元大 Smart API

策略範例

(v 1.0.0)

目錄

1. 策略邏輯說明 3

1-1 固定時間進出場 (FixTime) 4

1-2 破高破低進場 (HighLowSpread) 5

2. R 策略範例 6

2-1 撰寫行情&下單函式 6

2-1-1 行情函式程式碼 (Get_quote _base.R) : 6

2-1-2 下單函式程式碼 (Order_module_base.R) : 8

2-2 FixTime 10

2-3 HighLowSpread 13

3. Python 策略範例 16

3-1 撰寫行情&下單函式 16

3-1-1 行情函式程式碼 (Get_quote.py) : 16

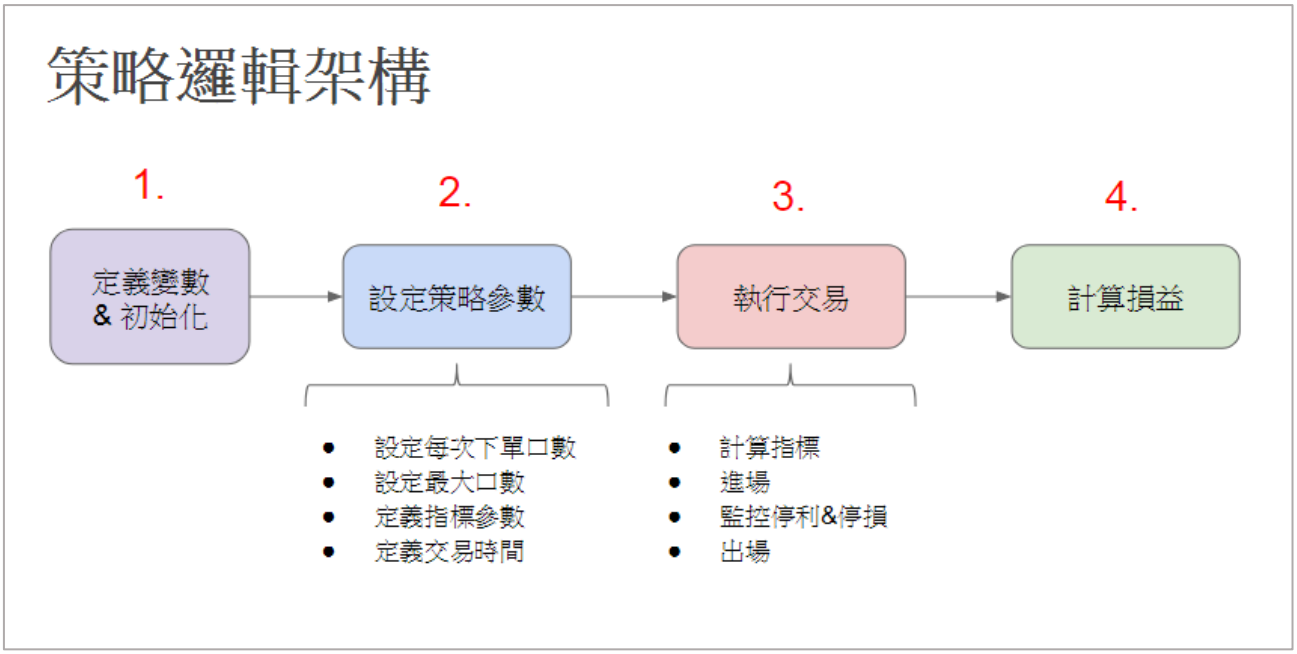
3-1-2 下單函式程式碼 (Order.py) : 17

3-2 FixTime 18

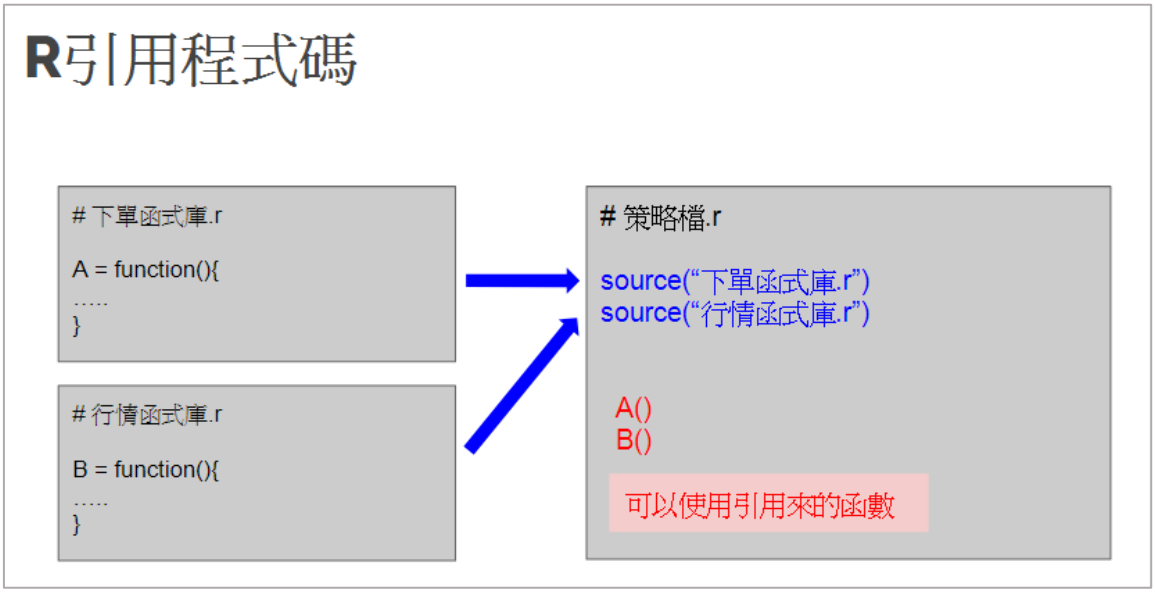
3-3 HighLowSpread 21

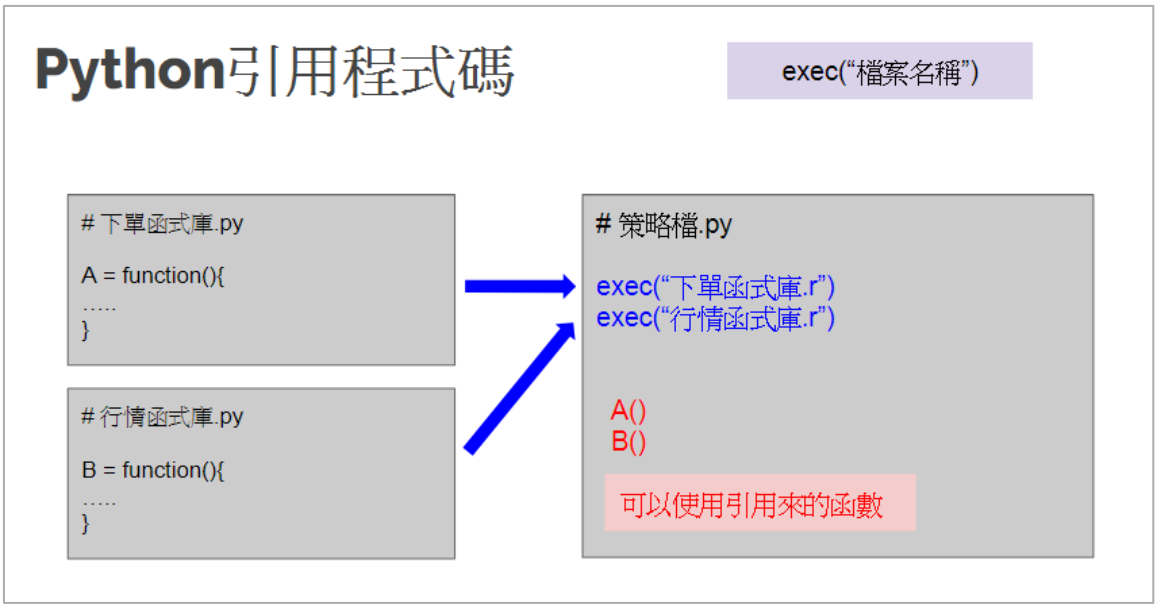
1. 策略邏輯說明

使用元大 Smart API 撰寫策略的建議流程大致上如下，一共分為以下四個步驟:



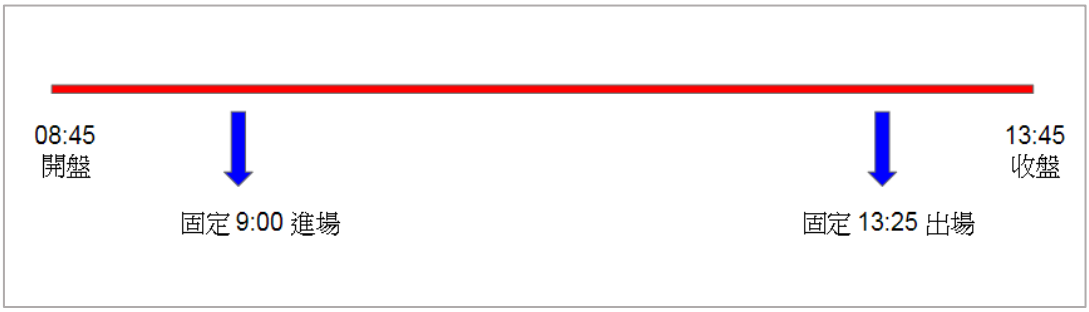
接著我們會習慣把下單和帳務查詢的語法包裝成簡單的 R 或 Python 函數，放在一個統一的函式腳本中，讓撰寫交易邏輯的程式碼中可方便地進行呼叫，引用的方式如下:





如此一來，就可以在撰寫策略的檔案中方便使用函式庫中的取得行情與下單函式，策略檔中只要專心寫策略進出場的邏輯運算就好。

1-1 固定時間進出場 (FixTime)

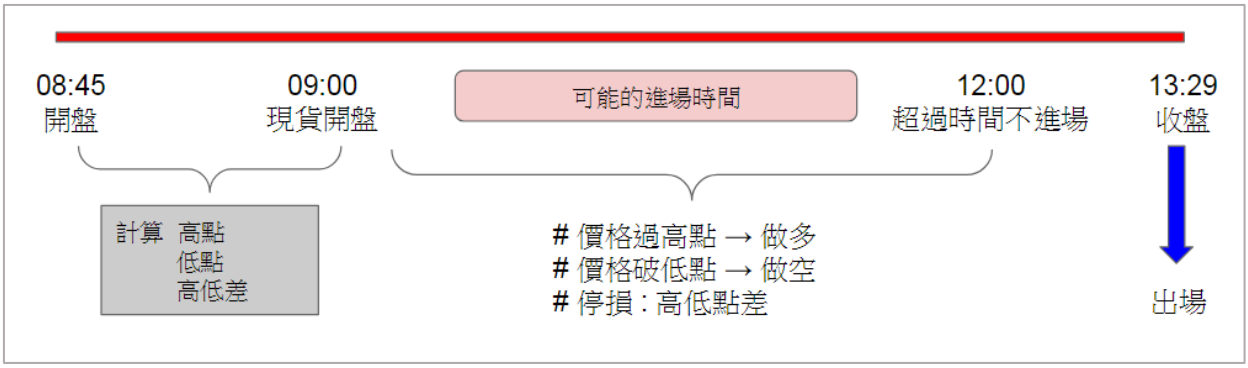


固定時間進出場的策略，策略名稱稱為 FixTime，並且可以使用 startTime 與 endTime 兩個參數控制進出場的時間，並且用第三的參數 BorS 決定要做多還是放空，舉個例子:

```
FixTime('09:00:00.00','13:25:00.00',"B")
```

的指令能夠讓策略自動在早上 9 點整的時候做多，並於下午 1 點 25 分的時候平倉出場。

1-2 破高破低進場 (HighLowSpread)



破高破低的策略，則是設定成用早上 8 點 45 分到 9 點整的區間當作計算指標的時間，算出這段時間中的最高成交價(High)、最低成交價(Low)、和高低點價差(Spread)，再利用這些指標當作交易準則，接著選定一個開始進場的時間和另一個停止進場的時間，當行情介於這兩個時間點中間時，只要成交價突破 high 就做多，反之成交價若跌破 low 就放空，並利用 spread 當作進場後設定的停損點，第三個時間則是部位平倉出場的時間。

以下程式碼為例:

```
HighLowSpread('09:00:00.00','12:00:00.00','13:29:00.00')
```

程式會先計算完 8:45~9:00 的 high、low、spread 之後，自動於 9 點後開始監控行情，一旦突破 high 就做多，跌破 low 就放空，若超過 12 點還未進場則放棄進場，若有未平倉的部位，則於現貨收盤前下午 1 點 29 分，進行反向平倉。

2. R 策略範例

2-1 撰寫行情&下單函式

2-1-1 行情函式程式碼 (Get_quote _base.R) :

```
#設置當天日期，取用當天的檔案名稱
Date <- gsub("-","",Sys.Date())
#檔案位置
DataPath <- "D:/data/"
#tail 執行檔位置
ExecPath <- "C:/Users/90813/Desktop/元大 SmartAPI/"

#取得成交資訊
GetMatchData <- function(DataPath,Date,Prodid)
{

   data <- system(paste0(ExecPath,'tail.exe -n1 ',DataPath,Date,"/",Prodid,"/",Date,"_Match.txt"),intern=TRUE)
   mdata <- strsplit(data,",")
   return(mdata)

}

#取得上下五檔價資訊
GetUpDn5Data <- function(DataPath,Date,Prodid)
{

   data <- system(paste0(ExecPath,'tail.exe -n1 ',DataPath,Date,"/",Prodid,"/",Date,"_UpDn5.txt"),intern=TRUE)
   mdata <- strsplit(data,",")
   return(mdata)

}

#判斷成交資訊是否更新
isMatchUpdate <- function(DataPath,Date,Prodid,lastTime,freq)
{

   data <- tryCatch(system(paste0(ExecPath,'tail.exe -n1 ',DataPath,Date,"/",Prodid,"/",Date,"_Match.txt"),intern=TRUE)
                    ,error=function(e) return("nodata"), warning=function(w) return("nodata"))
   # data <- system(paste0(tailPath,'tail.exe -n1 ',DataPath,Date,"/",Prodid,"/",Date,"_Match.txt"),intern=TRUE)
   mdata <- strsplit(data,",")[[1]]

   if(data=="nodata"){
      return("nodata")
   }else{
      # 小時 k,分 k,分 k
      if(freq=="hour"){
         newTime <- as.numeric(substr(mdata[2],1,2))
      }else if(freq=="min"){
         newTime <- as.numeric(paste0(substr(mdata[2],1,2),substr(mdata[2],4,5)))
      }else if(freq=="sec"){
         newTime <- as.numeric(paste0(substr(mdata[2],1,2),substr(mdata[2],4,5),substr(mdata[2],7,8)))
      }

      if(lastTime!=newTime){
         return(TRUE) # 資料有更新
      }else{
         return(FALSE) # 資料無更新
```

```
      }
   }

}

#判斷上下五檔資訊是否更新
isUpDn5Update <- function(DataPath,Date,lastTime,freq)
{
   data <- tryCatch(system(paste0(tailPath,'tail.exe -n1 ',DataPath,Date,"/",Prodid,"/",Date,"_UpDn5.txt"),intern=TRUE)
                            ,error=function(e) return("nodata"), warning=function(w) return("nodata"))
   mdata <- strsplit(data,",")[[1]]

   if(data=="nodata"){
      return("nodata")
   }else{
      # 小時 k,分 k,分 k
      if(freq=="hour"){
         newTime <- as.numeric(substr(mdata[2],1,2))
      }else if(freq=="min"){
         newTime <- as.numeric(paste0(substr(mdata[2],1,2),substr(mdata[2],4,5)))
      }else if(freq=="sec"){
         newTime <- as.numeric(paste0(substr(mdata[2],1,2),substr(mdata[2],4,5),substr(mdata[2],7,8)))
      }

      if(lastTime!=newTime){
         return(TRUE) # 資料有更新
      }else{
         return(FALSE) # 資料無更新
      }
   }
}
```

2-1-2 下單函式程式碼 (Order_module_base.R) :

```
# Order mudule base

#### 設定下單程式位置 ####
ExecPath <- "C:/Users/90813/Desktop/元大 SmartAPI/"

#### 市價委託單 ####
OrderMKT<-function(Product,BorS,Qty){

  # 下單後回傳委託書號 Order.exe TXFA8 B 0 3 MKT IOC 1
  OrderNo <- system2(paste0(ExecPath,'Order.exe'),args=paste(Product,BorS,'0',Qty,'MKT',"IOC",'1'),stdout = TRUE)
  # 回傳委託序號
  return(OrderNo)

}

#### 限價委託單 ####
OrderLMT<-function(Product,BorS,Price,Qty){

  # 下單後回傳委託書號 Order.exe TXFA8 B 10800 3 LMT ROD 1
  OrderNo<-system2(paste0(ExecPath,'Order.exe'),args=paste(Product,BorS,Price,Qty,'LMT',"ROD",'1'),stdout = TRUE)
  # 回傳委託序號
  return(OrderNo)

}

#### 單筆委託查詢 ####
QueryOrder<-function(OrderNo){
  Match<-system2(paste0(ExecPath,'GetAccount.exe'),args=paste(OrderNo),stdout = TRUE)
  return(Match)
}

#### 總委託查詢 ####
QueryAllOrder<-function(){
  Match<-system2(paste0(ExecPath,'GetAccount.exe'),args=paste("ALL"),stdout = TRUE)
  Match<-Match[-c(length(Match)-1,length(Match))]
  return(Match)
}

#### 未平倉查詢 ####
QueryOnOpen<-function(){
  onopeninfo<-system2(paste0(ExecPath,'OnOpenInterest.exe'),stdout = TRUE)
  return(onopeninfo)
}

#### 權益數查詢 ####
QueryRight<-function(){
  rightinfo<-system2(paste0(ExecPath,'FutureRights.exe'),stdout = TRUE)
  nn = as.numeric(strsplit(rightinfo,split=',')[[1]][c(7,1,3,2,20,17)])
  return = list("今日權益總值"=nn[1],"權益數"=nn[2],"可動用保證金"=nn[3],
                "原始保證金"=nn[4],"維持保證金"=nn[5],"未沖銷期貨浮動損益"=nn[6])
  return(return)
}

#### 取消委託單 ####
CancelOrder<-function(OrderNo){
  system2(paste0(ExecPath,'Order.exe')    ,args=paste('Delete',OrderNo),stdout = TRUE)
}

#### 查詢是否成交 ####
QueryMatch<-function(OrderNo){
```

```
  Match<-strsplit(system2(paste0(ExecPath,'GetAccount.exe'),args=paste(OrderNo),stdout = TRUE),",")[[1]]
  if( Match[2]=="全部成交" | Match[2]=="部分成交" ){
    return(TRUE)
  }else{
    return(FALSE)
  }
}

#### 限價單到期轉市價單 ####
LMT2MKT <- function(Product,BorS,Price,Qty,Sec){
  OrderNo<-system2(paste0(ExecPath,'Order.exe')    ,args=paste(Product,BorS,Price,Qty,'LMT','ROD','1'),stdout = TRUE)
  to <- Sys.time()
  while(as.numeric(difftime(Sys.time(), to, u = 'secs')) < Sec){
    if(isTRUE(QueryMatch(OrderNo))){
      return(QueryOrder(OrderNo))
    }
  }
  CancelOrder(OrderNo)
  Match<-OrderMKT(Product,BorS,Qty)
  return(Match)
}

#### 限價單到期轉刪單 ####
LMT2DEL <- function(Product,BorS,Price,Qty,Sec){
  OrderNo<-system2(paste0(ExecPath,'Order.exe')    ,args=paste(Product,BorS,Price,Qty,'LMT','ROD','1'),stdout = TRUE)
  to <- Sys.time()
  while(as.numeric(difftime(Sys.time(), to, u = 'secs')) < Sec){
    if(isTRUE(QueryMatch(OrderNo))){
      return(QueryOrder(OrderNo))
    }
  }
  CancelOrder(OrderNo)
  return(FALSE)
}

#### 查看所有未取消委託 ####
QueryAllUnfinished<-function(){
  system2(paste0(ExecPath,'GetUnfinished.exe'),stdout = TRUE)
}

#### 取消所有委託 ####
CancelAll<-function(){
  system2(paste0(ExecPath,'CancelALL.exe'),stdout = TRUE)
}

#### 變更連線中商品 ####
ChangeProd<-function(){
  system2(paste0(ExecPath,'ChangeProdid.exe'),stdout = TRUE)
}
```

2-2 FixTime

Fixtime 策略主程式碼如下:

```
## FixTime 固定時間進出場

source("C:/Users/90813/Desktop/R 策略模組/R 策略模組/下單&帳務模組/Order_module_base.R",encoding="UTF8")
source("C:/Users/90813/Desktop/R 策略模組/R 策略模組/下單&帳務模組/Get_quote_base.R",encoding="UTF8")

# startTime 進場,endTime 出場
FixTime = function(startTime,endTime,BorS){

  #### 定義變數 & 初始化 ####
  print("Initialize.....")
  Date <- gsub("-","",Sys.Date())
  code = "TXFC8"
  TXF_MatchPrice = NA
  TXF_MatchTime = NA
  BPrice = NA
  SPrice = NA
  BTime = NA
  STime = NA

  BorS = BorS
  position = 0
  TradingRecord = c()
  SingleRecord = c()
  profit = 0

  #### 設定策略參數 ####
  print("Set the parameter.....")

  #設定 單筆下單口數 & 最大在倉口數
  lot = 1
  Maxlot = 1

  # 定義交易時間
  options(digits.secs=2)
  startTime <- strptime('09:00:00.00','%H:%M:%OS')
  endTime <- strptime('13:25:00.00','%H:%M:%OS')


  #### 執行交易 ####
  print("Taking market quote.....")

  ## 進場
  if(BorS == "B"){

    ## 做多
    while( position < Maxlot ){

      ## 取得最新報價
      Mdata<-GetMatchData(DataPath,Date,code)
      TXF_MatchTime <- strptime(Mdata[[1]][2],'%H:%M:%OS')
      TXF_MatchPrice <- as.numeric(Mdata[[1]][3])

      ## 時間到 --> 進場
      if( TXF_MatchTime >= startTime ){
```

```
            OrderMKT(code,"B",lot)
            BPrice = TXF_MatchPrice
            BTime = TXF_MatchTime
            position = position + lot
            print(paste("Buy",code,"|","Buy Price:",TXF_MatchPrice,"|","Time:",Sys.time()))
        }
    }

    ## 出場
    while( position != 0 ){

        ## 取得最新報價
        Mdata<-GetMatchData(DataPath,Date,code)
        TXF_MatchTime <- strptime(Mdata[[1]][2],'%H:%M:%OS')
        TXF_MatchPrice <- as.numeric(Mdata[[1]][3])

        ## 時間到 --> 出場
        if( TXF_MatchTime >= endTime ){
            OrderMKT(code,"S",position)
            SPrice = TXF_MatchPrice
            STime = TXF_MatchTime
            position = 0
            print(paste("Sell",code,"|","Sell Price:",TXF_MatchPrice,"|","Time:",Sys.time()))
        }
    }

}else if(BorS == "S"){

    ## 做空
    while( position > -1*Maxlot ){

        ## 取得最新報價
        Mdata<-GetMatchData(DataPath,Date,code)
        TXF_MatchTime <- strptime(Mdata[[1]][1],'%H:%M:%OS')
        TXF_MatchPrice <- as.numeric(Mdata[[1]][2])

        ## 時間到 --> 進場
        if( TXF_MatchTime >= startTime ){

            OrderMKT(code,"S",lot)
            SPrice = TXF_MatchPrice
            STime = TXF_MatchTime
            position = position - lot
            print(paste("Sell",code,"|","Sell Price:",TXF_MatchPrice,"|","Time:",Sys.time()))
        }
    }

    ## 出場
    while( position != 0 ){

        ## 取得最新報價
        Mdata<-GetMatchData(DataPath,Date,code)
        TXF_MatchTime <- strptime(Mdata[[1]][2],'%H:%M:%OS')
        TXF_MatchPrice <- as.numeric(Mdata[[1]][3])

        ## 時間到 --> 出場
        if( TXF_MatchTime >= endTime ){
            OrderMKT(code,"B",position)
            BPrice = TXF_MatchPrice
            BTime = TXF_MatchTime
            position = 0
```

```
            print(paste("Buy",code,"|","Buy Price:",TXF_MatchPrice,"|","Time:",Sys.time()))
          }
        }

    }

    ####  計算損益  ####
    profit = SPrice - BPrice
    print(paste("Profit :",profit))

    ## 回傳交易紀錄
    #
c(Dir="Buy",OpenPrice=BPrice,ClosePrice=SPrice,OpenTime=as.character(BTime),CloseTime=as.character(STime),Profit=prof
it)
    if(BorS == "B"){
        SingleRecord = c(BorS="Buy",OpenPrice=BPrice,ClosePrice=SPrice,
                         OpenTime=as.character(BTime),CloseTime=as.character(STime),Profit=profit)
    }else if(BorS == "S"){
        SingleRecord = c(BorS="Sell",OpenPrice=SPrice,ClosePrice=BPrice,
                         OpenTime=as.character(STime),CloseTime=as.character(BTime),Profit=profit)
    }

    TradingRecord = rbind(TradingRecord,SingleRecord)
    rownames(TradingRecord)=(1:nrow(TradingRecord))
    return(TradingRecord)

}

## Testing
Record = FixTime('09:00:00.00','13:25:00.00',"B")
Record
```

2-3 HighLowSpread

HighLowSpread 策略主程式碼如下 ：

```
## 破高破低追價

source("C:/Users/90813/Desktop/R 策略模組/R 策略模組/下單&帳務模組/Order_module_base.R",encoding="UTF8")
source("C:/Users/90813/Desktop/R 策略模組/R 策略模組/下單&帳務模組/Get_quote_base.R",encoding="UTF8")

HighLow = function(startTime,endTime,outTime){

  #### 定義變數 & 初始化 ####
  print("Initialize.....")
  Date <- gsub("-","",Sys.Date())
  code = "TXFD8"
  TXF_MatchPrice = NA
  TXF_MatchTime = NA
  BPrice = NA
  SPrice = NA
  BTime = NA
  STime = NA

  BorS = NA
  position = 0
  TradingRecord = c()
  SingleRecord = c()
  profit = 0

  high = 0
  low = Inf
  spread = 0

  #### 設定策略參數 ####

  print("Set the parameter.....")
  #設定 單筆下單口數 & 最大在倉口數
  lot = 1
  Maxlot = 1
  # 定義指標時間

  # 定義交易時間
  options(digits.secs=2)
  startTime <- strptime(startTime,'%H:%M:%OS')
  endTime <- strptime(endTime,'%H:%M:%OS')
  outTime <- strptime(outTime,'%H:%M:%OS')

  #### 執行交易 ####
  print("Waiting for market open .....")
  while( Sys.time() < strptime("08:44:59",'%H:%M:%OS')){
    ## 等待開盤
  }

  print("Calculate indicator .....")
  ## 計算指標 {高低點}
  while( high == 0 )
  {
    while( Sys.time() > strptime("08:45:00",'%H:%M:%OS') &
           Sys.time() < strptime("09:00:00",'%H:%M:%OS') )
    {
```

```
      ## 取得最新報價
      Mdata<-GetMatchData(DataPath,Date,code)
      TXF_MatchTime <- strptime(Mdata[[1]][2],'%H:%M:%OS')
      TXF_MatchPrice <- as.numeric(Mdata[[1]][3])

      ## 計算高低點
      if( TXF_MatchPrice > high ) high = TXF_MatchPrice
      if( TXF_MatchPrice < low ) low = TXF_MatchPrice
      if( (high-low) >    spread ) spread = (high-low)
    }
  }
  print(paste0("high:",high))
  print(paste0("low:",low))
  print(paste0("spread:",spread))

  ## 進場
  while( abs(position) < Maxlot &
           Sys.time() > startTime &
           Sys.time() < endTime ){

    ## 取得最新報價
    Mdata<-GetMatchData(DataPath,Date,code)
    TXF_MatchTime <- strptime(Mdata[[1]][2],'%H:%M:%OS')
    TXF_MatchPrice <- as.numeric(Mdata[[1]][3])

    ## 破高 --> 進場多
    if( TXF_MatchPrice > high ){

      OrderMKT(code,"B",lot)
      BorS = "B"
      BPrice = TXF_MatchPrice
      BTime = TXF_MatchTime
      position = position + lot
      print(paste("Buy",code,"|","Buy Price:",TXF_MatchPrice,"|","Time:",Sys.time()))
    }
    ## 破低 --> 進場空
    else if( TXF_MatchPrice < low ){

      OrderMKT(code,"S",lot)
      BorS = "S"
      SPrice = TXF_MatchPrice
      STime = TXF_MatchTime
      position = position - lot
      print(paste("Sell",code,"|","Sell Price:",TXF_MatchPrice,"|","Time:",Sys.time()))
    }
  }

  ## 出場
  while( position != 0 ){

    ## 取得最新報價
    Mdata<-GetMatchData(DataPath,Date,code)
    TXF_MatchTime <- strptime(Mdata[[1]][2],'%H:%M:%OS')
    TXF_MatchPrice <- as.numeric(Mdata[[1]][3])

    ## 停損  = Spread
    if( position > 0 & (BPrice - TXF_MatchPrice) > spread ){

      OrderMKT(code,"S",position)
      SPrice = TXF_MatchPrice
      STime = TXF_MatchTime
```

```
        position = 0
        print(paste("Sell",code,"|","Sell Price:",TXF_MatchPrice,"|","Time:",Sys.time()))

      }else if( position < 0 & (TXF_MatchPrice - SPrice) > spread ){

        OrderMKT(code,"B",abs(position))
        BPrice = TXF_MatchPrice
        BTime = TXF_MatchTime
        position = 0
        print(paste("Buy",code,"|","Buy Price:",TXF_MatchPrice,"|","Time:",Sys.time()))

      }

      ## 時間到 --> 出場
      if( TXF_MatchTime >= outTime ){

        if( position > 0 ){
          OrderMKT(code,"S",position)
          SPrice = TXF_MatchPrice
          STime = TXF_MatchTime
          position = 0
          print(paste("Sell",code,"|","Sell Price:",TXF_MatchPrice,"|","Time:",Sys.time()))
        }else if( position < 0){
          OrderMKT(code,"B",abs(position))
          BPrice = TXF_MatchPrice
          BTime = TXF_MatchTime
          position = 0
          print(paste("Buy",code,"|","Buy Price:",TXF_MatchPrice,"|","Time:",Sys.time()))
        }

      }

    }

    #### 計算損益  ####
    profit = SPrice - BPrice
    print(paste("Profit :",profit))

    ## 回傳交易紀錄
    if(is.na(BorS)){
      print("No any signal !")
    }else if(BorS == "B"){
      SingleRecord = c(BorS="Buy",OpenPrice=BPrice,ClosePrice=SPrice,
                       OpenTime=as.character(BTime),CloseTime=as.character(STime),Profit=profit)
    }else if(BorS == "S"){
      SingleRecord = c(BorS="Sell",OpenPrice=SPrice,ClosePrice=BPrice,
                       OpenTime=as.character(STime),CloseTime=as.character(BTime),Profit=profit)
    }

    TradingRecord = rbind(TradingRecord,SingleRecord)
    rownames(TradingRecord)=(1:nrow(TradingRecord))
    return(TradingRecord)

}

## 測試
Record = HighLow("09:00:00","12:00:00","13:29:00")
Record
```

3. Python 策略範例

3-1 撰寫行情&下單函式

3-1-1 行情函式程式碼 (Get_quote.py) :

```
# -*- coding: UTF-8 -*-
#載入相關套件
import time
import tailer

#取得當天日期
Date=time.strftime("%Y%m%d")
#設定檔案位置
DataPath="D:/data/"+Date+"/TXFD8/"

#開啟檔案
def MatchFile(Prodid):
      return(open("D:/data/"+Date+"/"+Prodid+"/"+Date+'_Match.txt'))

def UpDn5File (Prodid):
      return(open("D:/data/"+Date+"/"+Prodid+"/"+Date+'_Match.txt'))

#持續取得成交資訊(QM)
def getMatch(Prodid):
  return tailer.follow(MatchFile(Prodid),0) #second=0

#持續取得上下五檔價資訊(QM)
def getUpDn5(Prodid):
  return tailer.follow(UpDn5File(Prodid),0)

#取得最新一筆成交資訊
def getLastMatch(Prodid):
  return tailer.tail(MatchFile(Prodid),3)[-2].split(",")

#取得最新一筆上下五檔價資訊
def getLastUpDn5(Prodid):
  return tailer.tail(UpDn5File(Prodid),3)[-2].split(",")
```

3-1-2 下單函式程式碼 (Order.py) :

```
# -*- coding: UTF-8 -*-
#載入相關套件
import subprocess

#下單子程式放置位置
ExecPath = "C:/Users/90813/Desktop/元大 SmartAPI/"

#市價單下單
def OrderMKT(Product,BS,Qty):
  OrderNo=subprocess.check_output([ExecPath+"order.exe",Product,BS,"0",Qty,"MKT",
              "IOC","1"]).decode("big5").strip('\r\n')
  time.sleep(0.3)
  return subprocess.check_output([ExecPath+"GetAccount.exe",OrderNo]).decode("big5").strip('\r\n').split(',')[0:2]

#限價單委託
def OrderLMT(Product,BS,Price,Qty):
  OrderNo=subprocess.check_output([ExecPath+"order.exe",Product,BS,Price,Qty,"LMT",
              "ROD","0"]).decode("big5").strip('\r\n')
  return OrderNo

#查詢帳務明細
def QueryOrder(OrderID):
  ReturnInfo=subprocess.check_output([ExecPath+"GetAccount.exe",OrderID]).decode("big5").strip('\r\n').split(',')
  return ReturnInfo

#查詢總帳務明細
def QueryAllOrder():
  ReturnInfo=subprocess.check_output([ExecPath+"GetAccount.exe","ALL"]).decode("big5").strip('\r\n').split('\n')
  ReturnInfo= [ line.split(',') for line in ReturnInfo]
  return ReturnInfo

#查詢未平倉資訊
def QueryOnOpen():
  ReturnInfo=subprocess.check_output([ExecPath+"OnOpenInterest.exe"]).decode("big5").strip('\r\n')
  return ReturnInfo.split(',')


#查詢權益數資訊
def QueryRight():
  ReturnInfo=subprocess.check_output([ExecPath+"FutureRights.exe"]).decode("big5").strip('\r\n')
  return ReturnInfo.split(',')


#取消委託
def CancelOrder(OrderID):
  ReturnInfo=subprocess.check_output([ExecPath+"order.exe","Delete",OrderID]).decode("big5")
  if "刪單已傳送" in ReturnInfo:
    return True
  else:
    return False

#查看所有未取消委託
def QueryAllUnfinished():
    ReturnInfo = subprocess.check_output([ExecPath+'GetUnfinished.exe']).decode("big5").strip('\r\n')
    return ReturnInfo
```

```
#取消所有委託
def CancelAll():
    ReturnInfo = subprocess.check_output([ExecPath+'CancelALL.exe']).decode("big5").strip('\r\n')
    return ReturnInfo

#限價轉刪單
def LMT2DEL(Product,BS,Price,Qty,Sec):
  OrderID=OrderLMT(Product,BS,Price,Qty).decode("big5")
  StartTime=time.time()
  while time.time()-StartTime<Sec:
    ReturnInfo=QueryOrder(OrderID)[1]
    if ReturnInfo == '全部成交':
      return ReturnInfo
  CancelOrder(OrderID)
  return False

#限價轉市價
def LMT2MKT(Product,BS,Price,Qty,Sec):
  OrderID=OrderLMT(Product,BS,Price,Qty)
  StartTime=time.time()
  while time.time()-StartTime<Sec:
    ReturnInfo=QueryOrder(OrderID)[1]
    if ReturnInfo == '全部成交':
      return ReturnInfo
  if CancelOrder(OrderID):
    ReturnInfo=OrderMKT(Product,BS,Qty)
    return ReturnInfo
```

3-2 FixTime

```
# -*- coding: utf-8 -*-

# 載入相關套件
from datetime import datetime
import pandas as pd
import os
os.chdir("C:/Users/90813/Desktop/python 策略模組")
exec(open('Order.py',encoding = 'utf8').read())
exec(open('Get_quote.py',encoding = 'utf8').read())

# startTime 進場,endTime 出場
def FixTime(startTime,endTime,BorS):

    currDate = datetime.now().strftime('%Y-%m-%d')
    startTime = datetime.strptime(currDate +' '+ startTime,'%Y-%m-%d %H:%M:%S.%f')
    endTime = datetime.strptime(currDate +' '+ endTime,'%Y-%m-%d %H:%M:%S.%f')
    BorS = 'B'

    #### 定義變數 & 初始化 ####
    print("Initialize.....")
    Date=time.strftime("%Y%m%d")
    code = "TXFC8"
    TXF_MatchPrice = ""
    TXF_MatchTime = ""
    BPrice = ""
    SPrice = ""
    BTime = ""
```

```
    STime = ""

    BorS = BorS
    position = 0
    TradingRecord = pd.DataFrame(columns=['BorS','OpenPrice','ClosePrice','OpenTime','CloseTime','Profit'])
    SingleRecord = pd.DataFrame(columns=['BorS','OpenPrice','ClosePrice','OpenTime','CloseTime','Profit'])
    profit = 0

    #### 設定策略參數 ####
    print("Set the parameter.....")

    #設定 單筆下單口數 & 最大在倉口數
    lot = 1
    Maxlot = 1

    # 定義交易時間
    # startTime = datetime.strptime('09:00:00.00','%H:%M:%S.%f')
    # endTime = datetime.strptime('13:25:00.00','%H:%M:%S.%f')

    # 定義加碼 & 停損

    #### 執行交易 ####
    print("Taking market quote.....")

    ## 進場
    if(BorS == "B"):

        ## 做多
        for Mdata in getMatch():

            #print(Mdata)

            ## 取得最新報價
            currDate = datetime.now().strftime('%Y-%m-%d')
            TXF_MatchTime = datetime.strptime(currDate + " " + Mdata.split(",")[0],"%Y-%m-%d %H:%M:%S.%f")
            TXF_MatchPrice = Mdata.split(",")[1]

            #print(TXF_MatchTime)
            #print(TXF_MatchPrice)

            ## 時間到 --> 進場
            if( TXF_MatchTime >= startTime ):

                OrderMKT(code,"B",str(lot))
                BPrice = TXF_MatchPrice
                BTime = TXF_MatchTime
                position = position + lot
                print("Buy "+ code +" | Buy Price: "+ TXF_MatchPrice +" | Time: "+ datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))

            ## 最大口數限制
            if position >= Maxlot : break

        for Mdata in getMatch():
            ## 出場
            if( position != 0 ):

                ## 取得最新報價
                currDate = datetime.now().strftime('%Y-%m-%d')
                TXF_MatchTime = datetime.strptime(currDate + " " + Mdata.split(",")[0],"%Y-%m-%d %H:%M:%S.%f")
                TXF_MatchPrice = Mdata.split(",")[1]
```

```
            ## 時間到 --> 出場
            if( TXF_MatchTime >= endTime ):
                OrderMKT(code,"S",str(position))
                SPrice = TXF_MatchPrice
                STime = TXF_MatchTime
                position = 0
                print("Sell "+code+" | Sell Price: "+TXF_MatchPrice+" | Time: "+datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))
                break

    elif(BorS == "S"):

        ## 做空
        for Mdata in getMatch():

            ## 取得最新報價
            currDate = datetime.now().strftime('%Y-%m-%d')
            TXF_MatchTime = datetime.strptime(currDate + " " + Mdata.split(",")[0],"%Y-%m-%d %H:%M:%S.%f")
            TXF_MatchPrice = Mdata.split(",")[1]

            ## 時間到 --> 進場
            if( TXF_MatchTime >= startTime ):

                OrderMKT(code,"S",str(lot))
                SPrice = TXF_MatchPrice
                STime = TXF_MatchTime
                position = position - lot
                print("Sell "+code+" | Sell Price: "+TXF_MatchPrice+" | Time: "+datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))

            ## 最大口數限制
            if position <= -1*Maxlot : break

            ## 停損 & 停利

        for Mdata in getMatch():
            ## 出場
            if( position != 0 ):

                ## 取得最新報價
                currDate = datetime.now().strftime('%Y-%m-%d')
                TXF_MatchTime = datetime.strptime(currDate + " " + Mdata.split(",")[0],"%Y-%m-%d %H:%M:%S.%f")
                TXF_MatchPrice = Mdata.split(",")[1]

            ## 時間到 --> 出場
            if( TXF_MatchTime >= endTime ):
                OrderMKT(code,"B",str(position))
                BPrice = TXF_MatchPrice
                BTime = TXF_MatchTime
                position = 0
                print("Buy "+ code +" | Buy Price: "+ TXF_MatchPrice +" | Time: "+ datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))
                break

    ####  計算損益  ####
    profit = int(SPrice) - int(BPrice)
    print("Profit :" + str(profit))
```

```
    ## 回傳交易紀錄
    if(BorS == "B"):
        SingleRecord = pd.DataFrame({'BorS':"Buy",'OpenPrice':BPrice,'ClosePrice':SPrice,
                                     'OpenTime':datetime.strftime(BTime,"%Y-%m-%d %H:%M:%S.%f"),
                                     'CloseTime':datetime.strftime(STime,"%Y-%m-%d %H:%M:%S.%f"),
                                     'Profit':profit}
                                     ,index=[0])

    elif(BorS == "S"):
        SingleRecord = pd.DataFrame({'BorS':"Sell",'OpenPrice':SPrice,'ClosePrice':BPrice,
                                     'OpenTime':datetime.strftime(STime,"%Y-%m-%d %H:%M:%S.%f"),
                                     'CloseTime':datetime.strftime(BTime,"%Y-%m-%d %H:%M:%S.%f"),
                                     'Profit':profit}
                                     ,index=[0])

    TradingRecord = TradingRecord.append([SingleRecord],ignore_index=True)
    #print(TradingRecord)
    return(TradingRecord)



## Testing
Record = FixTime('09:00:00.00','13:25:00.00',"B")
Record
```

3-3 HighLowSpread

```
from datetime import datetime
import pandas as pd
import math
import os
os.chdir("C:/Users/90813/Desktop/python 策略模組")
exec(open('Order.py',encoding = 'utf8').read())
exec(open('Get_quote.py',encoding = 'utf8').read())

# startTime 進場,endTime 出場
def HighLowSpread(startTime,endTime,outTime):

    currDate = datetime.now().strftime('%Y-%m-%d')
    startTime = datetime.strptime(currDate +' '+ startTime,'%Y-%m-%d %H:%M:%S.%f')
    endTime = datetime.strptime(currDate +' '+ endTime,'%Y-%m-%d %H:%M:%S.%f')
    outTime = datetime.strptime(currDate +' '+ outTime,'%Y-%m-%d %H:%M:%S.%f')

    #### 定義變數 & 初始化 ####
    print("Initialize.....")
    Date=time.strftime("%Y%m%d")
    code = "TXFD8"
    TXF_MatchPrice = ""
    TXF_MatchTime = ""
    BPrice = ""
    SPrice = ""
    BTime = ""
    STime = ""

    BorS = ''
    position = 0
    TradingRecord = pd.DataFrame(columns=['BorS','OpenPrice','ClosePrice','OpenTime','CloseTime','Profit'])
```

```
SingleRecord = pd.DataFrame(columns=['BorS','OpenPrice','ClosePrice','OpenTime','CloseTime','Profit'])
profit = 0

high = 0
low = math.inf
spread = 0

#### 設定策略參數 ####
print("Set the parameter.....")

#設定 單筆下單口數 & 最大在倉口數
lot = 1
Maxlot = 1

#### 執行交易 ####
print("Calculate indicator.....")

## 計算指標 {高低點}
for Mdata in getMatch(code):
    ## 取得最新報價
    currDate = datetime.now().strftime('%Y-%m-%d')
    TXF_MatchTime = datetime.strptime(currDate + " " + Mdata.split(",")[1],"%Y-%m-%d %H:%M:%S.%f")
    TXF_MatchPrice = Mdata.split(",")[2]

    #print(TXF_MatchTime)
    #print(TXF_MatchPrice)

    indicTime1 = datetime.strptime(currDate +' 08:45:00.00','%Y-%m-%d %H:%M:%S.%f')
    indicTime2 = datetime.strptime(currDate +' 09:00:00.00','%Y-%m-%d %H:%M:%S.%f')
    if( indicTime1 < TXF_MatchTime < indicTime2 ):
        if( int(TXF_MatchPrice) > high):
            high = int(TXF_MatchPrice)
        if( int(TXF_MatchPrice) < low):
            low = int(TXF_MatchPrice)
        if((high-low) > spread):
            spread = (high-low)
    else:
        print("High:",high," Low:",low," Spread:",spread)
        break


print("Open position.....")
## 進場
for Mdata in getMatch(code):

  ## 取得最新報價
  currDate = datetime.now().strftime('%Y-%m-%d')
  TXF_MatchTime = datetime.strptime(currDate + " " + Mdata.split(",")[1],"%Y-%m-%d %H:%M:%S.%f")
  TXF_MatchPrice = Mdata.split(",")[2]

  #print(TXF_MatchTime)
  #print(TXF_MatchPrice)

  ## 破高 --> 做多進場
  if( startTime < TXF_MatchTime < endTime):
      if( int(TXF_MatchPrice) > high ):

          OrderMKT(code,"B",str(lot))
          BorS = "B"
          BPrice = TXF_MatchPrice
          BTime = TXF_MatchTime
```

```
                position = position + lot
                print("Buy "+ code +" | Buy Price: "+ TXF_MatchPrice +" | Time: "+ datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))

            ## 破低 --> 放空進場
            elif( int(TXF_MatchPrice) < low ):

                OrderMKT(code,"S",str(lot))
                BorS = "S"
                SPrice = TXF_MatchPrice
                STime = TXF_MatchTime
                position = position - lot
                print("Sell "+ code +" | Sell Price: "+ TXF_MatchPrice +" | Time: "+ datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))
        else:
            break
        ## 最大口數限制
        if abs(position) >= Maxlot : break

    ## 出場
    print("Close position.....")
    for Mdata in getMatch(code):

        if( position != 0 ):

            ## 取得最新報價
            currDate = datetime.now().strftime('%Y-%m-%d')
            TXF_MatchTime = datetime.strptime(currDate + " " + Mdata.split(",")[1],"%Y-%m-%d %H:%M:%S.%f")
            TXF_MatchPrice = Mdata.split(",")[2]

            ## 停損 = Spread
            if( position > 0 and (int(BPrice) - int(TXF_MatchPrice)) > spread ):

                OrderMKT(code,"S",str(position))
                SPrice = TXF_MatchPrice
                STime = TXF_MatchTime
                position = 0
                print("Sell "+code+" | Sell Price: "+TXF_MatchPrice+" | Time: "+datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))
                break

            elif( position < 0 and (int(TXF_MatchPrice) - int(SPrice)) > spread ):

                OrderMKT(code,"B",str(abs(position)))
                BPrice = TXF_MatchPrice
                BTime = TXF_MatchTime
                position = 0
                print("Buy "+code+" | Buy Price: "+TXF_MatchPrice+" | Time: "+datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))
                break


            ## 時間到 --> 出場
            if( TXF_MatchTime >= outTime ):

                if( position > 0 ):
                    OrderMKT(code,"S",str(position))
                    SPrice = TXF_MatchPrice
                    STime = TXF_MatchTime
                    position = 0
                    print("Sell "+code+" | Sell Price: "+TXF_MatchPrice+" | Time: "+datetime.now().strftime('%Y/%m/%d
```

```
%H:%M:%S'))
                        break
                else:
                        OrderMKT(code,"B",str(abs(position)))
                        SPrice = TXF_MatchPrice
                        STime = TXF_MatchTime
                        position = 0
                        print("Buy "+code+" | Buy Price: "+TXF_MatchPrice+" | Time: "+datetime.now().strftime('%Y/%m/%d
%H:%M:%S'))
                        break

    ####  計算損益  ####
    if( SPrice!='' and BPrice!='' ):
        profit = int(SPrice) - int(BPrice)
        print("Profit :" + str(profit))


    ## 回傳交易紀錄
    if(BorS == "B"):
        SingleRecord = pd.DataFrame({'BorS':"Buy",'OpenPrice':BPrice,'ClosePrice':SPrice,
                                'OpenTime':datetime.strftime(BTime,"%Y-%m-%d %H:%M:%S.%f"),
                                'CloseTime':datetime.strftime(STime,"%Y-%m-%d %H:%M:%S.%f"),
                                'Profit':profit}
                                ,index=[0])

    elif(BorS == "S"):
        SingleRecord = pd.DataFrame({'BorS':"Sell",'OpenPrice':SPrice,'ClosePrice':BPrice,
                                'OpenTime':datetime.strftime(STime,"%Y-%m-%d %H:%M:%S.%f"),
                                'CloseTime':datetime.strftime(BTime,"%Y-%m-%d %H:%M:%S.%f"),
                                'Profit':profit}
                                ,index=[0])
    else:
        print("No any signal !")


    TradingRecord = TradingRecord.append([SingleRecord],ignore_index=True)
    #print(TradingRecord)
    return(TradingRecord)



## Testing
Record = HighLowSpread('09:00:00.00','12:00:00.00','13:29:00.00')
Record
```