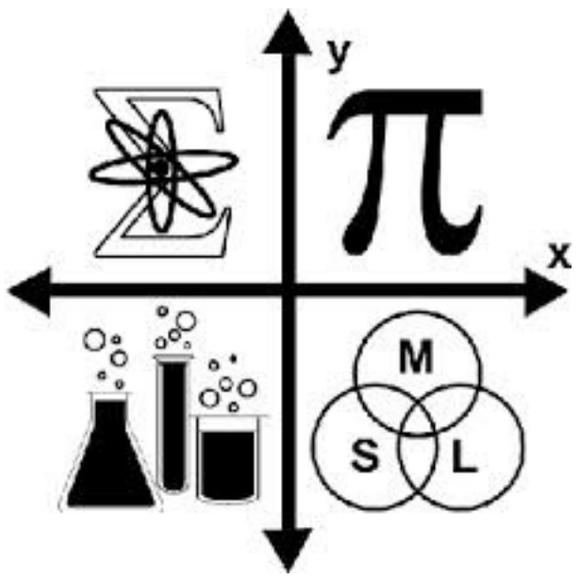
English

Quant Strategy



English

Interact with Hedge fund/ Quant shop like algorithms in English… Can Do?

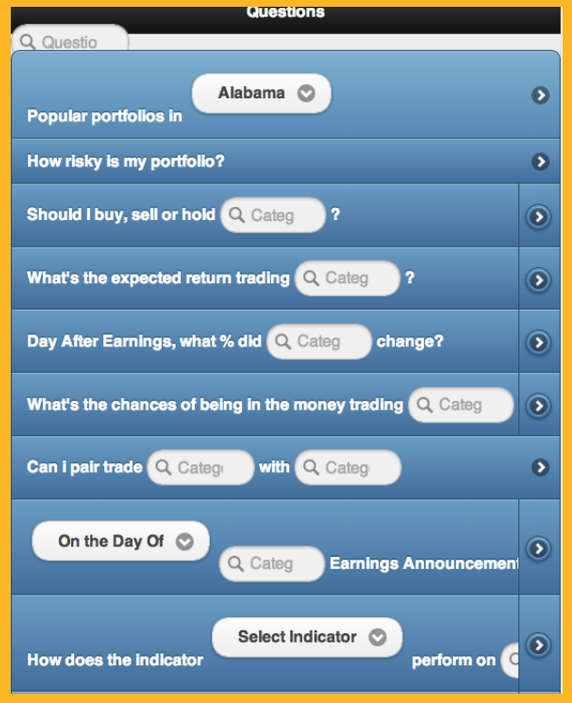
Rapidly build,test and deploy algorithms…Can Do?

R

Mobile App

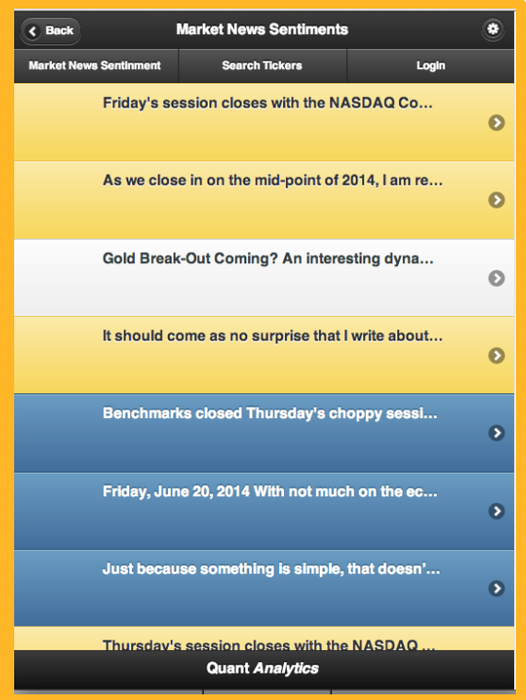
"Sont les mots qui vont très bien ensemble" ...?

source: picturesleevegallery.com



UI

Plots



Post Processing

JSONP

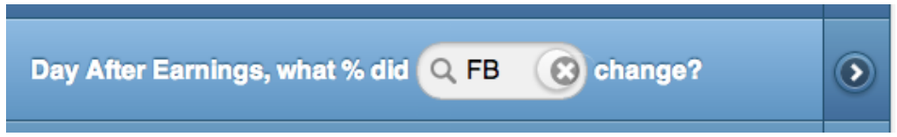
Yes & Yes !

xmlParse blotter twitteR
RMySQL quantstrat plyR
RJSONIO sentiment tm
FastRWeb

AJAX

JSO

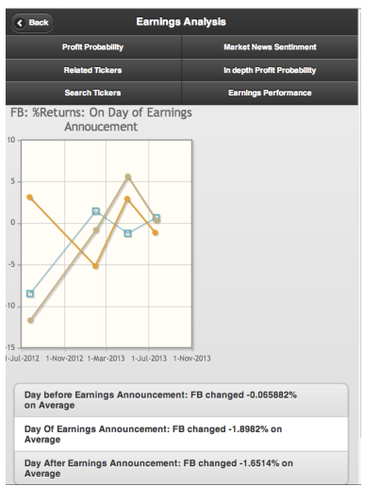
jqPlot



http:// myserver.com/ cgi-bin/R/ EarningsReturns? callback=**jsonpCallbackEarningsCal**&symbolImpact=FB&Earnsymbl=FB&_=1403372969812

run <- function(callback,symbolImpact, Earnsymbl,...) {}

jsonpCallbackEarningsCal({
“DatesDayN": ["2012-07-26", "2013-01-30", "2013-05-01", "2013-07-22"],
“ReturnsDayN": [-8.4867, 1.4615, -1.2243, 0.65688],})



2

1

1

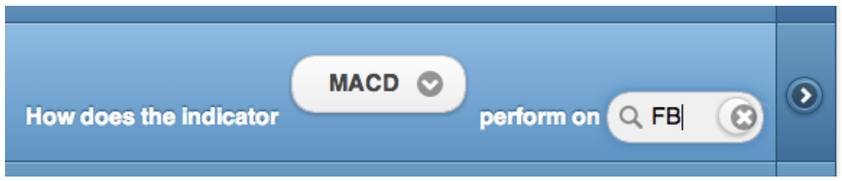
2

3

2

1

User experience driven Architecture



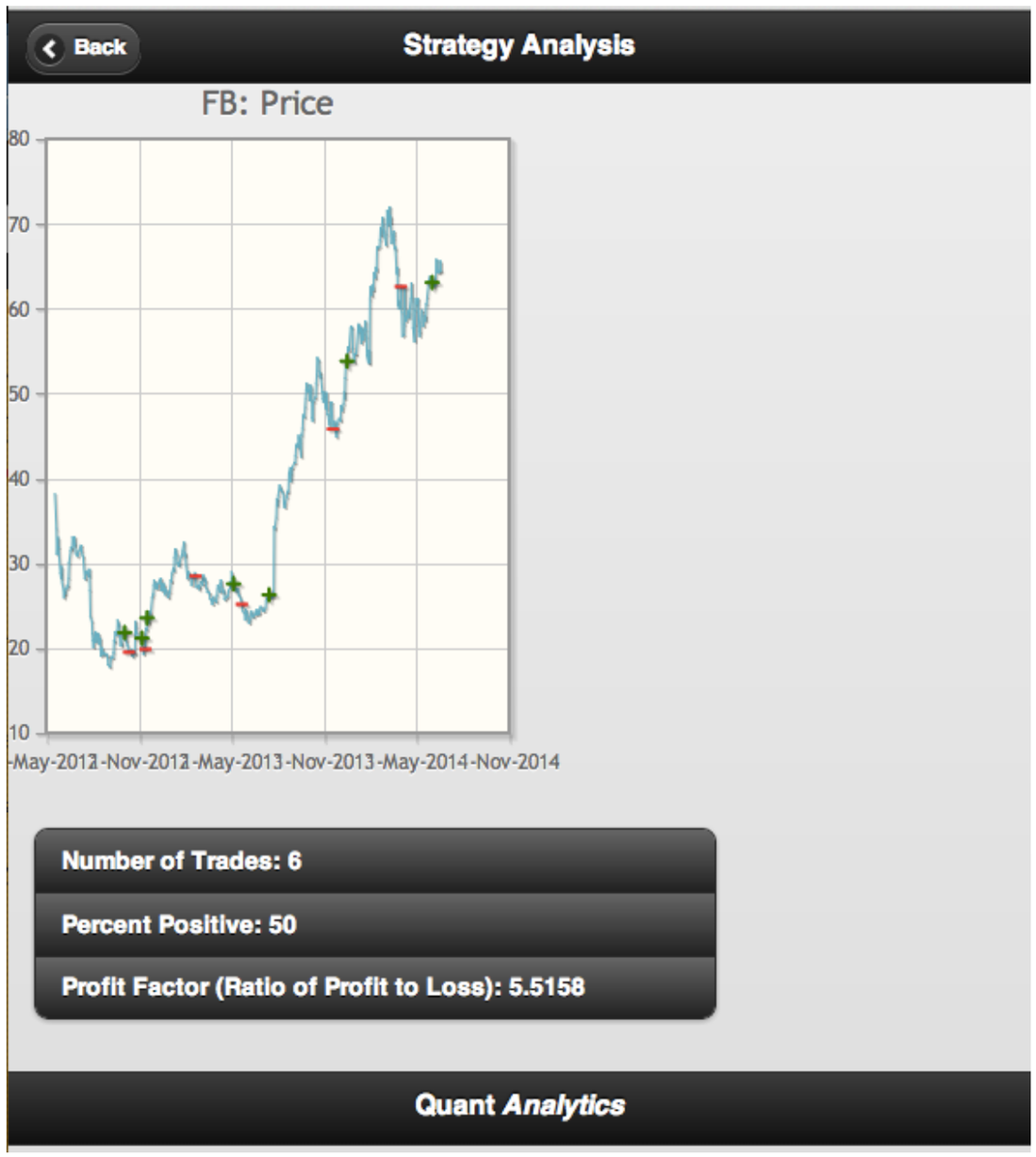

```
Strategy="MACD";
Symbol="FB";
RunStrategy=function (Strategy, Symbol) {
        var strURLbase= "http://54.245.107.178/cgi-bin/...
        var strURL= strURLbase.concat(Strategy);
        $.ajax({
                async : false,
                url : strURL,
                dataType : "jsonp",
                jsonp : 'callback',
                jsonpCallback : 'jsonpCallbackRunStrategy',
                data :
                {
                        symbl: StrategyCandidate
                },
                success : function(data) {
                        $.each(data, function(key, value) {
                                if (key == 'Num.Trades')
                                        {
                                                NumTrades=
        value;
                        …
                                        }
                        })
                }
        })

var plot = $.jqplot('plotStrategy', [line1,line2,line3],
        {…} );
```

1 Point to remote R function with parameters

2 Asynchronous Javascript call, continue UI execution, dont wait for R.

3 R returns results in JSONP. Parse data

4 Plot data with JS library



1 R response "Padded" with the same Callback function sent by JS.

2 JSONP response in Key/ Value format

```
jsonpCallbackRunStrategy({
    "Portfolio": "macd",
   "Symbol": "FB",
   "Num.Txns":    13,
   "Num.Trades": 6,
   "Net.Trading.PL":   2865,
   "Avg.Trade.PL": 453.83,
   "Txn.Date": [ "2012-01-01", "2012-10-03", "2012-10-12",
   "2012-11-06", "2012-11-13", "2012-11-16", "2013-02-20",
   "2013-05-06", "2013-05-22", "2013-07-15", "2013-11-18",
   "2013-12-16", "2014-04-01", "2014-06-02" ],
   "Txn.Qty": [    0,   100,  -100,   100,  -100,   100,  -100,
   100,  -100,   100,  -100,   100,  -100,   100 ],
   "Txn.Price": [    0,  21.83,  19.52,  21.17,  19.86,  23.56,
   28.46,  27.57,  25.16,  26.28,  45.83,  53.81,  62.62,  63.08 ],
   "Txn.PL": [    0,    0,  -231,    0,  -131,    0,   490,    0,
   -241,    0,  1955,    0,   881,    0 ],
   "PriceDate.Date": [ "2012-05-18", "2012-05-21",
   "2012-05-22","2014-03-13", "2014-03-14", …],
   "PriceDate.Price": [  38.23,  34.03,    31,    32,  33.03,  …]
})
```

```
run<-function(callback,symbl,...)
        {
        #Get required modules
        require(quantstrat)
        require(RJSONIO)

        #Get the callback function name; Send data back within this function
        callb <- substitute(callback); callb <-as.character(callb);
        stock.str<-symbl;

        #MA parameters for MACD
        fastMA = 12;slowMA = 26;signalMA = 9;maType="EMA";

        # define the strategy
        strategy(strat.st, store=TRUE)

        #Add indicators
        add.indicator(strat.st, name = "MACD",arguments = …)

        #Add signals, rules
         add.signal(strat.st,name="sigThreshold",…)
         add.rule(strat.st,name='ruleSignal',arguments=…)

         #Get latest data from Yahoo
         getSymbols(stock.str,from=initDate)

         #Apply strategy to data
         out<-applyStrategy(mktdata=x,strat.st ,
         portfolios=portfolio.st,parameters=list(nFast=fastMA, nSlow=slowMA,
         nSig=signalMA,maType=maType))

         #Send results back within the callback function
         jsonProb<-toJSON(output)
         jsonCallbackProb<-paste(callb,'(',jsonProb,')', sep='')
```

1 FastRWeb function Template

2 Quantstrat strategy

3 Return value in JSONP to remote Javascript client device

What's Next?

- UI Enhancement
- Social features
- User defined strategies

- Add/improve strategies,instruments
- Machine Learning/social algorithms
- Scale database, compute

amazon web services™
EC2 Cloud Installation

Feedback?
Interested?

Launch App!

Thank You!