firm_regressors

December 1, 2025

[78]:
```python
%reload_ext autoreload
%autoreload 2

from pathlib import Path
import sys

from dotenv import load_dotenv

# climb up until we hit the repo root, then add src
here = Path.cwd().resolve()
while here.name != "over-intra-news" and here.parent != here:
    here = here.parent

src_path = here / "src"
if str(src_path) not in sys.path:
    sys.path.insert(0, str(src_path))

load_dotenv()
```

[78]:
```
True
```

1 Firm-level regression features

This notebook builds the **firm-level regression panel** used in the Glasserman-style news regressions. The goal is to take the S&P-500 universe produced by the entity resolution pipeline, pull **fundamentals** and **daily prices** from EODHD for those firms, and construct a panel of returns and control variables (size, value, volatility, momentum). These controls are used to show that any incremental predictability from LDA-based news signals is **not** simply a relabeling of standard risk factors or simple firm characteristics.

1.1 Notebook roadmap

The notebook is split into four stages:

1. **Universe construction and fundamentals ingestion**
   - Start from the `ticker_cik_mapping` universe of S&P-500 episodes that actually appear in the news corpus.

- Drop a small set of problematic ticker–window episodes based on `fundamentals_manual_adjudication` and `FIRMS_TO_DROP`.
- For the remaining episodes, call the EODHD **fundamentals** endpoint to obtain quarterly balance-sheet snapshots and basic share-count data.

2. **Daily price panel and return decomposition**
   - For the same universe, call the EODHD **/eod** endpoint to obtain daily OHLCV data over each ticker's validity window.
   - Decompose prices into overnight, intraday, and close-to-close log returns.
3. **Feature construction (size, value, volatility, momentum)**
   - Align quarterly fundamentals to trading days via a backward merge-as-of with a 90-day tolerance.
   - Compute market capitalization, log size, book-to-market, realized volatility, and 1- and 12-month momentum, all with strict no-look-ahead.
4. **Persistence to Postgres**
   - Load the final `features_df` into the `equity_regression_panel` table, keyed by `(cik, eodhd_symbol, trading_day)`, to serve as the regression join point against LDA topic exposures and other signals.

1.2 1. Fundamentals ingestion and firm-level controls

1.2.1 1.1 Vendor fundamentals feed

Stage 1 focuses on **quarterly fundamentals** from EODHD and on making those snapshots usable as regression controls:

- For each active ticker–CIK–validity-window episode in `ticker_cik_mapping` that survives the drop list, we construct an EODHD symbol (e.g. `AAPL.US`) using `TICKER_ALIAS_MAPPING` for historical renames.

- We query the EODHD **fundamentals** endpoint with

  ```
  filter=Financials::Balance_Sheet::quarterly
  ```

  so that the JSON payload is limited to **quarterly balance-sheet data** and a small set of share-count fields.

- From each quarterly record we extract:

  - `filing_date` – the normalized filing date of the report.
  - `totalAssets` and `totalLiab` – used to construct **book equity** as `totalAssets - totalLiab` when both are present.
  - `commonStockSharesOutstanding` – used as **shares outstanding**.

- We keep only filings whose `filing_date` lies inside the ticker's validity window `[start, end)`. Filings with a missing `filing_date` are dropped with a logged warning.

These per-filing rows form the `fundamentals_df` used later when we align fundamentals to trading days and build the controls:

- **Market capitalization** is defined as `adjusted_close × shares_outstanding`.
- **Log size** is `log(market_cap)`.

- **Book-to-market** is `book_equity / market_cap` when both inputs are available.
- All of these are evaluated **as of** each trading day using the latest filing at or before that date, with a 90-day window, to avoid leaking future fundamentals into the past.

1.2.2 1.2 Dropped and aliased ticker–window episodes

Not every ticker–CIK episode is usable for fundamentals-based controls. Before we call EODHD, the notebook applies the curated decisions recorded in `fundamentals_manual_adjudication` and encoded in `FIRMS_TO_DROP` and `TICKER_ALIAS_MAPPING`. This ensures that the regression panel only includes episodes where we can defend the fundamentals data on an **as-of** basis.

The main patterns are:

- **Missing or unusable fundamentals (drop_ticker).**

  Some ticker–window–CIK episodes have no usable quarterly fundamentals: the EODHD endpoint either returns no data or returns payloads with missing `filing_date`, making point-in-time validation impossible. These episodes are recorded in `fundamentals_manual_adjudication` with `action = 'drop_ticker'` and the ticker is added to `FIRMS_TO_DROP`, so the notebook removes them from `active_firms` before any API calls are made.

  Examples:

  - **YHOO — no usable filing dates.**
    For the `[2016-08-01, 2017-06-19)` episode, the EODHD fundamentals endpoint returns balance-sheet data but without `filing_date`. Because we cannot align these records to trading days or confirm that they fall inside the validity window, the ticker is dropped from the regression universe for this horizon (`action = 'drop_ticker'` with a rationale documenting the missing filing dates).
  - **CA — no fundamentals returned.**
    For ticker `CA` in `[2016-08-01, 2018-11-06)`, the EODHD fundamentals endpoint returns no usable data at all. Rather than silently carrying a firm with structurally missing controls, the episode is explicitly adjudicated as `drop_ticker` and added to `FIRMS_TO_DROP`.

- **Ticker-level aliases where the issuer is unchanged (alias_rewrite).**

  In other cases the underlying issuer is well-identified, but the exchange ticker has changed while the CIK remains the same. These are treated as **aliases**, not distinct firms. The adjudication table records them with `action = 'alias_rewrite'`, and the ticker-normalization logic rewrites the historical symbol to a canonical one via `TICKER_ALIAS_MAPPING` before hitting EODHD.

  Example:

  - **LB → BBWI — symbol change without issuer change.**
    For `LB` in `[2016-08-01, 2021-08-03)`, EDGAR and contemporaneous filings show that the economic issuer continues as Bath & Body Works after the separation from Victoria's Secret. Fundamentals are more reliably served under the `BBWI` symbol, so the adjudication row records an `alias_rewrite` with a supporting 8-K, and `TICKER_ALIAS_MAPPING`

maps LB → BBWI before querying EODHD. The regression panel therefore carries a continuous BBWI.US time series and does not double-count the same CIK under two tickers.

By enforcing these drop and alias decisions upstream, Stage 1 guarantees that every firm–ticker episode that survives into `fundamentals_df` has:

- a defensible issuer identity (via the ticker→CIK mapping and adjudication),
- a usable stream of quarterly fundamentals with filing dates, and
- a well-defined canonical vendor symbol for EODHD.

That, in turn, makes the later regression results easier to interpret: when we say "book-to-market" or "size" for a firm-day, we know exactly **which** issuer, CIK, and ticker that control is referring to.

[79]:
```
import pandas as pd
from notebooks_utils.data_notebooks_utils.firm_regressors_utils.
 ↪firm_regressors_config import FIRMS_TO_DROP
from notebooks_utils.data_notebooks_utils.firm_regressors_utils.
 ↪firm_regressors_utils import extract_active_firms
active_firms: pd.DataFrame = extract_active_firms()
firms_to_drop_mask: pd.Series =active_firms['ticker'].isin(FIRMS_TO_DROP)
active_firms = active_firms.loc[~firms_to_drop_mask].reset_index(drop=True)
```

[132]:
```
import os
from infra.logging.infra_logger import InfraLogger, initialize_logger
from notebooks_utils.data_notebooks_utils.firm_regressors_utils.
 ↪firm_regressors_utils import build_fundamentals_df

api_key = os.getenv("EODHD")
logger: InfraLogger = initialize_logger("firm_regressors_notebook")
fundamentals_df: pd.DataFrame = build_fundamentals_df(
    active_firms=active_firms,
    api_key=api_key,
    logger=logger
) # This is a No-OP unless real_run = True
```

1.3 2. Price data and return decomposition

This stage builds the daily price panel and decomposes returns into the "overnight / intraday / close-to-close" pieces that will serve as the dependent variables in the Glasserman-style regressions, while also constructing price-based controls such as realized volatility and momentum. All price data come from EODHD's `/eod` endpoint and are keyed by the same ticker–CIK episodes defined in Stage 1.

1.3.1 2.1 Ingesting daily OHLCV from EODHD

For each `(ticker, validity_window, cik)` triple in `active_firms`:

- The raw exchange ticker is first normalized to an EODHD symbol: either `TICKER.US` or an alias such as LB → BBWI.US from `TICKER_ALIAS_MAPPING`.

- We query `https://eodhd.com/api/eod/{symbol}?period=d&fmt=json` and parse the JSON payload into a list of daily OHLCV records.
- Each record is retained only if its `date` lies inside the half-open validity window `[start, end)` for that ticker–CIK episode. This guarantees that price history respects the same point-in-time mapping used in the news and fundamentals layers.
- For every accepted trading day we materialize:
  - `open`, `high`, `low`, `close` — unadjusted daily prices.
  - `adjusted_close` — close price adjusted for splits and dividends.
  - `volume` — daily share volume (non-negative).
  - `trading_day` — normalized trading date.
  - `cik` — the firm identifier propagated from `active_firms`.

Some ticker–CIK episodes are explicitly removed from the research universe because the vendor cannot supply a usable price history. These decisions are recorded in `fundamentals_manual_adjudication` and reflected in `FIRMS_TO_DROP`. For example:

- `STI` (`[2016-08-01, 2019-12-09)`, CIK `0000750556`) is tagged `drop_ticker` with rationale:
  - No price data available from EODHD; cannot perform regression.

  and an associated `/eod/STI.US` URL showing the empty payload.

Episodes like this are dropped before any calls to `build_returns_df`, so they never enter the price panel or downstream regressions.

1.3.2 2.2 Return decomposition and price-based controls

Once the daily OHLCV panel is in place, we align it with the fundamentals from Stage 1 and compute return- and volatility-based features in `calculate_features`:

- Let O_t be the open price and A_t the adjusted close on trading day t; A_{t-1} is the prior adjusted close. All returns are **natural-log returns**:
  - **Overnight return**

$$r_t^{\text{overnight}} = \log\left(\frac{O_t}{A_{t-1}}\right) \tag{1}$$

    capturing the move from yesterday's adjusted close to today's open.
  - **Intraday return**

$$r_t^{\text{intraday}} = \log\left(\frac{A_t}{O_t}\right) \tag{2}$$

    capturing the open-to-close move.
  - **Close-to-close return**

$$r_t^{\text{close-to-close}} = \log\left(\frac{A_t}{A_{t-1}}\right) \tag{3}$$

    the total one-day return used for volatility and momentum.
- Using $r_t^{\text{close-to-close}}$, we build annualized realized volatilities with a 252-trading-day convention:
  - **21-day realized vol**

$$\sigma_t^{21} = \sqrt{252}\ \mathrm{stdev}(r_{t-21}, \dots, r_{t-1}) \tag{4}$$

- **252-day realized vol**

$$\sigma_t^{252} = \sqrt{252}\,\mathrm{stdev}(r_{t-252}, \dots, r_{t-1}) \tag{5}$$

  Both series are shifted by one day so that only information strictly prior to t is used (no look-ahead).

- We also construct **momentum controls** from adjusted prices:
  - **1-month reversal / short-term momentum**: the percentage return over the last 21 trading days, shifted so that the signal at t only depends on prices up to $t-1$.
  - **12-month momentum**: the percentage return from $t-12\text{m}$ to $t-1\text{m}$, again shifted to avoid look-ahead.
- Finally, we recompute the "size" and "value" style controls:
  - **Market capitalization**

$$\mathrm{mktcap}_t = A_t \times \mathrm{shares\_outstanding}_t \tag{6}$$

  - **Log market cap**: $\log(\mathrm{mktcap}_t)$, the standard size regressor.
  - **Book-to-market**: `book_equity / market_cap`, matching the value control in the original Glasserman specification whenever fundamentals are available.

Early days with insufficient history for a given window naturally carry `NaN` in the corresponding volatility or momentum feature. The output of this stage is a `features_df` DataFrame with one row per (cik, ticker, trading_day) observation and a full set of price levels, return decompositions, and price-based controls ready to be loaded into `equity_regression_panel` in Stage 3.

[90]:
```python
from notebooks_utils.data_notebooks_utils.firm_regressors_utils.
 ↪firm_regressors_utils import align_fundamentals_with_returns,␣
 ↪build_returns_df, calculate_features

returns_df: pd.DataFrame = build_returns_df(
    active_firms=active_firms,
    api_key=api_key,
    logger=logger,
    real_run=True
) # This is a No-OP unless real_run = True
features_df: pd.DataFrame = align_fundamentals_with_returns(fundamentals_df,␣
 ↪returns_df)
features_df = calculate_features(features_df)
```

1.4 3. Persistence to Postgres

The final stage takes the in-memory `features_df` panel and materializes it into the `equity_regression_panel` table in Postgres.

Each row in `equity_regression_panel` represents a single firm–ticker–day observation, keyed by `(cik, eodhd_symbol, trading_day)`, with price levels, return decompositions, size/value controls, and provenance metadata.

1.4.1 3.1 `equity_regression_panel` schema

The table DDL is structured to mirror the columns in `features_df` and to enforce the as-of semantics used throughout the pipeline:

- **Identifiers**
  - `cik` – immutable 10-digit firm identifier, foreign-keyed into `security_master (cik)` with `ON DELETE CASCADE`.
  - `trading_day` – NYSE trading date for the observation.
  - `eodhd_symbol` – vendor symbol used to query EODHD (e.g. `AAPL.US`), derived from the notebook's `ticker` column after applying `TICKER_ALIAS_MAPPING`.
- **Raw price levels**
  - `open_price`, `high_price`, `low_price`, `close_price` – unadjusted daily OHLC prices from the `/eod` endpoint.
  - `adjusted_close_price` – close price adjusted for splits and dividends.
  - `volume` – daily share volume (non-negative).
- **Return decomposition**
  - `overnight_log_return` –

    $$\log\left(\frac{O_t}{A_{t-1}}\right) \tag{7}$$

    where (O_t) is the open and (A_{t-1}) is the prior adjusted close.
  - `intraday_log_return` –

    $$\log\left(\frac{A_t}{O_t}\right) \tag{8}$$

    open-to-close log return.
  - `close_to_close_log_return` –

    $$\log\left(\frac{A_t}{A_{t-1}}\right) \tag{9}$$

    total one-day log return used for volatility and momentum.
- **Price-based controls**
  - `realized_vol_21d`, `realized_vol_252d` – annualized realized volatilities based on rolling 21- and 252-day windows of `close_to_close_log_return`, multiplied by $\sqrt{252}$ and shifted by one day so that only information strictly prior to `trading_day` is used.
  - `momentum_1m` – 1-month reversal / short-term momentum: percentage return over the last 21 trading days, shifted to avoid look-ahead.
  - `momentum_12m` – 12-month momentum: cumulative percentage return from $t - 12\text{m}$ to $t - 1\text{m}$, again shifted by one month.
- **Fundamentals-based controls**
  - `shares_outstanding` – share count carried forward from the most recent quarterly fundamentals snapshot.
  - `market_cap` –

    $$\text{adjusted\_close}_t \times \text{shares\_outstanding}_t \tag{10}$$

  - `log_market_cap` – natural log of `market_cap` (standard "size" regressor).

  - `book_to_market` – ratio of book equity to market cap, using the `book_equity` constructed in Stage 1 and the contemporaneous `market_cap`.
  - `filing_date` – filing date of the quarterly report that supplied the fundamentals for this observation.
- **Provenance and constraints**
  - `created_at` – timestamp when the row was first written.
  - **Primary key:** `(cik, eodhd_symbol, trading_day)`. This allows a CIK to appear under multiple vendor symbols on a given day (e.g. share classes) while still treating each (firm, symbol, day) as a distinct observation.
  - **Unique index** on `(eodhd_symbol, trading_day)` for vendor-side sanity checking.
  - **Check constraints:**
    * `volume >= 0`.
    * all price columns strictly positive.
    * whenever any of `shares_outstanding`, `market_cap`, `log_market_cap`, or `book_to_market` are non-null, `filing_date` must be present and `filing_date <= trading_day`, enforcing the as-of fundamentals discipline.

By construction, every column in `features_df` has a direct target in this schema: `ticker` is loaded as `eodhd_symbol` and the OHLCV and return columns map one-for-one, and the fundamentals-based fields map into `shares_outstanding`, `market_cap`, `log_market_cap`, `book_to_market`, and `filing_date`.

1.4.2 3.2 Loading pipeline and idempotency

The notebook wires this schema to the in-memory panel via the `load_equity_regression_panel` helper:

- `create_equity_regression_panel_row_generator(features_df)` iterates over `features_df` and yields tuples in the exact column order expected by the INSERT statement:

  - `cik`, `trading_day`, `ticker` (as `eodhd_symbol`), raw OHLCV prices, `adjusted_close`, the three log-return decompositions, realized volatilities, momentum signals, `shares_outstanding`, `market_cap`, `log_market_cap`, `book_to_market`, and `filing_date` (with NaNs converted to `NULL`).

- `generate_equity_regression_panel_query()` returns a parameterized INSERT of the form:

  ```sql
  INSERT INTO equity_regression_panel ( ... )
  VALUES %s
  ON CONFLICT (cik, eodhd_symbol, trading_day) DO NOTHING;
  ```

so that repeated runs of the notebook are **idempotent**: any row whose `(cik, eodhd_symbol, trading_day)` triple is already present is silently skipped rather than duplicated.

- `load_equity_regression_panel(features_df, real_run=True)` opens a database connection via `connect_to_db`, streams the row tuples into `load_into_table`, and uses batched `execute_values` calls to insert the data efficiently.

[ ]:
```
from notebooks_utils.data_notebooks_utils.firm_regressors_utils.
 ↪load_firm_regressors import load_equity_regression_panel
load_equity_regression_panel(features_df) # This is a No-OP unless real_run =␣
 ↪True
```