Step 1: Data Identification

Write 100-200 word paragraph to describe the data, and how the data items can help you to predict the ETF closing price.

For “Percentage Growth”, “Indexed Performance”, "$10k reinvested", “Rolling Performance”, they share the same concept of the change in a value over time - but present the information differently. This data can be informative for investors to get the intuition out of the quantitative data, however, maybe useless since it is in essence deduced from the changing of the combination of columns and price over time, which will be used for ML model for price prediction.

For **“Price History Data”**, this is the most informative data in terms of using ML model for close price prediction since it provide the exact “daily closing price” data of an ETF in the past 3 year we are interested in.
It also reports descriptive statistics like the minimum, maximum, average prices and trading volumes over a 10+ years period, which provides useful context on the investment's range of movements and identification of outlier highs and lows to evaluate the current price level.
"Dividend and Price History" is trivial for price prediction and confusing in the context since the timeframe of the data appears to be misaligned with the typically provided online data.

Step 2: Data Preprocessing

For all the features that you have in your dataset, explain why you are/are not pre-processing it, and what kind of processing that you needed to do using 100-200 words.

As mentioned earlier, a significant portion of the datasets are derived from raw data, such as historical daily prices. Therefore, these datasets require extensive preprocessing, including splitting them into training and testing sets and formatting them to match the desired model's requirements. However, in order to facilitate interpretation, I also perform some basic preprocessing on other datasets to gain a preliminary understanding of their structure.

Regarding the Dividend dataset, the data exhibits considerable overlap and lacks interpretability. Consequently, further processing of this dataset is deemed unnecessary as it would yield trivial results.

Step 3: Model Creation

Deliverables:

Use 100-200 words to explain your reasoning for choosing the model, and potential pros and cons. You must also explain the model structure/layers, and the reasoning behind it.

LSTM is a type of recurrent neural network (RNN) specifically designed to capture long-term dependencies and patterns in sequential data.

1. Model Structure:
   - LSTM layers: LSTM models consist of one or more LSTM layers. These layers have memory cells that can retain information over longer time intervals, allowing the model to capture dependencies over extended periods. We define single layer in this case for more efficient training time, and the number of neuron (denoted as “Units”) will be ranging from 10 to 100.
   - Dense layers: Following the LSTM layers, one or more dense layers are added to refine the learned representations and make predictions. In this case, we add one.

2. Reasoning:
  The main reason is to capture a longer dependencies, and it is useful for time series forecasting due to its RNN nature.

Pros:
- Ability to capture long-term dependencies and temporal patterns.
- Effective handling of sequential data.
- Suitable for capturing nonlinear relationships and complex patterns.
- Flexibility to handle multivariate time series data.

Cons:
- Complexity: computationally expensive compared to simpler models
- Overfitting: especially when the dataset is small or noisy. Regularization techniques and careful model tuning are needed to mitigate this.
- Interpretability: as a deep learning model, it is considered black-box model, making it challenging to interpret the specific features or factors driving the predictions.

Step 4: Hyperparameter Tuning

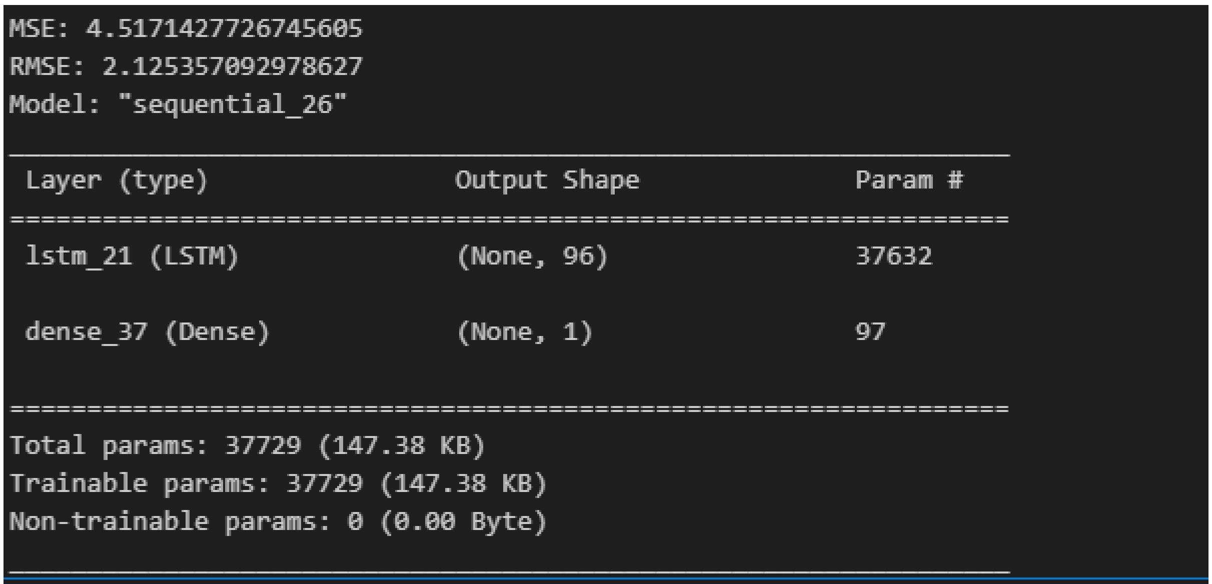
Since I failed to implement the auto tuning tool by Keras, I manually adjusted **the number of units, the batch size, and the epochs** to gain some insight.

We test vanilla model only (single layer):

Units	Batch	Epochs	MSE/RMSE
32	1	15	MSE: 9275.849609375 RMSE: 96.31121227237772
32	30	15	MSE: 1085.2200927734375 RMSE: 32.94267889491438

32	30	50	MSE: 1675.1051025390625 RMSE: 40.92804787110011
64	30	15	MSE: 1833.9105224609375 RMSE: 42.8241815153651
50	5	15	MSE: 67.43972778320312 RMSE: 8.212169493087872
64	5	15	MSE: 22.395854949951172 RMSE: 4.732425905384169
96	5	15	MSE: 4.5171427726745605 RMSE: 2.125357092978627

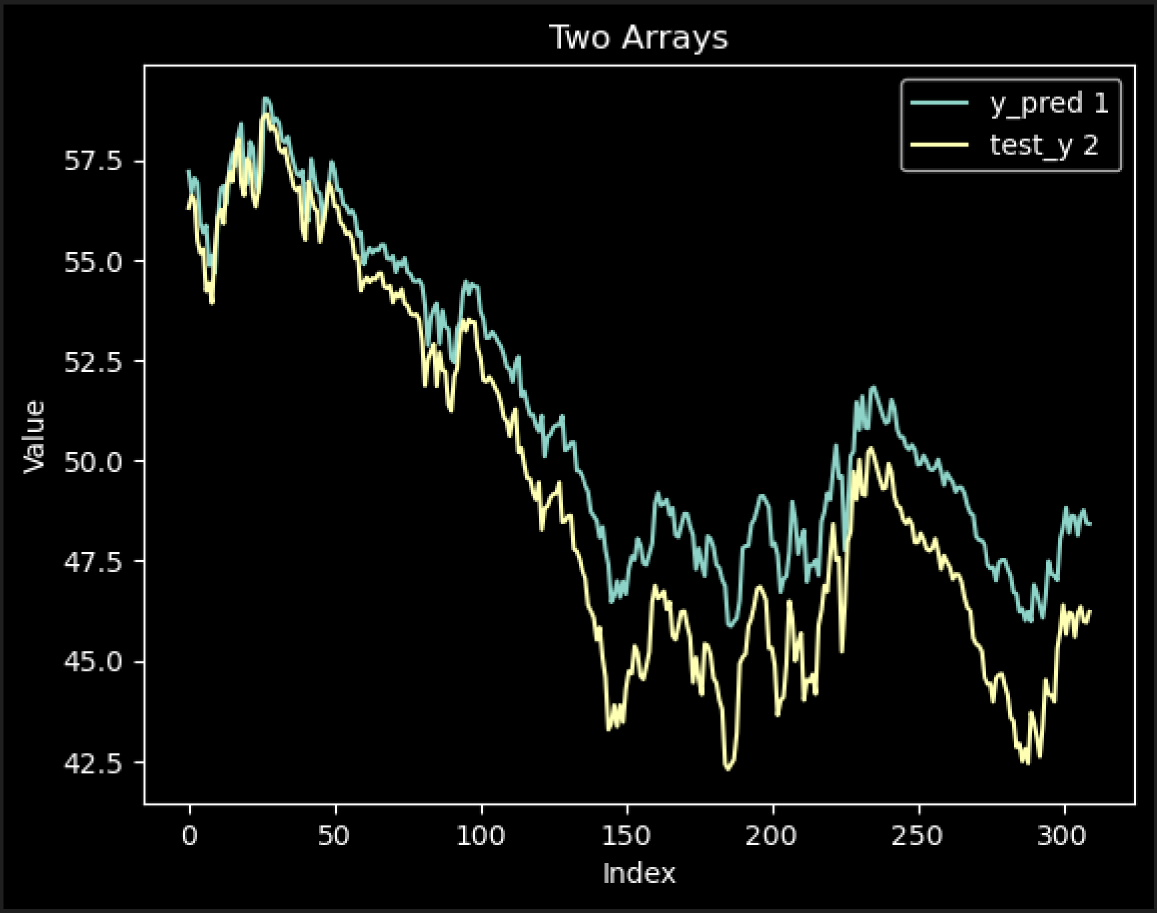
We can observe that there is a balance between the batch size and the size of epochs. Still, many tuning methods can be further tested such as learning rate, the number of layer, etc.



```
MSE: 4.5171427726745605
RMSE: 2.125357092978627
Model: "sequential_26"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_21 (LSTM)              (None, 96)                37632

 dense_37 (Dense)            (None, 1)                 97

=================================================================
Total params: 37729 (147.38 KB)
Trainable params: 37729 (147.38 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```



Step 5: Model Evaluation & Interpretation

See notebook