Sprint 1

Straddle screening tool

Bassam Rizk - YU id 303525

Research Question

What is the highest reachable consistency in screening stocks that will exhibit abnormal price volatility at a foreseeable market or stock event (dividends announcement…)?

Candidate stocks should have their at-the-money options tree premiums exhibiting high correlation (positive for calls and/or negative for puts) with their underlying stocks' prices.

Data sources

- S&P 100 list of symbols - Wikipedia
- S&P 100 Stock historical (12 years) – API from Quantmod; QuantTools
- S&P 100 stock historical events – API from Quantmod
- Option Tree historical data (call on select stocks in sprint 2) – Quantmod; QuantTools
- Greeks historical data (call on select stocks in sprint 2) – API from fOptions

Update summary

# After a struggle in tying back API feeds from QuantTools & Quantmod – I entertained the idea of leveraging csv download of S&P 500

I refined my data set to S&P 500 and with hours on google I was able to find ways to marry both libraries

So I moved back to APIs from QuantMod & QuantTools (vs Kaggle)

- ✓ # Downloaded S&P100 stocks individually via "getSymbols"
- ✓ # From here onward this is a coding heavy exercise – as I had to maintain all 100 symbols separate in xts files to be able to use library functions
- ✓ #Added dividend & stock splits events so to model causality / correlation of events with abnormal volatility (head to clean the data from NAs)
- ✓ #calculated trailing volatility for each stock (This was a data – prep /cleanup pain!!)
- ✓ #transformed data to enable moving it from xts format to data frame – head to individually clean data of each column from factor to numeric – without making decimals as integers (that was a few hours process)
- ✓ # merged all 100 S&P stocks into a central clean data frame.

R Studio Libraries

library(quandl)

library(QuantTools)

library(quantmod)

library(derivmkts)

library(RND)

setDefaults(getSymbols.av, api.key="V7YC53BOMBUB28FJ")

Downloading Data

#There weren't any API to filter S&P100 stocks of all listed stocks. So, I got a list of S&P 100 from Wikipedia

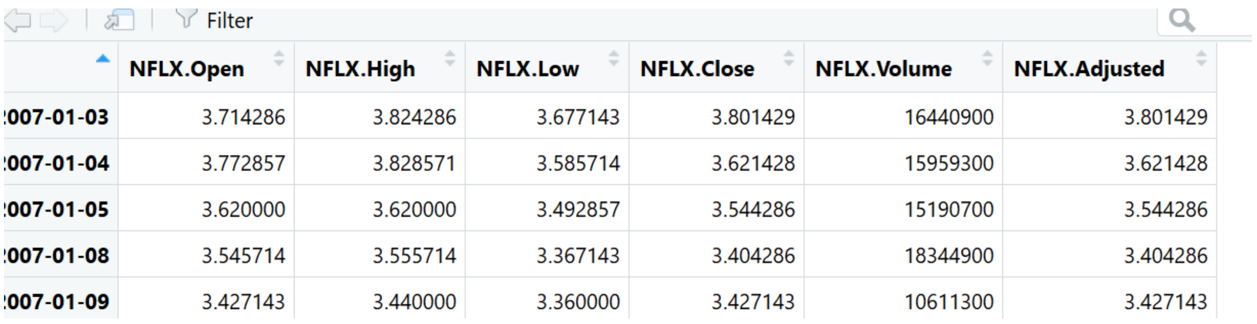
#Converted that list from table to csv using Excel / notepad (I tried doing that using R - it was too complicated...)

#Copy/ pasted that list into a getSymbols function from Quantmod.

getSymbols(c('AAPL', 'ABBV', 'ABT', 'ACN', 'ADBE', 'AGN', 'AIG', 'ALL', 'AMGN', 'AMZN', 'AXP', 'BA', 'BAC', 'BIIB', 'BK', 'BKNG', 'BLK', 'BMY', 'C', 'CAT', 'CELG', 'CHTR', 'CL', 'CMCSA', 'COF', 'COP', 'COST', 'CSCO', 'CVS', 'CVX', 'DD', 'DHR', 'DIS', 'DOW', 'DUK', 'EMR', 'EXC', 'F', 'FB', 'FDX', 'GD', 'GE', 'GILD', 'GM', 'GOOG', 'GOOGL', 'GS', 'HD', 'HON', 'IBM', 'INTC', 'JNJ', 'JPM', 'KHC', 'KMI', 'KO', 'LLY', 'LMT', 'LOW', 'MA', 'MCD', 'MDLZ', 'MDT', 'MET', 'MMM', 'MO', 'MRK', 'MS', 'MSFT', 'NEE', 'NFLX', 'NKE', 'NVDA', 'ORCL', 'OXY', 'PEP', 'PFE', 'PG', 'PM', 'PYPL', 'QCOM', 'RTN', 'SBUX', 'SLB', 'SO', 'SPG', 'T', 'TGT', 'TXN', 'UNH', 'UNP', 'UPS', 'USB', 'UTX', 'V', 'VZ', 'WBA', 'WFC', 'WMT', 'XOM'))

#This worked!!!!!! It took me couple of days to figure out how to call this to multiple symbols… plus it

# I received in my environment 100 xts objects with history from 2007-01-03 to 2019-08-02



Filter

	NFLX.Open	NFLX.High	NFLX.Low	NFLX.Close	NFLX.Volume	NFLX.Adjusted
007-01-03	3.714286	3.824286	3.677143	3.801429	16440900	3.801429
007-01-04	3.772857	3.828571	3.585714	3.621428	15959300	3.621428
007-01-05	3.620000	3.620000	3.492857	3.544286	15190700	3.544286
007-01-08	3.545714	3.555714	3.367143	3.404286	18344900	3.404286
007-01-09	3.427143	3.440000	3.360000	3.427143	10611300	3.427143

MEASURE VOLATILITY

I needed to add a trailing volatility measure of each stock for each time point.

#I added a column to measure volatility of each stock - I tried doing that in one function, it got too complicated.

AAPL$VOLA<- volatility(AAPL)

ABBV$VOLA<- volatility(ABBV)

… replicate to all 100 stocks

Clean up: #remove NAs FROM FIRST ROWS IN THE VOLATILITY COLUMN

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

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

… replicate to all 100 stocks

Add Symbol

## Add a column for ticker symbol - might be useful as we agreggate information

AAPL$Ticker <- NA

ABBV$Ticker <- NA

… replicate to all 100 stocks

Clean up Replace "NA"s with ticker symbol of each stock so I could aggregate the data at the end and have the stock symbol as an identifier of each row…

AAPL$Ticker[is.na(AAPL$Ticker)] <- "AAPL"

ABBV$Ticker[is.na(ABBV$Ticker)] <- "ABBV"

… replicate to all 100 stocks

Add Stock Events

# Track historical dividends & stock splits events for each stock – this would serve as a key trigger to identify correlation with price volatility… had to spend hours in testing multiple sources (Finam…) most were very unreliable (by either lack of correctness per each dividends or coverage per stock)… Yahoo seemed the most reliable.

SiDiAAPL<- get_yahoo_splits_and_dividends('AAPL','2007-01-03', '2019-08-02')

SiDiABBV<- get_yahoo_splits_and_dividends('ABBV','2007-01-03', '2019-08-02')

SiDiABT<- get_yahoo_splits_and_dividends('ABT','2007-01-03', '2019-08-02')

… replicate to all 100 stocks

USING DATES as a row label MERGE dividends & symbol xts file (aka stock historical data)– this was a tricky one – having dividends API from a different library wasn't compatible with the work I had…

AAPL<- merge(AAPL, SiDiAAPL)

ABBV<- merge(ABBV, SiDiAAPL)

ABT<- merge(ABT, SiDiAAPL)

… replicate to all 100 stocks

Clean-up after dividends & ticker addition

#REMOVE NAs FROM TICKER SYMBOLS

(Ticker symbol seems to be lost while doing the merge as this was an external field that I manually added and the merge worked because its is being sourced from the same API… I had to add it back.

AAPL$Ticker[is.na(AAPL$Ticker)] <- "AAPL"

ABBV$Ticker[is.na(ABBV$Ticker)] <- "ABBV"

… replicate to all 100 stocks

#REMOVE NAs FROM DIVIDEND VALUE

AAPL$value[is.na(AAPL$value)] <- 0

ABBV$value[is.na(ABBV$value)] <- 0

… replicate to all 100 stocks

Data Exploration

#Summary view of all stocks

summary (AAPL)

summary (ABBV)

… replicate to all 100 stocks

Key findings

####xts is being classified as factor - so will transfrom all the symbols xts into dataframes

#### than I will transform all columns (except date & ticker) into numeric

Clean-up: Transform Symbols into data frames as preparation to consolidate

AAPLfull<- data.frame(AAPL)

ABBVfull<- data.frame(ABBV)

… replicate to all 100 stocks

Cleanup: transform **Volatility** classification from factor into numeric

```
AAPLfull$VOLA<- as.numeric(as.character(AAPLfull$VOLA))

ABBVfull$VOLA<- as.numeric(as.character(ABBVfull$VOLA))
```

… replicate to all 100 stocks

Clean up: Transform **Open** column from factor to numeric

```
AAPLfull$AAPL.Open<- as.numeric(as.character(AAPLfull$AAPL.Open))

ABBVfull$ABBV.Open<- as.numeric(as.character(ABBVfull$ABBV.Open))
```

… replicate to all 100 stocks

Clean up: transform **High** column from factor to numeric

```
AAPLfull$AAPL.High<- as.numeric(as.character(AAPLfull$AAPL.High))

ABBVfull$ABBV.High<- as.numeric(as.character(ABBVfull$ABBV.High))
```

… replicate to all 100 stocks

Clean up: Transform **Low** Column classification from factor into numeric

```
AAPLfull$AAPL.Low<- as.numeric(as.character(AAPLfull$AAPL.Low))

ABBVfull$ABBV.Low<- as.numeric(as.character(ABBVfull$ABBV.Low))
```

… replicate to all 100 stocks

Clean up: Transform **Close** column classification from factor into numeric

```
AAPLfull$AAPL.Close<- as.numeric(as.character(AAPLfull$AAPL.Close))

ABBVfull$ABBV.Close<- as.numeric(as.character(ABBVfull$ABBV.Close))
```

… replicate to all 100 stocks

Clean up: Transform **Volume** column classification from factor into numeric

```
AAPLfull$AAPL.Volume<- as.numeric(as.character(AAPLfull$AAPL.Volume))

ABBVfull$ABBV.Volume<- as.numeric(as.character(ABBVfull$ABBV.Volume))

ABTfull$ABT.Volume<- as.numeric(as.character(ABTfull$ABT.Volume))
```

… replicate to all 100 stocks

Clean up: Transform **Adjusted** column classification from factor into numeric

```
AAPLfull$AAPL.Adjusted<- as.numeric(as.character(AAPLfull$AAPL.Adjusted))

ABBVfull$ABBV.Adjusted<- as.numeric(as.character(ABBVfull$ABBV.Adjusted))
```

… replicate to all 100 stocks

Clean up: Transform dividend **value** column's classification from factor to numeric

```
AAPLfull$value<- as.numeric(as.character(AAPLfull$value))
```

```
ABBVfull$value<- as.numeric(as.character(ABBVfull$value))
```

… replicate to all 100 stocks

Standardize nomenclature of header

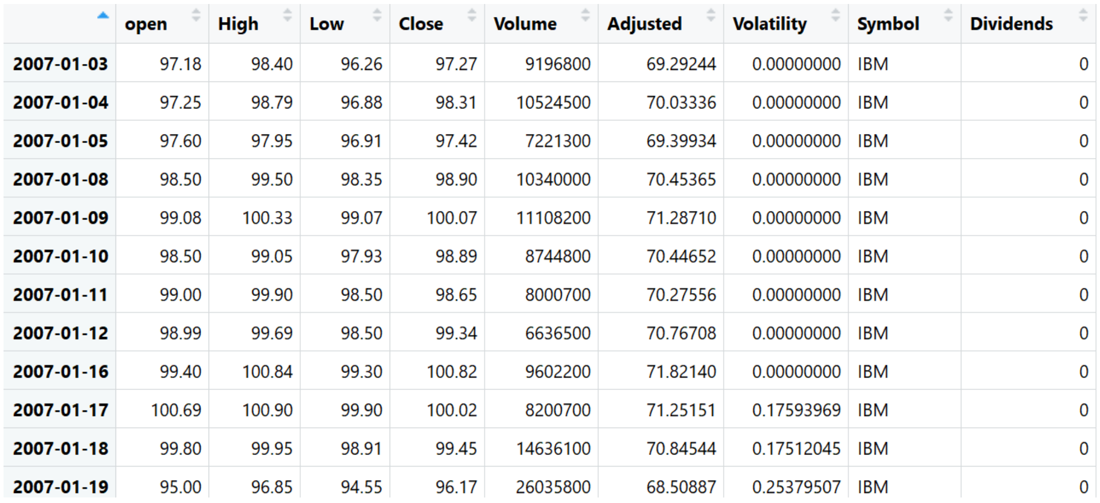
```
# This is done so we could merge the all data frames into 1 large one

setnames(AAPLfull, old=c("AAPL.Open", "AAPL.High", "AAPL.Low", "AAPL.Close", "AAPL.Volume", "AAPL.Adjusted", "VOLA", "Ticker", "value"), new=c("open", "High", "Low", "Close", "Volume", "Adjusted", "Volatility", "Symbol", "Dividends"))

setnames(ABBVfull, old=c("ABBV.Open", "ABBV.High", "ABBV.Low", "ABBV.Close", "ABBV.Volume", "ABBV.Adjusted", "VOLA", "Ticker", "value"), new=c("open", "High", "Low", "Close", "Volume", "Adjusted", "Volatility", "Symbol", "Dividends"))
```

… replicate to all 100 stocks

Below is sample stock data frame with no NAs, standardized headers and numeric value ready to be merged and analyzed



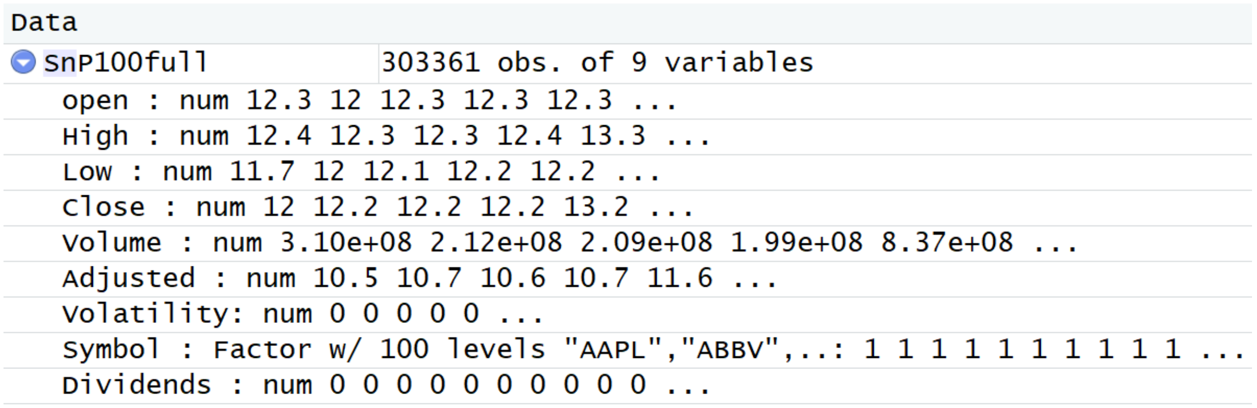
	open	High	Low	Close	Volume	Adjusted	Volatility	Symbol	Dividends
2007-01-03	97.18	98.40	96.26	97.27	9196800	69.29244	0.00000000	IBM	0
2007-01-04	97.25	98.79	96.88	98.31	10524500	70.03336	0.00000000	IBM	0
2007-01-05	97.60	97.95	96.91	97.42	7221300	69.39934	0.00000000	IBM	0
2007-01-08	98.50	99.50	98.35	98.90	10340000	70.45365	0.00000000	IBM	0
2007-01-09	99.08	100.33	99.07	100.07	11108200	71.28710	0.00000000	IBM	0
2007-01-10	98.50	99.05	97.93	98.89	8744800	70.44652	0.00000000	IBM	0
2007-01-11	99.00	99.90	98.50	98.65	8000700	70.27556	0.00000000	IBM	0
2007-01-12	98.99	99.69	98.50	99.34	6636500	70.76708	0.00000000	IBM	0
2007-01-16	99.40	100.84	99.30	100.82	9602200	71.82140	0.00000000	IBM	0
2007-01-17	100.69	100.90	99.90	100.02	8200700	71.25151	0.17593969	IBM	0
2007-01-18	99.80	99.95	98.91	99.45	14636100	70.84544	0.17512045	IBM	0
2007-01-19	95.00	96.85	94.55	96.17	26035800	68.50887	0.25379507	IBM	0

Merge all 100 data-frames into one data frame

```
SnP100full<- rbind(AAPLfull, ABBVfull, ABTfull, ACNfull, ADBEfull, AGNfull, AIGfull, ALLfull, AMGNfull, AMZNfull, AXPfull, BAfull, BACfull, BIIBfull, BKfull, BKNGfull, BLKfull, BMYfull, Cfull, CATfull, CELGfull, CHTRfull, CLfull, CMCSAfull, COFfull, COPfull, COSTfull, CSCOfull, CVSfull, CVXfull, DDfull, DHRfull, DISfull, DOWfull, DUKfull, EMRfull, EXCfull, Ffull, FBfull, FDXfull, GDfull, GEfull, GILDfull, GMfull, GOOGfull, GOOGLfull, GSfull, HDfull, HONfull, IBMfull, INTCfull, JNJfull, JPMfull, KHCfull, KMIfull, KOfull, LLYfull, LMTfull, LOWfull, MAfull, MCDfull, MDLZfull, MDTfull, METfull, MMMfull, MOfull, MRKfull, MSfull, MSFTfull, NEEfull, NFLXfull, NKEfull, NVDAfull, ORCLfull, OXYfull, PEPfull, PFEfull, PGfull, PMfull, PYPLfull, QCOMfull, RTNfull, SBUXfull, SLBfull,
```

SOfull, SPGfull, Tfull, TGTfull, TXNfull, UNHfull, UNPfull, UPSfull, USBfull, UTXfull, Vfull, VZfull, WBAfull, WFCfull, WMTfull, XOMfull)

#successfully merged all data sets and added ticker – I've been struggling in trying to call this function since 2 weeks now – it took rounds of clean-up and data transformation to capture that…



```
Data
SnP100full              303361 obs. of 9 variables
    open : num 12.3 12 12.3 12.3 12.3 ...
    High : num 12.4 12.3 12.3 12.4 13.3 ...
    Low : num 11.7 12 12.1 12.2 12.2 ...
    Close : num 12 12.2 12.2 12.2 13.2 ...
    Volume : num 3.10e+08 2.12e+08 2.09e+08 1.99e+08 8.37e+08 ...
    Adjusted : num 10.5 10.7 10.6 10.7 11.6 ...
    Volatility: num 0 0 0 0 0 ...
    Symbol : Factor w/ 100 levels "AAPL","ABBV",..: 1 1 1 1 1 1 1 1 1 1 ...
    Dividends : num 0 0 0 0 0 0 0 0 0 0 ...
```

summary(SnP100full)

```
     open             High              Low              Close        
 Min.   :   1.31   Min.   :   1.55   Min.   :   1.01   Min.   :   1.26  
 1st Qu.:  35.49   1st Qu.:  35.87   1st Qu.:  35.11   1st Qu.:  35.49  
 Median :  57.93   Median :  58.48   Median :  57.36   Median :  57.93  
 Mean   :  96.66   Mean   :  97.59   Mean   :  95.67   Mean   :  96.65  
 3rd Qu.:  93.22   3rd Qu.:  94.00   3rd Qu.:  92.37   3rd Qu.:  93.23  
 Max.   :2210.93   Max.   :2228.99   Max.   :2174.07   Max.   :2206.09  
 NA's   :56        NA's   :56        NA's   :56        NA's   :56       
     Volume             Adjusted          Volatility          Symbol      
 Min.   :0.000e+00   Min.   :   0.8996   Min.   :0.0000   AAPL   :  3169  
 1st Qu.:3.533e+06   1st Qu.:  29.1209   1st Qu.:0.1409   ABT    :  3169  
 Median :6.668e+06   Median :  49.1364   Median :0.2010   ACN    :  3169  
 Mean   :1.398e+07   Mean   :  88.5902   Mean   :0.2511   ADBE   :  3169  
 3rd Qu.:1.379e+07   3rd Qu.:  83.5500   3rd Qu.:0.2939   AGN    :  3169  
 Max.   :1.227e+09   Max.   :2206.0901   Max.   :7.0796   AIG    :  3169  
 NA's   :56          NA's   :56          NA's   :56       (Other):284347  
   Dividends     
 Min.   :0.0000  
 1st Qu.:0.0000  
 Median :0.0000  
 Mean   :0.0051  
 3rd Qu.:0.0000  
 Max.   :0.7700  
```

CLEAN-UP work-space Environment

At this point my R-studio and laptop became too slow because of the 100s of data frames

In addition, I wasn't able to leverage some of the specialized charts my libraries had because I had changed the xts files they had initially produced.

I deleted all initial symbols and call upon them again - that will enable me to call on the generic stock charts from fnancial R libraries

AAPL<- NULL

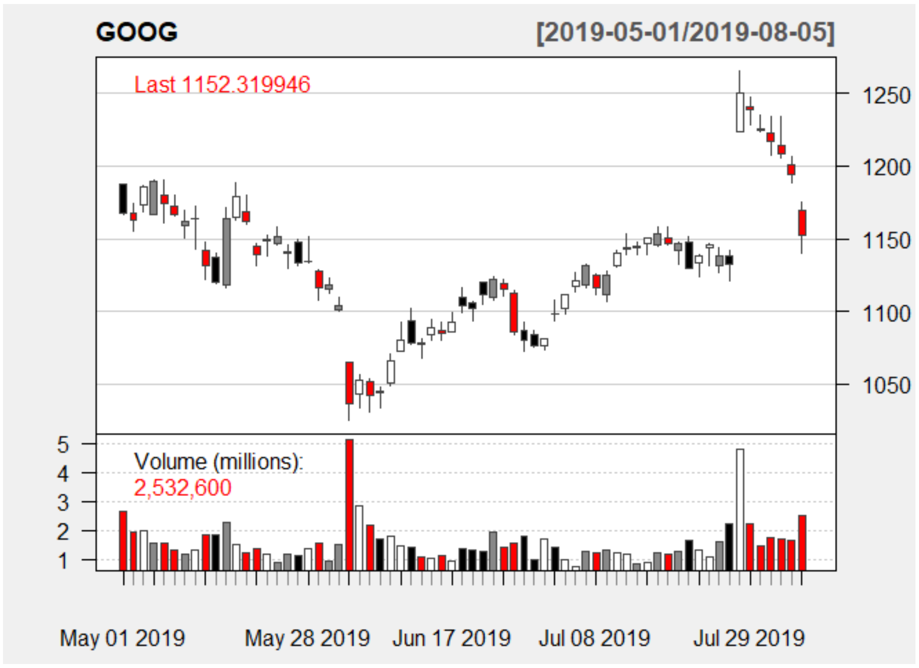
ABBV<- NULL

… replicate to all 100 stocks

#call the symbols back in their original format
getSymbols(c('AAPL', 'ABBV', 'ABT', 'ACN', 'ADBE', 'AGN', 'AIG', 'ALL', 'AMGN', 'AMZN', 'AXP', 'BA', 'BAC', 'BIIB', 'BK', 'BKNG', 'BLK', 'BMY', 'BRK.B', 'C', 'CAT', 'CELG', 'CHTR', 'CL', 'CMCSA', 'COF', 'COP', 'COST', 'CSCO', 'CVS', 'CVX', 'DD', 'DHR', 'DIS', 'DOW', 'DUK', 'EMR', 'EXC', 'F', 'FB', 'FDX', 'GD', 'GE', 'GILD', 'GM', 'GOOG', 'GOOGL', 'GS', 'HD', 'HON', 'IBM', 'INTC', 'JNJ', 'JPM', 'KHC', 'KMI', 'KO', 'LLY', 'LMT', 'LOW', 'MA', 'MCD', 'MDLZ', 'MDT', 'MET', 'MMM', 'MO', 'MRK', 'MS', 'MSFT', 'NEE', 'NFLX', 'NKE', 'NVDA', 'ORCL', 'OXY', 'PEP', 'PFE', 'PG', 'PM', 'PYPL', 'QCOM', 'RTN', 'SBUX', 'SLB', 'SO', 'SPG', 'T', 'TGT', 'TXN', 'UNH', 'UNP', 'UPS', 'USB', 'UTX', 'V', 'VZ', 'WBA', 'WFC', 'WMT', 'XOM'))

Charts

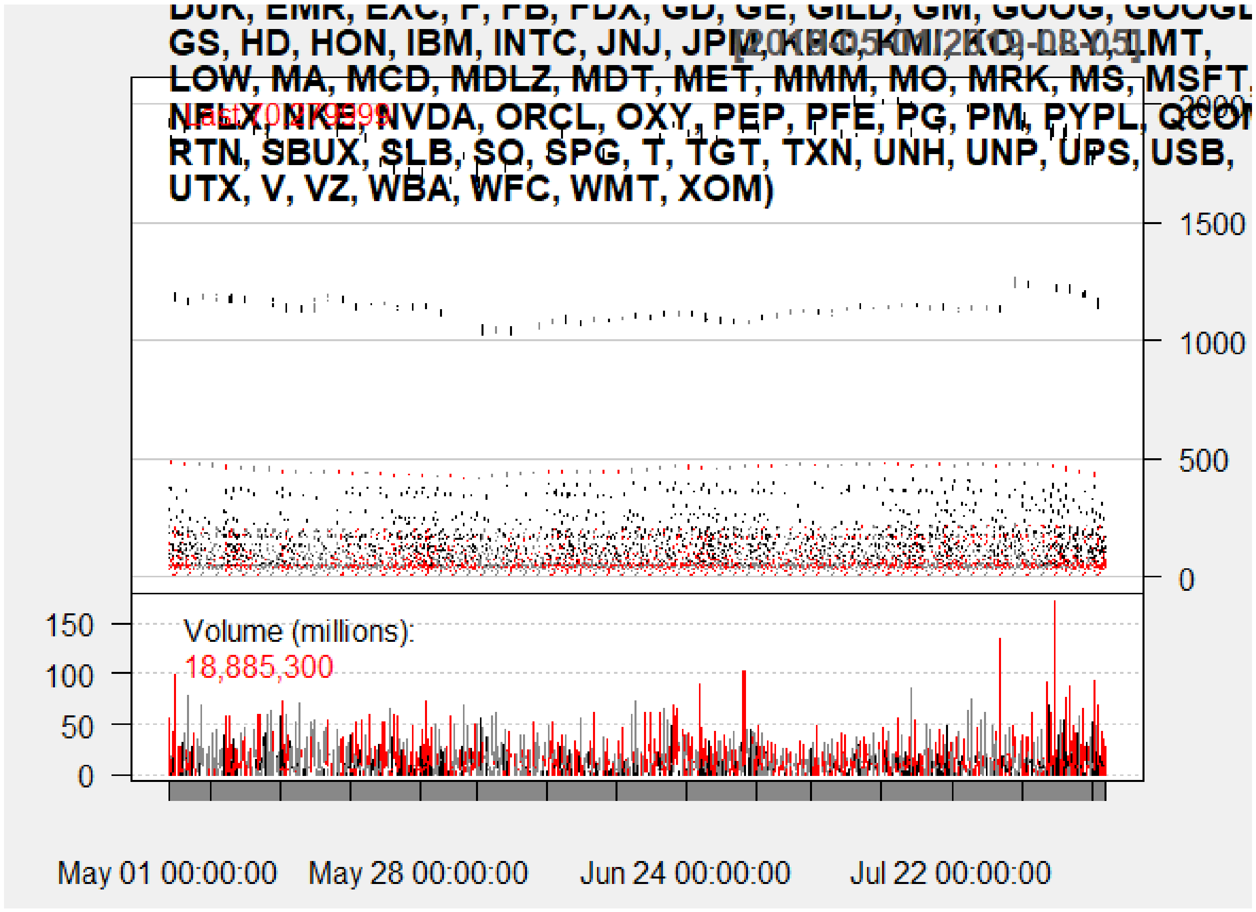
Sample 4 months view of a Google – a relatively volatile stockcandleChart(GOOG, subset= "last 4 months",multi.col=TRUE,theme="white", addMACD(fast = 12, slow = 26, signal = 9, type = "EMA"))



Spikes in volume definitely have effect on price – will test volume effect in clustering within sprint 2

Next will test a 4 months view across all 100 stocks

barChart(c(AAPL, ABBV, ABT, ACN, ADBE, AGN, AIG, ALL, AMGN, AMZN, AXP, BA, BAC, BIIB, BK, BKNG, BLK, BMY, C, CAT, CELG, CHTR, CL, CMCSA, COF, COP, COST, CSCO, CVS, CVX, DD, DHR, DIS, DOW, DUK, EMR, EXC, F, FB, FDX, GD, GE, GILD, GM, GOOG, GOOGL, GS, HD, HON, IBM, INTC, JNJ, JPM, KHC, KMI, KO, LLY, LMT, LOW, MA, MCD, MDLZ, MDT, MET, MMM, MO, MRK, MS, MSFT, NEE, NFLX, NKE, NVDA, ORCL, OXY, PEP, PFE, PG, PM, PYPL, QCOM, RTN, SBUX, SLB, SO, SPG, T, TGT, TXN, UNH, UNP, UPS, USB, UTX, V, VZ, WBA, WFC, WMT, XOM), subset="last 4 months" ,multi.col=TRUE,theme="white")



This is a very messy view and couldn’t conclude much in it – it’s all over the place.