Presented by: Colin Moughton

Date: Dec 24

Rev. 1

Project Definition

Title: FT100 predictive deep learning stock trading model

The project goal was to create a predictive model capable of identifying trading opportunities. The model operates on historical daily stock price data that has the data features ‘open’, ‘high’, ‘low’, ‘close’ and ‘volume’ (OHLCV) for stocks within the FT-100 index. This financial index is made up of the largest 100 public companies trading on the London Stock Exchange.

The machine learning (ML) software developed to achieve this goal is called ‘Ginger’. It runs as a FastAPI web application. These slides aim to provide an overview of what Ginger is and how it works. The software is available on GitHub and can easily be run using Docker.
Ginger software available at: https://github.com/colinmoughton/Ginger

Daily OHLCV data

- ‘OHLCV’ stands for ‘open’, ‘high’, ‘low’, ‘close’ & ‘volume’.
- The opening price is the price the stock starts trading when the market opens.
- The high price is the highest price the stock reaches over the day.
- The low price is the lowest price the stock reaches over the day.
- The close price is the price the stock finally trades at the end of that day.
- Volume is the amount of shares traded in the day.
- It looks like this

date	Open	High	Low	Close	Volume
2018-01-02	681.0	691.4	681.0	684.2	3119191.0
2018-01-03	686.8	688.2	681.0	686.0	3584812.0
2018-01-04	678.6	679.0	662.2	665.2	5992264.0
2018-01-05	663.2	675.4	663.2	674.8	3876711.0
2018-01-08	675.6	685.2	675.6	680.6	4334152.0
...	...	...	...	...	...
2024-10-21	432.8	436.0	428.2	428.6	1478519.0
2024-10-22	425.0	426.4	418.8	421.4	4141313.0
2024-10-23	421.4	423.2	414.4	420.0	2323142.0
2024-10-24	419.4	422.6	417.8	418.8	1289017.0
2024-10-25	418.6	419.8	414.8	415.6	1758057.0

1723 rows × 5 columns

A time series database

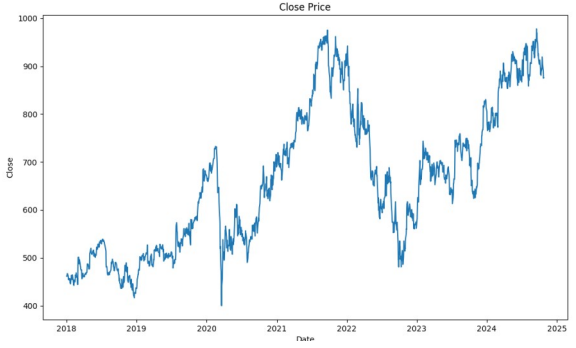
- Time series history data was downloaded for the 100 stocks in the FT 100 index.
- They were stored in an sqlite database
- The stock data for each can be explored by clicking the hyperlinks to show the graphed time history.

Stock Details for HWDN

ID	Stock Name	Start Date	End Date	Number of Records
3	HWDN	2018-01-02	2024-10-24	1722

Price Graphs

Close Price



Stock List

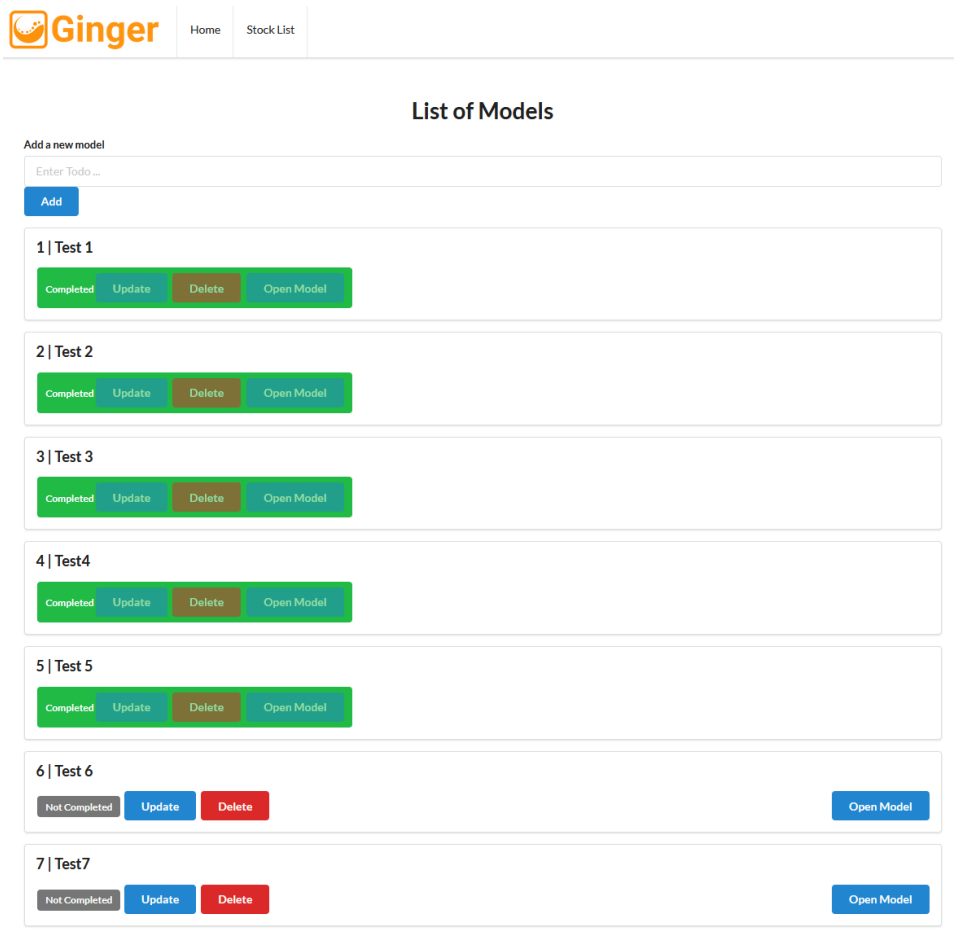
Show entries Search:

ID	Stock Name	Start Date	End Date	Number of Records
1	SN	2018-01-02	2024-10-25	1723
2	SPX	2018-01-02	2024-10-24	1722
3	HWDN	2018-01-02	2024-10-24	1722
4	PSON	2018-01-02	2024-10-23	1721
5	ANTO	2018-01-02	2024-10-23	1721
6	CPG	2018-01-02	2024-10-23	1721
7	LMP	2018-01-02	2024-10-23	1721
8	GSK	2018-01-02	2024-10-23	1721
9	SSE	2018-01-02	2024-10-24	1722
10	MNG	2019-10-21	2024-10-25	1267
11	LAND	2018-01-02	2024-10-23	1721
12	BP	2018-01-02	2024-10-25	1723
13	FCIT	2018-01-02	2024-10-23	1723
14	ADM	2018-01-02	2024-10-25	1723
15	IMB	2018-01-02	2024-10-24	1722

Showing 1 to 15 of 100 entries Previous 1 2 3 4 5 6 7 Next

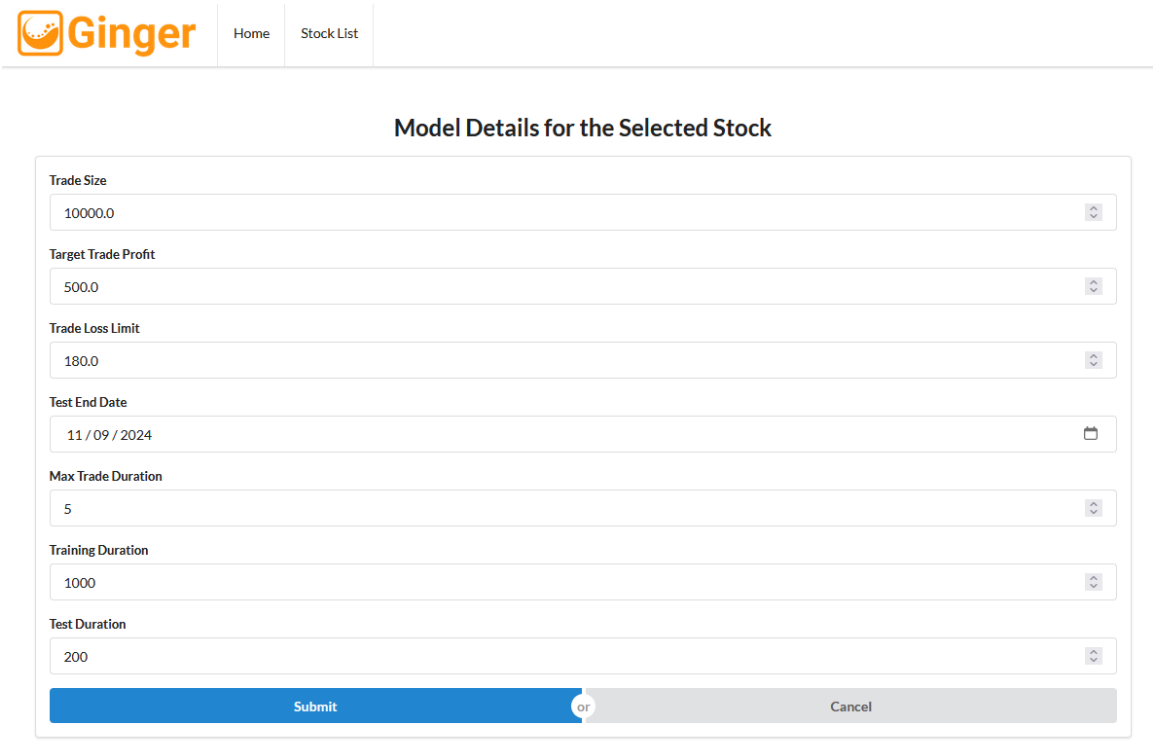
The Ginger Workflow

- Model creation and inputs
- Stock screening
- Stock data exploration & selection
- ML Model selection and training
- Back testing and results evaluation



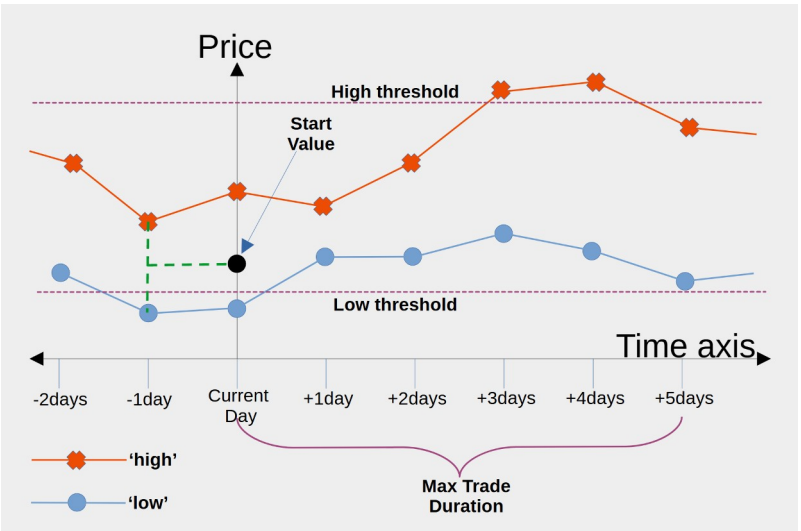
Model creation and inputs

- To start working with Ginger the user opens a new model and is presented with a form.
- The user enters a number of parameters into the model details form.
- These parameters define what a successful trading event looks like
- They include the maximum time span of that event.
- Also defined are the number of days that will be used by Ginger for training and testing a machine learning model.



Stock screening

- High & Low thresholds are calculated using the Start Value, Trade Size, Target Trade Profit and Trade Loss Limit.
- These parameters are used to screen the stocks and rank them by Occurrence opportunity.



Ginger | Home | Stock List

Screening Results

Show entries Search:

Stock Name	Total Records	Occurrence	Occurrence Interval	Average Duration	Four Sigma	Prepare Stock
FRES	1305	141	9.25531914893617	2.6595744680851063	3.0921019105010124	Gen Files
AAF	1305	135	9.666666666666666	2.7111111111111112	3.1239724678843603	Gen Files
KETL	1305	134	9.738805970149254	2.798507462686567	3.245535075132337	Gen Files
ENT	1305	125	10.44	2.816	3.267987444205359	Gen Files
EZJ	1305	121	10.785123966942148	2.520661157024793	2.785638098232624	Gen Files
AAL	1305	121	10.785123966942148	2.793388429752066	3.2626106263502472	Gen Files
VTY	1305	120	10.875	2.725	3.2368674554837344	Gen Files
FRAS	1305	117	11.153846153846153	2.8119658119658117	3.2771570493287663	Gen Files
JD	1305	115	11.347826086956522	2.8260869565217392	3.234067324114192	Gen Files
PSN	1305	114	11.447368421052632	2.8421052631578947	3.389894294158826	Gen Files
ANTO	1305	113	11.548672566371682	2.752212389380531	3.18617430477042	Gen Files
AHT	1305	113	11.548672566371682	3.0265486725663715	3.227567726019546	Gen Files
MKS	1305	113	11.548672566371682	2.7964601769911503	3.1694655716877227	Gen Files
NWG	1305	112	11.651785714285714	2.982142857142857	3.352326839390103	Gen Files
MRO	1305	111	11.756756756756756	2.891891891891892	3.3401731626903346	Gen Files

Showing 1 to 15 of 96 entries Previous 1 2 3 4 5 6 7 Next

Back to Home

Stock data exploration & selection

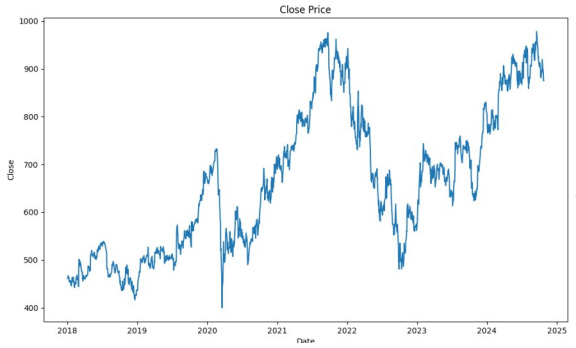
- The user can look at the curves for the preferred stocks in the database.
- Then choose which one to go for.

Stock Details for HWDN

ID	Stock Name	Start Date	End Date	Number of Records
3	HWDN	2018-01-02	2024-10-24	1722

Price Graphs

Close Price



Ginger Home Stock List

Screening Results

Show entries Search:

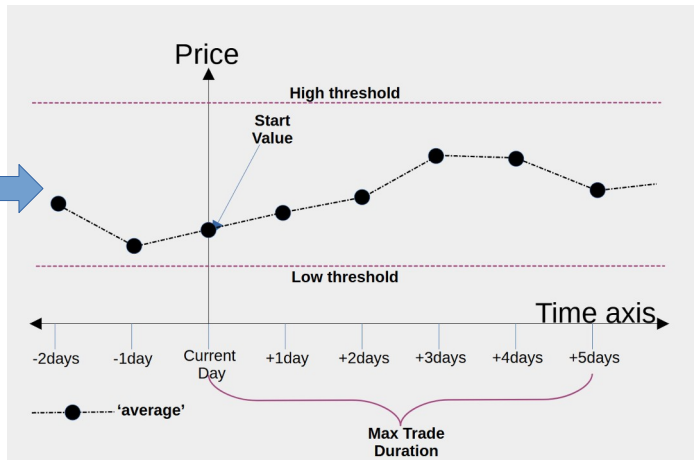
Stock Name	Total Records	Occurrence	Occurrence Interval	Average Duration	Four Sigma	Prepare Stock
FRES	1305	141	9.25531914893617	2.6595744680851063	3.0921019105010124	Prepare Stock
AAF	1305	135	9.666666666666666	2.7111111111111112	3.1239724678843603	Prepare Stock
KETL	1305	134	9.738805970149254	2.798507462686567	3.245535075132337	Prepare Stock
ENT	1305	125	10.44	2.816	3.267987444205359	Prepare Stock
EZJ	1305	121	10.785123966942148	2.520661157024793	2.785638098232624	Prepare Stock
AAL	1305	121	10.785123966942148	2.793388429752066	3.2626106263502472	Prepare Stock
VTY	1305	120	10.875	2.725	3.2368674554837344	Prepare Stock
FRAS	1305	117	11.153846153846153	2.8119658119658117	3.2771570493287663	Prepare Stock
JD	1305	115	11.347826086956522	2.8260869565217392	3.234067324114192	Prepare Stock
PSN	1305	114	11.447368421052632	2.8421052631578947	3.389894294158826	Prepare Stock
ANTO	1305	113	11.548672566371682	2.752212389380531	3.18617430477042	Prepare Stock
AHT	1305	113	11.548672566371682	3.0265486725663715	3.227567726019546	Prepare Stock
MKS	1305	113	11.548672566371682	2.7964601769911503	3.1694655716877227	Prepare Stock
NWG	1305	112	11.651785714285714	2.982142857142857	3.352326839390103	Prepare Stock
MRO	1305	111	11.756756756756756	2.891891891891892	3.3401731626903346	Prepare Stock

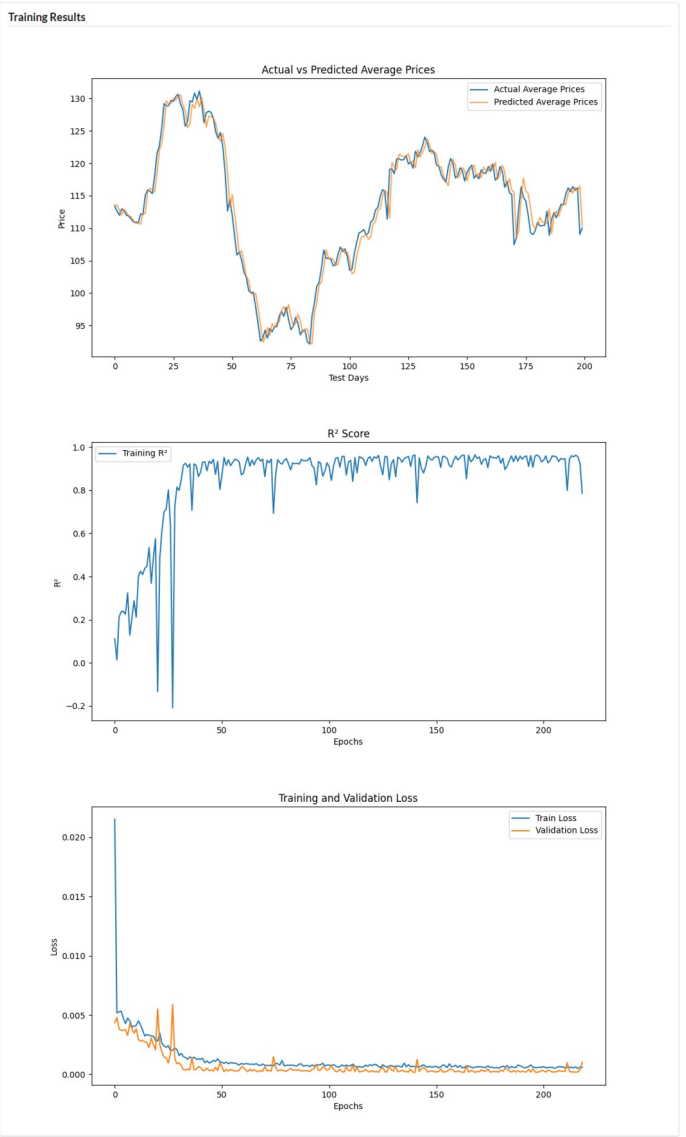
Showing 1 to 15 of 96 entries Previous 1 2 3 4 5 6 7 Next

Back to Home

ML Model selection and training

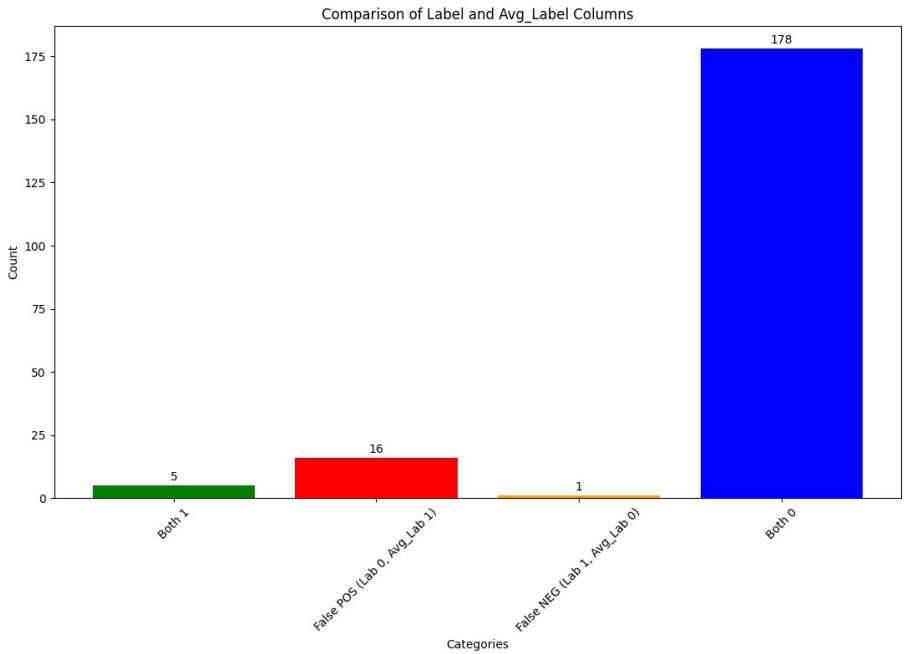
- Option of a ‘High Low Mean’ ML model or a ‘High Low’ ML model. Only former implemented at present.
- Once the user hits the Train Model button the selected machine learning model is trained. When it finishes the training results are presented.
- High Low Mean model uses a simplified way of classifying good trade events.



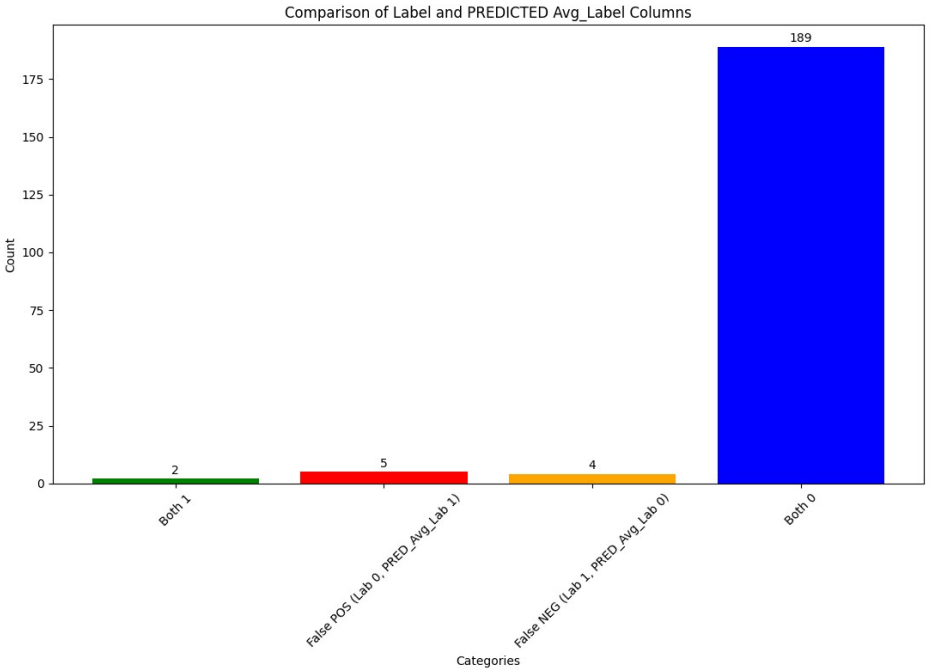


Back testing and results evaluation

- This graph shows how well the simplified classification works (not to great!)

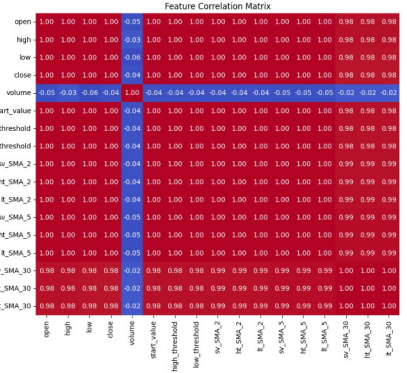


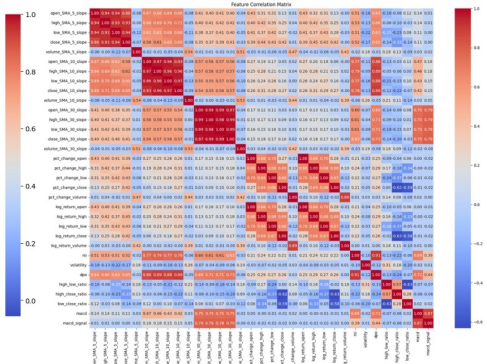
- This graph shows how well the ML model predicted using the simplified classification High-Low-Mean (averaged curves)
- Not too bad but can improve!



The Attention-GRU model

- This is the RNN model that was developed.
- Also tried LSTM, Dense Neural Network and a Random Forrest classifier.
- Hyper parameter tuning.
- Feature engineering.






```
X, y = self.create_sequences_multifeature(features_normalized, target_feature_index, window_size)

# Adjust X shape for the GRU model (samples, timesteps, features)
X = X.reshape((X.shape[0], X.shape[1], X.shape[2]))  # Shape (num_samples, window_size, num_features)

train_days = self.training_duration - window_size

X_train, X_test = X[:train_days], X[train_days:]
y_train, y_test = y[:train_days], y[train_days:]

# Input Layer
input_layer = Input(shape=(X.shape[1], X.shape[2]))

# GRU Layers with Dropout
gru_output = GRU(50, return_sequences=True)(input_layer)
gru_output = Dropout(0.2)(gru_output)

gru_output = GRU(50, return_sequences=True)(gru_output)
gru_output = Dropout(0.2)(gru_output)

gru_output = GRU(50, return_sequences=True)(gru_output)
gru_output = Dropout(0.2)(gru_output)

gru_output = GRU(50, return_sequences=True)(gru_output)
gru_output = Dropout(0.2)(gru_output)

# Attention Layer
attention_output = Attention()([gru_output, gru_output])

# Global Average Pooling Layer
pooled_output = GlobalAveragePooling1D()(attention_output)

# Fully Connected Output Layer
output_layer = Dense(1)(pooled_output)

# Define Model
attention_model = Model(inputs=input_layer, outputs=output_layer)

#attention_model.compile(optimizer='adam', loss='mean_squared_error', metrics=['accuracy'])

attention_model.compile(optimizer= tf.keras.optimizers.Adam(learning_rate=0.001),
                        loss='mean_squared_error', metrics=['accuracy']
                        )


# Instantiate the R² callback
r2_callback = R2ScoreCallback(validation_data=(X_test, y_test), target_scaler=target_scaler)

# Train the model with the callback
history = attention_model.fit(
        X_train, y_train,
        validation_data=(X_test, y_test),
        epochs=500,
        batch_size=16,
        callbacks=[
                tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=50, restore_best_weights=True),
                r2_callback  # Add the custom R² callback
        ]
)
```