freedom
## Residual deviance: 26043  on 19491  degrees of freedom
## AIC: 26047
##
## Number of Fisher Scoring iterations: 4
```

D model

```r
di=Di[Ai==1]
dim1=Dim1[Ai==1]
di=(di+abs(di))/2 # transform di to binary

m2=glm(di~dim1,family="binomial")
summary(m2)
```

```
##
## Call:
## glm(formula = di ~ dim1, family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6144  -1.1776   0.7963   1.1773   1.6141
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.000328   0.019663   0.017    0.987
## dim1        -0.985770   0.026086 -37.789   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 16489  on 11893  degrees of freedom
## Residual deviance: 14882  on 11892  degrees of freedom
## AIC: 14886
##
## Number of Fisher Scoring iterations: 4
```

S uptick model

```r
si=Si[Di==1]
sim1=Sim1[Di==1]
source(paste(datapath,"GeoSize.R",sep="/")) # R script to fit Geometric dist.
m3=GeoSize(si,sim1)
```

```
## [1] 3457.146
##   0:     3457.1459: -0.0929898  1.00000
##   3:     2244.2642: 0.929898  1.18565
##   6:     2241.7742: 0.929898  1.30063
## Estimates:  0.9298978 1.300627
##
## Coefficient(s):
##         Estimate  Std. Error  t value   Pr(>|t|)
## omega1 0.9298978   0.0442051  21.0360 < 2.22e-16 ***
## omega2 1.3006272   0.0670093  19.4096 < 2.22e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

S downtick model

```r
nsi=Si[Di==-1]
nsim1=Sim1[Di==-1]
m4=GeoSize(nsi,nsim1)
```

```
## [1] 3482.535
##   0:     3482.5349: -0.0912869  1.00000
##   3:     2290.9502: 0.912869  1.17864
##   6:     2290.9496: 0.912869  1.17696
## Estimates:  0.9128689 1.176964
##
## Coefficient(s):
##         Estimate  Std. Error  t value   Pr(>|t|)
## omega1 0.9128689   0.0440407  20.7279 < 2.22e-16 ***
## omega2 1.1769644   0.0642678  18.3134 < 2.22e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From model for Ai

```r
(beta_0 <- unname(m1$coefficients[1]))
```

```
## [1] 0.3552691
```

```r
(beta_1 <- unname(m1$coefficients[2]))
```

```
## [1] 0.1530012
```

```r
plogis(beta_0)             # prob of Ai = 1 when Aim1 = 0
```

```
## [1] 0.5878947
```

```r
plogis(beta_0 + 1*beta_1)  # prob of Ai = 1 when Aim1 = 1
```

```
## [1] 0.6244009
```

From model for Di

```r
(gamma_0 <- unname(m2$coefficients[1]))
```

```
## [1] 0.0003279979
```

```r
(gamma_1 <- unname(m2$coefficients[2]))
```

```
## [1] -0.9857697
```

```r
plogis(gamma_0 - 1*gamma_1)  # prob of Di = 1 when Ai = 1, Dim1 = -1
```

```
## [1] 0.7283165
```

```r
plogis(gamma_0)              # prob of Di = 1 when Ai = 1, Dim1 =  0
```

```
## [1] 0.500082
```

```r
plogis(gamma_0 + 1*gamma_1)  # prob of Di = 1 when Ai = 1, Dim1 = +1
```

```
## [1] 0.2718134
```

Parameters of the S uptick model are:

```r
(theta_u0 <- unname(m3$par[1]))
```

```
## [1] 0.9298978
```

```r
(theta_u1 <- unname(m3$par[2]))
```

```
## [1] 1.300627
```

Parameters of the S downtick model are:

```r
(theta_d0 <- unname(m4$par[1]))
```

```
## [1] 0.9128689
```

```r
(theta_d1 <- unname(m4$par[2]))
```

```
## [1] 1.176964
```

Finally, probability: $P(pch <= x) = P(A_i DiSi) = P(Si|AiDi)P(Di|Ai)P(Ai)$ This probability is calculated by the following function.

```r
# Pr( next_pch <= x | aim1, dim1, sim1 )
pch_decomposition_cdf <- function(x, aim1, dim1, sim1, decomp_params) {
    pch_cdf <- 0
    p <- plogis(decomp_params$beta_0 + decomp_params$beta_1 * aim1)    # Pr( Ai =  1 | aim1 )
    q <- plogis(decomp_params$gamma_0 + decomp_params$gamma_1 * dim1)  # Pr( Di = +1 | dim1 )

    lambda_up = plogis(decomp_params$theta_u0 + decomp_params$theta_u1 * sim1)
    lambda_down = plogis(decomp_params$theta_d0 + decomp_params$theta_d1 * sim1)

    if (x < 0) {
        # P( next_pch <= x ) = Pr( Ai = 1, Di = -1, Si >= -x ) = Pr( Ai = 1, Di = -1, Si > -x-1 )
        # since Si ~ 1 + geom(lambda_down) when Di = -1 we have:
        pch_cdf <- p * (1-q) * pgeom(-x-2, prob=lambda_down, lower.tail = FALSE)
    } else if (x >= 0) {
        # P( next_pch <= x ) = Pr( Ai = 0 ) + Pr( Ai = 1, Di = 1 ) + Pr( Ai = 1, Di = -1, Si <= x ) =
        # = (1-p) + p*(1-q) + Pr( Ai = 1, Di = 1, Si <= x )
        # since Si ~ 1 + geom(lambda_up) when Di = 1 we have:
        pch_cdf <- (1-p) + p * (1-q) + p * q * pgeom(x-1, prob=lambda_up)
    }

    return(pch_cdf)
}
```

```r
decomp_params <- list(beta_0 = beta_0, beta_1 = beta_1,
                      gamma_0 = gamma_0, gamma_1 = gamma_1,
                      theta_u0 = theta_u0, theta_u1 = theta_u1,
                      theta_d0 = theta_d0, theta_d1 = theta_d1)
```

What is probability that price change of the next trade will be category -1 tick if current price change is not zero (A=1), negative (D=-1), and S changed by 2 ticks?

```r
(decomp_cross_prob <- pch_decomposition_cdf(-1, aim1=1, dim1=-1, sim1=2, decomp_params))
```

```
## [1] 0.1696395
```