Portfolio Factor Exposure Attribution

personal mini-project, 2024-12-28

Yu-An(Aaron) Chen

Introduction

This document demonstrates a framework for gathering and combining the Fama-French 5 factors (plus risk-free rate) and Momentum data into a cohesive dataset, then using weighted regression to estimate how assets load onto these factors. The model provides quantitative insights into which factors are driving an asset's performance.

By showing factor exposures, it can help portfolio managers and analysts in several ways, specifically in risk management, performance attribution, and factor tilt/strategy design.

Preliminary Data Wrangling

In this section, we read the Fama-French 5-factor data and a separate momentum dataset from the Kenneth R. French Data Library at from Dartmouth University (https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html#Research). At time of writing, they include data from July 1963 to November 2024.

We then merge both datasets together based on their date columns. Finally, we filter the data to only include records after January 1, 2015.

```
df = read.csv('FF_FiveFactors.csv', header = FALSE)[-c(1,2),]
names(df) = c('date', unlist(df[1,-c(1)]))
df = df[-c(1),]
df$date = as.Date(df$date, format = '%Y%m%d')
df = subset(df, date > '2015-01-01')
df[-1] <- lapply(df[-1], function(x) as.numeric(as.character(x)))

mom = read.csv('momentum.csv', header = FALSE)[-c(1:12),-c(3)]
names(mom) = c('date', 'MOM')
mom$MOM = as.numeric(mom$MOM)
mom$date = as.Date(mom$date, format = '%Y%m%d')
mom = subset(mom, date > '2015-01-01')

df <- merge(df, mom, by = 'date', all.x = TRUE)
```

Factor Returns

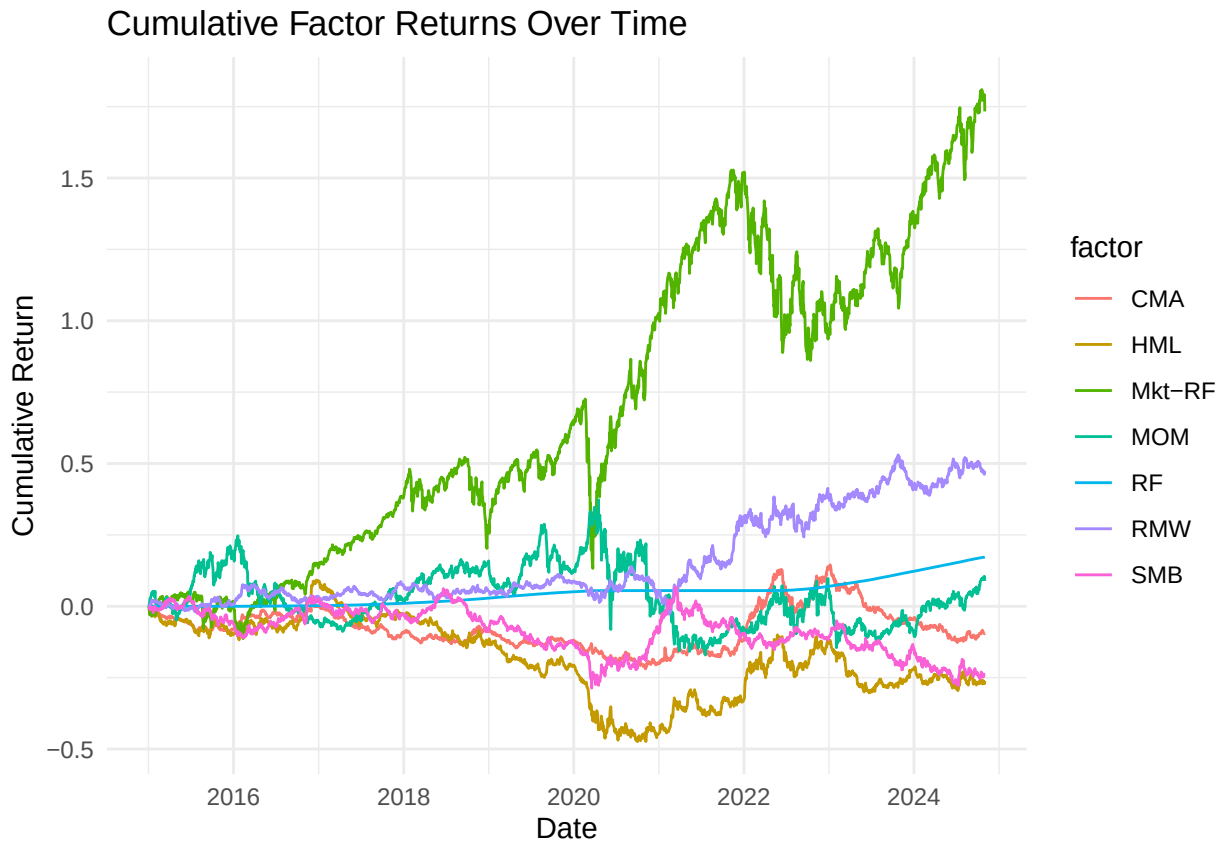
The following code defines a function to pivot the factor returns into a long format, compute cumulative returns, and plot them over time. This helps visualize how different factors evolve from the chosen start date.

```r
plot_cumulative_returns <- function(df){
df_long <- df %>%
  pivot_longer(cols = c(`Mkt-RF`, SMB, HML, RMW, CMA, MOM, RF),
               names_to = "factor",
               values_to = "value")

df_cum <- df_long %>%
  group_by(factor) %>%
  arrange(date) %>%
  mutate(cumulative_return = cumprod(1 + (value / 100)) - 1) %>%
  ungroup()

ggplot(df_cum, aes(x = date, y = cumulative_return, color = factor)) +
  geom_line(linewidth = 0.5) +
  labs(title = "Cumulative Factor Returns Over Time",
       x = "Date",
       y = "Cumulative Return") +
  theme_minimal()
}

plot_cumulative_returns(df)
```



Asset Data Wrangling

Here, we define **get_asset_data()**, which does the following:

1. Aggregates Fama-French factor returns into weekly data.
2. Uses quantmod to fetch historical prices for a list of tickers.
3. Computes weekly returns for those tickers.
4. Merges the asset returns with the weekly Fama-French data.
5. Computes excess returns by subtracting the risk-free rate.

```
get_asset_data <- function(tickers, df, n_cores = detectCores() - 1) {
  # Process Fama-French factors first
  df_weekly <- df %>%
    mutate(date = floor_date(date, "week") + days(5)) %>%  # Changed to days()
    group_by(date) %>%
    summarise(
      `Mkt-RF` = (prod(1 + `Mkt-RF`/100) - 1) * 100,
      SMB      = (prod(1 + SMB/100) - 1) * 100,
      HML      = (prod(1 + HML/100) - 1) * 100,
      RMW      = (prod(1 + RMW/100) - 1) * 100,
      CMA      = (prod(1 + CMA/100) - 1) * 100,
      MOM      = (prod(1 + MOM/100) - 1) * 100,
      RF       = sum(RF)
    ) %>%
    ungroup()

  end_date <- max(df_weekly$date)
  start_date <- min(df_weekly$date)

  # Process tickers in parallel using mclapply
  all_returns <- mclapply(tickers, function(tick) {
    invisible(getSymbols(tick, from = start_date, to = end_date, auto.assign = TRUE))
    stock_xts <- get(tick)
    daily_prices <- Cl(stock_xts)
    stock_ret <- weeklyReturn(daily_prices, type = "arithmetic") * 100
    stock_ret_dates <- index(stock_ret)

    data.frame(
      date = floor_date(stock_ret_dates, "week") + days(5),
      stock = tick,
      return = as.numeric(stock_ret)
    )
  }, mc.cores = n_cores) %>%
    bind_rows()

  # Merge with Fama-French factors and calculate excess returns
  merged_data <- merge(all_returns, df_weekly, by='date', all.x = TRUE) %>%
    mutate(excess_return = return - RF)

  return(merged_data)
}
```

Estimate Factor and Portfolio Exposures

In this code, we define two functions:

get_asset_exposures(): Performs weighted regressions on each asset's excess returns, using Fama-French factors plus momentum. The weights decay for older data. We also exclude factors with coefficients that are statistically insignificant exposures to retain only meaningful exposures.

get_portfolio_exposure(): Takes a dataframe of exposures and calculates the portfolio's weighted exposures by taking the weighted average of factor loadings.

```
get_asset_exposures <- function(tickers, dollar_exposures,factor_data, decay_rate) {

  merged_data <- get_asset_data(tickers, factor_data)

  final_date <- max(merged_data$date)
  merged_data <- merged_data[order(merged_data$date), ]

  joined_with_value <- mapply(list, tickers, dollar_exposures, SIMPLIFY=F)

  # Parallel computation over tickers
  results_list <- lapply(c(1:length(joined_with_value)), function(i) {
    tick = joined_with_value[[i]][[1]]

    stock_df <- subset(merged_data, stock == tick & !is.na(excess_return))
    stock_df <- stock_df[order(stock_df$date), ]

    # Calculate weights
    stock_df$months_from_end <- as.numeric((final_date - stock_df$date) / 30)
    stock_df$weight <- decay_rate^(stock_df$months_from_end)

    # Run weighted regression
    fit <- lm(excess_return ~ `Mkt-RF` + SMB + HML + RMW + CMA + MOM,
              data = stock_df, weights = stock_df$weight)

    # zero out non-significant exposures
    coefs <- coef(summary(fit))
    alpha <- round(ifelse(coefs["(Intercept)", "Pr(>|t|)"] > 0.05,
                          0, coefs["(Intercept)", "Estimate"]), 4)
    market <- round(coefs["`Mkt-RF`", "Estimate"], 4)
    size <- round(ifelse(coefs["SMB", "Pr(>|t|)"] > 0.05,
                         0, coefs["SMB", "Estimate"]), 4)
    value <- round(ifelse(coefs["HML", "Pr(>|t|)"] > 0.05,
                          0, coefs["HML", "Estimate"]), 4)
    prof <- round(ifelse(coefs["RMW", "Pr(>|t|)"] > 0.05,
                         0, coefs["RMW", "Estimate"]), 4)
    mom <- round(ifelse(coefs["MOM", "Pr(>|t|)"] > 0.05,
                        0, coefs["MOM", "Estimate"]), 4)
    invest <- round(ifelse(coefs["CMA", "Pr(>|t|)"] > 0.05,
                           0, coefs["CMA", "Estimate"]), 4)

    # Return the results for this ticker
    data.frame(
      asset = tick,
      alpha = alpha,
```

```
      market = market,
      size = size,
      value = value,
      profit = prof,
      investment = invest,
      momentum = mom,
      r2 = round(summary(fit)$r.squared, 4),
      stringsAsFactors = FALSE,
      dollar_exposure = joined_with_value[[i]][[2]]
    )
  })

  # Combine all results
  exposures <- do.call(rbind, results_list)
  exposures = exposures[order(-1*exposures$dollar_exposure),]
  rownames(exposures) = c(1:nrow(exposures))
  return(exposures)
}

get_portfolio_exposure <- function(data) {
  factor_cols <- c("alpha", "market", "size", "value",
                   "profit", "investment", "momentum", "r2")

  w <- data$dollar_exposure / sum(data$dollar_exposure, na.rm = TRUE)

  # Compute the weighted average for each factor column
  weighted_exposures <- round(colSums(data[factor_cols] * w, na.rm = TRUE), 4)
  names(weighted_exposures)[length(weighted_exposures)] = "r2_est"
  return(weighted_exposures)
}
```

Example Application

Next, we specify a small set of tickers (some are ETFs and some are individual assets) and assign hypothetical dollar exposures to each. We then call the functions from above to retrieve asset exposures.

```
tickers <- c("SPY", "VEU", "AVUV", "VB",
             "XBI", "AVEM", "TLT", "IAK",
             "BKNG")

# random assignment of exposures for demo purposes
set.seed(10)
dollar_exposures <- runif(9, 1000, 20000)

(exposures <- get_asset_exposures(tickers, dollar_exposures,
                                  df, decay_rate = 0.97))
```

```
##   asset   alpha market    size   value  profit investment momentum     r2
## 1    VB -0.0576 1.0358  0.5874  0.1830 -0.0515    -0.0666   0.0000 0.9727
## 2  BKNG  0.0000 1.1949  0.0000  0.3562  0.0000    -0.3479   0.0000 0.4777
## 3   SPY -0.0308 0.9897 -0.0853  0.0000  0.0704     0.0498   0.0000 0.9942
```

```
## 4  AVUV  0.0000 1.0766  0.8476  0.5656  0.1695     0.0000   0.0000 0.9736
## 5   VEU -0.1217 0.7725  0.0000  0.1000 -0.1217     0.0000  -0.0998 0.7518
## 6   TLT  0.0000 0.0334  0.0000 -0.4458  0.0000     0.3976  -0.1119 0.0990
## 7   IAK  0.0000 0.8777  0.0000  0.6090  0.0000     0.0000   0.0000 0.7134
## 8  AVEM  0.0000 0.7228  0.0000  0.0000 -0.2109     0.0000   0.0000 0.5574
## 9   XBI  0.0000 0.7789  0.8848 -0.5046 -0.9942     0.0000  -0.1176 0.7538
##   dollar_exposure
## 1       14168.940
## 2       12700.757
## 3       10642.086
## 4        9111.246
## 5        6828.602
## 6        6216.080
## 7        6173.796
## 8        5283.296
## 9        2617.583
```

```
(portfolio_exposures <- get_portfolio_exposure(exposures))
```

```
##      alpha     market       size      value     profit investment   momentum
##    -0.0268     0.9079     0.2367     0.1711    -0.0405    -0.0320    -0.0228
##     r2_est
##     0.7373
```

Conclusion

Overall, this project outlines how to:

1. Acquire and cleanse Fama-French factor data plus a momentum factor.
2. Aggregate daily returns into weekly intervals for both factors and assets.
3. Use weighted regressions to capture the decaying influence of older data.
4. Compute factor exposures and alpha estimates on both an asset and portfolio level.

Despite its utility, there are several enhancements that can improve the model's robustness and relevance:

Refine the Decay Mechanism Fine-tune the decay rate or explore alternative weighting schemes (e.g., exponentially-weighted rolling windows) to better reflect shifting market dynamics.

Expand the Factor Universe Incorporate additional factors such as liquidity, quality, low volatility, or even macroeconomic variables. This can provide more comprehensive coverage of potential return drivers.

Incorporate Portfolio Optimization Beyond merely estimating factor exposures, incorporate an optimization layer (e.g., mean-variance or factor-based optimization) to construct or rebalance portfolios with targeted factor profiles.

By following these suggested steps, investors can further refine their strategies and more confidently manage the factor risks and opportunities within their portfolios.