ABF\_Week4\_HW

taehwan.lee

2024-01-24

library(quantmod)

## Warning: 패키지 'quantmod'는 R 버전 4.3.2에서 작성되었습니다

## 필요한 패키지를 로딩중입니다: xts

## Warning: 패키지 'xts'는 R 버전 4.3.2에서 작성되었습니다

## 필요한 패키지를 로딩중입니다: zoo

## Warning: 패키지 'zoo'는 R 버전 4.3.2에서 작성되었습니다

##   
## 다음의 패키지를 부착합니다: 'zoo'

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

## 필요한 패키지를 로딩중입니다: TTR

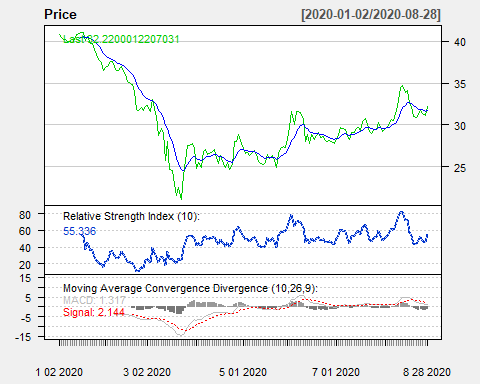
## Warning: 패키지 'TTR'는 R 버전 4.3.2에서 작성되었습니다

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

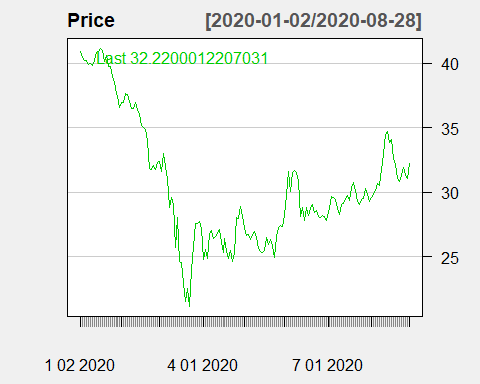
getSymbols("KB", from = "2020-01-01", to = "2020-08-31")

## [1] "KB"

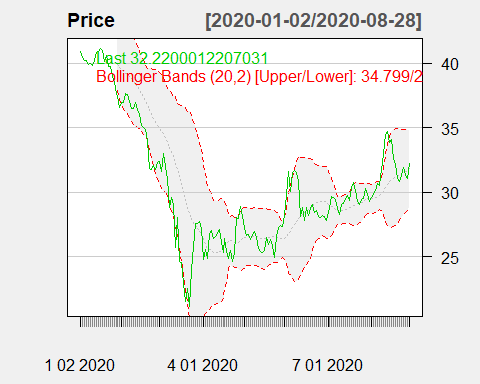
Price = Ad(KB)  
chartSeries(Price, theme = chartTheme("white"), TA = "addRSI(10) ; addEMA(10); addMACD(10)")



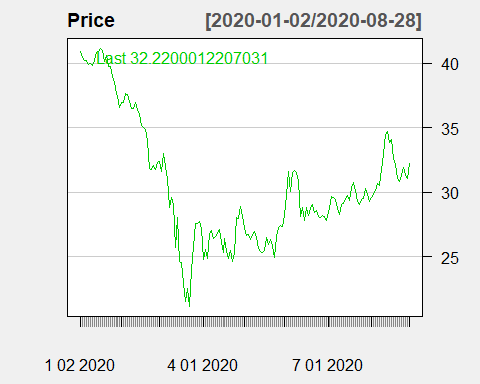
chartSeries(Price, theme = chartTheme("white"))



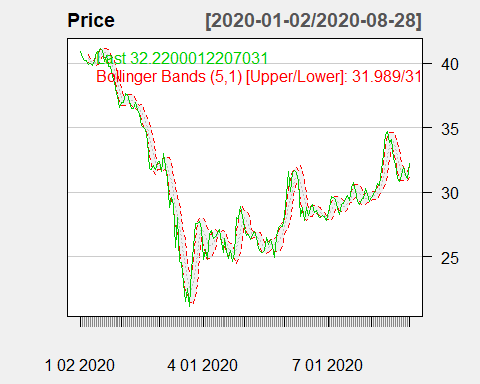
addBBands()



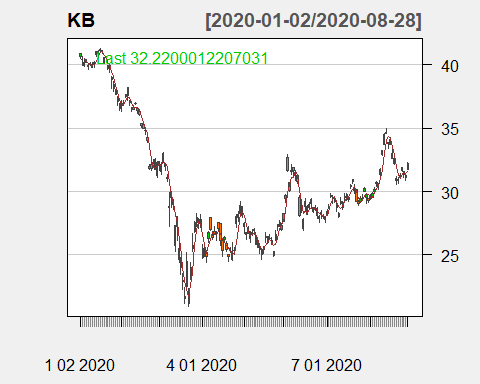
chartSeries(Price, theme = chartTheme("white"))



addBBands(5,1)



chartSeries(KB, theme = chartTheme("white"), TA = "addSMA(3)")



getSymbols("KB", from = "2020-01-01", to = "2020-12-31")

## [1] "KB"

Price<-Cl(KB)  
HLC<-matrix(c(Hi(KB), Lo(KB), Cl(KB)), nrow = length(Hi(KB)))  
library(TTR)  
r<- diff(log(Price))  
rsi<-RSI(Price)  
MACD<-MACD(Price)  
macd<-MACD[,1]  
will<-williamsAD(HLC)  
cci <- CCI(HLC)  
STOCH <- stoch(HLC)  
stochK <- STOCH[, 1]  
stochD <- STOCH[, 2]  
  
input <- matrix(c(as.numeric(rsi[26:242]), cci[26:242], as.numeric(macd[26:242]), will[26:242], stochK[26:242], stochD[26:242]), nrow = 217)  
Target <- matrix(c(r[27:243]), nrow = 217)  
trainingdata <- cbind(input, Target)  
colnames(trainingdata) <- c("RSI", "CCI", "MACD", "WILL", "STOCHK", "STOCHD", "Return")  
  
library(caret)

## Warning: 패키지 'caret'는 R 버전 4.3.2에서 작성되었습니다

## 필요한 패키지를 로딩중입니다: ggplot2

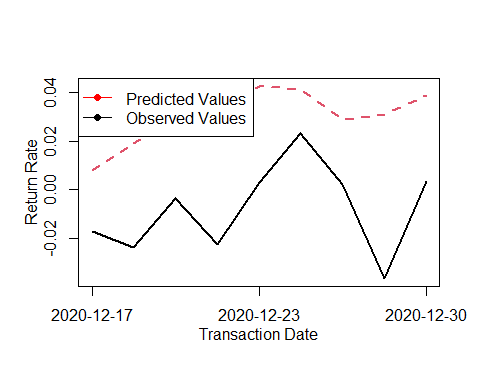
## Warning: 패키지 'ggplot2'는 R 버전 4.3.2에서 작성되었습니다

## 필요한 패키지를 로딩중입니다: lattice

trainIndex <- createDataPartition(1:217, p=.8, list = F)  
Train <- trainingdata[trainIndex,]  
Valid <- trainingdata[-trainIndex, ]  
  
library(nnet)  
best.network <- matrix(0,2)  
best.mae <- 1  
  
for (i in seq(10, 100, 5)) for(j in c(0.01, 0.001, 0.0001)){  
 Fit = nnet(Return ~ RSI + CCI + MACD + WILL + STOCHK +STOCHD, data = Train, maxit = 1000, size = i, decay = j,  
 trace = FALSE, linout = TRUE)  
 Predict = predict(Fit, newdata = Valid)  
 mae1 = mean(abs(Predict - Valid[,7]))  
 if (mae1 < best.mae){  
 best.network[1] <- i  
 best.network[2] <- j  
 best.mae = mae1  
 }  
}

InputTest <- matrix(c(as.numeric(rsi[243:251]), cci[243:251], as.numeric(macd[243:251]), will[243:251], stochK[243:251], stochD[243:251]), nrow = 9)  
TargetTest <- matrix(c(r[244:252]), nrow = 9)  
Testdata = cbind(InputTest, TargetTest)  
colnames(Testdata) <- c("RSI", "CCI", "MACD", "WILL", "STOCHK", "STOCHD", "Return")  
  
Fit = nnet(Return ~ RSI + CCI + MACD + WILL + STOCHK +STOCHD, data = Train, maxit = 3000, size = best.network[1], decay = best.network[2],trace = FALSE, linout = TRUE)  
  
Predict1 <- predict(Fit, newdata = Testdata)  
#repeat the best model 50 times  
for(i in 1:50){  
 Fit = nnet(Return ~ RSI + CCI + MACD + WILL + STOCHK +STOCHD, data = Train, maxit = 3000, size = best.network[1],  
 decay = best.network[2],trace = FALSE, linout = TRUE)  
 Predict1 <- Predict1 + predict(Fit, newdata = Testdata)  
}  
  
Predict1 <- Predict1/51

matplot(cbind(Testdata[,7], Predict1), type = "l", lwd = 2, xaxt = "n", ylab = "")  
legend("topleft", legend = c("Predicted Values", "Observed Values"), lwd = 1.5, pch = 19, col = c("red", "black"))  
axis(1, at = c(1,5,9), lab = c("2020-12-17", "2020-12-23", "2020-12-30"))  
mtext(side = 1, "Transaction Date", line = 2)  
mtext(side = 2, "Return Rate", line = 2)



library("devtools")

## Warning: 패키지 'devtools'는 R 버전 4.3.2에서 작성되었습니다

## 필요한 패키지를 로딩중입니다: usethis

## Warning: 패키지 'usethis'는 R 버전 4.3.2에서 작성되었습니다

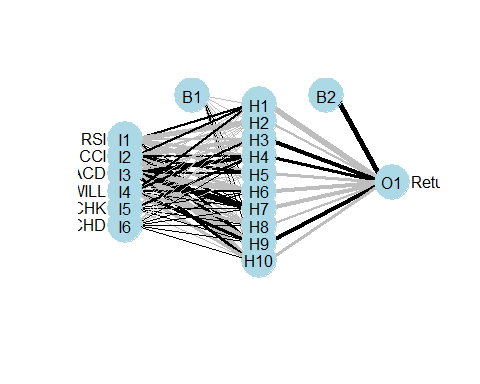
library("reshape2")

## Warning: 패키지 'reshape2'는 R 버전 4.3.2에서 작성되었습니다

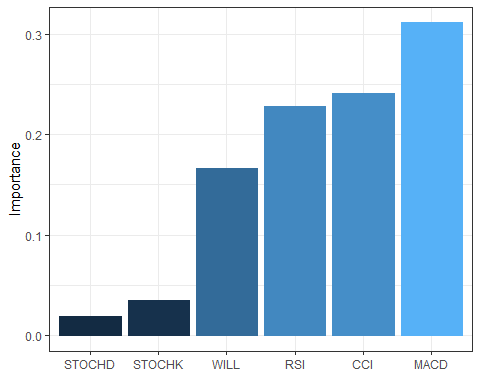
library("NeuralNetTools")

## Warning: 패키지 'NeuralNetTools'는 R 버전 4.3.2에서 작성되었습니다

plotnet(Fit)



garson(Fit)

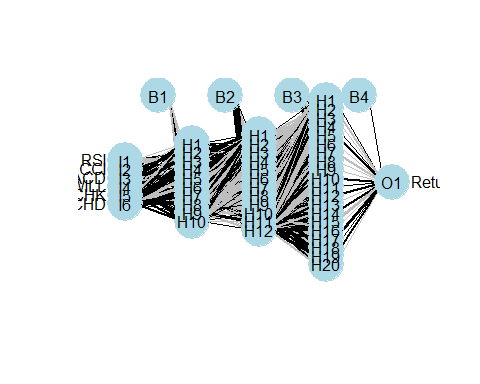


library(neuralnet)

## Warning: 패키지 'neuralnet'는 R 버전 4.3.2에서 작성되었습니다

set.seed(10000)  
Formula <- Return ~ RSI + CCI + MACD + WILL + STOCHK + STOCHD  
DNN1 <- neuralnet(Formula, data = trainingdata, hidden = c(10,12,20), algorithm = "rprop+", err.fct ="sse",  
 act.fct="tanh", threshold = 0.01, linear.output = TRUE)  
  
plot(DNN1)

plotnet(DNN1)



DNN1 <- neuralnet(Formula, data = trainingdata, hidden = c(10, 12, 20), rep = 10, algorithm = "rprop-", err.fct = "sse", act.fct = "logistic", threshold = 0.001, linear.output = TRUE)

library(TTR)  
MinMax <- function(x) {  
 return((x - min(x, na.rm = TRUE)) / (max(x, na.rm = TRUE) - min(x, na.rm = TRUE)))  
}  
r <- as.numeric(diff(log(Price)))  
print(head(r))

## [1] NA -0.0108241802 -0.0052076765 -0.0007462404 -0.0064902456  
## [6] 0.0015015361

r = MinMax(r)  
print(head(r))

## [1] NA 0.4655721 0.4874347 0.5048012 0.4824422 0.5135508

rsi <- as.numeric(diff(log(Price)))  
print(head(rsi))

## [1] NA -0.0108241802 -0.0052076765 -0.0007462404 -0.0064902456  
## [6] 0.0015015361

rsi = MinMax(rsi)  
print(head(rsi))

## [1] NA 0.4655721 0.4874347 0.5048012 0.4824422 0.5135508

MACD <- MACD(Price)  
macd <- as.numeric(MACD[, 1]); macd <- MinMax(macd)  
will <- as.numeric(williamsAD(HLC)); will <- MinMax(will)  
cci <- as.numeric(CCI(HLC)); cci <- MinMax(cci)  
STOCH <- stoch(HLC)  
stochK <- as.numeric(STOCH[,1]); stochK <- MinMax(stochK)  
stochD <- as.numeric(STOCH[,2]); stochD <- MinMax(stochD)