Project Overview: LSTMs for Time Series Predictions

Charif Salman

October 2025

1 Goal

This study investigates LSTM neural network architectures for predicting price movements in crude oil Exchange Traded Funds (ETFs), the ProShares Ultra DJ-UBS Crude Oil ETF (HOD) and ProShares Ultra Bloomberg Crude Oil ETF (UCO). The study employs an approach that combines custom loss functions, ensemble methods, and validation techniques to minimize severe false positives while maximizing prediction precision in high-confidence scenarios. This is a work in progress, and improvements that must be investigated in the next version are mentioned at the end of the document.

Thus far, the study is focused on model and evaluation architecture rather than feature extraction/engineering and data selection. The idea is to build a selective model, that performs well (selective high precision signals) with data from economic indicators and basic feature engineering.

2 Data

The data was selected based on variable selection methods employed the following paper: `https://energyinformatics.springeropen.com/articles/10.1186/s42162-021-00166-4#Sec8`

The only transformation performed on the variables is applying a percent change calculation of the value compared to the previous period. This makes the series closer to stationary with an approximately normal distribution. The next iteration of the study will include feature extraction methods such as the use of wavelets and Short Fourier Transforms.

The predictor variable is the ETF time series (UCO and HOD) to trade market surges and crashes. The predictor variable is transformed as such:

- (1) Create a time series of the data where each value represents the average price over the last 7 days of the month
- (2) Take the percent change of the time series. It now represents the percent change of the average price in last 7 days of the month compared to the average price in the last 7 days of the previous month.
- (3) The predictor variable passed to the LSTM is transformed into into a binary classes **(1 (positive class) if there is a + 0.10 % change in the ETF price over the past month and 0 (negative class) otherwise)**

So far, the **trading strategy** is simple: when there is a signal, hold the stock for one month and sell at the end of the month. In reality, this strategy would not be used as stated, but in this case a simple strategy is applied to evaluate the predictive power of LSTM based monthly predictions.

The explanatory variables are:

- US PPI, Energy, Seasonally Adjusted
- EU 28 Countries PPI

- US PMI Index
- Petroleum Inventory, Total OECD
- Crude Oil Inventory, SPR (Strategic Petroleum Reserve)
- Crude Oil Inventory, Non-SPR,
- Crude oil non-commercial net long ratio
- Real dollar index: generalized
- LME: Copper: Future closing price
- WTI crack spread
- Brent crack spread

3 Loss Functions

Quick Overview

- (A) Standard BCE (no weighting).
- (B) Increases FP penalty as p_i crosses $0.7, 0.8, 0.9$
- (C) Adds an extra multiplier λ_{sev} for very high-confidence FPs when realized return is severe ($r_i \leq -12\%$).

Notation

- Predictions $p_i \in [0,1]$ since it is the logit output of the LSTM passed into a sigmoid.
- $p_i > 0.5$ are classified as positive class predictions.

$$
\begin{aligned}
&z_i \in R \text{ (logit)}, \quad p_i = \sigma(z_i) = \frac{1}{1+e^{-z_i}}, \quad y_i \in \{0,1\},\\
&r_i = \text{realized return}, \quad \theta_{\text{sev}} = -0.12, \quad \lambda_{\text{sev}} = 1.5,\\
&\lambda_{\text{bal}} = \texttt{balancing\_Weight\_factor},\\
&(\lambda_1, \lambda_2, \lambda_3) = \begin{cases} (1.3,\ 1.7,\ 2.5) & \text{if } \texttt{use\_LOW\_weights} = \text{True} \\ (1.5,\ 2.0,\ 3.0) & \text{if } \texttt{use\_LOW\_weights} = \text{False} \end{cases}
\end{aligned}
$$

(A) Baseline BCE with logits

$$\mathcal{L}_{\text{BCE}} = \Big[-\, y_i \log p_i - (1-y_i)\log(1-p_i)\Big]$$

(B) Threshold-Weighted BCE (penalize confident FPs)

Per-sample weight w_i:

$$
w_i = \begin{cases}
1, & y_i = 1,\ p_i \geq 0.5 \quad \text{(TP)} \\
\lambda_{\text{bal}}, & y_i = 1,\ p_i < 0.5 \quad \text{(FN)} \\
1, & y_i = 0,\ p_i < 0.5 \quad \text{(TN)} \\
\lambda_1, & y_i = 0,\ p_i \in [0.7, 0.8) \quad \text{(FP, low hi-conf)} \\
\lambda_2, & y_i = 0,\ p_i \in [0.8, 0.9) \quad \text{(FP, mid hi-conf)} \\
\lambda_3, & y_i = 0,\ p_i \in [0.9, 1.0] \quad \text{(FP, very hi-conf)}
\end{cases}
$$

Weighted BCE:

$$\mathcal{L}_{\text{BCE-THR}} = \Big[- y_i \log p_i - (1 - y_i) \log(1 - p_i) \Big]$$

(C) Threshold + Severity-Weighted BCE (extra penalty for severe losses)

Same w_i as (B), but when the FP is both very high-confidence and *severe* ($r_i \le \theta_{\text{sev}}$), multiply by λ_{sev}:

$$\tilde{w}_i = \begin{cases} w_i \cdot \lambda_{\text{sev}}, & \big(y_i = 0,\ p_i \in [0.9, 1.0]\big) \ \wedge \ \big(r_i \le \theta_{\text{sev}}\big) \\ w_i, & \text{otherwise} \end{cases}$$

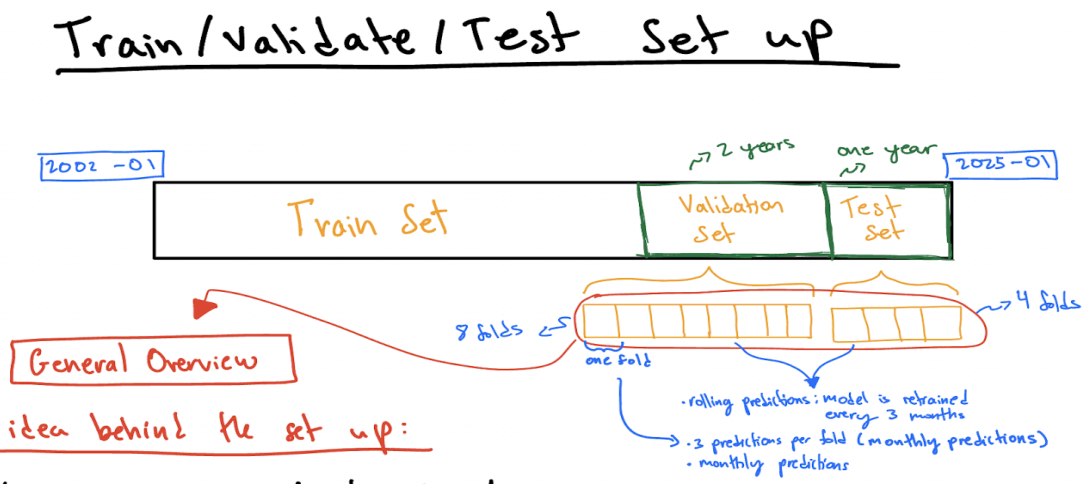
$$\mathcal{L}_{\text{BCE-THR-SEV}} = \Big[- y_i \log p_i - (1 - y_i) \log(1 - p_i) \Big]$$

4 Grid Search

LSTM Model Parameters for GS combinations:

- `binary_0_1_cutoff_ret_rate_percentage`
- `learning_rate`
- `num_epochs`
- `batch_size`
- `use_bidirectional`
- `lag`
- `input_size`
- `hidden_size`
- `num_layers`
- `custom_loss_function_BCE_THRESH`
- `custom_loss_function_BCE_THRESH_AND_SEVERITY`
- `use_LOW_weights_for_BCE_custom_loss`
- `use_dropout`
- `use_class_weighting`
- `seed_num`
- `use_dynamic_weights`
- `POS_weight_multiplier`
- `use_rolling_fixed_train_size`
- `use_UCO_wticoncat_predictor_WEEKLY_END_MO`
- `use_HOD_wticoncat_predictor_WEEKLY_END_MO`
- `train_start_month`
- `val_start_month`
- `val_end_month`
- `num_preds_per_fold`

5 Overview of the Model Selection and Inference Process



The idea behind the set up:

① • Models are grouped together based on their characteristics in the V set

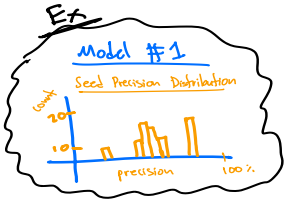
② • Model performance metrics in V set & T set are compared

③ • Track model metrics/characteristics that remain similar in the T set when compared to V set

④ • Ensemble Models, grouped together based on their characteristics and track forecasting performance

⑤ The process outlined is backtested over multiple years in standardized fashion (ie. the same tests conducted each year) to find patterns that remain consistent.

Deeper Dive



assess the distribution for different metrics (accuracy, recall, precision ...)

one seed = a model with a weight initialization

for each model combo, run the train / val / test process for 40 different seeds

Choose top 90 params from GS based on V set prediction performance → One param. combination = one model

≈ 100k total combinations of LSTM model parameters → choose 3500 combinations from the 100k an run the GS

the metrics from the GS are from the model performance on the validation set

① Grid Search

≈ 12 variable parameters

② Distribution Discovery

purpose: can we choose desirable models based on characteristics observed in the V set? If so, which characteristics?

ie. for a given group of models, track features that remain similar in the V & T set

Model & Seed Selection

group seeds based on model & seed characteristics

run standardized analytics on each group

Test Set Returns Performance

Compute returns per signal (one signal is a group of ensembled models)

backtest over multiple years to find patterns

Model Ensemble Method for Inference

Models are grouped together based on characteristics explored in the V set results/metrics

models & seeds in one group of models

A

B

Different subset chosen at next iteration

subset of chosen models

C

① Pick x seeds in total, one from each model in the subset

② Group models pairwise, independent models paired together

③ The models output a logit passed to a sigmoid (values range from [0,1]. Apply a threshold filter for the positive class → only accept predictions that fall within [0.7, 1.0] or [0.9, 1.0] bracket

④ Apply selective ensemble on models in ②: only consider a prediction if both models agree

this condition is included due to characteristics of conditional probabilities P(FP | both models predict P) → what is probability of observing a FP given both models predict a positive class observation → the idea: this probability is minimized if both models rarely make the same FP predictions

seeds are paired together the number of False Positive predictions that both models agree on (this is decided based on model metrics in the Val set)

Model #1: {learning_rate: 0.05, batch_size: 20, ...} → fixed LSTM parameters

- Seed #1: {learning_rate: 0.05, batch_size: 20, ..., seed = 113}
- Seed #2: {learning_rate: 0.05, batch_size: 20, ..., seed = 423}

(40 different seed initializations)

Purpose: this process decreases variance in the results

D Repeat A, B, C for the model group multiple times (a different subset of models for each iteration)

↳ Lastly, collect unique "up" predictions from the results from each subset picked

E Complete process D for each model group

the actual % change value represents the return over the past month if you buy and hold the ETF

⑤ Collect the actual % change value for all positive class predictions

The custom loss function penalizes severe (< -12% loss) predictions that models output in the [0.9, 1.0] bracket

6 Model Selection Features Tracking

Overview of some of the analytics

Purpose:

- Discover characteristics of the models (or seed) that remain similar in the V set and T set.
- There are high performing models in the T set, the performance of models is correlated (possibly) to the filters applied during model selection (grouping models based on their characteristics). The analysis below allows us to track characteristics of high performing models, checking if they remain consistent over multiple Test years.

Rank change per combo (V order kept)

- Rank shift per combo: $\Delta rank_i = rank_i^{(T)} - rank_i^{(V)}$
- $\Delta rank_i > 0$ (worse) , $\Delta rank_i < 0$ (better)
- Purpose: models are ranked based on mean value in V set, check if the rank of the models relative to others is preserved in the T set

Mean change per combo (V order kept)

- Mean shift per combo: $\Delta \mu_i = \mu_i^{(T)} - \mu_i^{(V)}$
- Purpose: visualize the difference in mean value for the same model in V set compared T set

Recall-up distributions (V vs T; low/high cohorts)

- $\text{Recall}\uparrow = \dfrac{\text{TP}\uparrow}{\text{TP}\uparrow + \text{FN}\uparrow}$
- Plot histograms for control / low / high recall cohorts , we hope that the relative positions of the recall up metrics remain consistent in the V set distribution of the recall metrics when compared to the T set distribution.
- Purpose: discover if the recall up remains similar in the T set for a given model.

T-set Means by V-set Mean Ranges (Same Model Comparisons)

- Bin by models in V-set by mean precision (positive class): $[0, 30)$, $[30, 60)$, $[60, 100] \longrightarrow$ Plot histograms of the mean precision (positive class) values, for the same models (and same seed initializations) in the T set $\longrightarrow$ compare distribution of model precision metrics
- Purpose: discover if the mean values and variance of the model precision up values are higher in the T set depending on the range of mean values in the V set. Want to see if models generalize better to T set based on the mean values of the models positive class precision metric in the V set. ie. high preforming models in the V set may generalize poorly ...

Severe FP ratio among all FPs (V vs T)

- SevereFP Ratio $= \dfrac{\text{Severe FPs}}{\text{All FPs}}$
- Compares the ratio distribution in the V set and T set, for a control and test model.
- Tested Models:
  - Models with a high ratio (SevereFP Ratio metric)
  - Models with a a low ratio (SevereFP Ratio metric)
  - Control Model
- Purpose: Discover if the distribution of the ratio metric remains similar in the V and T set. We do not expect this to be the case without additional model filters but there could be a weak trend. We may expect this to be the case when models with custom loss functions are filtered to "accept" predictions in the upper brackets [0.7, 1.0] and [0.9, 1.0].

FP ratio difference across brackets

- $\Delta_{\text{ratio}} = \text{FP\%}[0.5, 0.7] - \text{FP\%}[0.7, 1.0]$
- $\text{FP\%}[L, H] = \dfrac{\text{FP}[L, H]}{\text{TP}[L, H] + \text{FP}[L, H]}$
- Tested Models:
  - Models with a high ratio difference Δ_{ratio}
  - Models with a a low ratio difference Δ_{ratio}
  - Control Model
- Purpose: Discover if the distribution of the ratio difference metric remains similar in the V and T set. We do not expect this to be the case without additional model filters but there could be a weak trend.

Seed-level FP/TP in High Bracket (0.7–1.0)

- Per-seed histograms of TP and Severe FP counts in [0.7,1.0]
- Model of different groups are compared:
  - Model with $(FPs)_{Hb} = 0$ and $(TPs)_{Hb} \geq 1$
  - Model with $(TP - FP)_{Lb} - (TP - FP)_{Hb} \geq 3$
  - Control Model
- Metric: the metric analyzed is the count of FPs and TPs in the [0.7,1.0] bracket.
- Purpose: Discover if the distribution of the metric counts remains similar in the V and T set. We do not expect this to be the case without additional model filters but there could be a weak trend.

Correlation and Recall FP Ratio Analysis (V vs T)

- Models of different groups compared:
  - low recall models
  - high recall models
  - low recall models - custom BCE loss with severity penalization in high bracket [0.9 ,1.0]
  - high recall models - custom BCE loss with severity penalization in high bracket [0.9 ,1.0]
- **Note**: the models in each group are paired together, and only instances where both models agree on a prediction are considered. This is relevant since the model ensembling stategy employs a selective ensemble method that only considers trades that both models agree on.
- SevereFP ratio (matches) $= \dfrac{\text{Severe FPs (matches)}}{\text{All FPs (matches)}}$
- We hypothesize that the ratio in the [0.9,1] bracket should be lower when the test model is that which uses the custom loss BCE with a severity penalization in the [0.9,1] bracket. Additionally, the ratio should be lower when low recall seeds are used.
- Purpose: Test whether the custom function and low recall characteristic of a function serves to decrease the number of severe FPs that models "agree" on.

Severe FP Risk Under Different Loss Functions (V vs T)

- Models of different groups are compared:
  - Custom Loss BCE with *threshold penalization* in [0.7, 1.0] bracket
  - Custom Loss BCE with *threshold severity penalization* in [0.9, 1.0] bracket
  - Control Model
- **Metrics:**
  1. **Severe among all FPs in** $[0.7, 1.0]$: $r_{0.7\text{–}1.0} = \dfrac{\text{Severe FP}_{[0.7,1.0]}}{\text{All FP}_{[0.7,1.0]}}$.
  2. **Severe among all FPs in** $[0.9, 1.0]$: $r_{0.9\text{–}1.0} = \dfrac{\text{Severe FP}_{[0.9,1.0]}}{\text{All FP}_{[0.9,1.0]}}$.
  3. **Severe FPs per TPs in** $[0.9, 1.0]$: $r_{0.9\text{–}1.0} = \dfrac{\text{Severe FP}_{[0.9,1.0]}}{\text{TP}_{[0.9,1.0]}}$.
- We hypothesize that the ratio of severe FPs in the [0.9, 1.0] bracket is the lowest, since the custom loss function applies weights to the loss function to severe FP cases.
- **Purpose:** verify that the *custom severity + threshold loss* reduces the fraction of severe errors in the high-confidence bracket(s)

7 Models Criteria Explanations

Notice: Model Criteria are chosen based on the analysis above. For example, low recall models are associated with correlation stability in the V and T set. ie pairing models with low recall (as measured in the V set results and a certain correlation), tends to maintain similar pairwise correlation in the T set. Here correlation was measured as the number of times both models make a postive class predictions or the number of times both models make a FP postive class predictions.

Purpose: Test a number of groups of model's performances based on filter combinations. The same combinations (with similar filter metrics) are applied to models over multiple years to asses trends. The idea is to standardize the model selection process.

Notice: Due to compute limitations, the number of model grouping combinations attempted is minimal.

Terminology Notice:

- A "model" represents a combination of LSTM parameters.
- A "seed" of a given model represents a specific weight initialization of the LSTM parameter combination.
- "Model Level Criteria" are aggregated metrics from different seed initializations for a particular combination of LSTM parameters.
- "Seed Level Criteria" are metrics associated with a specific weight initialization for a given model.

Criteria Set 1

- Models are Chosen Based on **Model Level** Criteria

Criteria Set 2

- Models are Chosen Based on **Seed Level** Criteria

Criteria Set 3

- Models are Chosen Based on **Seed and Model Level** Criteria
- High precision models are added as a filter to the seed level set in Set 2

Criteria Set 4

- Models are Chosen Based on **Model Level** Criteria
- High precision models are added as a filter to the model level set in Set 1

Criteria Set 5 and Criteria Set 6

- **Low Recall** model and seed level filter added as additional filter to the criteria in set 3 and 4

Model & Seed Selection Acronyms

I use shorthand acronyms for different model/seed filters. Each acronym corresponds to a set of constraints in the selection function.

NOTE: Filters are applied gradually in order to isolate the effect of individual filters. Models with few filters are not expected to perform well.

Model-level precision bands

$P_{\textbf{mean}}$ = mean positive class precision for all seed in model (same combination of LSTM params but different seed initializations

Mp_H - $P_{\text{mean}} \in [75, 100]$

- Sometimes widened to $[50, 100]$ when combined with other filters to maintain enough models

Mp_M - $P_{\text{mean}} \in [40, 75]$

Mp_L - $P_{\text{mean}} \in [5, 40]$

Mp_ALL - $P_{\text{mean}} \in [5, 100]$

Model-level "severe FP" screens

Rfps_H - $\dfrac{\text{Severe FPs}}{\text{All FPs}} \leq 0.40$

Rfps_L - $\dfrac{\text{Severe FPs}}{\text{All FPs}} \leq 0.20$

unDES__Rfps_H - $0.50 \leq \dfrac{\text{Severe FPs}}{\text{All FPs}} \leq 0.70$

Note: **"unDES"** stands for undesirable, these are filter combinations that i expect to perform poorly

---

Model-level bracket quality (ratio-difference)

Let $r_b = \dfrac{\text{FP}_b}{\text{TP}_b + \text{FP}_b}$, and $\Delta r = r_{\text{Low}} - r_{\text{High}}$.

R_L - $0.10 \leq \Delta r \leq 0.30$

R_H - $\Delta r \geq 0.30$

unDES__R_neg - $\Delta r \leq 0$

Seed-level precision bands

Sp_H - $P_{\text{seed}} \in [75, 100]$

Sp_M - $P_{\text{seed}} \in [40, 75]$

Sp_L - $P_{\text{seed}} \in [1, 40]$

Sp_ALL - $P_{\text{seed}} \in [5, 100]$

Seed-level high-bracket filters

High Bracket (Hp) = positive class predictions with that fall within $p \in [0.7, 1.0]$ bracket

Hb_0fp1tp - Require Severe FP = 0 and TP ≥ 1

Hb_1fp1tp - Require Severe FP ≤ 1 and TP ≥ 1

Hb_tp_minus_fp - $(TP - FP)_{Lb} - (TP - FP)_{Hb} \geq k$ (k=2 or 3)

Other Filter Codes

Sr_L - Low seed recall $S_r \in [0, 30]$

Mr_L - Low model recall $M_r \in [0, 30]$

Sr_H - $S_r \geq 30$

Mr_H - $M_r \geq 30$

Low-recall test cases

- **NOTE** - model combinations are repeated with custom loss functions applied as a filter , these models perform best after the threshold filter is applied
- **NOTE** - Models with a ***** are expected to perform best. These models do seem to perform best with respect to the goal of study. They make a few high precision predictions, or no predictions at all.
- So far, model evaluation of the results in the "(4) All Models Model Performance Assessment" is mostly visual, based on the the distribution of the results - some patterns seem to emerge visually. However, a more robust analysis of the results in the (4) files needs to be conducted.
- Sr_L__Mr__L__Mp_H__Sp_H
- Sr_L__Mr__L__Mp_H__Sp_M *****
- Sr_L__Mr__L__Mp_H__Sp_L
- Sr_L__Mr__L__Mp_H__TH_Sp_ALL__Hb_0fp1tp
- Sr_L__Mr__L__TH_Mp_H__Sp_ALL__Hb_tp_minus_fp *****
- Sr_L__Mr__L__Mp_H__Rfps_H
- Sr_L__Mr__L__Mp_H__Rfps_L *****
- Sr_L__Mr__L__unDES_Mp_H__Rfps_H
- Sr_L__Mr__L__Mp_H__R_L
- Sr_L__Mr__L__Mp_H__R_H *****
- Sr_L__Mr__L__unDES_Mp_H__R_neg

8 Comments and Notes on Inference Results

- **NOTICE:** There are many model groups that are tested with very minimal filters. These models are not expected to perform well, but they are tested to isolate the predictive power of individual filters in a gradual way (ie isolating the effect of one variable).
- The bottom of the (4) files contain the most interesting results. These are results from model groups which all contain custom loss fucntions.
- The predictions made by the model groups starred (*) above behave in line with the goal of this study. The models make selective predictions, with few predictions made per year. Additionally, when the models generally perform poorly for a given ETF on a given year, the * model groups generally make no predictions at all! This is what we would hope for, to avoid making a trade for a severe FP cases.
- The purpose of the (1) analytics file is to explain and uncover the reasons why the model groups behave as they do.

9 IMPROVEMENTS for Next Iteration of Results

Although there are interesting results thus far, the current state of the LSTM architecture can be improved immensely. The goal for this iteration of the project has been to architect model filters, and a standardized model evaluation architecture that can be used as is in further iterations of the project. **Hence, much of the current code and architecture must be recycled - with additions rather than changes - for the next iteration.**

- The compute budget for the project has been minimal thus far (around 90 CAD per month). Hence, some aspects of the experimentation, such as the number of filters that can be applied for model grouping (because you run out of models that meet the criteria as more filters are applied), is limited. This is important because there are many more model filters that can be applied and experimented on (for ex, some of the LSTM parameters (number of epochs, batch size ...) can be experimented on as filters.
- Regarding the code, i have not been particular about the time complexity of the algorithms. There is room for improvement here.
- The current range for the positive class values is set 0.10 (10% returns over the past month) , this is rather high and should be subject to change. The idea here was that setting a high return for the positive class could push the model to be more selective, so that even if it makes a FP prediction, the prediction could still be in the 5% to 10% return range.
- Implementation of feature extraction methods on the explanatory data
- So far, i have assumed that models that generalize well can be spotted via the current val/test metric evaluation architecture. The reasoning is that Model groups that consistently perform well on the test set are those that generalized well. However, a train/val loss (and positive class precision) comparison across train and validation sets should be implemented and tested.
- The models currently make predictions on a rolling basis, with 3 predictions made before retraining. This is subject to experimentation but retraining more often requires more compute for testing.
- The model is currently only tested on two years (due to compute budget constraints). At least two more years of back-testing needs to be added.
- The decisions made in this study are mathematically informed. However, there is more room for mathematical analysis to inform further decisions(such as the weight values chosen in the custom loss functions for example). For the next iteration, I hope to spend less time organizing code and more time diving into the math.

- Some of the code (such as the (2) and (3) ipynb files) can be more efficient and automated. The process in these files is currently more manual to allow for proper model selection given the limited number of models available due to limited compute.
- Current predictions are monthly and predictions over shorter time frames can be experimented with
- Further experimentation with custom loss functions
- There are multiple ETFs that track crude oil spot prices. In this iteration of the project, 2x leveraged ETFs (inverse and non-inverse) were used as predictor variables but there are several other less volatile options
- There is no well thought out trading strategy implemented here yet. After the points above have been implemented, experimenting with trading strategies that make use of financial instruments (such as option contracts) must be tested.
- Further experimentation with ensemble models. For example, experiment with date ranges of explanatory data that is fed to the model. If we ensemble a model that is trained on recent data with a model that is trained on historical data, could this improve the results? Are they more likely to have independent FP predictions?