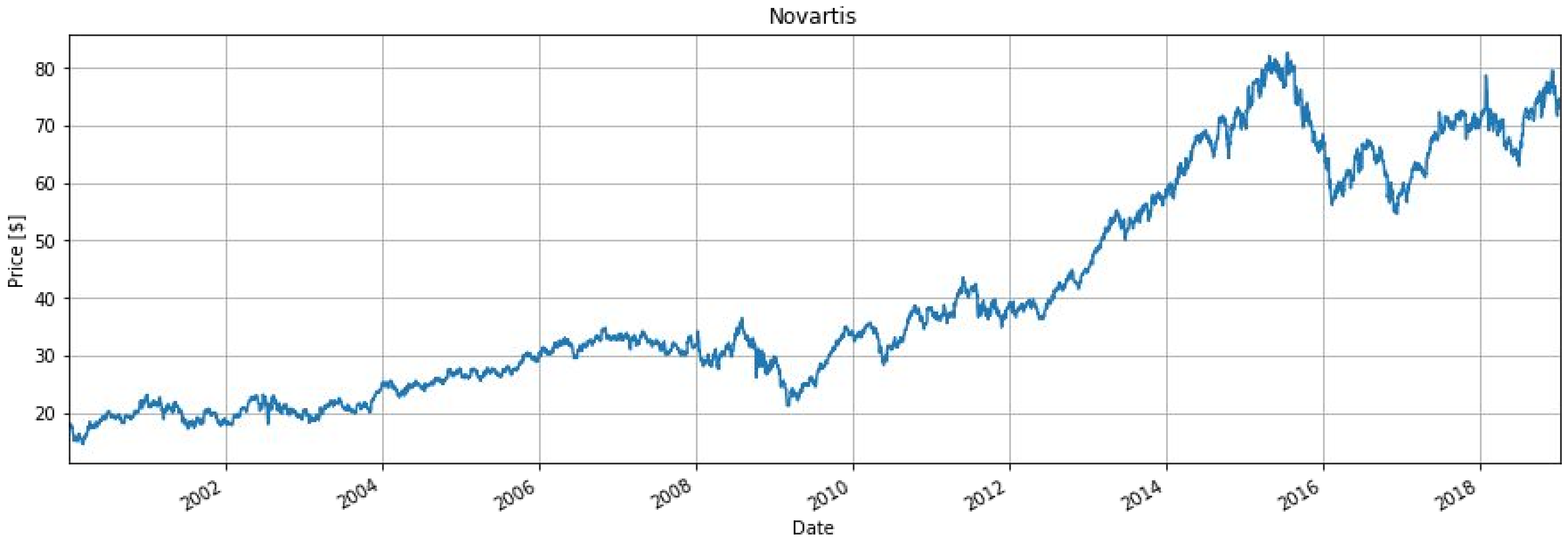
Stock Market Prediction

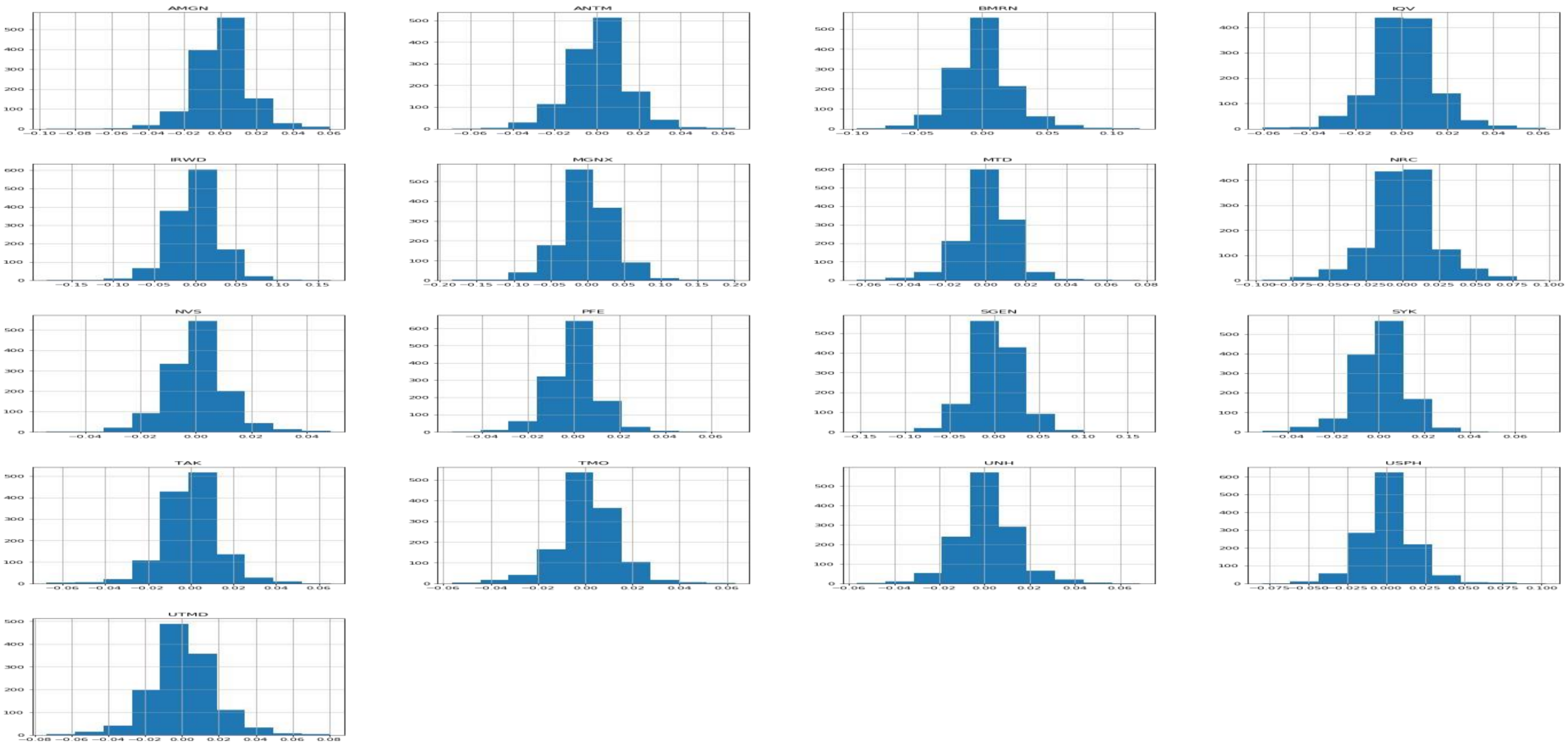


Biotech Industry

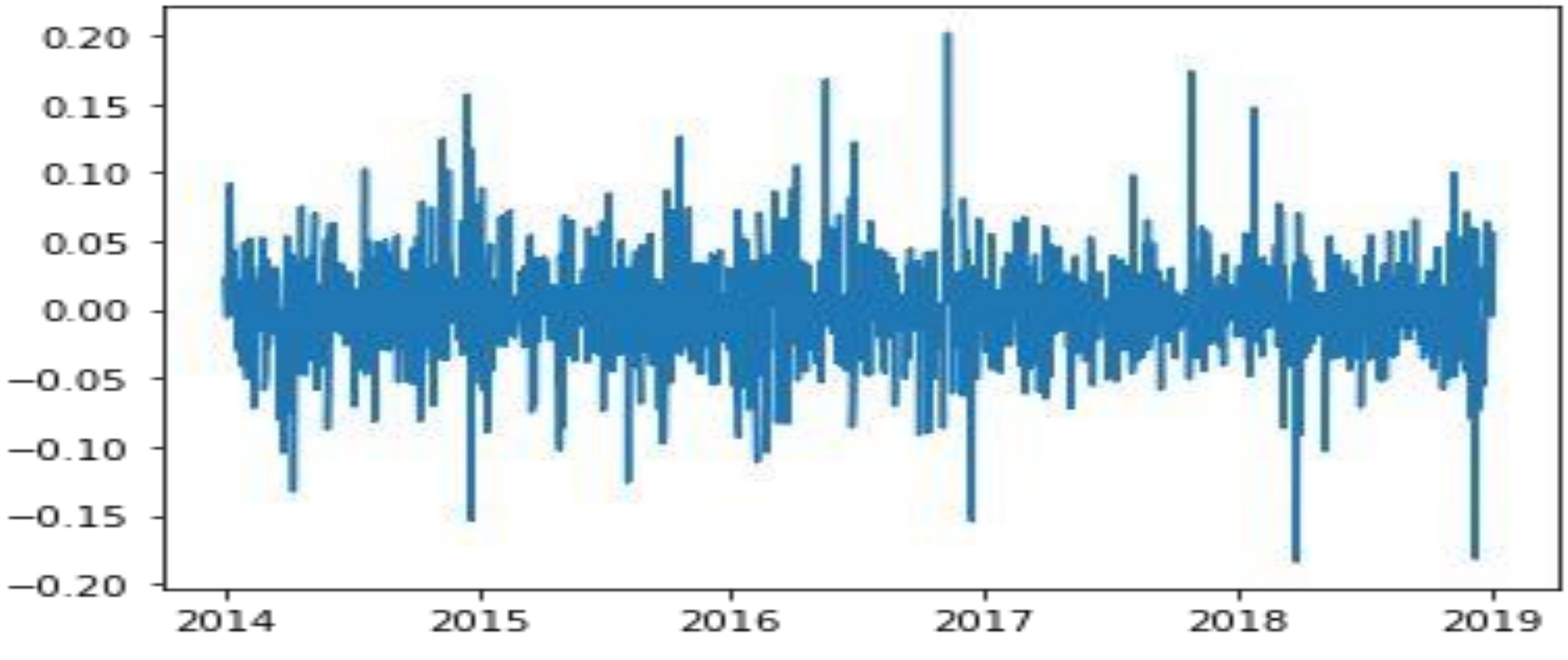
Portfolio Optimization for Stock Selection

- Choose a frequency at which to calculate returns, daily, monthly.
- Test tickers for normality using Kurtosis and Skewness thresholds.
- Skew: -0.5 < price < 0.5
- Kurtosis: price <= 3
- Finally, calculate Sortino Ratio by setting your Target Rate to 0 and your Risk Free Rate to the current RFR.
- Collect all the negative returns from your returns dataframe.
- Next calculate the expected returns on your returns dataframe.
- Lastly, calculate the Standard Deviation of the Negative Returns.
- Sort the values in descending order, choose top 5 stocks and you’re done.

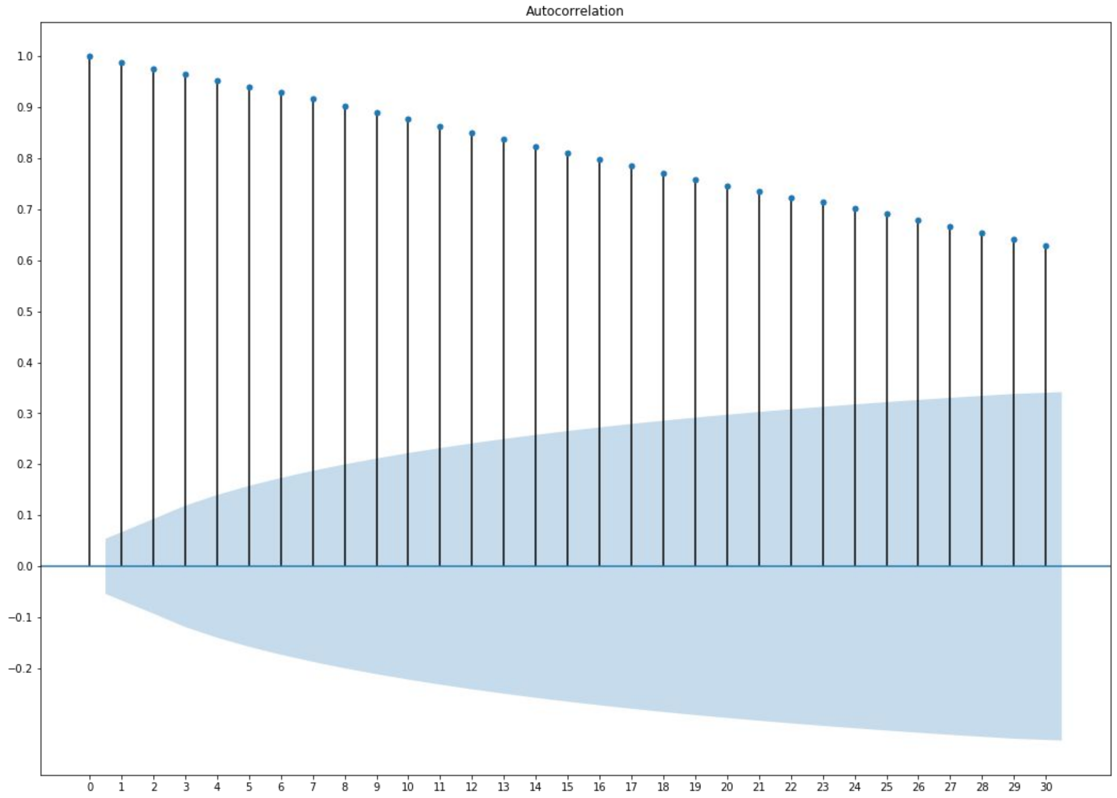
From 196 Tickers to 17 Normal Tickers



MGNX Returns for Best Sortino Ratio



Autocorrelation of MGNX with 30 Lags



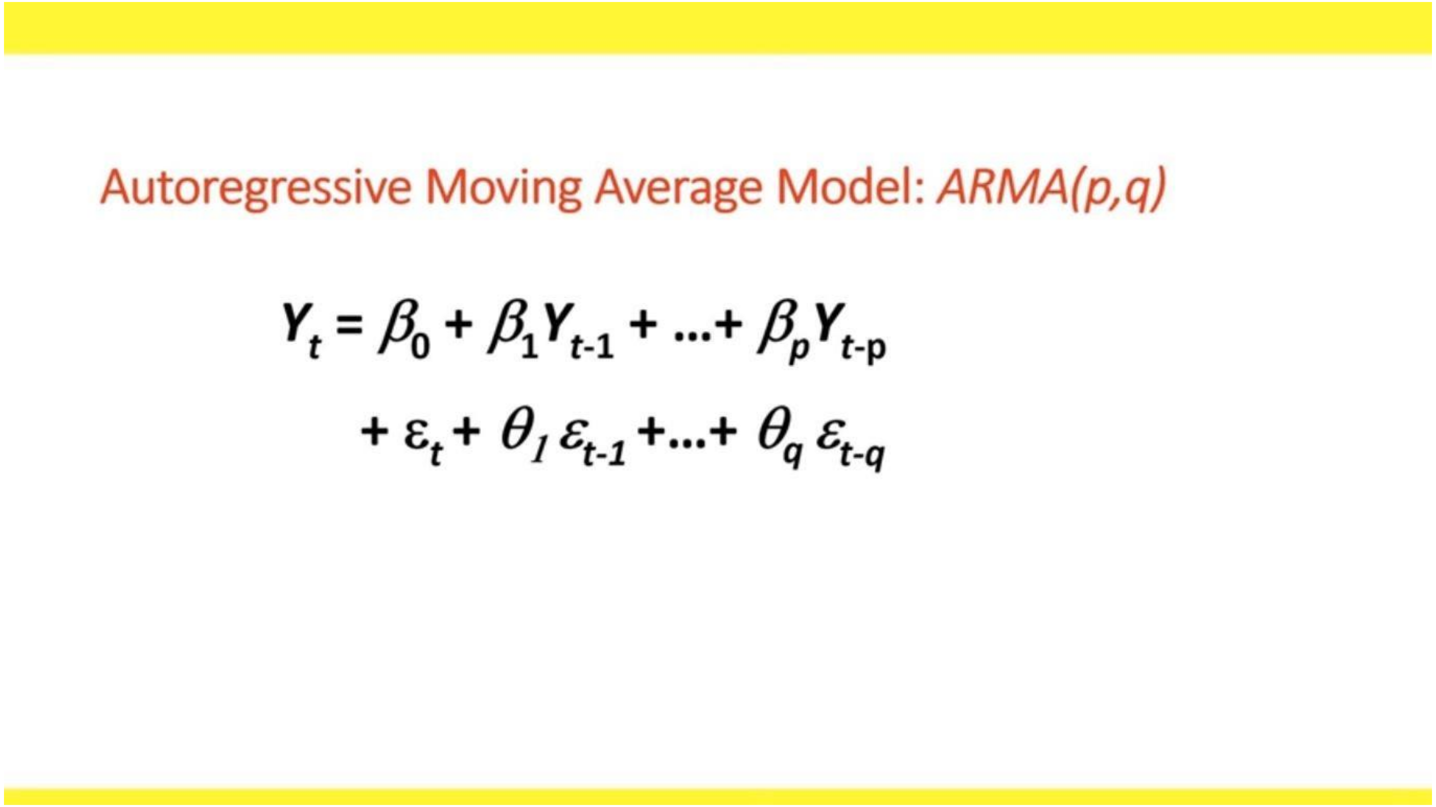
Autocorrelation and Partial Autocorrelation

- AC is a times series technique used specifically for autoregression models. Given some number of lag variables originating from a univariate time series, you can find the correlation between the lag variable and the original time series. If correlation exists on the lag variable, it may be considered statistically significant.
- PAC is a similar technique that is able to overlook intermediate time steps in the past and give an estimate of correlation between distant lags and the original time series thereby being useful in deciding the order of parameter p in an arima model. P is the number of lags.

Results from ACF/PACF

- ACF with 30 lags tells us that even though the lags are significant, they degrade over time.
- PACF from the exploratory notebook in the project repository tells us that even with 2 or more lags, the number of significant lags will always be 2.

Autoregressive Model



Autoregressive Moving Average Model: *ARMA(p,q)*

$$Y_t = \beta_0 + \beta_1 Y_{t-1} + \ldots + \beta_p Y_{t-p}$$

$$+ \varepsilon_t + \theta_1 \varepsilon_{t-1} + \ldots + \theta_q \varepsilon_{t-q}$$

MAPE Scores for all Models

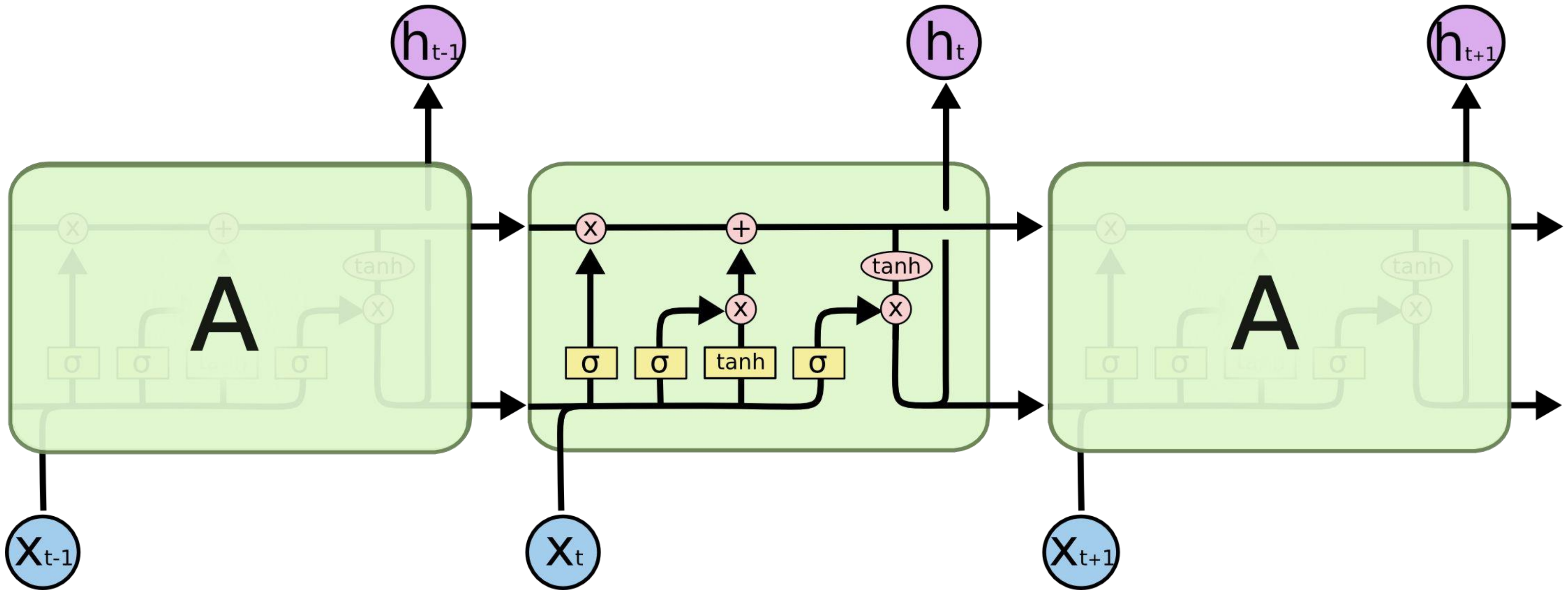
- MAPE Scores:

Auto Arima:	2.38
Simple Moving Average:	4.37
FB Prophet:	29.7
TBATS:	5.97

- MAE Score:

  LSTM: .06

LSTM Architecture (1)



LSTM Architecture (2)

- The architecture of an LSTM is special.
- It is an answer to the vanishing gradient problem.
- It contains a chain of cells, each with three gates regulating inputs based on weight outcomes. These gates are called the input, forget and output gates. These gates control the flow of information and activation functions such as the sigmoid and tanh function ensure that the scale of information is manageable so that the gradients do not explode.
- These are the basic ingredients governing LSTM model capacity to generate weights that generalize well for prediction.
- This may explain why LTSM outperforms autoregressive models in a short term time series.

Preparation/Results from LSTM

- Preparing the inputs on the daily stock data can be done in several ways. It may be customary to transpose your data so that the time slices are received in fixed intervals making your columns t, t1, t2,…,tn. Since we only worked with a column, we made the input simple with shape (1,1) for the length of the sequence or elements in each row vectors and number of signals/columns. The preparation of input is intuitive as the input must be taken in time slice sequence batches. The rest of the implementation is up to model architecture of memory and gradient optimization.
- The results based on the metric MAE were comparatively good with the following results in the tutorial: https://towardsdatascience.com/forecasting-stock-prices-using-prophet-652b31fb564e