Easy Volatility Investing, Replication

matReturns.com

```r
require(quantmod)
require(PerformanceAnalytics)

  ### Strategy 1: Buy and Hold SVXY
  # Function to calculate Buy and Hold returns for SVXY
  ezVol_buy_and_hold <- function(startDate = '2018-09-01', endDate = '2023-12-31') {

    tickers <- c("SVXY")# ProShares Short VIX Short-Term Futures ETF

    # Store log returns data calculated from adjusted prices
      price <- Ad(get(getSymbols("SVXY", from = startDate, to = endDate)))
      colnames(price) <- "SVXY"

    # Calculate log returns for SVXY
    svxyRets <- na.omit(Return.calculate(price, method = "log"))
    colnames(svxyRets)<- "SVXY.B/H"
    # Return SVXY returns data
    return(svxyRets)
  }

  # Call the function with specified date range
  startDate <- '2018-09-01'
  endDate <- '2023-12-31'
  svxyReturns <- ezVol_buy_and_hold(startDate, endDate)
```

```r
### Strategy 2: Momentum
# Function to calculate Momentum strategy returns
# Arguments:
# startDate: Start date for fetching data (default: '2018-09-01')
# endDate: End date for fetching data (default: '2023-12-31')
# k: Number of days for momentum calculation (default: 83)
ezVol_momentum_strategy_returns <- function(startDate = '2018-09-01',
                                            endDate = '2023-12-31',
                                            k = 83) {

  # 2 volatility ETFs
  tickers <- c("SVXY",   # ProShares Short VIX Short-Term Futures ETF
               "VIXY"    # ProShares VIX Short-Term Futures ETF
              )
  # Store Log returns data caluclated from adjusted prices to a list
  returns <- list()
  for(i in 1:length(tickers)) {
    rets <- Return.calculate(Ad(get(getSymbols(tickers[i],
                                               from = startDate,
                                               to = endDate,
                                               method = "log"))))
    colnames(rets) <- tickers[i]
    returns[[i]] <- rets
  }
  # cbind rets
  returns <- na.omit(do.call(cbind, returns))

  sigs <- list()
  # Calculate momentum signals for each day based on k-day returns
  for (i in 1:(nrow(returns) - k + 1)) {
    svxySig <- as.numeric(Return.cumulative(returns$SVXY[i:(i + k - 1), ]))
    vixySig <- as.numeric(Return.cumulative(returns$VIXY[i:(i + k - 1), ]))
    retSig <- svxySig > vixySig
    sigs[[i]] <- retSig
  }
  sigs <- do.call(rbind, sigs)
  sigDF <- cbind(returns[k:nrow(returns), ], sigs)
  sigDF$sigs <- lag(sigDF$sigs, n = 1)
  sigDF <- na.omit(sigDF)


# Calculate returns using the contango-backwardation roll yield strategy
  momReturns <- sigDF$sigs * sigDF$SVXY + (1 - sigDF$sigs) * sigDF$VIXY
  colnames(momReturns) <- "Momentum"
  return(momReturns)
}
# Call the function with specified date range and k-day value (default k = 83)
startDate <- '2018-09-01'
endDate <- '2023-12-31'
k <- 83
momReturns <- ezVol_momentum_strategy_returns(startDate, endDate, k)
```

```r
  ### Strategy 3: Contango-Backwardation Roll Yield
  # SVXY if vix3m > vix and VIXY otherwise
    # Define the function
ezVol_contango_backwardation_rollYield_returns <- function(startDate = '2018-09-01',
                                                           endDate = '2023-12-31',
                                                           SMA.n = 10) {

  # 3 volatility ETFs and the 2 indices
  tickers <- c("SVXY", # ProShares Short VIX Short-Term Futures ETF
               "VIXY", # ProShares VIX Short-Term Futures ETF
               "^VIX", # CBOE Volatility Index
               "^VIX3M" # CBOE S&P 500 3-Month Volatility
  )

  # Store log returns data calculated from adjusted prices to a list
  prices <- list()
  for (i in 1:length(tickers)) {
    price <- suppressWarnings(Ad(get(getSymbols(tickers[i],
                                                from = startDate,
                                                to = endDate))))
    colnames(price) <- tickers[i]
    prices[[i]] <- price
  }

  # Combine and remove NAs
  prices <- na.omit(do.call(cbind, prices))
  colnames(prices) <- tickers
  svxyRets <- Return.calculate(prices$SVXY)
  vixyRets <- Return.calculate(prices$VIXY)


  # Create the signal using SMA with n = 10
  vols <- merge(prices$`^VIX3M`, prices$`^VIX`, join='inner')
  vols$smaDiff <- SMA(vols[,1] - vols[,2], n = SMA.n)
  vols$signal <- vols$smaDiff > 0

  # Create the dataframe
  cbrDF <- cbind(svxyRets, vixyRets, lag(vols$signal[2:nrow(prices)], n = 1))
  cbrDF <- na.omit(cbrDF)

  # Calculate returns using the contango-backwardation roll yield strategy
  cbrReturns <- cbrDF$signal * cbrDF$SVXY + (1 - cbrDF$signal) * cbrDF$VIXY
  colnames(cbrReturns) <- "CBR Returns"

  return(cbrReturns)
}
# Call the function with specified date range
startDate <- '2018-09-01'
endDate <- '2023-12-31'
SMA.n <- 10
cbrReturns <- ezVol_contango_backwardation_rollYield_returns(startDate,
                                                             endDate,
                                                             SMA.n =  10)
```

```r
### Strategy 4: Volatility Risk Premium
# HVOL10S
# Historical Vol
# Define the function for the Volatility Risk Premium strategy
ezVol_volatility_risk_premium_returns <- function(startDate = '2018-09-01',
                                                  endDate = '2023-12-31',
                                                  sd.n = 10,
                                                  sma_length = 5) {
  tickers <- c("SVXY",  # ProShares Short VIX Short-Term Futures ETF
               "VIXY",  # ProShares VIX Short-Term Futures ETF
               "^GSPC", # S&P 500 Index
               "^VIX")  # CBOE Volatility Index
  # Store log returns data calculated from adjusted prices to a list
  prices <- list()
  for (i in 1:length(tickers)) {
    price <- suppressWarnings(Cl(get(getSymbols(tickers[i],
                                                from = startDate,
                                                to = endDate))))
    colnames(price) <- tickers[i]
    prices[[i]] <- price}
  # Combine and remove NAs
  prices <- na.omit(do.call(cbind, prices))
  colnames(prices) <- tickers
  svxyRets <- Return.calculate(prices$SVXY, method = "log")
  vixyRets <- Return.calculate(prices$VIXY, method = "log")

  # Calculate Historical Volatility and Standard Deviation
  spyRets <- Return.calculate(prices$`^GSPC`, method = "log")
  spyVol <- runSD(spyRets, n = sd.n)
  annSpyVol <- spyVol * 100 * sqrt(252)

  # Create the signal using SMA with the specified length
  smaDiff <- SMA(prices$`^VIX` - annSpyVol, n = sma_length)
  signal <- smaDiff > 0
  signal <- as.numeric(signal)

  # Create Date Frame and lag signal
  vrpDF <- cbind(svxyRets, vixyRets, signal)
  vrpDF$signal <- lag(vrpDF$signal, n = 1)
  vrpDF <- na.omit(vrpDF)

  # Calc Returns
  vrpReturns <- vrpDF$signal * vrpDF$SVXY + (1 - vrpDF$signal) * vrpDF$VIXY
  colnames(vrpReturns) <- "VRP Returns"
  return(vrpReturns)
}
startDate <- '2018-09-01'
endDate <- '2023-12-31'
sd.n <- 10
sma_length <- 5
vrpReturns <- ezVol_volatility_risk_premium_returns(startDate, endDate, sd.n, sma_length)
```

Thanks for clicking by! Please visit matReturns.com for more.

Note: The code and data provided in this analysis are for illustrative purposes only and do not constitute financial advice. Investors should conduct thorough due diligence and consult with financial professionals before making any investment decisions.