Principal Asset Management

Project Lab – The University of Chicago

Equity Factor Construction and Analysis

Leah Li
Matheus Raka Pradnyatama
Nicolas Donato

Deliverables

1. Factor Construction
2. Back-Testing Pipeline
   a) Formula Parser in R
   b) Point-in-Time Data Collector in Excel
3. Sensitivity Analysis on Factor Measures

Factor Construction

Factor Construction

- Using the paper titled “**Global Factor Data Documentation**” written by *Theis Ingerslev Jensen, Bryan Kelly, Lasse Heje Pedersen*

- The constructed factors are mapped into relevant Bloomberg fields

- This include:
  - Accounting variables: Gross Profit, Operating Income, Long-Term Debt, etc.
  - Market variables: Price, Return, Cash Dividend, Trading Volume, etc.

- Lag variables were also constructed manually for Momentum Factor, modeled with prices across monthly durations

- The constructed formula mappings are compiled into an excel file titled: “**Formulas**”
  - A previous version: “**Formula Mappings - Paper to Bloomberg FLDS**” was sent to Rajat Kathuria on April 16, 2025

Factor Construction

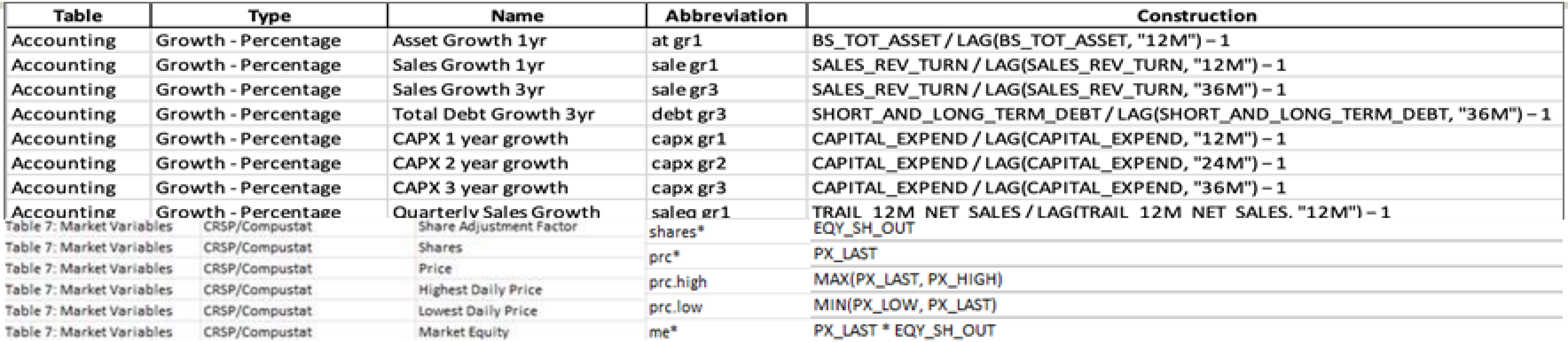


Table	Type	Name	Abbreviation	Construction
Accounting	Growth - Percentage	Asset Growth 1yr	at gr1	BS_TOT_ASSET / LAG(BS_TOT_ASSET, "12M") – 1
Accounting	Growth - Percentage	Sales Growth 1yr	sale gr1	SALES_REV_TURN / LAG(SALES_REV_TURN, "12M") – 1
Accounting	Growth - Percentage	Sales Growth 3yr	sale gr3	SALES_REV_TURN / LAG(SALES_REV_TURN, "36M") – 1
Accounting	Growth - Percentage	Total Debt Growth 3yr	debt gr3	SHORT_AND_LONG_TERM_DEBT / LAG(SHORT_AND_LONG_TERM_DEBT, "36M") – 1
Accounting	Growth - Percentage	CAPX 1 year growth	capx gr1	CAPITAL_EXPEND / LAG(CAPITAL_EXPEND, "12M") – 1
Accounting	Growth - Percentage	CAPX 2 year growth	capx gr2	CAPITAL_EXPEND / LAG(CAPITAL_EXPEND, "24M") – 1
Accounting	Growth - Percentage	CAPX 3 year growth	capx gr3	CAPITAL_EXPEND / LAG(CAPITAL_EXPEND, "36M") – 1
Accounting	Growth - Percentage	Quarterly Sales Growth	saleq gr1	TRAIL_12M_NET_SALES / LAG(TRAIL_12M_NET_SALES, "12M") – 1
Table 7: Market Variables	CRSP/Compustat	Share Adjustment Factor	shares*	EQY_SH_OUT
Table 7: Market Variables	CRSP/Compustat	Shares	prc*	PX_LAST
Table 7: Market Variables	CRSP/Compustat	Price	prc.high	MAX(PX_LAST, PX_HIGH)
Table 7: Market Variables	CRSP/Compustat	Highest Daily Price	prc.low	MIN(PX_LOW, PX_LAST)
Table 7: Market Variables	CRSP/Compustat	Lowest Daily Price	me*	PX_LAST * EQY_SH_OUT
Table 7: Market Variables	CRSP/Compustat	Market Equity		

Source: Formulas.xlsx

Table/Type - These columns determine the category of the factor. The factor can be from the accounting or market table, and there are various types within each.

Name/Abbreviation - These columns are the same factor names referenced in the Global Factor Data paper Documentation, Tables 5 - 8.

Construction - This column contains the formula to construct the factor using Bloomberg field names. This is what's processed in the R parser script to calculate the factors.

Back-Testing Pipeline

Formula Parser in R

Formula Parser R - Load Data

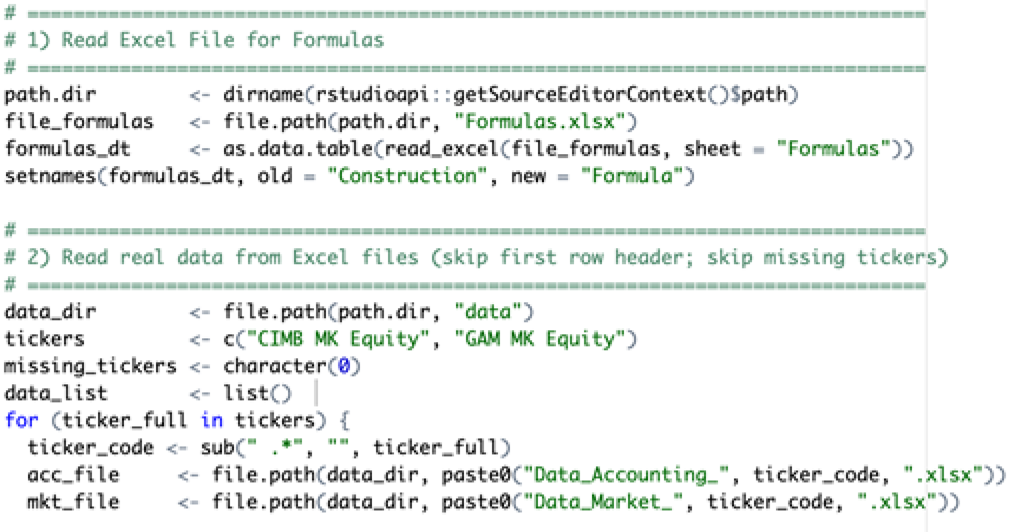
Workflow Description

Load Formula Definitions

- Read constructed formulas from Formulas.xlsx and recognizes variables

Ingest & Clean Raw Data

- Iterate over each ticker, reading its “Accounting” and “Market” sheets
- Standardize dates, strip non-numeric characters, lag accounting metrics by 4 months
- Merge all tickers’ data into one long format ready for analysis



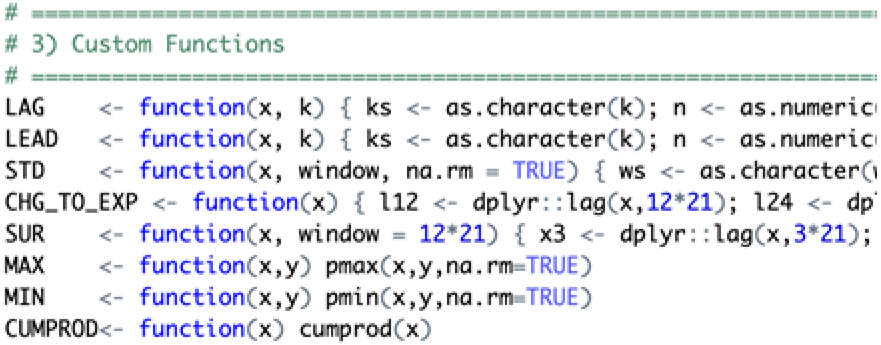
```
# ======================================================================
# 1) Read Excel File for Formulas
# ======================================================================
path.dir        <- dirname(rstudioapi::getSourceEditorContext()$path)
file_formulas   <- file.path(path.dir, "Formulas.xlsx")
formulas_dt     <- as.data.table(read_excel(file_formulas, sheet = "Formulas"))
setnames(formulas_dt, old = "Construction", new = "Formula")

# ======================================================================
# 2) Read real data from Excel files (skip first row header; skip missing tickers)
# ======================================================================
data_dir        <- file.path(path.dir, "data")
tickers         <- c("CIMB MK Equity", "GAM MK Equity")
missing_tickers <- character(0)
data_list       <- list()
for (ticker_full in tickers) {
  ticker_code <- sub(" .*", "", ticker_full)
  acc_file     <- file.path(data_dir, paste0("Data_Accounting_", ticker_code, ".xlsx"))
  mkt_file     <- file.path(data_dir, paste0("Data_Market_", ticker_code, ".xlsx"))
```

Source: formula_parser.r

Formula Parser R - Factor Computation

Formula Engine

- Balances syntax, cleans up formulas, and parses text into R expressions
- Utilizes helper functions built to accurately parse constructed formulas
- For each formula and ticker, builds a per-date data frame of inputs, then evaluates in a custom environment
- Automatically skips any formulas that error or yield no results

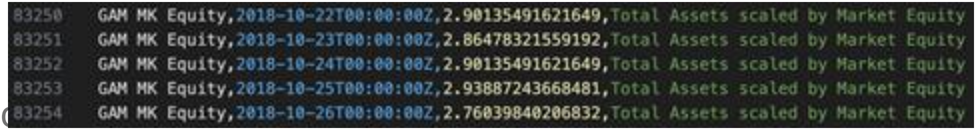


```
# ===========================================================================
# 3) Custom Functions
# ===========================================================================
LAG      <- function(x, k) { ks <- as.character(k); n <- as.numeric
LEAD     <- function(x, k) { ks <- as.character(k); n <- as.numeric
STD      <- function(x, window, na.rm = TRUE) { ws <- as.character(
CHG_TO_EXP <- function(x) { l12 <- dplyr::lag(x,12*21); l24 <- dp
SUR      <- function(x, window = 12*21) { x3 <- dplyr::lag(x,3*21);
MAX      <- function(x,y) pmax(x,y,na.rm=TRUE)
MIN      <- function(x,y) pmin(x,y,na.rm=TRUE)
CUMPROD<- function(x) cumprod(x)
```

Source: formula_parser.r

Export Step

- Run all parsed formulas and output combined fac results to **computed_factors.csv**



```
GAM MK Equity,2018-10-22T00:00:00Z,2.90135491621649,Total Assets scaled by Market Equity
GAM MK Equity,2018-10-23T00:00:00Z,2.86478321559192,Total Assets scaled by Market Equity
GAM MK Equity,2018-10-24T00:00:00Z,2.90135491621649,Total Assets scaled by Market Equity
GAM MK Equity,2018-10-25T00:00:00Z,2.93887243668481,Total Assets scaled by Market Equity
GAM MK Equity,2018-10-26T00:00:00Z,2.76039840206832,Total Assets scaled by Market Equity
```

Source: computed_factors.csv

Formula Parser R - Factor Computation

Feature	Functionality	Usefulness	Adaptability
Time-Aware Functions (LAG, LEAD, STD)	Recognizes flexible time units like "3M", "1Y" and converts to trading days	Handles financial data frequency naturally	Adapts to various periodicity in factor design
Nested functions	Supports formulas with compound functions STD (EXP (LOG ())) including custom functions	Enables mathematically complex and compact expressions	No need to manually split expressions
Formula Chaining	Allows one formula to reference results from previous ones	Enables complex multi-step modeling	Easily supports derived formulas without recomputation
Ticker-wise evaluation	Processes each equity's data independently	Maintains clean separation across securities and records error	Scales well to different groups of tickers of interest

Back-Testing Pipeline

Point-in-Time Data Collector in Excel

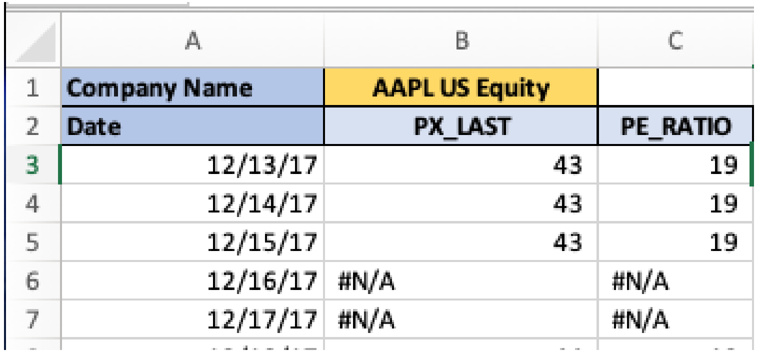
Point-in-Time Data Collector – Description

- We collected Point-in-Time data using Excel

- We used the BQL.QUERY that can be enabled in Excel

- Syntax:
  =@BQL.QUERY("get(" & B$2 & ") for('" & B1 & "') with(DATES=" & TEXT($A3,"yyyy-mm-dd") & ")")

- Users need only to change the ticker name for the company (cell B1) to the company they are interested in



	A	B	C
1	Company Name	AAPL US Equity	
2	Date	PX_LAST	PE_RATIO
3	12/13/17	43	19
4	12/14/17	43	19
5	12/15/17	43	19
6	12/16/17	#N/A	#N/A
7	12/17/17	#N/A	#N/A

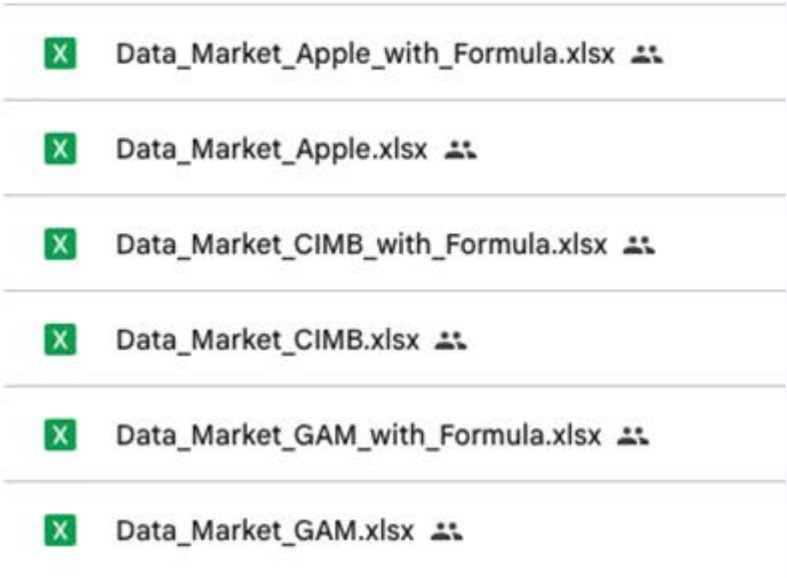
Point-in-Time Data Collector – Submitted Files

- We collected 2-years worth of point-in-time data for 3 companies: Apple US Equity (US Company), GAM MK Equity (Malaysian company), CIMB MK Equity (Malaysian company)

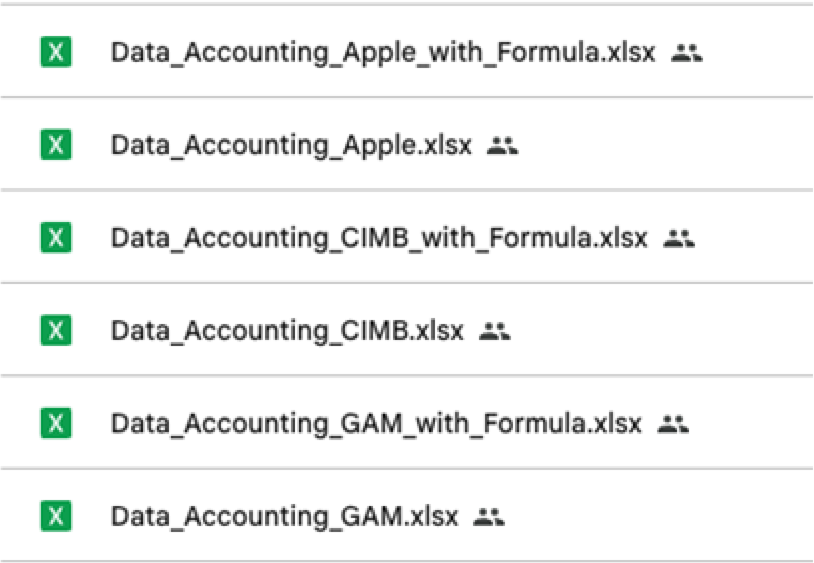
- The data collector is split between accounting and market variables to reduce the amount of data saved in one file

- For each company and variable type, there are two excel files:
  - The file containing the formula can be run on a computer connected to an active Bloomberg Terminal to observe how the BQL.QUERY formula functions in practice and to retrieve new data
  - The file that doesn't contain the formulas contains only the data extracted in the past

Point-in-Time Data Collector – Submitted Files

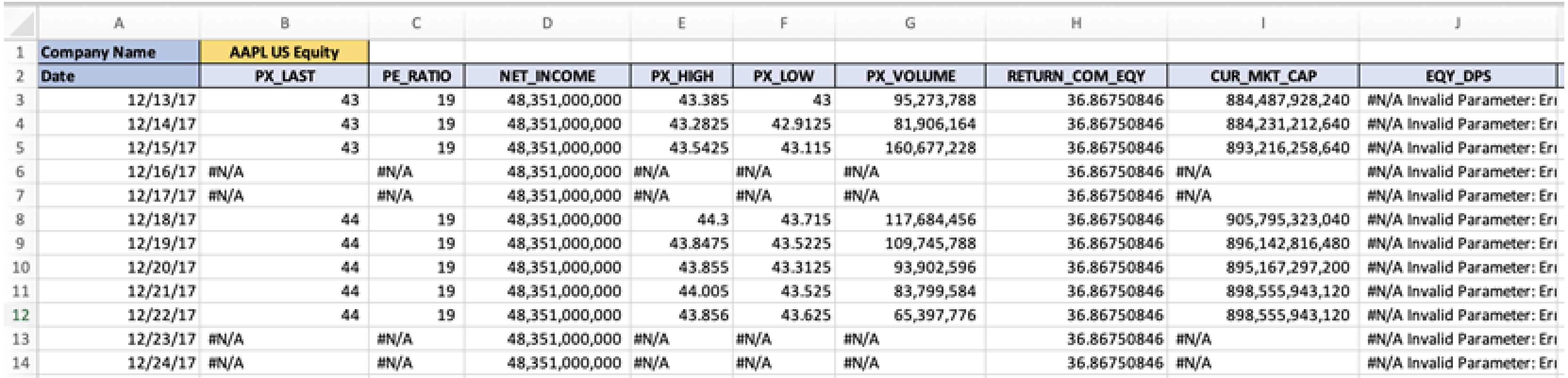


- Data_Market_Apple_with_Formula.xlsx
- Data_Market_Apple.xlsx
- Data_Market_CIMB_with_Formula.xlsx
- Data_Market_CIMB.xlsx
- Data_Market_GAM_with_Formula.xlsx
- Data_Market_GAM.xlsx



- Data_Accounting_Apple_with_Formula.xlsx
- Data_Accounting_Apple.xlsx
- Data_Accounting_CIMB_with_Formula.xlsx
- Data_Accounting_CIMB.xlsx
- Data_Accounting_GAM_with_Formula.xlsx
- Data_Accounting_GAM.xlsx

Point-in-Time Data Collector – Submitted Files



	A	B	C	D	E	F	G	H	I	J
1	Company Name	AAPL US Equity								
2	Date	PX_LAST	PE_RATIO	NET_INCOME	PX_HIGH	PX_LOW	PX_VOLUME	RETURN_COM_EQY	CUR_MKT_CAP	EQY_DPS
3	12/13/17	43	19	48,351,000,000	43.385	43	95,273,788	36.86750846	884,487,928,240	#N/A Invalid Parameter: Err
4	12/14/17	43	19	48,351,000,000	43.2825	42.9125	81,906,164	36.86750846	884,231,212,640	#N/A Invalid Parameter: Err
5	12/15/17	43	19	48,351,000,000	43.5425	43.115	160,677,228	36.86750846	893,216,258,640	#N/A Invalid Parameter: Err
6	12/16/17	#N/A	#N/A	48,351,000,000	#N/A	#N/A	#N/A	36.86750846	#N/A	#N/A Invalid Parameter: Err
7	12/17/17	#N/A	#N/A	48,351,000,000	#N/A	#N/A	#N/A	36.86750846	#N/A	#N/A Invalid Parameter: Err
8	12/18/17	44	19	48,351,000,000	44.3	43.715	117,684,456	36.86750846	905,795,323,040	#N/A Invalid Parameter: Err
9	12/19/17	44	19	48,351,000,000	43.8475	43.5225	109,745,788	36.86750846	896,142,816,480	#N/A Invalid Parameter: Err
10	12/20/17	44	19	48,351,000,000	43.855	43.3125	93,902,596	36.86750846	895,167,297,200	#N/A Invalid Parameter: Err
11	12/21/17	44	19	48,351,000,000	44.005	43.525	83,799,584	36.86750846	898,555,943,120	#N/A Invalid Parameter: Err
12	12/22/17	44	19	48,351,000,000	43.856	43.625	65,397,776	36.86750846	898,555,943,120	#N/A Invalid Parameter: Err
13	12/23/17	#N/A	#N/A	48,351,000,000	#N/A	#N/A	#N/A	36.86750846	#N/A	#N/A Invalid Parameter: Err
14	12/24/17	#N/A	#N/A	48,351,000,000	#N/A	#N/A	#N/A	36.86750846	#N/A	#N/A Invalid Parameter: Err

- #N/A:
  - Data is not available during those dates (e.g., not a business day)

- #N/A Invalid Parameter: Error encountered while validating *field_name*
  - The company does not have any data on that specific variable

Sensitivity Analysis

Sensitivity Analysis

Data Inputs

- Computed Factors: loads a pre-computed CSV of factor values (one row per ticker-date-factor).
- Market Prices: reads raw Excel price files for each ticker, cleans them, and computes forward returns.

Parameter Controls

- Date Range: restrict the analysis window
- Ticker: pick which instrument to study
- Factor: select which pre-computed metric (e.g. leverage, valuation)
- Horizon: choose the look-ahead period (1 day, week, month, quarter, year)

Forward Return Calculation

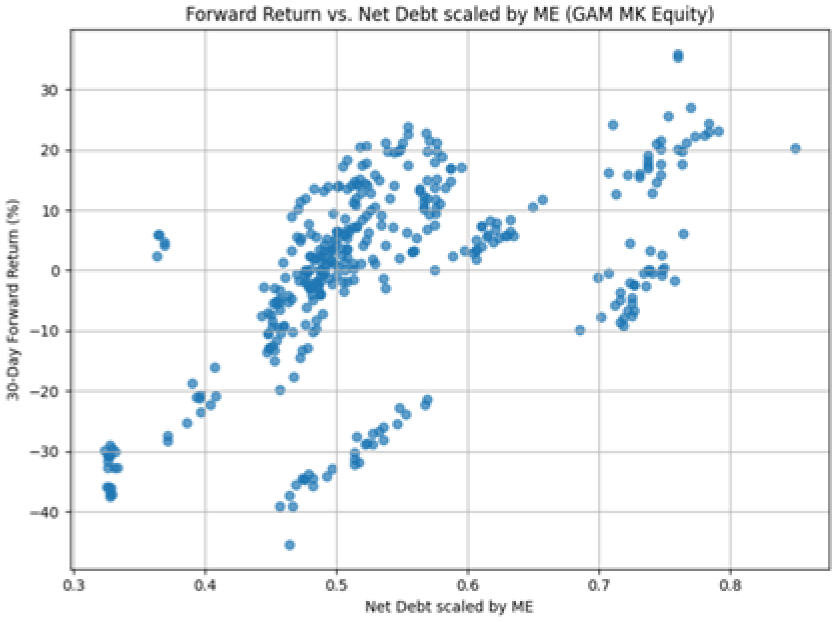
- Forward return measures how much the price increases or decreases after a chosen number of trading days. It reflects the future price movement from a given starting point.

Visualization

- The app creates a scatter plot showing the relationship between the selected factor and the forward return. Each point represents a historical observation for the selected ticker and date range.

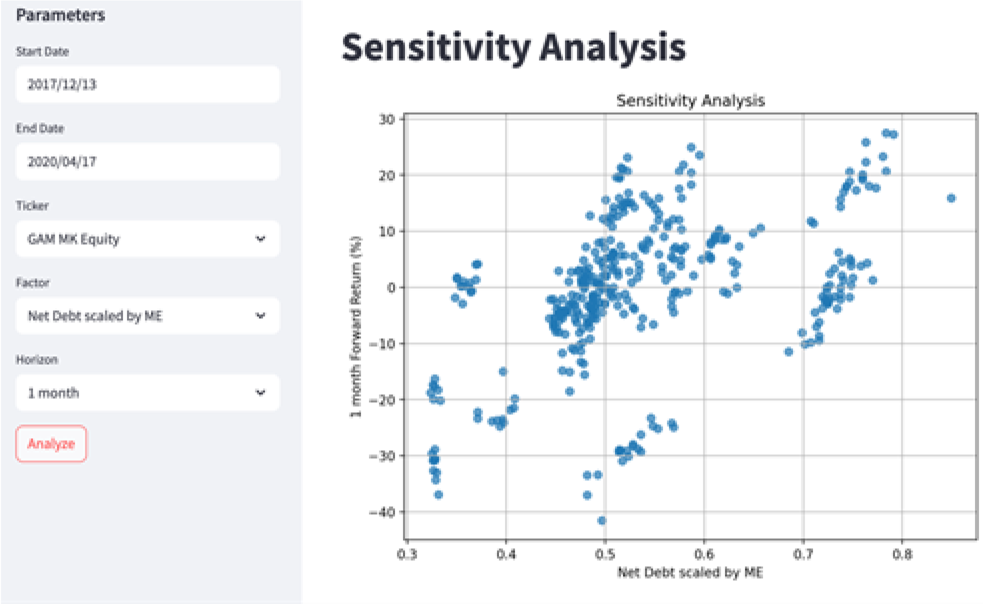
Sensitivity Analysis

Jupyter Notebook



Source: sensitivity_analysis_notebook.ipynb

Python Application UI



Parameters

Start Date: 2017/12/13

End Date: 2020/04/17

Ticker: GAM MK Equity

Factor: Net Debt scaled by ME

Horizon: 1 month

Source: sensitivity_analysis_UI.py

Sensitivity Analysis

Note:

- Sensitivity analysis is available in both the **sensitivity_analysis_notebook.ipynb**, for a python notebook version, and **sensitivity_analysis_UI.py**, for a clean browser-UI program.

- To run the **sensitivity_analysis_UI.py** version
  - Make sure streamlit is installed (“pip install streamlit”)
  - Run the program via terminal using “streamlit run sensitivity_analysis_UI.py”
  - The UI will then open in your browser

- Ensure the following data files are in the directory of the sensitivity analysis program:
  - computed_factors.csv - For all the computed factors calculated from formula_parser.r
  - Data_Market_{ticker}.xlsx - Which contains Bloomberg PX_LAST data for the specified ticker

Challenges

Challenges – Data Collection

- There are two alternative methods for data collection that may prove to be more effective

- **EE (Earnings Estimates)** and **Earnings Summary (ERN)** functions in BBG
  - We needed to learn this with a BBG representative
  - The Live Help feature is disabled in the Finmath student BBG subscription
  - We can only communicate with BBG with a one-business-day delay

- **Bloomberg BLP API in Python**
  - Succeeded in implementing blp/refdata using Python script, but blp/bql is disabled in the Finmath student BBG subscription on campus
  - We could only collect historical instead of point-in-time data as needed

- **Conclusion:** We used the **BQL.QUERY** directly in **Excel**

Challenges – Data

- There is often missing data from specific fields for particular tickers
- Accounting variables and market variables are recorded at different cadences (daily vs quarterly for example)
- Further consideration is needed for how to deal with accounting variables in the sensitivity analysis, perhaps using only quarterly data for accounting factors
- Simplification assumptions were made for market factors that did not have direct Bloomberg fields. Replication of those factors were done using basic variables.
- More complex market factors can be added to list of constructed variables with further attention to different time lags

Thank you