TM4

CW

2025-06-10

R Markdown

```r
load("TM4_data.RData")


### Needed Libraries ###

options(scipen = 999) # Disable scientific notation

library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(broom)
library(tidyr)

### Task 1: Merging the datasets ###

# We need
# ESG Indicator
# 1) ESG Rating

# Performance Indicator:
# Market is the S&P 500
# 1) Quarterly Abnormal Returns --> log return - capm beta * market log return
# 2) Daily Abnormal Return
# 3) Volatility
# 4) Idiosyncratic Volatility

# Corporate Finance Indicators:
# Statistics are winsorized at 1% level
# 1)Tobin's q
# 2)Size
# 3)Cash
# 4)Leverage
# 5)ROE
# 6)Advertising
# 7)Dividend Yield

# To merge all three dataframes accordingly, we need to first caluclate the daily
# daily and quarterly abnormal log-return, volatility and idiosyncratic volatility

## 1. Handle the Date Column ##

# First check which rows are numeric serials
is_numeric_date <- grepl("^\\d{5}$", stock_data$date)
# Convert to Date using Excel origin
stock_data$date[is_numeric_date] <- as.character(
  as.Date(as.numeric(stock_data$date[is_numeric_date]), origin = "1899-12-30")
)
stock_data$date <- trimws(as.character(stock_data$date))

# Parse each format conditionally
# Use ISO-style parser if format is yyyy-mm-dd
iso_format <- grepl("^\\d{4}-\\d{2}-\\d{2}$", stock_data$date)
# Use d/m/y parser for dd/mm/yyyy
euro_format <- grepl("^\\d{2}/\\d{2}/\\d{4}$", stock_data$date)

# Initialize empty column
stock_data$parsed_date <- NA

# Parse ISO format
stock_data$parsed_date[iso_format] <- stock_data$date[iso_format]

# Parse European-style format
stock_data$parsed_date[euro_format] <- as.character(
  as.Date(stock_data$date[euro_format], format = "%d/%m/%Y")
)
# Convert all to Date
stock_data$parsed_date <- as.Date(stock_data$parsed_date)

# Clean the dataframe
stock_data <- stock_data %>%
  select(-date) %>%
  rename(date = parsed_date)

## 2. Calculate the daily Log Return for the company and for the market ##

stock_data <- stock_data %>%
  mutate(
    log_ret = log(1 + RET),
    mkt_log_ret = log(1 + sprtrn)
  )

## 3. Estimate Betas on the daily log returns ##

# Regression for each Company
```

```
beta_df <- stock_data %>%
  filter(date <= as.Date("2020-01-01")) %>%
  group_by(TICKER) %>%
  filter(!is.na(log_ret) & !is.na(mkt_log_ret)) %>%
  do(tidy(lm(log_ret ~ mkt_log_ret, data = .))) %>%
  filter(term == "mkt_log_ret") %>%
  select(TICKER, beta = estimate)

# Add CAPM Beta to the dataframe

stock_data <- left_join(stock_data, beta_df, by = "TICKER")

## 3. Daily Abnormal Log Returns First Quarter 2020 ##

stock_data <- stock_data %>%
  mutate(
    abn_log_ret = log_ret - (beta * mkt_log_ret)
  )

daily_abn_ret<- stock_data %>%
  filter(date >= as.Date("2020-01-01") & date <= as.Date("2020-03-31")) %>%
  filter(!is.na(log_ret) & !is.na(mkt_log_ret) & !is.na(beta)) %>%
  mutate(
    abn_log_ret = log_ret - (beta * mkt_log_ret)
    )

## 4. Quarterly Abnormal Log Return First Quarter 2020 ##

quarterly_abn_ret <- stock_data %>%
  filter(date >= as.Date("2020-01-01") & date <= as.Date("2020-03-31")) %>%
  mutate(
    year = format(date, "%Y"),
    quarter = paste0("Q", lubridate::quarter(date))
  ) %>%
  group_by(TICKER, year, quarter) %>%
  summarise(
    qtr_abn_log_return = sum(abn_log_ret, na.rm = TRUE),
    .groups = "drop"
  )

# 5. Calculate the statistics on the daily returns

# When need to take the average per company for the abnormal returns and caluclate
# the volatilities for the time series

daily_abn_ret_stats <- daily_abn_ret %>%
  group_by(TICKER) %>%
  summarise(
    n_obs_daily = n(),
    hist_vol = sd(stock_data$log_ret[
      stock_data$date >= as.Date("2019-01-01") & stock_data$date <= as.Date("2020-01-01")], na.rm = TRU
    vol = sd(log_ret, na.rm = TRUE) *sqrt(4),
    idio_vol = sd(abn_log_ret, na.rm = TRUE)*sqrt(4),
```

```r
    daily_avg_abn_ret = mean(abn_log_ret, na.rm = TRUE) *100,
    .groups = "drop"
  )

quarterly_abn_ret_stats <- quarterly_abn_ret %>%
  group_by(TICKER) %>%
  summarise(
    n_obs_quarter = n(),
    quarterly_abn_ret = mean(qtr_abn_log_return, na.rm = TRUE) *100,
    .groups = "drop"
  )

stock_stats_df <- left_join(daily_abn_ret_stats, quarterly_abn_ret_stats, by = "TICKER")

## 6. Join all three data frames now

# Rename ticker column
compustat_data <- compustat_data %>%
  rename(TICKER = `Ticker Symbol`)

esg_data <- esg_data %>%
  rename(TICKER = Ticker)

# Join to one data frame
Full_data_set <- left_join(compustat_data, esg_data, by = "TICKER" )
Full_data_set <- left_join(Full_data_set,stock_stats_df , by = "TICKER" )


## 7. Calculate Corporate Finance Indicators:

# Rename for easier handling
original_names <- names(Full_data_set)
cleaned_names <- gsub("\\r\\n", "", original_names)
cleaned_names <- gsub(" ", "_", cleaned_names)
cleaned_names <- gsub("\\(", "_", cleaned_names)  # Escaped (
cleaned_names <- gsub("\\)", "", cleaned_names)   # Escaped )
names(Full_data_set) <- cleaned_names


Full_data_set <- Full_data_set %>%
  mutate(
    Market_Equity = Company_Market_Cap_USD / 1000,
    tobin_q = (`Assets_-_Total` - `Common/Ordinary_Equity_-_Total` + Market_Equity) / `Assets_-_Total` /
    Leverage = (`Debt_in_Current_Liabilities_-_Total` + `Long-Term_Debt_-_Total`) / `Assets_-_Total`,
    ROE = `Net_Income__Loss` / `Common/Ordinary_Equity_-_Total`,
    Dividend_yield = `Dividends_per_Share_-_Ex-Date_-_Fiscal` / `Price_Close_-_Annual_-_Calendar` * 100
    Size = log(1+`Sales/Turnover__Net`),
    ESG = ESG_Score_FY2018 / 100
  )

## 8. Winsorize Function the at 1% level ##

winsorize <- function(x, p = 0.01) {
  quantiles <- quantile(x, probs = c(p, 1 - p), na.rm = TRUE)
  pmax(pmin(x, quantiles[2]), quantiles[1])
}

## 9. Redo the summary table ##

summary_vars <- Full_data_set %>%
  select(
    TICKER,
    quarterly_abn_ret,
    `ESG_Score_FY2018`,
    tobin_q,
    Size,
    `Cash_and_Short-Term_Investments`,
    Leverage,
    ROE,
    Advertising_Expense,
    hist_vol,
    Dividend_yield,
    vol,
    idio_vol,
    daily_avg_abn_ret,
    n_obs_daily,
    n_obs_quarter) %>%
  mutate(across(
    c(tobin_q,
      Size,
      `Cash_and_Short-Term_Investments`,
      Leverage,
      ROE,
      Advertising_Expense),
    \(x) winsorize(x, p = 0.01)
  ))


summary_stats <- summary_vars %>%
  summarise(across(
    -c(TICKER, n_obs_daily, n_obs_quarter),
    list(
      Obs = ~ sum(!is.na(.)),
      Mean = ~ mean(., na.rm = TRUE),
      SD = ~ sd(., na.rm = TRUE),
      `25%` = ~ quantile(., 0.25, na.rm = TRUE),
      Median = ~ quantile(., 0.5, na.rm = TRUE),
      `75%` = ~ quantile(., 0.75, na.rm = TRUE)
    ),
    .names = "{.col}_{.fn}"
  )) %>%
  pivot_longer(
    everything(),
    names_to = c("Variable", ".value"),
    names_pattern = "^(.*)_(Obs|Mean|SD|25%|Median|75%)$"
  )
```

```r
summary_stats$Obs[nrow(summary_stats)] <- sum(summary_vars$n_obs_daily, na.rm= TRUE)



summary_stats <- summary_stats %>%
  mutate(across(where(is.numeric), ~ round(.x, 3)))

knitr::kable(summary_stats, caption = "Table 1: Summary Statistics")
```

Table 1: Table 1: Summary Statistics

Variable	Obs	Mean	SD	25%	Median	75%
quarterly_abn_ret	871	-11.061	34.163	-26.685	-5.303	8.780
ESG_Score_FY2018	1689	37.372	19.121	22.448	32.992	48.881
tobin_q	1686	2.833	4.355	0.617	1.302	2.903
Size	1537	6.826	2.272	5.785	7.098	8.285
Cash_and_Short-Term_Investments	1537	668.423	1769.937	59.732	161.468	433.554
Leverage	1661	0.370	0.234	0.197	0.361	0.505
ROE	1536	-0.089	0.991	-0.093	0.070	0.157
Advertising_Expense	543	187.588	549.105	4.200	20.200	106.500
hist_vol	871	0.498	0.000	0.498	0.498	0.498
Dividend_yield	1688	1.955	3.112	0.000	0.558	2.878
vol	871	0.120	0.048	0.088	0.111	0.143
idio_vol	871	0.101	0.050	0.065	0.090	0.125
daily_avg_abn_ret	54978	-0.176	0.543	-0.424	-0.084	0.139