AMProject_Clean

Laura Gullicksen, Erich Gozebina, Daria Palitzsch

23/05/2025

1. Introduction

#TODO: describe data choice -> Laura

2. Data & Descriptive Analysis

Data Aggregation and Strategy Frequency

The raw dataset provides CHAINLINK price data at *hourly frequency*. While such high-frequency data offers more granular insights, we chose to **aggregate the data to daily frequency** for the following reasons:

1. **Alignment with Trading Strategy**: Our core trading strategy is based on a **7-day momentum signal**, which inherently reflects **weekly price trends**. Applying such a signal at an hourly resolution would not be consistent with the strategy's time horizon.
2. **Noise Reduction**: Hourly crypto data can be highly volatile and noisy. Aggregating to daily returns reduces **microstructure noise, short-term reversals**, and **Whale-driven price spikes**, improving the signal-to-noise ratio.
3. **Practical Execution Perspective**: A strategy that rebalances daily is **more realistic to implement**, considering gas fees, latencyn and operational constraints on decentralized exchanges or CEX APIs.
4. **Interpretability and Robustness**: Daily returns are more interpretable and robust across backtests. Most financial and technical indicators (e.g., RSI, MACD, SMA) are commonly applied on daily charts.

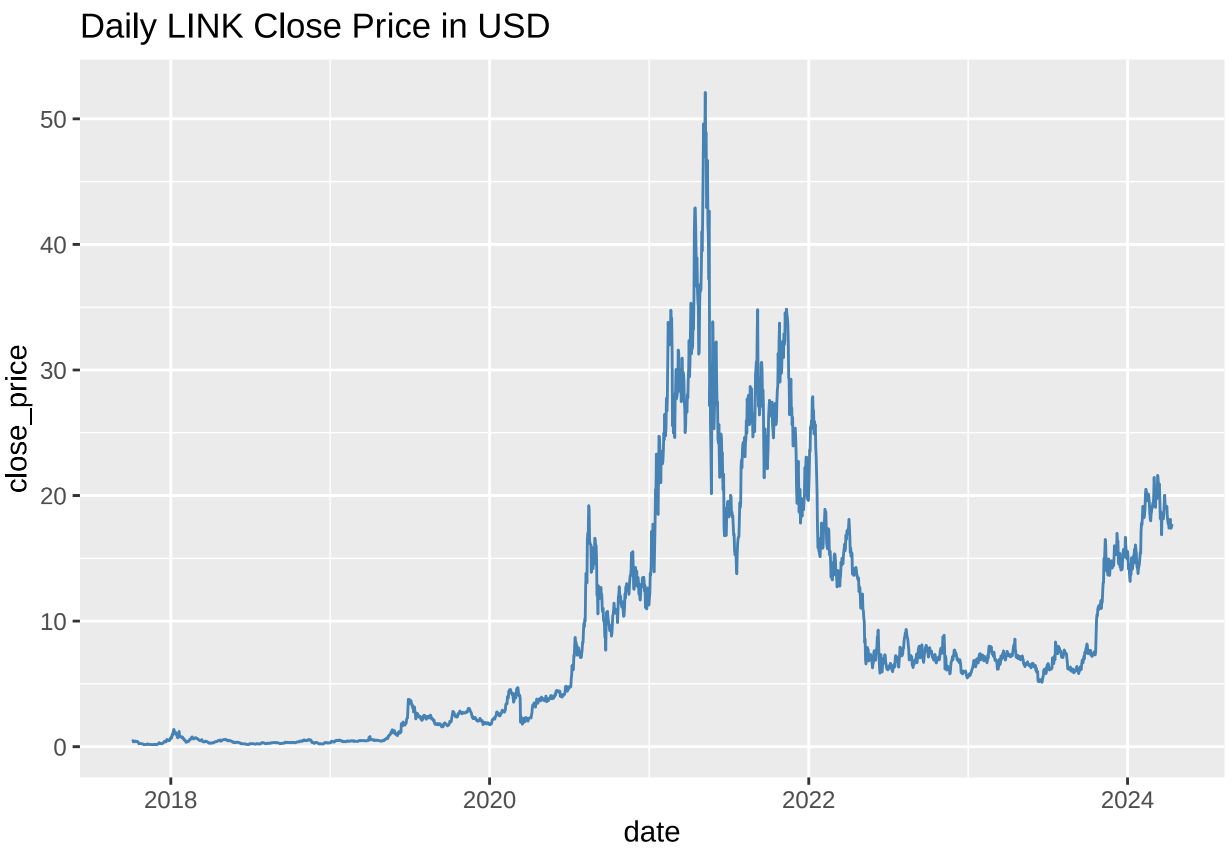
Our strategy issues long/short signals based on the past 7-day return of CHAINLINK, i.e.,

$$\text{Momentum}_t^{(7)} = \log\left(\frac{P_t}{P_{t-7}}\right)$$

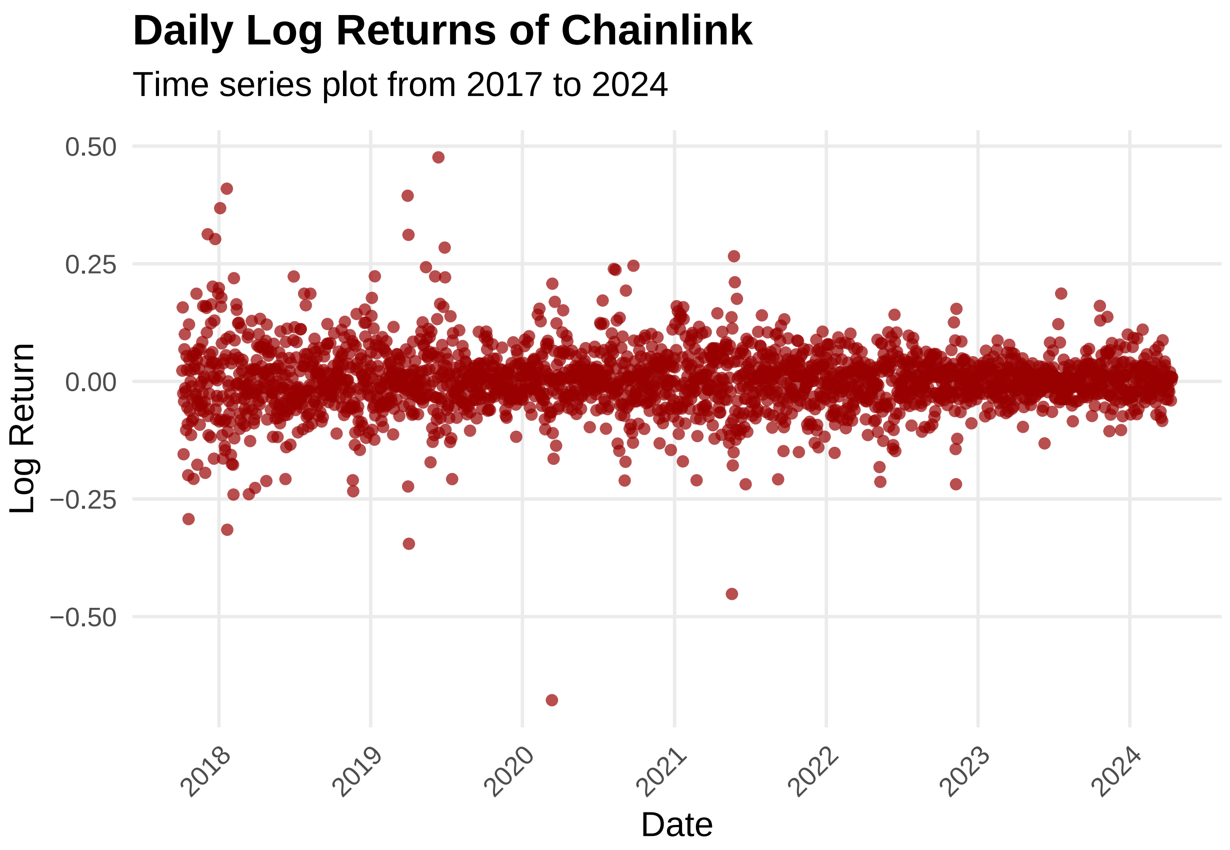
This naturally assumes daily data, as each observation reflects the cumulative return over the previous seven days.

In summary, aggregating to daily frequency is a theoretically and practically sound choice. It ensures consistency between our signal construction, model estimation, and backtesting logic.

```
## # A tibble: 6 x 9
##   date       close_price open_price high_price low_price log_return
##   <date>           <dbl>      <dbl>      <dbl>     <dbl>      <dbl>
## 1 2017-10-04       0.397      0.343      0.399     0.337    NA
## 2 2017-10-05       0.407      0.397      0.418     0.378     0.0228
## 3 2017-10-06       0.476      0.406      0.485     0.404     0.157
## 4 2017-10-07       0.464      0.472      0.476     0.424    -0.0258
## 5 2017-10-08       0.397      0.462      0.473     0.358    -0.155
## 6 2017-10-09       0.381      0.405      0.440     0.357    -0.0426
## # i 3 more variables: log_open_return <dbl>, log_high_return <dbl>,
## #   log_low_return <dbl>
```



```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```



#DONE: Explain output and log choice -> Erich

Having a first glance on the LINK's price data, we see that there is not much movement until 2020, followed by sharp increases by factor ten in the consecutive nine months. The price reaches its peak of over 50$/LINK in May 2021. In the second half of 2021 until April 2022 the data shows volatile behavior but with significant decrease on average. A trendless period of relatively low volatility, starting in May 2022 and ending in October 2023, shows prices between 5 and 10$/LINK. Finally, we observe another increase at the end of 2023 and beginning of 2024.

Moving away from non-stationary price data to stationary log returns, we observe a higher dispersion in the earlier years, suggesting a higher volatility then. Furthermore, most of points cluster around zero, indicating that there is no significant long term drift. Outliers, both positive and negative, imply extreme relative price movements, especially in the earlier period. Another important insight is the changing variance since the density of the points increases in the second half of the time window.

Why did we chose log returns over canonical (arithmetic) returns? Using log returns instead of canonical returns is a standard practice in financial econometrics and modeling.

$$\tilde{r}_t = \log(r_t + 1) = \log\left(\frac{p_t}{p_{t-1}}\right)$$

The underlying reason is the assumption that prices of an financial asset are log-normally distributed. This is reasonable since the log-normal distribution does not allow for negative values, which is also true for most asset prices (particularly for crypto currencies). Moreover, historical data provides evidence that the log-normal distribution gives a good fit for the prices of many financial assets. In reverse, since the logarithm function amplifies returns that are close to -1 more than positive returns, log-returns are distributed more symmetrically than canonical returns and indeed follow a normal distribution. Additionally, if returns are small, log returns approximate canonical returns very well. For x close to zero, it holds that

$$\log(x+1) \approx x.$$

We can expect small returns since we shorten the considered time interval. Another important property is the additivity of log returns. It allows us to aggregate returns over multiple periods by summing up the pointwise log returns - a property that canonical returns miss. These properties make log returns more suitable for linear regression models, hypothesis testing, and machine learning regressors.

To better understand the characteristics of the Chainlink price and return series, we compute a set of descriptive statistics based on the daily close prices and the corresponding log returns. These statistics provide a first impression of the dataset's distribution, dispersion, and extreme values, and help assess whether further preprocessing or transformation steps are necessary before applying predictive models.

Table 1: Summary Statistics for Chainlink Price and Returns

Statistic	Value
Number of Observations	2,383.0000
Mean Close Price	9.1484
Std. Dev. Close Price	9.5407
Minimum Close Price	0.1453
Maximum Close Price	52.1000
Mean Return	0.0016
Std. Dev. Return	0.0676
Minimum Return	-0.6776
Maximum Return	0.4762

The summary statistics reveal that the mean daily log return of Chainlink is close to zero, while the standard deviation is relatively high, reflecting the well-known volatility of cryptocurrency markets. The minimum and maximum returns further highlight the presence of large price swings. The wide range between the minimum and maximum close prices illustrates the strong appreciation potential, but also the riskiness of the asset over the observation period.

To evaluate the temporal dependence structure of Chainlink's daily log returns, we plot the autocorrelation function (ACF). The ACF helps determine whether past returns exhibit statistically significant correlation with future returns — a key consideration when assessing the potential for return predictability.

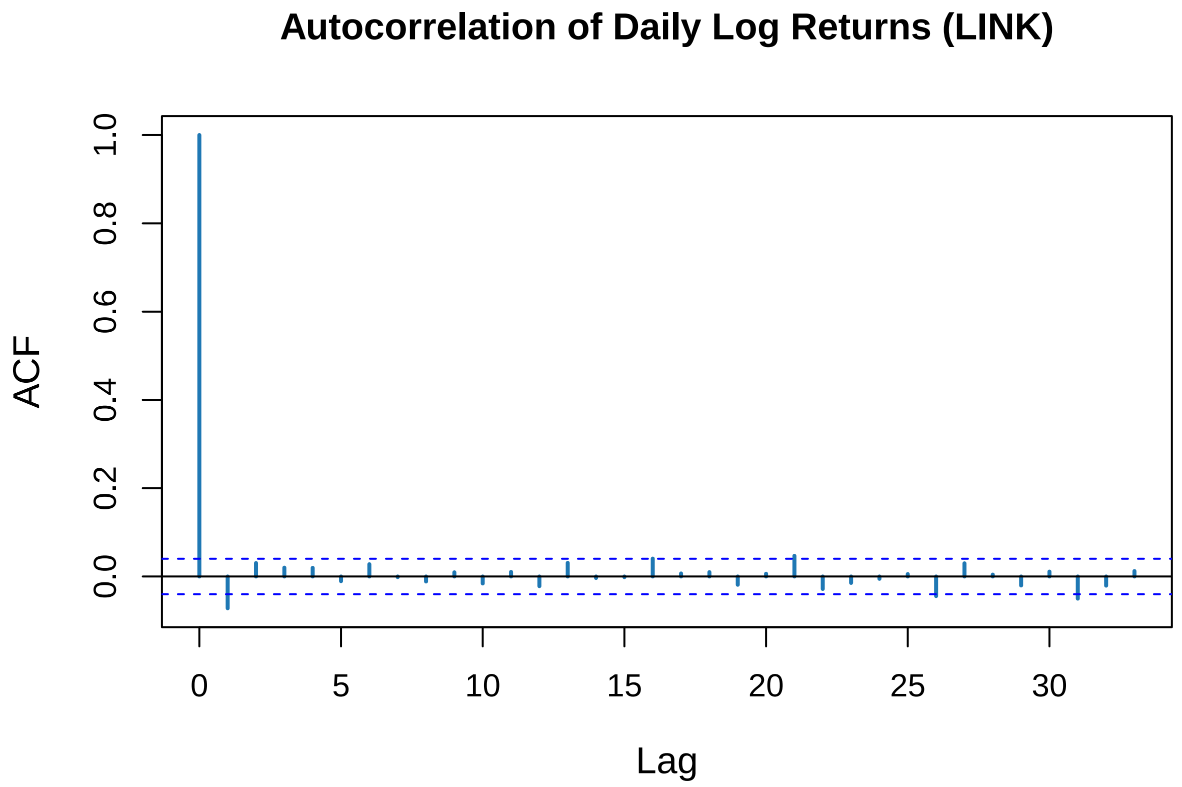


Figure 1: ACF of Daily Log Returns for Chainlink

The autocorrelation function (ACF) of daily log returns shows no statistically significant linear dependence at any lag, indicating that past returns do not linearly predict future returns. This finding supports the weak-form Efficient Market Hypothesis (EMH). However, it does not rule out the presence of exploitable patterns captured by non-linear or directional indicators. Therefore, we proceed with a momentum-based trading strategy, leveraging the sign of multi-day past returns to generate long or short signals.

3. Standard Model

#Momentum Signal Strategy

We define the 7-day momentum as the log return over the past 7 days:

$$\text{Momentum}_t = \log\left(\frac{P_t}{P_{t-7}}\right)$$

The trading signal is then determined as:

$$\text{Signal}_t = \begin{cases} +1 & \text{if Momentum}_t > 0 \quad \text{(go long)} \\ -1 & \text{if Momentum}_t < 0 \quad \text{(go short)} \\ 0 & \text{otherwise (no position)} \end{cases}$$

The strategy return is computed as:

$$r_{t+1}^{\text{strategy}} = \text{Signal}_t \cdot r_{t+1}$$

where $r_{t+1} = \log\left(\frac{P_{t+1}}{P_t}\right)$ is the daily log return.

#TODO: insert standard model with momentum -> Erich

Table 2: Regression Results: 7-Day Momentum Strategy

	Dependent variable:
	Strategy Return
Intercept	0.0052
	(0.0079)
7-Day Momentum	0.0023*
	(0.0014)
Observations	2,376
R^2	0.0002
Adjusted R^2	-0.0002
Residual Std. Error	0.0675 (df = 2374)
F Statistic	0.4294 (df = 1; 2374)
Note:	*p<0.1; **p<0.05; ***p<0.01

#TODO: explain result of standard 7 day momentum strategy -> Laura

4. Extension

Extension of our OLS

To enhance the predictive power of the benchmark model, we extend it by incorporating a broader set of explanatory variables that capture not only short- and medium-term price dynamics, but also market sentiment, technical indicators, and inter-asset relationships. These include:

- Momentum indicators over 3, 7, and 14 days,
- Lagged daily returns (1-day and 2-day),
- A 7-day rolling volatility measure,
- Technical indicators such as the 14-day Relative Strength Index (RSI), MACD value and histogram, Simple Moving Average difference, and Average True Range (ATR),
- Day-of-week dummy variables to capture potential calendar effects,

- BTC-based predictors: daily BTC return, 7-day BTC momentum, and 7-day BTC volatility,
- ETH-based predictors: daily ETH return, 7-day ETH momentum, and 7-day ETH volatility,
- ETH trading volume: daily ETH volume return, 7-day ETH volume momentum, and 7-day ETH volume volatility,
- ETH market capitalization: daily ETH market capitalization return, 7-day ETH market capitalization momentum, and 7-day ETH market capitalization volatility,
- Ethereum gas fees: daily gas return, 7-day gas momentum, and 7-day gas volatility.

The extended predictive regression model is specified as:

$$r_{t+1} = \alpha + \sum_{h\in\{3,7,14\}} \beta_h \cdot \text{Momentum}_t^{(h)} + \gamma_1 \cdot r_t + \gamma_2 \cdot r_{t-1} + \delta \cdot \text{Volatility}_t^{(7)} + \sum_j \theta_j \cdot X_t^{(j)} + \varepsilon_{t+1}$$

where $X_t^{(j)}$ represents the set of technical indicators (RSI, MACD, ATR, SMA), weekday dummies, and BTC-based predictors.

$$\begin{aligned}
r_{t+1} &:= \log\left(\frac{P_{t+1}}{P_t}\right) \quad \text{(one-day-ahead LINK return)} \\
\text{Momentum}_t^{(h)} &:= \log\left(\frac{P_t}{P_{t-h}}\right) \quad \text{for } h \in \{3,7,14\} \\
\text{Volatility}_t^{(7)} &:= \text{std}\left(r_{t-6},\ldots,r_t\right) \\
\text{BTC return}_t &:= \log\left(\frac{P_t^{\text{BTC}}}{P_{t-1}^{\text{BTC}}}\right) \\
\text{BTC Momentum}_t^{(7)} &:= \log\left(\frac{P_t^{\text{BTC}}}{P_{t-7}^{\text{BTC}}}\right) \\
\text{BTC Volatility}_t^{(7)} &:= \text{std}\left(r_{t-6}^{\text{BTC}},\ldots,r_t^{\text{BTC}}\right)
\end{aligned}$$

The ETH-based predictors are constructed analogously to the BTC-based predictors. This model is estimated via Ordinary Least Squares (OLS) on the in-sample period. By incorporating this rich feature set, we aim to capture a range of return drivers including price trends, market overreaction, volatility clustering, inter-market dependencies, and behavioral biases tied to trading weekdays.

#DONE: add ethereum data -> Erich

#TODO: generate nicer latex table output of the regression results -> Daria

#TODO: Description and interpretation of output -> Laura

Table 3: Extended Regression Model: Predicting LINK Returns with Crypto Features

	Dependent variable:
	Target Return
Intercept	-0.0743^{**}
	(0.0297)
Momentum (3d)	-0.0125
	(0.0173)
Momentum (7d)	-0.0182
	(0.0156)
Momentum (14d)	0.0743^{**}
	(0.0295)
Lagged Return (1d)	0.0873^{***}
	(0.0319)
Lagged Return (2d)	-0.0534
	(0.0483)
Volatility (7d)	0.0002
	(0.0003)
RSI (14)	-0.0014
	(0.0016)
SMA Diff	0.0004
	(0.0005)
MACD Value	0.0012
	(0.0015)
MACD Histogram	-0.0025^{**}
	(0.0012)
ATR (14)	0.0052
	(0.0041)
Monday	0.0005
	(0.0038)
Tuesday	0.0028
	(0.0038)
Wednesday	0.0055
	(0.0037)

Lasso Model

To prevent overfitting and perform automatic variable selection, we extend our linear modeling approach using the Lasso (Least Absolute Shrinkage and Selection Operator). The Lasso adds a penalty term to the standard OLS loss function, shrinking some coefficient estimates toward zero. This results in a sparse model that may improve predictive performance, particularly when dealing with multiple correlated predictors.

The Lasso estimator is defined as the solution to the following optimization problem:

$$\hat{\beta}^{\text{lasso}} = \arg\min_{\beta_0,\beta} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

where:

- y_i is the target variable (e.g., one-day-ahead return),
- x_{ij} are the predictor variables,
- β_j are the coefficients,
- $\lambda \geq 0$ is the tuning parameter controlling the strength of the penalty.

As λ increases, more coefficients are shrunk toward zero. For $\lambda = 0$, the solution coincides with OLS.

We use 10-fold cross-validation to select the optimal λ that minimizes the mean squared prediction error on held-out data.

The **optimal lambda** (λ^*) is a key regularization parameter in LASSO regression. It controls the strength of the penalty applied to the model's coefficients. A higher value of λ increases shrinkage, setting more coefficients to zero and reducing model complexity. The optimal value is chosen via cross-validation to balance model fit and generalization.

Optimal Lambda from Cross-Validation: $\lambda^* = 0.002823$

Table 4: Non-Zero Coefficients from LASSO Regression

Predictor	Coefficient
Intercept	0.001076
Bitcoin Daily Return	-0.937392
Ethereum 7-Day Volume Momentum	0.000755

#TODO: explain the results

5. Forecasting & Backtesting

In-Sample testing

To evaluate the performance of our predictive models, we begin by conducting in-sample (IS) testing. This involves fitting each model on a fixed training sample and evaluating how well the model explains historical variation in the data.

We assess in-sample performance using the following criteria:

- **Mean Squared Error (MSE)**: Measures the average squared difference between predicted and actual returns.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

- **Adjusted R^2**: Indicates the proportion of variance explained by the model, adjusted for the number of predictors.

$$R^2_{\text{adj}} = 1 - \frac{\text{RSS}/(n-p-1)}{\text{TSS}/(n-1)}$$

- **Directional Accuracy**: The fraction of times the predicted direction matches the actual direction of returns.

$$\text{Accuracy} = \frac{1}{n}\sum_{i=1}^{n} ⊮ (\text{sign}(\hat{y}_i) = \text{sign}(y_i))$$

These metrics are computed for all three models:

1. Benchmark (7-day momentum only),
2. Extended linear model with multiple features,
3. Lasso-regularized regression with automatic feature selection.

```
## # A tibble: 3 x 4
##   Model          MSE Directional_Accuracy    Adj_R2
##   <chr>        <dbl>                <dbl>     <dbl>
## 1 Benchmark 0.00450                0.508 -0.000426
## 2 Extended  0.00297                0.720  0.331
## 3 Lasso     0.00305                0.719 NA
```

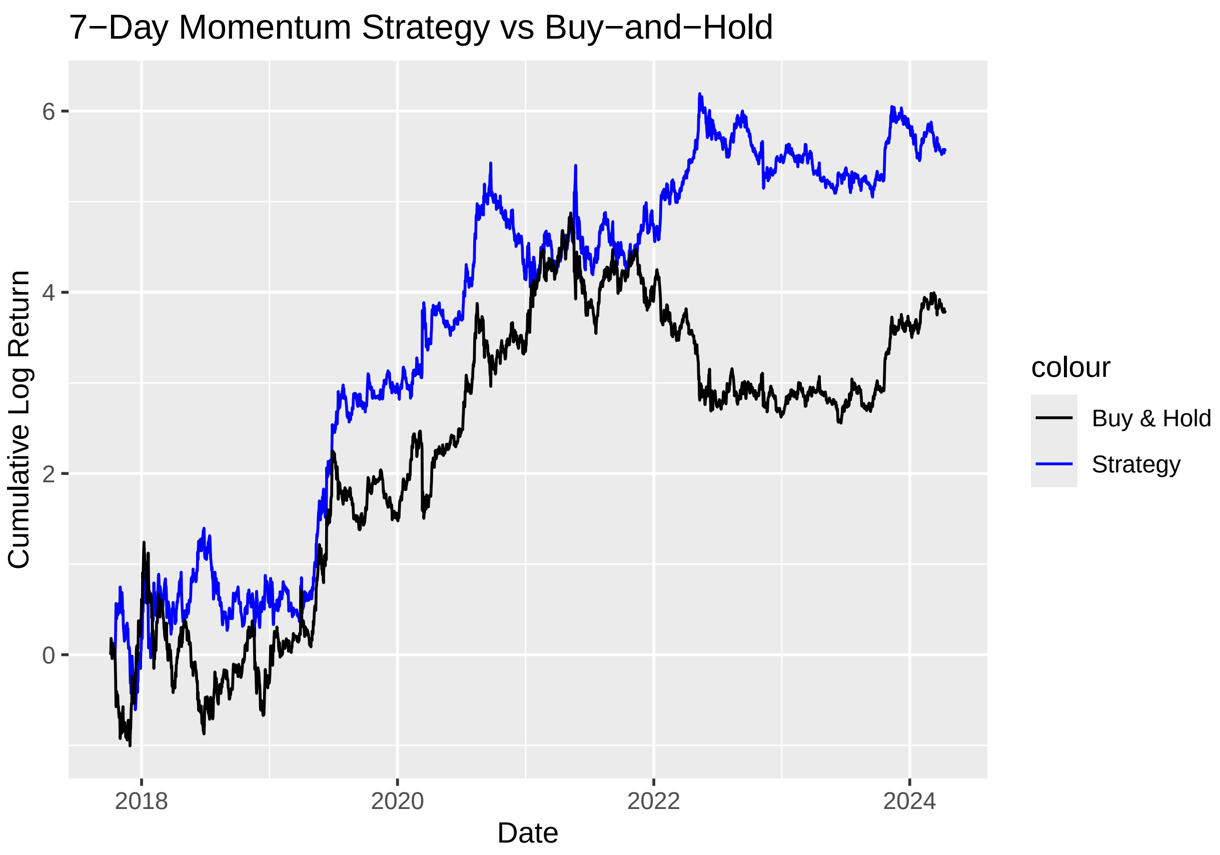
#TODO: interpret results

Out-of-sample testing:

#TODO: review code, does not work at the moment

#TODO: Evaluate: o Sharpe ratio o Cumulative return o OOSR2 o Hit rate (how often you correctly predict direction)

#TODO: include Transaction fees as extra path



6. Conclusion