
OB Oracle
Group 6

Our Project Goals

- Using machine learning models on a high frequency trading (HFT) crypto order book to predict price direction and price levels.
- Develop a strategy trading bot to place mock trades on predicted outcomes from ML models and compare actual versus realized strategy returns.

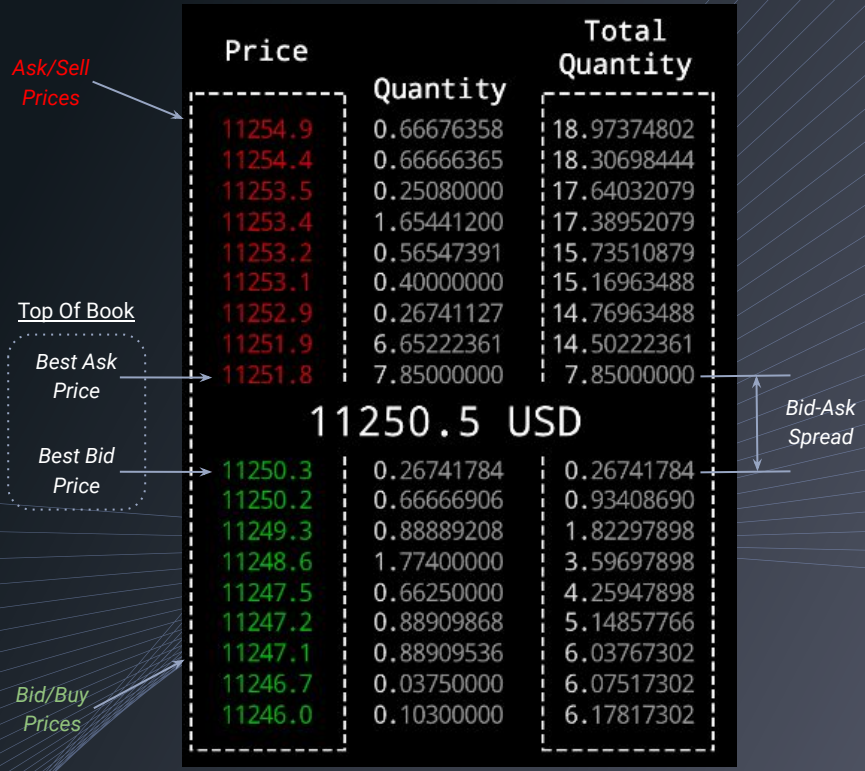
Agenda

1. Order Book (OB) Overview
2. The Selected ML Models
3. Data Prep and Feature Selection
4. Our Approach
5. Results and Conclusions
6. Next Steps

1. Understanding
Order Book Overview

What is an Order Book?

- A Limit order is an order you place on the order book with a specific limit price
- Top of Book represents the highest bid and the lowest ask that time.
- A bid-ask spread is the amount by which the ask price exceeds the bid price for an asset in the market.
- Market orders let you purchase instantly at best price currently available.
- Mid-price is the price between the best price of the sellers offer price and best price of the buyers bid price.
- Liquidity refers to how rapidly shares of a stock can be bought or sold without substantially impacting the stock stock price.



Order Book Data Source (Kaggle):

Crypto Assets:
- Bitcoin (BTC)
- Ethereum (ETH)
- Cardano (ADA)

Content:
- 12 days of limit order book data
- 15 best bid / ask price **levels**
- 5 min data / 1 min data /1 sec data

https://www.kaggle.com/datasets/martinsn/high-frequency-crypto-limit-order-book-data

High Frequency Crypto Limit Order Book Data

Bitcoin, Ethereum and Cardano

```
Crypto PROVIDED order book features/columns (_x represent order book level)

    * midpoint = the midpoint between the best bid and the best ask
    * spread = the difference between the best bid and the best ask

    * bids/asks_distance_x = the distance of bid/ask level x from the midprice in %

    * bids/asks_limit_x = volume (= price * quantity) of orders at bid/ask level_x
    * bids/asks_notional_x = (asks_limit_notional_x - asks_market_notional_x - asks_cancel_notional_x)
```

Order Book Feature Engineering:

system_time	midpoint	bids_distance_0	bids_price_0	bids_limit_quantity_0	bids_limit_cum_quantity_0	bids_limit_notional_0	bids_limit_cum_notional_0	asks_distance_0	asks_price_0	asks_limit_quantity_0	asks_limit_cum_quantity_0	asks_limit_notiona
2021-04-07 11:37:59.055697+00:00	1.17255	-0.000384	1.1721	2099.999925	2099.999925	2461.409912	2461.409912	0.000384	1.1730	8007.340453	8007.340453	9392.610
2021-04-07 11:42:59.055697+00:00	1.18390	-0.000591	1.1832	4633.062748	4633.062748	5481.839844	5481.839844	0.000591	1.1846	20004.229078	20004.229078	23697.009
2021-04-07 11:47:59.055697+00:00	1.17830	-0.000594	1.1776	2798.998045	2798.998045	3296.100098	3296.100098	0.000594	1.1790	500.000000	500.000000	589.500
2021-04-07 11:52:59.055697+00:00	1.1											069
2021-04-07 11:57:59.055697+00:00	1.1											269
2021-04-07 12:02:59.055697+00:00	1.1											050
2021-04-07 12:07:59.055697+00:00	1.1											250
2021-04-07 12:12:59.055697+00:00	1.1											300
2021-04-07 12:17:59.055697+00:00	1.1											570
2021-04-07 12:22:59.055697+00:00	1.1											740

New DERIVED order book features/columns developed in this notebook/python script

* bids/ask_price_x = the price at bid/ask level_x
  = midpoint * (1 + distance) (wehere distance is represented as %)

* bids/asks_limit_quantity_x = quantity (= bids/asks_limit/market/cancel_notional_x / bids/ask_price_x) of orders at bid/ask level_x

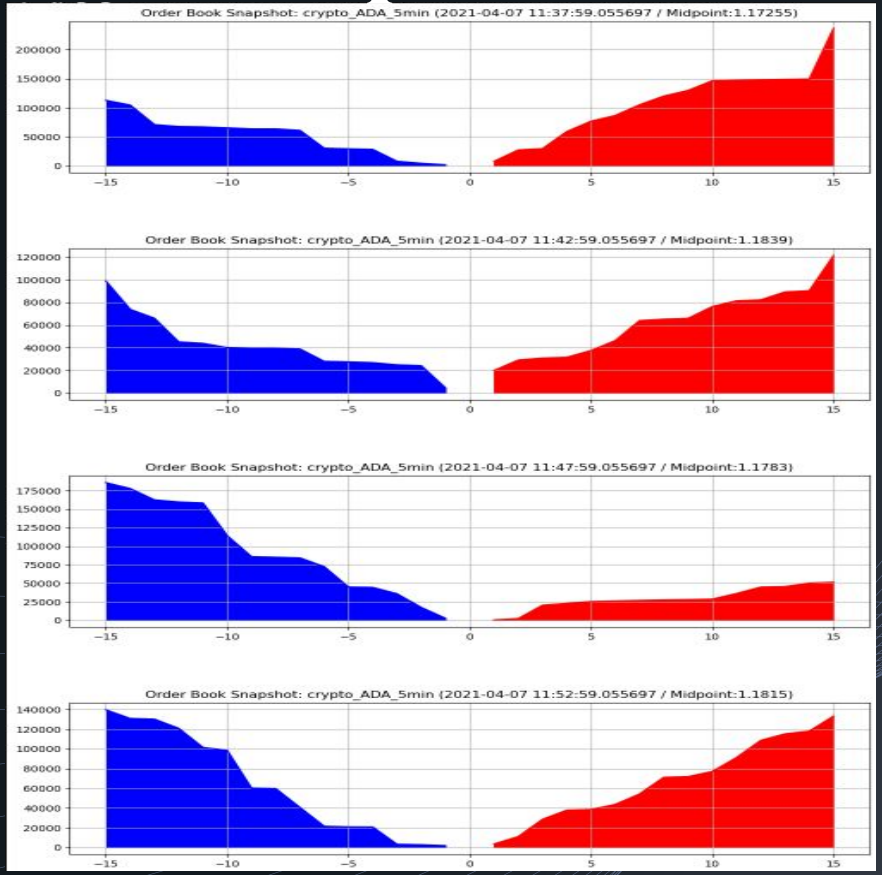
* bids/asks_cum_quantity_x = Cululative sum of quantities - i.e. bids/asks_limit/market/cancel_quantity_x
  Example. cum_quantity_0 = quantity_0
  cum_quantity_1 = quantity_0 + quantity_1
  cum_quantity_2 = quantity_0 + quantity_1 + quantity_2
  ...

* bids/asks_cum_notional_x = Cululative sum of notionals - i.e. bids/asks_limit/market/cancel_notional_x
  Example. cum_notional_0 = notional_0
  cum_notional_1 = notional_0 + notional_1
  cum_notional_2 = notional_0 + notional_1 + notional_2
  ...

* bid_ask_imbalance_limit_x = Bid Ask Imbalance = (bid notional / (bid notional + ask notional))
  = bids_notional_x / bids_notional_x + asks_notional_x

Order Book Perspectives

- BID LEVELS
- ASK LEVELS



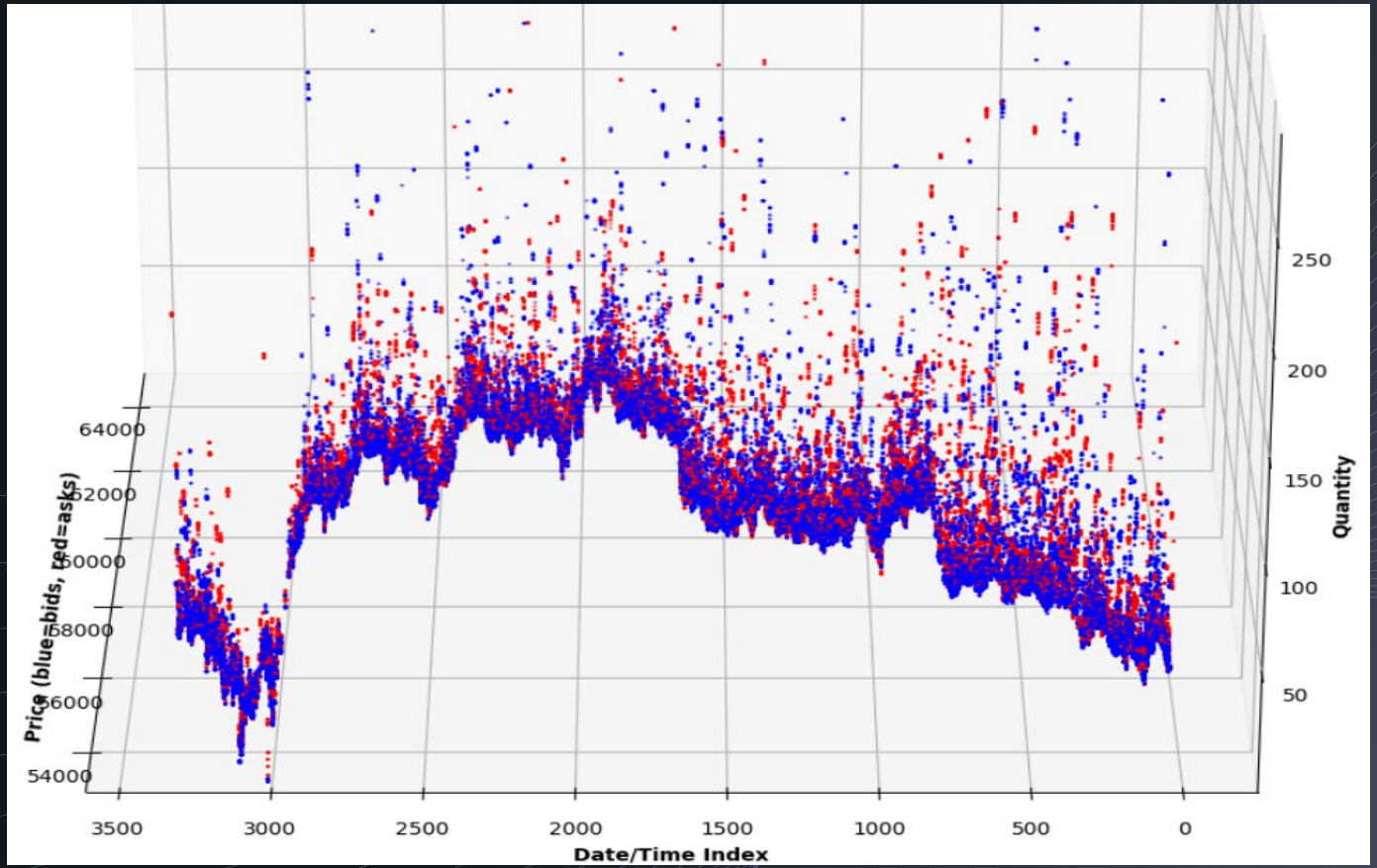
5min
Snapshots

CumQty

Price

Order Book Perspectives

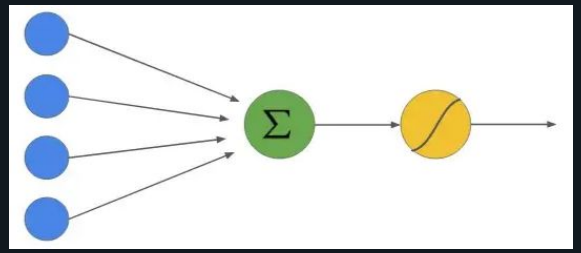
- BID LEVELS
- ASK LEVELS



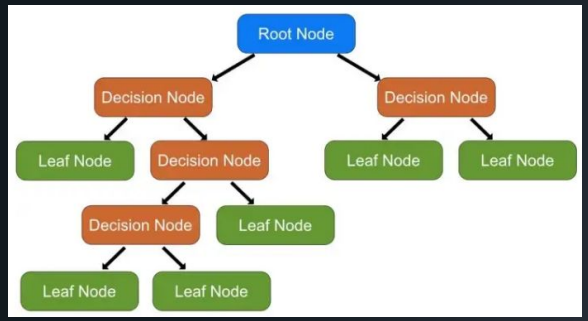
2. The Selected ML Models

ML Supervised Learning: Models

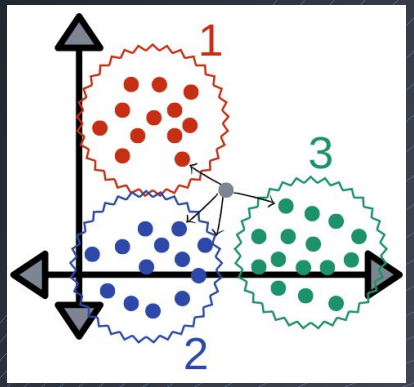
Logistic Regression



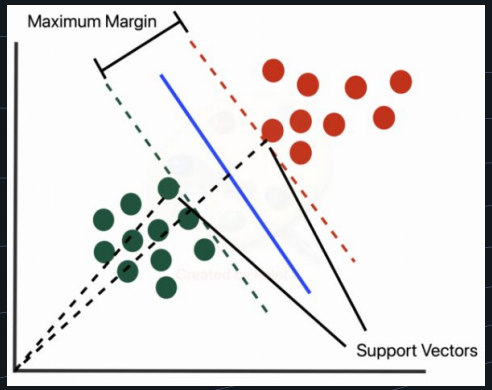
Decision Trees



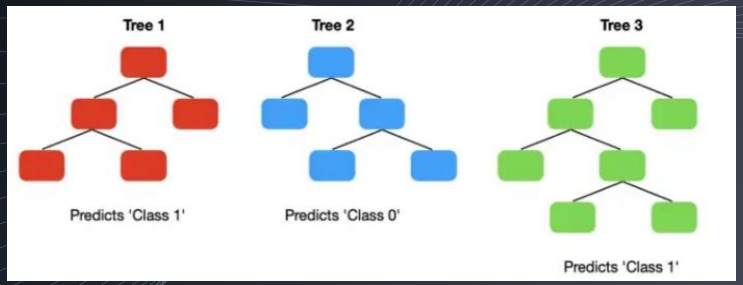
KNN



SVM

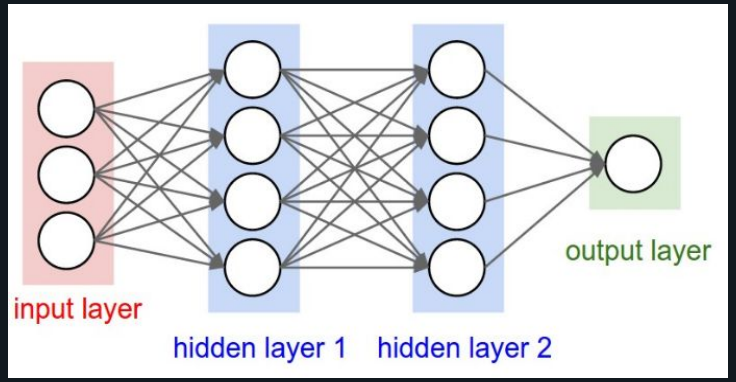


Random Forest

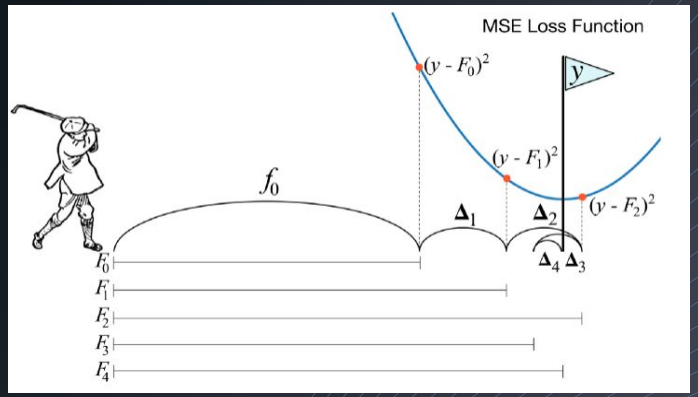


ML Neural Networks: Models

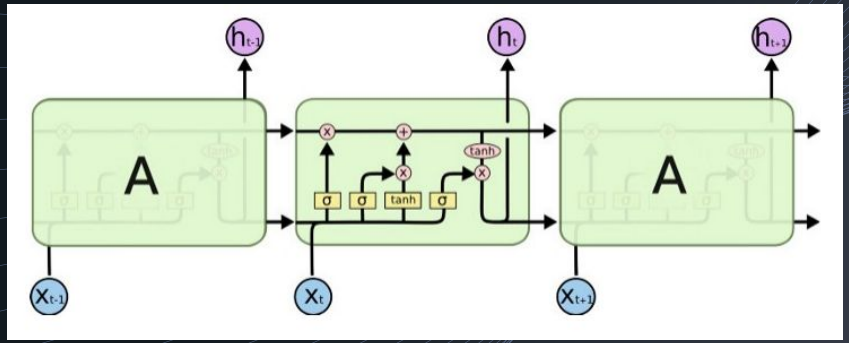
Sequential/Dense



Gradient Boosting*



LSTM



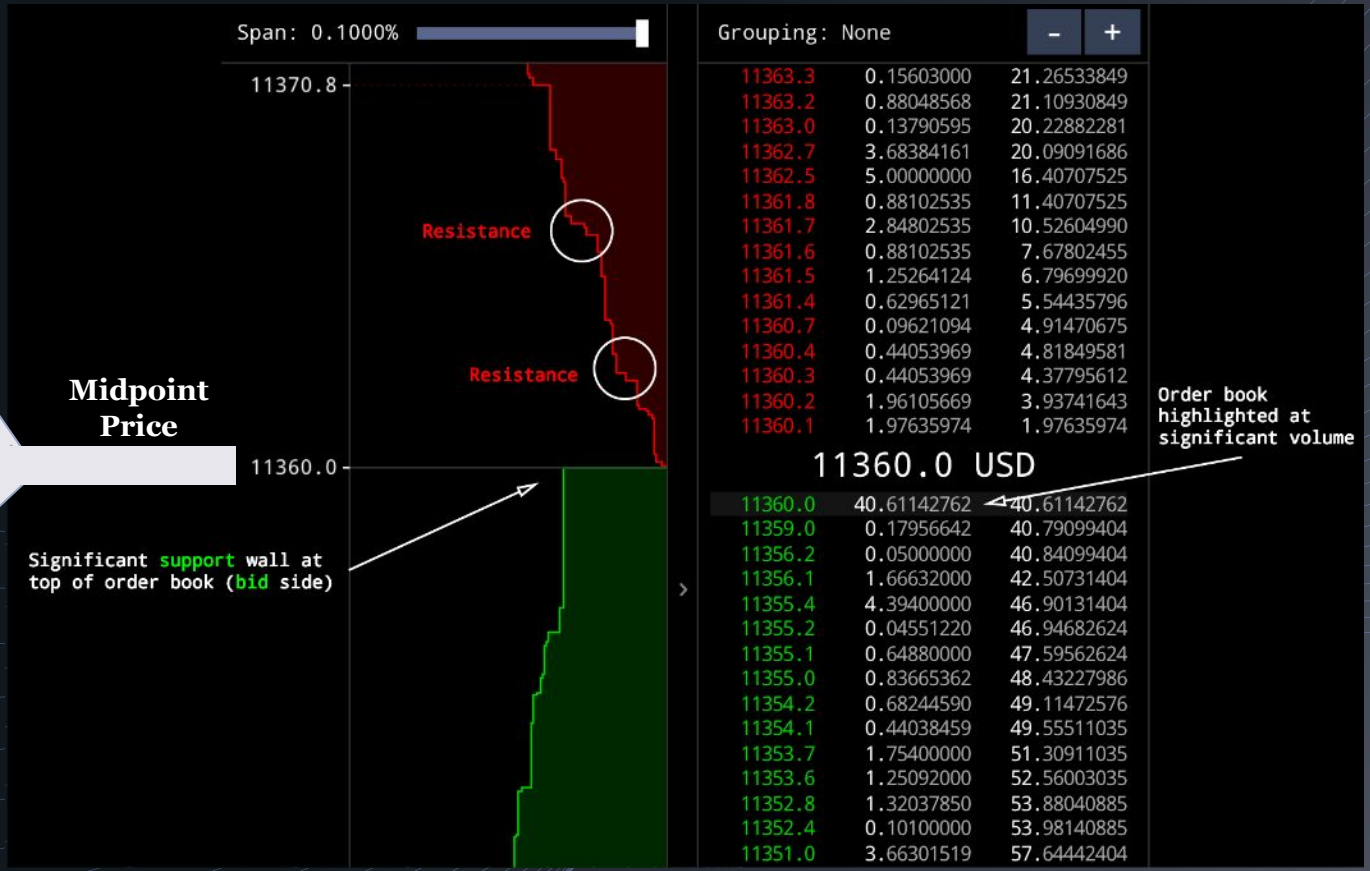
ML Supervised Learning: Classification

ML Supervised Learning: Regression

Predict Price = X

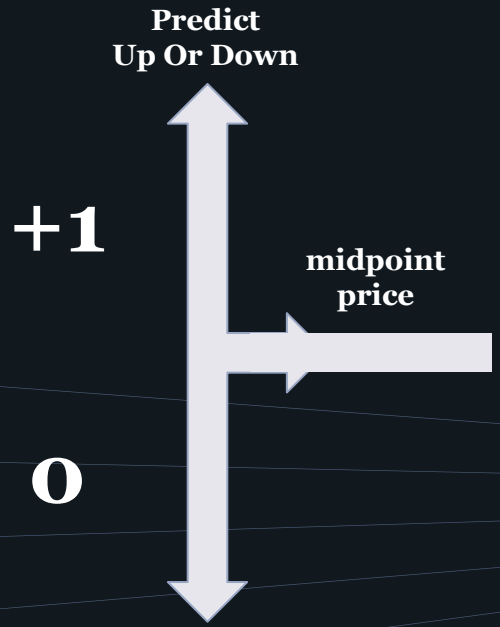
Midpoint Price

Predict Price = Y



3. Data Prep and Feature Selection

Data Preparation: Generate Target Signal



system_time	Signal	midpoint
2021-04-07 11:37:41.122161+00:00	1.0	55899.995
2021-04-07 11:42:41.122161+00:00	0.0	56111.655
2021-04-07 11:47:41.122161+00:00	1.0	56100.015
2021-04-07 11:52:41.122161+00:00	1.0	56163.445
2021-04-07 11:57:41.122161+00:00	0.0	56229.995
2021-04-07 12:02:41.122161+00:00	0.0	56119.455
2021-04-07 12:07:41.122161+00:00	0.0	56038.495
2021-04-07 12:12:41.122161+00:00	1.0	55854.915
2021-04-07 12:17:41.122161+00:00	0.0	56077.065
2021-04-07 12:22:41.122161+00:00	1.0	56037.000

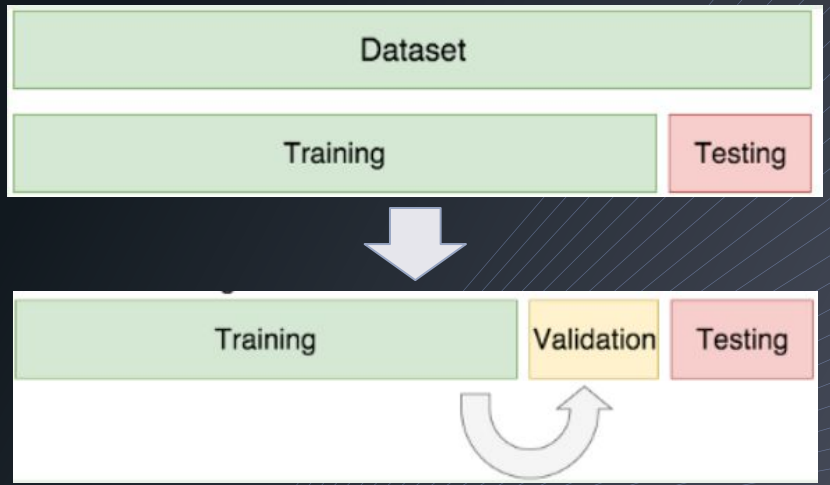
Price **Up** (Signal = **1.0**)

Price **Down**(Signal = **0.0**)

Price **Up** (Signal = **1.0**)

Data Preparation: Training/Test Data Set

- **Training Set**: Complete training set that will be applied to the ML models to train and fit the data.
- **Validation Set**: Divide training set into a train set and validation set. Based on validation test results, the model can be trained (e.g. changing parameter classifiers). This will help us get the most optimized model.
- **Testing Set**: Data set used to perform blind predictions and evaluations on the ML model.



- **Data Partition Permutations:**
  - Training Set: 50% → 90%, 10% increments
  - Validation Set: 0% → 30%, 5% increments
  - Testing Set: 10% → 50%, 10% increments

Data Preparation: Feature Selection / Cleanup

```
# Level selection
# Select number of orderbook levels to include in feature set (max = 15)
MAX_ORDER_BOOK_LEVELS = 15   # Do not modify
NUM_ORDER_BOOK_LEVELS = 15
```

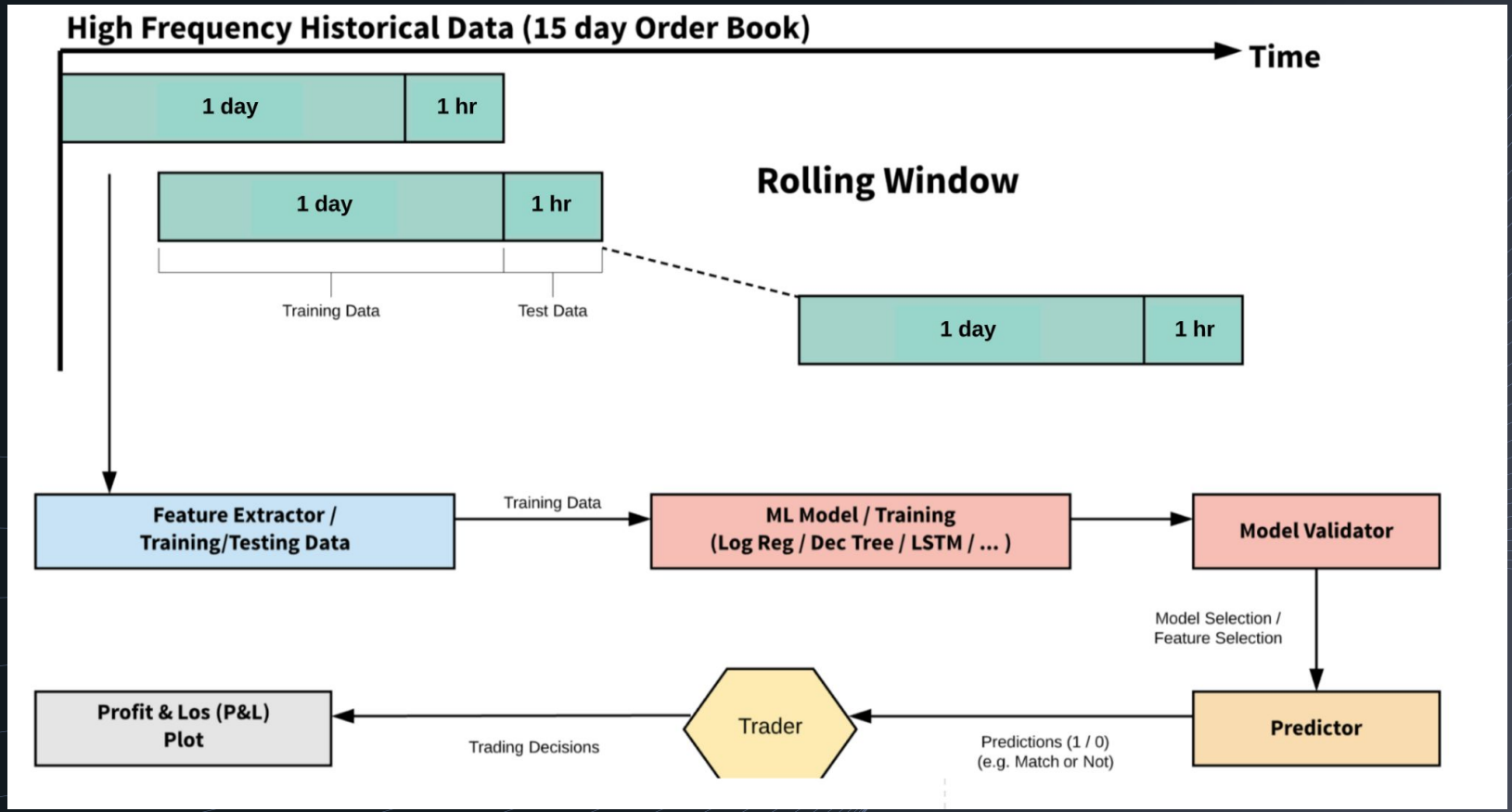
```
# Select crypto assets interested in evaluating
INCLUDE_BTEC = True
INCLUDE_ETH = True
INCLUDE_ADA = True

# Select the orderbook timeframe's interested in
INCLUDE_5MIN_DATA = True
INCLUDE_1MIN_DATA = True
INCLUDE_1SEC_DATA = False
```

```
# Optional removing of feature/columns from provided crypto order book
# Remove cancel features/columns
REMOVE_CANCEL_FEATURES = True
# Remove market features/columns
REMOVE_MARKET_FEATURES = True
# Remove price distance feature/column
REMOVE_PRICE_DISTANCES = False
# Remove notional feature/column
REMOVE_FINAL_NOTIONAL = True
```

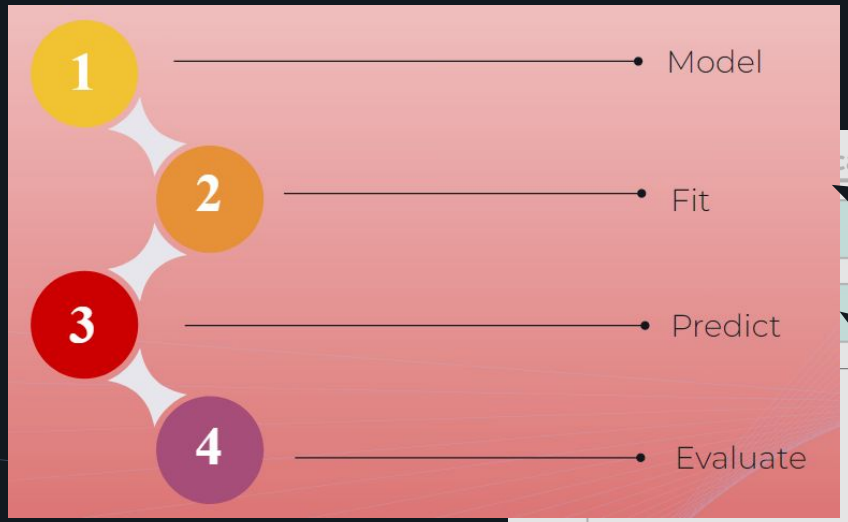
```
# Select which "derived" features to include in ML Model feature set

# Optional NEW derived feature/columns that can be added to dataframe order book
# Option to add aboslute price levels to feature set
# (default dataframe shows delta price with mid price = distance)
ADD_ABSOLUTE_PRICE_LEVELS = True
# Option to add quantity at each level
ADD_QUANTITY_FEATURE = True
# Option to add CUMULATIVE quantity "across" levels
ADD_CUM_QUANTITY_FEATURE = True
# Option to add CUMULATIVE quantity "across" levels
ADD_CUM_NOTIONAL_FEATURE = True
# Option to add bid and ask IMBALANCE
ADD_BID_ASK_IMBALANCE = True
# ToDo: ADD MORE FEATURES (e.g. logarithmic returns)
```

4. Our Approach

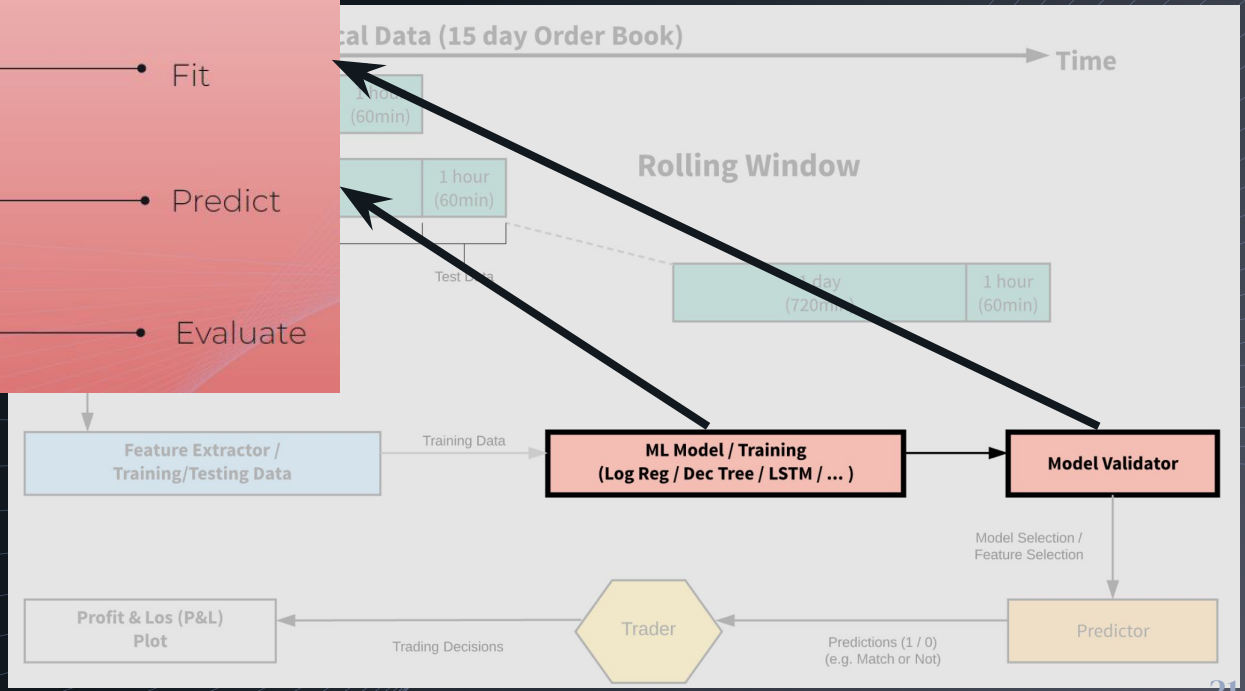
Framework for ML Limit Order Strategy Submission



ML Model Training / Validation Steps



- Logistic Regression
- Decision Trees
- KNN
- Random Forest
- SVM
- Sequential/Dense
- LSTM
- Gradient Boosting



ML Neural Network Model Validation

ML Parameter Permutations:

→ activation function = *{relu, sigmoid, exp, swish}*

→→ # hidden layers = *{1, 2, 3}*

→→ → # nodes, hidden layer 1 = *{32 : 32 : 128}*

→→ → → # nodes, hidden layer 2 = *{16 : 16 : 64}*

→→ → → → # nodes, hidden layer 3 = *{4 : 4 : 16}*

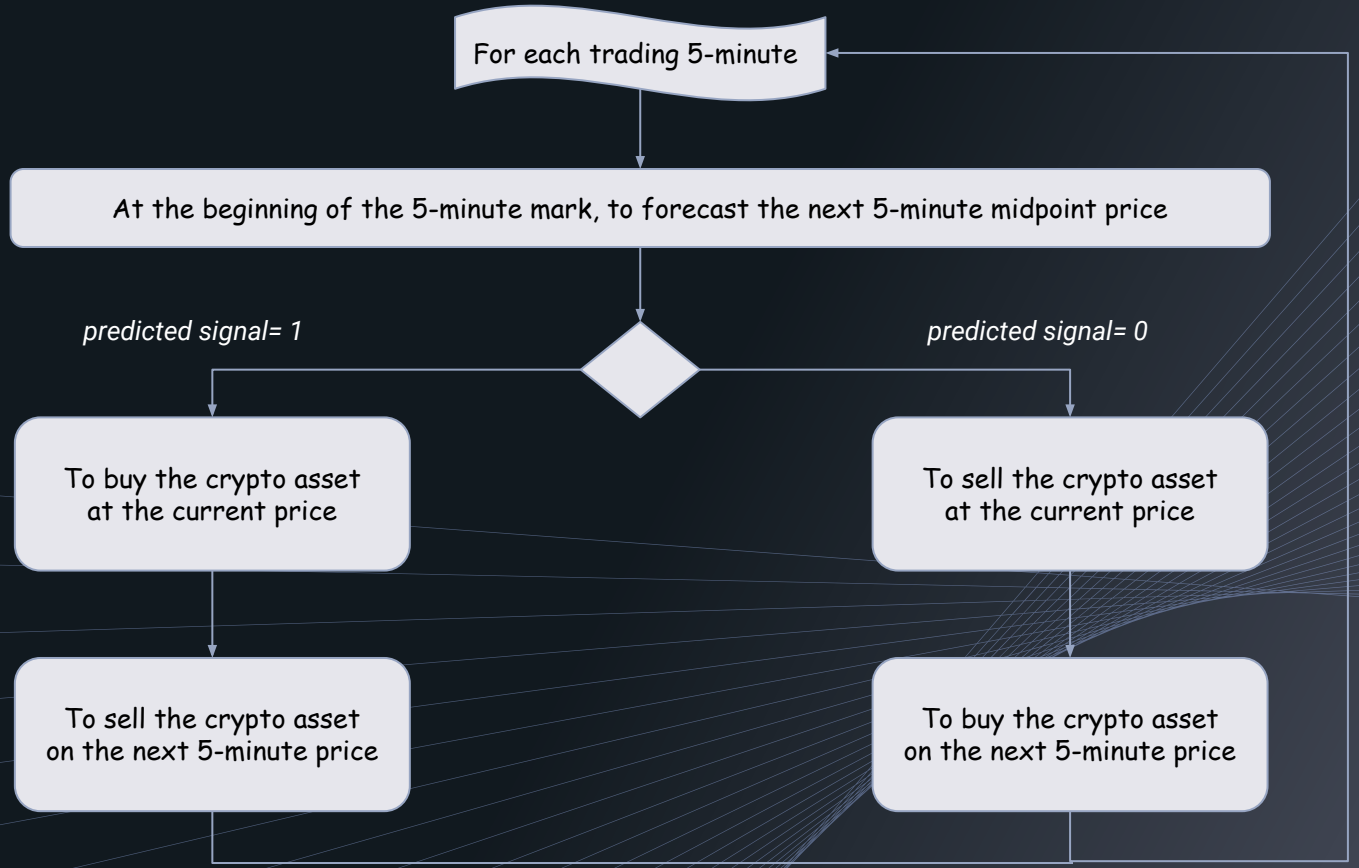
→→→→→→→ # of epochs = *{50 : 50 : 200}*

Total Permutations = 4096 runs

Keeping track of model parameters that provides the best accuracy performance

```
Running Permuation #27
        # Input Features = 199
         Use Model Activation Function = relu
        # Hidden Layers = 3
        # Layer 1 Nodes = 96
        # Layer 2 Nodes = 48
        # Layer 3 Nodes = 8
        # of Output Nuerons = 1
        # Epochs = 150
                        ***                     Permutation:27 ==> Accuracy: 0.5231316685676575 ***
                        *** Best Running Results: Permutation:7 ==> Accuracy: 0.5338078141212463 ***
Running Permuation #28
        # Input Features = 199
         Use Model Activation Function = relu
        # Hidden Layers = 3
        # Layer 1 Nodes = 96
        # Layer 2 Nodes = 48
        # Layer 3 Nodes = 8
        # of Output Nuerons = 1
        # Epochs = 200
                        ***                     Permutation:28 ==> Accuracy: 0.5397390127182007 ***
                        *** Best Running Results: Permutation:28 ==> Accuracy: 0.5397390127182007 ***
Running Permuation #29
        # Input Features = 199
         Use Model Activation Function = relu
        # Hidden Layers = 3
        # Layer 1 Nodes = 96
        # Layer 2 Nodes = 48
        # Layer 3 Nodes = 13
        # of Output Nuerons = 1
        # Epochs = 50
                        ***                     Permutation:29 ==> Accuracy: 0.4922894537448883 ***
                        *** Best Running Results: Permutation:28 ==> Accuracy: 0.5397390127182007 ***
```

Strategy Trading Bot Flowchart (Classification Model)

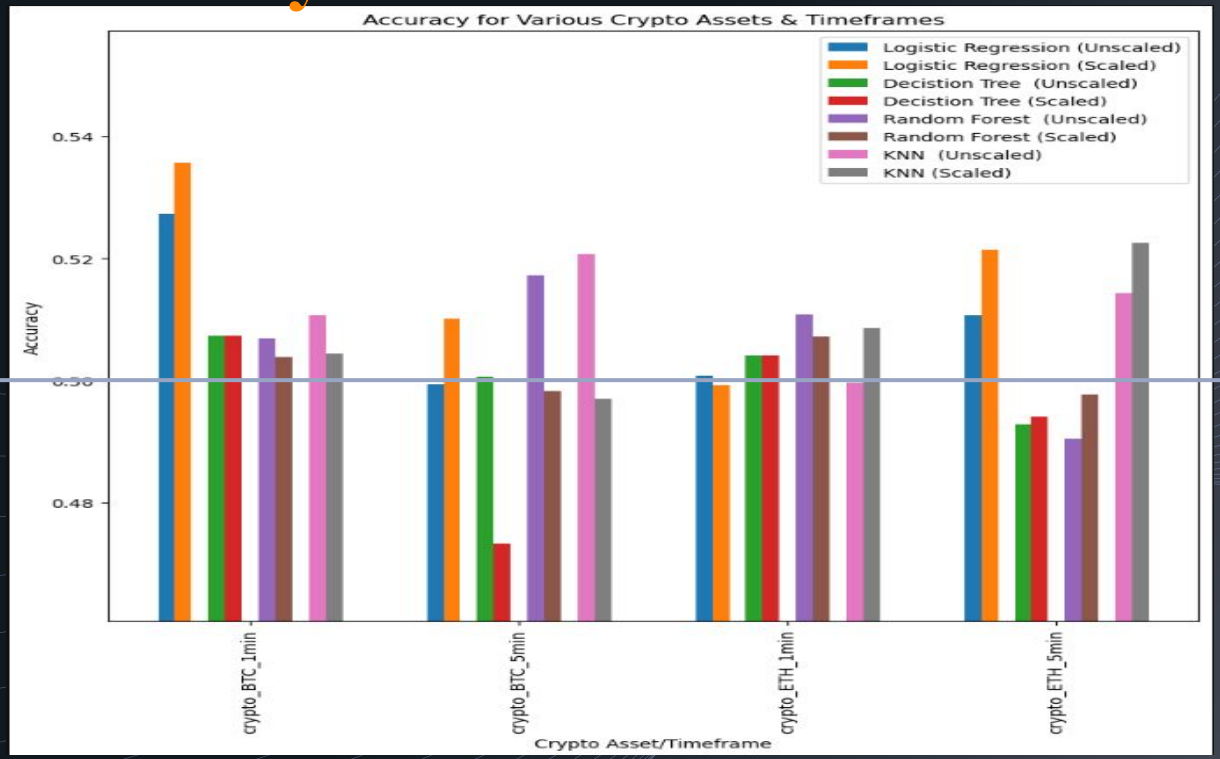


5. Results and Conclusions

ML Supervised Learning Model Results

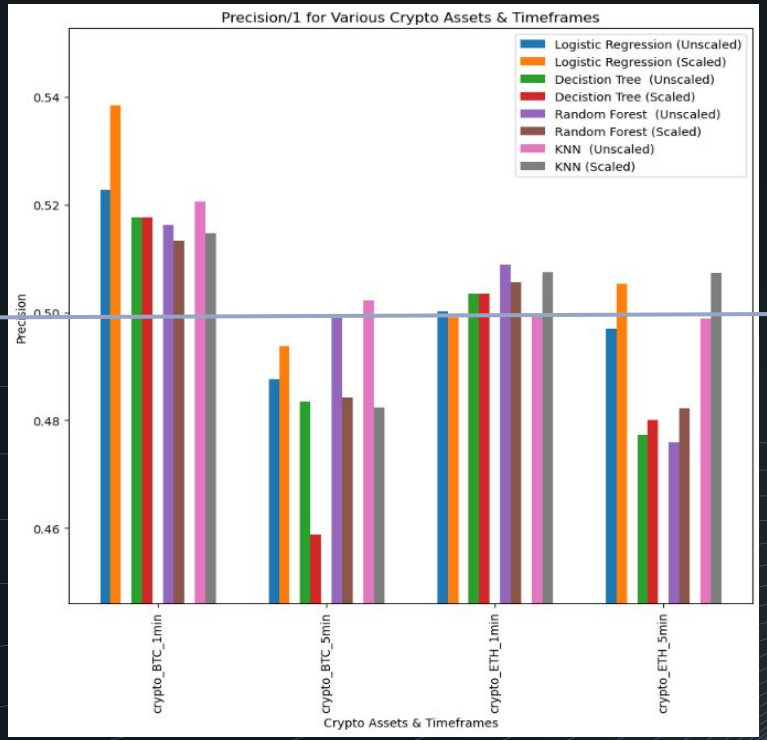
Accuracy

50%



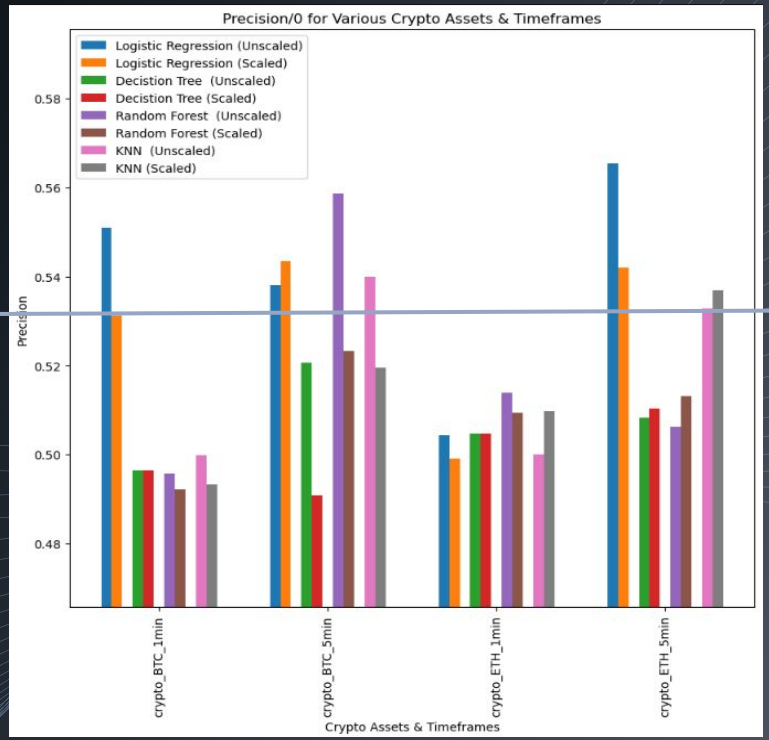
ML Supervised Learning Model Results

Precision-1



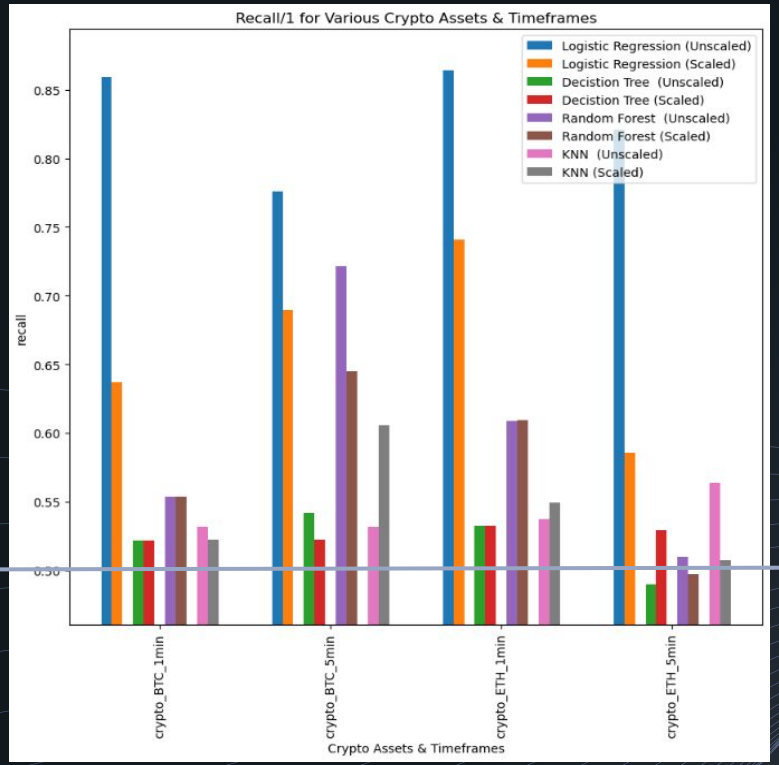
50%

Precision-0

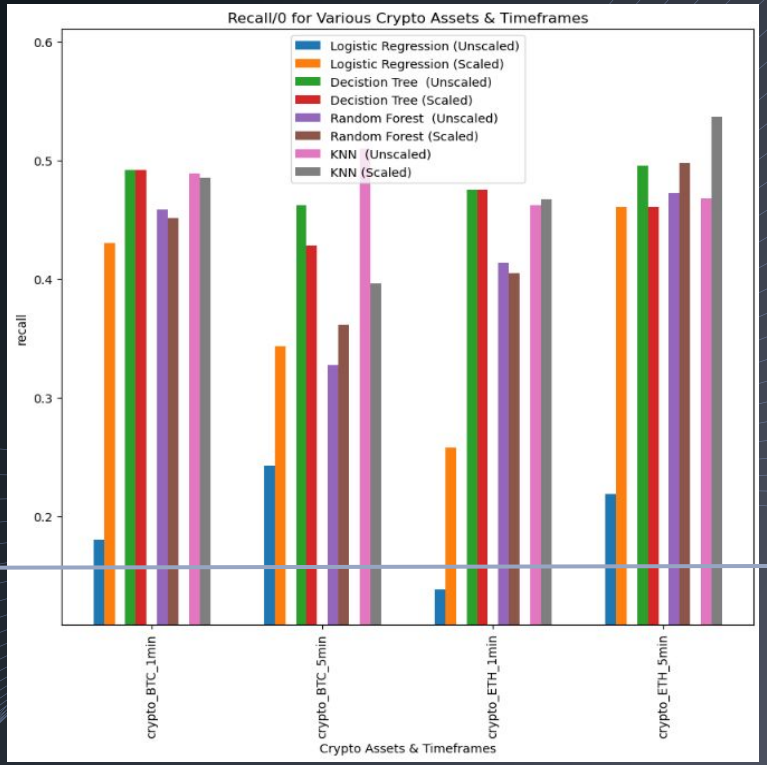


ML Supervised Learning Model Results

Recall-1

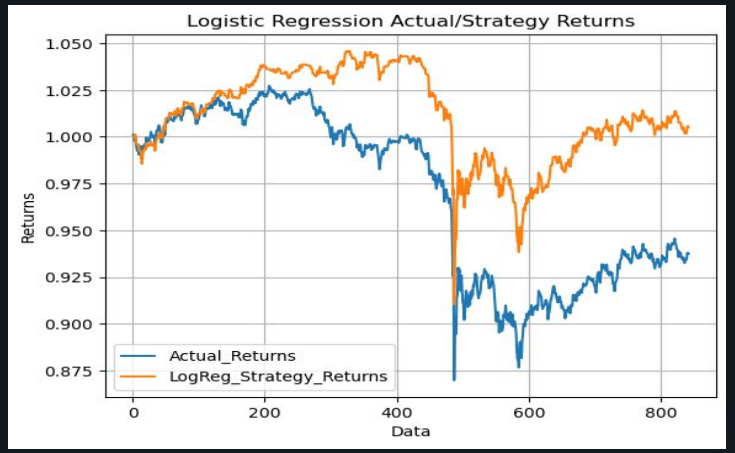


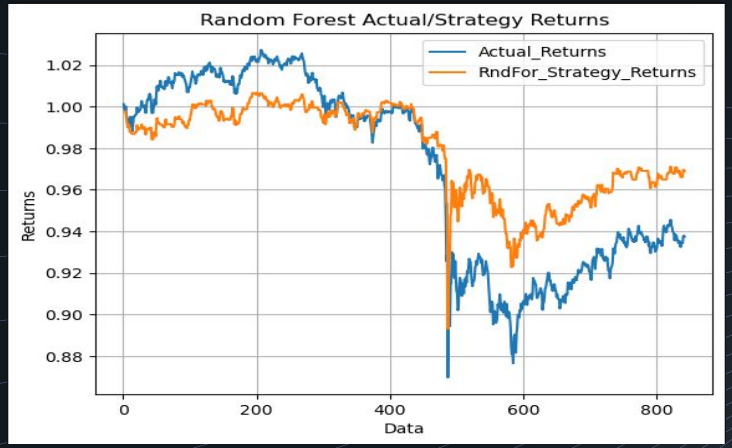
Recall-0

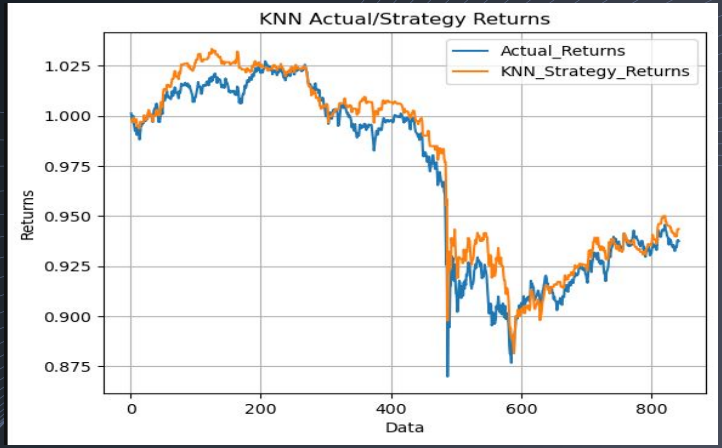


50%

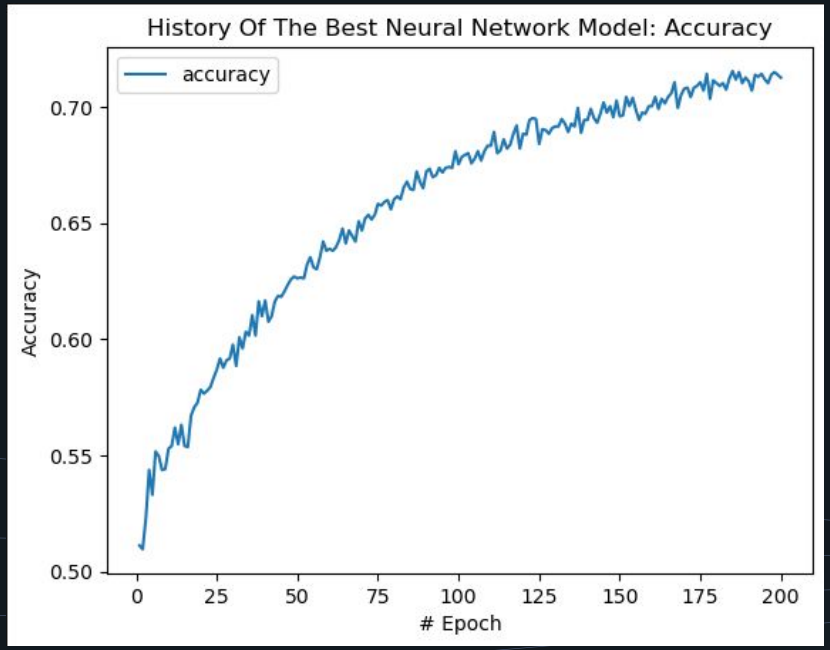
ML Supervised Learning Strategy Bot Results

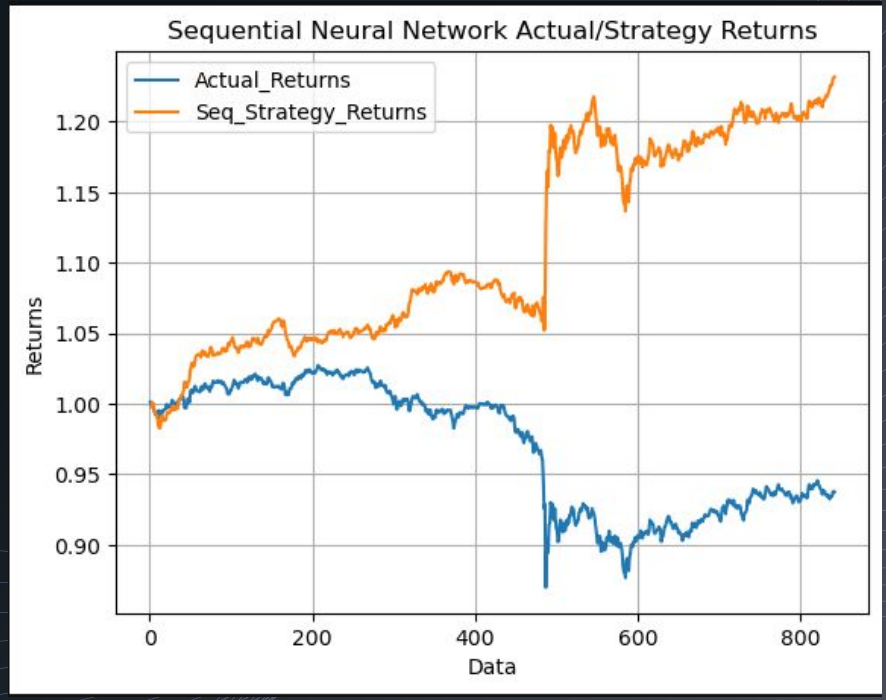






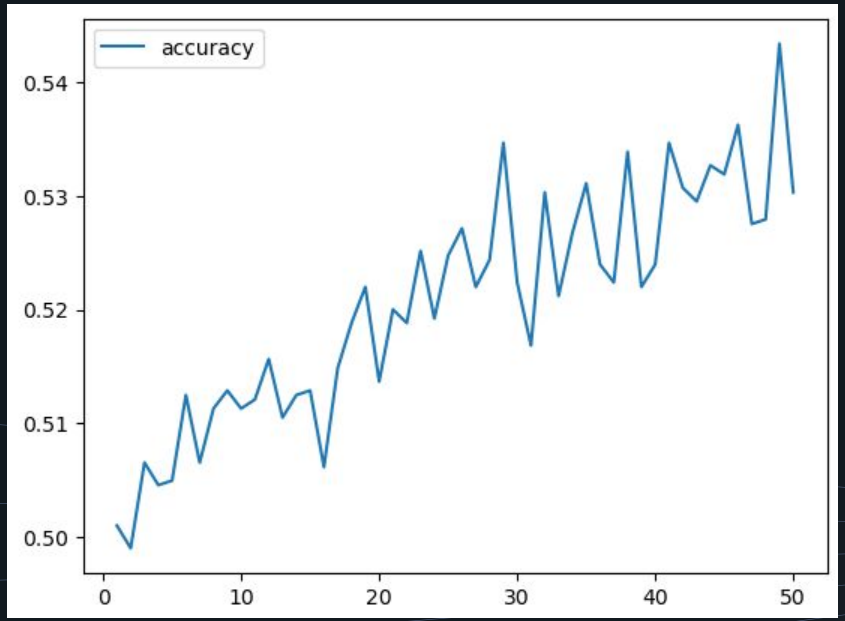
ML Neural Network Dense Strategy Bot Results

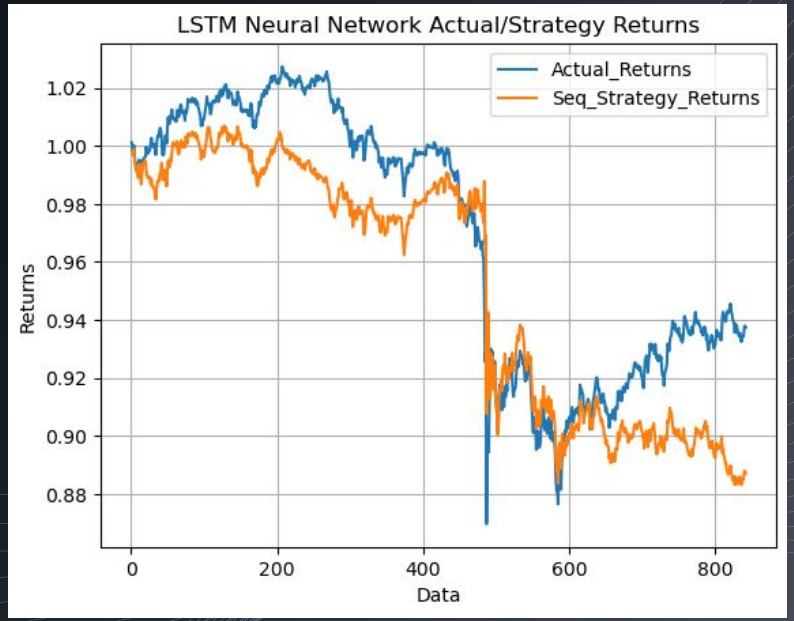





```
*** Final Best Results: Permutation:20 ==> Accuracy: 0.5397390127182007 ***
```

ML Neural Network LSTM Strategy Bot Results






```
Epoch 50/50
79/79 [==============================] - 1s 12ms/step - loss: 0.6890 - accuracy: 0.5303
```

6. Next Steps

Hurdles

- Data acquisition
- Model accuracy
- Binary +/-1 target signals ⟶ binary 1/0 target signals.

Further Analysis

- Complete neural networks “price prediction” for project submission.

- Additional ML models

- Additional feature engineering.

Thank you

https://github.com/lisetlopez/project2