Strategy Design (ML Fin Data - Project 1)

Hair Albeiro Parra Barrera

Libraries

0. Scraping the SP500

In order to test the logic within the strategy, I have fetched functions that retrieve a number of sample stocks by sector from the SP500. This is done automatically by `fetch_sp500_sectors.R`.

0.0.1 SP500 Economic Sectors

The following function fetches and extract the economic sectors from the SP500, taken from Wikipedia.

```
# fetch the sectors as a dataframe
sp500_sectors <- f_get_sp500_sectors()
head(sp500_sectors)
```

```
##   tickers                sectors
## 1     MMM             Industrials
## 2     AOS             Industrials
## 3     ABT             Health Care
## 4    ABBV             Health Care
## 5     ACN Information Technology
## 6    ATVI Communication Services
```

0.0.2 SP500 Sector Weight

```
# wrap into a single argument funciton
fetch_sp500_sector_data <- function(x){f_fetch_sector_data(x, sp500, sp500_sectors)}

# call the function
head(fetch_sp500_sector_data("Information Technology"))
```

```
##   ticker                 sector       weight shares_held
## 1   AAPL Information Technology 0.0721380790   160545598
## 2    ACN Information Technology 0.0053982462     6892028
## 3   ADBE Information Technology 0.0066170326     4980032
## 4    ADI Information Technology 0.0023836598     5478466
## 5   ADSK Information Technology 0.0012343184     2335139
## 6   AKAM Information Technology 0.0004426694     1667789
```

0.0.3 Retrieving top sectors and stocks

Pack everything into one function to retrieve all the data

```
# Retrieve top 10 stocks by weight for each sector in the top 5 sectors from the SP500 (by weight)
sector_list <- f_retrieve_top_sp500(top_n_sectors = 6, top_n_stocks = 20, only_tickers=TRUE)
sector_list
```

```
## $Industrials
##  [1] "ADP" "BA"  "CAT" "CSX" "DE"  "EMR" "ETN" "FDX" "GD"  "GE"  "HON" "ITW"
## [13] "LMT" "MMM" "NOC" "PH"  "RTX" "UNP" "UPS" "WM"
##
## $`Health Care`
##  [1] "ABBV" "ABT"  "AMGN" "BMY"  "CI"   "CVS"  "DHR"  "ELV"  "GILD" "ISRG"
## [11] "JNJ"  "LLY"  "MDT"  "MRK"  "PFE"  "REGN" "SYK"  "TMO"  "UNH"  "VRTX"
##
## $`Information Technology`
##  [1] "AAPL" "ACN"  "ADBE" "ADI"  "AMAT" "AMD"  "AVGO" "CRM"  "CSCO" "IBM"
## [11] "INTC" "INTU" "LRCX" "MSFT" "NOW"  "NVDA" "ORCL" "PANW" "QCOM" "TXN"
##
## $`Communication Services`
##  [1] "ATVI"  "CHTR"  "CMCSA" "DIS"   "EA"    "FOXA"  "GOOG"  "GOOGL" "IPG"
## [10] "LYV"   "META"  "MTCH"  "NFLX"  "NWSA"  "OMC"   "T"     "TMUS"  "TTWO"
## [19] "VZ"    "WBD"
##
## $Financials
##  [1] "AON"  "AXP"  "BAC"  "BLK"  "BX"   "C"    "CB"   "CME"  "FI"   "GS"
## [11] "ICE"  "JPM"  "MA"   "MMC"  "MS"   "PGR"  "SCHW" "SPGI" "V"    "WFC"
##
## $`Consumer Discretionary`
##  [1] "ABNB" "AMZN" "AZO"  "BKNG" "CMG"  "DHI"  "F"    "GM"   "HD"   "HLT"
## [11] "LEN"  "MAR"  "MCD"  "NKE"  "ORLY" "ROST" "SBUX" "TJX"  "TSLA" "YUM"
```

This logic is implemented under **functions/fetch_sp500_sectors.R**

0.0.4 Retrieving top sectors and stocks

```
# function to fetch all the information for one ticker into a nice xts dataframe
sp500_stocks <- lapply(sector_list,
                       f_fetch_all_tickers,
                       start_date="2016-01-01",
                       end_date="2022-12-01")
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ADP, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ADP, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker BA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker BA,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CAT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## CAT, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CSX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## CSX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker DE, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker DE,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker EMR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## EMR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ETN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ETN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker FDX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## FDX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker GD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker GD,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker GE, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker GE,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker HON, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## HON, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ITW, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ITW, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker LMT, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## LMT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker MMM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## MMM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker NOC, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## NOC, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker PH, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker PH,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker RTX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## RTX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker UNP, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## UNP, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker UPS, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## UPS, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker WM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker WM,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ABBV, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker BMY, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## BMY, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CI, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker CI,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CVS, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## CVS, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker DHR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## DHR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ELV, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ELV, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker GILD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## GILD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker LLY, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## LLY, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker MDT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## MDT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## MRK, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## PFE, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker REGN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## REGN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker SYK, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## SYK, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## TMO, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## UNH, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker VRTX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## VRTX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ACN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ACN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ADI, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ADI, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker AMAT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## AMAT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker AMD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## AMD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker AVGO, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## AVGO, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CRM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## CRM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker IBM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## IBM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker INTC, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## INTC, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker INTU, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## INTU, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## MSFT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker NOW, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## NOW, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## NVDA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ORCL, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker PANW, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## PANW, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker QCOM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## QCOM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker TXN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## TXN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ATVI, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ATVI, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CHTR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## CHTR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CMCSA, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## CMCSA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker DIS, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## DIS, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker EA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker EA,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker FOXA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## FOXA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker GOOG, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## GOOG, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker GOOGL, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## GOOGL, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker IPG, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## IPG, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker LYV, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## LYV, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## META, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker MTCH, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## MTCH, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker NFLX, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## NFLX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker NWSA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## NWSA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker OMC, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## OMC, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker T, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker T,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker TMUS, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## TMUS, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker TTWO, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## TTWO, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker VZ, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker VZ,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker WBD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## WBD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker AON, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## AON, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker BX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker BX,
## skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CB, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker CB,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CME, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## CME, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker FI, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker FI,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ICE, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ICE, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker MA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker MA,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker MMC, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## MMC, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker MS,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker PGR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## PGR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker SCHW, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## SCHW, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker SPGI, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## SPGI, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker V, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker V,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## WFC, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ABNB, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ABNB, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker AZO, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## AZO, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker BKNG, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## BKNG, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker CMG, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## CMG, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker DHI, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## DHI, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker GM,
## skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker HLT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## HLT, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker LEN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## LEN, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker MAR, skipping...
```

```
## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## MAR, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker MCD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## MCD, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## NKE, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ORLY, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ORLY, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker ROST, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## ROST, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## SBUX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker TJX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## TJX, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## TSLA, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No financial ratio data
## for ticker YUM, skipping...

## Warning in f_fetch_ind_base(x, from = from, to = to): No IV data for ticker
## YUM, skipping...
```

```
# clean the environment memory
xts_fama_french <- NULL
xts_financial_ratios <- NULL
xts_realized_vol <- NULL
```

```
# Show the available sectors
names(sp500_stocks)
```

```
## [1] "Industrials"            "Health Care"            "Information Technology"
## [4] "Communication Services" "Financials"             "Consumer Discretionary"
```

```
# Show available stocks for Industrials
names(sp500_stocks$Industrials)
```

```
##  [1] "ADP" "BA"  "CAT" "CSX" "DE"  "EMR" "ETN" "FDX" "GD"  "GE"  "HON" "ITW"
## [13] "LMT" "MMM" "NOC" "PH"  "RTX" "UNP" "UPS" "WM"
```

```
# access the xts of the stocks in industrials
tail(sp500_stocks$Industrials[[5]])
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2022-10-26       386.3109             -1      0.053484935     -0.013460026
## 2022-11-02       381.1460              1     -0.013369845      0.028455218
## 2022-11-09       392.1474              1      0.028863935      0.023174802
## 2022-11-16       401.3414              1      0.023445424      0.073784581
## 2022-11-23       432.0741              1      0.076574865      0.007922509
## 2022-11-30       435.5108             NA      0.007953975               NA
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2022-10-26      0.052103654       0.02760464       0.01517513       0.03140689
## 2022-11-02     -0.013460026       0.05210365       0.02760464       0.01517513
## 2022-11-09      0.028455218      -0.01346003       0.05210365       0.02760464
## 2022-11-16      0.023174802       0.02845522      -0.01346003       0.05210365
## 2022-11-23      0.073784581       0.02317480       0.02845522      -0.01346003
## 2022-11-30      0.007922509       0.07378458       0.02317480       0.02845522
##                 atr      adx aaron        bb chaikin_vol         clv        emv
## 2022-10-26 16.65889 10.90895   100 0.9253776  -0.7033540 -0.09383278 0.02591424
## 2022-11-02 16.30325 11.44200   100 0.8612485  -3.0070669 -0.24924990 0.06672785
## 2022-11-09 16.49302 12.20740    50 0.8917193   1.1519438 -0.35705376 0.16789580
## 2022-11-16 16.13566 13.04944   100 0.8988852  -0.8350064 -0.23171407 0.20368870
## 2022-11-23 17.98311 15.00531   100 1.0842430  13.4687113 -0.21044883 0.42019450
## 2022-11-30 17.32503 16.82149    50 1.0301560  -0.4276570 -0.01897729 0.53655500
##                    macd      mfi      sar      smi  volume     volat
## 2022-10-26 -0.366656927 65.40085 317.7989 12.41054 1157500 0.2062547
## 2022-11-02  0.002997301 56.44849 322.4772 15.17568 1719300 0.2189202
## 2022-11-09  0.414252559 59.56372 328.4654 18.43638 2182800 0.2277602
## 2022-11-16  0.867010039 59.83537 336.1099 22.58421 1101600 0.2253009
## 2022-11-23  1.447660474 67.42008 344.8063 27.55272 5080300 0.2610497
## 2022-11-30  2.082816118 69.08992 359.3094 32.75519 2397200 0.2627691
##            month_index Excess_Retun_Mkt Small_minus_Big High_minus_Low
## 2022-10-26          82          -0.0066          0.0070         0.0089
## 2022-11-02          83          -0.0267         -0.0087         0.0161
## 2022-11-09          83          -0.0225         -0.0052         0.0055
## 2022-11-16          83          -0.0103         -0.0107         0.0057
## 2022-11-23          83           0.0063         -0.0024        -0.0094
## 2022-11-30          83           0.0312         -0.0015        -0.0207
##            Robus_minus_Weak Conservative_minus_Aggressive Risk_free_rate
## 2022-10-26          -0.0080                        0.0067        0.00011
## 2022-11-02           0.0021                        0.0105        0.00014
## 2022-11-09           0.0095                        0.0106        0.00014
## 2022-11-16           0.0119                        0.0093        0.00014
## 2022-11-23          -0.0075                       -0.0057        0.00014
## 2022-11-30          -0.0077                       -0.0141        0.00014
##            Momentum
## 2022-10-26   0.0049
## 2022-11-02   0.0216
## 2022-11-09   0.0164
## 2022-11-16   0.0269
## 2022-11-23  -0.0184
## 2022-11-30  -0.0282
```

BACKTESTING LOGIC

Adding a numeric index

The data-fetching logic includes addition of a numerical index indicating to which month in the simulation the observations belong.

```
# count number of weeks in data from one of the dataframes
sample_xts <- sp500_stocks$Industrials$CSX
tail(sample_xts, 10)
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2022-09-28       27.24850              1     -0.051818809      0.006853096
## 2022-10-05       27.43588             -1      0.006876632     -0.042966012
## 2022-10-12       26.28204              1     -0.042056052      0.046554042
## 2022-10-19       27.53450              1      0.047654694      0.029990060
## 2022-10-26       28.37277             -1      0.030444291     -0.008377096
## 2022-11-02       28.13608              1     -0.008342106      0.031058390
## 2022-11-09       29.02365              1      0.031545734      0.059684720
## 2022-11-16       30.80866              1      0.061501824      0.026221770
## 2022-11-23       31.62720              1      0.026568585      0.022307721
## 2022-11-30       32.34066             NA      0.022558399               NA
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2022-09-28     -0.053209666     -0.069267411     -0.020913290      0.007554347
## 2022-10-05      0.006853096     -0.053209666     -0.069267411     -0.020913290
## 2022-10-12     -0.042966012      0.006853096     -0.053209666     -0.069267411
## 2022-10-19      0.046554042     -0.042966012      0.006853096     -0.053209666
## 2022-10-26      0.029990060      0.046554042     -0.042966012      0.006853096
## 2022-11-02     -0.008377096      0.029990060      0.046554042     -0.042966012
## 2022-11-09      0.031058390     -0.008377096      0.029990060      0.046554042
## 2022-11-16      0.059684720      0.031058390     -0.008377096      0.029990060
## 2022-11-23      0.026221770      0.059684720      0.031058390     -0.008377096
## 2022-11-30      0.022307721      0.026221770      0.059684720      0.031058390
##                 atr      adx aaron         bb chaikin_vol         clv
## 2022-09-28 1.441481 16.24190  -100 0.04467755  2.43234200  0.21475805
## 2022-10-05 1.384232 17.10559   -50 0.13495813 -0.44268680  0.22116568
## 2022-10-12 1.379644 18.24157   -50 0.07457368  0.43839330  0.07934922
## 2022-10-19 1.394670 18.58490    50 0.23730603 -1.12835800  0.03125187
## 2022-10-26 1.398622 18.20787   100 0.36428555  0.36773750 -0.10430028
## 2022-11-02 1.385863 17.63796   100 0.36718737 -8.91414900 -0.26417408
## 2022-11-09 1.385444 17.00435    50 0.43456871 -0.08886197 -0.35167976
## 2022-11-16 1.429341 16.04316   100 0.61239403 -0.69757770 -0.28307675
## 2022-11-23 1.395102 15.54651   100 0.68335600 -2.77541900 -0.16462184
## 2022-11-30 1.369024 15.36369   100 0.70213009 -0.65517410  0.02947430
##                        emv      macd      mfi      sar       smi   volume
## 2022-09-28 -1.787304e-04 -2.031918 46.90353 34.67000 -18.01681 18306500
## 2022-10-05 -2.096124e-04 -2.290153 46.43088 34.38840 -22.89976 16028700
## 2022-10-12 -3.472192e-04 -2.649750 46.62430 34.11806 -28.89441 13763100
## 2022-10-19 -3.458817e-04 -2.983549 54.92321 33.66998 -32.89471 15446400
## 2022-10-26 -2.858648e-04 -3.232381 56.20916 33.24878 -34.78229 21083400
## 2022-11-02 -1.913069e-04 -3.420978 48.82911 32.85285 -36.26677 15289700
## 2022-11-09 -1.696224e-04 -3.505779 48.94612 32.48068 -36.24474 10546600
## 2022-11-16 -6.177828e-05 -3.415472 46.83053 32.13084 -32.84559 10016300
## 2022-11-23  6.920197e-05 -3.168499 45.87661 26.65000 -26.53377  9659000
## 2022-11-30  2.043992e-04 -2.797269 55.72098 26.65000 -18.89848 24182500
##                   volat month_index Excess_Retun_Mkt Small_minus_Big
## 2022-09-28 0.2279791          81           0.0215          0.0092
## 2022-10-05 0.2353109          82          -0.0022         -0.0037
## 2022-10-12 0.2481376          82          -0.0027          0.0002
```

```
## 2022-10-19 0.2465206          82          -0.0087         -0.0120
## 2022-10-26 0.2484444          82          -0.0066          0.0070
## 2022-11-02 0.2806964          83          -0.0267         -0.0087
## 2022-11-09 0.2819226          83          -0.0225         -0.0052
## 2022-11-16 0.2767814          83          -0.0103         -0.0107
## 2022-11-23 0.2587499          83           0.0063         -0.0024
## 2022-11-30 0.2672197          83           0.0312         -0.0015
##            High_minus_Low Robus_minus_Weak Conservative_minus_Aggressive
## 2022-09-28        -0.0033          -0.0087                        -0.0071
## 2022-10-05         0.0006           0.0035                         0.0016
## 2022-10-12         0.0002          -0.0002                         0.0001
## 2022-10-19         0.0121           0.0070                         0.0077
## 2022-10-26         0.0089          -0.0080                         0.0067
## 2022-11-02         0.0161           0.0021                         0.0105
## 2022-11-09         0.0055           0.0095                         0.0106
## 2022-11-16         0.0057           0.0119                         0.0093
## 2022-11-23        -0.0094          -0.0075                        -0.0057
## 2022-11-30        -0.0207          -0.0077                        -0.0141
##            Risk_free_rate Momentum
## 2022-09-28        0.00009  -0.0135
## 2022-10-05        0.00011   0.0049
## 2022-10-12        0.00011  -0.0060
## 2022-10-19        0.00011   0.0196
## 2022-10-26        0.00011   0.0049
## 2022-11-02        0.00014   0.0216
## 2022-11-09        0.00014   0.0164
## 2022-11-16        0.00014   0.0269
## 2022-11-23        0.00014  -0.0184
## 2022-11-30        0.00014  -0.0282
```

```
sample_xts[, c( "month_index")]
```

```
##            month_index
## 2016-01-06           1
## 2016-01-13           1
## 2016-01-20           1
## 2016-01-27           1
## 2016-02-03           2
## 2016-02-10           2
## 2016-02-17           2
## 2016-02-24           2
## 2016-03-02           3
## 2016-03-09           3
##        ...
## 2022-09-28          81
## 2022-10-05          82
## 2022-10-12          82
## 2022-10-19          82
## 2022-10-26          82
## 2022-11-02          83
## 2022-11-09          83
## 2022-11-16          83
## 2022-11-23          83
## 2022-11-30          83
```

BACKTESTING_PROCEDURE

1. Assume we have N_{years} years of weekly data, giving a total of N_{months} many months. 2. We want to fix a window of $N_W = 12$ months at the time (i.e. a year of data).

2. The total number of runs is given by

$$N^{runs} = \left\lfloor \frac{N_{months} - N_W}{s} \right\rfloor + 1$$

, where $s = 1$ is the number of months to move at the time (because of monthly rebalance).

i.e., we can move N^{runs} times when predicting one month at the time, starting with having all the data until month 12.

That is, $\tau = 1, \ldots, 48$

```
# Set up backtesting simulation parameters
sample_xts <- sp500_stocks$Industrials$ADP
sectors <- names(sp500_stocks)
N_sector_best_stocks <- 3 # new strategy: 3x2 = 6

# Formula parameters
slide <- 1
N_months <- length(names(split.xts(sample_xts, f= "months")))
N_window <- 18 # number of months in size for each window
N_runs <- floor((N_months - N_window)/slide)

# display parameters
print(paste0("N_months: ", N_months))
```

```
## [1] "N_months: 83"
```

```
print(paste0("N_runs: ", N_runs))
```

```
## [1] "N_runs: 65"
```

```
print(paste0("slide: ", slide))
```

```
## [1] "slide: 1"
```

```
# setup initial portfolio tracking variables
initial_capital <- 500000
num_tickers <- length(sectors)*N_sector_best_stocks*2 # two sub-strategies for picking
initial_tickers <- rep(NA, num_tickers)
weights <- rep(1/num_tickers, num_tickers) # initialize to 1/n
returns <- rep(NA, N_runs)

# repack the portfolio
portfolio <- list(tickers = initial_tickers,
                  weights = weights,
                  capital = initial_capital,
                  returns = returns,
                  data = NA
                  )
portfolio
```

```
## $tickers
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [26] NA NA NA NA NA NA NA NA NA NA NA
##
## $weights
##  [1] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
```

```
##  [7] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
## [13] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
## [19] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
## [25] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
## [31] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
##
## $capital
## [1] 5e+05
##
## $returns
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [26] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [51] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
##
## $data
## [1] NA
```

```
# Initiate backtesting
print(paste(rep("-", 100), collapse = ""))
```

```
## [1] "----------------------------------------------------------------------------------------------------"
```

```
print("BACKTESTING")
```

```
## [1] "BACKTESTING"
```

```
print(paste(rep("-", 100), collapse = ""))
```

```
## [1] "----------------------------------------------------------------------------------------------------"
```

```
print("")
```

```
## [1] ""
```

```
# for every run (sliding window of time to consider)
for(tau in seq(N_runs)){
  # close any positions
  print("################")
  print(paste0("### (tau=", tau, ") ###"))
  print("################")
  print("CLOSE all positions")

  # Calculate and record profit-loss
  print("(1) COMPUTE_P/L(portfolio)")
  portfolio$capital <- portfolio$capital * (1 + runif(1, -0.05, 0.10))
  print(paste0("--> Capital:", portfolio$capital, "$"))

  # variables
  i_sector <- 1 # keep index counter for sectors
  num_top_pick <- N_sector_best_stocks*2 # number of stocks picked per sector

  # current portf
  cur_tickers <- rep(NA, num_tickers)

  print("")
  print("(2) PORTFOLIO_LOOP:")
```

```
  # loop through all the sectors
  for(G in sectors){
    # execute sector procedure
    print(paste0("    SECTOR_PROCEDURE(G=", G, ", tau=",tau, ")"))

    # return top 3 best stocks according to procedure
    top_sector_stocks <- sample(names(sp500_stocks[[G]]), num_top_pick)

    # assign best stocks to portfolio (NEED TO UPDATE LOGIC!)
    i_replace <- rep(i_sector, num_top_pick) + seq(0, num_top_pick-1) # indexes to choose from
    cur_tickers[i_replace] <- top_sector_stocks
    i_sector <- i_sector + num_top_pick
  }

  # Assign tickers for this simulation
  portfolio$tickers <- as.vector(cur_tickers)

  # Display selected portfolio tickers
  print("Cur Portfolio:")
  print(portfolio$tickers)

  # Optimize portfolio weights using modified min_variance
  print("")
  print("(3) OPTIMIZE_PORTFOLIO(portfolio)")
  # simulate the optimization
  portfolio$weights <- runif(length(portfolio$weights)) / sum(runif(length(portfolio$weights)))
  print("weights: ")
  print(paste(" ", portfolio$weights))

  print("")
  print("(4) LONG PORTFOLIO()")

  # Separate similuation (over)
  print(paste(rep("-", 100), collapse = ""))

  # TEST: Just for this small printing simulation !!
  if(tau > 4){
    break
  }
}
```

```
## [1] "###############"
## [1] "### (tau=1) ###"
## [1] "###############"
## [1] "CLOSE all positions"
## [1] "(1) COMPUTE_P/L(portfolio)"
## [1] "--> Capital:514398.439443903$"
## [1] ""
## [1] "(2) PORTFOLIO_LOOP:"
## [1] "    SECTOR_PROCEDURE(G=Industrials, tau=1)"
## [1] "    SECTOR_PROCEDURE(G=Health Care, tau=1)"
## [1] "    SECTOR_PROCEDURE(G=Information Technology, tau=1)"
## [1] "    SECTOR_PROCEDURE(G=Communication Services, tau=1)"
## [1] "    SECTOR_PROCEDURE(G=Financials, tau=1)"
## [1] "    SECTOR_PROCEDURE(G=Consumer Discretionary, tau=1)"
## [1] "Cur Portfolio:"
##  [1] "ITW"   "PH"    "CAT"   "LMT"   "HON"   "EMR"   "GILD"  "CI"    "PFE"
## [10] "ABBV"  "MRK"   "MDT"   "IBM"   "AMAT"  "AAPL"  "LRCX"  "ACN"   "INTC"
## [19] "NFLX"  "META"  "CMCSA" "TMUS"  "MTCH"  "OMC"   "MA"    "PGR"   "SCHW"
```

```
## [28] "CB"    "CME"   "FI"    "NKE"   "YUM"   "SBUX"  "LEN"   "HLT"   "F"
## [1] ""
## [1] "(3) OPTIMIZE_PORTFOLIO(portfolio)"
## [1] "weights: "
##  [1] "  0.0207497029851801"  "  0.0127641249398861"  "  0.0189940351346527"
##  [4] "  0.00470381873495352" "  0.00586964005916823" "  0.00550971704014758"
##  [7] "  0.0256771705194327"  "  0.0435498426517931"  "  0.0352062134802418"
## [10] "  0.00373308845696858" "  0.0508591422307508"  "  0.0249637360387276"
## [13] "  0.0102498303461893"  "  0.0324374317877958"  "  0.0505227822774149"
## [16] "  0.0470809309647434"  "  0.0269554642515344"  "  0.00959789964629614"
## [19] "  0.0321141667534063"  "  0.0324472987659578"  "  0.0429501514913486"
## [22] "  0.0405448971445847"  "  0.0264808515215116"  "  0.0462632305328128"
## [25] "  0.00807995980555032" "  0.0384087827878298"  "  0.018075576187117"
## [28] "  0.0225280420722378"  "  0.0477253037619536"  "  0.0492879620864143"
## [31] "  0.0146655147334975"  "  0.0452027343998356"  "  0.0185993244304504"
## [34] "  0.0525203109803629"  "  0.0350219839285189"  "  0.0537631859746642"
## [1] ""
## [1] "(4) LONG PORTFOLIO()"
## [1] "--------------------------------------------------------------------------------------------------------"
## [1] "###############"
## [1] "### (tau=2) ###"
## [1] "###############"
## [1] "CLOSE all positions"
## [1] "(1) COMPUTE_P/L(portfolio)"
## [1] "--> Capital:549554.47119339$"
## [1] ""
## [1] "(2) PORTFOLIO_LOOP:"
## [1] "    SECTOR_PROCEDURE(G=Industrials, tau=2)"
## [1] "    SECTOR_PROCEDURE(G=Health Care, tau=2)"
## [1] "    SECTOR_PROCEDURE(G=Information Technology, tau=2)"
## [1] "    SECTOR_PROCEDURE(G=Communication Services, tau=2)"
## [1] "    SECTOR_PROCEDURE(G=Financials, tau=2)"
## [1] "    SECTOR_PROCEDURE(G=Consumer Discretionary, tau=2)"
## [1] "Cur Portfolio:"
##  [1] "ETN"   "UNP"   "PH"    "ADP"   "EMR"   "DE"    "CVS"   "AMGN"  "JNJ"
## [10] "SYK"   "UNH"   "REGN"  "LRCX"  "INTU"  "ACN"   "ORCL"  "NOW"   "AMD"
## [19] "WBD"   "ATVI"  "CMCSA" "TTWO"  "MTCH"  "DIS"   "ICE"   "AXP"   "MS"
## [28] "V"     "GS"    "C"     "ROST"  "ORLY"  "GM"    "NKE"   "F"     "YUM"
## [1] ""
## [1] "(3) OPTIMIZE_PORTFOLIO(portfolio)"
## [1] "weights: "
##  [1] "  0.0401785312920211"   "  0.0496956975649703"   "  0.0416183714006695"
##  [4] "  0.0306040024262047"   "  0.0417178717410298"   "  0.0484753089301125"
##  [7] "  0.0030656256952222"   "  0.0231089473823961"   "  0.0356886522635086"
## [10] "  0.0485790181763477"   "  0.00755377488084258"  "  0.027064113794399"
## [13] "  0.0574102714315212"   "  0.030569208566625"    "  0.0356765706741654"
## [16] "  0.0462783038416148"   "  0.0158584058560595"   "  0.0272297002465736"
## [19] "  0.0554160066933546"   "  0.0496049903527214"   "  0.0464398236333135"
## [22] "  0.000780661072953871" "  0.0091077266217007"   "  0.0100772685025123"
## [25] "  0.015932538674253"    "  0.0540947649136729"   "  0.0182397026195977"
## [28] "  0.00865240195203857"  "  0.0264529787913168"   "  0.000135701410947946"
## [31] "  0.0060318625404483"   "  0.0231198468805895"   "  0.0328891164243043"
## [34] "  0.0324452423410819"   "  0.0502134540671272"   "  0.0134603660166527"
## [1] ""
## [1] "(4) LONG PORTFOLIO()"
## [1] "--------------------------------------------------------------------------------------------------------"
## [1] "###############"
## [1] "### (tau=3) ###"
## [1] "###############"
## [1] "CLOSE all positions"
```

```
## [1] "(1) COMPUTE_P/L(portfolio)"
## [1] "--> Capital:590475.616844835$"
## [1] ""
## [1] "(2) PORTFOLIO_LOOP:"
## [1] "    SECTOR_PROCEDURE(G=Industrials, tau=3)"
## [1] "    SECTOR_PROCEDURE(G=Health Care, tau=3)"
## [1] "    SECTOR_PROCEDURE(G=Information Technology, tau=3)"
## [1] "    SECTOR_PROCEDURE(G=Communication Services, tau=3)"
## [1] "    SECTOR_PROCEDURE(G=Financials, tau=3)"
## [1] "    SECTOR_PROCEDURE(G=Consumer Discretionary, tau=3)"
## [1] "Cur Portfolio:"
##  [1] "CAT"   "BA"    "GE"    "DE"    "WM"    "FDX"   "PFE"   "MRK"   "DHR"
## [10] "MDT"   "LLY"   "ABT"   "NOW"   "NVDA"  "ADBE"  "MSFT"  "TXN"   "AVGO"
## [19] "CHTR"  "EA"    "CMCSA" "GOOGL" "ATVI"  "TTWO"  "BX"    "FI"    "BLK"
## [28] "ICE"   "SCHW"  "BAC"   "ROST"  "TSLA"  "AMZN"  "GM"    "ORLY"  "HLT"
## [1] ""
## [1] "(3) OPTIMIZE_PORTFOLIO(portfolio)"
## [1] "weights: "
##  [1] "  0.0156858460810247"  "  0.0102436440406541"  "  0.0434926642216094"
##  [4] "  0.0285927247648165"  "  0.0341302607344187"  "  0.00697301810227253"
##  [7] "  0.0293426877540643"  "  0.0244007649874976"  "  0.0322338748239995"
## [10] "  0.00474113480454827" "  0.00179991223224701" "  0.0178229360602156"
## [13] "  0.0418649308975857"  "  0.00881995554918558" "  0.0351574967209249"
## [16] "  0.0425929991149377"  "  0.0258651483483022"  "  0.0380654594556954"
## [19] "  0.0117618565235548"  "  0.0148361769863946"  "  0.0296638905835905"
## [22] "  0.0018546566983571"  "  0.0316176648038681"  "  0.0121591139463601"
## [25] "  0.0233486777450318"  "  0.0475137344343488"  "  0.0226814785209041"
## [28] "  0.0343945492409521"  "  0.0480576986051591"  "  0.0549464434285251"
## [31] "  0.0211373543484158"  "  0.0145085664913367"  "  0.00703623679553853"
## [34] "  0.0262777024666608"  "  0.0138537827394307"  "  0.00675017682783522"
## [1] ""
## [1] "(4) LONG PORTFOLIO()"
## [1] "-------------------------------------------------------------------------------------------------------"
## [1] "###############"
## [1] "### (tau=4) ###"
## [1] "###############"
## [1] "CLOSE all positions"
## [1] "(1) COMPUTE_P/L(portfolio)"
## [1] "--> Capital:605393.614548772$"
## [1] ""
## [1] "(2) PORTFOLIO_LOOP:"
## [1] "    SECTOR_PROCEDURE(G=Industrials, tau=4)"
## [1] "    SECTOR_PROCEDURE(G=Health Care, tau=4)"
## [1] "    SECTOR_PROCEDURE(G=Information Technology, tau=4)"
## [1] "    SECTOR_PROCEDURE(G=Communication Services, tau=4)"
## [1] "    SECTOR_PROCEDURE(G=Financials, tau=4)"
## [1] "    SECTOR_PROCEDURE(G=Consumer Discretionary, tau=4)"
## [1] "Cur Portfolio:"
##  [1] "EMR"  "PH"   "ADP"  "RTX"  "UNP"  "BA"   "MDT"  "VRTX" "BMY"  "TMO"
## [11] "PFE"  "AMGN" "PANW" "INTC" "IBM"  "TXN"  "NVDA" "AAPL" "IPG"  "NWSA"
## [21] "T"    "MTCH" "DIS"  "LYV"  "WFC"  "V"    "BAC"  "AON"  "MMC"  "FI"
## [31] "NKE"  "TSLA" "DHI"  "BKNG" "LEN"  "AZO"
## [1] ""
## [1] "(3) OPTIMIZE_PORTFOLIO(portfolio)"
## [1] "weights: "
##  [1] "  0.00706994007772683" "  0.0205372308339213"  "  0.0343381022633838"
##  [4] "  0.030865855042769"   "  0.022648764010159"   "  0.0297961082127872"
##  [7] "  0.0056040386677175"  "  0.0459296288858487"  "  0.0214440492959764"
## [10] "  0.00500296481991139" "  0.00397786927665191" "  0.0189498009758069"
## [13] "  0.0115560029829306"  "  0.00845017309868794" "  0.00791277202688269"
```

```
## [16] "  0.0121428897763471"  "  0.0219983071906645"  "  0.0256849046363538"
## [19] "  0.0496781240397854"  "  0.0445262640178593"  "  0.0417155245712763"
## [22] "  0.0474488743903027"  "  0.00109849576819036" "  0.00677690263528848"
## [25] "  0.0175778116391461"  "  0.0445363801322463"  "  0.0486316952459632"
## [28] "  0.0204020005380137"  "  0.0517757307567397"  "  0.0393179157886852"
## [31] "  0.0317882289696976"  "  0.0119184133584844"  "  0.0448758447503635"
## [34] "  0.00423524366167183" "  0.0172160650942885"  "  0.0334014486851988"
## [1] ""
## [1] "(4) LONG PORTFOLIO()"
## [1] "-------------------------------------------------------------------------------------------------------"
## [1] "################"
## [1] "### (tau=5) ###"
## [1] "################"
## [1] "CLOSE all positions"
## [1] "(1) COMPUTE_P/L(portfolio)"
## [1] "--> Capital:625347.972525997$"
## [1] ""
## [1] "(2) PORTFOLIO_LOOP:"
## [1] "    SECTOR_PROCEDURE(G=Industrials, tau=5)"
## [1] "    SECTOR_PROCEDURE(G=Health Care, tau=5)"
## [1] "    SECTOR_PROCEDURE(G=Information Technology, tau=5)"
## [1] "    SECTOR_PROCEDURE(G=Communication Services, tau=5)"
## [1] "    SECTOR_PROCEDURE(G=Financials, tau=5)"
## [1] "    SECTOR_PROCEDURE(G=Consumer Discretionary, tau=5)"
## [1] "Cur Portfolio:"
##  [1] "HON"   "PH"    "EMR"   "GD"    "MMM"   "UNP"   "ABBV"  "AMGN"  "ISRG"
## [10] "BMY"   "CVS"   "JNJ"   "ADI"   "INTU"  "TXN"   "IBM"   "ORCL"  "NOW"
## [19] "TMUS"  "NWSA"  "GOOGL" "WBD"   "FOXA"  "MTCH"  "MS"    "BAC"   "SCHW"
## [28] "WFC"   "MA"    "AON"   "YUM"   "AZO"   "MAR"   "GM"    "ROST"  "F"
## [1] ""
## [1] "(3) OPTIMIZE_PORTFOLIO(portfolio)"
## [1] "weights: "
##  [1] "  0.0620281732631475"  "  0.0512594739887319"  "  0.026433936547767"
##  [4] "  0.0564744987115449"  "  0.0582414941552104"  "  0.0261193403150482"
##  [7] "  0.0382944629278354"  "  0.00453587537156573" "  0.0560922072869385"
## [10] "  0.0442651429992457"  "  0.0367843822157951"  "  0.00536997522924203"
## [13] "  0.0396267562790427"  "  0.0245031759357546"  "  0.0416525015592389"
## [16] "  0.0190571343451119"  "  0.00744417970032052" "  0.0478635782656836"
## [19] "  0.0443068375483094"  "  0.00413396364565969" "  0.0255884967821204"
## [22] "  0.0375137749217412"  "  0.0224760969347357"  "  0.00752909921345859"
## [25] "  0.0299046394545892"  "  0.00829143297173139" "  0.00705773669652614"
## [28] "  0.0117736142153464"  "  0.0124381742658554"  "  0.0584629530218132"
## [31] "  0.0350346413919681"  "  0.0448226528769774"  "  0.0342468222013941"
## [34] "  0.0545078186227667"  "  0.0475829467351184"  "  0.00426342431027426"
## [1] ""
## [1] "(4) LONG PORTFOLIO()"
## [1] "-------------------------------------------------------------------------------------------------------"
```

SECTOR_PROCEDURE

τ and window logic

1. Sector G contains tickers $\{S_1, S_1, \ldots, S_{|G|}\}$, where $|G|$= number of stocks per sector (before selection).
2. For each ticker, want to calculate **current window:**

$$[t_1 = \text{week } W_{s\times\tau}\ ,\ t_{12} = \text{week } W_{s\times\tau+11}]$$

e.g. with $s = 1$ (slide one month at the time)

$$\begin{cases} \tau = 1 \implies [t_1 = W_1 \ , \ t_{12} = W_{12}] \\ \tau = 2 \implies [t_1 = W_2 \ , \ t_{12} = W_{13}] \\ \vdots \\ \tau = i \implies [t_1 = W_i \ , \ t_{12} = W_{i+11}] \\ \vdots \\ \tau = T \implies [t_1 = W_{T-12} \ , \ t_{12} = W_T] \end{cases}$$

EXTRACT_STATIC_FEATURES()

We had a set of features for some stock:

```
#get a sample stock xts data
sample_xts <- sp500_stocks$Industrials$ADP
tail(sample_xts, 5)
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2022-11-02       232.4444              1      0.009781441      0.012306040
## 2022-11-09       235.3226              1      0.012382071      0.053615960
## 2022-11-16       248.2840              1      0.055079334      0.034718650
## 2022-11-23       257.0555              1      0.035328373      0.005923636
## 2022-11-30       258.5827             NA      0.005941215               NA
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2022-11-02      0.009733913      0.008113141      0.039930970     -0.064535800
## 2022-11-09      0.012306040      0.009733913      0.008113141      0.039930970
## 2022-11-16      0.053615960      0.012306040      0.009733913      0.008113141
## 2022-11-23      0.034718650      0.053615960      0.012306040      0.009733913
## 2022-11-30      0.005923636      0.034718650      0.053615960      0.012306040
##                  atr      adx aaron        bb chaikin_vol        clv        emv
## 2022-11-02  9.885942 13.58997   100 0.6303335  2.90314600 -0.2863719 0.02711271
## 2022-11-09  9.762661 13.77107    50 0.6307783 -0.09676625 -0.3920529 0.04765004
## 2022-11-16 10.232471 14.68326   100 0.8325740 -0.38397100 -0.4461119 0.09074850
## 2022-11-23 10.243009 15.95273   100 0.9310325 -0.20180520 -0.3205142 0.11758529
## 2022-11-30 10.247795 16.53998   100 0.8907336  0.48394890 -0.1089895 0.12144667
##                macd      mfi      sar       smi  volume     volat month_index
## 2022-11-02 1.939312 49.23300 258.6055  5.546375 1592400 0.2606250          83
## 2022-11-09 1.866926 49.20839 257.2257  3.943960 1242900 0.2653165          83
## 2022-11-16 1.906715 48.83463 256.7200  6.291102 1430800 0.2641173          83
## 2022-11-23 2.068291 49.31528 224.1100 11.099826 1386300 0.2624611          83
## 2022-11-30 2.300754 42.97382 224.1100 16.713518 4155500 0.2759187          83
##            Excess_Retun_Mkt Small_minus_Big High_minus_Low Robus_minus_Weak
## 2022-11-02          -0.0267         -0.0087         0.0161           0.0021
## 2022-11-09          -0.0225         -0.0052         0.0055           0.0095
## 2022-11-16          -0.0103         -0.0107         0.0057           0.0119
## 2022-11-23           0.0063         -0.0024        -0.0094          -0.0075
## 2022-11-30           0.0312         -0.0015        -0.0207          -0.0077
##            Conservative_minus_Aggressive Risk_free_rate Momentum
## 2022-11-02                        0.0105        0.00014   0.0216
## 2022-11-09                        0.0106        0.00014   0.0164
## 2022-11-16                        0.0093        0.00014   0.0269
## 2022-11-23                       -0.0057        0.00014  -0.0184
## 2022-11-30                       -0.0141        0.00014  -0.0282
```

The follwoing function extracts the specific window

```r
# source the feature engineering file
library("here")
source(here("functions", "feature_engineering.R"))

# test out for a sample run
tau = 10 # suppose we're at run number 3
sample_xts_window <- f_extract_window(sample_xts, # stock xts
                                      tau=tau, # current run
                                      n_months = N_window # size of window
                                      )

# display some columns for the extracted data
tail(sample_xts_window[,c("direction_lead", "clv", "volat", "month_index")], 10)
```

```
##            direction_lead          clv     volat month_index
## 2018-01-24              1 -0.091779472 0.1559787          25
## 2018-01-31             -1 -0.011071824 0.1926336          25
## 2018-02-07             -1 -0.020451236 0.2037605          26
## 2018-02-14              1  0.145819436 0.2180265          26
## 2018-02-21             -1  0.024760832 0.2316219          26
## 2018-02-28             -1 -0.158012235 0.2332037          26
## 2018-03-07              1  0.005197972 0.2378317          27
## 2018-03-14              1 -0.087021937 0.2396534          27
## 2018-03-21             -1 -0.044196849 0.2438210          27
## 2018-03-28              1 -0.128497622 0.2438491          27
```

EXTRACT_DYNAMIC_FEATURES

Three functions: - `f_add_garch_forecast()`: Computes the GARCH - `f_add_arima_forecast()`: Computes additional ARIMA features - `f_extract_dynamic_features()`: Combines the previous two functions

```r
# add GARCH features only
sample_xts_with_garch <- f_add_garch_forecast(sample_xts, volat_col="volat")

# display
tail(sample_xts_with_garch, 3)
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2022-11-16       248.2840              1      0.055079334      0.034718650
## 2022-11-23       257.0555              1      0.035328373      0.005923636
## 2022-11-30       258.5827             NA      0.005941215               NA
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2022-11-16      0.053615960       0.01230604      0.009733913      0.008113141
## 2022-11-23      0.034718650       0.05361596      0.012306040      0.009733913
## 2022-11-30      0.005923636       0.03471865      0.053615960      0.012306040
##                  atr      adx aaron        bb chaikin_vol        clv       emv
## 2022-11-16 10.23247 14.68326   100 0.8325740  -0.3839710 -0.4461119 0.0907485
## 2022-11-23 10.24301 15.95273   100 0.9310325  -0.2018052 -0.3205142 0.1175853
## 2022-11-30 10.24779 16.53998   100 0.8907336   0.4839489 -0.1089895 0.1214467
##                macd      mfi    sar       smi  volume     volat month_index
## 2022-11-16 1.906715 48.83463 256.72  6.291102 1430800 0.2641173          83
## 2022-11-23 2.068291 49.31528 224.11 11.099826 1386300 0.2624611          83
## 2022-11-30 2.300754 42.97382 224.11 16.713518 4155500 0.2759187          83
##            Excess_Retun_Mkt Small_minus_Big High_minus_Low Robus_minus_Weak
## 2022-11-16          -0.0103         -0.0107         0.0057           0.0119
## 2022-11-23           0.0063         -0.0024        -0.0094          -0.0075
## 2022-11-30           0.0312         -0.0015        -0.0207          -0.0077
##            Conservative_minus_Aggressive Risk_free_rate Momentum vol_forecast
```

```
## 2022-11-16                        0.0093        0.00014   0.0269    0.2782794
## 2022-11-23                       -0.0057        0.00014  -0.0184    0.2794421
## 2022-11-30                       -0.0141        0.00014  -0.0282    0.2805933
```

```
# Example usage
sample_xts_with_arima <- f_add_arima_forecast(sample_xts_with_garch,
                                              arima_col="realized_returns")
tail(sample_xts_with_arima)
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2022-10-26       230.1928              1      0.008146142      0.009733913
## 2022-11-02       232.4444              1      0.009781441      0.012306040
## 2022-11-09       235.3226              1      0.012382071      0.053615960
## 2022-11-16       248.2840              1      0.055079334      0.034718650
## 2022-11-23       257.0555              1      0.035328373      0.005923636
## 2022-11-30       258.5827             NA      0.005941215               NA
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2022-10-26      0.008113141      0.039930970     -0.064535800      0.030150980
## 2022-11-02      0.009733913      0.008113141      0.039930970     -0.064535800
## 2022-11-09      0.012306040      0.009733913      0.008113141      0.039930970
## 2022-11-16      0.053615960      0.012306040      0.009733913      0.008113141
## 2022-11-23      0.034718650      0.053615960      0.012306040      0.009733913
## 2022-11-30      0.005923636      0.034718650      0.053615960      0.012306040
##                  atr      adx aaron        bb chaikin_vol        clv
## 2022-10-26  9.676399 13.39493   100 0.6110784 -1.49750300 -0.1320576
## 2022-11-02  9.885942 13.58997   100 0.6303335  2.90314600 -0.2863719
## 2022-11-09  9.762661 13.77107    50 0.6307783 -0.09676625 -0.3920529
## 2022-11-16 10.232471 14.68326   100 0.8325740 -0.38397100 -0.4461119
## 2022-11-23 10.243009 15.95273   100 0.9310325 -0.20180520 -0.3205142
## 2022-11-30 10.247795 16.53998   100 0.8907336  0.48394890 -0.1089895
##                     emv     macd      mfi      sar       smi  volume     volat
## 2022-10-26 -0.01707202 2.049576 51.52422 260.0428  8.131402 2942400 0.2269538
## 2022-11-02  0.02711271 1.939312 49.23300 258.6055  5.546375 1592400 0.2606250
## 2022-11-09  0.04765004 1.866926 49.20839 257.2257  3.943960 1242900 0.2653165
## 2022-11-16  0.09074850 1.906715 48.83463 256.7200  6.291102 1430800 0.2641173
## 2022-11-23  0.11758529 2.068291 49.31528 224.1100 11.099826 1386300 0.2624611
## 2022-11-30  0.12144667 2.300754 42.97382 224.1100 16.713518 4155500 0.2759187
##            month_index Excess_Retun_Mkt Small_minus_Big High_minus_Low
## 2022-10-26          82          -0.0066          0.0070         0.0089
## 2022-11-02          83          -0.0267         -0.0087         0.0161
## 2022-11-09          83          -0.0225         -0.0052         0.0055
## 2022-11-16          83          -0.0103         -0.0107         0.0057
## 2022-11-23          83           0.0063         -0.0024        -0.0094
## 2022-11-30          83           0.0312         -0.0015        -0.0207
##            Robus_minus_Weak Conservative_minus_Aggressive Risk_free_rate
## 2022-10-26          -0.0080                        0.0067        0.00011
## 2022-11-02           0.0021                        0.0105        0.00014
## 2022-11-09           0.0095                        0.0106        0.00014
## 2022-11-16           0.0119                        0.0093        0.00014
## 2022-11-23          -0.0075                       -0.0057        0.00014
## 2022-11-30          -0.0077                       -0.0141        0.00014
##            Momentum vol_forecast sarima_100_001 sarima_010_001 sarima_110_001
## 2022-10-26   0.0049    0.2624611    0.005473012    0.034718650     0.04342600
## 2022-11-02   0.0216    0.2759187    0.003833988    0.005923636     0.01919158
## 2022-11-09   0.0164    0.2771050    0.003715045    0.005923636     0.01307808
## 2022-11-16   0.0269    0.2782794    0.003708274    0.005923636     0.01589501
## 2022-11-23  -0.0184    0.2794421    0.003707889    0.005923636     0.01459705
## 2022-11-30  -0.0282    0.2805933    0.003707867    0.005923636     0.01519511
##            sarima_020_001 sarima_120_001 sarima_100_011 sarima_010_011
```

```
## 2022-10-26     0.01582134     0.05513160    0.005473012    0.034718650
## 2022-11-02    -0.02287138    -0.01640901    0.003833988    0.005923636
## 2022-11-09    -0.05166639    -0.04296104    0.003715045    0.005923636
## 2022-11-16    -0.08046141    -0.06675816    0.003708274    0.005923636
## 2022-11-23    -0.10925642    -0.09235401    0.003707889    0.005923636
## 2022-11-30    -0.13805143    -0.11677544    0.003707867    0.005923636
##            sarima_110_011 sarima_020_011 sarima_120_011 best_shifted_arima
## 2022-10-26     0.04342600     0.01582134     0.05513160         0.05513160
## 2022-11-02     0.01919158    -0.02287138    -0.01640901        -0.01640901
## 2022-11-09     0.01307808    -0.05166639    -0.04296104        -0.04296104
## 2022-11-16     0.01589501    -0.08046141    -0.06675816        -0.06675816
## 2022-11-23     0.01459705    -0.10925642    -0.09235401        -0.09235401
## 2022-11-30     0.01519511    -0.13805143    -0.11677544        -0.11677544
```

```
sample_xts_with_arima[, c("discrete_returns", "volat",  "vol_forecast")]
```

```
##            discrete_returns     volat vol_forecast
## 2016-01-06               NA        NA           NA
## 2016-01-13     -0.048239939        NA           NA
## 2016-01-20      0.011378375        NA           NA
## 2016-01-27      0.028892496        NA           NA
## 2016-02-03      0.020750704        NA           NA
## 2016-02-10     -0.016067960        NA    0.2380100
## 2016-02-17      0.055672632        NA    0.2389290
## 2016-02-24     -0.000820507        NA    0.2214060
## 2016-03-02      0.004574200        NA    0.1992566
## 2016-03-09      0.007060459 0.2380100    0.1872713
##         ...
## 2022-09-28      0.006640017 0.2449987    0.2269538
## 2022-10-05      0.030610122 0.2057967    0.2606250
## 2022-10-12     -0.062497445 0.1956467    0.2653165
## 2022-10-19      0.040738929 0.1976342    0.2641173
## 2022-10-26      0.008146142 0.2269538    0.2624611
## 2022-11-02      0.009781441 0.2606250    0.2759187
## 2022-11-09      0.012382071 0.2653165    0.2771050
## 2022-11-16      0.055079334 0.2641173    0.2782794
## 2022-11-23      0.035328373 0.2624611    0.2794421
## 2022-11-30      0.005941215 0.2759187    0.2805933
```

```
# Example usage
sample_xts_full <- f_extract_dynamic_features(sample_xts_with_garch,
                                              arima_col = "realized_returns", # used as data for the ARIMA
                                              volat_col = "volat") # historical volat, used by GARCH
tail(sample_xts_full)
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2022-10-26       230.1928              1      0.008146142      0.009733913
## 2022-11-02       232.4444              1      0.009781441      0.012306040
## 2022-11-09       235.3226              1      0.012382071      0.053615960
## 2022-11-16       248.2840              1      0.055079334      0.034718650
## 2022-11-23       257.0555              1      0.035328373      0.005923636
## 2022-11-30       258.5827             NA      0.005941215               NA
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2022-10-26      0.008113141      0.039930970     -0.064535800      0.030150980
## 2022-11-02      0.009733913      0.008113141      0.039930970     -0.064535800
## 2022-11-09      0.012306040      0.009733913      0.008113141      0.039930970
## 2022-11-16      0.053615960      0.012306040      0.009733913      0.008113141
## 2022-11-23      0.034718650      0.053615960      0.012306040      0.009733913
## 2022-11-30      0.005923636      0.034718650      0.053615960      0.012306040
```

```
##                  atr      adx aaron        bb chaikin_vol        clv
## 2022-10-26  9.676399 13.39493   100 0.6110784 -1.49750300 -0.1320576
## 2022-11-02  9.885942 13.58997   100 0.6303335  2.90314600 -0.2863719
## 2022-11-09  9.762661 13.77107    50 0.6307783 -0.09676625 -0.3920529
## 2022-11-16 10.232471 14.68326   100 0.8325740 -0.38397100 -0.4461119
## 2022-11-23 10.243009 15.95273   100 0.9310325 -0.20180520 -0.3205142
## 2022-11-30 10.247795 16.53998   100 0.8907336  0.48394890 -0.1089895
##                   emv     macd      mfi      sar       smi  volume     volat
## 2022-10-26 -0.01707202 2.049576 51.52422 260.0428  8.131402 2942400 0.2269538
## 2022-11-02  0.02711271 1.939312 49.23300 258.6055  5.546375 1592400 0.2606250
## 2022-11-09  0.04765004 1.866926 49.20839 257.2257  3.943960 1242900 0.2653165
## 2022-11-16  0.09074850 1.906715 48.83463 256.7200  6.291102 1430800 0.2641173
## 2022-11-23  0.11758529 2.068291 49.31528 224.1100 11.099826 1386300 0.2624611
## 2022-11-30  0.12144667 2.300754 42.97382 224.1100 16.713518 4155500 0.2759187
##            month_index Excess_Retun_Mkt Small_minus_Big High_minus_Low
## 2022-10-26          82          -0.0066          0.0070         0.0089
## 2022-11-02          83          -0.0267         -0.0087         0.0161
## 2022-11-09          83          -0.0225         -0.0052         0.0055
## 2022-11-16          83          -0.0103         -0.0107         0.0057
## 2022-11-23          83           0.0063         -0.0024        -0.0094
## 2022-11-30          83           0.0312         -0.0015        -0.0207
##            Robus_minus_Weak Conservative_minus_Aggressive Risk_free_rate
## 2022-10-26          -0.0080                        0.0067        0.00011
## 2022-11-02           0.0021                        0.0105        0.00014
## 2022-11-09           0.0095                        0.0106        0.00014
## 2022-11-16           0.0119                        0.0093        0.00014
## 2022-11-23          -0.0075                       -0.0057        0.00014
## 2022-11-30          -0.0077                       -0.0141        0.00014
##            Momentum vol_forecast sarima_100_001 sarima_010_001 sarima_110_001
## 2022-10-26   0.0049    0.2624611    0.005473012    0.034718650     0.04342600
## 2022-11-02   0.0216    0.2759187    0.003833988    0.005923636     0.01919158
## 2022-11-09   0.0164    0.2771050    0.003715045    0.005923636     0.01307808
## 2022-11-16   0.0269    0.2782794    0.003708274    0.005923636     0.01589501
## 2022-11-23  -0.0184    0.2794421    0.003707889    0.005923636     0.01459705
## 2022-11-30  -0.0282    0.2805933    0.003707867    0.005923636     0.01519511
##            sarima_020_001 sarima_120_001 sarima_100_011 sarima_010_011
## 2022-10-26     0.01582134     0.05513160    0.005473012    0.034718650
## 2022-11-02    -0.02287138    -0.01640901    0.003833988    0.005923636
## 2022-11-09    -0.05166639    -0.04296104    0.003715045    0.005923636
## 2022-11-16    -0.08046141    -0.06675816    0.003708274    0.005923636
## 2022-11-23    -0.10925642    -0.09235401    0.003707889    0.005923636
## 2022-11-30    -0.13805143    -0.11677544    0.003707867    0.005923636
##            sarima_110_011 sarima_020_011 sarima_120_011 best_shifted_arima
## 2022-10-26     0.04342600     0.01582134     0.05513160         0.05513160
## 2022-11-02     0.01919158    -0.02287138    -0.01640901        -0.01640901
## 2022-11-09     0.01307808    -0.05166639    -0.04296104        -0.04296104
## 2022-11-16     0.01589501    -0.08046141    -0.06675816        -0.06675816
## 2022-11-23     0.01459705    -0.10925642    -0.09235401        -0.09235401
## 2022-11-30     0.01519511    -0.13805143    -0.11677544        -0.11677544
```

SECTOR PROCEDURE

```
SECTOR_PROCEDURE <- function(G, tau){
  ##
  ## Params:
  ##  - G (str): Economic sector name; will be used to fetch the  List of lists
  ## which are the pre-selected stocks for that sector.
  ##  - tau (numeric): Integer that corresponds to the actual run of the backtest.
```

```
##


### TEST ###
# NOTE: For testing only, will be removed later!
num_top_pick <- N_sector_best_stocks*2 # number of stocks picked per sector
### TEST ###

print(paste0("SECTOR_PROCEDURE(G=", G, ", tau=",tau, ")"))

# retrieve sector data
sector_data <- sp500_stocks[[G]]

# stocks for sector provided
sector_stocks <- names(sector_data)

# to store subset features for window
sector_stocks_window <- rep(NA, length(sector_stocks))
names(sector_stocks_window) <- sector_stocks

# extract static list for all stocks
list_xts_sector <- lapply(sector_data,
                          f_extract_window,
                          tau=tau, # current run
                          n_months = N_window# size of window
                          )

# compute dynamic features for all stocks
list_xts_sector <- lapply(list_xts_sector,
                          function(x, arima_col, volat_col) {
                            tryCatch({
                              f_extract_dynamic_features(x, arima_col, volat_col)
                            },
                            error = function(e){
                              warning("error with this dataframe:")
                              print(head(x))
                              print(tail(x))
                              print(colnames(x))
                              stop(e)
                            }
                            )
                          },
                          arima_col = "realized_returns",
                          volat_col = "volat"
                          )

# return top 3 best stocks according to modelling procedure
print("  MODELLING_PROCEDURE(list_train_val_sector)")
top_sector_stocks <- sample(names(sp500_stocks[[G]]), num_top_pick)

########## Inside MODELLING_PROCEDURE ##########################
### NOTE: The MODELLING_PROCEDURE internally will use the train and

# should return the list for the chosen stocks
chosen_stocks <- sector_data[top_sector_stocks]

########## Inside MODELLING_PROCEDURE ##########################

return(chosen_stocks) # not actual return value!
```

```
}

# peform the sector procedure
G = names(sp500_stocks)[[1]]
tau = 10
sector_stocks_window <- SECTOR_PROCEDURE(G, tau)
```

```
## [1] "SECTOR_PROCEDURE(G=Industrials, tau=10)"
## [1] "  MODELLING_PROCEDURE(list_train_val_sector)"
```

```
names(sector_stocks_window) # names are tickers, values are list of xts
```

```
## [1] "NOC" "WM"  "EMR" "ADP" "PH"  "BA"
```

```
head(sector_stocks_window[[2]]) # show ticker xts
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2016-01-06       45.75028             -1               NA    -0.0197050800
## 2016-01-13       44.85759             -1    -0.0195122000    -0.0065270830
## 2016-01-20       44.56576             -1    -0.0065058280    -0.0005780327
## 2016-01-27       44.54000              1    -0.0005778657     0.0226740500
## 2016-02-03       45.56144              1     0.0229330600     0.0082554170
## 2016-02-10       45.93913              1     0.0082895870     0.0254577500
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2016-01-06               NA               NA               NA               NA
## 2016-01-13    -0.0197050800               NA               NA               NA
## 2016-01-20    -0.0065270830    -0.0197050800               NA               NA
## 2016-01-27    -0.0005780327    -0.0065270830    -0.0197050800               NA
## 2016-02-03     0.0226740500    -0.0005780327    -0.0065270830     -0.019705080
## 2016-02-10     0.0082554170     0.0226740500    -0.0005780327     -0.006527083
##            atr adx aaron bb chaikin_vol clv emv macd mfi      sar smi  volume
## 2016-01-06  NA  NA    NA NA          NA  NA  NA   NA  NA 52.33923  NA 3277400
## 2016-01-13  NA  NA     0 NA          NA  NA  NA   NA  NA 53.35000  NA 2445100
## 2016-01-20  NA  NA   -50 NA          NA  NA  NA   NA  NA 53.35000  NA 4118900
## 2016-01-27  NA  NA   -50 NA          NA  NA  NA   NA  NA 53.29280  NA 2885400
## 2016-02-03  NA  NA   100 NA          NA  NA  NA   NA  NA 53.27000  NA 2225200
## 2016-02-10  NA  NA   100 NA          NA  NA  NA   NA  NA 51.89000  NA 1686300
##            volat month_index Excess_Retun_Mkt Small_minus_Big High_minus_Low
## 2016-01-06    NA           1          -0.0135         -0.0023         0.0000
## 2016-01-13    NA           1          -0.0267         -0.0062         0.0081
## 2016-01-20    NA           1          -0.0094          0.0173        -0.0127
## 2016-01-27    NA           1          -0.0111         -0.0042         0.0171
## 2016-02-03    NA           2           0.0046         -0.0025         0.0047
## 2016-02-10    NA           2           0.0001         -0.0021        -0.0055
##            Robus_minus_Weak Conservative_minus_Aggressive Risk_free_rate
## 2016-01-06           0.0015                        0.0004          0e+00
## 2016-01-13           0.0040                        0.0063          0e+00
## 2016-01-20           0.0008                       -0.0052          0e+00
## 2016-01-27          -0.0013                        0.0092          0e+00
## 2016-02-03           0.0041                        0.0032          1e-05
## 2016-02-10          -0.0030                       -0.0069          1e-05
##            Momentum
## 2016-01-06   0.0192
## 2016-01-13   0.0016
## 2016-01-20  -0.0011
## 2016-01-27  -0.0048
## 2016-02-03  -0.0241
## 2016-02-10   0.0065
```

MODELLING_PROCEDURE

Recall that the **SECTOR_PROCEDURE**(G, τ) function takes the argument G, which is the **sector name**, and **tau**, which is the current run in the backtesting.

This procedure happens in a loop, for every sector G. Here, we fix one sector only, and a specific τ. The code does the following:

1. Retrieves the actual sector stock data (list of key-value pairs, keys are stock tickers, values are xts full data for that stock.)
2. Creates a variable to store the subset of data that goes into the current window.
3. The `f_extract_window()` function extracts the appropriate window of data corresponding to the τ, with the appropriate window size, for all sectors.
4. Extracts the dynamic features (ARIMA and GARCH) for that each stock in the sector.

```
# parameters
G <- names(sp500_stocks)[1] # sample sector
tau <- 10 # suppose we are in run 5 of the backtest

####### Inside SECTOR_PROCEDURE ########

# retrieve sector data
sector_data <- sp500_stocks[[G]]

# stocks for sector provided
sector_tickers <- names(sector_data)

# to store subset features for window
sector_stocks_window <- rep(NA, length(sector_tickers))
names(sector_stocks_window) <- sector_tickers

# extract static train-val for all stocks
list_xts_sector <- lapply(sector_data,
                          f_extract_window,
                          tau=tau, # current run
                          n_months = N_window# size of window
                          )

# compute dynamic features for all stocks
list_xts_sector <- lapply(list_xts_sector,
                          f_extract_dynamic_features,
                          arima_col = "realized_returns",
                          volat_col = "volat"
                          )

####### Inside SECTOR_PROCEDURE ########

# keys are stock tickers for that sector
names(list_xts_sector)
```

```
##  [1] "ADP" "BA"  "CAT" "CSX" "DE"  "EMR" "ETN" "FDX" "GD"  "GE"  "HON" "ITW"
## [13] "LMT" "MMM" "NOC" "PH"  "RTX" "UNP" "UPS" "WM"
```

```
# each stock has the xts subset (for window)
tail(list_xts_sector[[1]])
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2018-02-21      103.02951             -1      0.030077100     -0.006741057
## 2018-02-28      102.33732             -1     -0.006718387     -0.001214733
```

```
## 2018-03-07      102.21308             1     -0.001213996      0.013440750
## 2018-03-14      103.59618             1      0.013531485      0.002236909
## 2018-03-21      103.82817            -1      0.002239413     -0.040335250
## 2018-03-28       99.72357             1     -0.039532614      0.043424250
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2018-02-21      0.029633650     -0.007513064     -0.084962130      0.022660500
## 2018-02-28     -0.006741057      0.029633650     -0.007513064     -0.084962130
## 2018-03-07     -0.001214733     -0.006741057      0.029633650     -0.007513064
## 2018-03-14      0.013440750     -0.001214733     -0.006741057      0.029633650
## 2018-03-21      0.002236909      0.013440750     -0.001214733     -0.006741057
## 2018-03-28     -0.040335250      0.002236909      0.013440750     -0.001214733
##                 atr      adx aaron        bb  chaikin_vol          clv
## 2018-02-21 3.566973 20.54379    50 0.5534310  0.286293200  0.024760832
## 2018-02-28 3.458618 20.05893    50 0.5001998  0.005312863 -0.158012235
## 2018-03-07 3.332288 19.20823  -100 0.4048161  0.037957300  0.005197972
## 2018-03-14 3.293553 18.93669    50 0.5352393 -0.180221000 -0.087021937
## 2018-03-21 3.202586 18.60882    50 0.5135005 -0.021485810 -0.044196849
## 2018-03-28 3.335972 17.31306  -100 0.2052840 -0.406369400 -0.128497622
##                      emv     macd      mfi      sar       smi  volume     volat
## 2018-02-21 -0.005023182 2.714423 69.33525 124.7388 31.707790 1817200 0.2316219
## 2018-02-28 -0.004592290 2.549235 70.40971 124.2576 26.296191 1579900 0.2332037
## 2018-03-07 -0.005879919 2.373012 61.94528 123.7957 21.298126 1403100 0.2378317
## 2018-03-14 -0.003486557 2.206376 61.63417 123.3523 17.452348 1885000 0.2396534
## 2018-03-21 -0.008126241 2.053026 50.88007 122.9266 14.393438 4032800 0.2438210
## 2018-03-28 -0.012681549 1.847957 44.97873 122.5179  8.336334 2573800 0.2438491
##            month_index Excess_Retun_Mkt Small_minus_Big High_minus_Low
## 2018-02-21          26          -0.0043          0.0080         0.0023
## 2018-02-28          26          -0.0111         -0.0043        -0.0031
## 2018-03-07          27           0.0005          0.0076        -0.0044
## 2018-03-14          27          -0.0053          0.0015        -0.0057
## 2018-03-21          27          -0.0006          0.0069         0.0045
## 2018-03-28          27          -0.0035          0.0021         0.0065
##            Robus_minus_Weak Conservative_minus_Aggressive Risk_free_rate
## 2018-02-21           0.0007                       -0.0019          6e-05
## 2018-02-28           0.0021                       -0.0026          6e-05
## 2018-03-07          -0.0004                       -0.0031          5e-05
## 2018-03-14          -0.0011                       -0.0039          5e-05
## 2018-03-21          -0.0108                        0.0087          5e-05
## 2018-03-28           0.0075                       -0.0008          5e-05
##            Momentum sarima_100_001 sarima_010_001 sarima_110_001 sarima_020_001
## 2018-02-21   0.0079   0.0042935397    0.002236909    0.008548106   -0.008966932
## 2018-02-28   0.0052   0.0094352534   -0.040335250   -0.016354075   -0.082907409
## 2018-03-07   0.0079  -0.0006809206    0.043424250   -0.003758023    0.127183750
## 2018-03-14   0.0033   0.0046459448    0.043424250    0.022820059    0.210943250
## 2018-03-21  -0.0030   0.0040025849    0.043424250    0.007848452    0.294702750
## 2018-03-28  -0.0119   0.0040802876    0.043424250    0.016282056    0.378462250
##            sarima_120_001 sarima_100_011 sarima_010_011 sarima_110_011
## 2018-02-21    0.009860553   0.0042935397    0.002236909    0.008548106
## 2018-02-28   -0.060068972   0.0094352534   -0.040335250   -0.016354075
## 2018-03-07    0.035205033  -0.0006809206    0.043424250   -0.003758023
## 2018-03-14    0.093953071   0.0046459448    0.043424250    0.022820059
## 2018-03-21    0.103944034   0.0040025849    0.043424250    0.007848452
## 2018-03-28    0.149433725   0.0040802876    0.043424250    0.016282056
##            sarima_020_011 sarima_120_011 best_shifted_arima vol_forecast
## 2018-02-21   -0.008966932    0.009860553        0.009860553    0.2438210
## 2018-02-28   -0.082907409   -0.060068972       -0.060068972    0.2438491
## 2018-03-07    0.127183750    0.035205033        0.035205033    0.2335522
## 2018-03-14    0.210943250    0.093953071        0.093953071    0.2247342
## 2018-03-21    0.294702750    0.103944034        0.103944034    0.2172172
## 2018-03-28    0.378462250    0.149433725        0.149433725    0.2108369
```

```
# save data in tests
save(list_xts_sector, file = here("tests","jair", "sample_data.rda"))
```

The result is the `list_train_val_sector` oject, which is a list of lists. - The first level are the stock tickers - The second level are train and val xts for each stock.

Feature Selection

Notes: - This will use **forward selection** to extract the features from a sample stock for the current sector. - The `target_var` argument specifies the target variable, in this case is called "realized_returns". - `f_select_features()` is found under `functions/feature_engineering.R`

```
# Extract a sample stock in the list_xts_sector
sample_sector_stock <- list_xts_sector[[1]]

# Define the formula for regression
fmla <- realized_returns ~ . -realized_returns -month_index

# try obtaining best features for a sample train set for a stock in the sample sector
best_feat_list <- f_select_features(
                    fmla = fmla, # formula for regression
                    data = sample_sector_stock, # for one stock of current sector
                    target_var = "realized_returns", # future-lagged log-returns
                    volat_col = "volat", # we always want to keep the volatility col
                    garch_col = "vol_forecast", # GARCH column
                    nvmax = 25, # maximum number of subsets to examine
                    method="backward") #  we always want to use forward selection
```

```
## Loading required package: leaps
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 6 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
print("")
```

```
## [1] ""
```

```
best_feat_list
```

```
## $featnames
##  [1] "adjusted_close"                "direction_lead"
##  [3] "discrete_returns"              "log_returns_lag0"
##  [5] "log_returns_lag1"              "log_returns_lag2"
##  [7] "atr"                           "adx"
##  [9] "clv"                           "macd"
## [11] "mfi"                           "sar"
## [13] "smi"                           "volume"
## [15] "volat"                         "Excess_Retun_Mkt"
## [17] "Small_minus_Big"               "High_minus_Low"
## [19] "Conservative_minus_Aggressive" "sarima_100_001"
## [21] "sarima_010_001"                "sarima_110_001"
## [23] "sarima_120_001"                "vol_forecast"
##
## $fmla
```

```
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     log_returns_lag0 + log_returns_lag1 + log_returns_lag2 +
##     atr + adx + clv + macd + mfi + sar + smi + volume + volat +
##     Excess_Retun_Mkt + Small_minus_Big + High_minus_Low + Conservative_minus_Aggressive +
##     sarima_100_001 + sarima_010_001 + sarima_110_001 + sarima_120_001 +
##     vol_forecast
## <environment: 0x0000021590c285c0>
```

Regularized MLR (Elasticnet)

$$\mathcal{L}(\beta) = \frac{1}{2}\sum_{i=1}^{n}(y_i - x_i^T\beta)^2 + \lambda\left[\alpha||\beta||_1 + (1-\alpha)||\beta||_2^2\right]$$

```
# load required libraries
library("caret")
library("Metrics")

# Define the formula for regression
fmla <- realized_returns ~ . -realized_returns -month_index

# Create a grid for elastic net regression hyperparameters
grid_enet <- expand.grid(alpha = seq(from = 0, to = 1, by = 0.1),  # Elastic net mixing parameter
                         lambda = seq(from = 0, to = 0.05, by = 0.01))  # Regularization strength

# Initialize variable to save forecasted returns, MSEs and Sharpe Ratios
sector_tracker <- as.list(rep(NA, length(sector_tickers)))
names(sector_tracker) <- sector_tickers

# transform into a list of lists
sector_tracker <- lapply(sector_tracker, function(x) list(
  forecasted_ret = NA,
  sharpe = NA,
  msr = NA, # modified sharpe ratio
  rmse = NA,
  data = NA
))

# display values
fmla # all initial variables
```

```
## realized_returns ~ . - realized_returns - month_index
```

```
names(sector_tracker) # list of lists
```

```
##  [1] "ADP" "BA"  "CAT" "CSX" "DE"  "EMR" "ETN" "FDX" "GD"  "GE"  "HON" "ITW"
## [13] "LMT" "MMM" "NOC" "PH"  "RTX" "UNP" "UPS" "WM"
```

```
names(sector_tracker[[1]]) # to store the values as the loop happens
```

```
## [1] "forecasted_ret" "sharpe"         "msr"            "rmse"
## [5] "data"
```

Fitting all the models

Next, we loop through every stock doing the following: 1. Extracting the train and validation sets, and filter NAs 2. Perform feature selection for every stock 3. Fit an Elasticnet model for that stock, and obtain predictions for the returns 4. Compute the RMSE 5. Compute the Sharpe Ratio and Modified Sharpe 6. Save everything

```r
library("glmnet")

system.time(
  # Loop for every stock ticker in sector G
  for(ticker in sector_tickers){
    print(paste0("ticker: ", ticker))

    ### Step 0: Data Preparation

    ##############################################################################
    ### NOTE: Need to refactor

    # fetch data for that ticker
    full_train <- list_xts_sector[[ticker]]

    # Re-extract train and val with full features
    full_train <- f_extract_train_val_no_window(full_train,
                                                val_lag = 1) # number of months in val

    # Reassign to train and val
    ticker_data_train <- full_train$train
    ticker_data_val <- full_train$val

    # remove nas
    ticker_data_train <- na.omit(ticker_data_train) # data cannot contain nas
    ticker_data_val <- na.omit(ticker_data_val) # data cannot contain nas

    # re-stack train and val for later
    full_train <- rbind.xts(ticker_data_train, ticker_data_val)

    ##############################################################################

    ### Step 1: Feature Selection

    # Perform feature selection for that stock
    best_feat_list <- f_select_features(
                        fmla = fmla, # formula for regression
                        data = ticker_data_train, # train data for one stock of current sector
                        target_var = "realized_returns", # forecast future log returns
                        volat_col = "volat", # always keep the actual volatility
                        garch_col = "vol_forecast",
                        nvmax = 20, # total number of max subsets
                        method="backward")

    print(best_feat_list$fmla)

    ### Step 2: Elasticnet

    # # Set up time-slice cross-validation parameters
    # ctr_train <- trainControl(method = "timeslice", # cross validation
    #                           initialWindow = 52,  # Consecutive number of weeks
    #                           horizon = 4,         # Horizon is one month prediction (4 weeks)
    #                           skip = 1,            # No skip, our data will overlap in practice
    #                           fixedWindow = TRUE,   # Use a fixed window
    #                           allowParallel = TRUE) # Enable parallel processing

    # Set up K-fold CV parameters
    ctr_train <- trainControl(method = "cv", # cross validation \
                              number = 10, # number of folds
```

```
                          allowParallel = TRUE) # Enable parallel processing


  # Train the elastic net regression model using time-slice cross-validation
  model_enet_best <- train(form = best_feat_list$fmla,            # Formula from feature selection
                           data = ticker_data_train,              # Training data
                           method = "glmnet",                     # Model method = Elasticnet
                           tuneGrid = grid_enet,                  # Hyperparameter grid
                           trControl = ctr_train,                 # Cross-validation control
                           preProc = c("center", "scale"),        # Preprocessing steps
                           metric = "Rsquared",                   # Metric for selecting the best model
                           threshold = 0.2)

  # Extract the best alpha and beta fitted
  best_alpha <- model_enet_best$bestTune$alpha
  best_lambda <- model_enet_best$bestTune$lambda

  # Subset features and targets for retraining
  X_train <- model.matrix(best_feat_list$fmla, data = ticker_data_train)
  X_test <- model.matrix(best_feat_list$fmla, data = ticker_data_val)
  y_train <- ticker_data_train[, "realized_returns"]

  # refit the model and assign test
  refitted_model <- glmnet(X_train, y_train, alpha = best_alpha, lambda = best_lambda, standardize = TRUE)

  # Use the best-fitted elastic net regression model to make predictions on the val_data
  pred_enet_best <- predict(refitted_model, newx = X_test, s = refitted_model$lambda, type = "response")
  pred_enet_best <- mean(pred_enet_best) # take the average

  # Compute the RMSE on the validation set
  enet_rmse <- sqrt(mse(actual = ticker_data_val[, "realized_returns"], predicted = pred_enet_best))


  ### Step 3: Sharpe Ratio

  # Calculate the Sharpe Ratio and MSR (on historical discrete returns)
  scaling_factor <- as.vector(ticker_data_val$month_index)[1] - as.vector(ticker_data_train$month_index)[1]

  # Pack returns and compute mean and std
  hist_returns <- na.trim(as.vector(full_train[, "discrete_returns"]))
  mean_rets <- mean(hist_returns)
  std_rets <- sd(hist_returns)

  # Calculate the ES and set risk-free
  VaR <- quantile(hist_returns, 0.05)
  ES <- mean(hist_returns[hist_returns < VaR])
  Rf <- 0.0002 # 0

  # Calculate the Sharpe and MSR
  stock_sharpe <- ((mean_rets- Rf)/ std_rets ) * sqrt(scaling_factor) # annualized
  stock_msr <- ((mean_rets- Rf)/ ES ) * sqrt(scaling_factor) # annualized

  ### Step 4: Track the measures

  sector_tracker[[ticker]]$forecasted_ret = pred_enet_best
  sector_tracker[[ticker]]$rmse = enet_rmse
  sector_tracker[[ticker]]$sharpe = stock_sharpe
  sector_tracker[[ticker]]$msr = stock_msr
  sector_tracker[[ticker]]$data = full_train[, c("realized_returns",
```

```
                                                "best_shifted_arima",
                                                "volat",
                                                "vol_forecast",
                                                "month_index"
                                                )] # features to be kept

    # show values
    print("****************************************")
    print(paste("forecasted_ret: ", pred_enet_best))
    print(paste("rmse: ", enet_rmse))
    print(paste("sharpe: ", stock_sharpe))
    print(paste("msr: ", stock_msr))
    print("****************************************")

    print("############################################")
  }
)
```

```
## [1] "ticker: ADP"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + log_returns_lag0 +
##     log_returns_lag1 + log_returns_lag2 + log_returns_lag3 +
##     atr + adx + clv + emv + macd + mfi + smi + volat + High_minus_Low +
##     Conservative_minus_Aggressive + Momentum + sarima_100_001 +
##     sarima_110_001 + sarima_120_001 + vol_forecast
## <environment: 0x0000021584e63038>
## [1] "****************************************"
## [1] "forecasted_ret:  0.00408551914266138"
## [1] "rmse:  0.0300486196877053"
## [1] "sharpe:  0.580363405258177"
## [1] "msr:  -0.209984455037706"
## [1] "****************************************"
## [1] "############################################"
## [1] "ticker: BA"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + log_returns_lag1 +
##     atr + adx + aaron + bb + clv + macd + sar + smi + volume +
##     volat + Excess_Retun_Mkt + Small_minus_Big + Robus_minus_Weak +
##     Conservative_minus_Aggressive + Risk_free_rate + sarima_100_001 +
##     sarima_110_001 + sarima_120_001 + vol_forecast
## <environment: 0x00000215904b6168>
## [1] "****************************************"
## [1] "forecasted_ret:  -0.0390244833918096"
## [1] "rmse:  0.0437596842157666"
## [1] "sharpe:  1.72185688827279"
## [1] "msr:  -1.06686915166474"
## [1] "****************************************"
## [1] "############################################"
## [1] "ticker: CAT"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + atr + aaron +
##     clv + mfi + sar + smi + volat + Excess_Retun_Mkt + Small_minus_Big +
##     Robus_minus_Weak + Risk_free_rate + sarima_100_001 + sarima_110_001 +
##     sarima_120_001 + vol_forecast
## <environment: 0x0000021585442ae0>
## [1] "****************************************"
## [1] "forecasted_ret:  0.00782035247426104"
## [1] "rmse:  0.0400277008212605"
## [1] "sharpe:  0.952813134078394"
```

```
## [1] "msr:  -0.586123056703663"
## [1] "****************************************"
## [1] "########################################"
## [1] "ticker: CSX"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     log_returns_lag0 + log_returns_lag1 + log_returns_lag2 +
##     atr + adx + aaron + bb + clv + emv + macd + smi + volume +
##     volat + Risk_free_rate + Momentum + sarima_100_001 + sarima_110_001 +
##     sarima_120_001 + vol_forecast
## <environment: 0x00000215f48e8be8>
## [1] "****************************************"
## [1] "forecasted_ret:  0.00879942604629165"
## [1] "rmse:  0.0286320792512246"
## [1] "sharpe:  0.7680356305991"
## [1] "msr:  -0.559638067420589"
## [1] "****************************************"
## [1] "########################################"
## [1] "ticker: DE"
## Reordering variables and trying again:
## realized_returns ~ direction_lead + atr + adx + emv + macd +
##     mfi + sar + smi + volat + Excess_Retun_Mkt + Small_minus_Big +
##     Risk_free_rate + Momentum + sarima_100_001 + sarima_110_001 +
##     sarima_120_001 + vol_forecast
## <environment: 0x000002158ccb9b60>
## [1] "****************************************"
## [1] "forecasted_ret:  0.00868707589324324"
## [1] "rmse:  0.0297724896356422"
## [1] "sharpe:  1.20343167526832"
## [1] "msr:  -0.698481866823606"
## [1] "****************************************"
## [1] "########################################"
## [1] "ticker: EMR"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     log_returns_lag1 + log_returns_lag2 + log_returns_lag3 +
##     adx + clv + emv + macd + mfi + sar + smi + volat + Robus_minus_Weak +
##     Conservative_minus_Aggressive + Risk_free_rate + Momentum +
##     sarima_110_001 + sarima_120_001 + vol_forecast
## <environment: 0x000002158a7efba0>
## [1] "****************************************"
## [1] "forecasted_ret:  0.00415727942837838"
## [1] "rmse:  0.0307914042398209"
## [1] "sharpe:  0.601410782241761"
## [1] "msr:  -0.305878189295345"
## [1] "****************************************"
## [1] "########################################"
## [1] "ticker: ETN"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     log_returns_lag0 + log_returns_lag2 + atr + adx + clv + emv +
##     macd + mfi + sar + smi + Robus_minus_Weak + Momentum + sarima_100_001 +
##     sarima_110_001 + sarima_120_001 + vol_forecast + volat
## <environment: 0x0000021590e2c528>
## [1] "****************************************"
## [1] "forecasted_ret:  -0.00511627690183044"
## [1] "rmse:  0.0168426268102897"
## [1] "sharpe:  0.574351021179967"
## [1] "msr:  -0.306189661933098"
## [1] "****************************************"
```

```
## [1] "########################################"
## [1] "ticker: FDX"
## Reordering variables and trying again:
## realized_returns ~ direction_lead + log_returns_lag0 + atr +
##     adx + bb + clv + emv + mfi + sar + Excess_Retun_Mkt + Risk_free_rate +
##     sarima_120_001 + volat + vol_forecast
## <environment: 0x00000215897b5480>
## [1] "########################################"
## [1] "forecasted_ret:  0.00260139231144531"
## [1] "rmse:  0.0333194573850167"
## [1] "sharpe:  0.687184607903824"
## [1] "msr:  -0.346399808466754"
## [1] "########################################"
## [1] "########################################"
## [1] "ticker: GD"
## Reordering variables and trying again:
## realized_returns ~ direction_lead + discrete_returns + log_returns_lag1 +
##     atr + adx + aaron + emv + mfi + sar + volat + Excess_Retun_Mkt +
##     Small_minus_Big + High_minus_Low + Robus_minus_Weak + Conservative_minus_Aggressive +
##     Risk_free_rate + Momentum + sarima_100_001 + sarima_110_001 +
##     sarima_120_001 + vol_forecast
## <environment: 0x00000215884bc990>
## [1] "########################################"
## [1] "forecasted_ret:  -0.0179572698506873"
## [1] "rmse:  0.01897235461254"
## [1] "sharpe:  0.932449159643492"
## [1] "msr:  -0.592189890157435"
## [1] "########################################"
## [1] "########################################"
## [1] "ticker: GE"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + log_returns_lag2 +
##     atr + bb + emv + macd + mfi + volat + sarima_100_001 + sarima_110_001 +
##     vol_forecast
## <environment: 0x000002159125b500>
## [1] "########################################"
## [1] "forecasted_ret:  -0.00892763859864865"
## [1] "rmse:  0.0147865678355847"
## [1] "sharpe:  -1.34639524295417"
## [1] "msr:  0.479718049673436"
## [1] "########################################"
## [1] "########################################"
## [1] "ticker: HON"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     log_returns_lag0 + log_returns_lag1 + log_returns_lag2 +
##     atr + adx + aaron + bb + clv + macd + sar + smi + volume +
##     volat + Risk_free_rate + sarima_100_001 + sarima_110_001 +
##     sarima_120_001 + vol_forecast
## <environment: 0x00000215865e9cf8>
## [1] "########################################"
## [1] "forecasted_ret:  0.00552835523382836"
## [1] "rmse:  0.0305104747584328"
## [1] "sharpe:  0.689985463550163"
## [1] "msr:  -0.262927579077277"
## [1] "########################################"
## [1] "########################################"
## [1] "ticker: ITW"
## Reordering variables and trying again:
## realized_returns ~ direction_lead + log_returns_lag1 + aaron +
```

```
##     bb + chaikin_vol + clv + macd + mfi + smi + volat + Excess_Retun_Mkt +
##     Small_minus_Big + Risk_free_rate + sarima_100_001 + sarima_110_001 +
##     sarima_120_001 + vol_forecast
## <environment: 0x00000215f6f0ce90>
## [1] "**************************************"
## [1] "forecasted_ret:  0.0484259922346935"
## [1] "rmse:  0.0727986673852752"
## [1] "sharpe:  0.645705804079312"
## [1] "msr:  -0.272070878661102"
## [1] "**************************************"
## [1] "##############################################"
## [1] "ticker: LMT"
## Reordering variables and trying again:
## realized_returns ~ direction_lead + discrete_returns + adx +
##     bb + chaikin_vol + macd + mfi + sar + volume + High_minus_Low +
##     Conservative_minus_Aggressive + Risk_free_rate + sarima_100_001 +
##     sarima_110_001 + sarima_120_001 + volat + vol_forecast
## <environment: 0x000002158f5f54a0>
## [1] "**************************************"
## [1] "forecasted_ret:  0.0124045762688761"
## [1] "rmse:  0.0229869053559309"
## [1] "sharpe:  0.886855077620766"
## [1] "msr:  -0.494485428172536"
## [1] "**************************************"
## [1] "##############################################"
## [1] "ticker: MMM"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + atr + adx +
##     bb + clv + emv + macd + sar + smi + volume + volat + Excess_Retun_Mkt +
##     Small_minus_Big + Risk_free_rate + sarima_100_001 + sarima_110_001 +
##     sarima_120_001 + vol_forecast
## <environment: 0x000002158a2f98a0>
## [1] "**************************************"
## [1] "forecasted_ret:  0.0625700301997266"
## [1] "rmse:  0.0891229074383427"
## [1] "sharpe:  0.558571054297422"
## [1] "msr:  -0.21209743627236"
## [1] "**************************************"
## [1] "##############################################"
## [1] "ticker: NOC"
## Reordering variables and trying again:
## realized_returns ~ direction_lead + log_returns_lag0 + log_returns_lag1 +
##     log_returns_lag2 + adx + bb + chaikin_vol + emv + macd +
##     volat + Conservative_minus_Aggressive + Risk_free_rate +
##     sarima_100_001 + sarima_110_001 + sarima_120_001 + vol_forecast
## <environment: 0x00000215fcc51f38>
## [1] "**************************************"
## [1] "forecasted_ret:  0.00677380438848428"
## [1] "rmse:  0.0188258053803541"
## [1] "sharpe:  1.187465697052"
## [1] "msr:  -0.721821136388378"
## [1] "**************************************"
## [1] "##############################################"
## [1] "ticker: PH"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     bb + macd + mfi + sar + smi + volume + Excess_Retun_Mkt +
##     Robus_minus_Weak + Momentum + sarima_110_001 + sarima_120_001 +
##     vol_forecast + volat
## <environment: 0x000002158ea418e0>
```

```
## [1] "****************************************"
## [1] "forecasted_ret:  0.00477312520373501"
## [1] "rmse:  0.0423392247020034"
## [1] "sharpe:  0.624342756037441"
## [1] "msr:  -0.300322691526086"
## [1] "****************************************"
## [1] "########################################"
## [1] "ticker: RTX"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     log_returns_lag0 + log_returns_lag3 + atr + adx + chaikin_vol +
##     clv + emv + macd + sar + smi + volume + volat + Excess_Retun_Mkt +
##     Small_minus_Big + Conservative_minus_Aggressive + Risk_free_rate +
##     sarima_110_001 + sarima_120_001 + vol_forecast
## <environment: 0x0000021587d5ea40>
## [1] "****************************************"
## [1] "forecasted_ret:  -0.0357754600396533"
## [1] "rmse:  0.0286234450665977"
## [1] "sharpe:  0.596570490546308"
## [1] "msr:  -0.245421866427336"
## [1] "****************************************"
## [1] "########################################"
## [1] "ticker: UNP"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     log_returns_lag0 + log_returns_lag1 + log_returns_lag2 +
##     log_returns_lag3 + atr + chaikin_vol + clv + macd + smi +
##     Small_minus_Big + Conservative_minus_Aggressive + Risk_free_rate +
##     sarima_100_001 + sarima_110_001 + volat + vol_forecast
## <environment: 0x00000215fd12d580>
## [1] "****************************************"
## [1] "forecasted_ret:  0.00435942663503293"
## [1] "rmse:  0.0357274118958291"
## [1] "sharpe:  0.686410202206334"
## [1] "msr:  -0.329577950741156"
## [1] "****************************************"
## [1] "########################################"
## [1] "ticker: UPS"
## Reordering variables and trying again:
## realized_returns ~ adjusted_close + direction_lead + discrete_returns +
##     adx + bb + clv + emv + macd + mfi + sar + smi + volat + Excess_Retun_Mkt +
##     Risk_free_rate + Momentum + sarima_100_001 + sarima_110_001 +
##     sarima_120_001 + vol_forecast
## <environment: 0x000002158fac2a78>
## [1] "****************************************"
## [1] "forecasted_ret:  0.000625212825675676"
## [1] "rmse:  0.030264651829061"
## [1] "sharpe:  -0.00543853637562523"
## [1] "msr:  0.00187381658823174"
## [1] "****************************************"
## [1] "########################################"
## [1] "ticker: WM"
## Reordering variables and trying again:
## realized_returns ~ direction_lead + log_returns_lag0 + log_returns_lag1 +
##     log_returns_lag2 + adx + aaron + emv + macd + sar + smi +
##     volume + volat + Excess_Retun_Mkt + Small_minus_Big + High_minus_Low +
##     Robus_minus_Weak + Risk_free_rate + Momentum + sarima_100_001 +
##     sarima_110_001 + sarima_120_001 + vol_forecast
## <environment: 0x00000215873a9d60>
## [1] "****************************************"
```

```
## [1] "forecasted_ret:  0.112308267585507"
## [1] "rmse:  0.112823313139057"
## [1] "sharpe:  0.900756069075672"
## [1] "msr:  -0.37697740758592"
## [1] "**************************************"
## [1] "##########################################"
```

```
##    user  system elapsed
##   32.09    0.10   41.85
```

Now that all the models have been trained and the metrics recorded, we now simply choose the top 3 stocks based on the return, and the top 3 based on the best sharpe or modified sharpe ratio.

Let's first show some values for the `sector_tracker` object:

```
names(sector_tracker)
```

```
##  [1] "ADP" "BA"  "CAT" "CSX" "DE"  "EMR" "ETN" "FDX" "GD"  "GE"  "HON" "ITW"
## [13] "LMT" "MMM" "NOC" "PH"  "RTX" "UNP" "UPS" "WM"
```

```
names(sector_tracker[[1]])
```

```
## [1] "forecasted_ret" "sharpe"         "msr"            "rmse"
## [5] "data"
```

```
source(here("functions","modelling.R"))
```

```
## Getting holdings for SP500
```

```
# Obtain the top picks with the function
best_sector_stocks <- f_select_top_stocks(sector_tracker, n=3)
names(best_sector_stocks)
```

```
## [1] "BA"  "DE"  "NOC" "WM"  "MMM" "ITW"
```

```
best_sector_stocks[[1]]
```

```
## $forecasted_ret
## [1] -0.03902448
##
## $sharpe
## [1] 1.721857
##
## $msr
## [1] -1.066869
##
## $rmse
## [1] 0.04375968
##
## $data
##            realized_returns best_shifted_arima     volat vol_forecast
## 2016-10-05    -0.0112018165      -0.0367049329 0.1215014    0.2190119
## 2016-10-12     0.0224261163      -0.0056228624 0.1234677    0.2294365
## 2016-10-19     0.0664735107       0.0478255195 0.1197529    0.2335310
## 2016-10-26    -0.0334656437       0.0072485232 0.2165795    0.2328868
## 2016-11-02     0.0380189471       0.0086823197 0.2190119    0.2368343
```

```
## 2016-11-09     0.0092614691      0.0181248516 0.2294365    0.2391726
## 2016-11-16     0.0222847609      0.0072379994 0.2335310    0.2369913
## 2016-11-23     0.0054610837      0.0105886016 0.2328868    0.2381453
## 2016-11-30     0.0234994653     -0.0065053820 0.2368343    0.2360335
## 2016-12-07     0.0021387063      0.0052921501 0.2391726    0.1529022
## 2016-12-14     0.0192987330      0.0179230909 0.2369913    0.1477469
## 2016-12-21    -0.0088014756     -0.0181026899 0.2381453    0.1364027
## 2016-12-28     0.0160142231      0.0681376838 0.2360335    0.1564194
## 2017-01-04     0.0049053847     -0.0035621035 0.1529022    0.1615261
## 2017-01-11    -0.0067984489     -0.0342343179 0.1477469    0.1536365
## 2017-01-18     0.0555289040      0.0601387279 0.1364027    0.1458138
## 2017-01-25    -0.0204637306      0.0529350445 0.1564194    0.1456174
## 2017-02-01     0.0075887999      0.0531698993 0.1615261    0.1477425
## 2017-02-08     0.0329650995     -0.0230495995 0.1536365    0.1474300
## 2017-02-15     0.0351688512     -0.0583786007 0.1458138    0.1469237
## 2017-02-22     0.0476055433     -0.0107205311 0.1456174    0.1459196
## 2017-03-01    -0.0118696717      0.0128225824 0.1477425    0.1385115
## 2017-03-08    -0.0168124172      0.0036714424 0.1474300    0.1175867
## 2017-03-15    -0.0097277224     -0.0112755960 0.1469237    0.1050752
## 2017-03-22     0.0036659527      0.0176957224 0.1459196    0.1078830
## 2017-03-29    -0.0031012050      0.0328897058 0.1385115    0.1180396
## 2017-04-05    -0.0058335220      0.0096289816 0.1175867    0.1164732
## 2017-04-12     0.0132601553     -0.0005794957 0.1050752    0.1794254
## 2017-04-19     0.0183839000     -0.0362546573 0.1078830    0.1829514
## 2017-04-26     0.0092030236      0.0345293210 0.1180396    0.1825702
## 2017-05-03     0.0064804174      0.0454866271 0.1164732    0.1829054
## 2017-05-10    -0.0243132289     -0.0161706100 0.1794254    0.1852170
## 2017-05-17     0.0355502292      0.0221728066 0.1829514    0.1813394
## 2017-05-24     0.0127656470      0.0521182821 0.1825702    0.1835803
## 2017-05-31     0.0025019748     -0.0002082794 0.1829054    0.1826345
## 2017-06-07     0.0224986615     -0.0080258891 0.1852170    0.1790989
## 2017-06-14     0.0346862008      0.0323611682 0.1813394    0.1809055
## 2017-06-21     0.0022568373      0.0287244811 0.1835803    0.1121147
## 2017-06-28     0.0109111627      0.1271399360 0.1826345    0.1471265
## 2017-07-05     0.0226830146      0.0459655624 0.1790989    0.1549219
## 2017-07-12     0.0212795428     -0.0743124741 0.1809055    0.1600004
## 2017-07-19     0.1016786034      0.0023949534 0.1121147    0.1593822
## 2017-07-26     0.0190925216      0.0139224429 0.1471265    0.1561130
## 2017-08-02    -0.0094687147      0.0045435382 0.1549219    0.1555096
## 2017-08-09     0.0139013945     -0.0306740471 0.1600004    0.1570565
## 2017-08-16     0.0021022750      0.0256675405 0.1593822    0.1769497
## 2017-08-23     0.0099051563      0.0993300970 0.1561130    0.1816451
## 2017-08-30    -0.0245868473     -0.0028666379 0.1555096    0.1921472
## 2017-09-06     0.0306813563     -0.0342926816 0.1570565    0.1672635
## 2017-09-13     0.0544173875      0.0302970498 0.1769497    0.1626518
## 2017-09-20    -0.0007048060     -0.0008560105 0.1816451    0.1638539
## 2017-09-27     0.0018784430     -0.0248961646 0.1921472    0.2126903
## 2017-10-04     0.0219654005      0.0018723560 0.1672635    0.2154109
## 2017-10-11    -0.0053692256      0.0401477593 0.1626518    0.2146093
## 2017-10-18    -0.0062494423      0.0025485626 0.1638539    0.2200428
## 2017-10-25     0.0003094814     -0.0082785538 0.2126903    0.2024393
## 2017-11-01     0.0269829947      0.0268952716 0.2154109    0.2000690
## 2017-11-08    -0.0048957717      0.0430303809 0.2146093    0.1989672
## 2017-11-15     0.0086743304      0.0640654010 0.2200428    0.2080107
## 2017-11-22     0.0155301951      0.0216367191 0.2024393    0.2063782
## 2017-11-29     0.0327657257     -0.0351302451 0.2000690    0.2006618
## 2017-12-06     0.0476139023     -0.0066790252 0.1989672    0.1475433
## 2017-12-13     0.0205521877      0.1042714052 0.2080107    0.1571894
## 2017-12-20    -0.0076831127      0.1418092446 0.2063782    0.1904561
## 2017-12-27     0.0073473482     -0.0806142746 0.2006618    0.1872014
```

```
## 2018-01-03     0.0727109998      -0.0020100186 0.1475433    0.1998259
## 2018-01-10     0.0916815659       0.0279441988 0.1571894    0.2305311
## 2018-01-17    -0.0476099298      -0.0504747277 0.1904561    0.2286338
## 2018-01-24     0.0571367848       0.0390583816 0.1872014    0.2348489
## 2018-01-31    -0.0177944155       0.0470218389 0.1998259    0.2478484
## 2018-02-07    -0.0045133538      -0.0618705165 0.2305311    0.2589373
## 2018-02-14     0.0215721934      -0.0990154505 0.2286338    0.3118876
## 2018-02-21     0.0275423844       0.0389570118 0.2348489    0.3122960
## 2018-02-28    -0.0427840879      -0.0287536518 0.2478484    0.3217395
## 2018-03-07    -0.0495597325      -0.0011621146 0.2589373    0.3065227
## 2018-03-14     0.0204993357       0.0412425484 0.3118876    0.2929860
## 2018-03-21    -0.0519962768       0.0389944257 0.3122960    0.2809838
## 2018-03-28     0.0229214278       0.0667346484 0.3217395    0.2703776
##            month_index
## 2016-10-05          10
## 2016-10-12          10
## 2016-10-19          10
## 2016-10-26          10
## 2016-11-02          11
## 2016-11-09          11
## 2016-11-16          11
## 2016-11-23          11
## 2016-11-30          11
## 2016-12-07          12
## 2016-12-14          12
## 2016-12-21          12
## 2016-12-28          12
## 2017-01-04          13
## 2017-01-11          13
## 2017-01-18          13
## 2017-01-25          13
## 2017-02-01          14
## 2017-02-08          14
## 2017-02-15          14
## 2017-02-22          14
## 2017-03-01          15
## 2017-03-08          15
## 2017-03-15          15
## 2017-03-22          15
## 2017-03-29          15
## 2017-04-05          16
## 2017-04-12          16
## 2017-04-19          16
## 2017-04-26          16
## 2017-05-03          17
## 2017-05-10          17
## 2017-05-17          17
## 2017-05-24          17
## 2017-05-31          17
## 2017-06-07          18
## 2017-06-14          18
## 2017-06-21          18
## 2017-06-28          18
## 2017-07-05          19
## 2017-07-12          19
## 2017-07-19          19
## 2017-07-26          19
## 2017-08-02          20
## 2017-08-09          20
## 2017-08-16          20
```

```
## 2017-08-23          20
## 2017-08-30          20
## 2017-09-06          21
## 2017-09-13          21
## 2017-09-20          21
## 2017-09-27          21
## 2017-10-04          22
## 2017-10-11          22
## 2017-10-18          22
## 2017-10-25          22
## 2017-11-01          23
## 2017-11-08          23
## 2017-11-15          23
## 2017-11-22          23
## 2017-11-29          23
## 2017-12-06          24
## 2017-12-13          24
## 2017-12-20          24
## 2017-12-27          24
## 2018-01-03          25
## 2018-01-10          25
## 2018-01-17          25
## 2018-01-24          25
## 2018-01-31          25
## 2018-02-07          26
## 2018-02-14          26
## 2018-02-21          26
## 2018-02-28          26
## 2018-03-07          27
## 2018-03-14          27
## 2018-03-21          27
## 2018-03-28          27
```

```
# pack the data into a format for modelling (only keep the data)
top_sector_stocks <- lapply(best_sector_stocks, function(x) x$data)
top_sector_stocks[[1]]
```

```
##            realized_returns best_shifted_arima     volat vol_forecast
## 2016-10-05    -0.0112018165      -0.0367049329 0.1215014    0.2190119
## 2016-10-12     0.0224261163      -0.0056228624 0.1234677    0.2294365
## 2016-10-19     0.0664735107       0.0478255195 0.1197529    0.2335310
## 2016-10-26    -0.0334656437       0.0072485232 0.2165795    0.2328868
## 2016-11-02     0.0380189471       0.0086823197 0.2190119    0.2368343
## 2016-11-09     0.0092614691       0.0181248516 0.2294365    0.2391726
## 2016-11-16     0.0222847609       0.0072379994 0.2335310    0.2369913
## 2016-11-23     0.0054610837       0.0105886016 0.2328868    0.2381453
## 2016-11-30     0.0234994653      -0.0065053820 0.2368343    0.2360335
## 2016-12-07     0.0021387063       0.0052921501 0.2391726    0.1529022
## 2016-12-14     0.0192987330       0.0179230909 0.2369913    0.1477469
## 2016-12-21    -0.0088014756      -0.0181026899 0.2381453    0.1364027
## 2016-12-28     0.0160142231       0.0681376838 0.2360335    0.1564194
## 2017-01-04     0.0049053847      -0.0035621035 0.1529022    0.1615261
## 2017-01-11    -0.0067984489      -0.0342343179 0.1477469    0.1536365
## 2017-01-18     0.0555289040       0.0601387279 0.1364027    0.1458138
## 2017-01-25    -0.0204637306       0.0529350445 0.1564194    0.1456174
## 2017-02-01     0.0075887999       0.0531698993 0.1615261    0.1477425
## 2017-02-08     0.0329650995      -0.0230495995 0.1536365    0.1474300
## 2017-02-15     0.0351688512      -0.0583786007 0.1458138    0.1469237
## 2017-02-22     0.0476055433      -0.0107205311 0.1456174    0.1459196
```

```
## 2017-03-01    -0.0118696717       0.0128225824 0.1477425    0.1385115
## 2017-03-08    -0.0168124172       0.0036714424 0.1474300    0.1175867
## 2017-03-15    -0.0097277224      -0.0112755960 0.1469237    0.1050752
## 2017-03-22     0.0036659527       0.0176957224 0.1459196    0.1078830
## 2017-03-29    -0.0031012050       0.0328897058 0.1385115    0.1180396
## 2017-04-05    -0.0058335220       0.0096289816 0.1175867    0.1164732
## 2017-04-12     0.0132601553      -0.0005794957 0.1050752    0.1794254
## 2017-04-19     0.0183839000      -0.0362546573 0.1078830    0.1829514
## 2017-04-26     0.0092030236       0.0345293210 0.1180396    0.1825702
## 2017-05-03     0.0064804174       0.0454866271 0.1164732    0.1829054
## 2017-05-10    -0.0243132289      -0.0161706100 0.1794254    0.1852170
## 2017-05-17     0.0355502292       0.0221728066 0.1829514    0.1813394
## 2017-05-24     0.0127656470       0.0521182821 0.1825702    0.1835803
## 2017-05-31     0.0025019748      -0.0002082794 0.1829054    0.1826345
## 2017-06-07     0.0224986615      -0.0080258891 0.1852170    0.1790989
## 2017-06-14     0.0346862008       0.0323611682 0.1813394    0.1809055
## 2017-06-21     0.0022568373       0.0287244811 0.1835803    0.1121147
## 2017-06-28     0.0109111627       0.1271399360 0.1826345    0.1471265
## 2017-07-05     0.0226830146       0.0459655624 0.1790989    0.1549219
## 2017-07-12     0.0212795428      -0.0743124741 0.1809055    0.1600004
## 2017-07-19     0.1016786034       0.0023949534 0.1121147    0.1593822
## 2017-07-26     0.0190925216       0.0139224429 0.1471265    0.1561130
## 2017-08-02    -0.0094687147       0.0045435382 0.1549219    0.1555096
## 2017-08-09     0.0139013945      -0.0306740471 0.1600004    0.1570565
## 2017-08-16     0.0021022750       0.0256675405 0.1593822    0.1769497
## 2017-08-23     0.0099051563       0.0993300970 0.1561130    0.1816451
## 2017-08-30    -0.0245868473      -0.0028666379 0.1555096    0.1921472
## 2017-09-06     0.0306813563      -0.0342926816 0.1570565    0.1672635
## 2017-09-13     0.0544173875       0.0302970498 0.1769497    0.1626518
## 2017-09-20    -0.0007048060      -0.0008560105 0.1816451    0.1638539
## 2017-09-27     0.0018784430      -0.0248961646 0.1921472    0.2126903
## 2017-10-04     0.0219654005       0.0018723560 0.1672635    0.2154109
## 2017-10-11    -0.0053692256       0.0401477593 0.1626518    0.2146093
## 2017-10-18    -0.0062494423       0.0025485626 0.1638539    0.2200428
## 2017-10-25     0.0003094814      -0.0082785538 0.2126903    0.2024393
## 2017-11-01     0.0269829947       0.0268952716 0.2154109    0.2000690
## 2017-11-08    -0.0048957717       0.0430303809 0.2146093    0.1989672
## 2017-11-15     0.0086743304       0.0640654010 0.2200428    0.2080107
## 2017-11-22     0.0155301951       0.0216367191 0.2024393    0.2063782
## 2017-11-29     0.0327657257      -0.0351302451 0.2000690    0.2006618
## 2017-12-06     0.0476139023      -0.0066790252 0.1989672    0.1475433
## 2017-12-13     0.0205521877       0.1042714052 0.2080107    0.1571894
## 2017-12-20    -0.0076831127       0.1418092446 0.2063782    0.1904561
## 2017-12-27     0.0073473482      -0.0806142746 0.2006618    0.1872014
## 2018-01-03     0.0727109998      -0.0020100186 0.1475433    0.1998259
## 2018-01-10     0.0916815659       0.0279441988 0.1571894    0.2305311
## 2018-01-17    -0.0476099298      -0.0504747277 0.1904561    0.2286338
## 2018-01-24     0.0571367848       0.0390583816 0.1872014    0.2348489
## 2018-01-31    -0.0177944155       0.0470218389 0.1998259    0.2478484
## 2018-02-07    -0.0045133538      -0.0618705165 0.2305311    0.2589373
## 2018-02-14     0.0215721934      -0.0990154505 0.2286338    0.3118876
## 2018-02-21     0.0275423844       0.0389570118 0.2348489    0.3122960
## 2018-02-28    -0.0427840879      -0.0287536518 0.2478484    0.3217395
## 2018-03-07    -0.0495597325      -0.0011621146 0.2589373    0.3065227
## 2018-03-14     0.0204993357       0.0412425484 0.3118876    0.2929860
## 2018-03-21    -0.0519962768       0.0389944257 0.3122960    0.2809838
## 2018-03-28     0.0229214278       0.0667346484 0.3217395    0.2703776
##            month_index
## 2016-10-05          10
## 2016-10-12          10
```

```
## 2016-10-19          10
## 2016-10-26          10
## 2016-11-02          11
## 2016-11-09          11
## 2016-11-16          11
## 2016-11-23          11
## 2016-11-30          11
## 2016-12-07          12
## 2016-12-14          12
## 2016-12-21          12
## 2016-12-28          12
## 2017-01-04          13
## 2017-01-11          13
## 2017-01-18          13
## 2017-01-25          13
## 2017-02-01          14
## 2017-02-08          14
## 2017-02-15          14
## 2017-02-22          14
## 2017-03-01          15
## 2017-03-08          15
## 2017-03-15          15
## 2017-03-22          15
## 2017-03-29          15
## 2017-04-05          16
## 2017-04-12          16
## 2017-04-19          16
## 2017-04-26          16
## 2017-05-03          17
## 2017-05-10          17
## 2017-05-17          17
## 2017-05-24          17
## 2017-05-31          17
## 2017-06-07          18
## 2017-06-14          18
## 2017-06-21          18
## 2017-06-28          18
## 2017-07-05          19
## 2017-07-12          19
## 2017-07-19          19
## 2017-07-26          19
## 2017-08-02          20
## 2017-08-09          20
## 2017-08-16          20
## 2017-08-23          20
## 2017-08-30          20
## 2017-09-06          21
## 2017-09-13          21
## 2017-09-20          21
## 2017-09-27          21
## 2017-10-04          22
## 2017-10-11          22
## 2017-10-18          22
## 2017-10-25          22
## 2017-11-01          23
## 2017-11-08          23
## 2017-11-15          23
## 2017-11-22          23
## 2017-11-29          23
## 2017-12-06          24
```

```
## 2017-12-13          24
## 2017-12-20          24
## 2017-12-27          24
## 2018-01-03          25
## 2018-01-10          25
## 2018-01-17          25
## 2018-01-24          25
## 2018-01-31          25
## 2018-02-07          26
## 2018-02-14          26
## 2018-02-21          26
## 2018-02-28          26
## 2018-03-07          27
## 2018-03-14          27
## 2018-03-21          27
## 2018-03-28          27
```

```
save(top_sector_stocks, file = here("tests","jair", "top_sector_stocks.rda"))
```

Aside: Format for Portfolio Optimization

```
## This chunk of code simply obtains some portfolio stock tickers
## in a way that will be similar to the final result

# repack the portfolio (repeated from before)
portfolio <- list(tickers = initial_tickers,
                  weights = weights,
                  capital = initial_capital,
                  returns = returns,
                  data = NA
                  )
portfolio
```

```
## $tickers
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [26] NA NA NA NA NA NA NA NA NA NA NA
##
## $weights
##  [1] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
##  [7] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
## [13] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
## [19] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
## [25] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
## [31] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778
##
## $capital
## [1] 5e+05
##
## $returns
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [26] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [51] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
##
## $data
## [1] NA
```

The following simulates best tickers that would be obtained after modelling procedure for all sectors

```
# Set up backtesting simulation parameters
sample_xts <- sp500_stocks$Industrials$ADP
sectors <- names(sp500_stocks)
N_sector_best_stocks <- 3
tau <- 3

# store ticker for current portfolio
cur_tickers <- rep(NA, num_tickers)

# store actual data for each run
portf_stocks_data <- as.list(rep(NA, length(sectors)))
names(portf_stocks_data) <- sectors

# keep index counter for sectors
i_sector <- 1

print("")
```

```
## [1] ""
```

```
print("(2) PORTFOLIO_LOOP:")
```

```
## [1] "(2) PORTFOLIO_LOOP:"
```

```
# loop through all the sectors
for(G in sectors){

  # return top 3 best stocks (xts data) according to procedure
  top_sector_stocks <- SECTOR_PROCEDURE(G, tau)

  # assign best stocks to portfolio (NEED TO UPDATE LOGIC!)
  i_replace <- rep(i_sector, num_top_pick) + seq(0, num_top_pick-1) # indexes to choose from
  cur_tickers[i_replace] <- names(top_sector_stocks)
  i_sector <- i_sector + num_top_pick

  # assign the data to the portfolio
  portf_stocks_data[[G]] <- top_sector_stocks
}
```

```
## [1] "SECTOR_PROCEDURE(G=Industrials, tau=3)"
## [1] "  MODELLING_PROCEDURE(list_train_val_sector)"
## [1] "SECTOR_PROCEDURE(G=Health Care, tau=3)"
## [1] "  MODELLING_PROCEDURE(list_train_val_sector)"
## [1] "SECTOR_PROCEDURE(G=Information Technology, tau=3)"
## [1] "  MODELLING_PROCEDURE(list_train_val_sector)"
## [1] "SECTOR_PROCEDURE(G=Communication Services, tau=3)"
## [1] "  MODELLING_PROCEDURE(list_train_val_sector)"
## [1] "SECTOR_PROCEDURE(G=Financials, tau=3)"
## [1] "  MODELLING_PROCEDURE(list_train_val_sector)"
## [1] "SECTOR_PROCEDURE(G=Consumer Discretionary, tau=3)"
## [1] "  MODELLING_PROCEDURE(list_train_val_sector)"
```

```
# Portfolio tickers get updated
portfolio$tickers <- cur_tickers
```

```
# unlist data best stocks data format into a singles list
portf_data <- f_unlist_portf_data(portf_stocks_data)

# assign list to portfolio
portfolio$data <- portf_data
```

Data format for portfoli optimization

Note that at this point, the portfolio will have the tickers and the weights attributes.

```
# Checko out the resulting portfolio
portfolio$tickers
```

```
##  [1] "BA"   "UPS"  "NOC"  "ETN"  "CSX"  "EMR"  "ISRG" "ABBV" "SYK"  "BMY"
## [11] "REGN" "TMO"  "CSCO" "ACN"  "CRM"  "AAPL" "ORCL" "PANW" "DIS"  "META"
## [21] "GOOG" "TMUS" "T"    "EA"   "PGR"  "CB"   "FI"   "ICE"  "BX"   "MA"
## [31] "HD"   "ROST" "CMG"  "SBUX" "TJX"  "NKE"
```

```
portfolio$capital
```

```
## [1] 5e+05
```

```
portfolio$returns
```

```
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [26] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [51] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
```

```
print("")
```

```
## [1] ""
```

```
# inspect the names and data for one stock
names(portfolio$data)
```

```
##  [1] "BA"   "UPS"  "NOC"  "ETN"  "CSX"  "EMR"  "ISRG" "ABBV" "SYK"  "BMY"
## [11] "REGN" "TMO"  "CSCO" "ACN"  "CRM"  "AAPL" "ORCL" "PANW" "DIS"  "META"
## [21] "GOOG" "TMUS" "T"    "EA"   "PGR"  "CB"   "FI"   "ICE"  "BX"   "MA"
## [31] "HD"   "ROST" "CMG"  "SBUX" "TJX"  "NKE"
```

```
head(portfolio$data[[1]])
```

```
##            adjusted_close direction_lead discrete_returns realized_returns
## 2016-01-06       124.5074             -1               NA    -0.0802827826
## 2016-01-13       114.9023             -1      -0.07714466    -0.0477992191
## 2016-01-20       109.5393             -1      -0.04667482    -0.0465904308
## 2016-01-27       104.5528              1      -0.04552176     0.0443771574
## 2016-02-03       109.2971             -1       0.04537655    -0.0370547637
## 2016-02-10       105.3212             -1      -0.03637664    -0.0001719132
##            log_returns_lag0 log_returns_lag1 log_returns_lag2 log_returns_lag3
## 2016-01-06               NA               NA               NA               NA
## 2016-01-13      -0.08028278               NA               NA               NA
## 2016-01-20      -0.04779922      -0.08028278               NA               NA
## 2016-01-27      -0.04659043      -0.04779922      -0.08028278               NA
```

```
## 2016-02-03       0.04437716      -0.04659043      -0.04779922      -0.08028278
## 2016-02-10      -0.03705476       0.04437716      -0.04659043      -0.04779922
##            atr adx aaron bb chaikin_vol clv emv macd mfi      sar smi   volume
## 2016-01-06  NA  NA    NA NA          NA  NA  NA   NA  NA 135.1129  NA  4254100
## 2016-01-13  NA  NA   -50 NA          NA  NA  NA   NA  NA 139.3500  NA  5073100
## 2016-01-20  NA  NA  -100 NA          NA  NA  NA   NA  NA 139.3500  NA  7636600
## 2016-01-27  NA  NA  -100 NA          NA  NA  NA   NA  NA 138.6616  NA 27945900
## 2016-02-03  NA  NA   -50 NA          NA  NA  NA   NA  NA 137.3367  NA  7361900
## 2016-02-10  NA  NA    50 NA          NA  NA  NA   NA  NA 136.0913  NA  4888600
##            volat month_index Excess_Retun_Mkt Small_minus_Big High_minus_Low
## 2016-01-06    NA           1          -0.0135         -0.0023         0.0000
## 2016-01-13    NA           1          -0.0267         -0.0062         0.0081
## 2016-01-20    NA           1          -0.0094          0.0173        -0.0127
## 2016-01-27    NA           1          -0.0111         -0.0042         0.0171
## 2016-02-03    NA           2           0.0046         -0.0025         0.0047
## 2016-02-10    NA           2           0.0001         -0.0021        -0.0055
##            Robus_minus_Weak Conservative_minus_Aggressive Risk_free_rate
## 2016-01-06           0.0015                         0.0004          0e+00
## 2016-01-13           0.0040                         0.0063          0e+00
## 2016-01-20           0.0008                        -0.0052          0e+00
## 2016-01-27          -0.0013                         0.0092          0e+00
## 2016-02-03           0.0041                         0.0032          1e-05
## 2016-02-10          -0.0030                        -0.0069          1e-05
##            Momentum
## 2016-01-06   0.0192
## 2016-01-13   0.0016
## 2016-01-20  -0.0011
## 2016-01-27  -0.0048
## 2016-02-03  -0.0241
## 2016-02-10   0.0065
```