BT2101 HW 3
TRADING PLATFORMS

Sung Zheng Jie
A0168188M

A0168188M

1. Algorithmic Trading

Algorithmic trading is the process of using computers programmed to follow algorithms and mathematical models based on parameters such as timing, price, quantity to automate a user's trading activities. It optimises the work and increases the profits of a trader, reduces the risk of manual errors and eliminates any impact of human emotions on trading activities. (Seth, 2018)

2. QuantConnect

QuantConnect is an open-source algorithmic trading platform where users can develop, backtest and deploy their trading strategies. It can be developed in Python, C# and F#. It offers Equity, Forex, Futures, Options and CFD data for users to conduct backtesting on their algorithms. Additionally, it supports live trading with Interactive Brokers, GDAX, FXCM and OANDA.

It also has a live trading environment which deploys the backtested algorithms to the market. It provides a research command-line interface for data exploration and manipulation. The "Alpha Streams" feature in QuantConnect connects traders with a marketplace of quantitative funds, competing to license their algorithm.

Additionally, the "QuantConnect Framework" provides a foundation for traders to develop well-defined algorithms with key quantitative finance concepts. These concepts include: (i) Portfolio Selection: Selecting assets to trade; (ii) Alpha Creation: Derive asset signals; (iii) Portfolio Construction: Determine position sizing; (iv) Execution Module: Intelligently placing trades to reach own position size; (v) Risk Management: Monitor ongoing risk.

3. Overview of Research Paper

This paper highlights the difficulties in a FX market, the challenges with modelling financial time series and the limitations of existing trading models. To overcome these challenges, it proposes a novel FX rate forecasting technique (Stacked Generalisation System) that allows for intelligently combining predictive models driven from individual currency exchange rate signals, and inferred cross-currency correlation signals. (Petropoulos, Chatzis, Siakoulis, & Vlachogiannakis, 2017)

3.1 Characteristics of a FX market

The peculiarities in the FX market that the trading model must consider include: (i) the very high liquidity of the FX market; (ii) its geographical dispersion; (iii) its continuous 24h per day/ 5 days per week operation; (iv) the variety and limited predictability of factors that affect exchange rates; (v) the low margins of relative profit compared to other markets of fixed income; (vi) the widespread use of leverage as a means of enhancing profit and loss margins.

3.2 Challenges with modelling Financial Time Series

The paper pointed out 3 key challenges with modelling financial time series. 1) Financial time series is noisy. They are not used as variables for modelling due to their fluctuations caused by factors that are difficult to record and quantify. 2) Financial time series is non-stationary. Because of the ever-changing external environment, the underlying distribution of the time series does not remain the same over time. This renders it difficult to predict trends over a fixed time period. 3) Financial time series is non-linear. Complex non-linear time-series models must be developed to capture the trends and accurately predict future price fluctuations.

3.3 Limitation of Existing Financial Time Series Forecasting

Existing trading models leverage on a single machine model. However, no single predictive model has managed to consistently dominate other alternatives in all experimental scenarios. As different models work best at different settings, developing methods for intelligently combining predictions stemming from alternative trained models might allow for increased predictive performance compared to any individual trained model.

3.4 Proposed Solution

To capture non-linear dependence patterns, the paper proposes to employ a variety of machine learning techniques that have shown to perform well in real-world modelling scenarios, where the absence of linearities is the norm. To solve the underlying non-stationary nature of the modelled time-series, continuous retraining on a moving window is executed to adjust to the changing dynamics of FX over time, influenced by monetary policy divergences, political instability and economic growth. To overcome the limitations posed by the noisy nature of the modelled data, the paper proposes a multilevel, multicomponent modelling framework for FX price forecasting instead of the traditional univariate modelling adopted by existing FX portfolio trading systems. Correlation signals across currency pairs are measured to deduce any structural relationships between the pairs, hence allowing for better price fluctuation prediction and trading decision. The use of such correlation information has never been leveraged before in FX portfolio trading and has the potential of providing diversification and money management benefits.

3.5 Choice of Currency Pairs

The 4 major currencies, EUR/USD, USD/JPY, GBP/USD and NDZ/USD, followed by the 3 commodity pairs, AUD/USD, USD/CAD and NZD/USD were chosen for FX portfolio trading because they have reasonable liquidity profile and high trading volume. 3 other currency pairs from the peripheral economics of the European Union were also chosen. (Total 10 pairs)

3.6 Data Collection and Pre-processing

To account for the non-normal and non-stationary nature of the currencies, exchange rate time-series is transformed into time-series of daily rates of returns.

The dataset is first split into train and test set. The test set is then further split into 3 periods corresponding to the different economic cycles of the US economy (Weak USD period, Recession, Strong USD period) to examine specific performance under different economic conditions.

3.7 Breakdown of Algorithm (Stacked Generalisation System)

The Stacked Generalisation System comprises 3 levels of functionality.

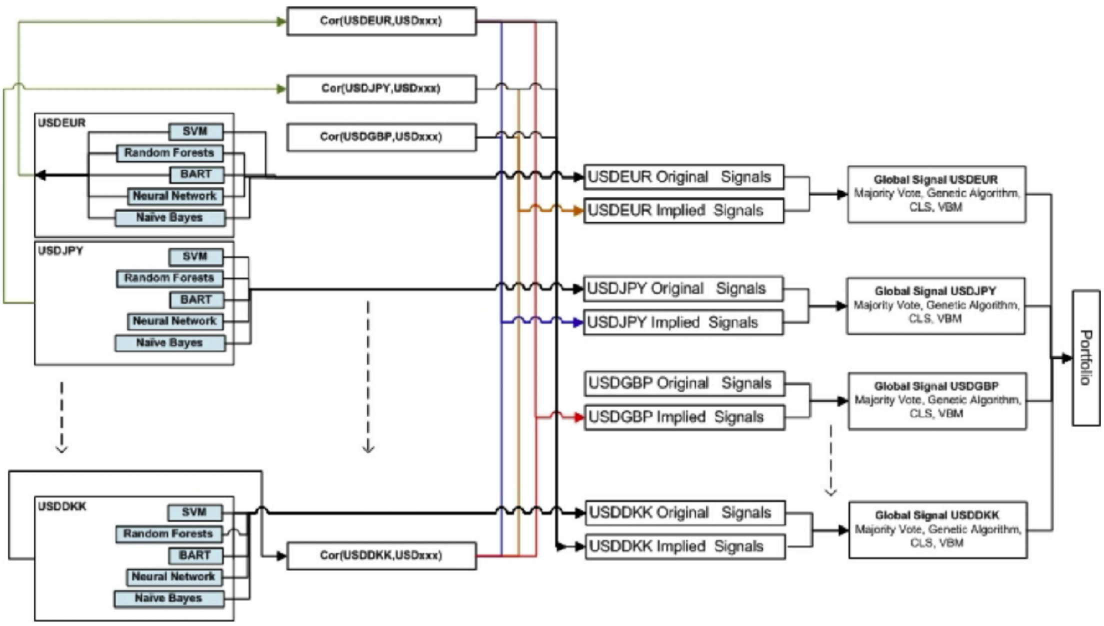


Fig 1: Flowchart of the Stacked Generalisation System

In the first level, the train data is fitted into the 5 different machine learning algorithms namely support vector machines (SVMs), random forests (RFs), Bayesian autoregressive trees (BART), dense-layer neural networks (NNs), and naïve bayes (NB) classifiers. The trained models are then used to perform prediction on each of the 10 currency pairs. At the end of the first level, every currency pair has 5 predictive signals that correspond to the 5 machine learning models employed. These signals are either -1 (“Sell”) or 1 (“Buy”). To account for non-stationary time series where conditions differ over time, models follow a moving-window retraining approach. This means at certain windows of time, the model is fitted with a different training data.

In the second level, correlations across currency pairs are computed to generate additional implied predictive signals which are derivatives of the respective original predictive signals of those currency pairs. Thus, for each currency pair, there are a total of 45 implied predictive signals and 5 original predictive signals. This is to reflect the co-movement between different currencies and to capture the relationship across currency pairs.

In the third level, predictive signals are aggregated into a single global predictive signal for each currency pair by one of the 3 methods – (i) majority voting; (ii) genetic algorithm optimisation; (iii) constrained least square forecast combination scheme. The final model utilises the genetic algorithm to calculate the weighted average as it achieves the highest performance among the methods proposed when the models are tested.

3.8 Conclusion

The stacked generalisation algorithm discussed, leverages the attractive properties of popular machine learning algorithms, as well as the correlation among currency pairs, which in turn, facilitates and informs trading process. It can be seen as a valuable tool to provide forecast for the contemporary evolution of multiple FX rates on a daily basis that has grown increasingly convoluted over time.

4. Evaluation of Machine Learning Strategies in Forex Trading (Question 2)

In this section, we will look into 3 machine learning strategies – support vector machines (SVMs), Artificial Neural Network (ANN) and Random Forests (RFs).

Support Vector Machine has been used extensively in traditional time-series modelling. The advantage of using SVM is that it has many versatile kernels (linear, polynomial, radial basis, spline) to model currency pairs that have different treads and patterns. Furthermore, SVMs are able to handle nonlinear classification and regression tasks. Moving-average trained SVMs also outperforms the ANN and ARIMA models (Kamruzzaman, Sarker & Ahmad, 2003). However, SVMs have high algorithmic complexity. It is also difficult to choose an appropriate kernel that models the data well.

Artificial Neural Network is another popular trading model. Unlike many other techniques of technical analysis which are based on price trend analysis, ANNs offer autocorrelation analysis and the possible errors in forecasting. Additionally, ANNs are able to extract complex nonlinear and interactive effects. It is shown to outperform the ARIMA model in stable market conditions (Vyklyuk, Vukovic & Jovanovic, 2013). However, ANN is known for its “black box” nature. It is difficult to interpret the mechanism and rationale behind the output generated by ANN. Moreover, ANN is computationally expensive and has a long development process due to its many dense layers. (Donges, 2018)

Random Forest is another widely used model for trading because of its capability in handling noisy data. This is especially important since time series data on currency prices do tend to have such outliers and noises present. As RF generally does not overfit due to the large number of trees and the law of large numbers, RF produces a limiting value of generalised error. Also, it is shown that RF performs significantly better than ANN in predicting currency prices in terms of Mean Average Percentage Error (MAPE). However, RF is known for being computationally expensive. (Basav, Sharad & Mousumi, 2015)

5. Mechanism of Recurrent Neural Network Model in Forex Trading (Question 3)

In this section, we will look at the use of Recurrent Neural Network (RNN) model in forex trading. RNNs have loops in their networks that allow information that are important for predictive analysis to persist. In particular, Long Short Term Memory (LSTM) is a special kind of RNN that is capable of learning long-term dependencies. (Olah, 2015)

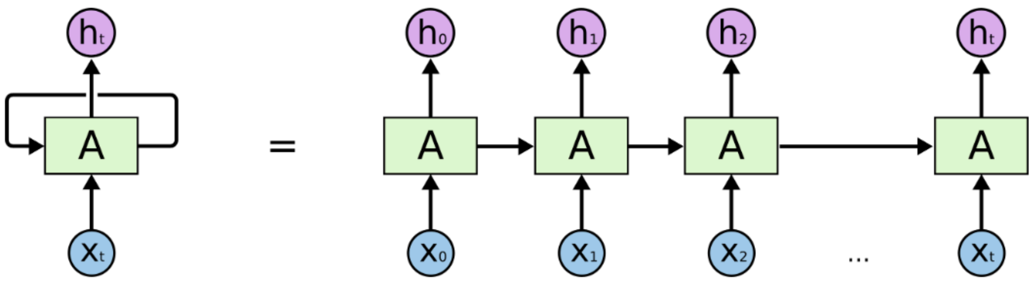


Fig 2: Unrolled Recurrent Neural Network

The RNN model is built by adding a LSTM layer. That layer has been shaped to fit the dimensions of the training set inputs. Dropout and activation functions are also included in the model. During data pre-processing, the data is first normalised before being split into test and train sets. LSTM model will use the previous data to predict the next day's opening or closing price. Supposed the number of previous days is 10. This means that the dataframe will consists of 10 consecutive days of data (called windows) in each entry. For example, the first entry will be 0 – 9th rows of the training set, the second entry will be will 1st – 10th rows, etc. This dataframe will be the input features. The labels will contain outputs that are derived by subtracting the price of the currency on the last day of each entry in the features from the next day price. The labels will then be binarized by setting 1 if output is positive and 0 otherwise. These features and labels will then be fitted as our modified training set to the LSTM model. (Sheehan, 2017)

The LSTM model is then back-tested on the test data. Prediction is done based on the direction of prices. The LSTM model will predict if the next day's opening or closing price will increase or decrease. If the price is expected to increase, the model will send a signal to buy the currency. If the price is expected to decrease, the model will send a signal to sell the currency.

6. Technical Differences between Fiat and Cryptocurrency (Question 4)

Fiat currency is 'legal tender' backed by a central government. The government controls its supply and uses it to pay taxes. On the other hand, cryptocurrency is decentralised through blockchains. Its supply is controlled by an algorithm rather than any central authority. In this section, we will look into the 3 technical differences when trading these currencies.

1. Trading Time: Cryptocurrency trading is available 24/7 but not Forex trading. Events that occur outside of the Forex's trading sessions have no instant impact on the Forex market but have impact on the Cryptocurrency market. As such, traders have to be mindful of the different trading sessions in these 2 markets. Generally, in the Forex market, it is more risky to hold any open positions between trading sessions, closing them and opening the next day because the model may not be updated to the latest changes that occur during the break.

2. Volatility: Cryptocurrencies are more volatile than Forex because the Cryptocurrency market is unregulated. The trend observe in a Cryptocurrency market may be constantly changing. While short-term profits can be easily attained without leverage, traders might find it useful to employ a rolling window that captures relevant trend in order to ensure sustainable profits in the long run. Forex trading, however, is less volatile due to the large volume of trading activities in the Forex market. Any single trader has very little impact on his own. To be profitable in trading Forex currency pairs, traders need to have a much higher leverage.

3. Liquidity: Because of the large trading volume in the Forex market, there is much more liquidity in the Forex market. On the other hand, cryptocurrency is more turbulent. There is higher propensity for the trade price to deviate from the execution price in the cryptocurrency market. Hence, models for cryptocurrency have to be sensitive to price changes. One way to do so is to ensure our brokerage have extremely fast servers that enables lower latency in trading.

7. Breakdown of QuantConnect Cryptocurrency Model

The cryptocurrency model is designed based on the code described in question 3 by setting the code into the cryptocurrency market instead of a forex market.

7.1 Data Preprocessing

The window_len is set to 10. This means that the previous 10 days of the data is used to predict the direction of price for the 11th day closing price. The features are a list of the change in price in every consecutive 2 days within the 10 days period. The label is the change in price from the last day of the 10 days period to the next day. The label is then binarised to 1 (increase in price) or 0 (decrease in price). To fit the data into the LSTM model, I reshaped the labels and features from 2-dimension to 3-dimension. Refer to the fig 3 for the code implementation in data preprocessing.

```
def preprocessing(self):

    # Create our input feature dataset and corresponding labels
    all_data = np.append(self.hist_data.open.values.flatten().astype(np.float32), self.current_price)
    features   = []
    labels     = []
    for i in range(self.window_len+1, len(all_data)-1):

        # input is change in priice
        features.append(np.diff(all_data[i-self.window_len-1:i].copy()))

        # label is change in price from last day in input to the next day
        dp = 100.*(all_data[i+1]-all_data[i])/all_data[i]

        # binarise the labels
        if dp > 0.0:
            dp = 1
        else:
            dp = 0
        labels.append(dp)

    self.features = np.array(features)
    self.labels   = np.array(labels)
    self.labels = to_categorical(self.labels)

    # Standardise features
    scaler = MinMaxScaler()
    self.features = scaler.fit_transform(self.features)

    # Reshape labels
    self.labels = self.labels.reshape((self.labels.shape[0], 1, self.labels.shape[1]))

    # Reshape features
    self.features = self.features.reshape((self.features.shape[0], self.features.shape[1], 1))
```

Fig 3: Code Breakdown for Pre-Processing

7.2 Model Building

The LSTM model is built by adding LSTM and Dense layers. To obtain the optimal results, the number of neurons, dropout rate, activation type and the overall structure of the model are all determined by trial and error. The eventual model is seen in fig 4. A Reshape layer is added between the Dense and LSTM layers to ensure consistency in the dimension of the input and output.

```python
    def keras_setup(self):

        self.session = K.get_session()
        self.graph = tf.get_default_graph()

        model = Sequential()

        model.add(LSTM(20, input_shape=(10,1), return_sequences=True))
        model.add(Dropout(0.25))
        model.add(Dense(units=1))
        model.add(Reshape((1, 10)))
        model.add(LSTM(2,return_sequences=True))

        model.add(Activation("linear"))

        model.compile(loss="mae", optimizer="adam")

        self.y_pred = model
```

Fig 4: Code Breakdown for LSTM model building

7.3 Rebalancing

The rebalancing function is called on a daily basis using the `self.Schedule.On()` function as seen in fig 5.

On every datapoint after the warmup period, the LSTM model is refitted with a new training dataset that represents the last 10 days of data (rolling window interval). The training set is modified each day by removing the first datapoint (earliest day) in the training set, and appending it with the latest datapoint (current day). This is to ensure the model captures the most recent trend for prediction of the every 11th day's closing price.

```python
    def Rebalance(self):

        # store current price for model to use at end of historical data
        self.model.current_price = float(self.Securities[self.symbol].Price)

        # Accrew history over time vs making huge, slow history calls each step. # Updates the training set on rolling window basis
        if not self.do_once:
            new_hist      = self.History([self.symbol,], 1, Resolution.Minute).astype(np.float32)
            self.model.hist_data = self.model.hist_data.append(new_hist).iloc[1:] #append and pop stack
        else:
            self.do_once  = False

        # Prepare our data now that it has been updated
        self.model.preprocessing()

        # Fit the model
        self.model.train()

        # Using the latest input feature set, lets get the predicted assets expected to make the desired profit by the next open
        signal = self.model.predict()

        # In case of repeated forecast, lets skip rebalance and reduce fees/orders
        if signal != self.target:
            # track our current target to allow for above filter
            self.target = signal
            # rebalance
            self.SetHoldings(self.symbol, self.target, liquidateExistingHoldings = True)
```

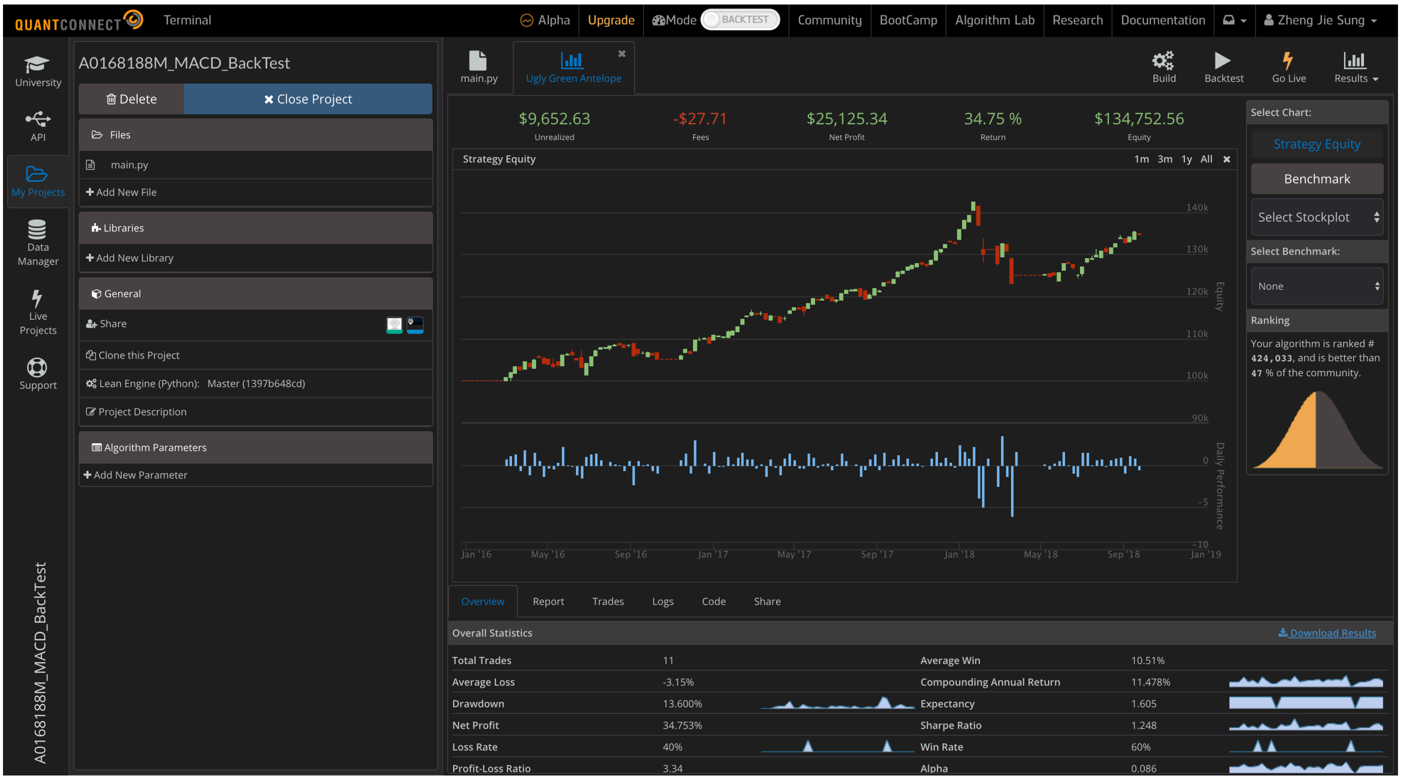
```python
        # trading on open-to-open daily price changes
        self.Schedule.On(self.DateRules.EveryDay(self.symbol), \
            self.TimeRules.AfterMarketOpen(self.symbol), \
            Action(self.Rebalance))

```

Fig 5: Code Breakdown for Rebalancing

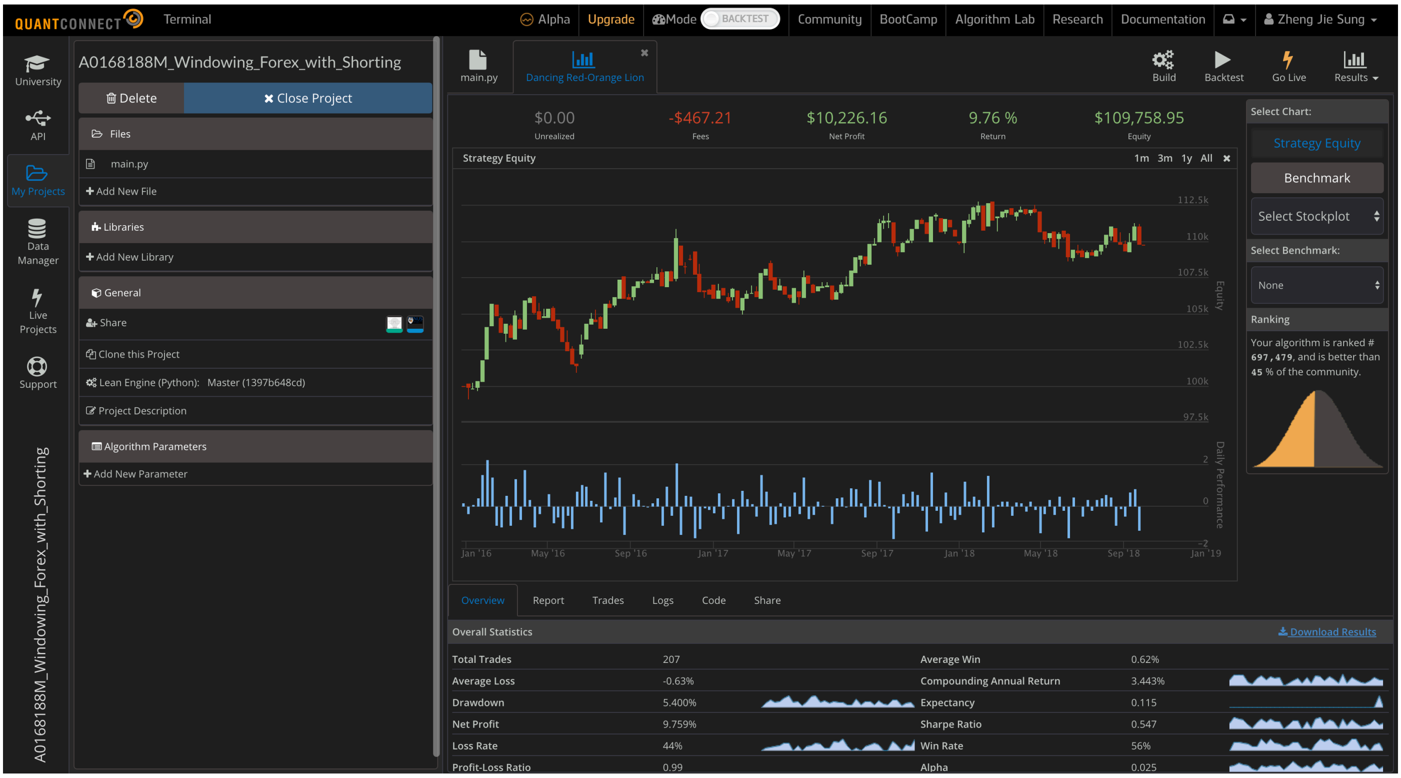
A0168188M

Appendix

Basic Back-test Model 1 (MACD)

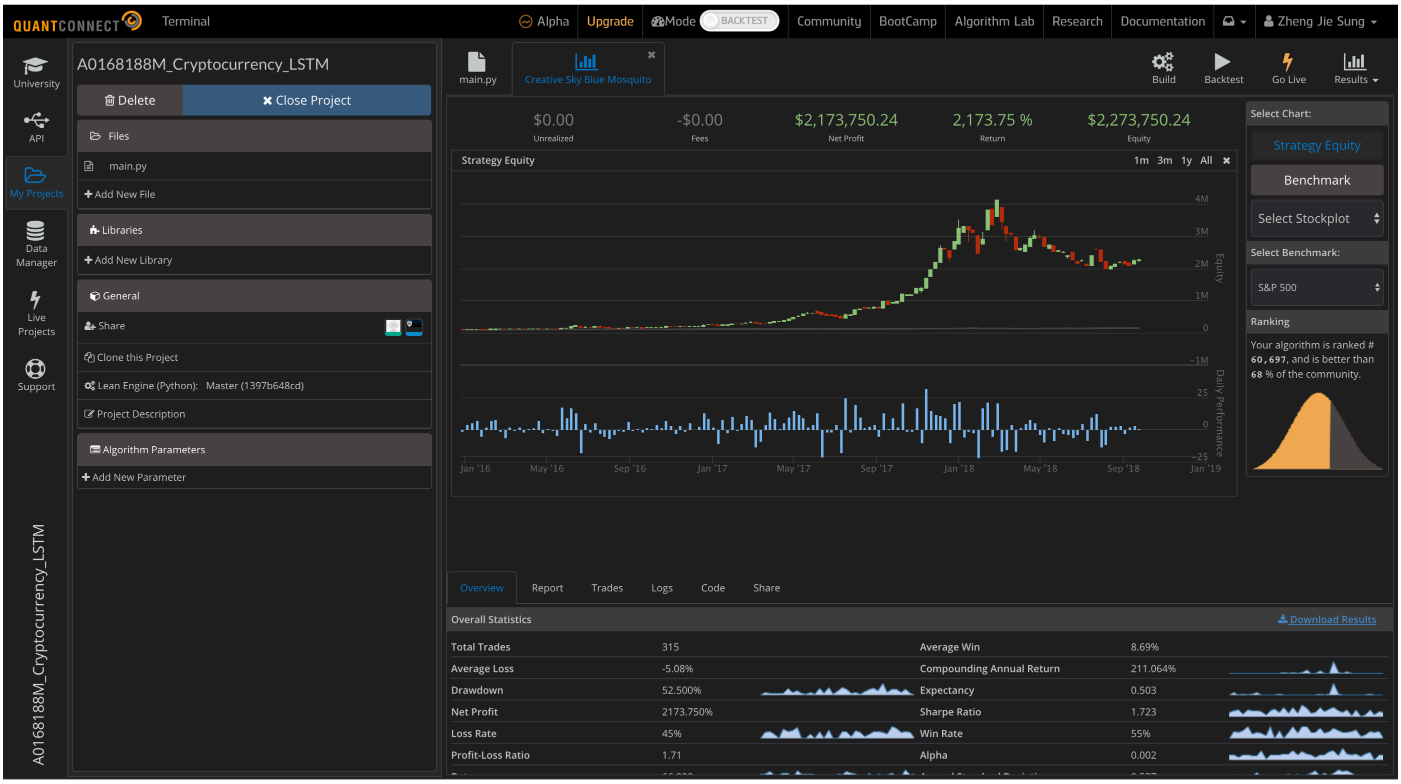


Forex Model 2 (Windowing and Shorting)



A0168188M

Cryptocurrency Model 3 (LSTM)



Main Research Paper

Petropoulos, A., Chatzis, S. P., Siakoulis, V., & Vlachogiannakis, N. (2017). A stacked generalization system for automated FOREX portfolio trading. *Expert Systems with Applications, 90*, 290-302.

Secondary Research Papers

Basav, R., Sharad, N., & Mousumi, B. (2015). Applicability of Machine Learning Tools in Foreign Exchange Market. *International Journal of Digital Content Technology and its Applications (JDCTA).*

Kamruzzaman, J., Sarker., & Ahmad, I. (2003). SVM based models for predicting foreign currency exchange rates. *Third IEEE International Conference on Data Mining.*

Vyklyuk, Y., Vukovic, D., & Jovanovic, A (2013). Forex prediction with neural networks: usd/eur currency pair. *Actual Problems of Economics, 10,* 251-261.

References

Seth, S. (2018). Basics of algorithmic trading: Concepts and examples. Retrieved from https://www.investopedia.com/articles/active-trading/101014/basics-algorithmic-trading-concepts-and-examples.asp

QuantConnect: Frequently Asked Questions. Retrieved from https://www.quantconnect.com/faq

Sheehan (2017). Predicting Cryptocurrency Prices With Deep Learning. Retrieved from https://dashee87.github.io/deep learning/python/predicting-cryptocurrency-prices-with-deep-learning/

Olah (2015). Understanding LSTM Networks. Retrieved from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Angelo, V. (2018). Crypto Trading vs Stock Trading: Three Main Differences. Retrieved from https://hackernoon.com/crypto-trading-vs-stock-trading-three-main-differences-851c1fdf082b

Donges, N. (2018). Pros and Cons of Neural Networks – Towards Data Science. Retrieved from https://towardsdatascience.com/hype-disadvantages-of-neural-networks-6af04904ba5b