Forecasting Report

Group MSG

Martin Fluder
Jonathan Chan
Saloni Jaitly
Gareth Goh

Overview of Methodology

1. Features

- General Market Data (Betas & Residuals)
- Fundamental Factors
- Time Series Factors (ARIMA/LSTM)
- Analyst Reports
- Price Movement (Bollinger Bands/SMA/EMA)
- Sentiment Analysis
- "Alpha" Features

99 Features

2. Feature Selection

Model Selection:

❖ Full Multiple Linear Regression (Issue of Overfitting)

❖ PCA Model [Unsupervised] (Not entirely predictive)

❖ **Selective Factor Model:**

Backtesting for robust features: 1) feature correlation, 2) Significance, 3) Consistency & Expectation

3. Binning/ Decision

Binning

❖ Bootstrapping Prediction Errors

❖ **'Bayesian' Uniform prior**

Decision:

❖ Portfolio Optimization

❖ "Ad-hoc" Approaches

Week 0 - 2 Highlights:

1 Features

1) Momentum:
Predicted Price Change = Pct Change of last week

2) ARIMA:
Train/test set

3) Moving Average:
5, 10, 15, 20, 25 weeks

4) Volume

2 Market Betas

1) 'Scrape' names:
- 500 stocks
- 600 ETFs
- 50 representative cryptos

2) Perform PCAs on all ~1200 prices in the test set
(~1/1/2021 - 6/1/2022)

3) Extract 2 PCA Components
Captures up <80% of the R^2

3 Binning

- Ranking based on prediction → mean prediction ranking

- Create distribution bootstrapping prediction errors in test set

- We later realize that metric strongly prefers "flatter" distributions

4 Decisions

Simple Idea:

Scale by the standard deviation of prediction error in test set (provides a measure of uncertainty)

Week 0 – 2 Lessons:

- **Market beta:**
  - Captures large amount of variance → up to 90% R^2
- **ARIMA:**
  - Not very predictive too far into the future.
- **Bottomline**: More features!


```python
X = combined_members_pcaed.reset_index(drop=True)[:-1]
y = weekly_data[weekly_data.symbol == 'ABBV']['close'][1:].reset_index(drop=True)
X = sm.add_constant(X)
est = sm.OLS(y, X).fit()
est.summary()
```

OLS Regression Results

Dep. Variable:	close	**R-squared:**	0.904
Model:	OLS	**Adj. R-squared:**	0.903
Method:	Least Squares	**F-statistic:**	1113.
Date:	Sat, 02 Dec 2023	**Prob (F-statistic):**	1.79e-180
Time:	16:23:05	**Log-Likelihood:**	-1366.0
No. Observations:	360	**AIC:**	2740.
Df Residuals:	356	**BIC:**	2756.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	99.5562	0.570	174.630	0.000	98.435	100.677
0	1.3552	0.026	52.582	0.000	1.304	1.406
1	1.1012	0.045	24.332	0.000	1.012	1.190
2	-0.0733	0.054	-1.355	0.176	-0.180	0.033

Omnibus:	4.331	**Durbin-Watson:**	0.169
Prob(Omnibus):	0.115	**Jarque-Bera (JB):**	4.340
Skew:	0.239	**Prob(JB):**	0.114
Kurtosis:	2.754	**Cond. No.**	22.1

Week 3 – 4 Highlights:

New Features

LSTM + Light GBM:

- Performs well even on OOS data vs. ARIMA and Momentum-based ideas for predicting returns

Fundamental Data:

- A total of 22 Valuation Ratios/Factors from Factset
- EPS, BVG, P/E, P/S, EV/EBITDA *etc*.

Overall Approach:

- Shared Drive Folder where we collect features

- Central notebook running model, binning and decisions

PCA Feature Selection

- Done cross-sectionally
- Distinguish between Stocks/ETFs/Cryptos (specifically have 'more data' for stocks)
- CV on % of variation → roughly 2-3 vectors

Decision

Portfolio Optimization

- Maximize Sharpe Ratio
- Predicted Sharpe:

Stocks: ~2

ETFs: ~ 3

Cryptos: ~ 0

Week 3 – 4 Lessons:

LSTM/LGBM:

- LGBM outperforms 'momentum' substantially for OOS MSE
- LSTM not so much

Fundamental Data

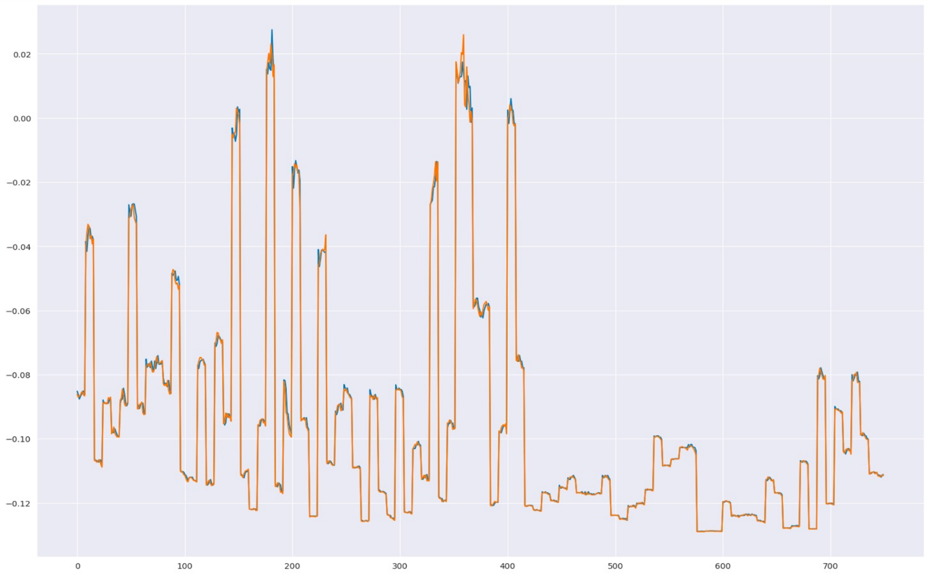
- Importance of accounting for time lag
- Not Universal → cross-sectional feature selection fails.
- Using all 22 valuation ratios does not significantly improve the prediction
  - Need for Selection

3.1.3.a) Test Set performance

```
a = y_test_s
b = model.predict(X_test_s).reshape(-1)
c = X_test_s[:,0]
d = gbm.predict(X_test_s).reshape(-1)
```

```
print(f'LGBM loss = {((a-d)**2).sum()/a.shape[0]}, LSTM loss = {((a-b)**2).sum()/a.shape[0]} and momentum loss = {((a-c)**2).sum()/a.shape[0]}')
```

```
LGBM loss = 0.0011615374440617882, LSTM loss = 0.0033138850596911205 and momentum loss = 0.0029761479631527643
```



Week 3 – 4 Lessons:

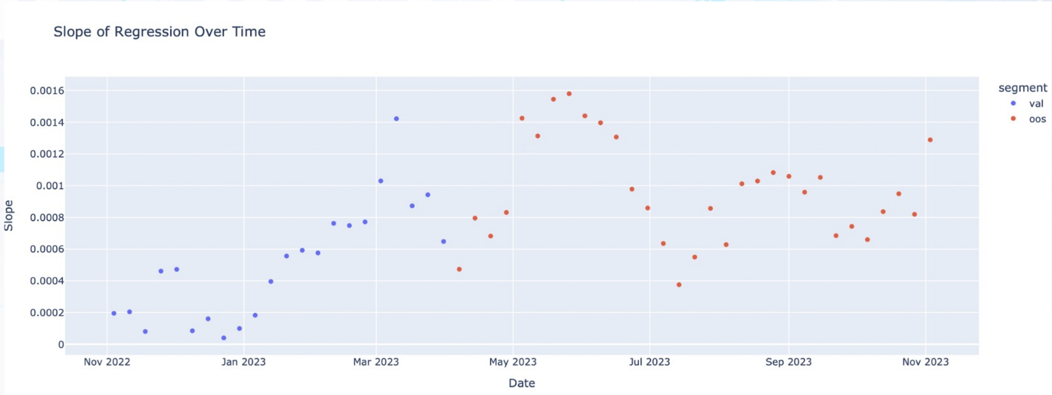
LSTM/LGBM:

- LGBM outperforms ‘momentum’ substantially for OOS MSE
- LSTM not so much

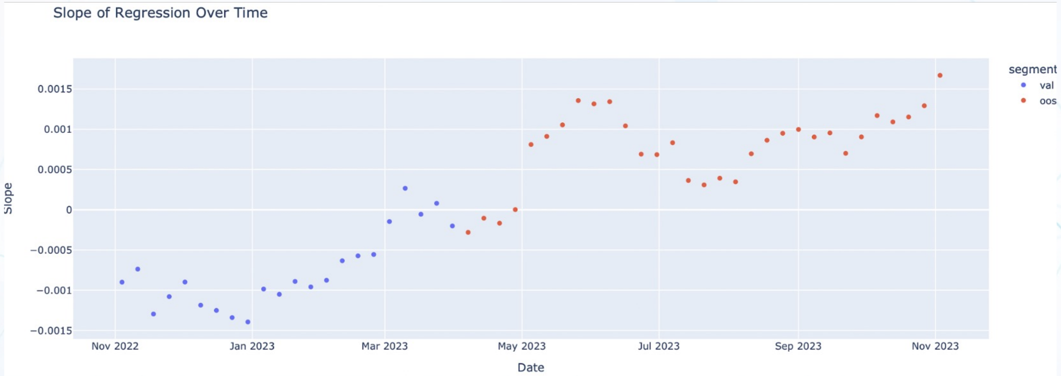
Fundamental Data

- Importance of accounting for time lag
- Not Universal → cross-sectional feature selection fails.
- Using all 22 valuation ratios does not significantly improve the prediction
    - Need for Selection

P/BV



P/S



Week 5 – 6 Highlights:

Production Model

- Add a 'production model' that trains until Friday
- Continued cross-sectional PCA for feature selection

01

02

03

New Features

Analyst Reports (FactSet)

- Combining Buy/Overweight/Hold/ Underweight/Sell Ratings
- Accounting for predicted target price

Started working on scraping FactSet headlines → not yet any features

Overall Approach

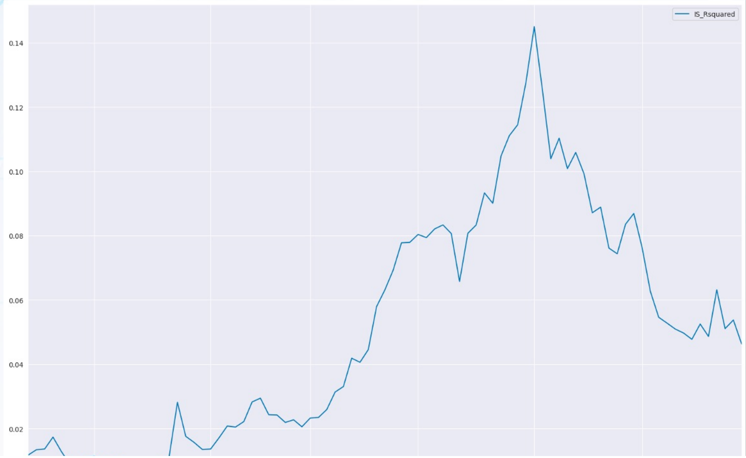
Naïve Backtesting

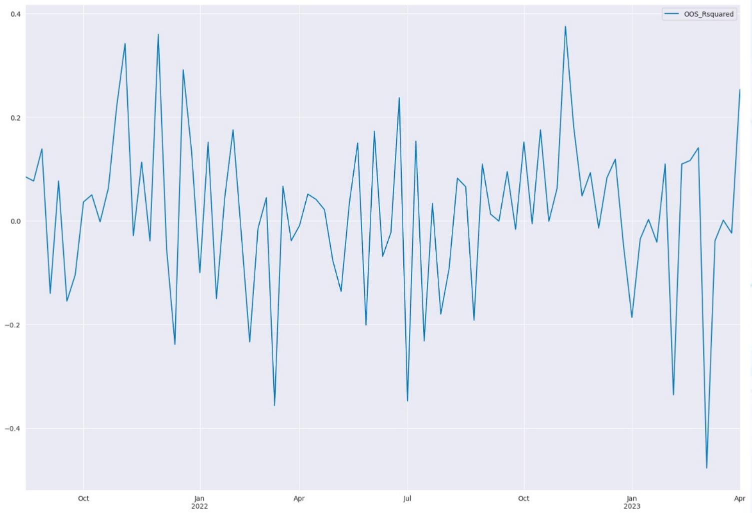
- Comparing portfolio with predictions against actual values with OOS R^2
- Found R^2 to be variable, and we later switched to correlation
- More carefully looking at individual feature backtesting

Week 5 – 6 Lessons:

Overall Findings

- Some more testing of our inputs reveals that cryptos contribute significantly to our errors in the decision portion
  - Set all crypto decisions to zero.
- Remove Light-GBM → seems to overfit.
  - Train LSTM/LGBM/ARIMA on all ~1200 market data in the training period to avoid overfitting
  - Found LGBM performs poorly → drop
- IS and OOS R^2 of our full model.

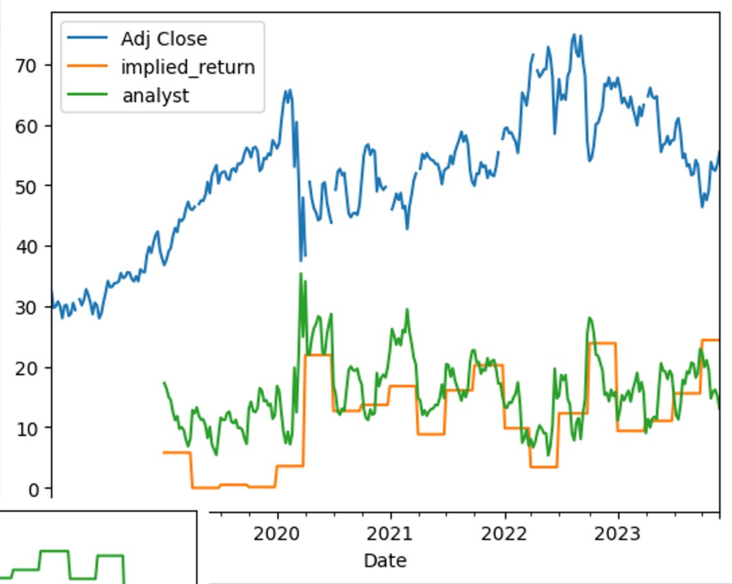


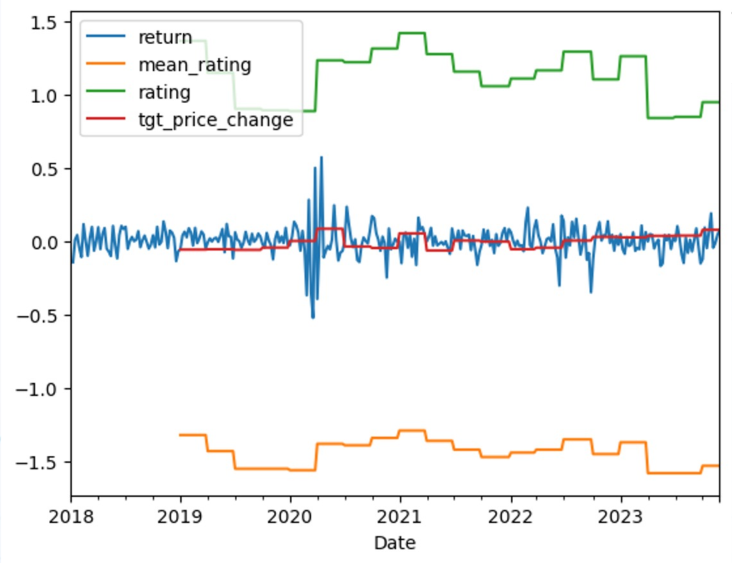


Week 5 - 6 Lessons:

Analyst Reports

- New ideas for more features from the same FactSet information:
- Key Features:
    - Rating: Creating scores for analysts' buy/hold/sell ratings
    - Target Price: Using deviation from target price as a feature
    - Target Price Change:
- Deciding on frequency of downloading data





Week 7 – 10 Highlights:

Backtesting

Backtesting Model Selection:

- Backtesting individual features and combinations of features

Backtesting Decisions:

- Testing how portfolios would have performed in previous weeks
- Backtesting our portfolio against **residual price**

Binning/Decisions

Optimizing Ranking

- Realizing it's very hard to outperform 0.2 version which gives 16%
- Changing to using 0.2 as a prior and superimposing our prediction distribution with some weighing hyperparameter ('Bayes').

New Features

Sentiment Analysis:

- Bing Scrape and Factset Headline Analysis

Alpha Features:

- A sample of the 101 Formulaic Alphas

Price Movement:

- Simple/Exponential Moving Averages for 5-50 weeks
- Revamped momentum for 5-50 weeks, z-scoring by ticker
- Bollinger Bands: A weighted signal when the price goes beyond 2σ of SMA

Week 7 – 10 Lessons:

- **Predict on residual price** = price – market, market = running PCA on ticker type (stock, etf, crypto)
- Focus on the **decision column** and **by-hand** feature selection
- Automated feature selection (PCA) performs poorly → get roughly middle-of-pact performance
- Can improve a lot by more broad testing of feature selection and several thresholds (→ include into decision if past prediction > threshold correlation)

Week 10 submission (rank 1 decision):

Features:

```
1 ['Mean_summ', 'Mean_summ_agg_diff_1',
2  'nr_of_search_results_pct_change',
3  'LSTM_pred_for_following_week',
4  'rating', 'sma_5', 'volume',
5  'Mean_roberta_diff_1']
```

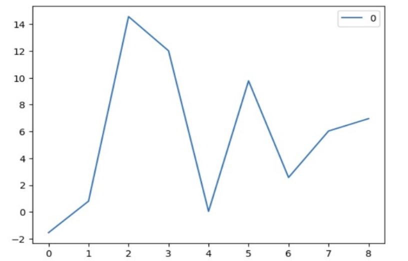
Last 10 week decision

```
1 print(pd.DataFrame(dec_metric[-10:]).mean())
2 print(pd.DataFrame(dec_metric[-10:]).median())
3 print(pd.DataFrame(dec_metric[-10:]).std())
```

```
0    5.688926
dtype: float64
0    6.030732
dtype: float64
0    5.647706
dtype: float64
```

```
1 pd.DataFrame(dec_metric[-10:]).plot()
```

```
<Axes: >
```



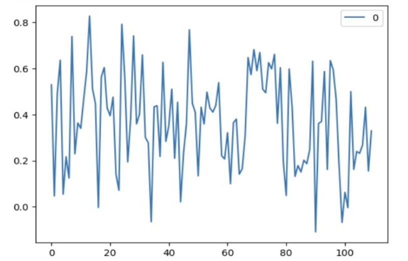
Residual prediction

```
1 print(pd.DataFrame(symb_corr).median())
2 print(pd.DataFrame(symb_corr).mean())
3 print(pd.DataFrame(symb_corr).std())
```

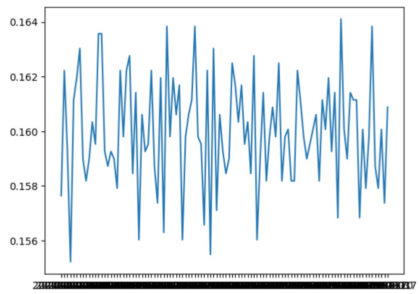
```
0    0.375099
dtype: float64
0    0.369917
dtype: float64
0    0.212196
dtype: float64
```

```
1 pd.DataFrame(symb_corr).plot()
```

```
<Axes: >
```



Ranking performance

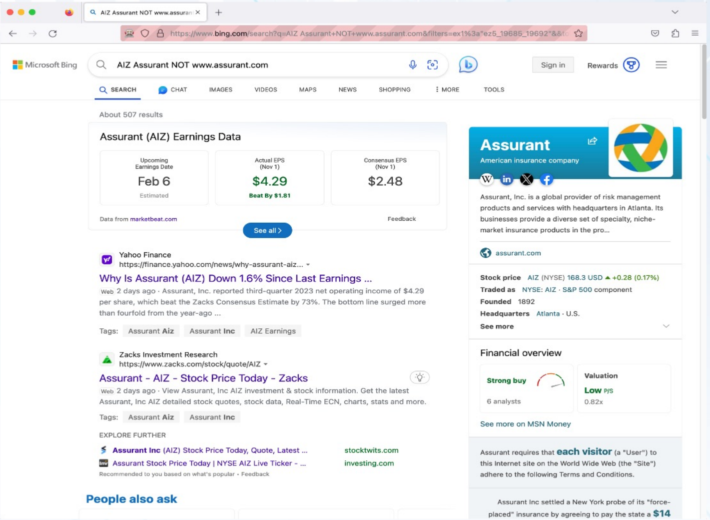


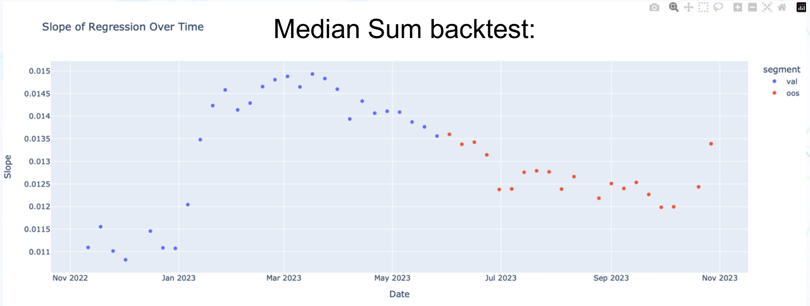
Bing Scrape & Sentiments: Methodology

- Bing search: Extract weekly 50 headlines, <u>summaries</u>, <u>date</u>, <u>number of search results</u>, and <u>website</u> for specific search terms
    - Clean data (remove ads, …)
- Do for all 110 tickers → get a lot of data ~ 10 * 300k.

- Run sentiment analyzer (parallelize for speed) on title and summary and aggregate over week:
    - Sentiment of each article → Mean/median over week
    - String together articles and get overall sentiment
- Use DistilBERT financial sentiment analyzer from Huggingface.

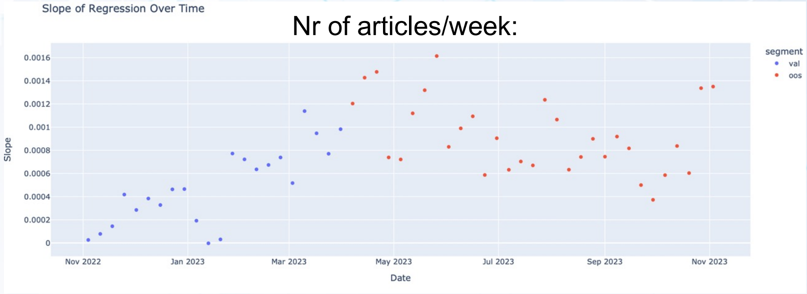
- Upshot: Get roughly ~5% correlation of oos residual predictions vs actual price for a single sentiment feature.



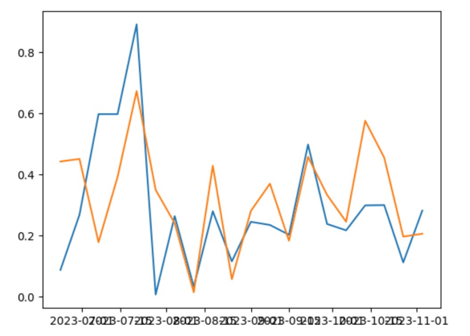


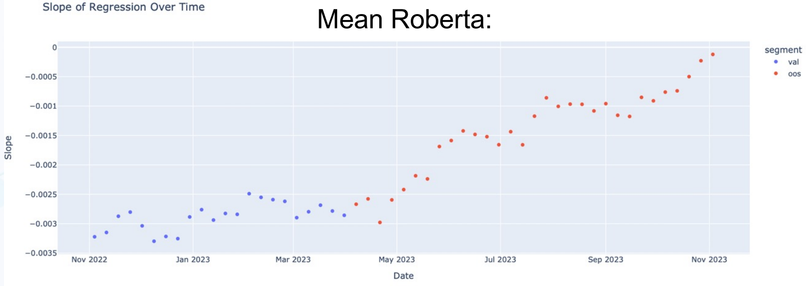
Factset Headline & Sentiments: Methodology

- Download factset news headlines per stock as pdf.
- Clean data.
- Run 2 (clsf, roberta) different sentiment analyzers and aggregate by week (same as before) → engineer sentiment score.
- Add additional feature: number of articles per week.
- Take small sample and compare vs full articles → high correlation → restrict to headlines only.



Full articles vs headlines for 6 months example:





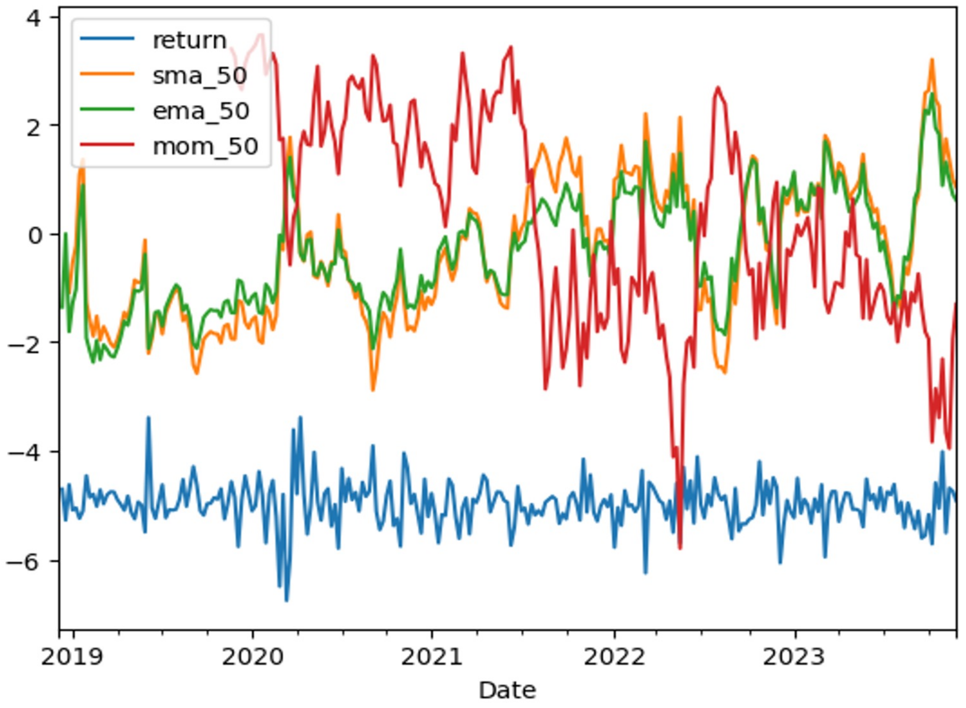
	Mean_roberta_diff1	Mean_clsf_diff1	Mean_roberta_agg_diff1	Mean_clsf_agg_diff1
Mean_roberta_diff1	1.000000	0.223829	0.824872	0.214711
Mean_clsf_diff1	0.223829	1.000000	0.227825	0.581925
Mean_roberta_agg_diff1	0.824872	0.227825	1.000000	0.227604
Mean_clsf_agg_diff1	0.214711	0.581925	0.227604	1.000000

Lessons: Other Features

Formulaic Alphas:
- Don't perform very strongly on backtesting
- Only one alpha made it in the final model

Price Movement:
- 50-week SMA/EMA/Momentum analysis found to be the most robust

Alpha#2: (-1 * correlation(rank(delta(log(volume), 2)), rank(((close - open) / open)), 6))



Lessons: Feature Selection

01 Individual Features

Backtesting individual features against future returns:

- Looking at consistency of the slope of the regression
- Given 99 features, this would be laborious

02 Mass Testing

Testing all features at once on two metrics:

- Whether average regression slope meets expectation based on theory
- Whether t-value is robust/significant

03 Correlation Matrix

Combinations of Features:

- Looking for paired combinations that are least correlated with each other
- Minimizing repetition of features selected from the same "group"

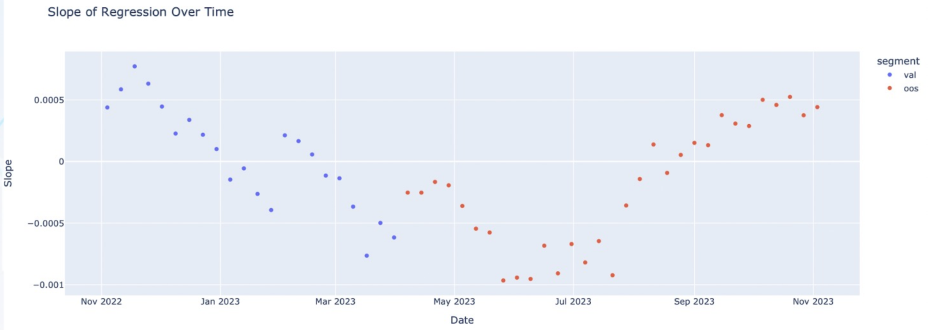
Lessons: Feature Selection

01 Individual Features

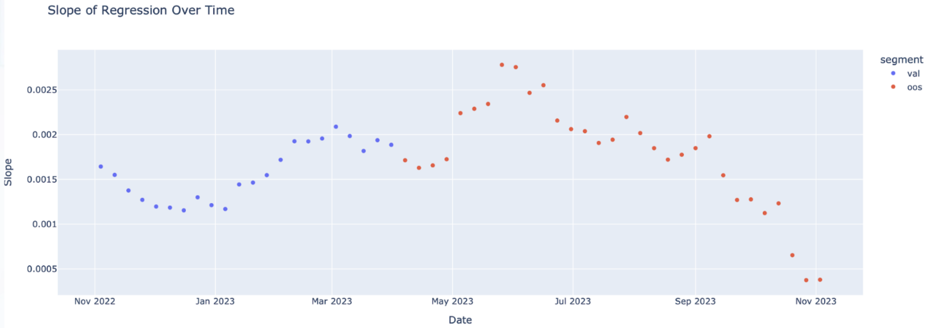
Backtesting individual features against future returns:

- Looking at consistency of the slope of the regression
- Given 99 features, this would be laborious

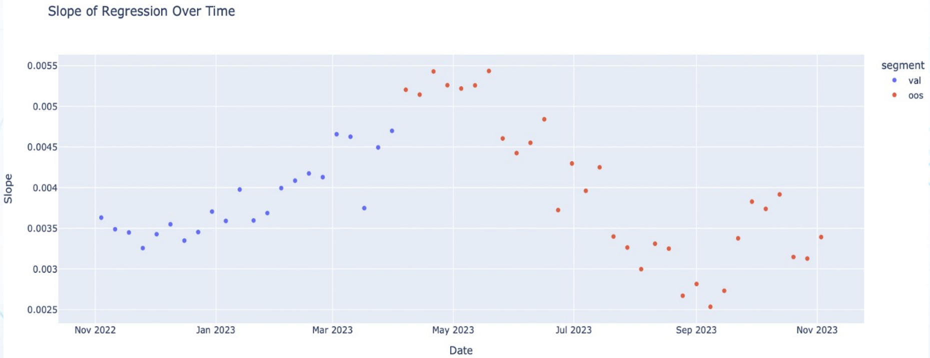
Alpha #4



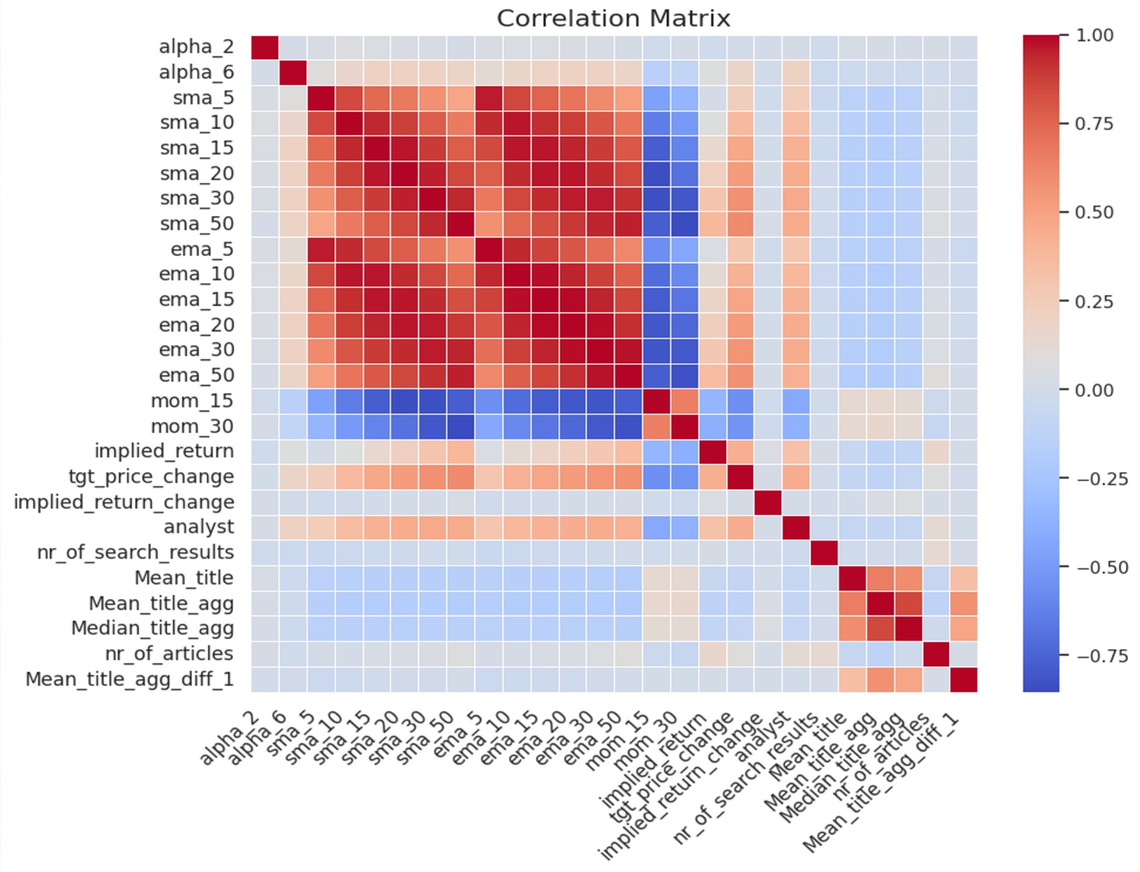
Alpha #2



EMA (50 Weeks)



Lessons: Feature Selection



03 Correlation Matrix

Combinations of Features:

- Looking for paired combinations that are least correlated with each other
- Minimizing repetition of features selected from the same "group"

Final model:

- More comprehensive feature selection.
- Fix backtested portfolio optimization for decision
- Select top 4 models found with around >52% mean correlation vs residual price.
- Grid search over threshold and model (btw ad-hoc, min-variance, and max-sharpe portfolio optimization)
  - Best 90 week average ~4.36
  - Found models that performed better recently (~12+ in last 11 weeks), but worse overall.

```
print(pd.DataFrame(dec_metric[20:]).mean())
print(pd.DataFrame(dec_metric[20:]).median())
print(pd.DataFrame(dec_metric[20:]).std())
```

```
0    1.22768
dtype: float64
0    1.411904
dtype: float64
0    16.332702
dtype: float64
```

```
print(pd.DataFrame(dec_metric[-11:]).mean())
print(pd.DataFrame(dec_metric[-11:]).median())
print(pd.DataFrame(dec_metric[-11:]).std())
```

```
0    12.040511
dtype: float64
0    5.54236
dtype: float64
0    24.470449
dtype: float64
```

Week 11 submission:

Features:

```
['LSTM_pred_for_following_week',
 'ema_50', 'mom_50', 'alpha_2',
 'tgt_price_change', 'analyst',
 'nr_of_articles', 'Mean_title_agg',
 'Median_roberta']
```

Correlation:

```
------------------------------------------------
Mean: 0.52, Median: 0.59, Std: 0.21
------------------------------------------------
```

Performance:

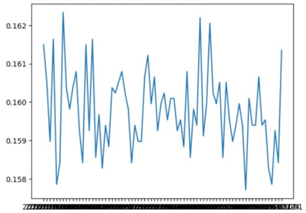
```
dec_metric, last_column = decision_opt(corr_threshold = 0.2
                                       rfr = 0.02,
                                       maximize_sharpe = False,
                                       verbose = False,
                                       verbose1 = False)
```

```
pd_dec_metric = pd.DataFrame(dec_metric)
print(f'Overall: Mean: {pd_dec_metric[10:].mean().values[0]
print(f'Recent:  Mean: {pd_dec_metric[-10:].mean().values[0

```

```
Overall: Mean: 4.36, Median: 3.34, Std: 13.56.
Recent:  Mean: 0.41, Median: 3.71, Std: 8.91.
```

```
forecast_perf    0.159794
dtype: float64 forecast_perf    0.159747
dtype: float64
[<matplotlib.lines.Line2D at 0x7b44f7e1f880>]
```



Future Directions/Highlights

01

Overall Methodology

- Automate weekly data generation: Currently generating features over multiple notebooks and combining them with another notebook

02

Data

- Generate more data for more thorough performance analysis
- Consider scraping Yahoo finance research reports
- Consider including more macro data such as unemployment etc.

03

Model Selection

- More comprehensive feature selection: Consider testing combinations of features
- Consider other forms of regression: Robust regression chosen now but can backtest against LASSO, Ridge, etc.
    - OLS vs Huber → relatively similar performance for the tests we performed.