COMP226: Slides 20

Cross-validation

Rahul Savani

rahul.savani@liverpool.ac.uk

Overview

- **Cross-validation**: In-sample optimization, out-of-sample testing
- How to **pick parameter combinations** from in-sample results

Cross-validation

What can we try to do to ensure our results are **robust**? **Central question:** How will a strategy perform **in the future**?

- In statistics, cross-validation checks: How does a trained model generalize to an **unseen data set**?
- In our context of trading strategies, we want to know: **how well will our strategy perform in the future?**
- Thus we testing optimized strategy's on unseen, **future** data.
- Cross-validation mitigates the risk of **data-snooping bias**, which we discuss later

In-sample, out-of-sample test

Means **one round of cross validation**. Involves partitioning a sample of data into complementary subsets:

- optimize on one subset (the **training set**/**in-sample set**)
- validate on the other (the **test set**/**out-of sample set**)

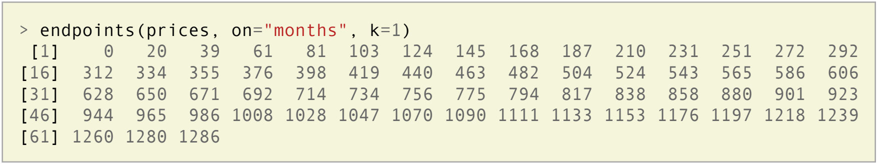
Why is it important?

- Protects against over-fitting
- Useful indicator of **future** performance

Subsetting xts objects by time

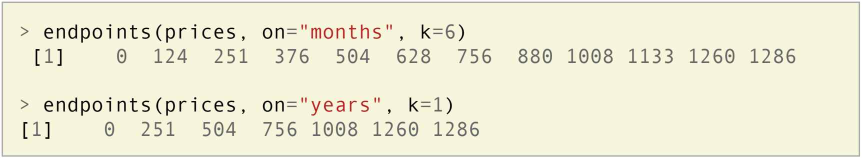
```
source('utilities.R'); prices <- getPrices()

> start(prices); end(prices)
[1] "2007-01-03 GMT"
[1] "2012-02-08 GMT"

> head(prices["2008"],n=1)
           Adjusted
2008-01-02  1447.16

> head(prices["2009-04"],n=1)
           Adjusted
2009-04-01   811.08

> head(prices["2009-03-04"])
           Adjusted
2009-03-04   712.87
```

Endpoints of time periods in xts



```
> endpoints(prices, on="months", k=1)
 [1]    0   20   39   61   81  103  124  145  168  187  210  231  251  272  292
[16]  312  334  355  376  398  419  440  463  482  504  524  543  565  586  606
[31]  628  650  671  692  714  734  756  775  794  817  838  858  880  901  923
[46]  944  965  986 1008 1028 1047 1070 1090 1111 1133 1153 1176 1197 1218 1239
[61] 1260 1280 1286
```

Consecutive endpoints differ by ~20 (trading days in a month)



```
> endpoints(prices, on="months", k=6)
 [1]    0  124  251  376  504  628  756  880 1008 1133 1260 1286

> endpoints(prices, on="years", k=1)
[1]    0  251  504  756 1008 1260 1286
```

Subset of endpoints above (and consecutive ones differ by ~120 for 6 months and ~250 for a year, which is roughly the number of trading days in a year)

Example

For simplicity we are going to use do a backtest with no slippage

It will use a simple rule using BBands that we have seen before

```r
# BBands strategy
getPos <- function(prices,n,sd) {
    bbands  <- BBands(prices,n=n,sd=sd)
    long    <- ifelse(prices<bbands$dn,1,0)
    short   <- ifelse(prices>bbands$up,-1,0)
    pos     <- long + short
    pos     <- lag(pos)
    pos[is.na(pos)] <- 0
    return(pos)
}
```

run: returns, fitness, equity curve

```r
run <- function(prices,param1,param2) {
    pos <- getPos(prices,param1,param2)
    log_rets <- getLogReturn(prices)
    eq  <- getEquityCurve(log_rets,pos)
    fit <- as.numeric(last(eq))
    lst <- list(log_rets=log_rets,equity=eq,fitness=fit)
    return(lst)
}
```

backtest

```r
# Backtester using apply for 2-parameter strategies
backtest <- function(prices,params) {
    results <- apply(params,1,
                function(x) run(prices,x[1],x[2])$fitness)
    results <- cbind(params,results)
    colnames(results)[length(results)] <- "fitness"
    return(results)
}
```

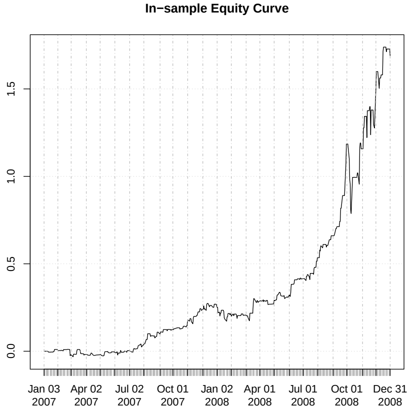
in-out-test.R

```r
in_out_test <-
    function(prices,startIn=1,endIn,startOut=endIn+1,endOut=nrow(prices)) {

    pricesIn   <- prices[startIn:endIn,] # in-sample period
    pricesOut  <- prices[startOut:endOut,] # out-of-sample period
    pricesBoth <- prices[startIn:endOut,] # both in and out

    # get fitness on in-sample period for each param combination
    results <- backtest(pricesIn,params)
    # get param combination that gives best fitness
    best <- results[which.max(results$fitness),1:2]

    lst <- list(inSample=run(pricesIn,best[,1],best[,2]),
                outSample=run(pricesOut,best[,1],best[,2]),
                both=run(pricesBoth,best[,1],best[,2]),
                best=best)
    return(lst)
}
```

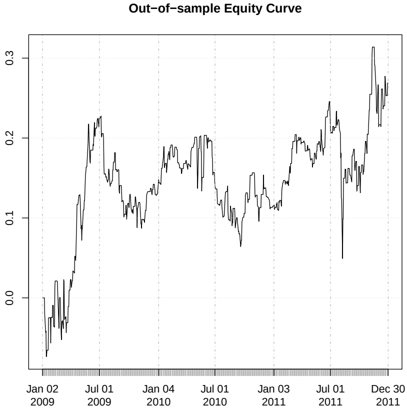
main.R

```
library(quantmod); library(PerformanceAnalytics); source('../../utilities.R');
source('../functions/bbands.R'); source('../functions/run.R');
source('../functions/in-out-test.R'); source('../functions/backtest.R')

# Define parameter ranges for BBands strategy
n  <- seq(5,by=5,to=10); sd <- seq(0.5,by=0.5,to=1.5)
params <- expand.grid(n=n,sd=sd) # all combinations

prices <- getPrices(readCsvData('../../GSPC.csv'))
ep <- endpoints(prices,on='years')
# ep: 0  251  504  756 1008 1260 1286

ret <- in_out_test(prices,startIn=1,endIn=ep[3],endOut=ep[6])

source('../functions/plotInOut.R') # plotting
pdf('pdf/in.pdf'); plot(ret$inSample$equity,main="In-sample");
dev.off()
pdf('pdf/out.pdf'); plot(ret$outSample$equity,main="Out-of-sample");
dev.off()
pdf('pdf/both.pdf'); plotInOut(prices,ret$both,ret$best,endIn=ep[3]);
dev.off()
```

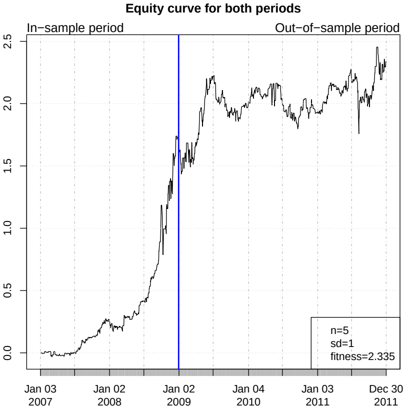
In-sample results



Out-sample results



Both together



Degradation in performance out of sample is common

Recap: one round cross-validation

- **In-sample optimization**

  1. Investigate fitness landscape e.g. via a grid search
  2. Choose parameters

- **Out-of-sample testing**

  1. Apply chosen parameters to unseen data
  2. Evaluate results

QUESTION: How should we choose parameters based on in-sample results?

Example: RSI strategy

- RSI stands for **Relative Strength Index**

- RSI is a standard indicator, implemented in the TTR package; you do not need to know its definition

- It takes values between 0 and 100, and higher (lower) values indicate that the market has recently been rising (falling)

- Let's look at an example **mean-reversion strategy**, like the one we have already seen that used Bollinger band, but using the RSI indicator

Example: RSI strategy

```r
# RSI strategy
getPos <- function(prices,n,thresh) {
    rsi     <- RSI(prices,n=n)
    long    <- ifelse(rsi<50-thresh,1,0)
    short   <- ifelse(rsi>50+thresh,-1,0)
    pos     <- long + short
    pos     <- lag(pos)
    pos[is.na(pos)] <- 0
    return(pos)
}
```

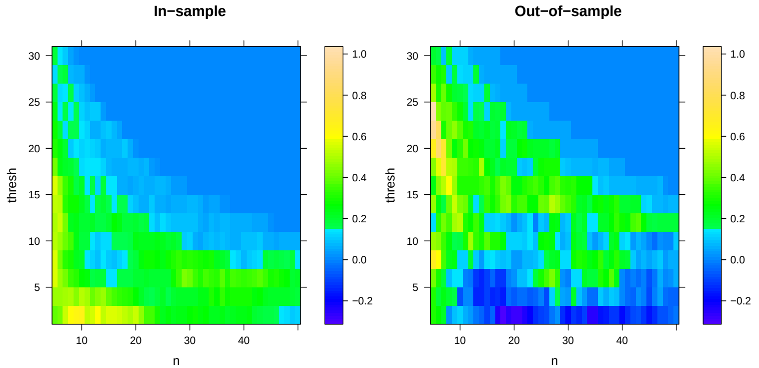
Plotting fitness landscape (1/2)

```
source('../../utilities.R'); source('../functions/run.R')
source('../functions/rsi.R'); source('../functions/backtest.R')
library(quantmod); library(PerformanceAnalytics)
library(lattice) # needed for levelplot
library(gridExtra) # used to arrange levelplots in a grid

# Define parameter ranges for RSI
n  <- seq(5,by=1,to=50); thresh <- seq(2,by=2,to=30)
params <- expand.grid(n=n,thresh=thresh) # all combinations
prices <- getPrices(readCsvData('../../GSPC.csv'))
ep <- endpoints(prices,on='months')

startIn  <- 1       ; endIn  <- ep[21]
startOut <- ep[21]+1; endOut <- ep[63]

resultsIn  <- backtest(prices[startIn:endIn,],params)
resultsOut <- backtest(prices[startOut:endOut,],params)
```

Plotting fitness landscape (2/2)

```r
fRange <- range(c(resultsIn$fitness,resultsOut$fitness))
# breakpoint for fitness contours
at <- seq(from=fRange[1],to=fRange[2],length.out=100)

plot1 <- levelplot(fitness ~ n * thresh,resultsIn,
                   at=at, main="In-sample",
                   col.regions=topo.colors(100))

plot2 <- levelplot(fitness ~ n * thresh,resultsOut,
                   at=at, main="Out-of-sample",
                   col.regions=topo.colors(100))

pdf('pdf/landscapes.pdf', width=10, height=5)
grid.arrange(plot1,plot2,ncol=2); dev.off()
```

Resulting plots



- Not always easy to pick parameters from in-sample results
- **Dataset drift** can cause the good parameter combination to differ between in-sample and out-of-sample periods

How to pick parameters

1. Can pick **multiple parameter combinations** (often a luxury one can't afford)

2. You have to pick a **single parameter combination** (due to trading constraints (both financial and due to market impact)

Picking multiple parameter combinations:

- Pick best k; Best p %; All above a certain fitness threshold

Picking a unique parameter combination:

- Best result; Average of best results

- Region analysis for in-sample fitness landscape (e.g. take the center of a fit region)