Algorithmic Trading with the Relative Strength Index

Umurzakov Sodik

02/10/2020

Introduction

The Relative Strength Index (RSI) is one of the most popular indicators in the market.

The RSI is a basic measure of how well a stock is performing against itself by comparing the strength of the up days versus the down days. This number is computed and has a range between 0 and 100. A reading above 70 is considered bullish, while a reading below 30 is an indication of bearishness.

Investment in stock market is common scenario for making capital gains. One of the major concerns of today's investors is regarding choosing the right securities for investment, because selection of inappropriate securities may lead to losses being suffered by the investor. In order to reduce the risk of incurring losses and increase the return many tools are available, of which RSI is a powerful analytical tool which will help the investor choose the right combination of securities for their portfolio construction thus reducing the risk and increasing the return. RSI is developed J. Welles Wilder, the Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements. RSI is an extremely popular momentum indicator that has been featured in a number of articles, interviews and books over the years. RSI oscillates between zero and 100. Traditionally, and according to Wilder, RSI is considered overbought when above 70 and oversold when below 30. Signals can also be generated by looking for divergences, failure swings and center line crossovers. RSI can also be used to identify the general trend.

In my work I will try to make RSI trading strategy with stocks of Apple Inc. with the time interval from 2018.01.01 to 2020.09.04. Further, we will use the strategy from Bhargavi. R et al paper. In the following paper the author attempted to use this strategy for Indian companies and got some results for. The strategy was used correctly.

Link to the paper https://www.ripublication.com/ijaer17/ijaerv12n19_124.pdf

Main body of the report

Here we have to install and run some libraries in order to start working on building of algorithmic strategy. Below the list of libraries are attached to work on. Mostly, for filtration of data we use rugarch package in order to forecast our outcomes.

```r
library(xml2)
library(rvest)
library(pbapply)
library(TTR)
library(dygraphs)
library(lubridate)
library(zoo)
library(lattice)
library(quantmod)
library(rusquant)
library(tseries)
library(moments)
library(ggplot2)
library(repr)
library(timeSeries)
library(rugarch)
library(foreach)
library(PerformanceAnalytics)
library(blotter)
library(quantstrat)
library(FinancialInstrument)
library(pastecs)
```

Let's try to download data for stock of the Apple Company from 2010-01-01 to 2020-09-04. Then create new variable for the close price in order to find returns.

```r
# Downloading data and splitting it
AAPL_model <- quantmod::getSymbols("AAPL", scr = "yahoo", from = '2020-01-01', to = '2020-09-04', auto.a
AAPL_test <- quantmod::getSymbols("AAPL", scr = "yahoo", from = '2019-01-01', to = '2019-12-31', auto.as
AAPL_train <- quantmod::getSymbols("AAPL", scr = "yahoo", from = '2010-01-01', to = '2018-12-31', auto.a
AAPL_all <- quantmod::getSymbols("AAPL", scr = "yahoo", from = '2010-01-01', to = '2020-09-04', auto.ass

model_setAAPL = AAPL_model
test_setAAPL = AAPL_test
train_setAAPL = AAPL_train
all_setAAPL = AAPL_all
```

Next step is to run Augmented Dickey-Fuller Test. If a return series is mean reverting, then the current price level will tell us something about what the price's next move will be: If the price level is higher than the mean, the next move will be a downward move; if the price level is lower than the mean, the next move will be an upward move. The ADF test is designed to test for mean reversion.

Calculating returns for each sets and test for stationarity via ADF test in order to define it has unit root or not

```r
returnAAPL = Return.calculate(train_setAAPL$AAPL.Close)
returnAAPL_test = Return.calculate(test_setAAPL$AAPL.Close)
returnAAPL_model = Return.calculate(model_setAAPL$AAPL.Close)
returnAAPL_all = Return.calculate(all_setAAPL$AAPL.Close)
```

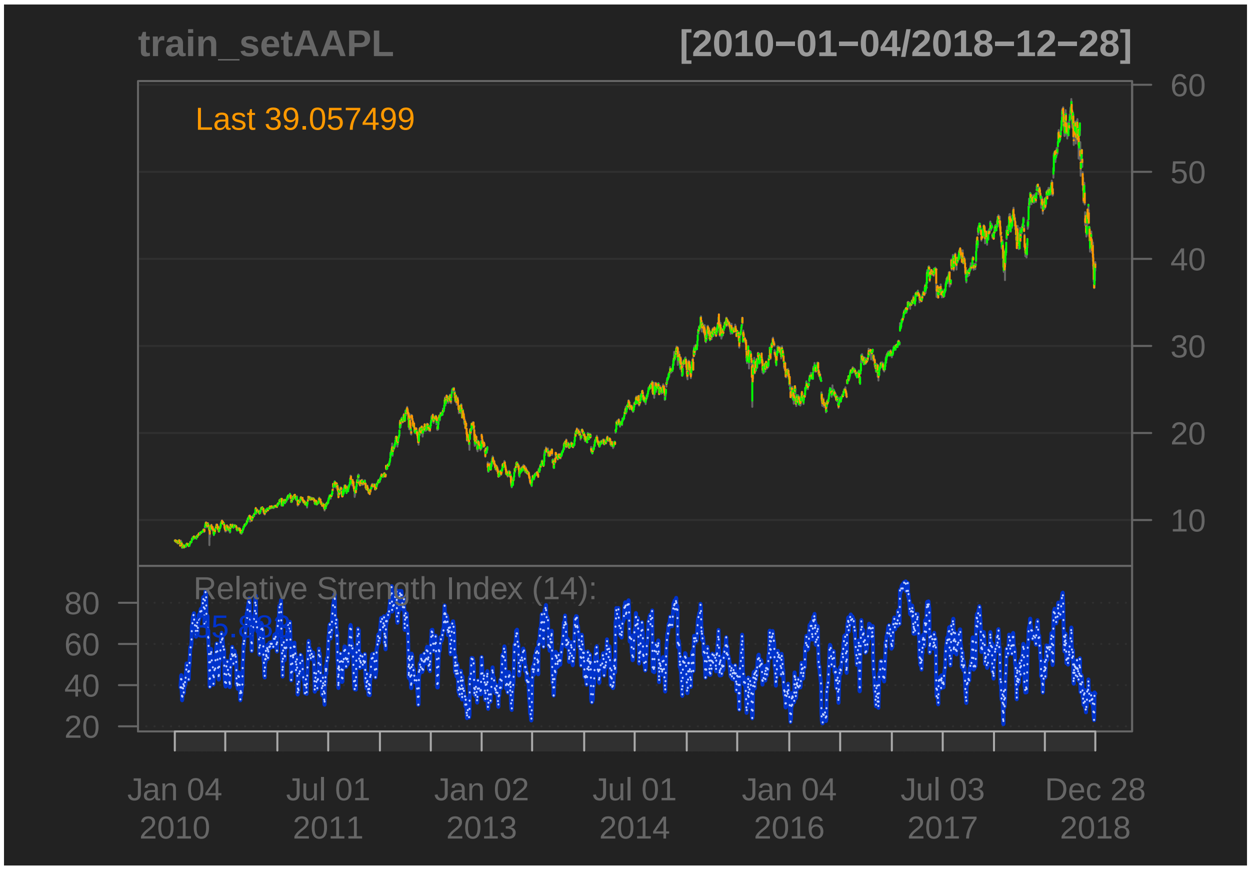
It has unit root and not stationary as Null hypothesis is not rejected because p-value is high. Then we have to work on in order to get the stationary process.

Here we try to filter out data of apple return on ARMA and GARCH models to eliminate residuals for all periods. (Please, follow rscript file for filtration)

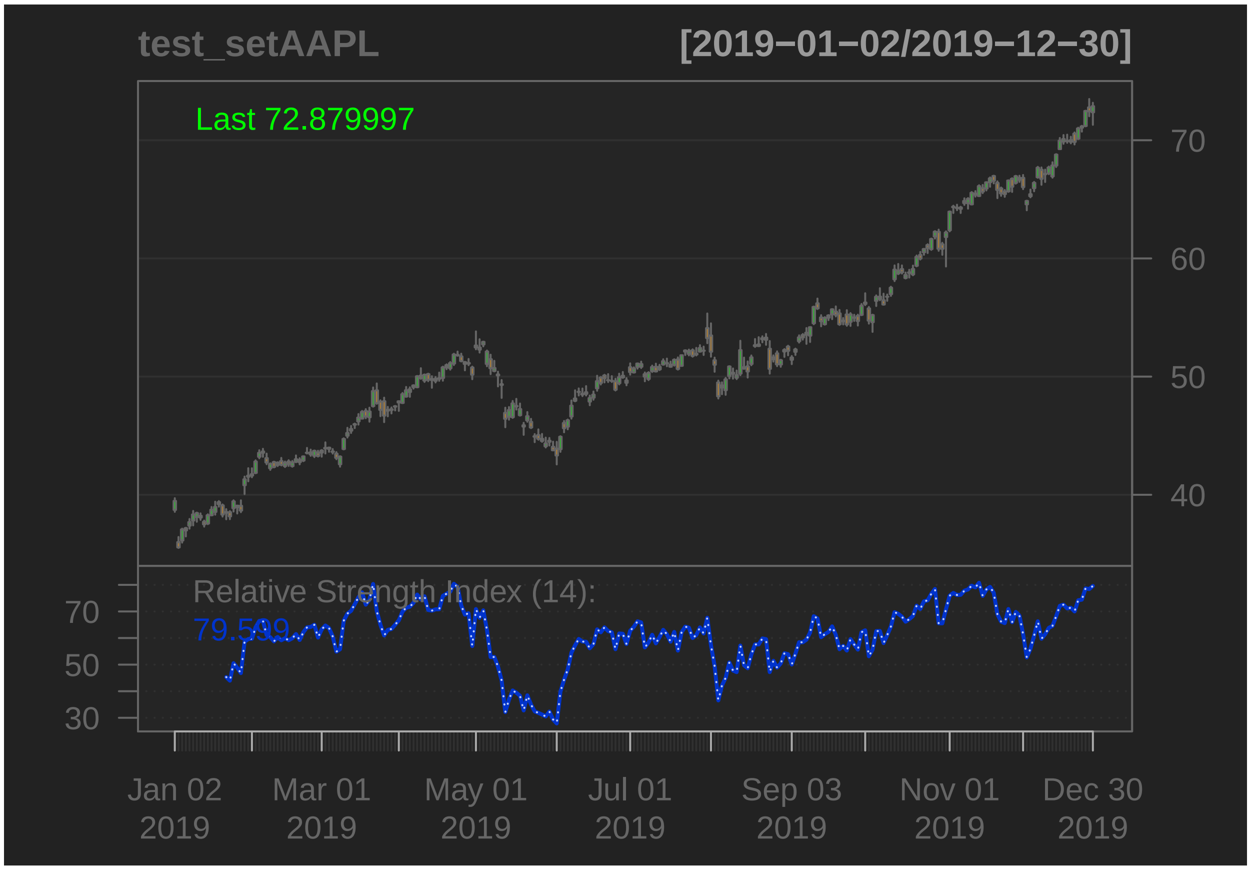
Calculate returns for stretegies

Now after filtration process we can start calculating of RSI and generating signals for all sets

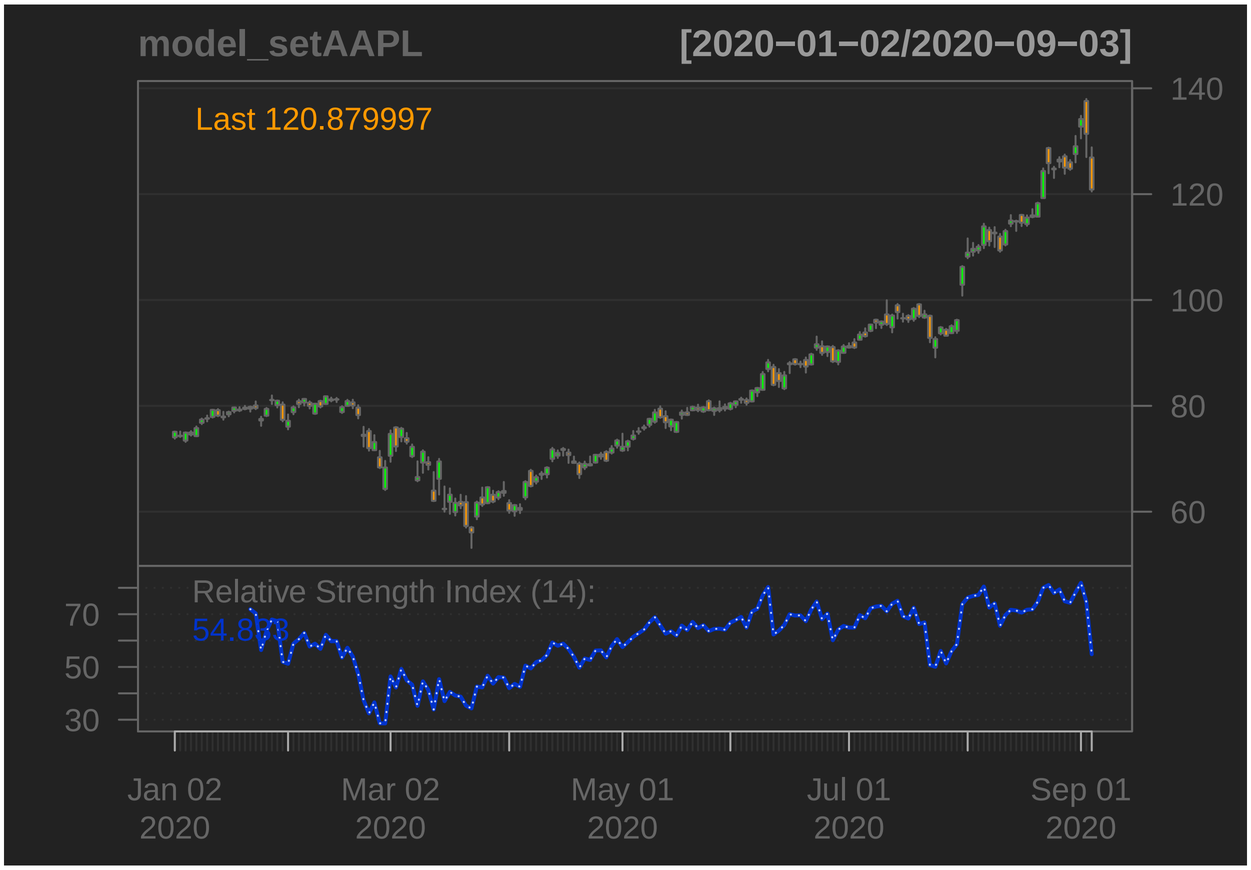
```r
RSI_train = RSI(train_setAAPL$AAPL.Close, maType = "EMA")
RSI_test = RSI(test_setAAPL$AAPL.Close, maType = "EMA")
RSI_model = RSI(model_setAAPL$AAPL.Close, maType = "EMA")

chartSeries(train_setAAPL, TA = "addRSI()")
```



```
chartSeries(test_setAAPL, TA = "addRSI()")
```



```
chartSeries(model_setAAPL, TA = "addRSI()")
```



```
# Here, we have to set treshholds for the signals
signal_train = Lag(ifelse(RSI_train$rsi > 85, -1, 1))
signal_test = Lag(ifelse(RSI_test$rsi > 85, -1, 1))
signal_model = Lag(ifelse(RSI_model$rsi > 85, -1, 1))
```

In this stage we have to calculate alpha and beta for AAPL test, model and training sets. Then, evaluate performance and compare them. After that we have to compute returns from our strategies

```
# Calculating returns from strategies
overall_train = lag(signal_train, 1) * returnAAPL
overall_test = lag(signal_test, 1) * returnAAPL_test
overall_model = lag(signal_model, 1) * returnAAPL_model


CAPM.alpha(overall_test, returnAAPL_test)
```

```
## [1] 0.0002352326
```

```
CAPM.alpha(overall_train, returnAAPL)
```

```
## [1] -3.031783e-05
```

```
CAPM.alpha(overall_model, returnAAPL_model)
```

```
## [1] 0.001650295
```

```
# Calculating beta only for training part
CAPM.beta(overall_train, returnAAPL)
```

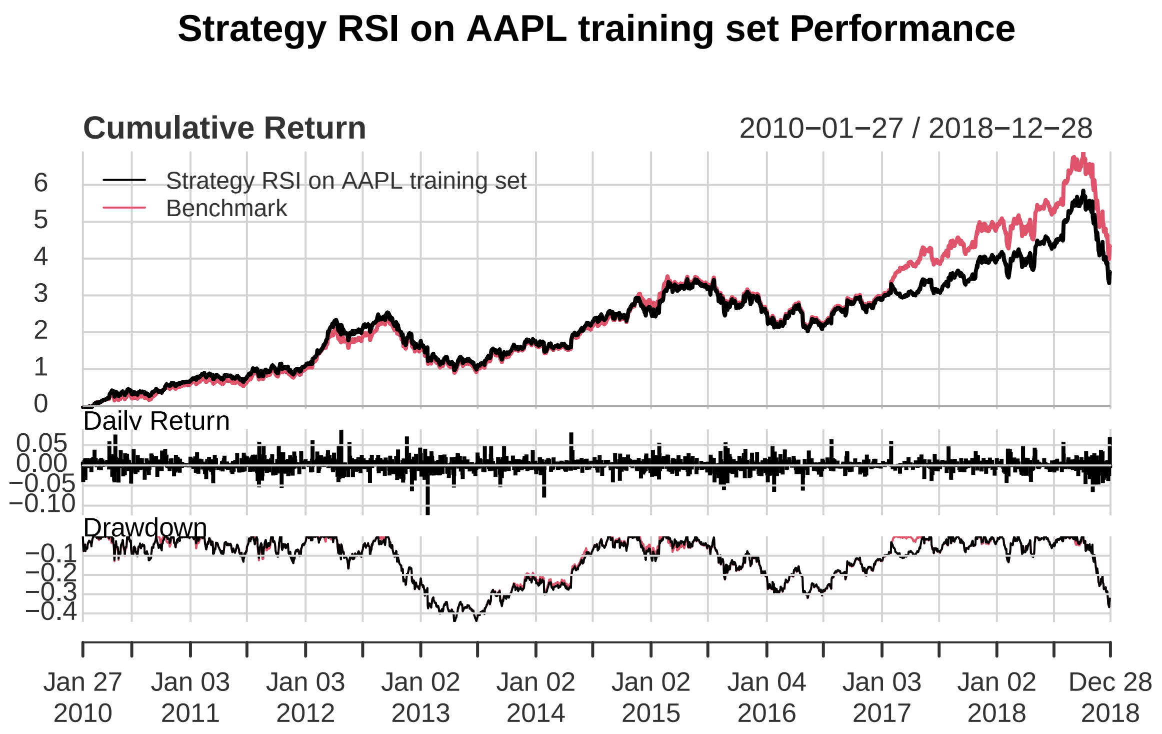
```
## [1] 0.962612
```

```r
compareAAPL <- na.omit(cbind(overall_train, returnAAPL))
colnames(compareAAPL) <- c("Strategy RSI on AAPL training set", "Benchmark")

compare_testAAPL <- na.omit(cbind(overall_test, returnAAPL_test))
colnames(compare_testAAPL) <- c("Strategy RSI on AAPL test set", "Benchmark")

compare_modelAAPL <- na.omit(cbind(overall_model, returnAAPL_model))
colnames(compare_modelAAPL) <- c("Strategy RSI on AAPL model set", "Benchmark")
```

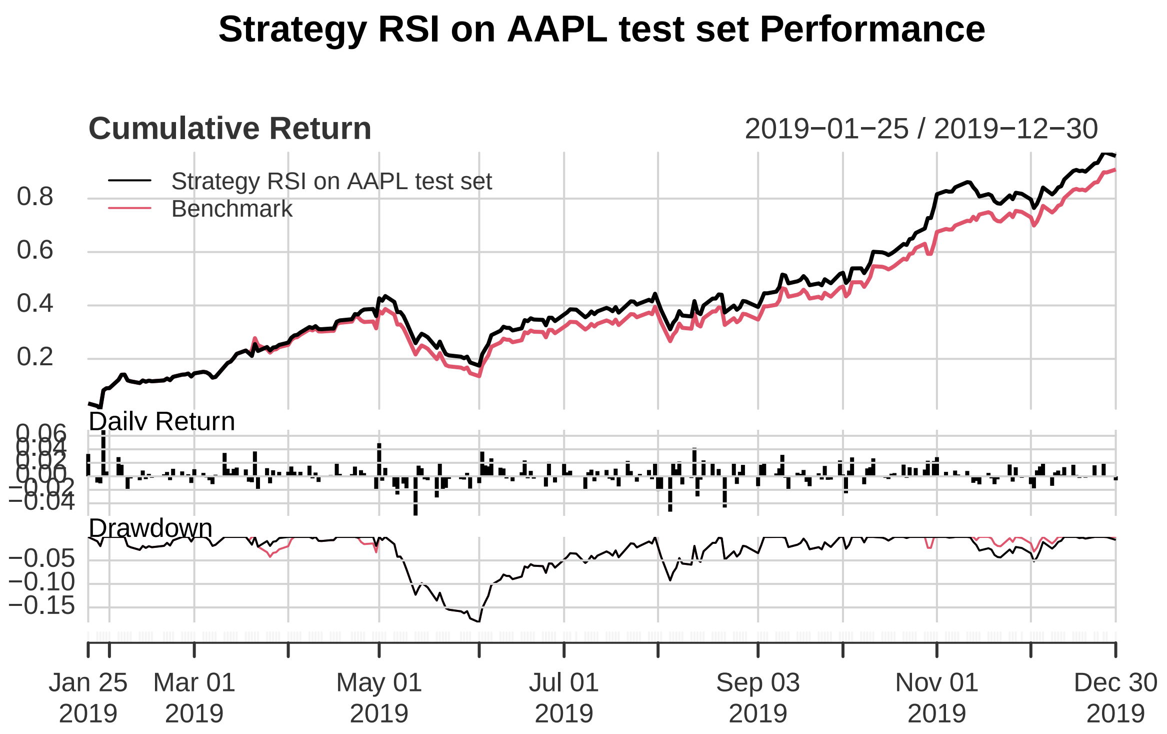
Here, we can see that the value of alpha is high as in the GARCH model filter

Plots for the results

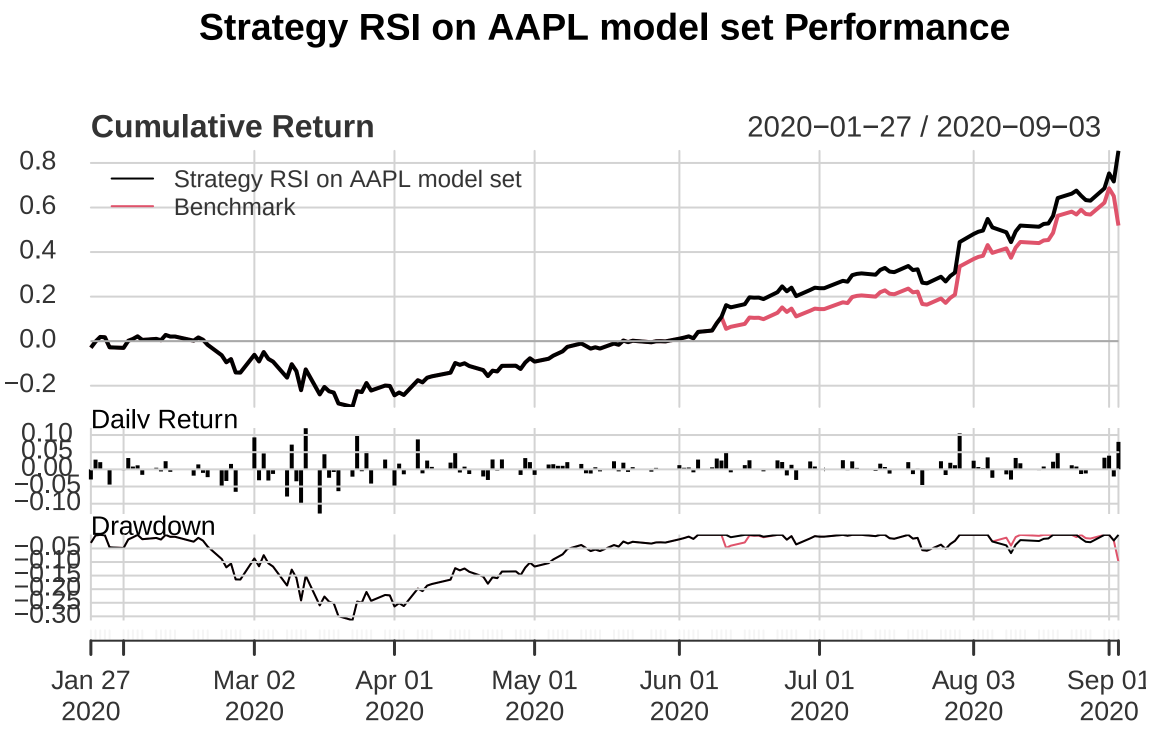
```r
charts.PerformanceSummary(compareAAPL)
```



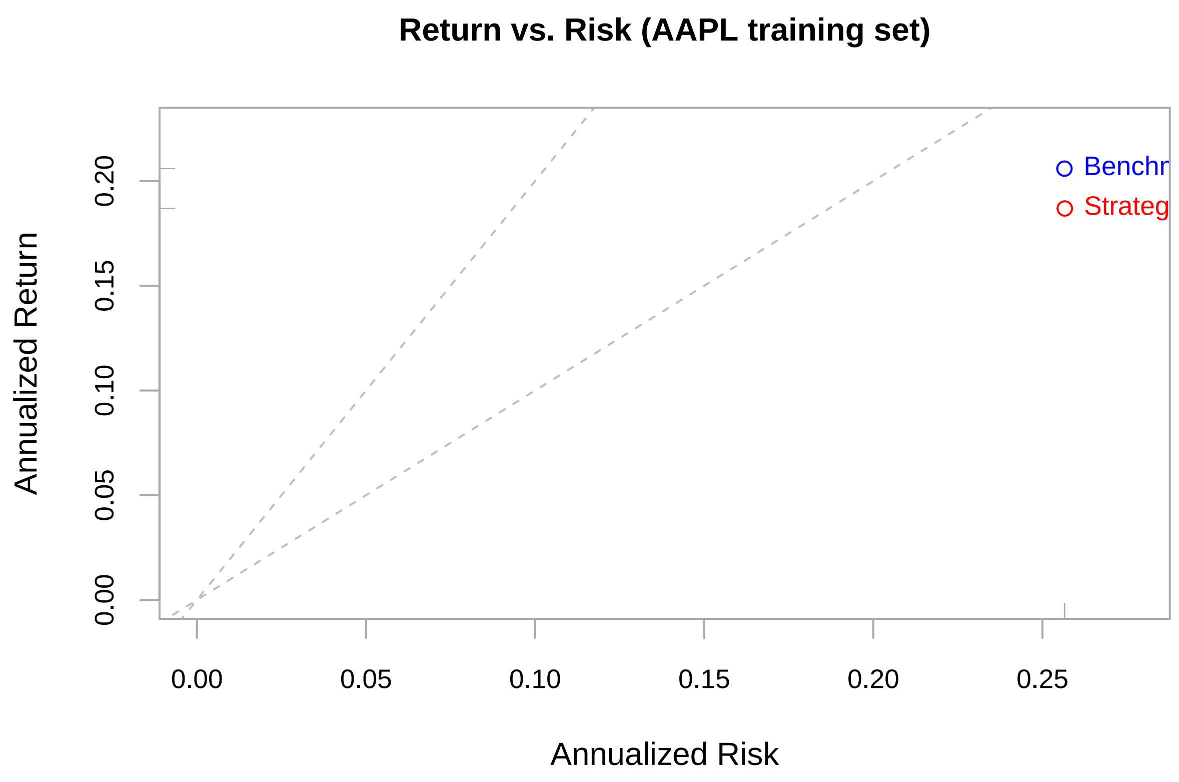
```r
charts.PerformanceSummary(compare_testAAPL)
```



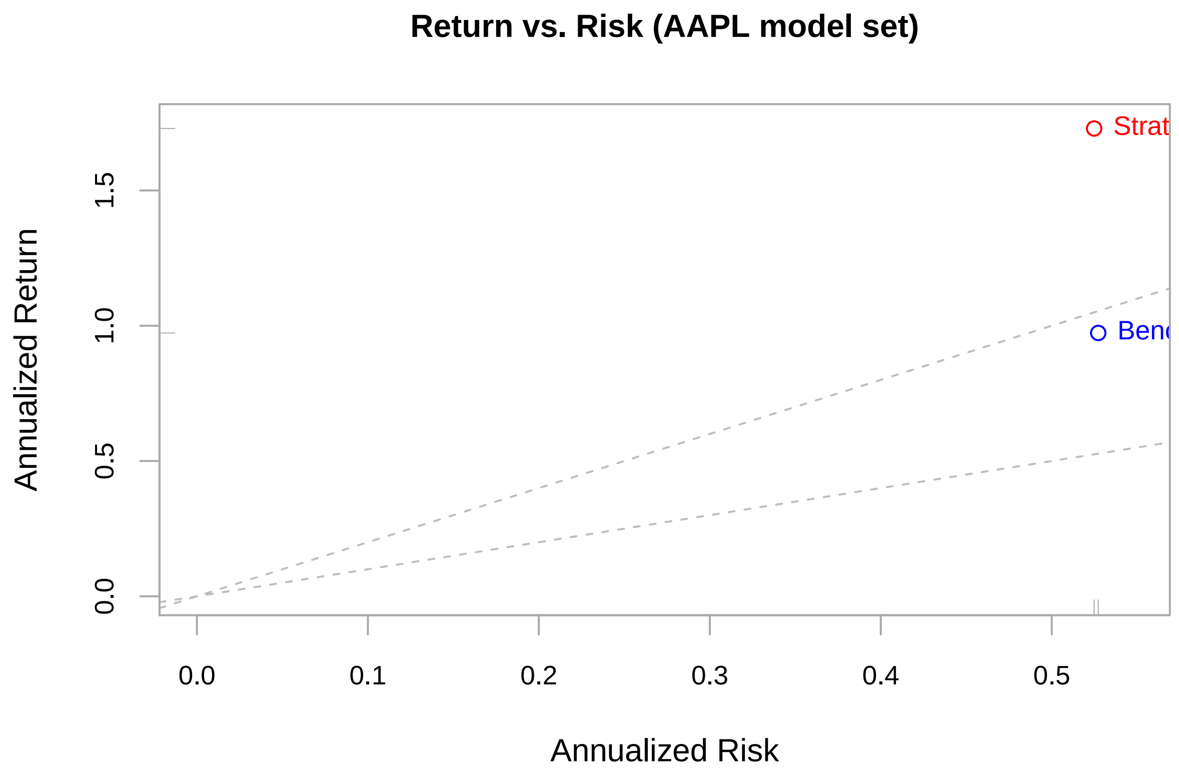
```
charts.PerformanceSummary(compare_modelAAPL)
```



```
chart.RiskReturnScatter(compareAAPL, Rf = 0, add.sharpe = c(1, 2),
                        main = "Return vs. Risk (AAPL training set)", colorset = c("red", "blue"))
```



```r
chart.RiskReturnScatter(compare_modelAAPL, Rf = 0, add.sharpe = c(1, 2),
                        main = "Return vs. Risk (AAPL model set)", colorset = c("red", "blue"))
```

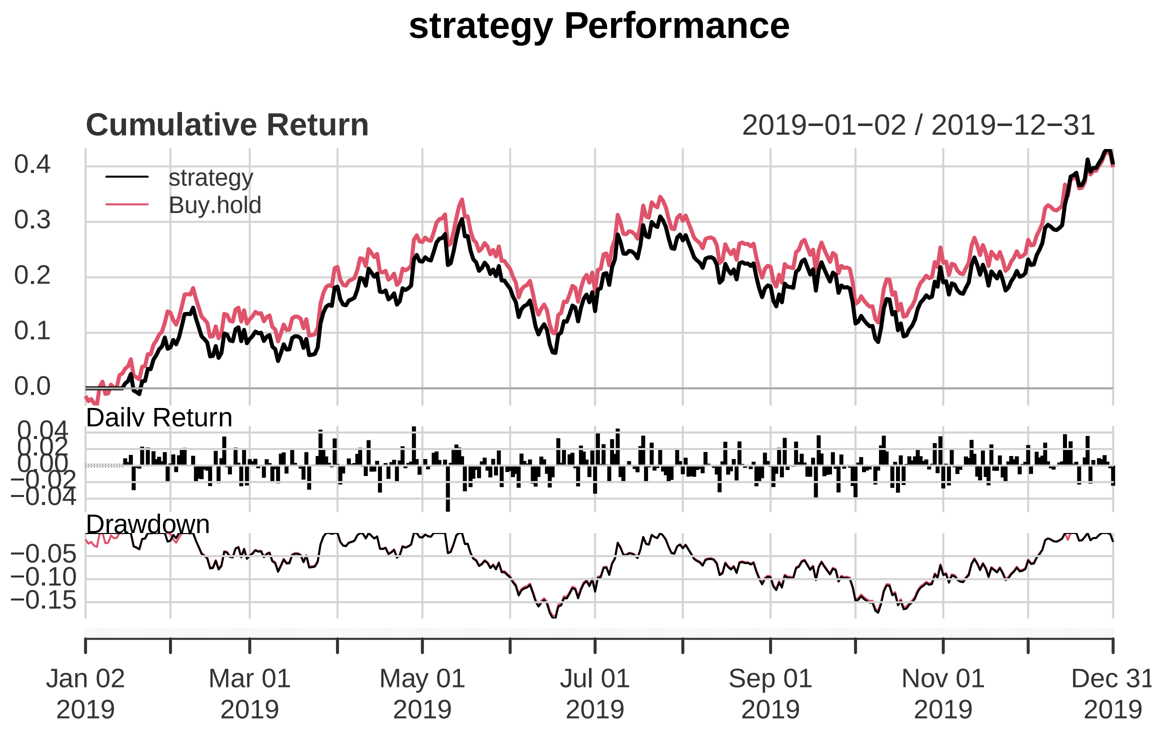


Monte-Carlo Simulation for our case

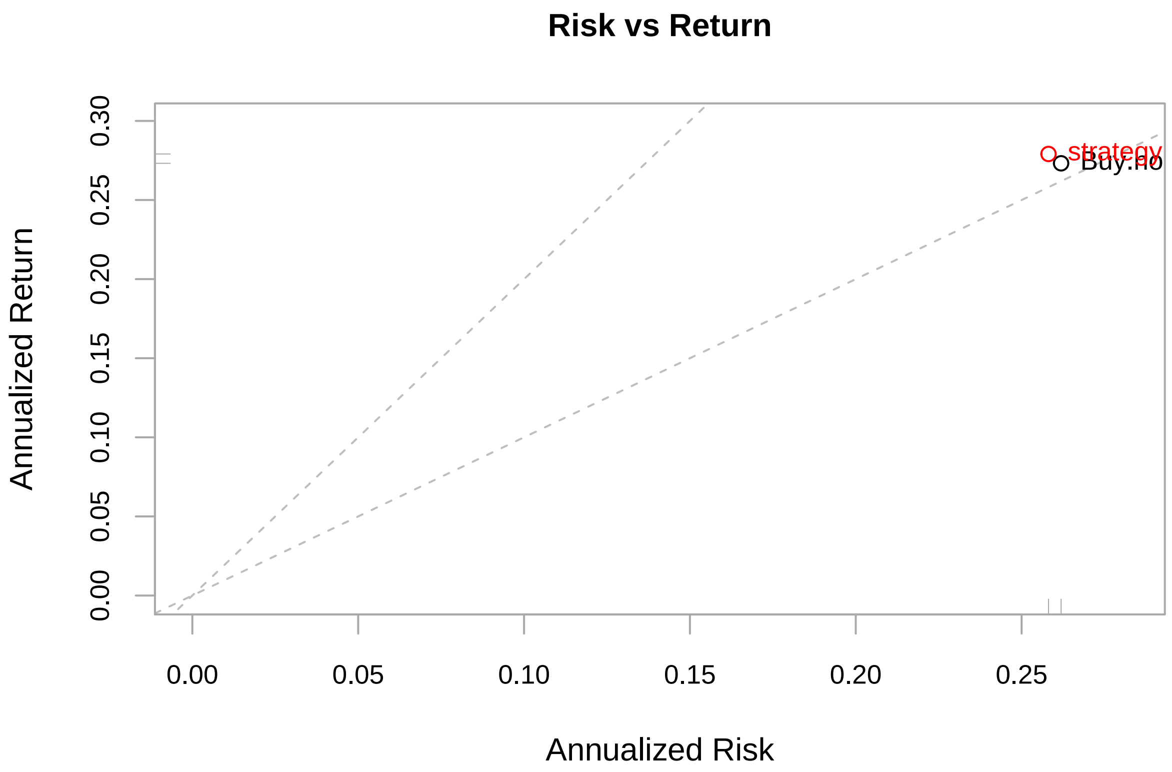
```r
daily<-Delt(test_setAAPL$AAPL.Close, type = c("log") )
stat.desc(daily)
```

```
##                 Delt.1.log
## nbr.val       2.500000e+02
## nbr.null      0.000000e+00
## nbr.na        1.000000e+00
## min          -1.049244e-01
## max           6.610106e-02
## range         1.710254e-01
## sum           6.130200e-01
## median        2.748754e-03
## mean          2.452080e-03
## SE.mean       1.053401e-03
## CI.mean.0.95  2.074711e-03
## var           2.774132e-04
## std.dev       1.665572e-02
## coef.var      6.792488e+00
```

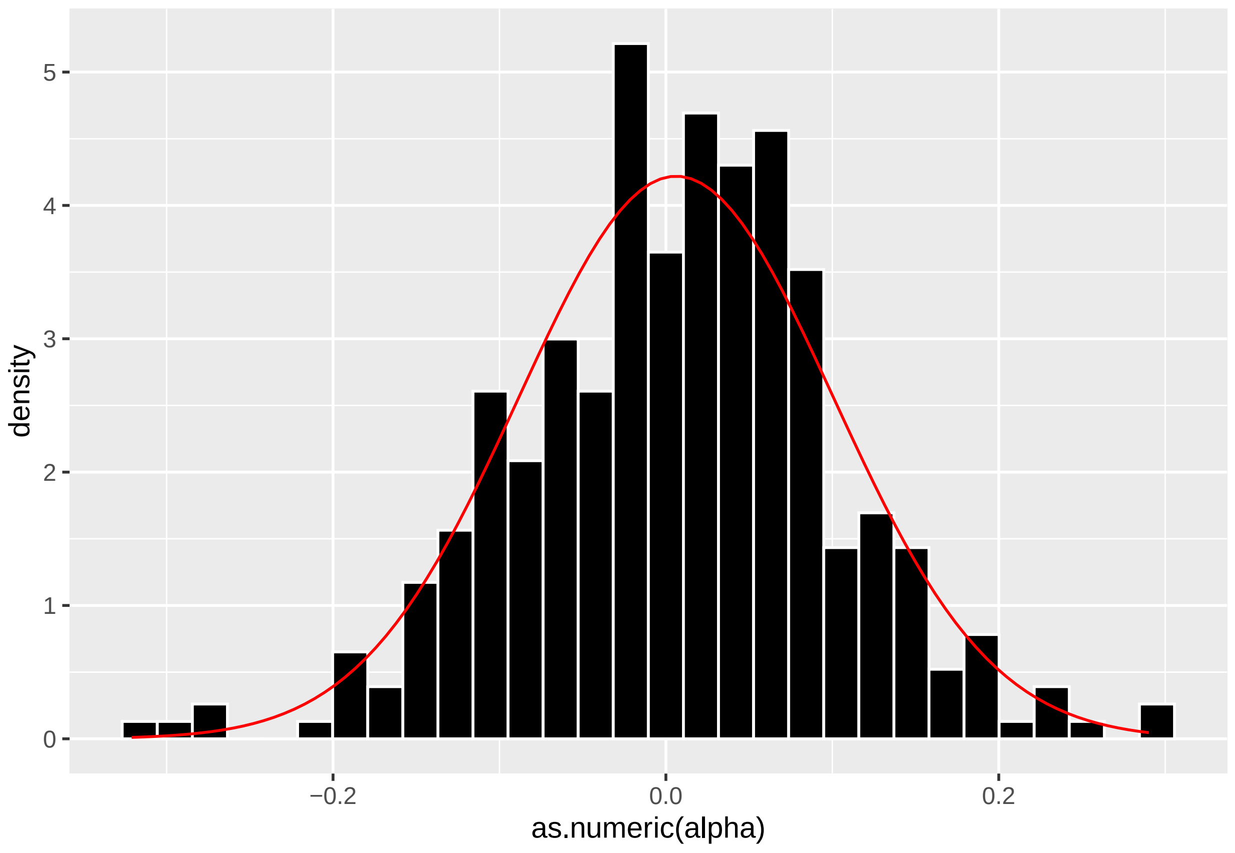
```r
p<-1
m<-365
alpha = numeric()
# trading days
while(p<m){
  stock_price <- 72.88 # last stock price
  stock_mu <- 2.452080e-03 # mean from stat.desc(daily)
  stock_sigma <-1.665572e-02 # SD from stat.desc(daily)
  i<-1
  n <- 365


  while(i <n) {
    epsilon <- runif(n=1, min=0, max=1) # random generated number
    # calculate stock price (using quantile function of normal distribution)
    stock_price[seq(i+1,i+1)] <- stock_price[seq(i,i)] * exp(qnorm(epsilon,
                                                                  stock_mu ,
                                                                  stock_sigma))
    i<-i+1
  }

  stock_price<-as.data.frame(stock_price)
  values = seq(from = as.Date("2019-01-01"), to = as.Date("2019-12-31"), by = 'day')
  values<-as.data.frame(values)
  values$price<-stock_price$stock_price
  colnames(values)[1]<-"Date"
  Date<-as.Date(as.character(values$Date),"%Y-%m-%d")
  values<-xts(values$price, order.by = as.Date((values$Date),"%Y-%m-%d"))


  colnames(values)[1]<-"Price"
  daily_return<-Return.calculate(values$Price)
  rsi <- RSI(values$Price,  maType = "EMA")

  signal <- NULL
  signal <-
    ifelse (rsi$rsi > 85 , -1, 1)


  signal[is.na(signal)] <- 0

  trade_return <- daily_return*lag(signal, na.pad = FALSE)
  cumm_return <- Return.cumulative(trade_return)
  annual_return <- as.ts(Return.annualized(trade_return))
  stat.desc(trade_return)



  summary(as.ts(trade_return))

  colnames(trade_return)[1]<-"strategy"

  alpha[seq(p,p)]<-CAPM.alpha(trade_return[,1,drop = FALSE],daily_return[,1,drop = FALSE])
  p<-p+1

}

alpha<-as.data.frame(alpha)

stock_prices.ret <- Return.calculate(values)
colnames(stock_prices.ret)<-c("Buy&hold")
charts.PerformanceSummary(cbind(trade_return$strategy,stock_prices.ret$`Buy&hold`), geometric=FALSE, wea
```



```
compare<-cbind(trade_return$strategy,stock_prices.ret$`Buy&hold`)
chart.RiskReturnScatter(compare, Rf = 0, add.sharpe = c(1, 2),
                        main = "Risk vs Return", colorset = c("red", "black"))
```



```
alpha = alpha*(252) #annualizing alpha
hist<-ggplot(alpha,aes(as.numeric(alpha)),binwidth=30)+geom_histogram(aes(y=..density..), fill = "Black
hist+stat_function(fun = dnorm, args = list(mean=mean(alpha$alpha, na.rm = TRUE),sd=sd(alpha$alpha,na.r
```



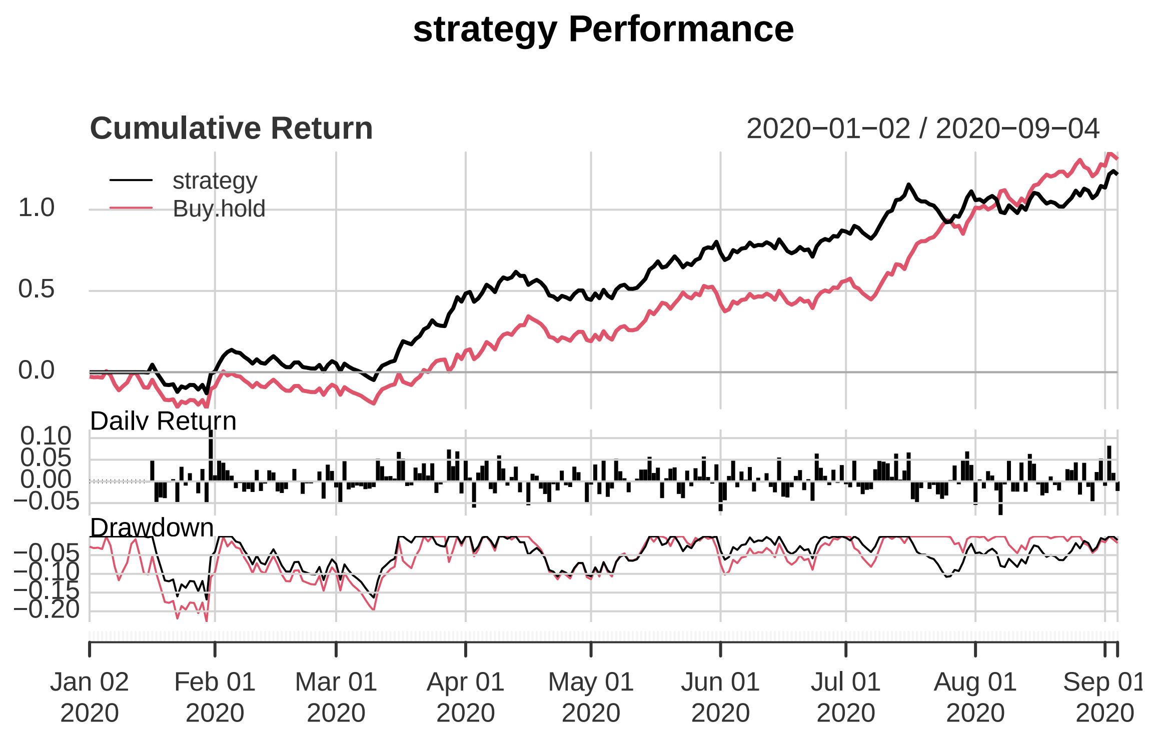
```
stat.desc(alpha)
```

```
##                     alpha
## nbr.val      364.000000000
## nbr.null       0.000000000
## nbr.na         0.000000000
## min           -0.321046225
## max            0.290162266
## range          0.611208491
## sum            2.231930654
## median         0.011782226
## mean           0.006131678
## SE.mean        0.004955872
## CI.mean.0.95   0.009745824
## var            0.008940082
## std.dev        0.094552007
## coef.var      15.420250887
```

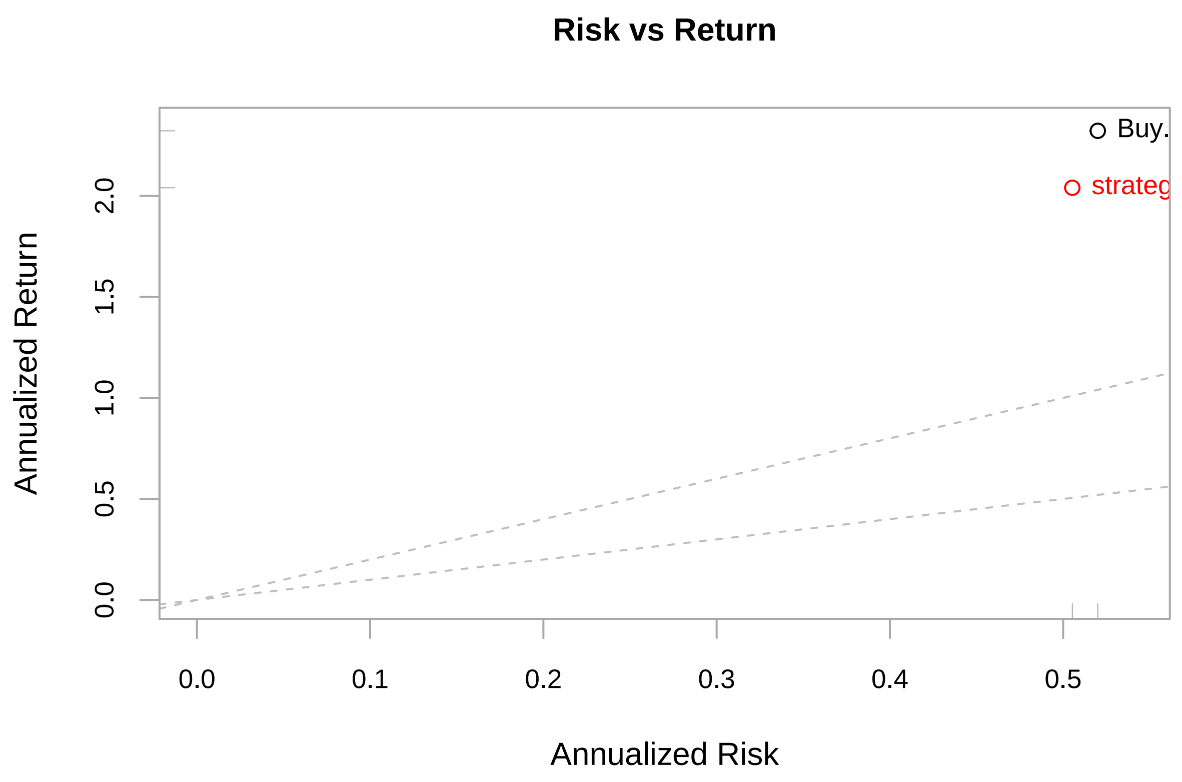
```
daily<-Delt(model_setAAPL$AAPL.Close, type = c("log") )
stat.desc(daily)
```

```
##                Delt.1.log
## nbr.val      170.000000000
## nbr.null       1.000000000
## nbr.na         1.000000000
## min           -0.137708043
## max            0.113157475
## range          0.250865517
## sum             0.476144166
## median          0.003590640
## mean            0.002800848
## SE.mean         0.002445064
## CI.mean.0.95    0.004826802
## var             0.001016317
## std.dev         0.031879732
## coef.var       11.382171411
```

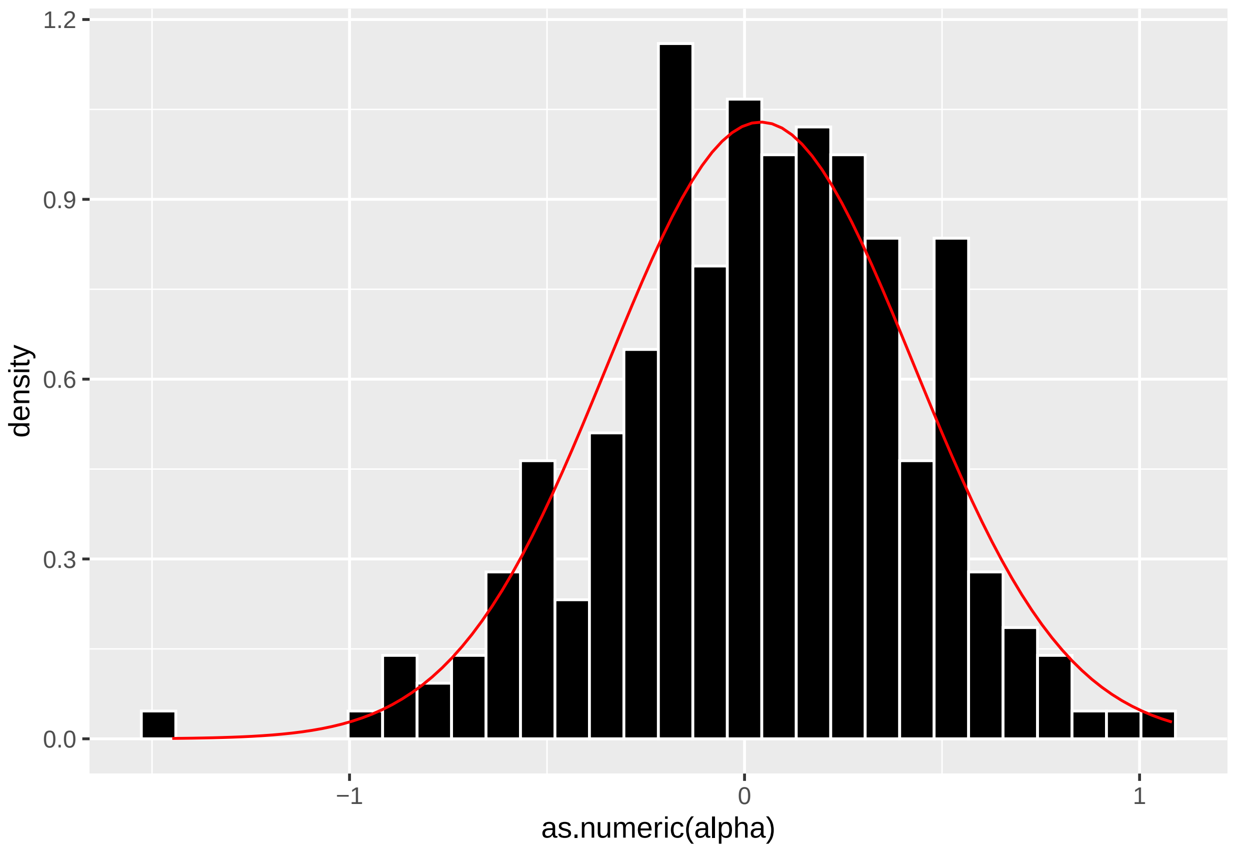
```
p<-1
m<-248
alpha = numeric()
# trading days
while(p<m){
  stock_price <- 120.88 # last stock price
  stock_mu <- 0.002800848 # mean from stat.desc(daily)
  stock_sigma <-0.031879732 # SD from stat.desc(daily)
  i<-1
  n <- 248


  while(i <n) {
    epsilon <- runif(n=1, min=0, max=1) # random generated number
    # calculate stock price (using quantile function of normal distribution)
    stock_price[seq(i+1,i+1)] <- stock_price[seq(i,i)] * exp(qnorm(epsilon,
                                                               stock_mu ,
                                                               stock_sigma))
    i<-i+1
  }

  stock_price<-as.data.frame(stock_price)
  values = seq(from = as.Date("2020-01-01"), to = as.Date("2020-09-04"), by = 'day')
  values<-as.data.frame(values)
  values$price<-stock_price$stock_price
  colnames(values)[1]<-"Date"
  Date<-as.Date(as.character(values$Date),"%Y-%m-%d")
  values<-xts(values$price, order.by = as.Date((values$Date),"%Y-%m-%d"))


  colnames(values)[1]<-"Price"
  daily_return<-Return.calculate(values$Price)


  rsi <- RSI(values$Price,  maType = "EMA")

  signal <- NULL
  signal <-
    ifelse (rsi$rsi > 70 , -1, 1)


  signal[is.na(signal)] <- 0

  trade_return <- daily_return*lag(signal, na.pad = FALSE)
  cumm_return <- Return.cumulative(trade_return)
  annual_return <- as.ts(Return.annualized(trade_return))
  stat.desc(trade_return)



  summary(as.ts(trade_return))

  colnames(trade_return)[1]<-"strategy"

  alpha[seq(p,p)]<-CAPM.alpha(trade_return[,1,drop = FALSE],daily_return[,1,drop = FALSE])
  p<-p+1

}

alpha<-as.data.frame(alpha)

stock_prices.ret <- Return.calculate(values)
colnames(stock_prices.ret)<-c("Buy&hold")
charts.PerformanceSummary(cbind(trade_return$strategy,stock_prices.ret$`Buy&hold`), geometric=FALSE, wea
```



```
compare<-cbind(trade_return$strategy,stock_prices.ret$`Buy&hold`)
chart.RiskReturnScatter(compare, Rf = 0, add.sharpe = c(1, 2),
                        main = "Risk vs Return", colorset = c("red", "black"))
```



```
alpha = alpha*(252) #annualizing alpha
hist<-ggplot(alpha,aes(as.numeric(alpha)),binwidth=30)+geom_histogram(aes(y=..density..), fill = "Black
hist+stat_function(fun = dnorm, args = list(mean=mean(alpha$alpha, na.rm = TRUE),sd=sd(alpha$alpha,na.rm
```



```
stat.desc(alpha)
```

```
##                     alpha
## nbr.val      247.00000000
## nbr.null       0.00000000
## nbr.na         0.00000000
## min           -1.44874238
## max            1.08185359
## range          2.53059597
## sum           10.07441920
## median         0.04914342
## mean           0.04078712
## SE.mean        0.02467079
## CI.mean.0.95   0.04859293
## var            0.15033608
## std.dev        0.38773196
## coef.var       9.50623488
```

Conclusion

From the results obtained, we can clearly find that RSI is one of the most effective technical analysis tools available, it can be effectively used to create a portfolio or just use for one instrument. Just as it performs well in other stock markets around the world.

In our case we can see that our strategy is good due to it is more useful than benchmark. Most parameters have significance for our strategy.