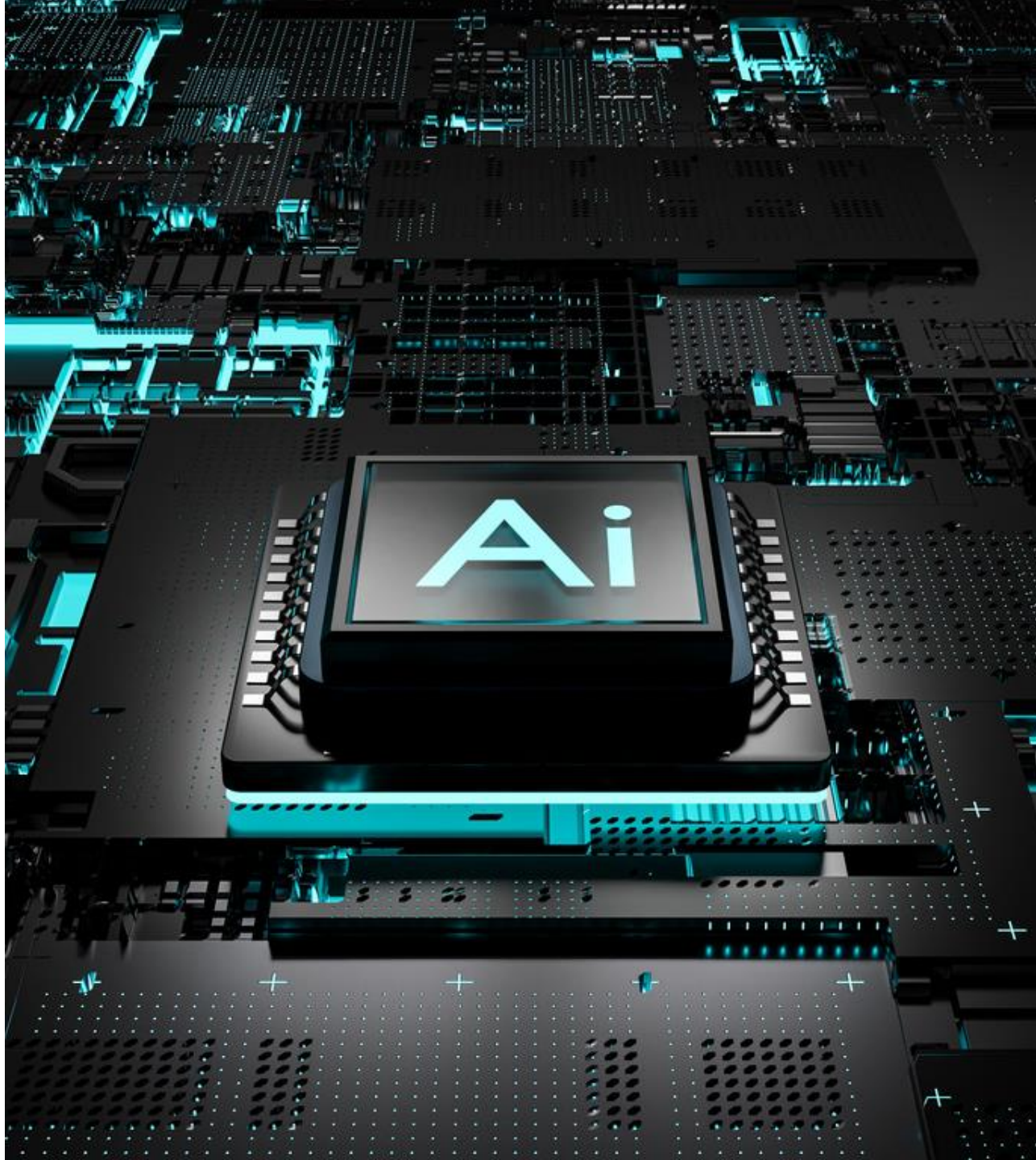
FULLERTON
FUND MANAGEMENT

Fullerton Data Science meetup

Practical applications of Data Science in investment management

Chen Chao Jen, Managing Director, Head of Augmented Intelligence
Lim Shi Hui, Director, Augmented Intelligence
Chng Yan Rong, Manager, Augmented Intelligence

Fullerton Fund Management



Content

1

Introduction of Fullerton Fund Management

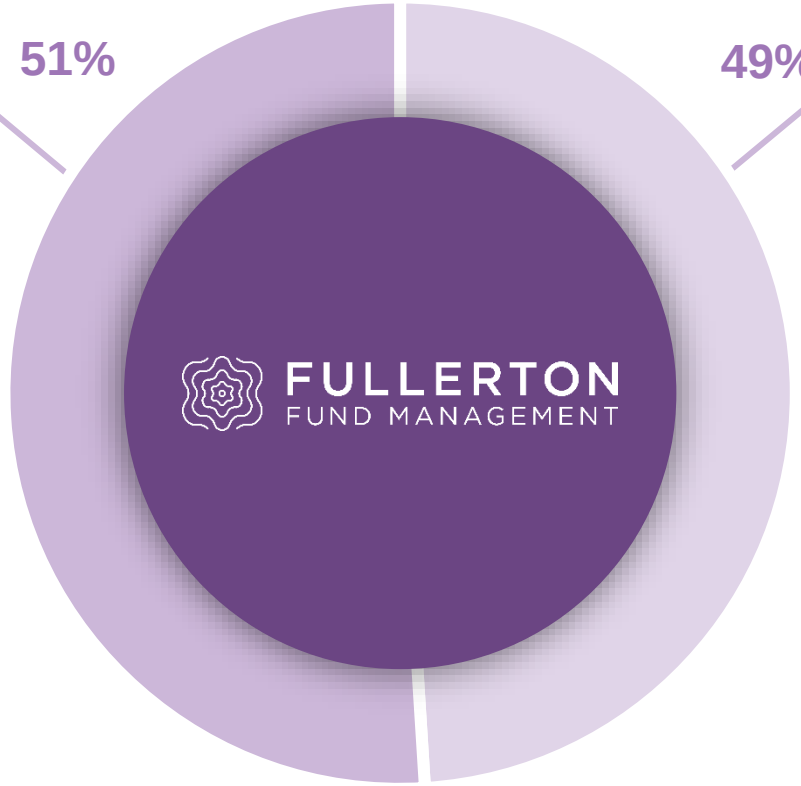
Strong shareholders

Established parentage and synergistic partnership

TEMASEK

SEVIORA HOLDINGS

51%



49%

income

About Temasek
Multi-generational global investor seeking to make a difference

Net portfolio value[3]:
SGD 382 billion (financial figures as at 31 March 2023)

Ratings
Moody's: Aaa[1]
S&P**:** AAA[2]

About Income Insurance Limited
Leading composite insurer in Singapore, offering life, health and general insurance

Assets under management[4]:
SGD 47.3 billion (financial figures as at 31 December 2021)

Ratings
S&P: AA-[4]

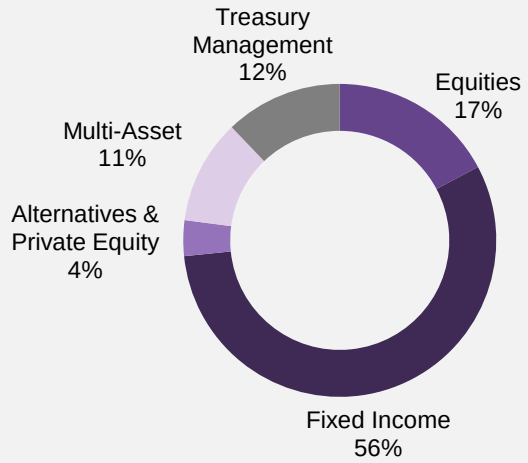
1. Source: Moody's Investor Service, https://www.temasek.com.sg/en/our-financials/credit-quality, as at 31 March 2023 | 2. Source: Standard and Poor's, Ratings Direct, https://www.temasek.com.sg/en/our-financials/credit-quality, 31 March 2023 | 3. Source: https://www.temasek.com.sg/en/what-we-do/our-portfolio | 4. Source: https://www.income.com.sg/about-us/corporate-information
For more information on Moody's and Standard & Poor's (the "Rating Agencies") ratings, products and services, including any updates to Moody's and Standard & Poor's ratings and ratings actions taken by Moody's and Standard & Poor's, visit http://www.moodys.com and http://www.standardandpoors.com. Temasek bears no responsibility for, and is not endorsing, the contents on the Rating Agencies' websites. Temasek disclaims any and all liability whatsoever and howsoever arising in respect of information contained on such websites.

Diversified network

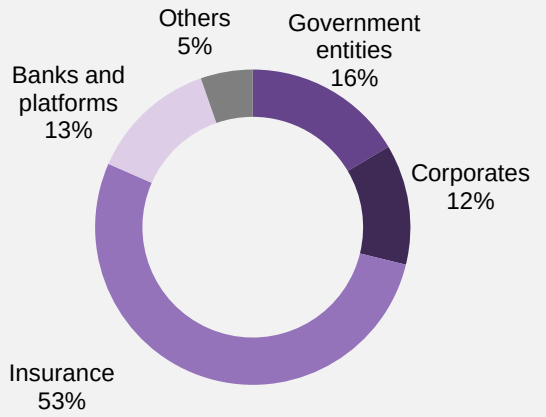
Geographical footprint serves wide spectrum of clients

- Founded in 2003 and headquartered in Singapore
- Total AUM: USD 37.2 billion*

Asset class breakdown*



Client segment*



*Data as at end March 2023; percentage may not add to 100 due to rounding.

What does the AI team do at Fullerton Fund Management?

ML/Data Science-driven Investment Analytics

AI-driven Investment Strategies

Equities

Fixed Income

Multi-Asset

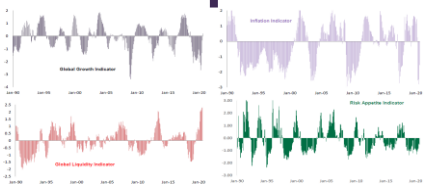
Digital Solutions

Fund Selection

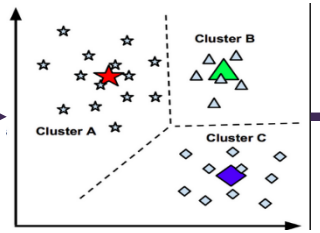
Global Macro Regime Model

(An example of the investment analytics produced by the AI team)

Semi-supervised learning structure for Fullerton Macro Regime model

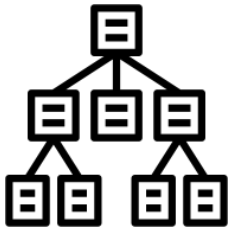


Macro factors

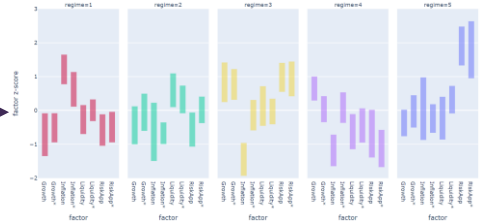


Unsupervised clustering

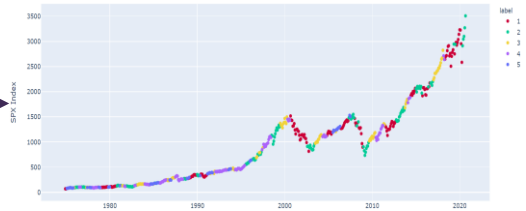
Training



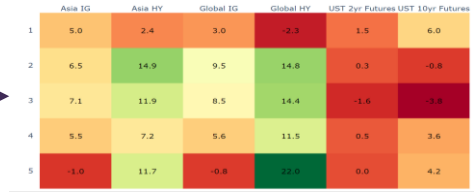
Supervised classification



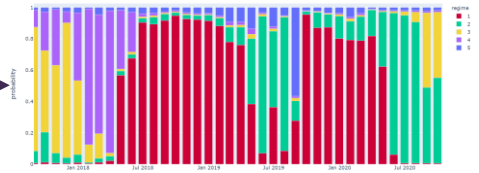
Data-driven regime definitions



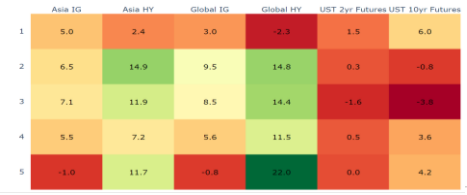
Regime mapping of each month



“Actual” - based asset performance by regime



Regime forecast

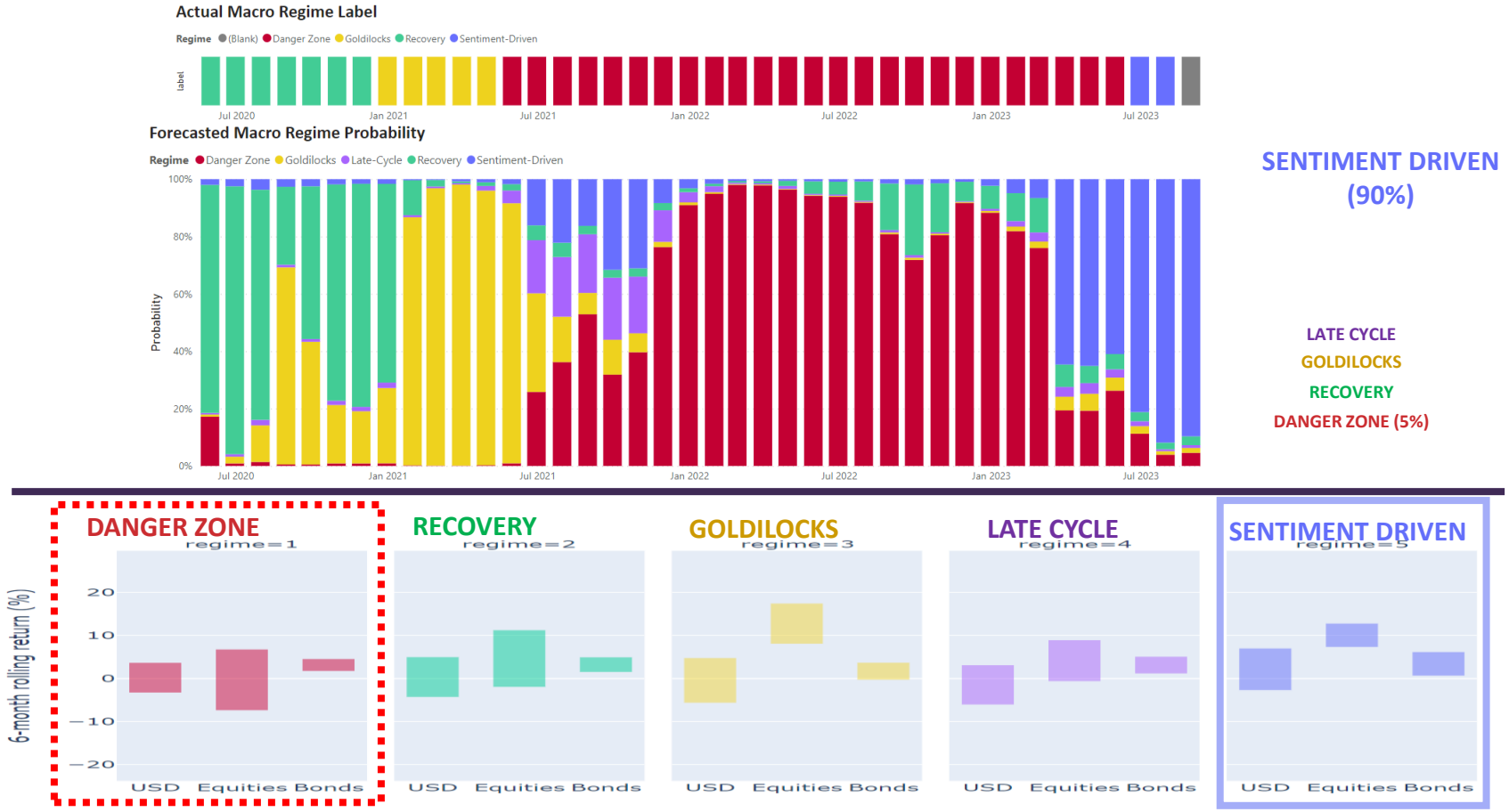


“Forecast” - based asset performance by regime

Regime Model Forecast: transitioned into Sentiment Driven

As of end July 2023: One month ahead probability forecast of being in a particular regime

- Macro Regime stays in Sentiment Driven (SD) for July 2023, and the 1-month ahead forecast shows high probability of continuing to stay in SD for Aug 2023 as well.
- SD: remains positive for risk assets



Returns shown are based on interquartile range

Source: Fullerton, Bloomberg, June 2023. The information presented in the images above are calculated based on Fullerton's internal methodology and subject to changes.

2

Time-based cross validation for financial data

Use Case: Identifying stocks with >5% 3-month forward return

Extracting Financial Data

Define stock tickers and pull historical prices and volumes

```
# pull historical prices (adjusted close, close, high, low, open) and volumes
ticker_list = ['AAPL', 'MSFT', 'INTC','NVDA','AMZN','GOOGL']
start = datetime.datetime(2011,1,1)
end = datetime.datetime(2023,7,31)
df_raw = yf.download(ticker_list, start=start, end=end)
df_raw.tail(5)
```

```
[**********100%**********]  6 of 6 completed
```

	Adj Close						Close				...	Open				Volume			
	AAPL	AMZN	GOOGL	INTC	MSFT	NVDA	AAPL	AMZN	GOOGL	INTC	...	GOOGL	INTC	MSFT	NVDA	AAPL	AMZN	GOOGL	INTC
Date																			
2023-07-24	192.750000	128.800003	121.529999	33.509445	345.109985	446.119995	192.750000	128.800003	121.529999	33.630001	...	121.660004	33.840000	345.850006	447.309998	45377800	45591100	29686100	25225800
2023-07-25	193.619995	129.130005	122.209999	33.977760	350.980011	456.790009	193.619995	129.130005	122.209999	34.099998	...	121.360001	33.700001	347.109985	449.410004	37283200	39236700	52509600	31771100
2023-07-26	194.500000	128.149994	129.270004	34.236828	337.769989	454.519989	194.500000	128.149994	129.270004	34.360001	...	130.070007	33.720001	341.440002	460.209991	47471900	53910100	61682100	32643200
2023-07-27	193.220001	128.250000	129.399994	34.426147	330.720001	459.000000	193.220001	128.250000	129.399994	34.549999	...	131.669998	34.820000	340.480011	465.190002	47460200	52610700	44952100	58890800
2023-07-28	195.830002	132.210007	132.580002	36.697975	338.369995	467.500000	195.830002	132.210007	132.580002	36.830002	...	130.779999	36.750000	333.670013	466.679993	48291400	46317400	36591200	90863000

Source: Yahoo Finance & Fullerton, Jul 2023.
For illustration purpose only and does not represent Fullerton's current view of the security or constitute any recommendation.

Extracting Financial Data

Transform data

```python
# transform wide table to long table and index by date and ticker
# for price, we use adjusted close because it takes into account splits and dividend distributions
df = df_raw.copy()
df_melt = pd.melt(df.reset_index(),id_vars='Date')
df_melt.columns = ['Date','Feature','Ticker','Feature_Value']
df_pivot = df_melt.pivot(index=['Date','Ticker'],columns='Feature',values='Feature_Value')
feature_list = ['Adj Close','Volume']
df_feature = df_pivot[feature_list]
df_feature.sort_index(inplace=True)
df_feature.tail(12)
```

	Feature	Adj Close	Volume
Date	Ticker		
2023-07-27	AAPL	193.220001	47460200.0
	AMZN	128.250000	52610700.0
	GOOGL	129.399994	44952100.0
	INTC	34.549999	58890800.0
	MSFT	330.720001	39635300.0
	NVDA	459.000000	45597600.0
2023-07-28	AAPL	195.830002	48254600.0
	AMZN	132.210007	46234700.0
	GOOGL	132.580002	36572900.0
	INTC	36.830002	90635600.0
	MSFT	338.369995	28463000.0
	NVDA	467.500000	33039600.0

Source: Yahoo Finance & Fullerton, Jul 2023.
For illustration purpose only and does not represent Fullerton's current view of the security or constitute any recommendation.

Feature Engineering

Compute rolling change in prices and volumes over 3, 6, 9 and 12 months

Date	Ticker	Vol(62D)	Vol(126D)	Vol(186D)	Vol(252D)	Rtn(62D)	Rtn(126D)	Rtn(186D)	Rtn(252D)
2023-07-03	AAPL	-0.447871	-0.584456	-0.641829	-0.682126	0.159814	0.489241	0.323345	0.416100
	AMZN	-0.312889	-0.486056	-0.550014	-0.710637	0.271555	0.546923	0.075399	0.226062
	GOOGL	-0.422102	-0.379952	-0.498527	-0.664862	0.148908	0.355568	0.179654	0.100374
	INTC	-0.723817	-0.490932	-0.628708	-0.551720	0.026381	0.303663	0.250198	-0.064953
	MSFT	-0.497305	-0.367311	-0.641465	-0.605788	0.179295	0.408976	0.368267	0.328728
	NVDA	-0.502882	-0.441544	-0.660707	-0.711095	0.516803	1.905197	2.222864	1.800189
2023-07-05	AAPL	0.013873	-0.390916	-0.409592	-0.339631	0.156763	0.476851	0.312879	0.385392
	AMZN	-0.262360	-0.424764	-0.255552	-0.508425	0.254257	0.552143	0.077966	0.190033
	GOOGL	0.129594	0.150023	0.243853	-0.222438	0.162624	0.379916	0.200335	0.119669
	INTC	-0.323965	0.213812	0.250488	-0.072213	-0.013803	0.250128	0.211546	-0.069202
	MSFT	-0.296306	-0.171666	-0.106880	-0.204281	0.180059	0.416638	0.367157	0.315273
	NVDA	-0.122016	0.042282	-0.361865	-0.439729	0.541594	1.896440	2.205345	1.916200

Take for example,

$$\text{Rtn (62D)} = \frac{Price_T}{Price_{T-62days}} - 1$$

$$\text{Vol (62D)} = \frac{Vol_T}{Vol_{T-62days}} - 1$$

Source: Yahoo Finance & Fullerton, Jul 2023.
For illustration purpose only and does not represent Fullerton's current view of the security or constitute any recommendation.

Use Case: Identifying stocks with >5% 3-month forward return

1. Data Labelling

```python
# label data as positive if return >5%, otherwise negative
# this is a binary classification
rtn_threshold = 0.05

# predict 3 months forward return, whether return is >5% or otherwise
rolling_window_for_labeling = 62 # in days

df_rtn = df_rolling_returns[f'Rtn({rolling_window_for_labeling}D)'].unstack().shift(-rolling_window_for_labeling)
df_y = pd.melt(df_rtn.reset_index(),id_vars='Date',value_name=f'Rtn({rolling_window_for_labeling}D)')
df_y['y'] = df_y[f'Rtn({rolling_window_for_labeling}D)'].apply(lambda x: 1 if x>rtn_threshold else 0)
df_y.dropna(inplace=True)
df_y.drop(f'Rtn({rolling_window_for_labeling}D)',axis=1,inplace=True)
df_y = df_y.sort_values(['Date','Ticker']).set_index(['Date','Ticker'])
```

Date	Ticker	Rtn(62D)	y
2023-04-03	AAPL	0.159814	1
2023-04-03	AMZN	0.271555	1
2023-04-03	GOOGL	0.148908	1
2023-04-03	INTC	0.026381	0
2023-04-03	MSFT	0.179295	1
2023-04-03	NVDA	0.516803	1

2. Size of data

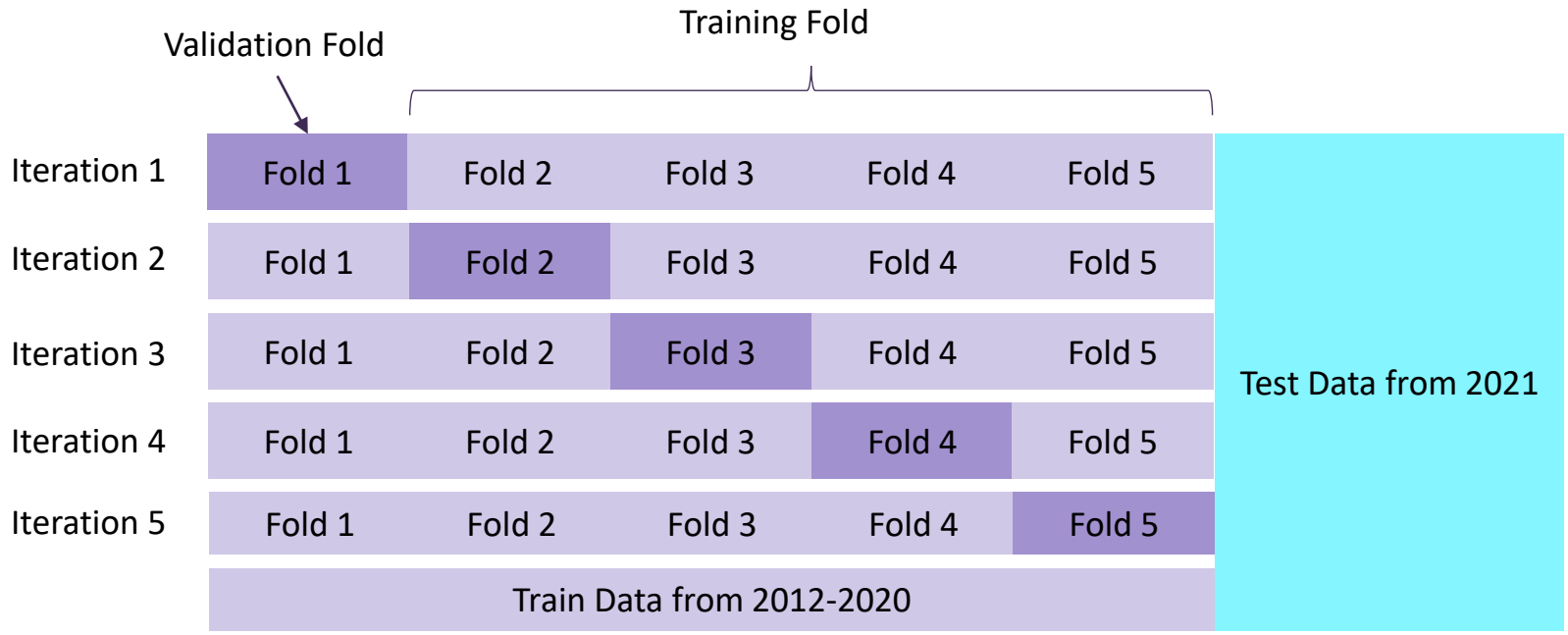
```
number of train data    = 13596, from 2012-01-03 to 2021-01-04
number of test data     = 3498,  from 2021-01-05 to 2023-04-28
```

3. Class distribution

name	rtn <= 5.0%	rtn > 5.0%
test	0.565466	0.434534
train	0.441600	0.558400

Source: Yahoo Finance & Fullerton, Jul 2023.
For illustration purpose only and does not represent Fullerton's current view of the security or constitute any recommendation.

K-fold Cross Validation without temporal consideration



This will introduce look-ahead bias because training fold includes future data and validation fold includes past data.

Source: Fullerton, July 2023.

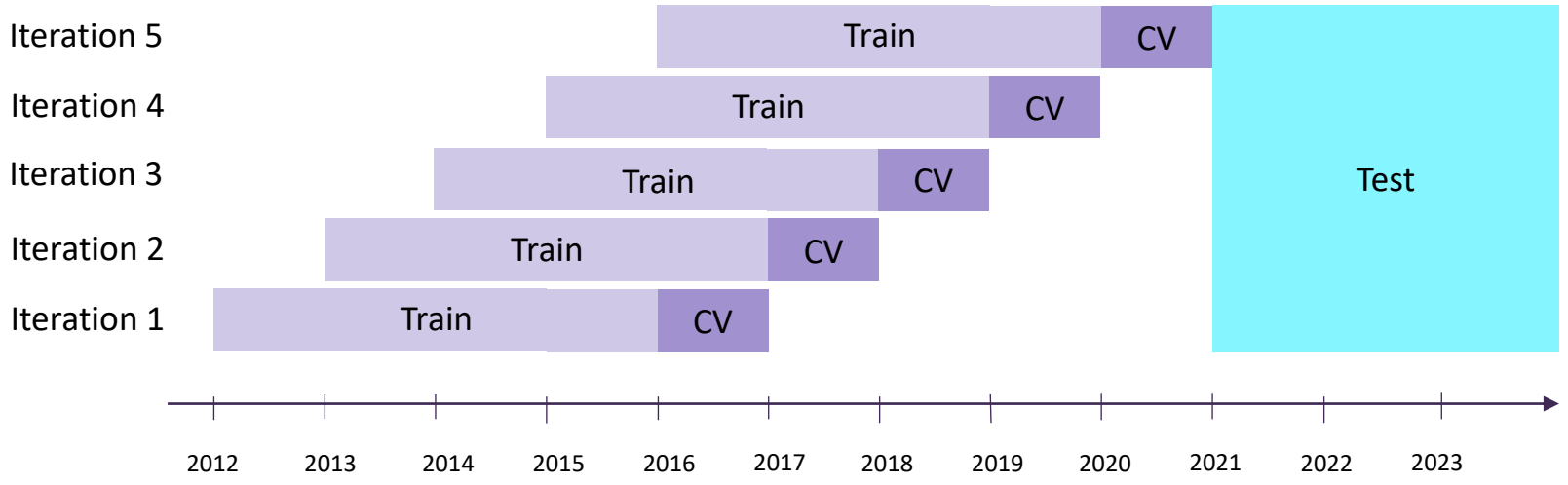
K-fold Time-based Cross Validation

Limitation of TimeSeriesSplit method in Scikit-learn

- Splits data in a sliding window approach based on the **fixed number of records in training fold and validation fold**, instead of fixed number of days
- Assumes there is **one observation per date** (recall we have six observations per date)

Suggested solution from Medium article

- The author, Or Herman-Saffar has written a solution to **consider the window size of training and validation fold based on number of days**
- The returned CV splits works like any other scikit-learn cross validator and could be used with any of the methods



Source: Medium & Fullerton, July 2023.

K-fold Time-based Cross Validation

5-fold time-based CV

```
train_period = 4
test_period = 1

data_for_modeling = df_X_train.reset_index().drop('Ticker',axis=1)
tscv = TimeBasedCV(train_period=train_period,
                   test_period=test_period,
                   freq='years')
index_output = tscv.split(data_for_modeling, date_column='Date')
print(f'number of splits: {tscv.get_n_splits()}')
```

```
Train period: 2012-01-03 - 2016-01-03 , Test period 2016-01-03 - 2017-01-03 # train records 6036 , # test records 1512
Train period: 2013-01-03 - 2017-01-03 , Test period 2017-01-03 - 2018-01-03 # train records 6042 , # test records 1512
Train period: 2014-01-03 - 2018-01-03 , Test period 2018-01-03 - 2019-01-03 # train records 6042 , # test records 1506
Train period: 2015-01-03 - 2019-01-03 , Test period 2019-01-03 - 2020-01-03 # train records 6036 , # test records 1512
Train period: 2016-01-03 - 2020-01-03 , Test period 2020-01-03 - 2021-01-03 # train records 6042 , # test records 1512
number of splits: 5
```

Applying in grid search

```
grid_lgb_timebased = GridSearchCV(
    estimator, param_grid, cv=index_output, refit=True, scoring='f1', return_train_score=True)

grid_lgb_timebased.fit(
    df_X_train,
    df_y_train)

print('Best parameters found by grid search are:', grid_lgb_timebased.best_params_)
```

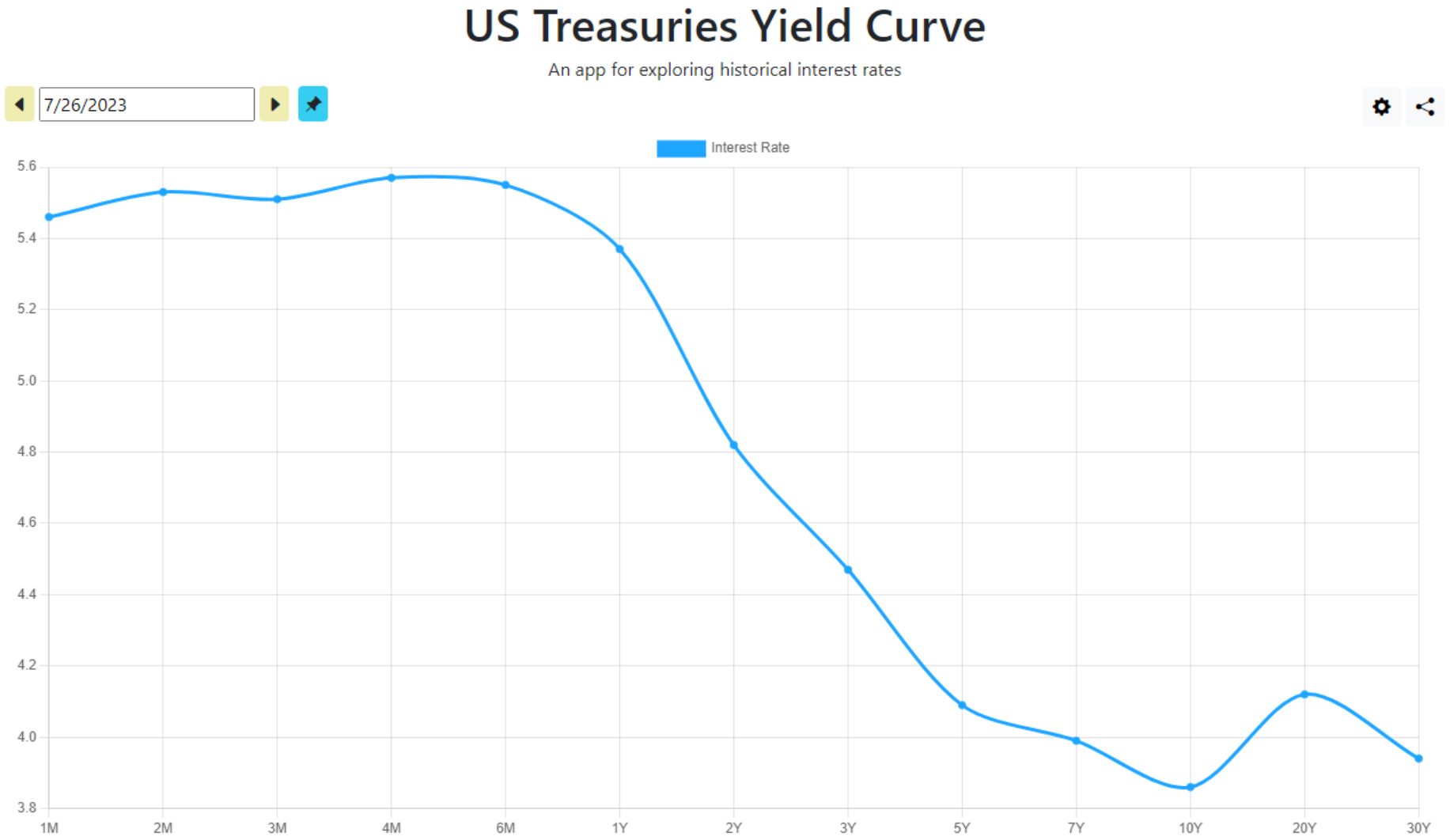
```
Best parameters found by grid search are: {'colsample_bytree': 0.8, 'learning_rate': 0.1, 'max_depth': -1, 'n_estimators': 5,
'num_leaves': 5, 'random_state': 0, 'reg_alpha': 1, 'reg_lambda': 1}
```

Source: Medium & Fullerton, July 2023.

3

Principal Component Analysis (PCA) decomposition of the US treasury yield curve

PCA on UST: introduction



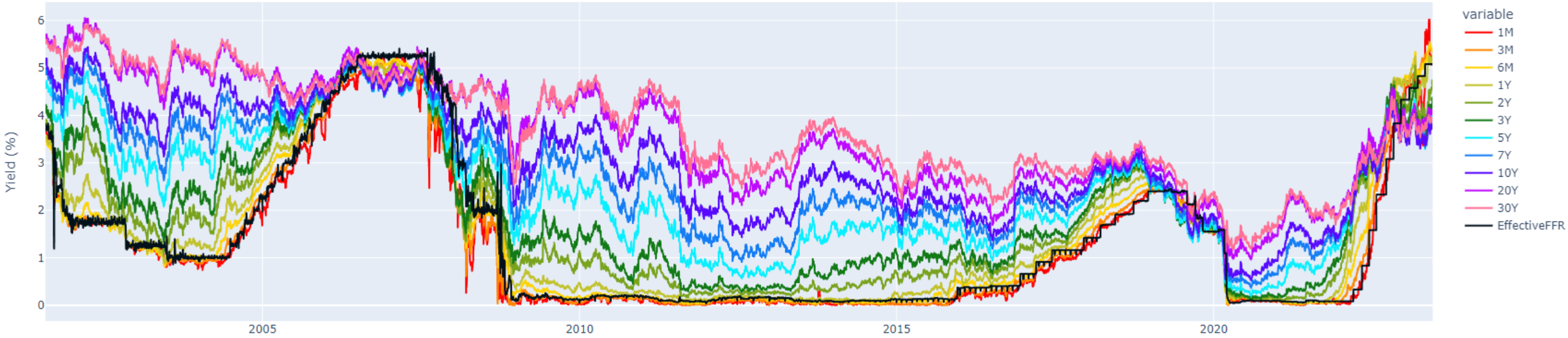
Source: https://www.ustreasuryyieldcurve.com/.

PCA on UST: the process

1. Data from https://fred.stlouisfed.org/:
   - Effective Fed Funds rate
   - Yields on US treasuries of various tenors (3months – 30years, total of 11 tenors)

DATE	1M	3M	6M	1Y	2Y	3Y	5Y	7Y	10Y	20Y	30Y
2023-06-09	5.25	5.37	5.39	5.17	4.59	4.23	3.92	3.84	3.75	4.05	3.89
2023-06-12	5.24	5.40	5.38	5.18	4.55	4.16	3.89	3.82	3.73	4.04	3.87
2023-06-13	5.19	5.36	5.36	5.26	4.67	4.30	4.01	3.94	3.84	4.12	3.94
2023-06-14	5.18	5.36	5.36	5.27	4.74	4.37	4.06	3.95	3.83	4.09	3.90
2023-06-15	5.18	5.33	5.33	5.21	4.62	4.23	3.91	3.82	3.72	4.02	3.85

Time series of each tenor and Fed Fund rate



Source: FRB St. Louis, Fullerton, July 2023.

PCA on UST: the process

2. Standardising the data

```
# scaler = StandardScaler(with_std=False) # Centering the data only
scaler = StandardScaler(with_std=True) # Standardising the data
```

```
df_ustyields_stand = scaler.fit_transform(df_ustyields_raw)
```

```
df_ustyields_raw.describe()
```

	1M	3M	6M	1Y	2Y	3Y	5Y	7Y	10Y	20Y	30Y
count	5708.000000	5708.000000	5708.000000	5708.000000	5708.000000	5708.000000	5708.000000	5708.000000	5708.000000	5708.000000	5708.000000
mean	1.285366	1.357342	1.474128	1.578395	1.790895	2.000638	2.413019	2.746594	3.041594	3.583537	3.697864
std	1.527495	1.566839	1.601385	1.562512	1.462090	1.388535	1.280468	1.211023	1.171078	1.213935	1.120423

```
### Notice that mean for all columns here is effectively 0 now, and all columns have standard deviation of effectively 1.
pd.DataFrame(df_ustyields_stand, columns=df_ustyields_raw.columns).describe()
```

	1M	3M	6M	1Y	2Y	3Y	5Y	7Y	10Y	20Y	30Y
count	5.708000e+03	5.708000e+03	5.708000e+03	5.708000e+03	5.708000e+03	5708.000000	5.708000e+03	5.708000e+03	5.708000e+03	5.708000e+03	5708.000000
mean	-7.966842e-17	5.975132e-17	1.991711e-17	1.394197e-16	-1.991711e-17	0.000000	3.186737e-16	-1.195026e-16	3.186737e-16	3.983421e-16	0.000000
std	1.000088e+00	1.000088e+00	1.000088e+00	1.000088e+00	1.000088e+00	1.000088	1.000088e+00	1.000088e+00	1.000088e+00	1.000088e+00	1.000088

Source: FRB St. Louis, Fullerton, July 2023.

PCA on UST: the process

3. Creating the PCA model

- 3-factor PCA model

```
pca_model = PCA(n_components = 3) # here we create a 3-factor PCA model
pca_features = pca_model.fit_transform(df_ustyields_stand)
```

- Outputs from model:

1

```
df_variance_ratio
```

	%Variance	Cumul_%Variance
PC1	83.583801	83.583801
PC2	15.052522	98.636323
PC3	1.094312	99.730636

2

```
df_factorloadings
```

	1M	3M	6M	1Y	2Y	3Y	5Y	7Y	10Y	20Y	30Y
PC1	0.298265	0.300201	0.302820	0.308683	0.318998	0.324226	0.324227	0.314526	0.298179	0.267549	0.250015
PC2	-0.303547	-0.308618	-0.301859	-0.269791	-0.175946	-0.084279	0.093978	0.219663	0.330145	0.447932	0.493887
PC3	0.442258	0.324421	0.176466	-0.013204	-0.298866	-0.415742	-0.376478	-0.251666	-0.012858	0.221383	0.389146

3

```
df_pcafeatures # the time series of values of each PC
```

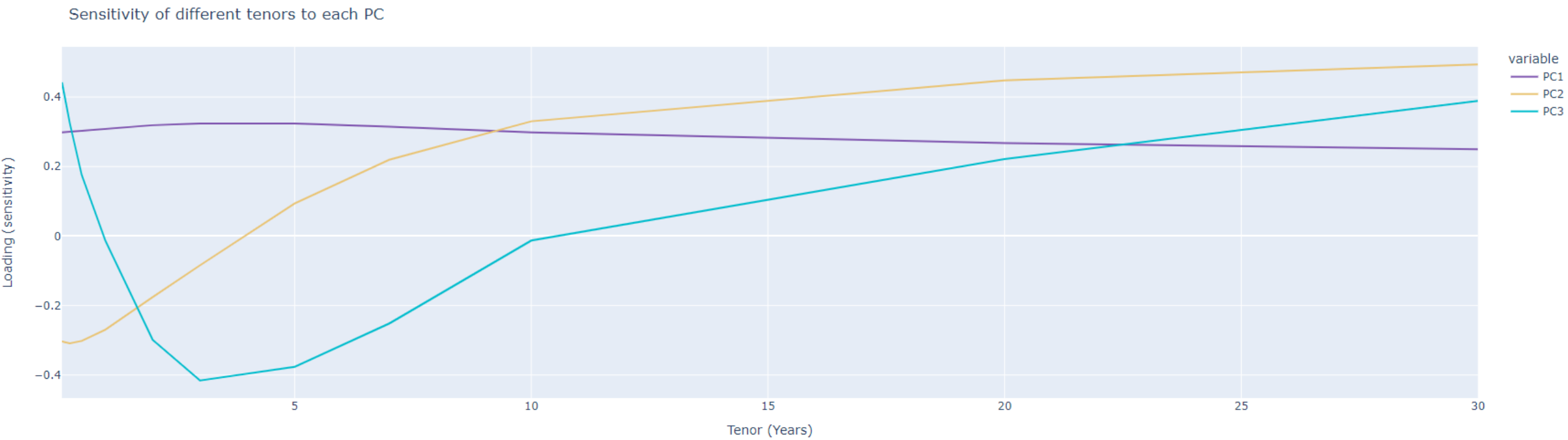
DATE	PC1	PC2	PC3
2001-07-31	5.026786	-0.677542	-0.223893
2001-08-01	5.084734	-0.710072	-0.185757
2001-08-02	5.187726	-0.768991	-0.135574

Source: Fullerton, July 2023

PCA on UST: investigating the results (1)

Intuition behind each PC

By plotting $df\_factorloadings$ aross the yield curve in this fashion, we can see clearly how different parts of the yield curve react differently to each PC.



PC1: The sensitivity towards PC1 stays roughly the same across the various tenors; basically the magnitude of impact for each unit change in PC1 is roughly the same across tenors -- based on this observation, we understand that PC1 controls for the overall level of the tenors.

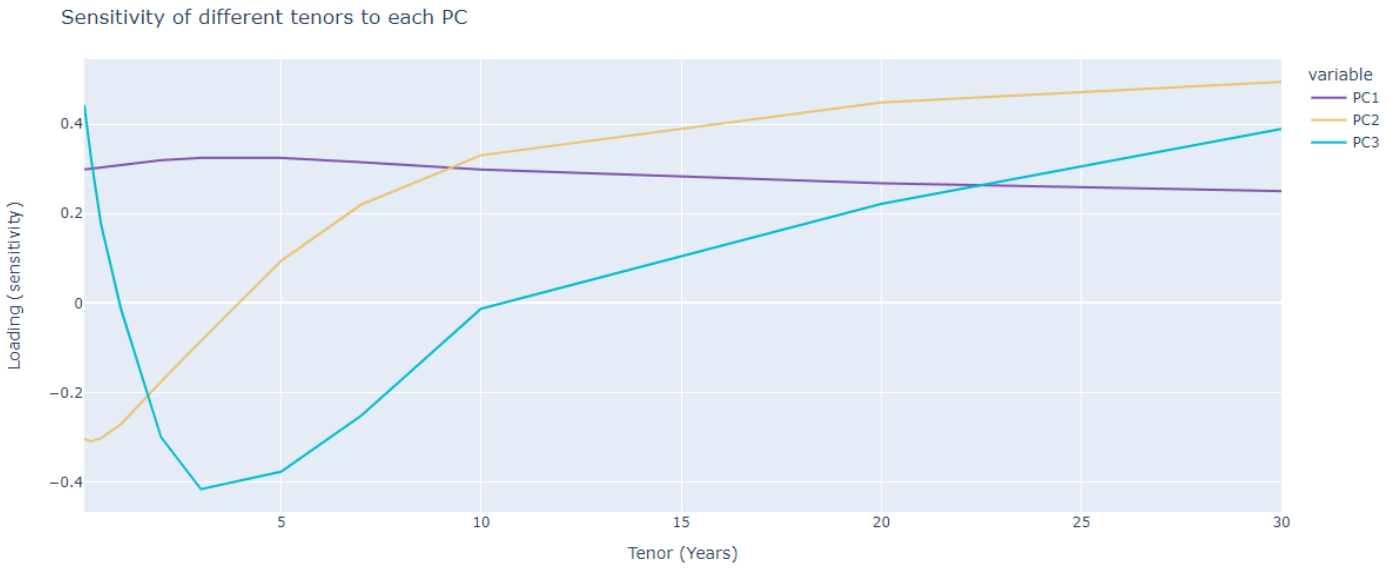
Source: Fullerton, July 2023

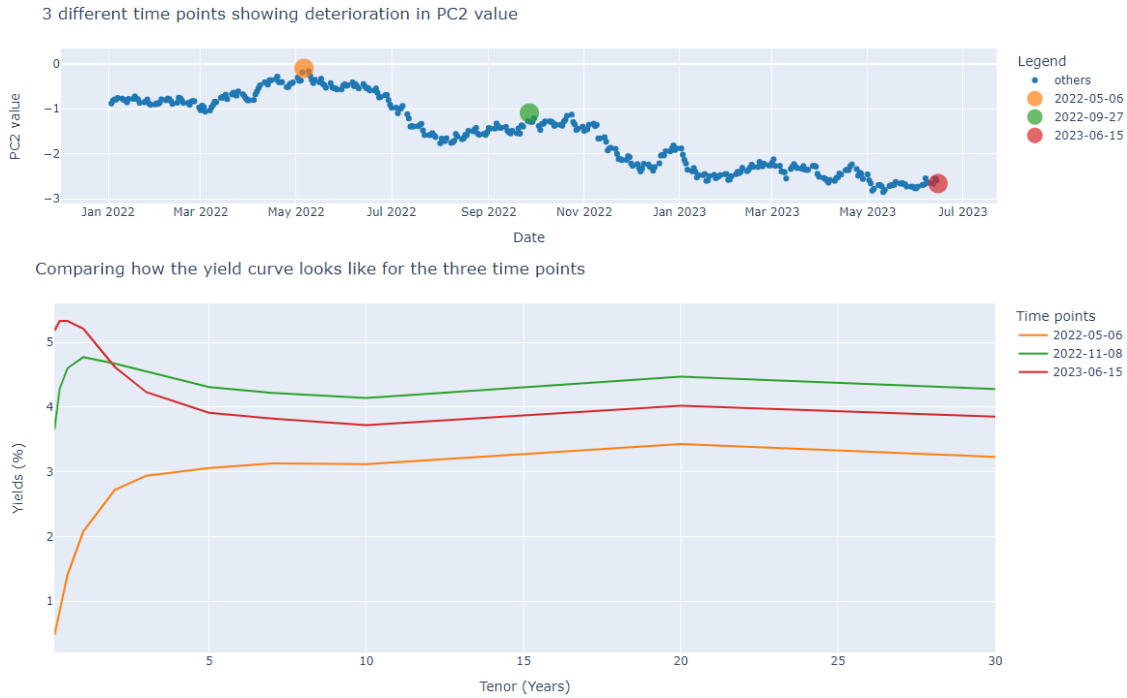
PCA on UST: investigating the results (2)

Intuition behind each PC

PC2:

- For each unit increase in PC2, the long-end will move up while the short-end will move down (reference chart on left)
- Illustration of how the yield curve looks like as PC2 deteriorates (reference charts on right)





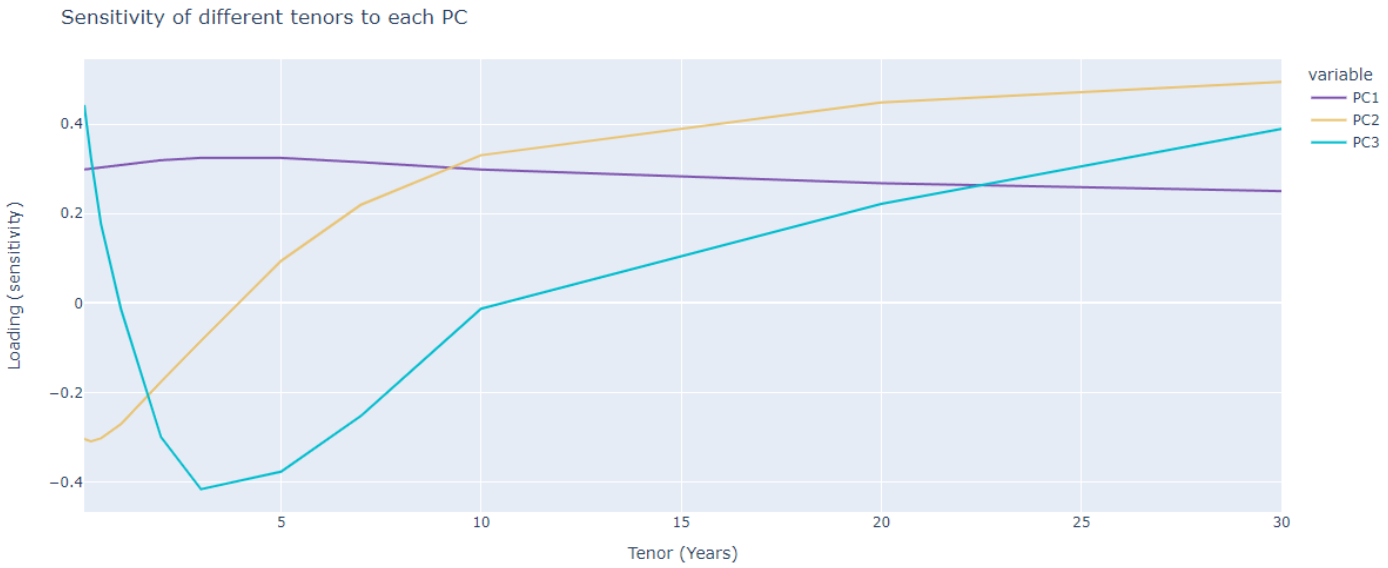
Source: Fullerton, July 2023

PCA on UST: investigating the results (3)

Intuition behind each PC

PC3:

- Both ends of the yield curve are affected in a similar way, and this is opposite to how the belly of the curve is affected. (reference chart on left)
- As PC3 improves from negative to positive value, the curve changes from ∩-shape to ∪-shape . (reference charts on right)

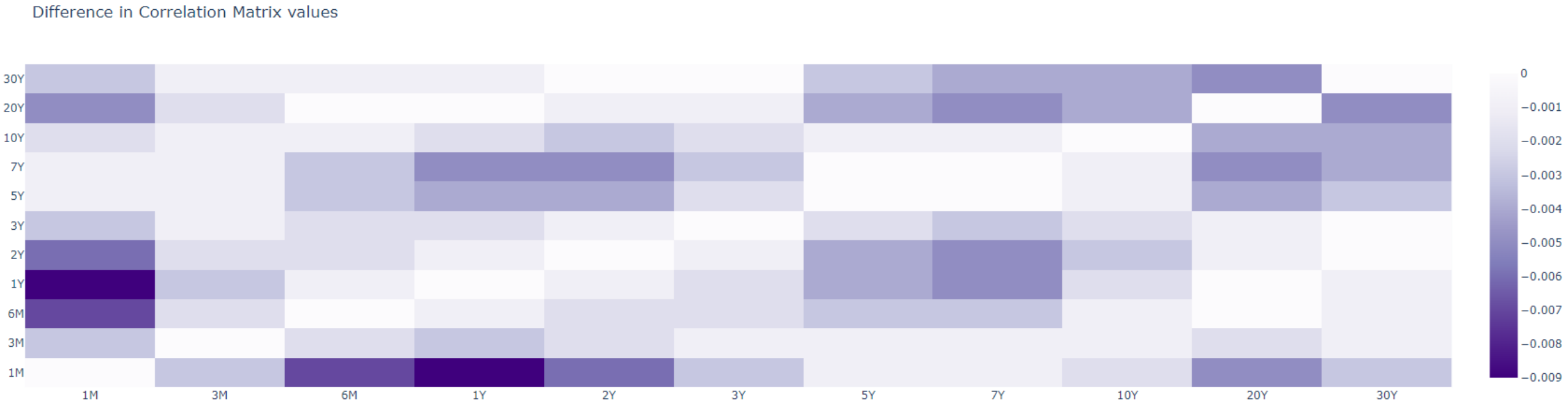


Source: Fullerton, July 2023

PCA on UST: investigating the results (4)

Checking if PCA can recreate yield curve satisfactorily

To verify that the PCs can replicate the original matrix almost perfectly, we compare the correlation matrix of the actual yields against that of the yields that are calculated based on the PCs and its loadings.

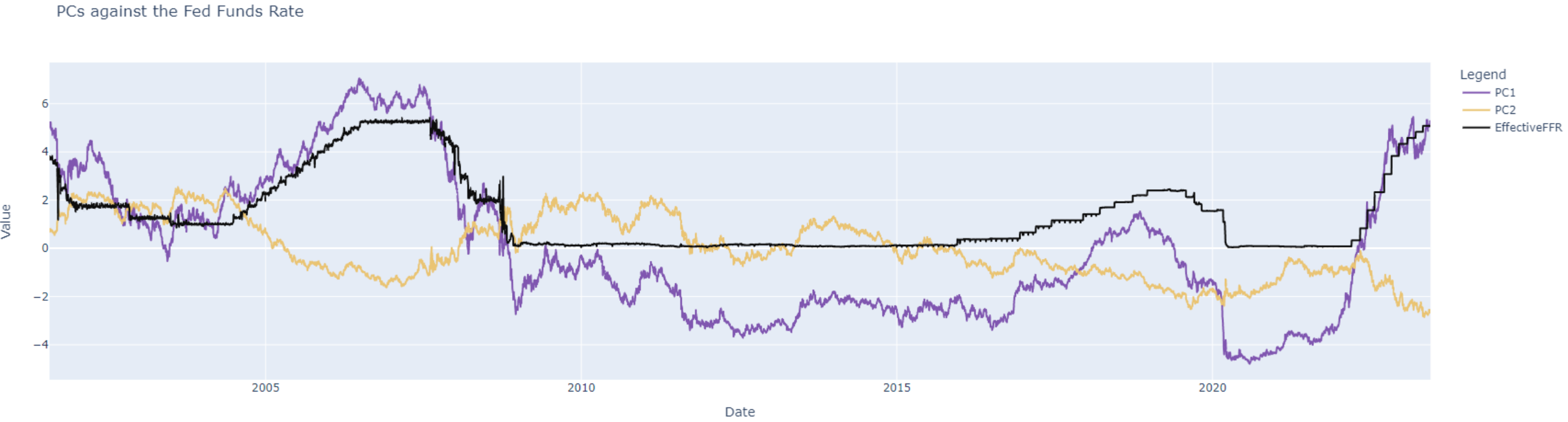


Source: Fullerton, July 2023

PCA on UST: Investigating the results (5)

Attaching meaning to PCs

To be able to meaningfully use the PCs, we need to identify what affects the PCs – here we show an example using Effective Fed Funds Rate.



From the chart, we notice:

- PC1 generally falls during a rate cut, and increases during a rate hike. In other words, generally we see that PC1 tends to rise and fall along with the effective Fed Funds.
- PC2 seems to show a general pattern of falling during rate hikes (yield curve flattens) while increasing during rate cuts.

Source: FRB St. Louis, Fullerton, July 2023.

QR Code to Time-based CV and PCA on UST

Disclaimer

No offer or invitation is considered to be made if such offer is not authorised or permitted. This is not the basis for any contract to deal in any security or instrument, or for Fullerton Fund Management Company Ltd (UEN: 200312672W) (“Fullerton”) or its affiliates to enter into or arrange any type of transaction. Any investments made are not obligations of, deposits in, or guaranteed by Fullerton. The contents herein may be amended without notice. Fullerton, its affiliates and their directors and employees, do not accept any liability from the use of this publication.

References to specific securities are for illustration purpose only and does not represent Fullerton’s current view of the security or constitute any recommendation.

Thank you