# Assignment 1

*Philipp Epstein*

*10/14/2018*

## Introduction

Blabla Blabla

## Trading Idea

Explain here the traiding idea, where I found it. What the theoretical foundation is –> Long Term positive autocorreleation –> Short term negative autocorreleation

Trade reverse patterns. When observing a fallback, trade to profit from the upward trend after the fallback. etc. . . .

## Implementation of the trading idea

Short paragraph about what is needed for the implementation

### Part A: Initialization

#### Step 1: General Setup

We have to clear the environment and initialize the most important packages that we will use.

```
# Clear Environment
rm(list=ls())

# Loading libraries
library(blotter)
library(INFT361Course)
```

#### Step 2: Setting the Variables

The variables set in the next section can be adjusted to test the strategy with different parameters.

```
# Set values:
startCapital <- 1e+6
transactionCost <- -20
daterange <- '2013::2018'
emaPeriod <- 200
maxHoldingPeriod <- 7

InstrumentDirectory <- "~/Desktop/R/DownloadedData/"
instrumentlist <- c("SAP.csv","DBK.csv")
BuyHoldDirectory <- InstrumentDirectory
BuyHoldInstrument <- "DAXEX.csv"

currency("EUR")
Sys.setenv(TZ="UTC")
initdate <- '1999-12-31'
```

```
startdate <- '2000-01-01'
enddate <- '2018-12-31'
portfolioname <- "Smash Day"
accountname <- portfolioname
```

### Step 3: Presetup for plotting graphs

```
# Settings for graph
myTheme <- chart_theme()
myTheme$col$up.col <- 'lightblue'
myTheme$col$dn.col <- 'brown'
myTheme$col$dn.border <- 'lightgray'
myTheme$col$up.border <- 'lightgray'

# Concatenate string for EMA with input parameter
addEMAString <- paste("add_EMA(n=",emaPeriod,")",sep = "")
```

### Step 3: Initializing the portfolio

The portfolio takes the instrumentlist which includes all the stocks we defined in Step 2.

```
# Clear portfolio and Account
suppressWarnings(rm("account.Smash Day","portfolio.Smash Day","account.buyhold","portfolio.buyhold",pos

# Initialize Portfolio and Account
initPortf(portfolioname,instrumentlist,initDate=initdate,currency="EUR")
initAcct(accountname,portfolios=portfolioname,initDate=initdate,initEq=startCapital,currency="EUR")
```

## Part B: Bar by bar processing

### Step 1: Go through the data bar by bar

Loading the instrument, initializing it and adding the ema to the data. After that we go through the available data bar by bar and apply the buy and sell rules which are defined in the following part.

```
# Go through the instrumentlist and perform the activities for all instruments in that list.
for (instrument in instrumentlist) {
    LoadCourseFile(InstrumentDirectory, instrument, debugme = TRUE, dates = daterange)

    # Initialize the instrument
    stock(instrument, currency = "EUR")

    # Load the XTS file
    symbol <- get(instrument)

    # Calculate the Exponential Moving Average
    ema <- EMA(symbol$Close, n=emaPeriod)

    # Merge the xts file with the Exponential Moving Average
    symbol <- merge(symbol,ema)
    assign(instrument,symbol)
  # Starting to go bar by bar through using a "for loop"
  for (i in (emaPeriod + 1):(nrow(symbol) - 1)) {
    # Dates
    CurrentDate <- time(symbol[i])
```

```
  TomorrowDate <- time(symbol[i + 1])

  # Today's variables
  CloseToday <- as.numeric(symbol[i, "Close"])
  EMA_today <- as.numeric(symbol[i, "EMA"])
  LowToday <- as.numeric(symbol[i, "Low"])
  HighToday <- as.numeric(symbol[i, "High"])

  # Yesterday's variables
  LowYesterday <- as.numeric(symbol[i - 1, "Low"])
  HighYesterday <- as.numeric(symbol[i - 1, "High"])

  # Tomorrow's variables
  OpenTomorrow <- as.numeric(symbol[i + 1, "Open"])
  LowTomorrow <- as.numeric(symbol[i + 1, "Low"])
  HighTomorrow <- as.numeric(symbol[i + 1, "High"])

  # Config
  Equity <- getEndEq(accountname, CurrentDate)
  Position <-
    getPosQty(portfolioname, Symbol = instrument, Date = CurrentDate)

  # Check whether we have a position
  if (Position == 0) {
    # Start checking BUY rules

    # Check whether we have a Smash Day (Long).
    # Smash Day (Long) is when Todays Close is below Yesterdays Low.
    if (CloseToday < LowYesterday) {
      # Smash Day (Long)

      #Check whether todays close is above today's EMA
      if (CloseToday > EMA_today) {

        # BUY RULE: If today was a smash day place a STOP BUY order at todays high price.
        # (Buy tomorrow for 'price >= todays high')

        # Simulate STOP BUY order:

        # Option 1 to check: Check whether the open price tomorrow is above today's high
        # and add the transaction tomorrow at tomorrows open price.

        # Option 2 to check: Check whether today's high was lower than tomorrows high
        # and add the transaction tomorrow at today's high price.

        # Check Option 1
        if (OpenTomorrow > HighToday) {
          # Don't trade at the day before the last day
          if (CurrentDate != time(symbol[nrow(symbol) - 1])) {
            # Calculate the buy quantity
            BuyQuantity <- as.numeric(trunc(Equity / OpenTomorrow))
            # Add transaction
            addTxn(
```

```
            portfolioname,
            Symbol = instrument,
            TxnDate = TomorrowDate ,
            TxnPrice = OpenTomorrow,
            TxnQty = BuyQuantity,
            TxnFees = transactionCost
          )
          # Store the bar at which we placed the transaction
          BuyBar <- i
        }

      } else {
        # Check Option 2
        if (HighToday < HighTomorrow) {
          # Don't trade at the day before the last day
          if (CurrentDate != time(symbol[nrow(symbol) - 1])) {
            # Calculate the buy quantity
            BuyQuantity <- as.numeric(trunc(Equity / HighToday))
            # Add transaction
            addTxn(
              portfolioname,
              Symbol = instrument,
              TxnDate = TomorrowDate ,
              TxnPrice = HighToday,
              TxnQty = BuyQuantity,
              TxnFees = transactionCost
            )
            # Store the bar at which we placed the transaction
            BuyBar <- i
          }
        }
      }
    }
  }
} else {
  # We already have a position

  # Check the sell rules in the following order and sell at the
  # first condition which is satisfied.

  # Sell rules:
  # Rule 1: Sell if we hold the position longer than the specified
  # maximum holding period

  # Rule 2: Sell at tomorrow's opening price if the close price
  # today falls below the EMA

  # Rule 3: Sell if we meet the Smash Day (Short) requirements.
  # Today's close must be higher than yesterday's high

  # Rule 4: If no sell rule can be applied and we reach the
  # second last day. Sell at the last day.
```

```
    # Check Rule 1:
    if ((i - BuyBar) > maxHoldingPeriod) {
      # Place the sell transaction at todays close price
      addTxn(
        portfolioname,
        Symbol = instrument,
        TxnDate = CurrentDate,
        TxnPrice = as.numeric(symbol[i, "Close"]),
        TxnQty = -Position,
        TxnFees = transactionCost
      )

    } else {
      # Check Rule 2:
      if (as.numeric(symbol[i, "Close"]) < EMA_today) {
        # Place the sell transaction at tomorrow's open price
        addTxn(
          portfolioname,
          Symbol = instrument,
          TxnDate = time(symbol[i + 1]),
          TxnPrice = OpenTomorrow,
          TxnQty = -Position,
          TxnFees = transactionCost
        )

      } else {
        # Check Rule 3:

        # Sell Rule 3: If today was a smash day (short) place an order at todays
        # low price. (Buy tomorrow for 'price <= todays low')

        # Simulate this behaviour:

        # Option 1 to check: Check whether the open price tomorrow is below today's
        # low and add the transaction tomorrow at tomorrows open price.

        # Option 2 to check: Check whether today's low was larger than tomorrow's
        # low and add the transaction tomorrow at today's low price.

        # Check for Smash Day (Short)
        if (CloseToday > HighYesterday) {
          # Check for Option 1
          if (OpenTomorrow < LowToday) {
            # Add Sell transaction tomorrow at tomorrow's open price
            addTxn(
              portfolioname,
              Symbol = instrument,
              TxnDate = time(symbol[i + 1]),
              TxnPrice = OpenTomorrow,
              TxnQty = -Position,
              TxnFees = transactionCost
            )
```

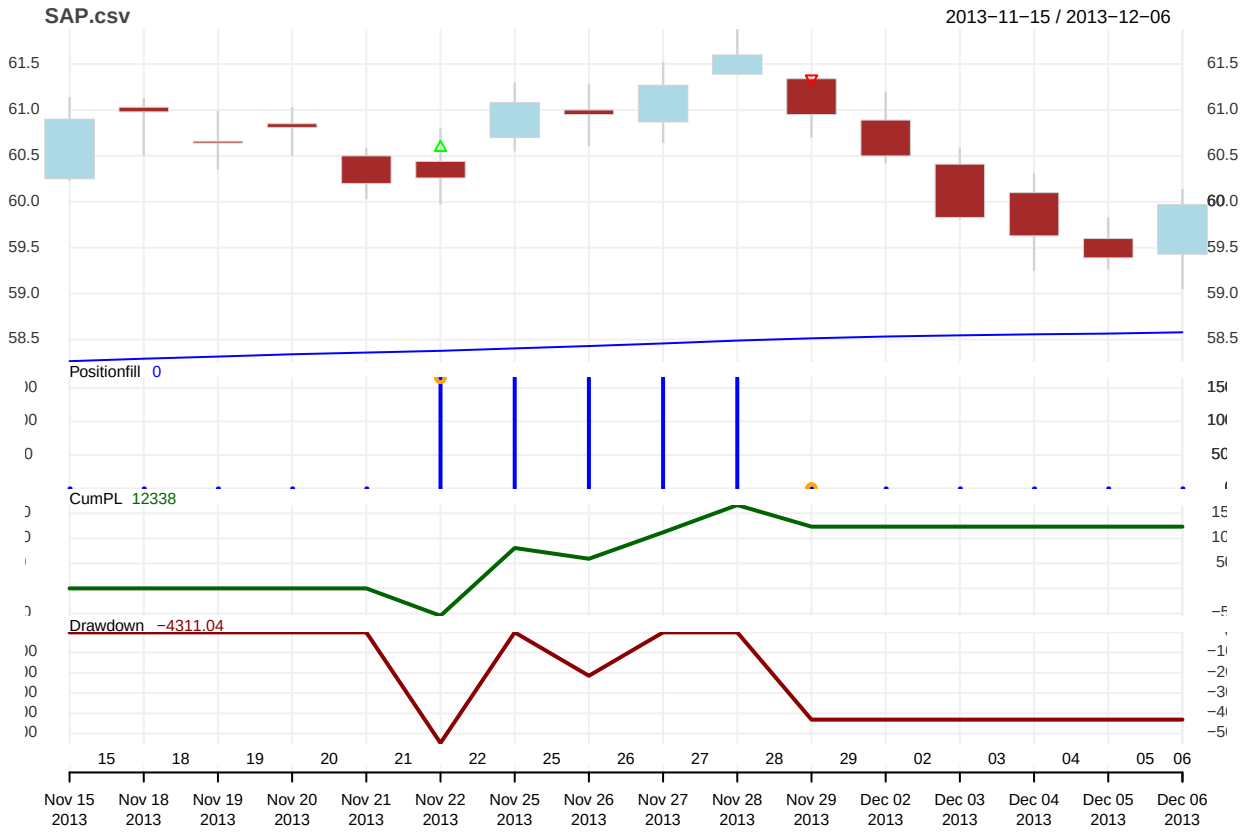
```
            } else {
              # Check for Option 2
              if (LowToday > LowTomorrow) {
                # Add Sell transaction tomorrow at today's low price
                addTxn(
                  portfolioname,
                  Symbol = instrument,
                  TxnDate = time(symbol[i + 1]),
                  TxnPrice = LowToday,
                  TxnQty = -Position,
                  TxnFees = transactionCost
                )
              }
            }
          } else {
            # Check Rule 4
            if (i == nrow(symbol) - 1) {
              # Add Sell transaction for the last day at the close price
              addTxn(
                portfolioname,
                Symbol = instrument,
                TxnDate = time(symbol[i + 1]),
                TxnPrice = as.numeric(symbol[i, "Close"]),
                TxnQty = -Position,
                TxnFees = transactionCost
              )
            }
          }
        }
      }
    }

    updatePortf(portfolioname, Symbols = instrument, Dates = CurrentDate)
    updateAcct(accountname, Dates = CurrentDate)
    updateEndEq(accountname, CurrentDate)

  } # End Bar-by-bar processing
} # End for loop for multiple instruments
```

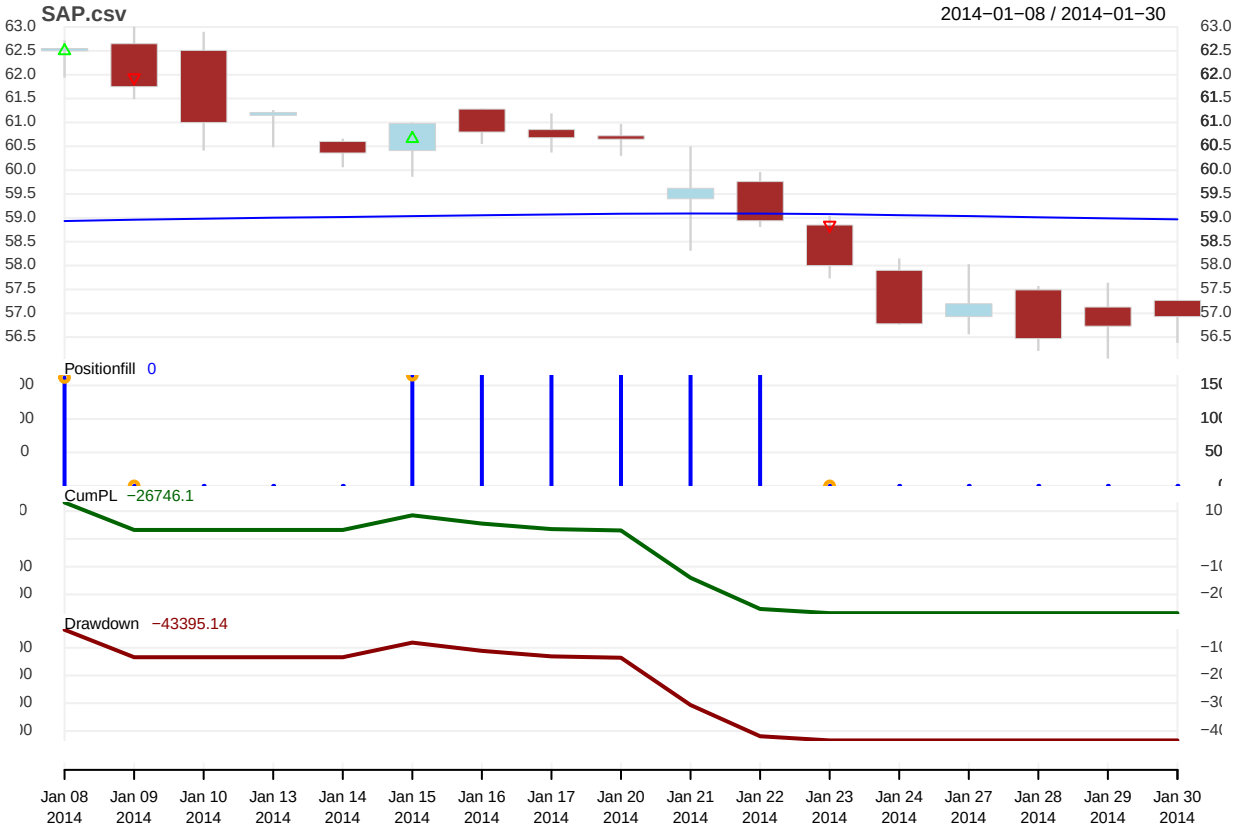
### Step 2: System Check

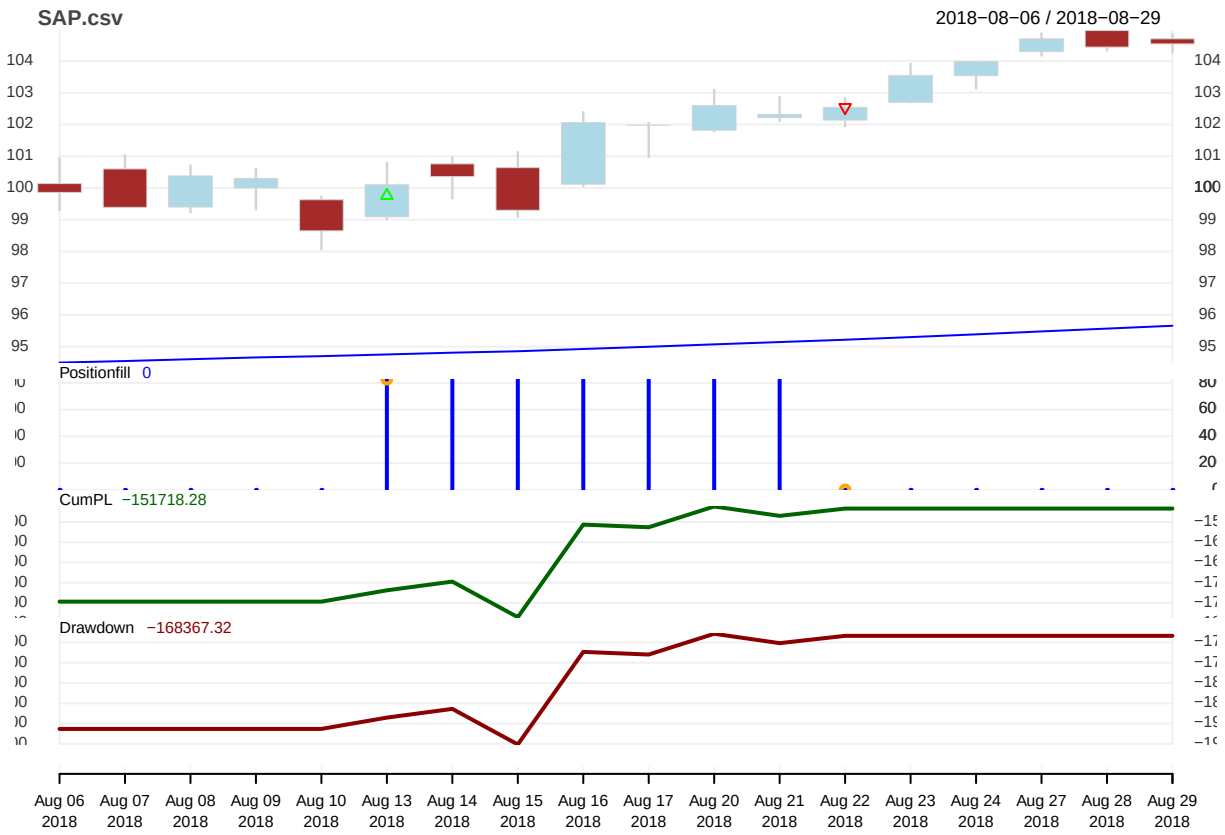
In order to make sure that the system works as designed the plots of some choosen transactions are printed in the following.

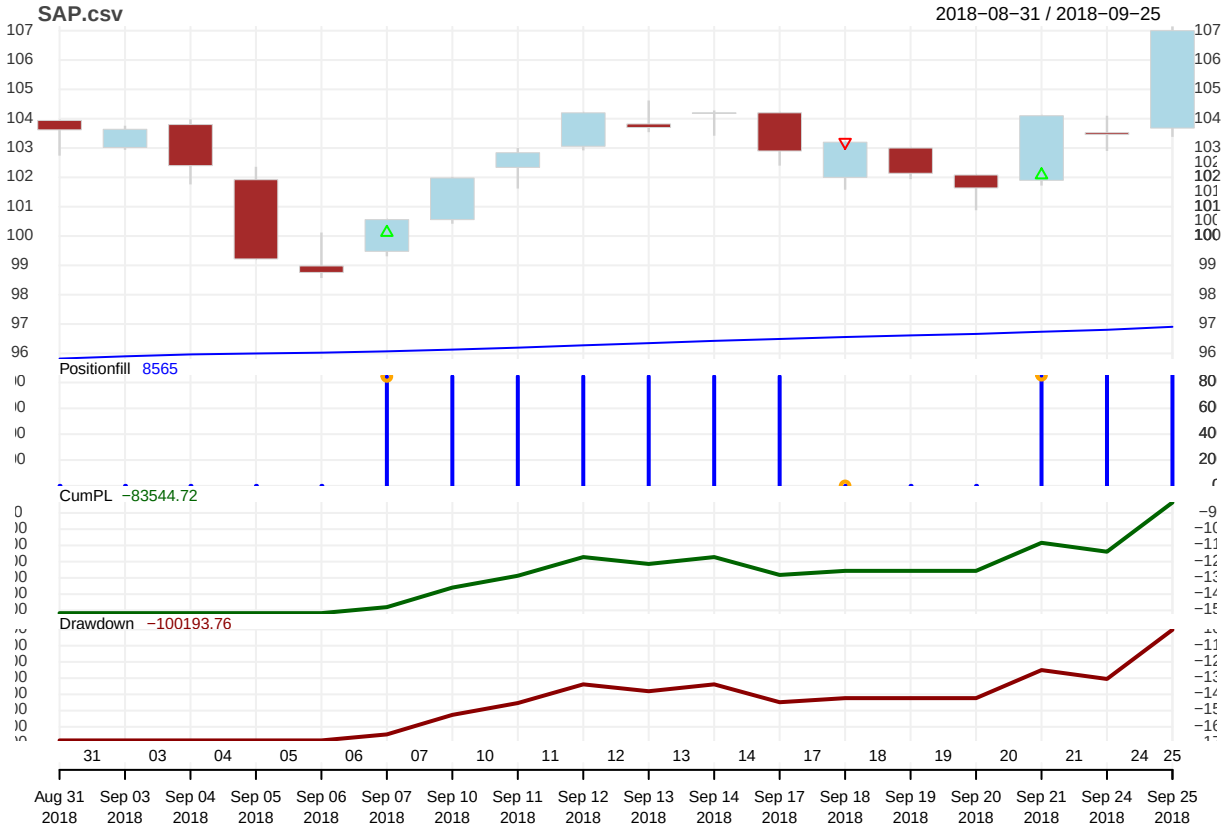
```
for (instrument in instrumentlist){
  rm(daterange_check)
  daterange_check <- c()
  transactionsInstrument <- getTxns(Portfolio=portfolioname,Symbol=instrument)
  for (i in c(2,6, (nrow(transactionsInstrument)-7), (nrow(transactionsInstrument)-5))) {
    from <- as.Date(index(transactionsInstrument[i,1]))-7
    to <- as.Date(index(transactionsInstrument[i+1,1]))+7
    daterange_check <- c(daterange_check, paste(from, "::", to, sep = ""))
  }
  for (daterange_check_i in daterange_check){
```

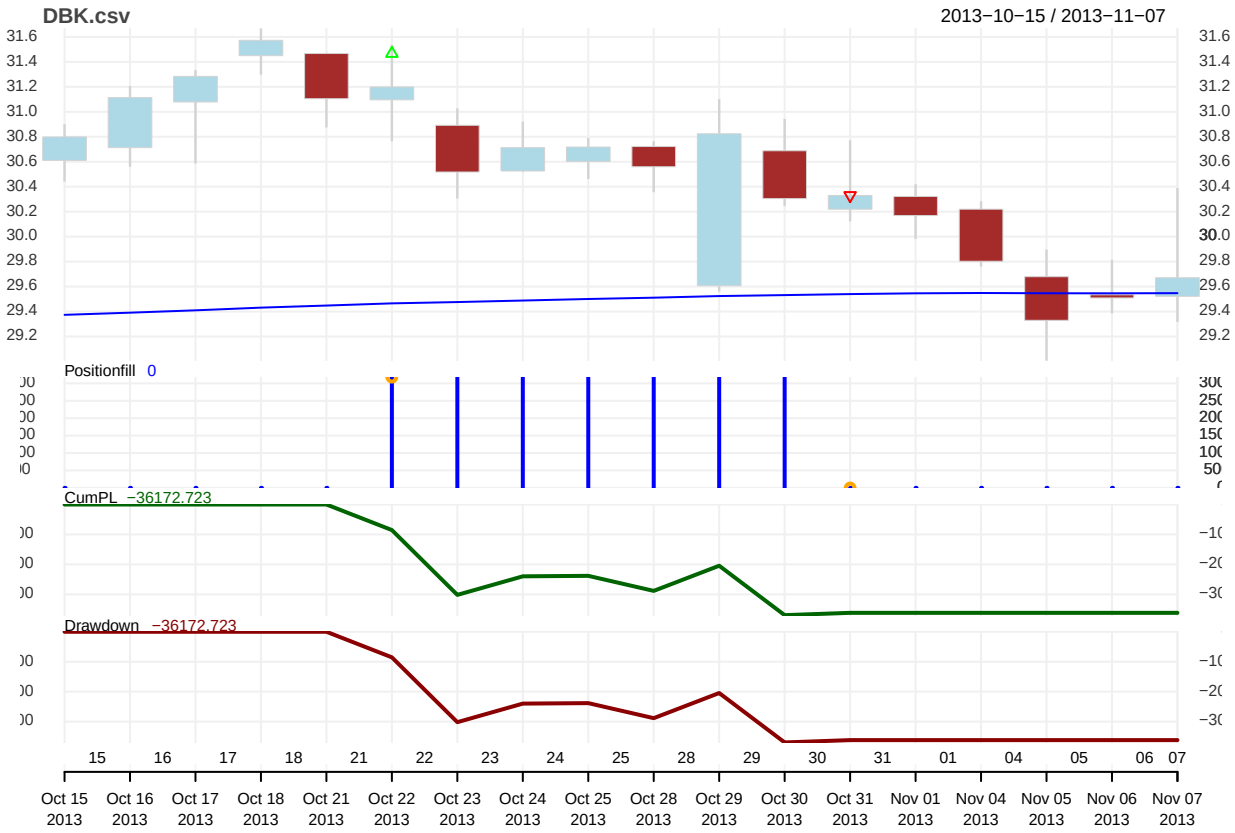
```
    print(chart.Posn(portfolioname,Symbol=instrument,type='candlesticks', theme=myTheme,subset=daterang
  }
}
```

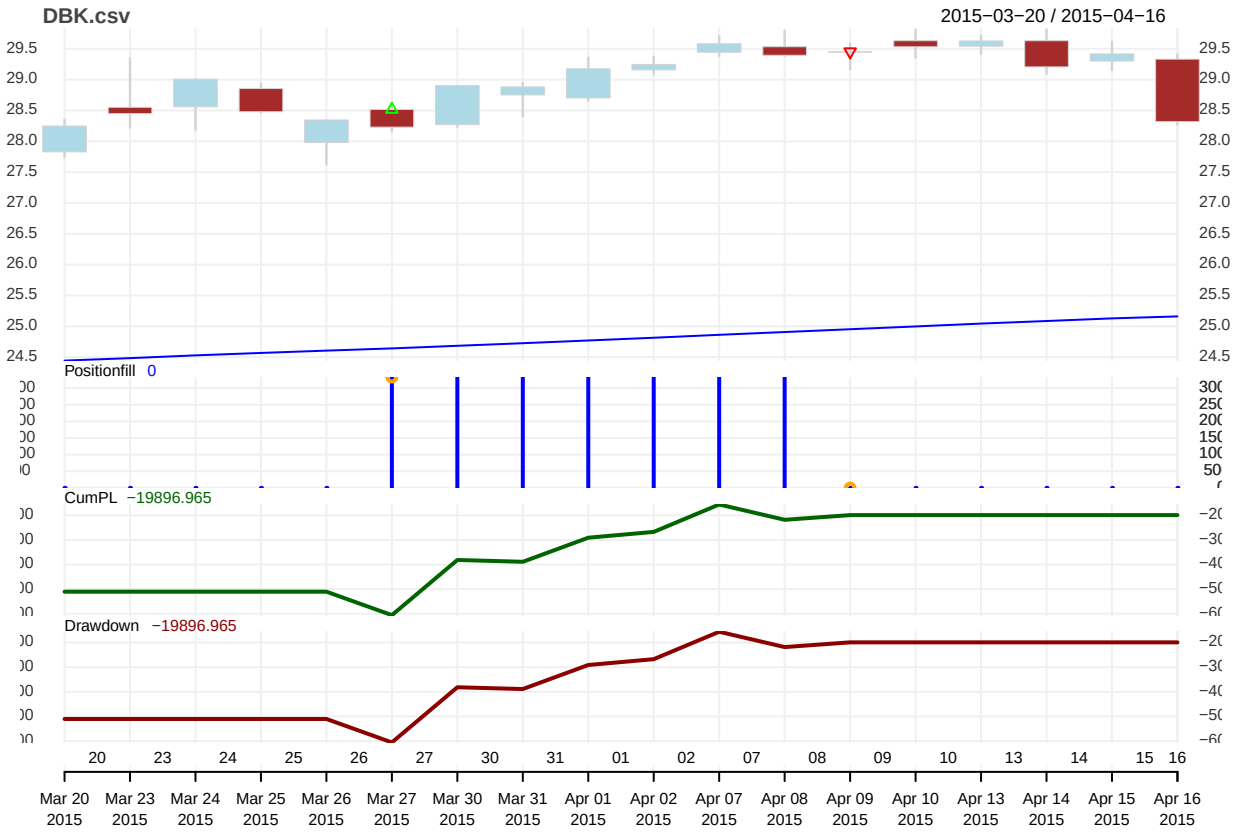


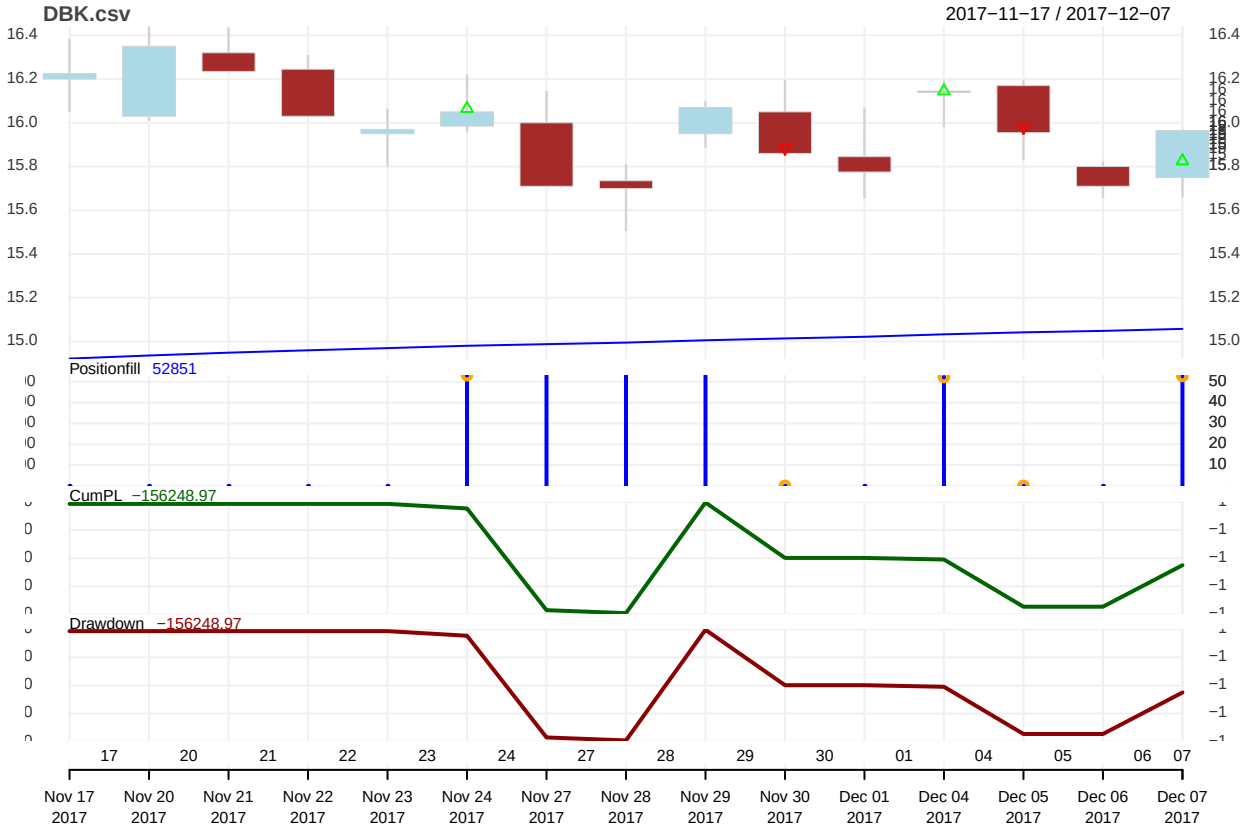

SAP.csv
2014−01−08 / 2014−01−30
Positionfill 0
CumPL −26746.1
Drawdown −43395.14
Jan 08 2014
Jan 09 2014
Jan 10 2014
Jan 13 2014
Jan 14 2014
Jan 15 2014
Jan 16 2014
Jan 17 2014
Jan 20 2014
Jan 21 2014
Jan 22 2014
Jan 23 2014
Jan 24 2014
Jan 27 2014
Jan 28 2014
Jan 29 2014
Jan 30 2014

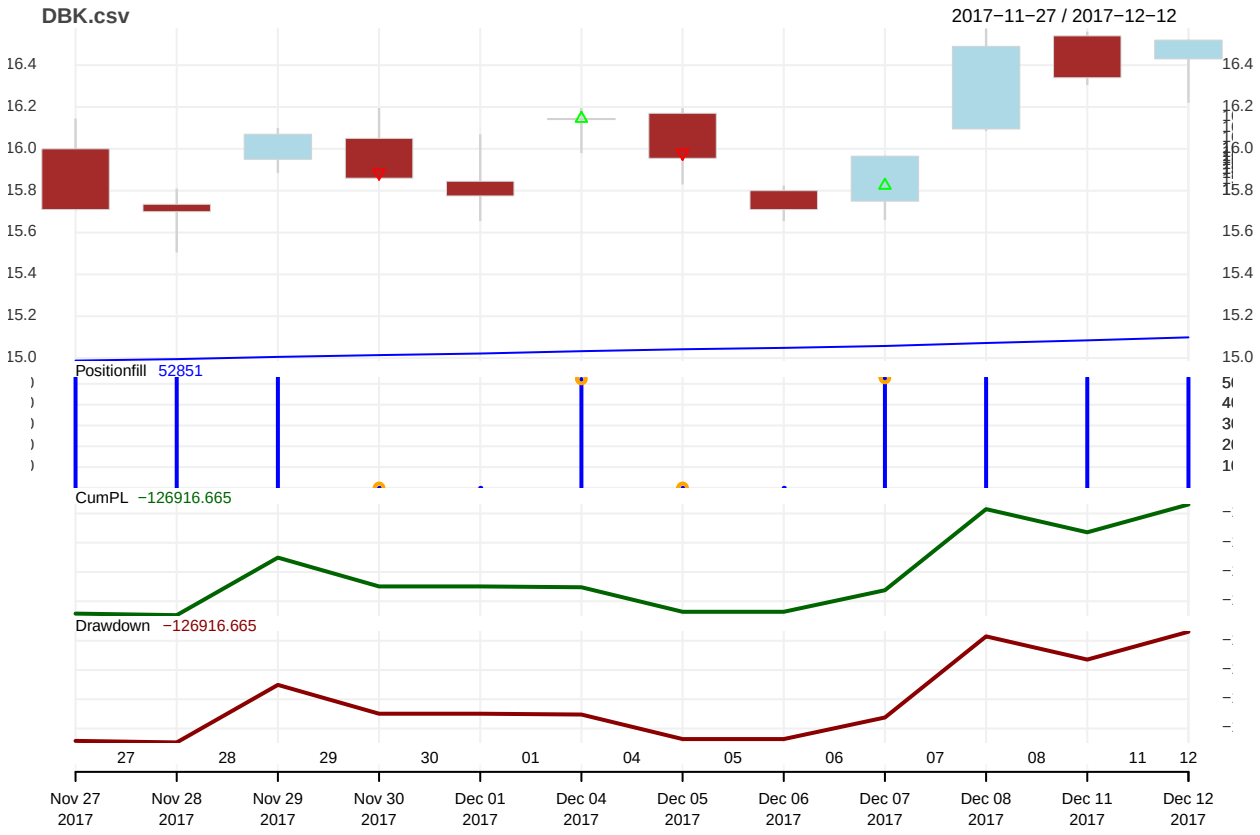

SAP.csv
2018−08−06 / 2018−08−29
Positionfill 0
CumPL −151718.28
Drawdown −168367.32
Aug 06 2018
Aug 07 2018
Aug 08 2018
Aug 09 2018
Aug 10 2018
Aug 13 2018
Aug 14 2018
Aug 15 2018
Aug 16 2018
Aug 17 2018
Aug 20 2018
Aug 21 2018
Aug 22 2018
Aug 23 2018
Aug 24 2018
Aug 27 2018
Aug 28 2018
Aug 29 2018


SAP.csv
2018−08−31 / 2018−09−25
Positionfill 8565
CumPL −83544.72
Drawdown −100193.76
Aug 31 2018
Sep 03 2018
Sep 04 2018
Sep 05 2018
Sep 06 2018
Sep 07 2018
Sep 10 2018
Sep 11 2018
Sep 12 2018
Sep 13 2018
Sep 14 2018
Sep 17 2018
Sep 18 2018
Sep 19 2018
Sep 20 2018
Sep 21 2018
Sep 24 2018
Sep 25 2018


DBK.csv
2013−10−15 / 2013−11−07
Positionfill 0
CumPL −36172.723
Drawdown −36172.723
Oct 15 2013
Oct 16 2013
Oct 17 2013
Oct 18 2013
Oct 21 2013
Oct 22 2013
Oct 23 2013
Oct 24 2013
Oct 25 2013
Oct 28 2013
Oct 29 2013
Oct 30 2013
Oct 31 2013
Nov 01 2013
Nov 04 2013
Nov 05 2013
Nov 06 2013
Nov 07 2013


DBK.csv
2015−03−20 / 2015−04−16
Positionfill 0
CumPL −19896.965
Drawdown −19896.965
Mar 20 2015
Mar 23 2015
Mar 24 2015
Mar 25 2015
Mar 26 2015
Mar 27 2015
Mar 30 2015
Mar 31 2015
Apr 01 2015
Apr 02 2015
Apr 07 2015
Apr 08 2015
Apr 09 2015
Apr 10 2015
Apr 13 2015
Apr 14 2015
Apr 15 2015
Apr 16 2015

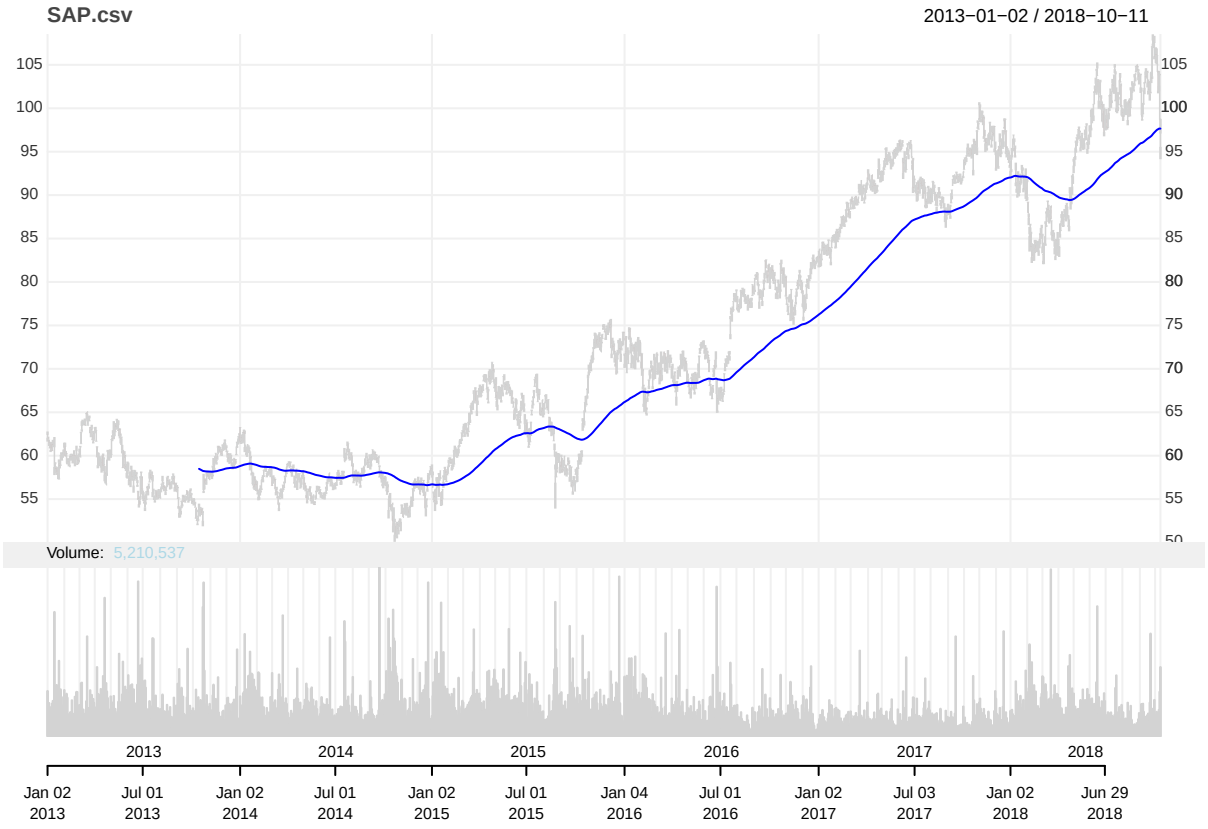

DBK.csv
2017−11−17 / 2017−12−07
16.4
16.2
16.0
15.8
15.6
15.4
15.2
15.0
Positionfill 52851
50
40
30
20
10
CumPL −156248.97
Drawdown −156248.97
17
20
21
22
23
24
27
28
29
30
01
04
05
06
07
Nov 17 2017
Nov 20 2017
Nov 21 2017
Nov 22 2017
Nov 23 2017
Nov 24 2017
Nov 27 2017
Nov 28 2017
Nov 29 2017
Nov 30 2017
Dec 01 2017
Dec 04 2017
Dec 05 2017
Dec 06 2017
Dec 07 2017

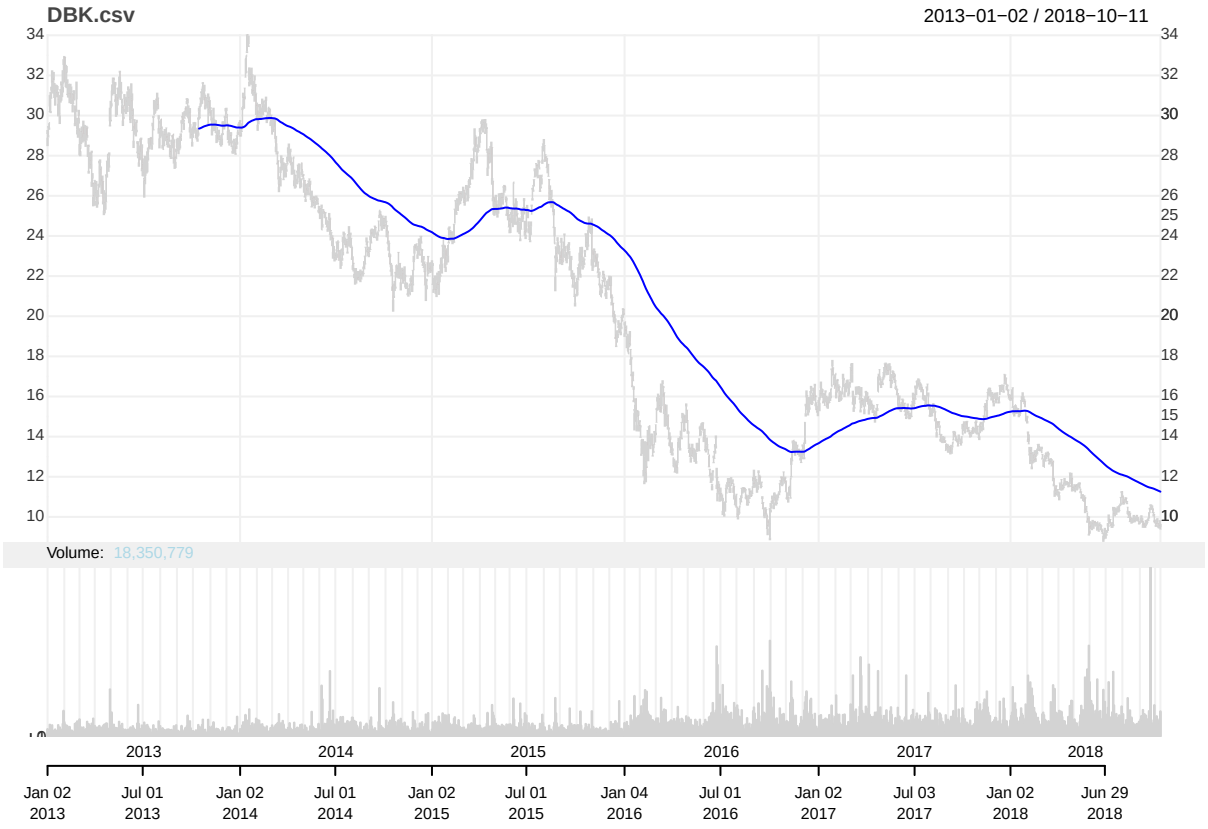



## Part C: Analysis and Reporting

### Step 1: Visualize original data

Plot of the instrument with the EMA line which indicates the general trend of the stock exponentially smoothed for the last 200 days. Moreover, the tradevolume is added below the graph.


SAP.csv
2013−01−02 / 2018−10−11
105
100
95
90
85
80
75
70
65
60
55
Volume: 5,210,537
2013
2014
2015
2016
2017
2018
Jan 02 2013
Jul 01 2013
Jan 02 2014
Jul 01 2014
Jan 02 2015
Jul 01 2015
Jan 04 2016
Jul 01 2016
Jan 02 2017
Jul 03 2017
Jan 02 2018
Jun 29 2018




### Step 2: All transactions performed by the trading system

The following table can be used to get a better overview of the transactions performed and the exact details per transaction.

```
## [1] "SAP.csv"
##            Txn.Qty Txn.Price Txn.Fees  Txn.Value Txn.Avg.Cost
## 1999-12-31       0      0.00        0        0.0         0.00
## 2013-11-22   16504     60.59      -20   999977.4        60.59
## 2013-11-29  -16504     61.34      -20 -1012355.4        61.34
## 2014-01-08   16197     62.50      -20  1012312.5        62.50
## 2014-01-09  -16197     61.94      -20 -1003242.2        61.94
## 2014-01-15   16538     60.66      -20  1003195.1        60.66
## 2014-01-23  -16538     58.85      -20  -973261.3        58.85
## 2014-09-04   16404     59.33      -20   973249.3        59.33
## 2014-09-15  -16404     59.96      -20  -983583.8        59.96
## 2014-09-17   16436     59.84      -20   983530.2        59.84
## 2014-09-22  -16436     57.76      -20  -949343.4        57.76
## 2015-02-10   16095     58.98      -20   949283.1        58.98
## 2015-02-16  -16095     60.44      -20  -972781.8        60.44
## 2015-03-27   14540     66.90      -20   972726.0        66.90
## 2015-04-01  -14540     66.82      -20  -971562.8        66.82
## 2015-05-07   14492     67.04      -20   971543.7        67.04
## 2015-05-12  -14492     67.31      -20  -975456.5        67.31
## 2015-05-13   14590     67.53      -20   985262.7        67.53
## 2015-05-21  -14590     68.17      -20  -994600.3        68.17
```

```
## 2015-06-10   15084     65.28      -20   984683.5        65.28
## 2015-06-12  -15084     65.11      -20  -982119.2        65.11
## 2015-06-16   15161     64.78      -20   982129.6        64.78
## 2015-06-24  -15161     66.59      -20 -1009571.0        66.59
## 2015-06-25   15092     66.94      -20  1010258.5        66.94
## 2015-07-06  -15092     61.13      -20  -922574.0        61.13
## 2015-08-10   13878     66.42      -20   921776.8        66.42
## 2015-08-13  -13878     64.36      -20  -893188.1        64.36
## 2015-11-25   12013     74.35      -20   893166.5        74.35
## 2015-11-27  -12013     74.05      -20  -889562.7        74.05
## 2015-12-07   12251     72.61      -20   889545.1        72.61
## 2015-12-08  -12251     72.55      -20  -888810.0        72.55
## 2015-12-15   12444     71.42      -20   888750.5        71.42
## 2015-12-18  -12444     72.03      -20  -896341.3        72.03
## 2015-12-23   12373     72.44      -20   896300.1        72.44
## 2016-01-07  -12373     69.60      -20  -861160.8        69.60
## 2016-01-21   12050     71.46      -20   861093.0        71.46
## 2016-01-22  -12050     70.34      -20  -847597.0        70.34
## 2016-02-25   12112     69.98      -20   847597.8        69.98
## 2016-03-07  -12112     69.57      -20  -842631.8        69.57
## 2016-03-09   12071     69.80      -20   842555.8        69.80
## 2016-03-10  -12071     69.40      -20  -837727.4        69.40
## 2016-03-29   11744     71.33      -20   837699.5        71.33
## 2016-04-07  -11744     68.19      -20  -800823.4        68.19
## 2016-04-27   11263     71.10      -20   800799.3        71.10
## 2016-04-28  -11263     70.18      -20  -790437.3        70.18
## 2016-06-03   10915     72.41      -20   790355.1        72.41
## 2016-06-08  -10915     71.64      -20  -781950.6        71.64
## 2016-09-13    9905     78.94      -20   781900.7        78.94
## 2016-09-22   -9905     82.36      -20  -815775.8        82.36
## 2016-10-14   10356     78.77      -20   815742.1        78.77
## 2016-10-17  -10356     78.49      -20  -812842.4        78.49
## 2016-10-27    9989     81.37      -20   812804.9        81.37
## 2016-11-07   -9989     78.30      -20  -782138.7        78.30
## 2016-12-05   10104     77.41      -20   782150.6        77.41
## 2016-12-12  -10104     79.32      -20  -801449.3        79.32
## 2017-02-02    9411     85.15      -20   801346.7        85.15
## 2017-02-13   -9411     87.38      -20  -822333.2        87.38
## 2017-03-28    9141     89.96      -20   822324.4        89.96
## 2017-04-06   -9141     91.43      -20  -835761.6        91.43
## 2017-04-12    9122     91.62      -20   835757.6        91.62
## 2017-04-25   -9122     93.13      -20  -849531.9        93.13
## 2017-05-12    8985     94.54      -20   849441.9        94.54
## 2017-05-23   -8985     94.68      -20  -850699.8        94.68
## 2017-06-16    9042     94.08      -20   850671.4        94.08
## 2017-06-23   -9042     95.44      -20  -862968.5        95.44
## 2017-06-29    9179     94.01      -20   862917.8        94.01
## 2017-07-10   -9179     91.31      -20  -838134.5        91.31
## 2017-07-12    9144     91.66      -20   838139.0        91.66
## 2017-07-20   -9144     90.18      -20  -824605.9        90.18
## 2017-08-14    9286     88.79      -20   824503.9        88.79
## 2017-08-17   -9286     89.96      -20  -835368.6        89.96
## 2017-08-22    9360     89.25      -20   835380.0        89.25
## 2017-08-29   -9360     87.14      -20  -815630.4        87.14
```

```
## 2017-11-14    8451     96.51      -20   815606.0        96.51
## 2017-11-22   -8451     96.44      -20  -815014.4        96.44
## 2017-12-12    8529     95.55      -20   814945.9        95.55
## 2017-12-19   -8529     97.65      -20  -832856.9        97.65
## 2018-01-03    8864     93.95      -20   832772.8        93.95
## 2018-01-12   -8864     91.00      -20  -806624.0        91.00
## 2018-05-16    8364     96.44      -20   806624.2        96.44
## 2018-05-18   -8364     95.72      -20  -800602.1        95.72
## 2018-05-30    8329     96.12      -20   800583.5        96.12
## 2018-06-04   -8329     96.72      -20  -805580.9        96.72
## 2018-06-20    7889    102.10      -20   805466.9       102.10
## 2018-06-21   -7889    101.78      -20  -802942.4       101.78
## 2018-06-27    8076     99.42      -20   802915.9        99.42
## 2018-06-28   -8076     97.75      -20  -789429.0        97.75
## 2018-07-05    7990     98.80      -20   789412.0        98.80
## 2018-07-11   -7990    100.82      -20  -805551.8       100.82
## 2018-07-12    8048    101.44      -20   816389.1       101.44
## 2018-07-19   -8048    103.90      -20  -836187.2       103.90
## 2018-08-13    8273     99.76      -20   825314.5        99.76
## 2018-08-22   -8273    102.54      -20  -848313.4       102.54
## 2018-09-07    8472    100.12      -20   848216.6       100.12
## 2018-09-18   -8472    103.20      -20  -874310.4       103.20
## 2018-09-21    8565    102.08      -20   874315.2       102.08
## 2018-09-27   -8565    107.02      -20  -916626.3       107.02
## 2018-10-09    8793    104.24      -20   916582.3       104.24
## 2018-10-11   -8793     98.71      -20  -867957.0        98.71
##            Net.Txn.Realized.PL
## 1999-12-31                0.00
## 2013-11-22              -20.00
## 2013-11-29            12358.00
## 2014-01-08              -20.00
## 2014-01-09            -9090.32
## 2014-01-15              -20.00
## 2014-01-23           -29953.78
## 2014-09-04              -20.00
## 2014-09-15            10314.52
## 2014-09-17              -20.00
## 2014-09-22           -34206.88
## 2015-02-10              -20.00
## 2015-02-16            23478.70
## 2015-03-27              -20.00
## 2015-04-01            -1183.20
## 2015-05-07              -20.00
## 2015-05-12             3892.84
## 2015-05-13              -20.00
## 2015-05-21             9317.60
## 2015-06-10              -20.00
## 2015-06-12            -2584.28
## 2015-06-16              -20.00
## 2015-06-24            27421.41
## 2015-06-25              -20.00
## 2015-07-06           -87704.52
## 2015-08-10              -20.00
## 2015-08-13           -28608.68
```

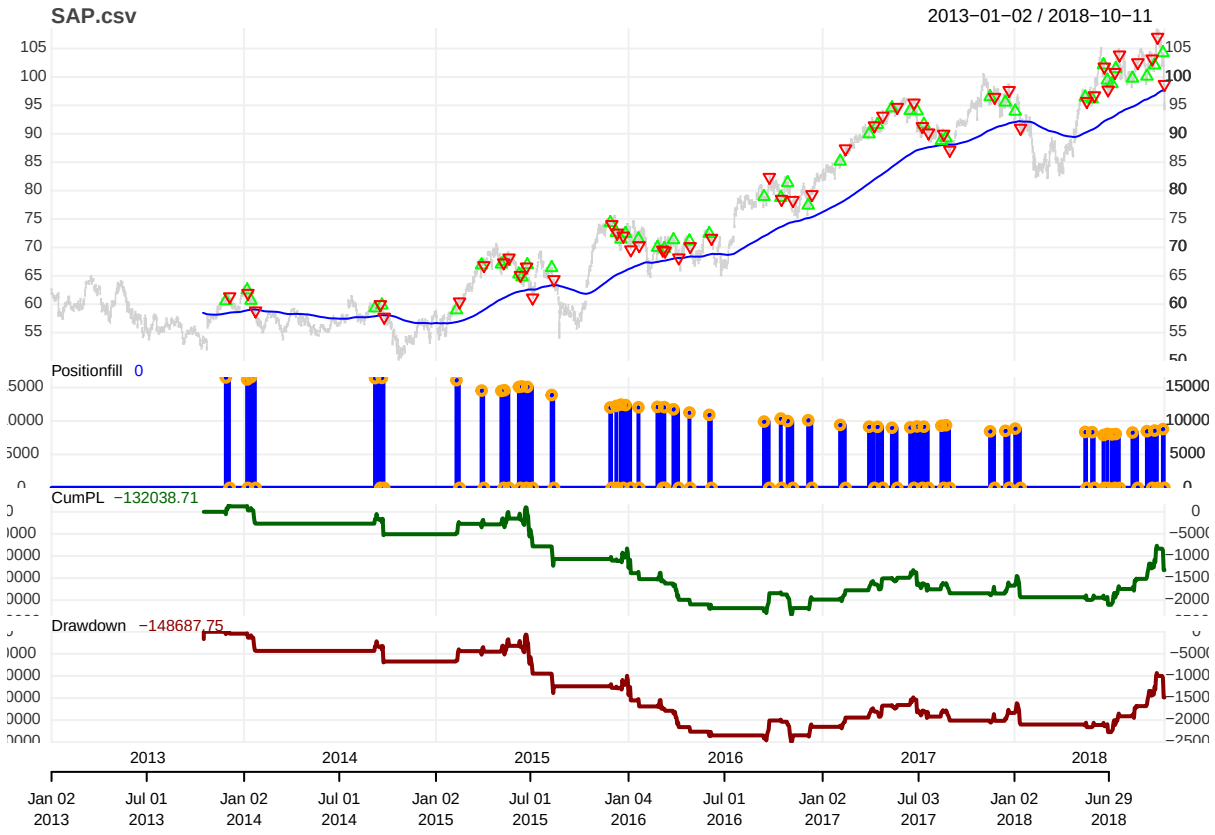
```
## 2015-11-25            -20.00
## 2015-11-27          -3623.90
## 2015-12-07            -20.00
## 2015-12-08           -755.06
## 2015-12-15            -20.00
## 2015-12-18           7570.84
## 2015-12-23            -20.00
## 2016-01-07         -35159.32
## 2016-01-21            -20.00
## 2016-01-22         -13516.00
## 2016-02-25            -20.00
## 2016-03-07          -4985.92
## 2016-03-09            -20.00
## 2016-03-10          -4848.40
## 2016-03-29            -20.00
## 2016-04-07         -36896.16
## 2016-04-27            -20.00
## 2016-04-28         -10381.96
## 2016-06-03            -20.00
## 2016-06-08          -8424.55
## 2016-09-13            -20.00
## 2016-09-22          33855.10
## 2016-10-14            -20.00
## 2016-10-17          -2919.68
## 2016-10-27            -20.00
## 2016-11-07         -30686.23
## 2016-12-05            -20.00
## 2016-12-12          19278.64
## 2017-02-02            -20.00
## 2017-02-13          20966.53
## 2017-03-28            -20.00
## 2017-04-06          13417.27
## 2017-04-12            -20.00
## 2017-04-25          13754.22
## 2017-05-12            -20.00
## 2017-05-23           1237.90
## 2017-06-16            -20.00
## 2017-06-23          12277.12
## 2017-06-29            -20.00
## 2017-07-10         -24803.30
## 2017-07-12            -20.00
## 2017-07-20         -13553.12
## 2017-08-14            -20.00
## 2017-08-17          10844.62
## 2017-08-22            -20.00
## 2017-08-29         -19769.60
## 2017-11-14            -20.00
## 2017-11-22           -611.57
## 2017-12-12            -20.00
## 2017-12-19          17890.90
## 2018-01-03            -20.00
## 2018-01-12         -26168.80
## 2018-05-16            -20.00
## 2018-05-18          -6042.08
```

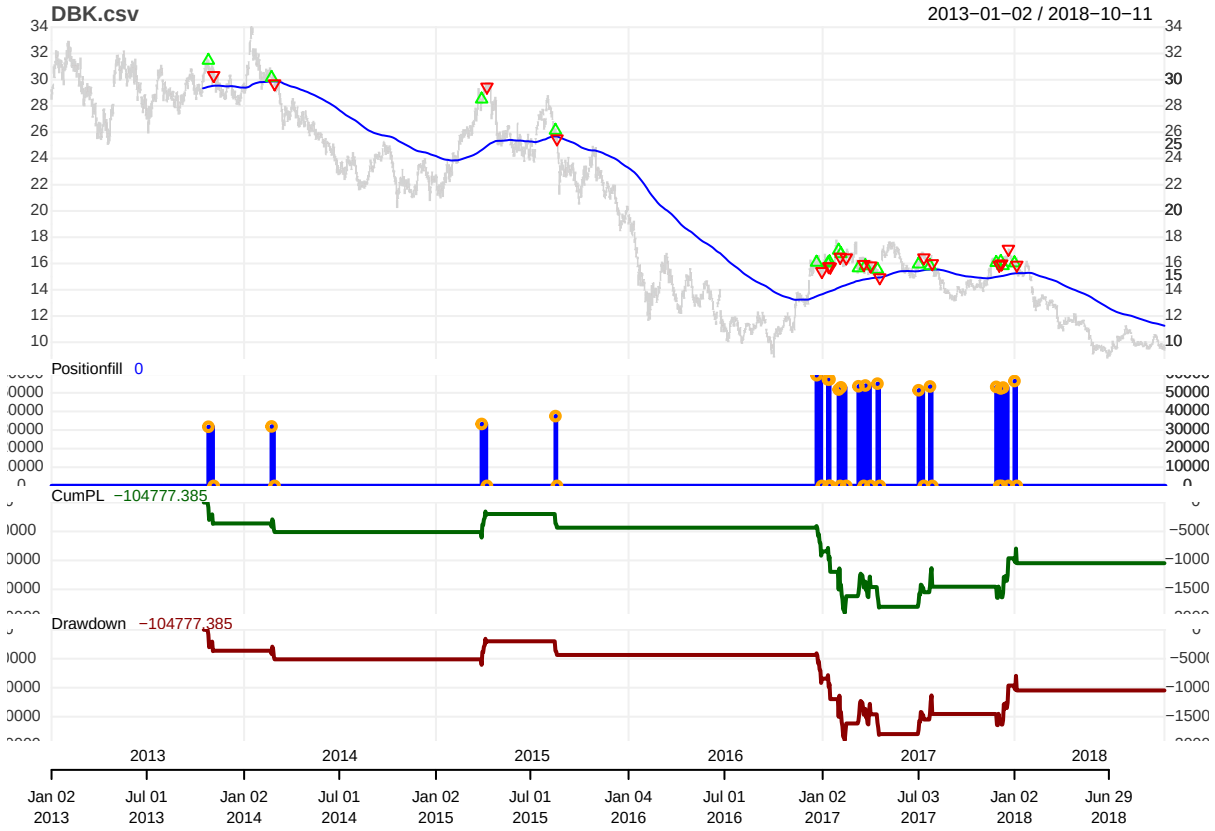
```
## 2018-05-30             -20.00
## 2018-06-04            4977.40
## 2018-06-20             -20.00
## 2018-06-21           -2544.48
## 2018-06-27             -20.00
## 2018-06-28          -13506.92
## 2018-07-05             -20.00
## 2018-07-11           16119.80
## 2018-07-12             -20.00
## 2018-07-19           19778.08
## 2018-08-13             -20.00
## 2018-08-22           22978.94
## 2018-09-07             -20.00
## 2018-09-18           26073.76
## 2018-09-21             -20.00
## 2018-09-27           42291.10
## 2018-10-09             -20.00
## 2018-10-11          -48645.29
## [1] "DBK.csv"
##            Txn.Qty Txn.Price Txn.Fees Txn.Value Txn.Avg.Cost
## 1999-12-31       0     0.000        0       0.0        0.000
## 2013-10-22   31779    31.467      -20  999989.8       31.467
## 2013-10-31  -31779    30.330      -20 -963857.1       30.330
## 2014-02-21   31952    30.164      -20  963800.1       30.164
## 2014-02-27  -31952    29.700      -20 -948974.4       29.700
## 2015-03-27   33278    28.516      -20  948955.4       28.516
## 2015-04-09  -33278    29.453      -20 -980136.9       29.453
## 2015-08-17   37498    26.137      -20  980085.2       26.137
## 2015-08-19  -37498    25.508      -20 -956499.0       25.508
## 2016-12-20   59471    16.083      -20  956472.1       16.083
## 2016-12-30  -59471    15.396      -20 -915615.5       15.396
## 2017-01-11   57148    16.021      -20  915568.1       16.021
## 2017-01-12  -57148    15.771      -20 -901281.1       15.771
## 2017-01-13   57211    16.110      -20  921669.2       16.110
## 2017-01-16  -57211    15.744      -20 -900730.0       15.744
## 2017-01-31   51774    17.002      -20  880261.5       17.002
## 2017-02-02  -51774    16.494      -20 -853960.4       16.494
## 2017-02-03   52958    16.735      -20  886252.1       16.735
## 2017-02-14  -52958    16.440      -20 -870629.5       16.440
## 2017-03-08   53532    15.659      -20  838257.6       15.659
## 2017-03-17  -53532    15.940      -20 -853300.1       15.940
## 2017-03-21   53953    15.815      -20  853266.7       15.815
## 2017-03-30  -53953    15.830      -20 -854076.0       15.830
## 2017-04-12   54939    15.545      -20  854026.8       15.545
## 2017-04-19  -54939    14.925      -20 -819964.6       14.925
## 2017-07-03   51406    15.950      -20  819925.7       15.950
## 2017-07-12  -51406    16.440      -20 -845114.6       16.440
## 2017-07-24   53419    15.820      -20  845088.6       15.820
## 2017-07-27  -53419    16.000      -20 -854704.0       16.000
## 2017-11-24   53200    16.065      -20  854658.0       16.065
## 2017-11-30  -53200    15.885      -20 -845082.0       15.885
## 2017-12-04   52341    16.145      -20  845045.4       16.145
## 2017-12-05  -52341    15.980      -20 -836409.2       15.980
## 2017-12-07   52851    15.825      -20  836367.1       15.825
```

```
## 2017-12-18  -52851    17.100      -20 -903752.1       17.100
## 2018-01-02   56362    16.034      -20  903708.3       16.034
## 2018-01-05  -56362    15.884      -20 -895254.0       15.884
##            Net.Txn.Realized.PL
## 1999-12-31               0.000
## 2013-10-22             -20.000
## 2013-10-31          -36152.723
## 2014-02-21             -20.000
## 2014-02-27          -14845.728
## 2015-03-27             -20.000
## 2015-04-09           31161.486
## 2015-08-17             -20.000
## 2015-08-19          -23606.242
## 2016-12-20             -20.000
## 2016-12-30          -40876.577
## 2017-01-11             -20.000
## 2017-01-12          -14307.000
## 2017-01-13             -20.000
## 2017-01-16          -20959.226
## 2017-01-31             -20.000
## 2017-02-02          -26321.192
## 2017-02-03             -20.000
## 2017-02-14          -15642.610
## 2017-03-08             -20.000
## 2017-03-17           15022.492
## 2017-03-21             -20.000
## 2017-03-30             789.295
## 2017-04-12             -20.000
## 2017-04-19          -34082.180
## 2017-07-03             -20.000
## 2017-07-12           25168.940
## 2017-07-24             -20.000
## 2017-07-27            9595.420
## 2017-11-24             -20.000
## 2017-11-30          -9596.000
## 2017-12-04             -20.000
## 2017-12-05          -8656.265
## 2017-12-07             -20.000
## 2017-12-18           67365.025
## 2018-01-02             -20.000
## 2018-01-05          -8474.300
```

**Step 3: Graph which visualize the transactions**

The following graph shows the combined view of the performance of the Smash Day trading system. It visualizes the trades (buy-transactions are visualized in green and sell-transactions are visualized in red). Moreover, the size of the blue squares indicates the size of the position (height) and the holding duration of the position (width). The green line shows the cummulative net profit curve, while the red curve indicates the drawdown on each day compared to the last reached high.


SAP.csv
2013−01−02 / 2018−10−11
Positionfill 0
CumPL −132038.71
Drawdown −148687.75
2013
2014
2015
2016
2017
2018
Jan 02 2013
Jul 01 2013
Jan 02 2014
Jul 01 2014
Jan 02 2015
Jul 01 2015
Jan 04 2016
Jul 01 2016
Jan 02 2017
Jul 03 2017
Jan 02 2018
Jun 29 2018




### Step 4: Performance Statistics

The following table summarizes some important trading statistics for all instruments.

```
library(PerformanceAnalytics) # contains lots of methods to investigate performance
# obtain the portfolio returns - with these you can compute virtually any financial metrics you wish
rets <- PortfReturns(Account=accountname)
rownames(rets) <- NULL # this step is important!

tstats <- tradeStats(Portfolio=portfolioname, Symbols=instrumentlist)
for (i in 1:nrow(tstats)) {
  trades.tab <- cbind(
    c("Trades", "Win Percent", "Loss Percent","W/L Ratio"),
    c(tstats[i,"Num.Trades"],tstats[i,"Percent.Positive"],tstats[i,"Percent.Negative"],tstats[i,"Percen
  print(row.names(tstats[i,]))
  print(trades.tab)
  }
```

```
## [1] "SAP.csv"
##      [,1]           [,2]
## [1,] "Trades"       "49"
## [2,] "Win Percent"  "44.8979591836735"
## [3,] "Loss Percent" "55.1020408163265"
## [4,] "W/L Ratio"    "0.814814814814815"
## [1] "DBK.csv"
##      [,1]           [,2]
## [1,] "Trades"       "18"
```

```
## [2,] "Win Percent"  "33.3333333333333"
## [3,] "Loss Percent" "66.6666666666667"
## [4,] "W/L Ratio"    "0.5"
```

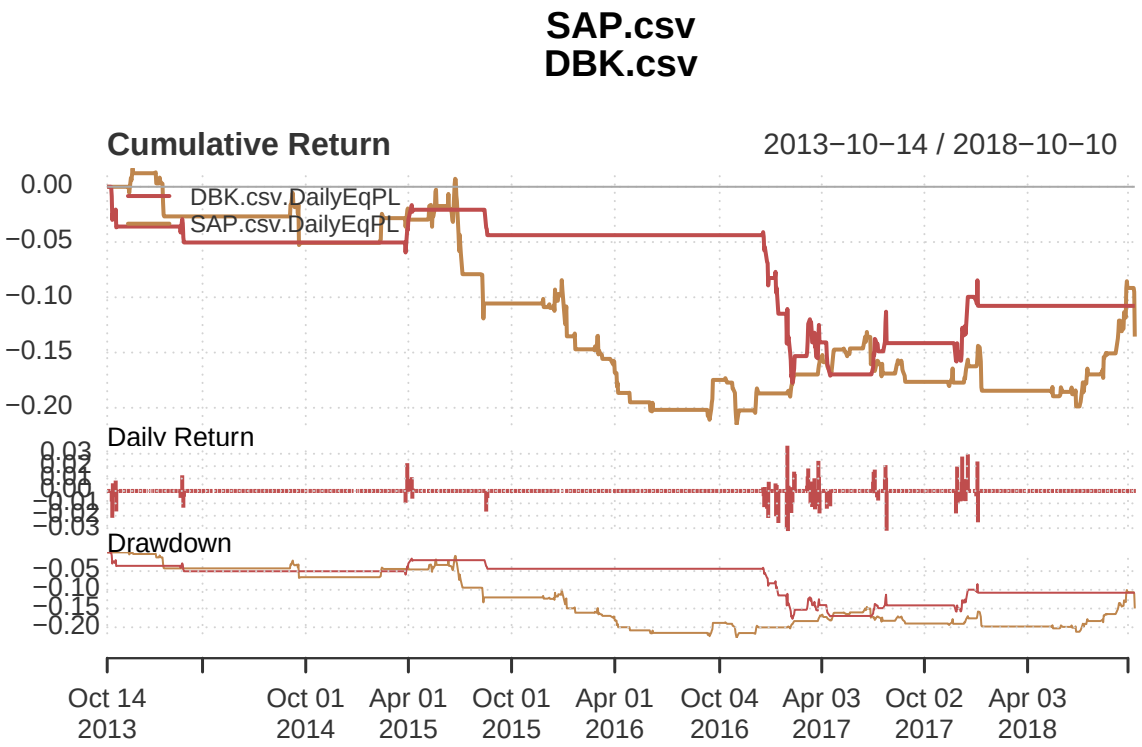
**Step 5: Calculate statistics of the Portfolio and all instruments in the portfolio**

```
tab.perf <- table.Arbitrary(rets, metrics=c("Return.cumulative", "Return.annualized","SharpeRatio.annual
                            metricsNames=c("Cumulative Return", "Annualized Return", "Annualized Sharp
tab.risk <- table.Arbitrary(rets, metrics=c("StdDev.annualized", "maxDrawdown","VaR","ES"),
                            metricsNames=c("Annualized StdDev", "Max Drawdown", "Value-at-Risk","Conditi
# present the portfolio statistics
for (i in 1:ncol(tab.perf)) {
somestats <- data.frame(rownames(tab.perf),tab.perf[i,1], rownames(tab.risk),tab.risk[i,1])
colnames(rets[,i])
colnames(somestats) <- c("Performance Metric", "Performance Value", "Risk Metric", "Risk Value")
print(somestats)
}
```

```
##       Performance Metric Performance Value       Risk Metric Risk Value
## 1      Cumulative Return        -0.1077696 Annualized StdDev 0.06075979
## 2      Annualized Return        -0.1077696      Max Drawdown 0.06075979
## 3 Annualized Sharp Ratio        -0.1077696     Value-at-Risk 0.06075979
## 4           Calmar Ratio        -0.1077696   Conditional VaR 0.06075979
##       Performance Metric Performance Value       Risk Metric Risk Value
## 1      Cumulative Return       -0.02251276 Annualized StdDev  0.1774743
## 2      Annualized Return       -0.02251276      Max Drawdown  0.1774743
## 3 Annualized Sharp Ratio       -0.02251276     Value-at-Risk  0.1774743
## 4           Calmar Ratio       -0.02251276   Conditional VaR  0.1774743
```

**Step 6: Visualize returns of the trading strategy**

```
charts.PerformanceSummary(rets,colorset=rainbow12equal,main=instrumentlist)
```




## Compare with Buy and Hold Strategy

In order to compare the trading strategy properly we need to define a benchmark against which we can measure the results. In this case a simple buy and hold strategy is used. At the first date of the trading period we place a buy order and sell our position at the last day of the selected period. In order to do this we create a new Portfolio and a new Account.

### Step 1: Perform the Buy and Hold Strategy

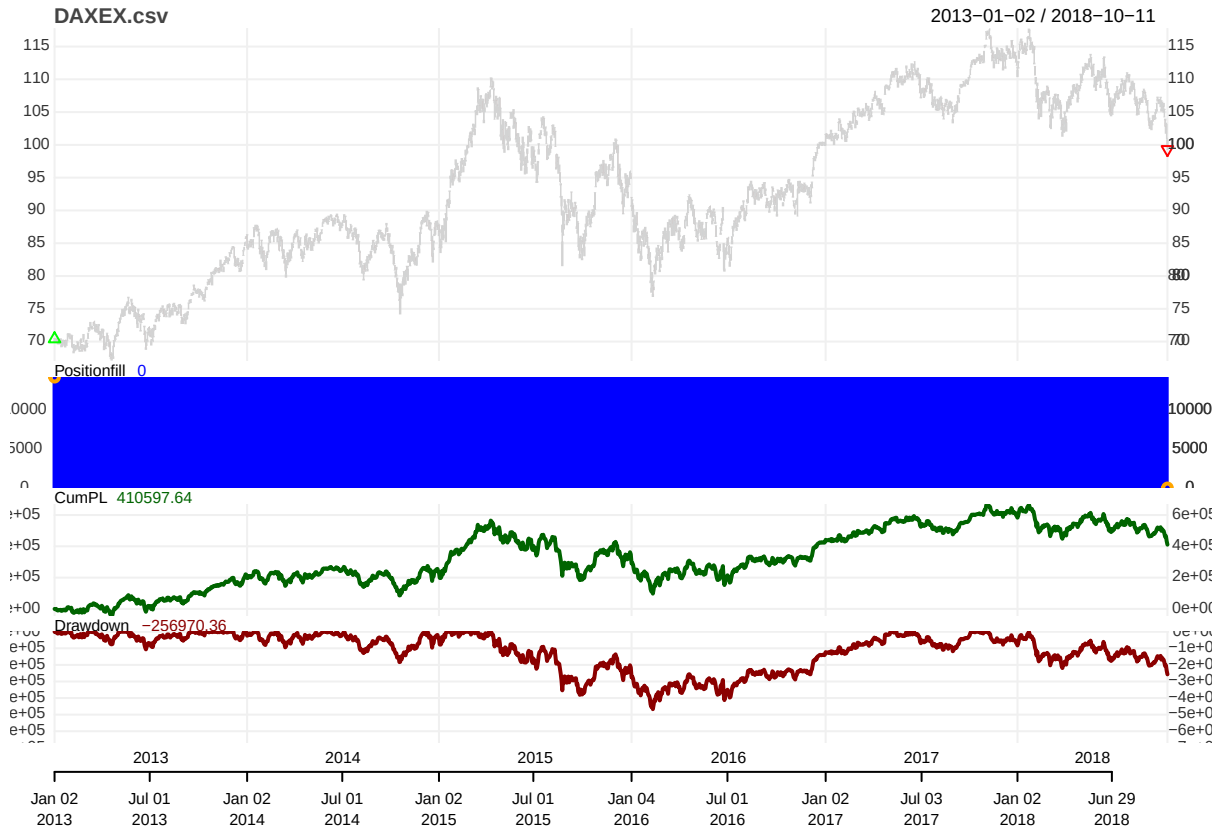
```
# We remove any objects, in case there was a buyhold portfolio initialized before
suppressWarnings(try(rm(list=c("account.buyhold","portfolio.buyhold"),pos=.blotter)))

# The Buy and hold symbol is loaded
LoadCourseFile(BuyHoldDirectory,BuyHoldInstrument,debugme=TRUE,dates=daterange)
# The Buy and hold instrument is initialized
stock(BuyHoldInstrument,currency="EUR")

BuyHoldSymbol<-get(BuyHoldInstrument)

# The portfolio and account "buyhold" is initialized
initPortf("buyhold",BuyHoldInstrument,initDate=initdate,currency="EUR")
initAcct("buyhold",portfolios="buyhold",initDate=initdate,initEq=startCapital,currency="EUR")

# The first date of the defined daterange is selected
currentdate <- first(time(BuyHoldSymbol))
```

```
# The close price at this date is selected
closeprice <- as.numeric(Cl(BuyHoldSymbol[currentdate,]))

# Calculate the unitsize we can buy with our startingcapital
unitsize <- as.numeric(trunc(startCapital/closeprice))

# Place the transaction for the instrument at the first date
addTxn("buyhold",Symbol=BuyHoldInstrument,TxnDate=currentdate,TxnPrice=closeprice,TxnQty=unitsize,Txn

# Select the last date of the daterange period
lastdate <-last(time(BuyHoldSymbol))

# Select the price at the last date
lastprice <- as.numeric(Cl(BuyHoldSymbol[lastdate,]))

# Sell the position at the last date of the daterange
addTxn("buyhold",Symbol=BuyHoldInstrument,TxnDate=lastdate,TxnPrice=lastprice,TxnQty=-unitsize,TxnFee

# update portfolio and account
updatePortf(Portfolio="buyhold")
updateAcct(name="buyhold")
updateEndEq(Account="buyhold")
```

### Step 2: Visualize the Buy and Hold strategy

We can see that we hold the position from the first until the last date. The cumulative profits are visualized by the green line.

```
chart.Posn("buyhold",Symbol=BuyHoldInstrument, theme=myTheme)
```




### Step 3: Compare the returns of the trading strategy with the buy and hold strategy

In order to compare the results of both strategies we calculat the returns for the buy and hold strategy and combine them with the returns of the trading strategy which were calculatet before.

```
rets.bh <- PortfReturns(Account='buyhold')
returns <- cbind(rets,rets.bh)
#rulecol <- paste(portfolioname,instrument,sep="-")
#colnames(returns) <- c(rulecol,"Buy-and-hold")
```

We compare the two strategies by showing some statistical metrics of the returns and plot the returns in one chart to directly compare the performance of the strategies.

```
table.Stats(returns)
```

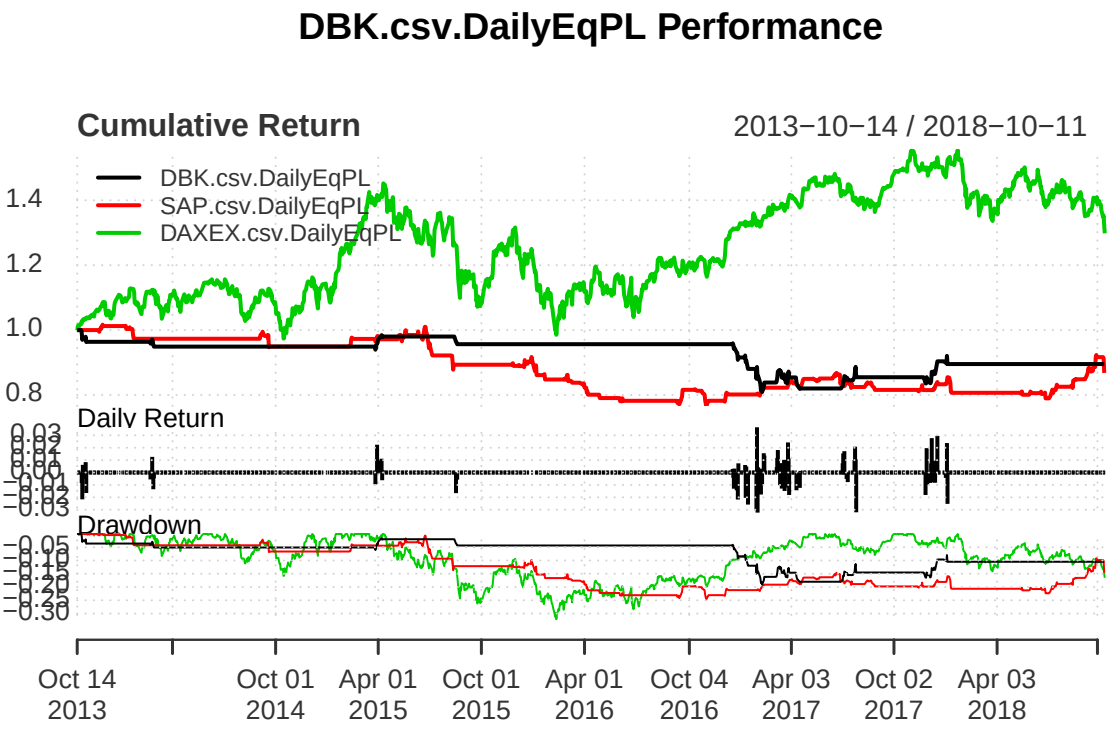
```
##                  DBK.csv.DailyEqPL SAP.csv.DailyEqPL DAXEX.csv.DailyEqPL
## Observations            1262.0000         1262.0000           1463.0000
## NAs                      201.0000          201.0000              0.0000
## Minimum                   -0.0323           -0.0448             -0.0857
## Quartile 1                 0.0000            0.0000             -0.0070
## Median                     0.0000            0.0000              0.0010
## Arithmetic Mean           -0.0001           -0.0001              0.0003
## Geometric Mean            -0.0001           -0.0001              0.0002
## Quartile 3                 0.0000            0.0000              0.0078
## Maximum                    0.0365            0.0349              0.0607
## SE Mean                    0.0001            0.0001              0.0004
## LCL Mean (0.95)           -0.0003           -0.0004             -0.0005
## UCL Mean (0.95)            0.0001            0.0002              0.0010
```

```
## Variance                   0.0000           0.0000            0.0002
## Stdev                      0.0038           0.0046            0.0143
## Skewness                  -0.3552          -1.5626           -0.3312
## Kurtosis                  34.9750          24.7729            2.0578
```

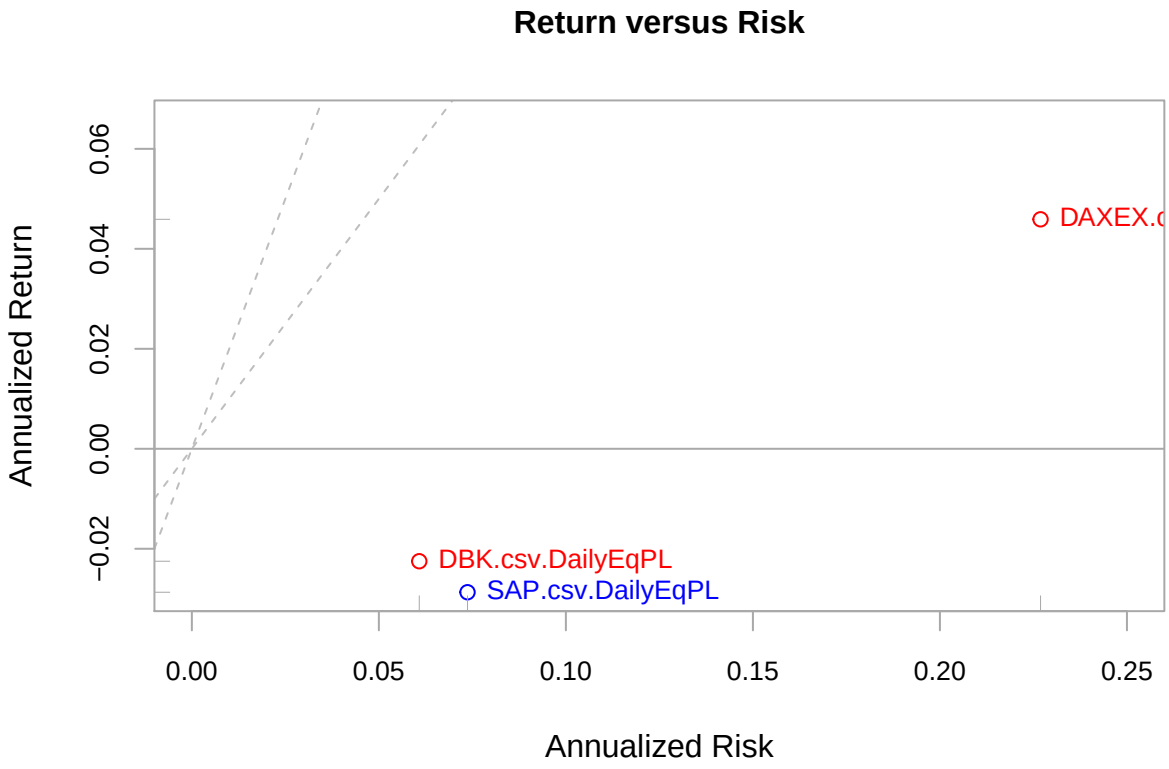
```
table.AnnualizedReturns(returns)
```

```
##                           DBK.csv.DailyEqPL SAP.csv.DailyEqPL
## Annualized Return                   -0.0225           -0.0287
## Annualized Std Dev                   0.0608            0.0737
## Annualized Sharpe (Rf=0%)           -0.3705           -0.3892
##                           DAXEX.csv.DailyEqPL
## Annualized Return                      0.0459
## Annualized Std Dev                     0.2269
## Annualized Sharpe (Rf=0%)              0.2024
```

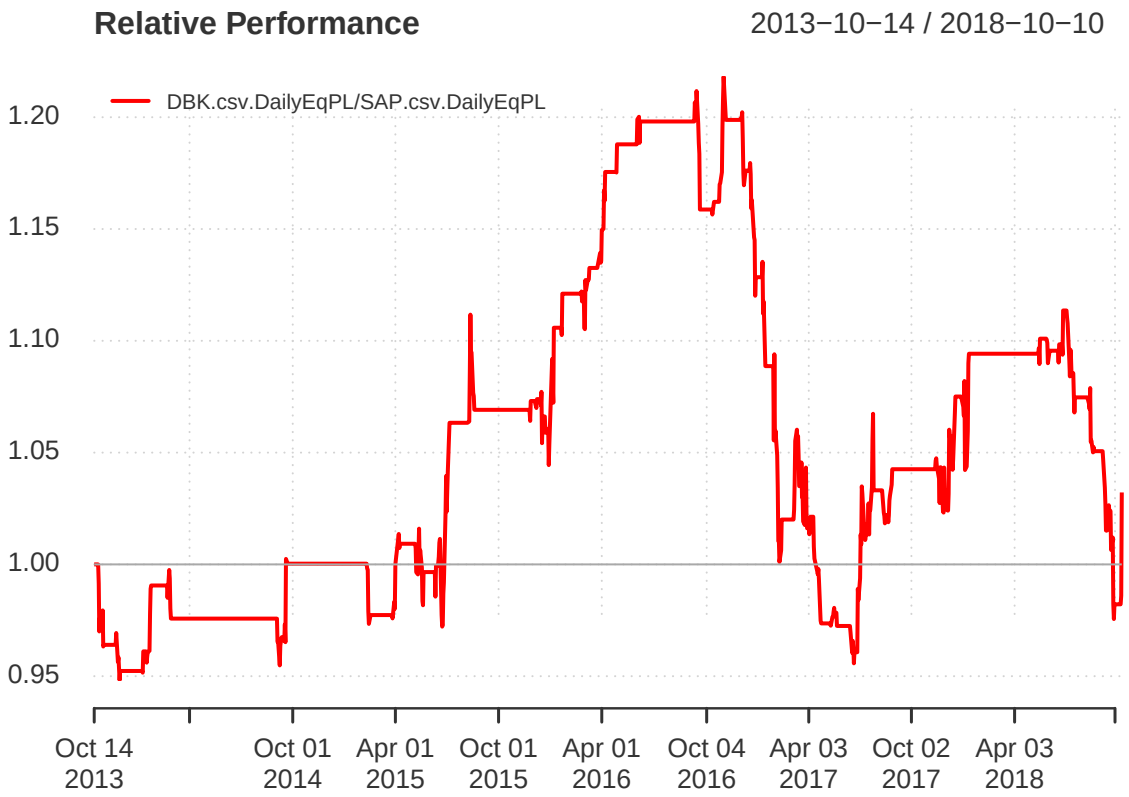
```
charts.PerformanceSummary(returns,geometric=FALSE,wealth.index=TRUE)
```




```
chart.RiskReturnScatter(returns,Rf=0,add.sharpe=c(1,2),xlim=c(0,0.25),main="Return versus Risk",colorse
```




```
chart.RelativePerformance(returns[,1],returns[,2],colorset=c("red","blue"),lwd=2,legend.loc="topleft")
```




**Historical VaR**

**Equity curve**

# Conclusion and Suggestions