Actively Managing a Passive Index

Steve Teeters
Alex Mowlem
Rex Goodstein
Chris Contos

We decided to ask, “is it possible to Actively Manage a Passive Index in an effort to Control Downside Capture?”

Our Project

- Use an API to Gather Data on an index-based ETF
- Establish a Baseline
- Predict the future movement through a Machine Learning Model
- Implement an Algorithmic Trading Strategy

Building an API

```
# In code install of iexfinance
%pip install "iexfinance"
```

```
Requirement already satisfied: iexfinance in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (0.5.0)
Requirement already satisfied: requests in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from iexfinance) (2.27.1)
Requirement already satisfied: pandas in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from iexfinance) (1.3.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from pandas->iexfinance) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from pandas->iexfinance) (2021.3)
Requirement already satisfied: numpy>=1.17.3 in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from pandas->iexfinance) (1.20.3)
Requirement already satisfied: certifi>=2017.4.17 in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from requests->iexfinance) (2021.10.8)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from requests->iexfinance) (1.26.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from requests->iexfinance) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from requests->iexfinance) (2.8)
Requirement already satisfied: six>=1.5 in /Users/steveteeters/opt/anaconda3/lib/python3.9/site-packages (from python-dateutil>=2.7.3->pandas->iexfinance) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

"Data provided for free by IEX. By accessing or using IEX Historical Data, you agree to the IEX Historical Data Terms of Use."

https://exchange.iex.io/products/market-data-connectivity/hist-terms/

```
# Import iexfinance dependencies
from iexfinance.stocks import Stock
from iexfinance.stocks import get_historical_data
```

```
# Daily historical closing data for SPDR S&P 500 ETF covering part of 2007
# Note that the token used is a public, publishable key and not our secure key.
spy_hist_07 = get_historical_data("SPY", start="20070501", end="20111231", token ='pk_acbe732a9a1646a9b674c3bc5f75d030',
                  output_format='pandas')
spy_hist_07.head()
```

	close	high	low	open	symbol	volume	id	key	subkey	updated	...	uLow	uVolume	fOpen	fClose	fHigh	fLow	fVolume	label	change	changePercent
2007-05-01	148.67	149.47	147.67	148.53	SPY	134447334	HISTORICAL_PRICES	SPY		1648509555000	...	147.67	134447334	92.7672	92.8546	93.3542	92.23	134447334	May 1, 07	0.38	0.0026
2007-05-02	149.54	149.95	148.75	149.12	SPY	87183213	HISTORICAL_PRICES	SPY		1648509670000	...	148.75	87183213	93.1356	93.398	93.654	92.9046	87183213	May 2, 07	0.87	0.0059
2007-05-03	150.35	150.4	149.05	149.72	SPY	87865741	HISTORICAL_PRICES	SPY		1648509483000	...	149.05	87865741	93.5104	93.9039	93.9351	93.0919	87865741	May 3, 07	0.81	0.0054
2007-05-04	150.92	151.12	150.22	150.77	SPY	96424729	HISTORICAL_PRICES	SPY		1648509670000	...	150.22	96424729	94.1662	94.2599	94.3848	93.8227	96424729	May 4, 07	0.57	0.0038
2007-05-07	150.95	151.2	150.81	150.92	SPY	63691549	HISTORICAL_PRICES	SPY		1648509753000	...	150.81	63691549	94.2599	94.2786	94.4347	94.1912	63691549	May 7, 07	0.03	0.0002

ARIMA Model Results

Dep. Variable:	D.close	**No. Observations:**	1182
Model:	ARIMA(5, 1, 1)	**Log Likelihood**	-2377.216
Method:	css-mle	**S.D. of innovations**	1.808
Date:	Wed, 13 Apr 2022	**AIC**	4770.431
Time:	16:37:51	**BIC**	4811.031
Sample:	1	**HQIC**	4785.736

	coef	std err	z	P>|z|	[0.025	0.975]
const	-0.0199	0.042	-0.472	0.637	-0.102	0.063
ar.L1.D.close	0.1293	0.704	0.184	0.854	-1.251	1.510
ar.L2.D.close	-0.0578	0.083	-0.695	0.487	-0.221	0.105
ar.L3.D.close	0.0278	0.067	0.414	0.679	-0.104	0.160
ar.L4.D.close	0.0006	0.030	0.020	0.984	-0.058	0.060
ar.L5.D.close	-0.0485	0.029	-1.665	0.096	-0.106	0.009
ma.L1.D.close	-0.2402	0.706	-0.340	0.733	-1.623	1.143

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	1.4862	-1.0221j	1.8037	-0.0959
AR.2	1.4862	+1.0221j	1.8037	0.0959
AR.3	-0.4576	-1.7001j	1.7606	-0.2918
AR.4	-0.4576	+1.7001j	1.7606	0.2918
AR.5	-2.0448	-0.0000j	2.0448	-0.5000
MA.1	4.1627	+0.0000j	4.1627	0.0000

ARMA Model Results

Dep. Variable:	close	**No. Observations:**	1182
Model:	ARMA(2, 1)	**Log Likelihood**	-2299.888
Method:	css-mle	**S.D. of innovations**	1.694
Date:	Wed, 13 Apr 2022	**AIC**	4609.775
Time:	16:35:21	**BIC**	4635.150
Sample:	0	**HQIC**	4619.341

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.0002	0.041	0.005	0.996	-0.080	0.080
ar.L1.close	-0.2610	0.204	-1.277	0.202	-0.662	0.140
ar.L2.close	-0.1270	0.034	-3.761	0.000	-0.193	-0.061
ma.L1.close	0.1477	0.205	0.719	0.472	-0.255	0.550

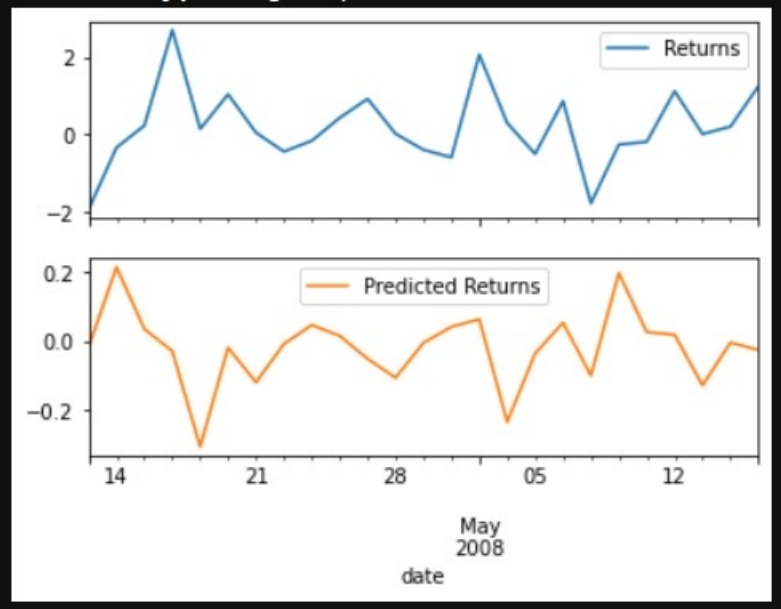
Roots

	Real	Imaginary	Modulus	Frequency
AR.1	-1.0276	-2.6112j	2.8061	-0.3097
AR.2	-1.0276	+2.6112j	2.8061	0.3097
MA.1	-6.7688	+0.0000j	6.7688	0.5000

To use a Machine or Not...

- Decompose using Hodrick Prescott
- Build ARMA and ARIMA models
- Results do not satisfy goals of project

```
array([<AxesSubplot:xlabel='date'>, <AxesSubplot:xlabel='date'>],
      dtype=object)
```



Out-of-Sample Performance

```
# Out-of-Sample Performance analyzes the data that was left out of the training during the split.
# This is to test the level of error on the testing data. (X_test)(y_test)
from sklearn.metrics import mean_squared_error
# Calculate the mean_squared_error (MSE) on actual versus predicted test "y"
mse = mean_squared_error(
    Results["Returns"],
    Results["Predicted Returns"]
)

# Using that mean-squared-error, calculate the root-mean-squared error (RMSE):
out_of_sample_rmse = np.sqrt(mse)
print(f"Out-of-Sample Root Mean Squared Error (RMSE): {out_of_sample_rmse}")
```

```
Out-of-Sample Root Mean Squared Error (RMSE): 1.8109575138860936
```

In-Sample Performance

```
# In-Sample Performance used to test the trained data set used in the model.
# Evaluate the model using the X_train and the y_train data.

in_sample_results = y_train.to_frame()

# Add the column of "in-sample" predictions to dataframe.
in_sample_results["In-sample Predictions"] = model.predict(X_train)

# Calculate in-sample mean_squared_error (for comparison to out-of-sample)
in_sample_mse = mean_squared_error(
    in_sample_results["Returns"],
    in_sample_results["In-sample Predictions"]
)

# Calculate in-sample root mean_squared_error (for comparison to out-of-sample)
in_sample_rmse = np.sqrt(in_sample_mse)
print(f"In-sample Root Mean Squared Error (RMSE): {in_sample_rmse}")
```

```
In-sample Root Mean Squared Error (RMSE): 1.4712260923811535
```

Regression Analysis

- Out-Of-Sample and In-Sample Analysis
- Root Mean Squared Error Analysis
- Model showed poor results for forecasting accurate future data.

Machine Learning

LSTM modeling S&P 500 Index (2017-2022)

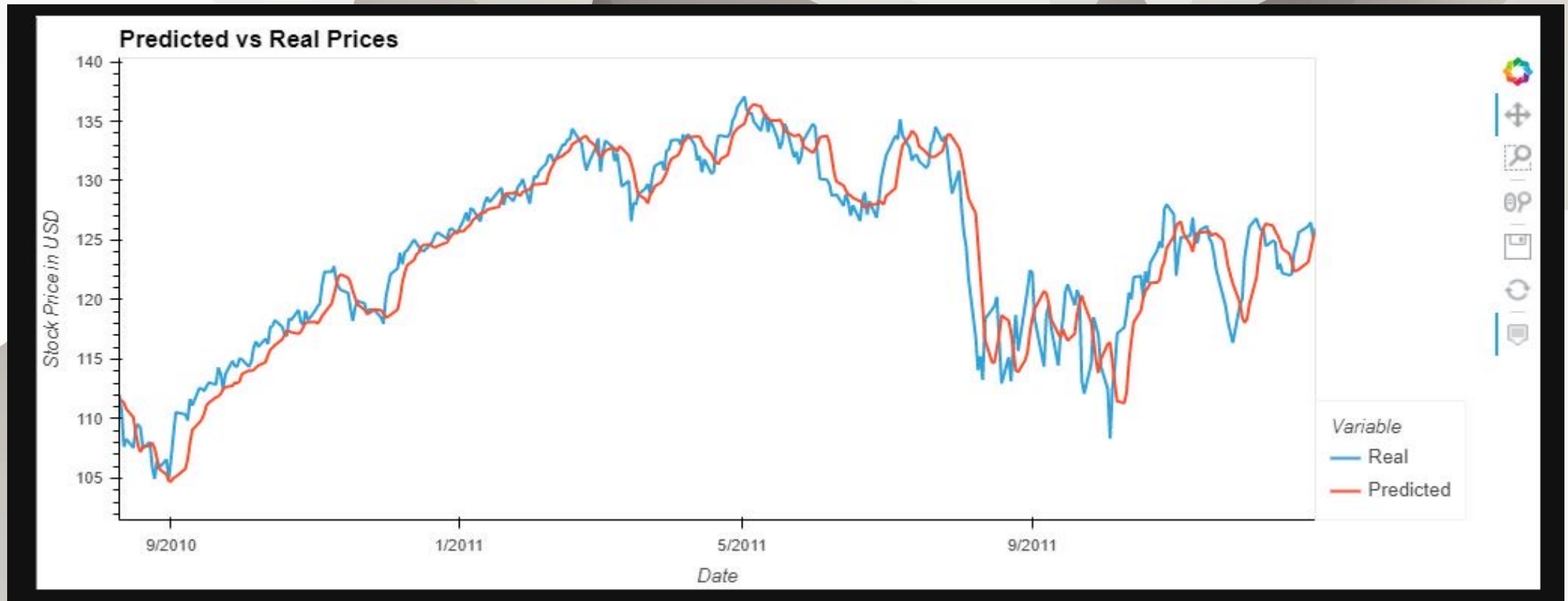
- Out-Of-Sample and In-Sample Analysis
- Using 2017 to 2022 to back test the index
- Predicts prices to a day out given the current price and the last 10 historical prices

Machine Learning

Model Setup

- Number of Units: 10
- 3 Layers
- Epoch = 20
- Batch Size = 1

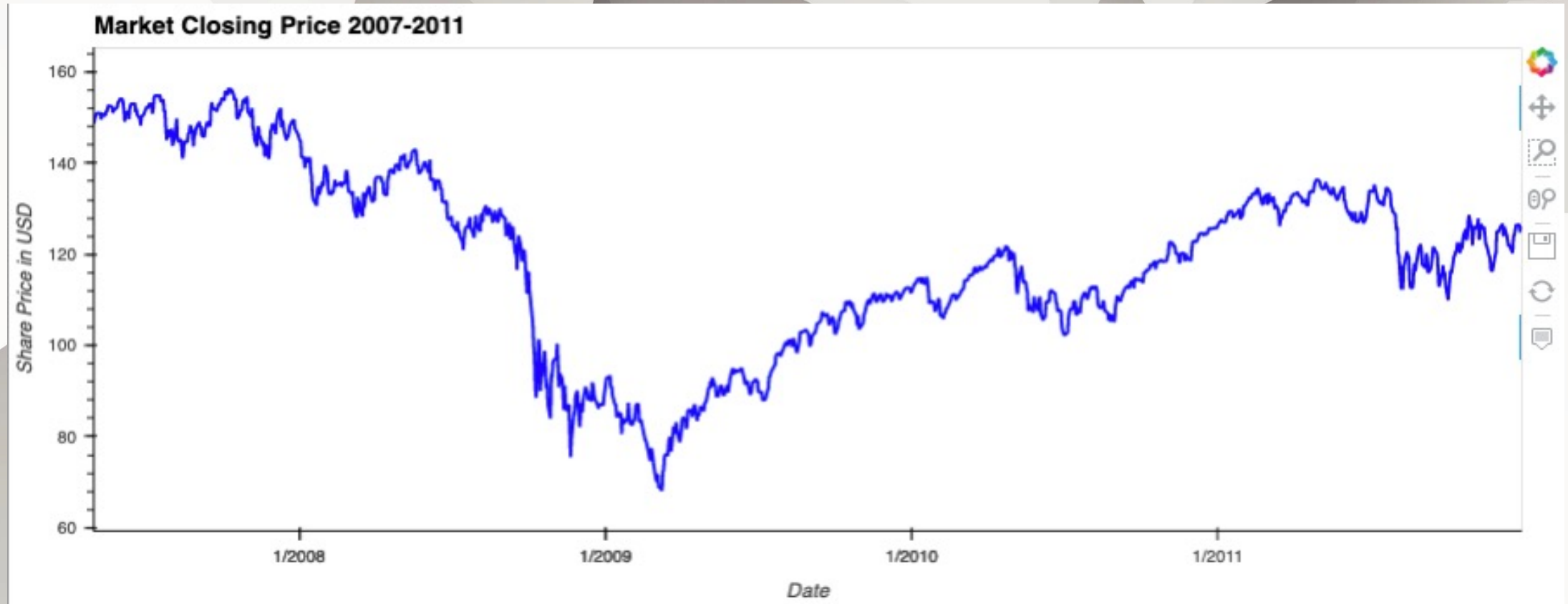
Results: Test Data



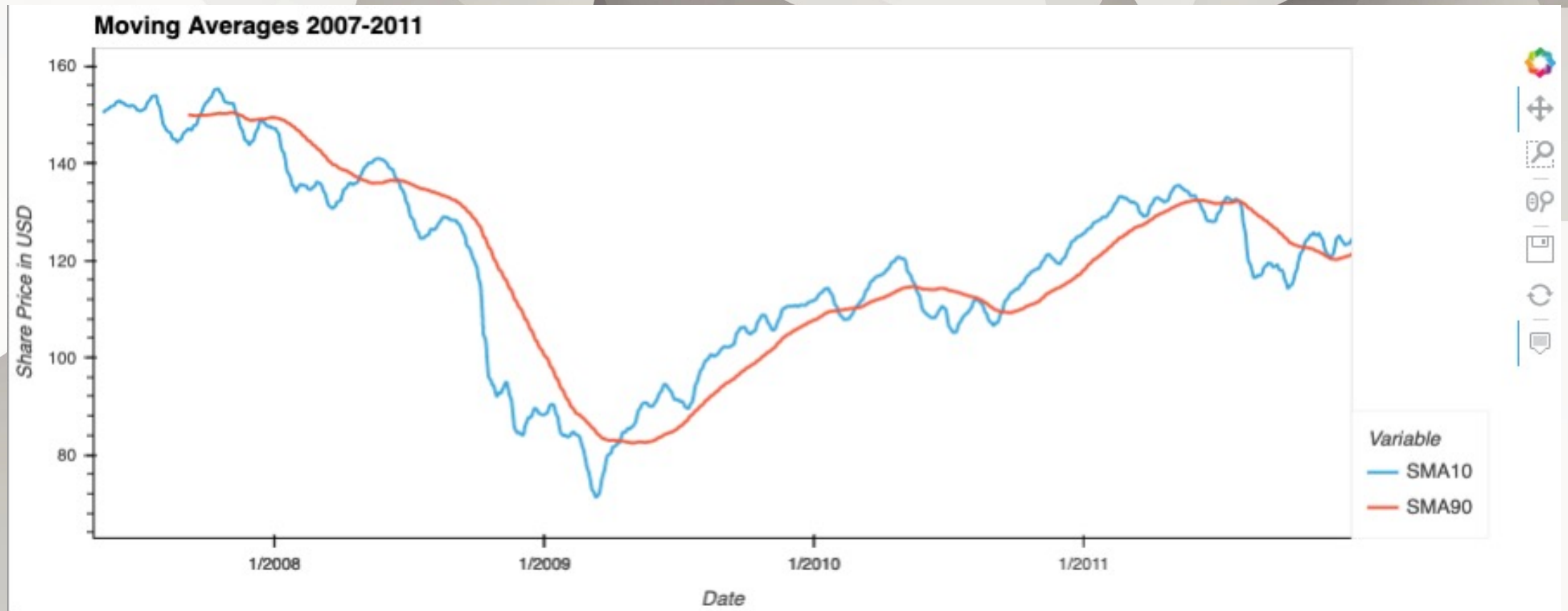
Algorithmic Trading

- Establish a Baseline
- Utilize a DMAC
- Tune the Short & Long
- Run Predicted Data
- Compare Results

Baseline 2007



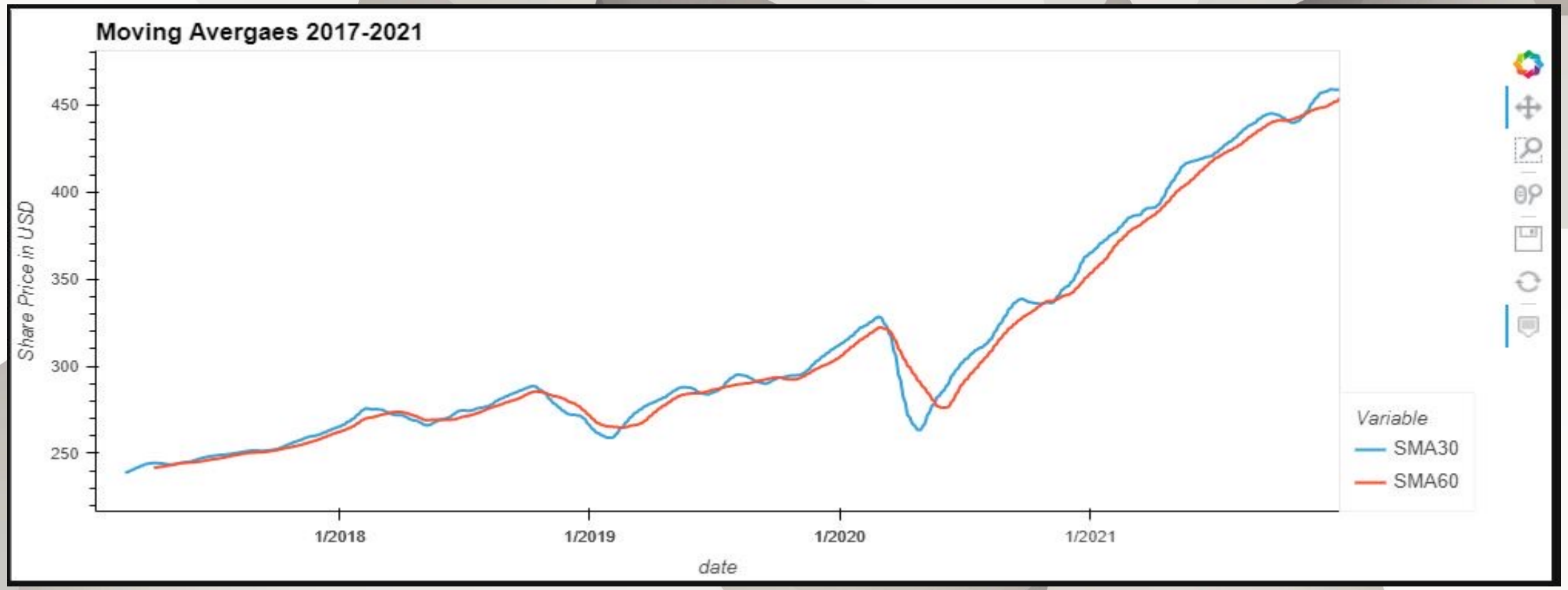
Tuned the Short & Long



DMAC vs Actual
2007- 2011

Tuned the Short & Long 2017- 2021

[Predicted Data]



DMAC vs Predicted 2017- 2021

Summary

- Using linear regression and time series analysis proved to be insufficient for the goals of our project.
- The LSTM model works well at predicting prices a day out but the close price was the only variable fed into the data. To predict prices further out, you need supplemental data such as sentiment analysis.
- Given current volatility of the market, DMAC may no longer be a viable trading strategy.

Questions ?