Description

Danusio Guimarães

10/12/2021

Contents

1 General Process Description

The operation is based on choosing a strike for the chosen option (put or call) so that the bidder is not exercised. The design parameter is the percentage of times the launcher wants the operation to be successful.

The two operations worked on are the put sale and the covered call (for the latter, whether or not you have the stock beforehand). Once you determine the desired level of success in the trade, you choose a quantile of the variation over a period equal to the time to maturity based on the following rule:

$$q_{put} = 1 - \alpha$$

$$q_{call} = \alpha \ ,$$

where q is the quantile and α is the percentage of success - a kind of *confidence level* for the operation. For example, if you want to be successful in 90% of trades, and the option expires in 20 trading sessions (time is counted in trading sessions, not business days), the put strike will be the 10% quantile of the historical stock price changes over 20 trading sessions, multiplied by the current stock price.

2 Libraries

```
library(dplyr)
library(BatchGetSymbols)
library(future)
library(ggplot2)
```

3 Backtest Example

This section contains an example of the process for the ETF **BOVA11**, with a 5-year backtest.

3.1 Loading Data

```
stock="BOVA11.SA"
```

```
t0=Sys.time()

batch=BatchGetSymbols(stock,
                      first.date = Sys.Date()-10*365.25,
                      last.date = Sys.Date()-1,
                      do.complete.data = T,
                      do.fill.missing.prices = T,
                      do.cache = F,
                      be.quiet = T)
```

```
>>>  [1] "BOVA11.SA"
```

```
t1=Sys.time()
```

```
t1-t0
```

```
>>>  Time difference of 3.477997 secs
```

```
prices=batch$df.tickers |>
  select(
```

```
    contains("price."),
    ref.date
  )
rm(batch)
```

```
prices |> head()
```

```
>>>  # A tibble: 6 x 6
>>>    price.open price.high price.low price.close price.adjusted ref.date
>>>         <dbl>      <dbl>     <dbl>       <dbl>          <dbl> <date>
>>>  1       56.5       57.2      55.8        57.0           57.0 2012-01-02
>>>  2       57.2       58.4      57.1        58.4           58.4 2012-01-03
>>>  3       57.7       58.6      57.7        58.4           58.4 2012-01-04
>>>  4       58.1       58.5      57.2        57.6           57.6 2012-01-05
>>>  5       57.8       58.4      57.5        57.8           57.8 2012-01-06
>>>  6       58.2       58.3      57.8        58.2           58.2 2012-01-09
```

3.2 Strategy Application

Based on quantiles of price variation.

- Sell put: low quantile;
- Covered Call: high quantile.

The required success level is 95%.

```
conf=0.95

# quantiles
qput=1-conf
qcall=conf
```

Backtest time interval - 5 years - and time to expiration - 15 trading sessions:

```
max_date=max(prices$ref.date)
ini_date=max_date-5*365.25

ini_pos=which.min(abs(prices$ref.date-ini_date))
N=nrow(prices)
int_val=ini_pos:N

n=15 # nº of trading floors until expiration
```

Price variations in 15 trading sessions - historical data:

```
plan(multicore)
calc_ret=future({
  TTR::ROC(prices$price.adjusted,n,"discrete")
})
```

```
price_var=value(calc_ret)
```

Calculation of the quantiles for the 5-year test interval:

```
plan(multicore)
calc_var=future({
  put_var <- call_var <- NULL
  for (i in int_val) {
```

```
    put_var=c(put_var,
              quantile(price_var[1:(i-1)],qput,names = F,na.rm = T))
    call_var=c(call_var,
               quantile(price_var[1:(i-1)],qcall,names = F,na.rm = T))
  }

  aim_var=cbind(
    put=put_var,
    call=call_var
  )
})
```

```
aim_var=value(calc_var)

# Actual stock price after 15 trading sessions, for all the trading sessions within the test range:
actual_var=price_var[int_val+n-1]
```

```
perf1=cbind(
  aim_var[,"put"]<actual_var,
  aim_var[,"call"]>actual_var
) |> colMeans(na.rm = T)
```

```
perf1
```

```
>>>  [1] 0.9609756 0.9520325
```

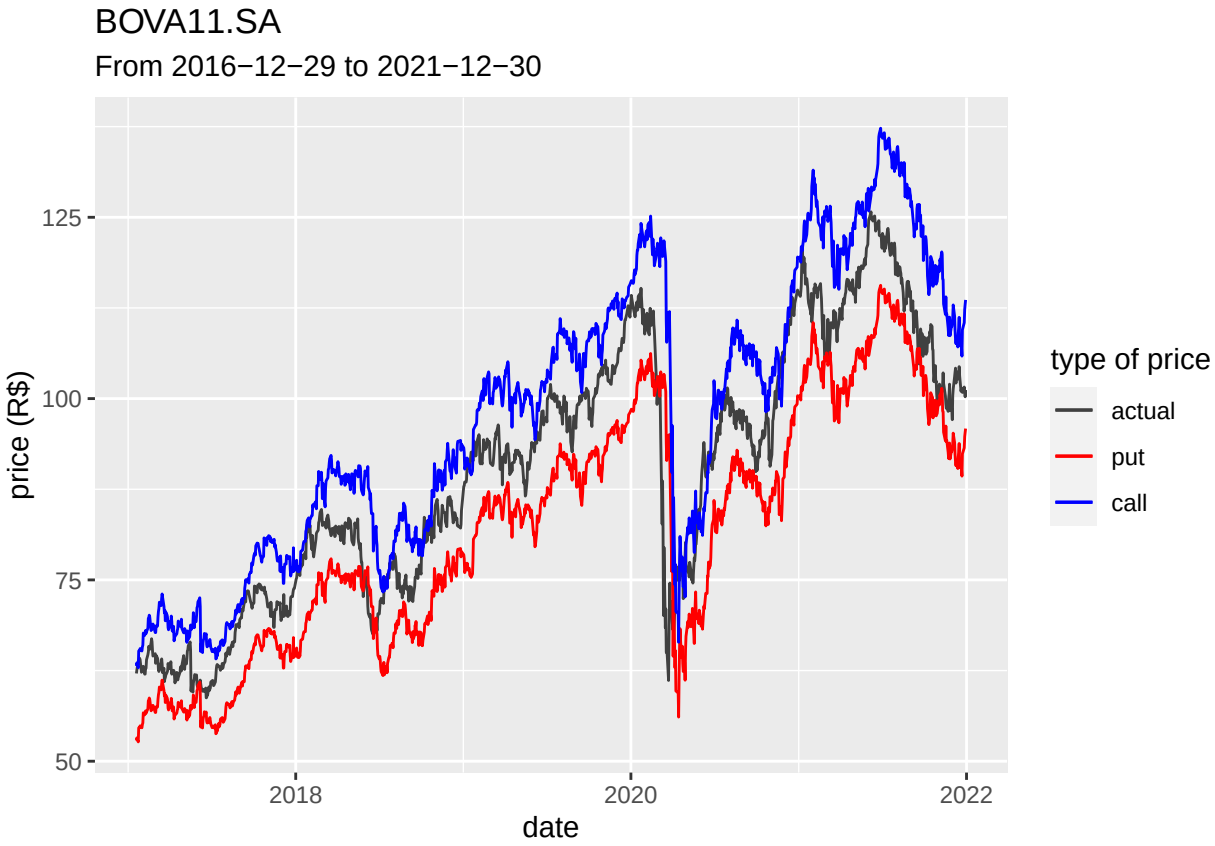
```
aim_strikes=(1+aim_var)*matrix(prices$price.adjusted[int_val-1],
                               length(int_val),2)
```

```
colour=c(actual="grey25",
         put="red",
         call="blue")

df_results=data.frame(actual=prices$price.adjusted[int_val+n-1],
                      aim_strikes,
                      date=prices$ref.date[int_val+n-1])

g=ggplot(df_results |> na.omit(),
         aes(x=date))+
  geom_line(aes(y=actual,color="actual"))+
  geom_line(aes(y=put,color="put"))+
  geom_line(aes(y=call,color="call"))+
  labs(title = stock,x="date",y="price (R$)",color="type of price",
       subtitle = paste0("From ",ini_date," to ",max_date))+
  scale_color_manual(values=colour)
```

```
print(g)
```



3.3 Financial Performance

Actual returns in the period:

```
plan(multicore)

log_ret=future({
  TTR::ROC(prices$price.adjusted,n)
}) |> value()
```

Annualized volatility in the period:

```
volatility=future({
  plan(multicore)

  sig <- NULL
  for (i in int_val) {
    sig=c(sig,
          sd(log_ret[1:(i-1)],na.rm = T))
  }

  sig=sig*sqrt(247/n)
  sig
}) |> value()
```

For the purposes of this test, the prize for a put is 0.72% of the strike, while for a call it is 0.19% of the strike. Note that in case of exercise, the return is:

- $\frac{P-St+Pr}{St}$, for put;

- $\frac{St-P_0+Pr}{P_0}$, for call.

For both cases, St is the strike, Pr is the prize, P is the stock price at exercise and P_0 is the stock price at the begining (in this case, 15 days before exercise).

For the expected scenario (non-exercise), the return is:

- $\frac{Pr}{St}$, for put;
- $\frac{P-P_0+Pr}{P_0}$, for call.

```
df_results=df_results |>
  mutate(
   past_price=prices$price.adjusted[int_val-1]
  ) |>
  select(
    past_price,actual,put,call,date
  )
```

```
ret_put=0.72/100
ret_call=0.19/100

df_results=df_results |>
  mutate(
    price_put=ret_put*put,
    price_call=ret_call*call,
    result_put=ifelse(put<actual,ret_put,(actual-put+price_put)/put),
    result_call=ifelse(call<=actual,(price_call+call-past_price)/past_price,(price_call+actual-past_pri
  )
```

These are the metrics that we monitor:

- Return;
- Standard deviation as risk;
- 97.5% quantile as drawdown;

We compare naked put, covered call and buy & hold.

```
fin_perf1=df_results |>
  summarise(
    ret_put=100*mean(result_put,na.rm=T),
    risk_put=100*sd(result_put,na.rm=T),
    draw_put=100*quantile(result_put,.025,names=F,na.rm=T),
    ret_call=100*mean(result_call,na.rm=T),
    risk_call=100*sd(result_call,na.rm=T),
    draw_cal=100*quantile(result_call,.025,names=F,na.rm=T),
    ret_bh=100*mean(actual/past_price-1,na.rm=T),
    risk_bh=100*sd(actual/past_price-1,na.rm=T),
    draw_bh=100*quantile(actual/past_price-1,.025,names=F,na.rm=T)
  )
```

```
fin_perf1 |> t()
```

```
>>>                  [,1]
>>>  ret_put     0.3137028
>>>  risk_put    2.9950424
>>>  draw_put   -1.8124295
>>>  ret_call    0.8923715
>>>  risk_call   5.8048002
>>>  draw_cal  -10.0120635
```

```
>>>  ret_bh      0.8517300
>>>  risk_bh     6.1349894
>>>  draw_bh   -10.2184886
```