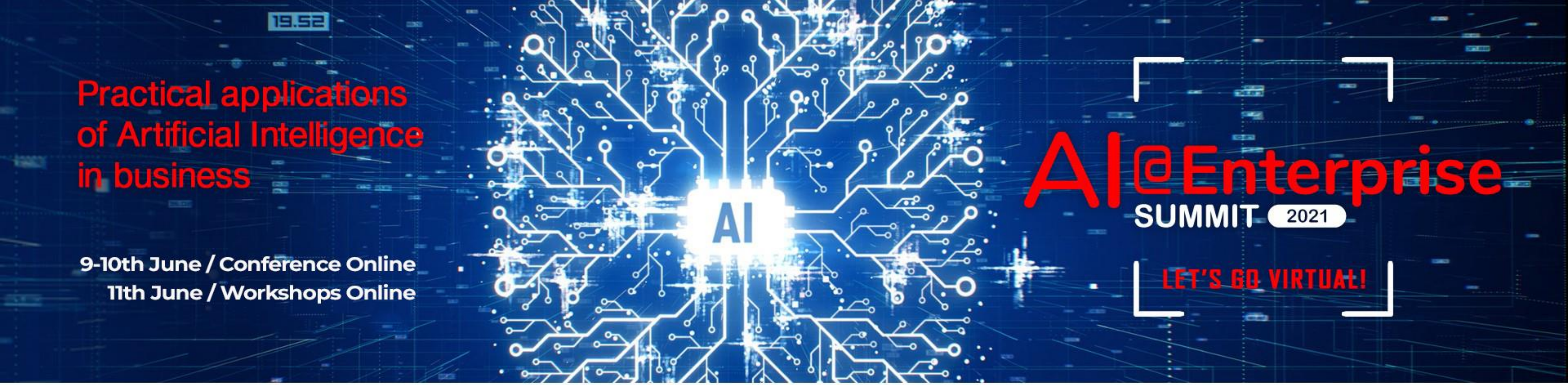


Stock Price Prediction and Portfolio Optimization Using Recurrent Neural Networks and Autoencoders

Julian Quernheim

Senior Consultant, BIVAL GmbH

https://github.com/QUER01/FinanceModule

https://www.linkedin.com/in/julianquernheim/

https://www.bival.de/en/jobs/

Introduction

Goal:

- Apply deep learning to beat traditional portfolio optimization methods.

Problem Statements:

1. Calculating an optimal stock portfolio allocation still **uses historic stock returns.**
2. Risk is quantified **without capturing non-linearities** of the timeseries.

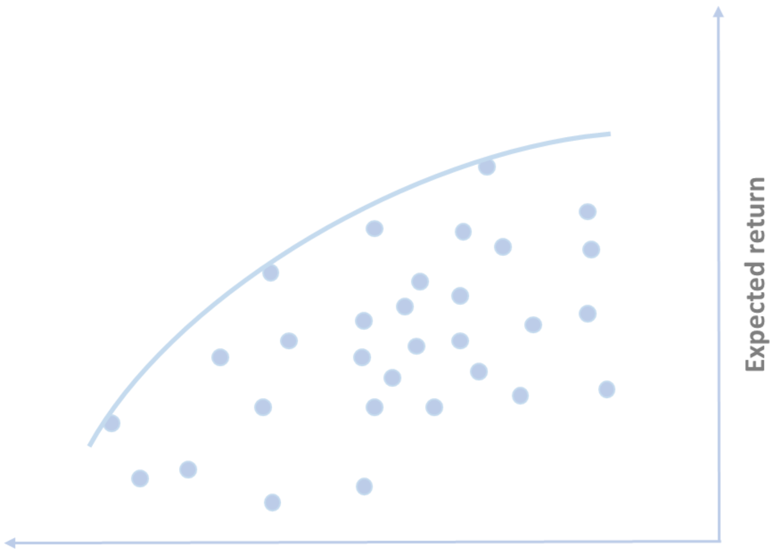
Requirements:

- A basic knowledge of deep learning and portfolio management is required.

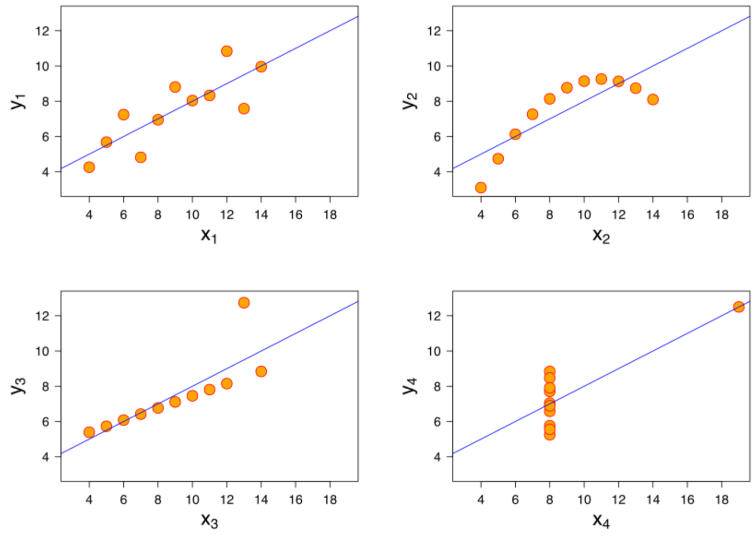
Introduction

Problem statements explained : What is wrong with how we calculate expected returns and risks?

Investor can construct a portfolio of multiple assets that will **maximize returns** (r_i) for a given level of portfolio risk, **<u>but no future predictions are considered</u>**.



The covariance indicates a **linear** relationship between two variables. Hence it can be fallacious in situations where two variables have a relationship, but it is **nonlinear**.



Anscombe's quartet: All four sets are identical when examined using simple summary statistics but vary considerably when graphed.

Introduction

Four steps to calculate your portfolio: Focus, forecast, clean and optimize.

1 Focus: Which stocks to analyze?
- **Focus on stocks that move the market to decrease computation time!**

2 Forecast: Does forecasting improve the portfolio?
- **Don’t forecast too far. A forecast is only a strong indicator.**

3 Clean: How to improve the risk calculation of a stock?
- **Try to capture non-linearities in the time series.**

4 Optimize: How to calculate an optimal portfolio?
- **Don’t trust the in-sample results. Look at the out-of-sample results.**

Literature Review

Literature Review

A short recap of what you probably already know!

Portfolio Optimization

(Markowitz, 1952)

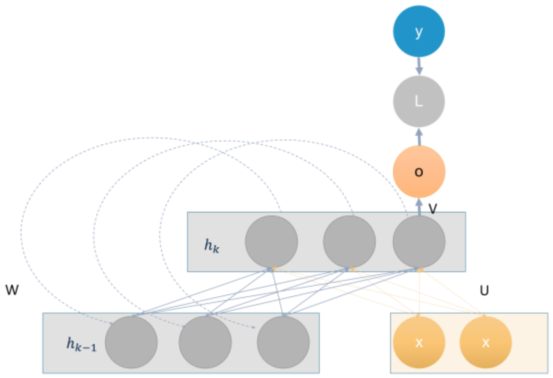
$$\text{minimize } C\, w^T w$$

$$s.t. \quad w^T \mu \geq \mu_b$$

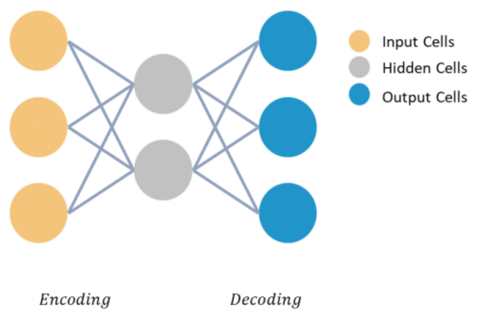
$$w^T \mathbf{1} = 1$$

$$w_i \geq 0$$

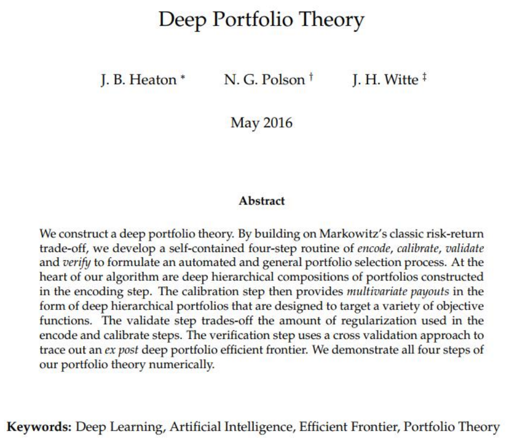
LSTMs (RNNs)
(Hochreiter, 1997)



Autoencoders
(Goodfellow et al., 2016)



Deep Portfolio
(Heaton et. al 2016)



Deep Portfolio Theory

J. B. Heaton * N. G. Polson † J. H. Witte ‡

May 2016

Abstract

We construct a deep portfolio theory. By building on Markowitz's classic risk-return trade-off, we develop a self-contained four-step routine of *encode, calibrate, validate* and *verify* to formulate an automated and general portfolio selection process. At the heart of our algorithm are deep hierarchical compositions of portfolios constructed in the encoding step. The calibration step then provides *multivariate payouts* in the form of deep hierarchical portfolios that are designed to target a variety of objective functions. The validate step trades-off the amount of regularization used in the encode and calibrate steps. The verification step uses a cross validation approach to trace out an *ex post* deep portfolio efficient frontier. We demonstrate all four steps of our portfolio theory numerically.

Keywords: Deep Learning, Artificial Intelligence, Efficient Frontier, Portfolio Theory

Data, Methodology & Results

Data, Methodology & Results

Focus, forecast, clean and optimize.

→ Autoencoder, LSTMs, Shrinkage Estimator and Linear Optimization

1 → **Apply an autoencoder model and filter stocks that can be recreated best.**

2 → **Forecast the next 10-days of a stock closing value into the future using Recurrent Neural networks.**

3 → **Apply latent features of an autoencoder model to clean the sample covariance matrix.**

4 → **Apply linear portfolio optimization and find the optimal stocks for the portfolio by optimizing the sharpe ratio.**

Datasets

Dateset 1 (only stocks):

- Stock exchanges: NYSE and NASDAQ
- Tickers: 5685 (only stocks)
- Range: 2014 -2018
- Final dataset: [1000 rows x 13925 columns]

	KOOL_Open	KOOL_High	KOOL_Low	KOOL_Close	KOOL_Volume	ADXS_Open	ADXS_High	ADXS_Low	ADXS_Close	ADXS_Volume
2018-07-13T00:00:00.000000000	0.47000	0.51000	0.44000	0.46000	414300.00000	1.60000	1.62000	1.30000	1.34000	6967800.00000
2018-07-16T00:00:00.000000000	0.51000	0.52000	0.47000	0.48000	1027000.00000	1.34000	1.51000	1.31000	1.44000	2091200.00000
2018-07-17T00:00:00.000000000	0.50000	0.50000	0.43000	0.43000	843200.00000	1.44000	1.47500	1.36000	1.41000	1318600.00000
2018-07-18T00:00:00.000000000	0.43000	0.45000	0.42000	0.43000	301500.00000	1.40000	1.46000	1.37000	1.43000	560700.00000
2018-07-19T00:00:00.000000000	0.45000	0.45000	0.41000	0.42000	219400.00000	1.44000	1.49000	1.41100	1.45000	693500.00000
2018-07-20T00:00:00.000000000	0.42000	0.43000	0.42000	0.42000	85800.00000	1.46000	1.49000	1.42000	1.48000	533700.00000
2018-07-23T00:00:00.000000000	0.42000	0.43000	0.41000	0.42000	87700.00000	1.48000	1.62000	1.47000	1.59000	978100.00000
2018-07-24T00:00:00.000000000	0.42000	0.43000	0.41000	0.42000	65300.00000	1.62000	1.64000	1.48000	1.51000	568000.00000
2018-07-25T00:00:00.000000000	0.42000	0.42000	0.41000	0.42000	158500.00000	1.51000	1.52000	1.46100	1.50000	201700.00000
2018-07-26T00:00:00.000000000	0.42000	0.42000	0.39000	0.40000	285400.00000	1.48000	1.50000	1.38000	1.40000	663700.00000
2018-07-27T00:00:00.000000000	0.42000	0.42000	0.39000	0.40000	169800.00000	1.40000	1.43000	1.36000	1.38000	452200.00000
2018-07-30T00:00:00.000000000	0.41000	0.42000	0.38000	0.40000	108200.00000	1.50000	1.50000	1.40000	1.43000	369900.00000
2018-07-31T00:00:00.000000000	0.40000	0.41000	0.38000	0.38000	120800.00000	1.42000	1.48000	1.41000	1.46000	332400.00000
2018-08-01T00:00:00.000000000	0.37000	0.39000	0.36000	0.38000	148100.00000	1.45000	1.49000	1.41000	1.43000	134400.00000
2018-08-02T00:00:00.000000000	0.38000	0.40000	0.37000	0.38000	101200.00000	1.42000	1.44000	1.33000	1.38000	422300.00000
2018-08-03T00:00:00.000000000	0.37000	0.40000	0.37000	0.39000	67000.00000	1.42000	1.42000	1.35000	1.41000	205000.00000

Dateset 2 (only ETFs):

- Stock exchanges: Frankfurt
- Tickers: 1098
- Range: 2020-current
- Final dataset: [1000 rows x 4318 columns]

	H4ZR.DE_Open	H4ZR.DE_High	H4ZR.DE_Low	H4ZR.DE_Close	H4ZR.DE_Volume	UIMP.DE_Open	UIMP.DE_High	UIMP.DE_Low	UIMP.DE_Close	UIMP.DE_Volume
2020-04-27T00:00:00.000000000	12.95200	12.95200	12.95200	12.95200	0.00000	108.98000	110.08000	108.94000	110.08000	278.00000
2020-04-28T00:00:00.000000000	12.95200	12.95200	12.95200	12.95200	0.00000	110.58000	111.76000	109.96000	110.48000	9466.00000
2020-04-29T00:00:00.000000000	12.95200	12.95200	12.95200	12.95200	0.00000	110.86000	111.76000	110.56000	111.76000	27074.00000
2020-04-30T00:00:00.000000000	14.04600	14.04600	13.54000	13.54000	5.00000	112.42000	112.42000	109.98000	109.98000	42671.00000
2020-05-01T00:00:00.000000000	14.04600	14.04600	13.54000	13.54000	5.00000	112.42000	112.42000	109.98000	109.98000	42671.00000
2020-05-04T00:00:00.000000000	13.54000	13.54000	13.54000	13.54000	0.00000	106.30000	106.86000	105.72000	106.74000	7558.00000
2020-05-05T00:00:00.000000000	13.54000	13.54000	13.54000	13.54000	0.00000	108.50000	110.02000	108.50000	110.02000	416.00000
2020-05-06T00:00:00.000000000	13.52600	13.55600	13.42200	13.42200	8.00000	110.20000	110.58000	109.36000	109.36000	12791.00000
2020-05-07T00:00:00.000000000	13.42200	13.42200	13.42200	13.42200	0.00000	109.48000	110.64000	109.48000	110.62000	2703.00000
2020-05-08T00:00:00.000000000	13.67200	13.67200	13.63400	13.64000	1000.00000	111.20000	111.56000	111.10000	111.56000	212.00000
2020-05-11T00:00:00.000000000	13.64000	13.64000	13.64000	13.64000	0.00000	112.58000	112.66000	111.00000	112.00000	1599.00000
2020-05-12T00:00:00.000000000	13.64000	13.64000	13.64000	13.64000	0.00000	111.60000	112.38000	111.60000	111.68000	2761.00000
2020-05-13T00:00:00.000000000	13.51200	13.53600	13.31800	13.31800	7471.00000	109.48000	109.92000	108.54000	108.54000	4503.00000
2020-05-14T00:00:00.000000000	13.31800	13.31800	13.31800	13.31800	0.00000	108.18000	108.32000	107.12000	107.62000	3702.00000
2020-05-15T00:00:00.000000000	13.30200	13.32800	13.15200	13.20600	500.00000	109.70000	109.70000	107.34000	108.88000	25174.00000
2020-05-18T00:00:00.000000000	13.38000	13.67400	13.37400	13.67400	9626.00000	111.16000	113.32000	111.16000	113.04000	14313.00000

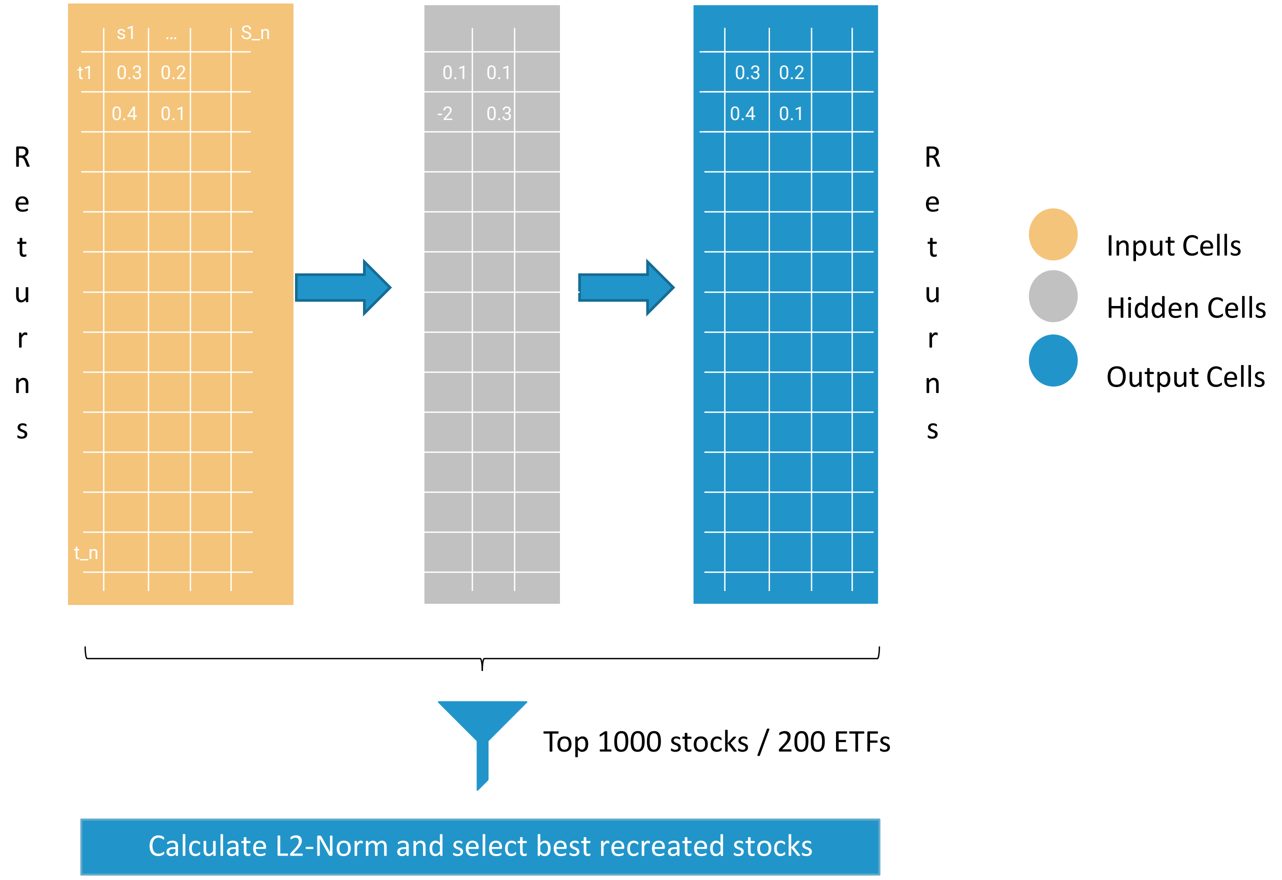
Transformed stock datasets

1 Focus: Which stocks to analyze?

Focus on stocks that move the market!

Intuition:

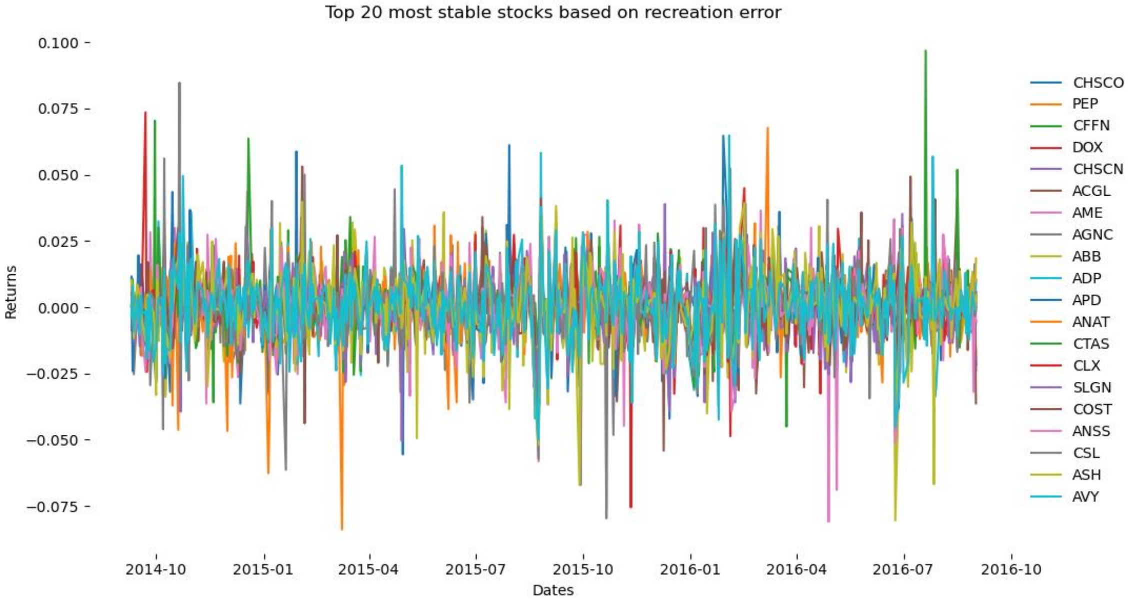
The stocks with the lowest recreation error (L2-norm) represent the market better. They are less volatile and are considered to be similar to large cap stocks.

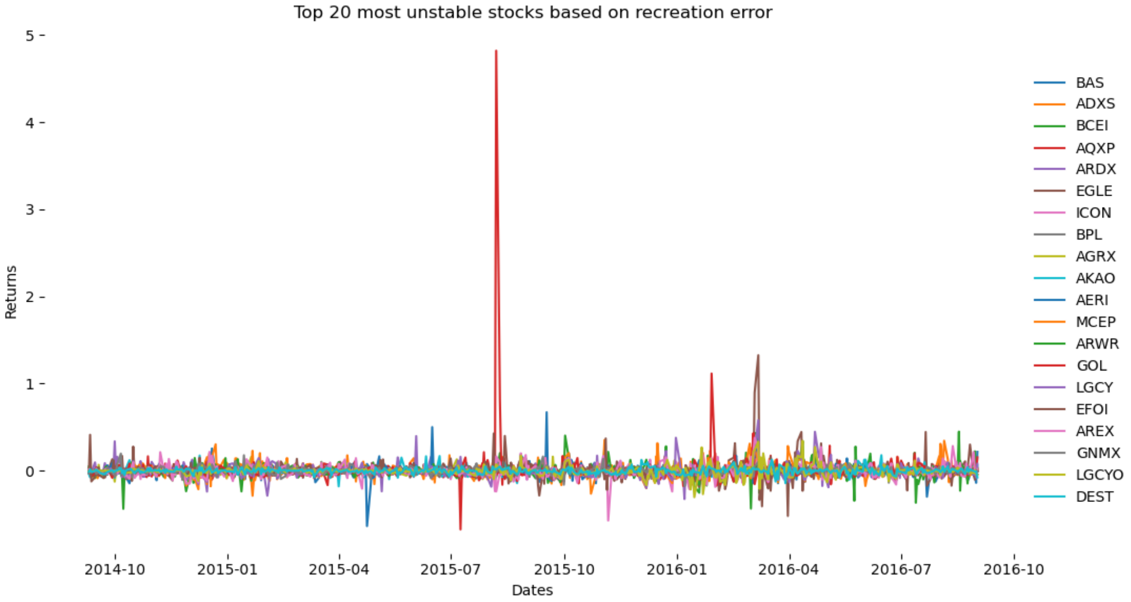


Autoencoder model with ranked recreation error.

1 Focus: Which stocks to analyze?

Unstable stocks tend to be more volatile and have more unexpected spikes!

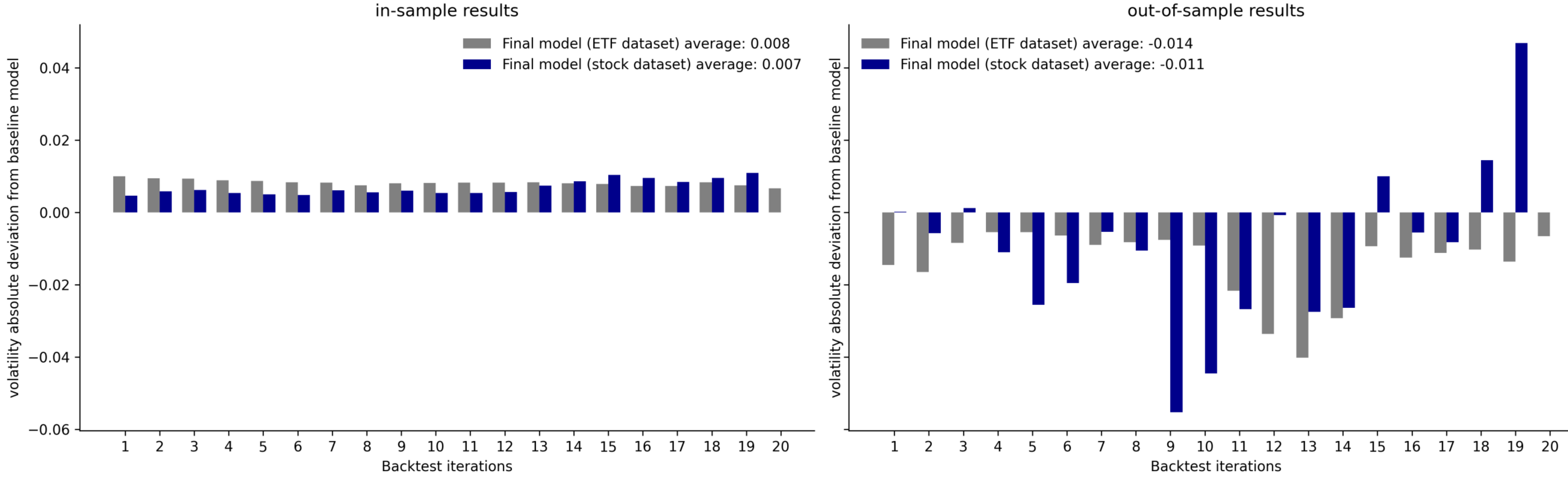




Top 20 most stable and most unstable stocks ranked by recreation error.

Focus: Which stocks to analyze?

Filtering based on recreation error decreases out-of sample volatility!



In-sample and out-of-sample volatility deviation for ETFS and stocks
baseline = full dataset
challenger = filtered dataset

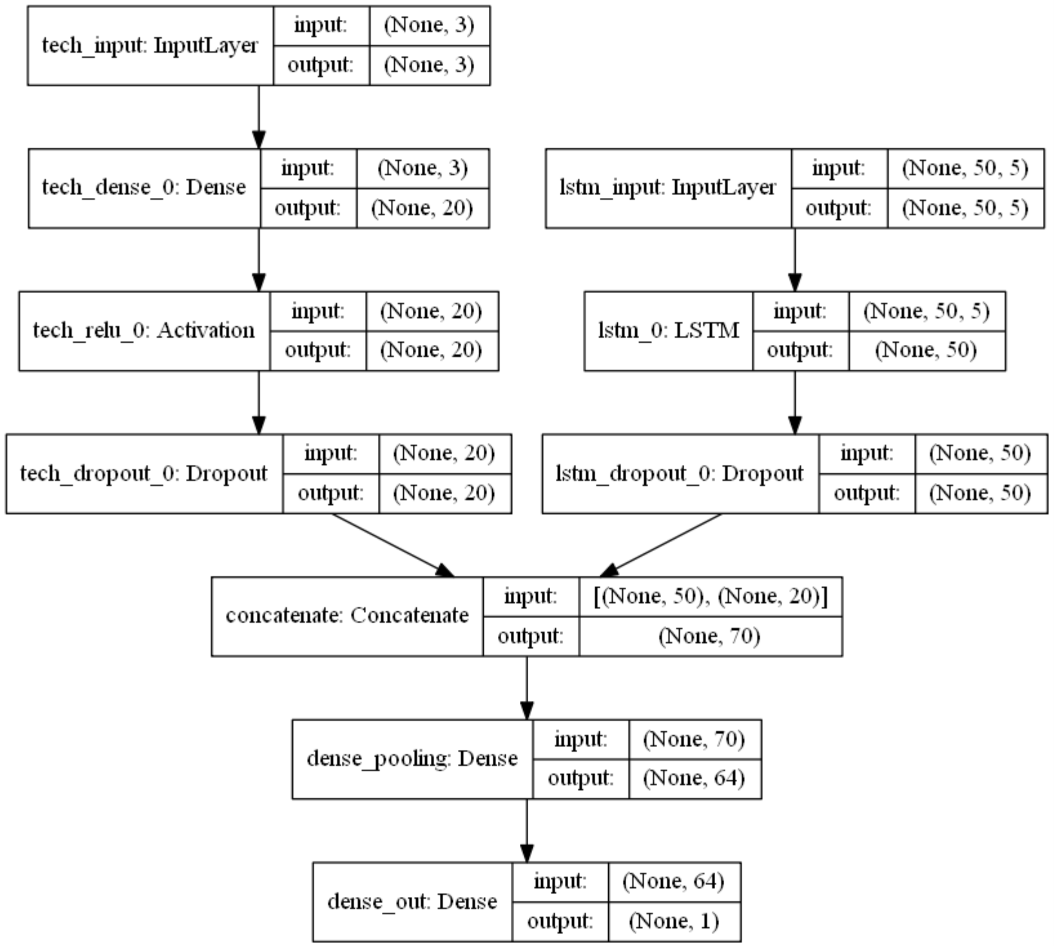
2 Forecast: Does forecasting improve the portfolio?

Using a multi-input model is a good way to improve accuracy!

Model Design:

A multi-input model has been applied using Keras functional API, to include:

- historic stock prices (ohlcv)
- additional technical indicators e.g. exponential moving average



Keras RNN model.

2 Forecast: Does forecasting improve the portfolio?

Don‘t forecast too far. RNNs do a great job at forecasting timeseries data!

RNN model results fit on entire dataset with 10-days out-of-sample forecast.

③ Clean: How to improve the risk calculation of a stock?

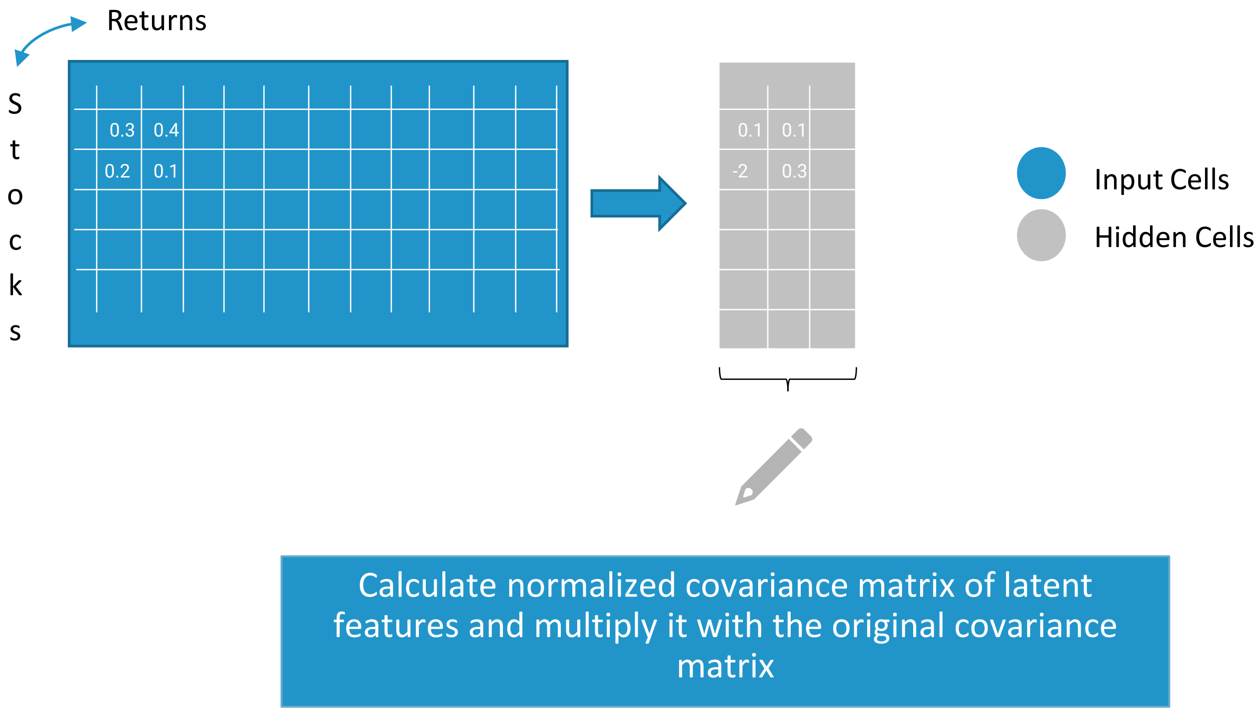
Latent features catch non-linearities and can be used to improve the sample covariance matrix!

- We transpose the input matrix and get a compressed time series in form of latent features.
- Calculating the normalized covariance of the latent feature vectors B, we are able to use this as a shrinkage estimator.

$$\hat{C} = B * C$$

Intuition:

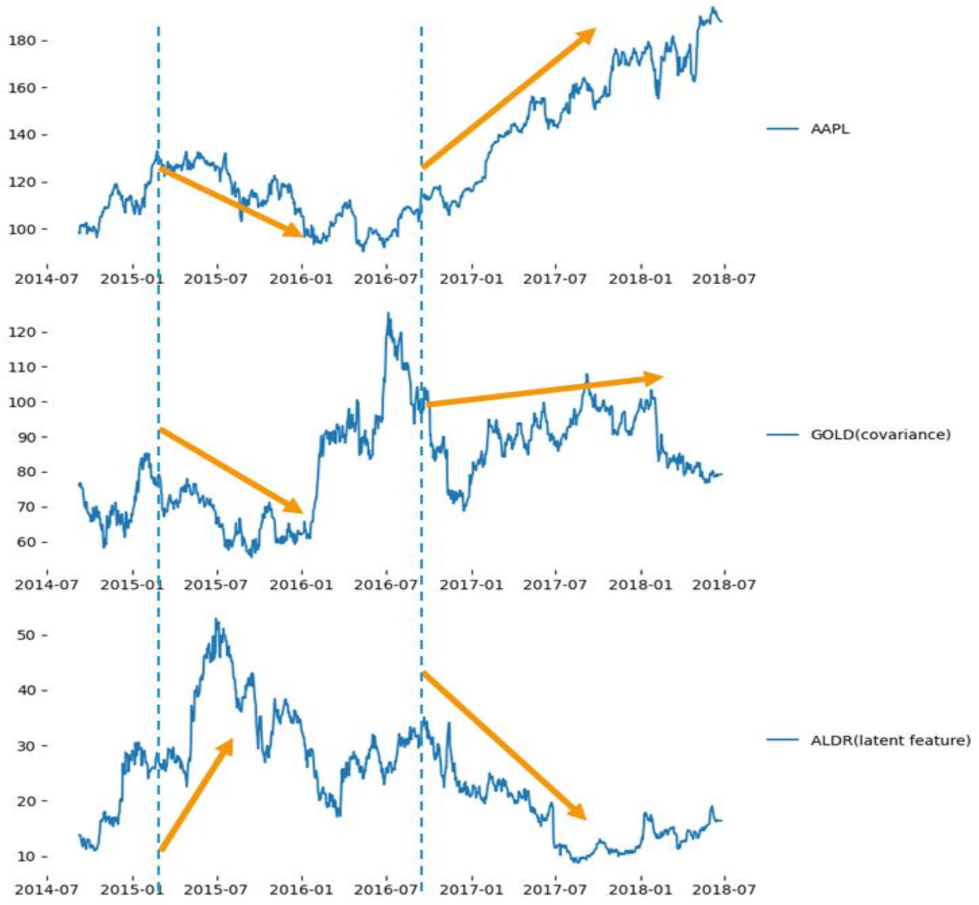
Using the adjusted covariance matrix better captures non-linearities.

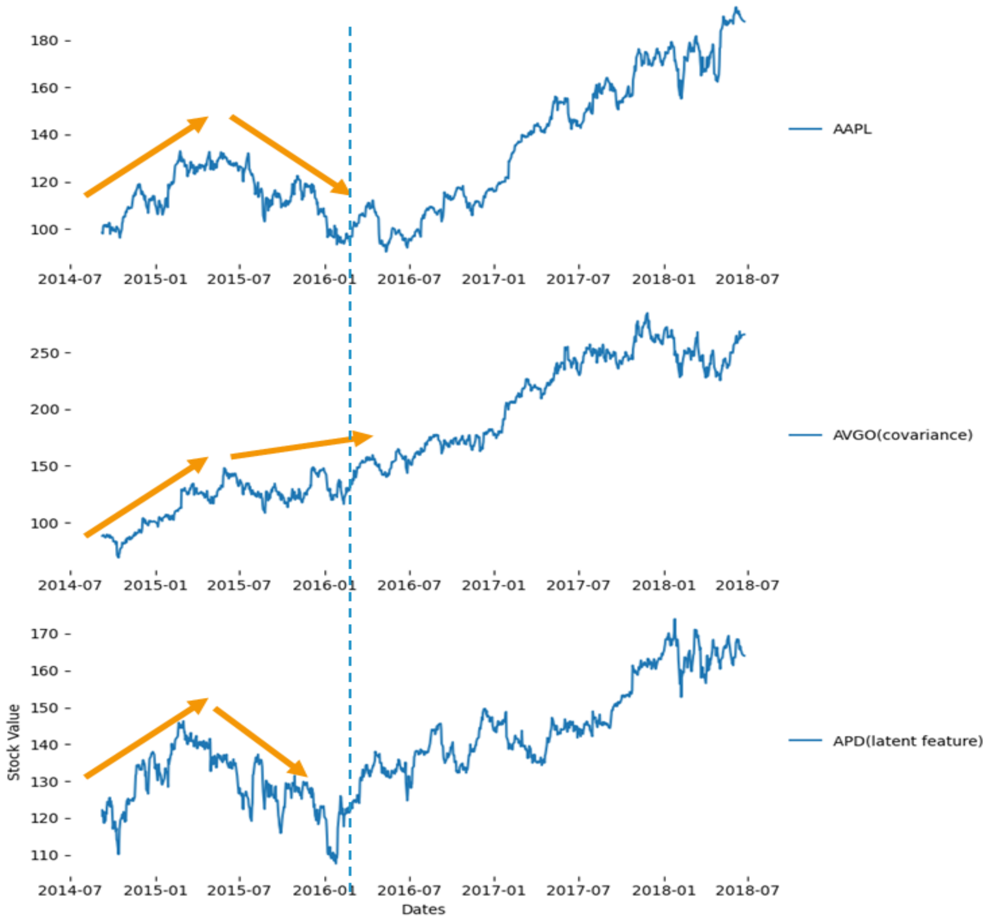


Autoencoder model with calculated covariance of latent features.

3 Clean: How to improve the risk calculation of a stock?

Latent features catch non-linearities and can be used to improve the sample covariance matrix!



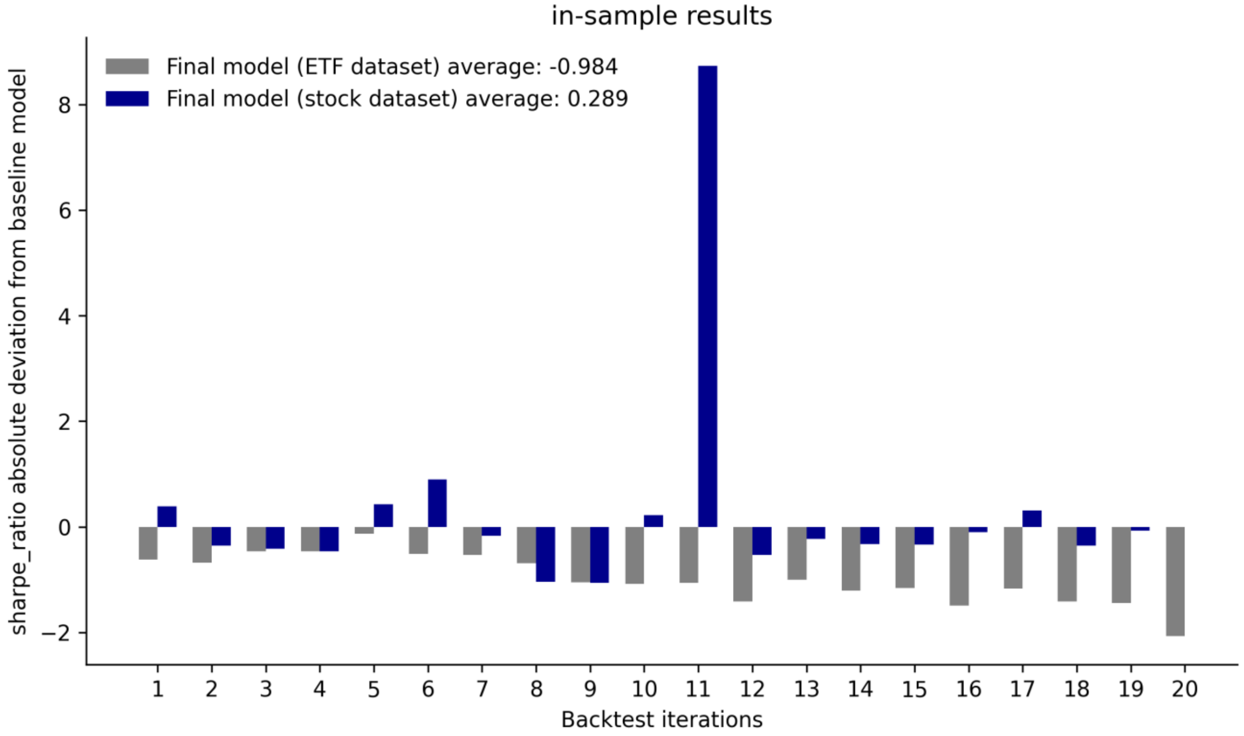


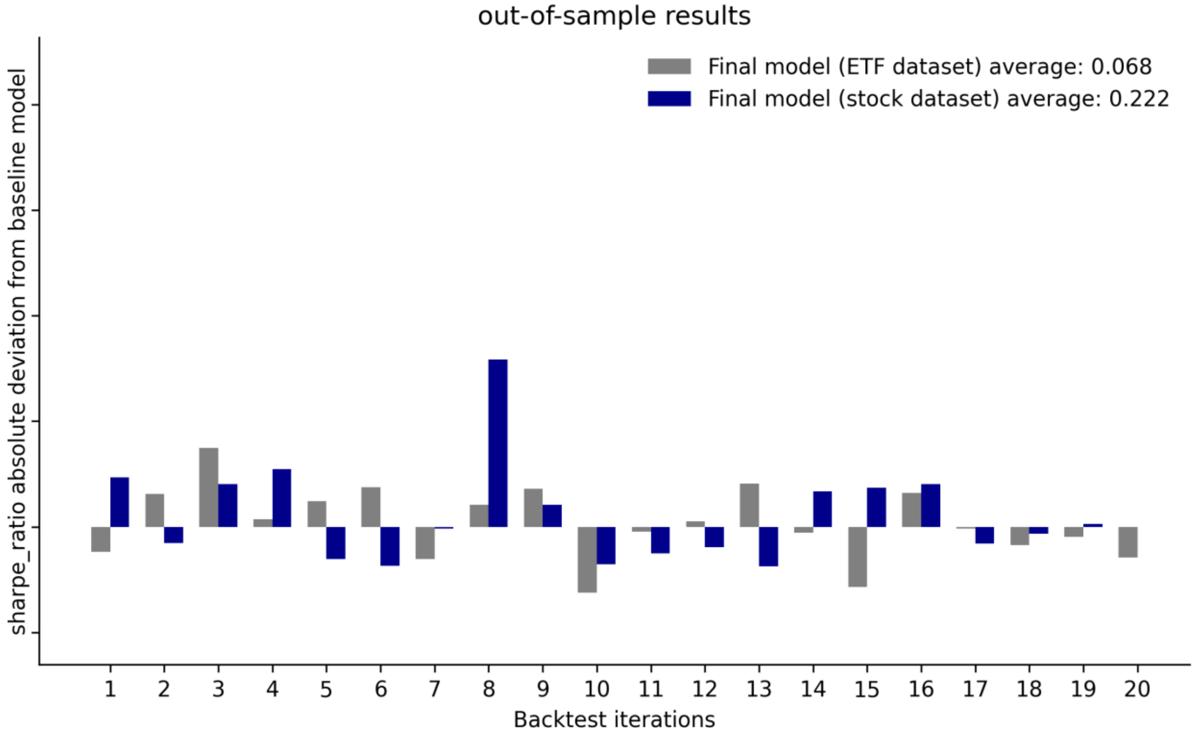
Baseline stock : APPL (Apple) compared to least (left) and most (right) related stocks

2 3 4

Optimize: How to calculate an optimal portfolio?

In-Sample results look good stocks only, out-of sample results improve the sharpe ratio for both datasets.





In-sample and out-of-sample sharpe ratio deviation for ETFS and stocks
baseline = filtered dataset
challenger = forecasted and cleaned dataset

Conclusion and Future Research

Takeaways

1 Choosing your subset wisely helps to avoid taking a sledgehammer to crack a nut.

2 Don't forecast too far!

3 Don't use a linear method on data that is non-linear in nature!

4 Getting the best out-of sample results is a tough nut to crack!

References

- J. B. Heaton, N. G. Polson, & J. H. Witte. (2016). Deep Portfolio Theory.
- Werbos, P. (1990): Backpropagation Through Time: What It does and How to Do It. https://doi.org/10.1109/5.58337 Retrieved from http://axon.cs.byu.edu/~martinez/classes/678/Papers/Werbos_BPTT.pdf
- https://towardsdatascience.com/getting-rich-quick-with-machine-learning-and-stock-market-predictions-696802da94fe
- Olivier Ledoit & Michael Wolf, 2003. "**Honey, I shrunk the sample covariance matrix**," Economics Working Papers 691, Department of Economics and Business, Universitat Pompeu Fabra.

Q & A

Yes, my code is on github!

Add me on LinkedIn!

Looking for a new job?

https://github.com/QUER01/FinanceModule

https://www.linkedin.com/in/julianquernheim/

BIVAL® https://www.bival.de/en/jobs/