A Work Project, presented as part of the requirements for the Award of a Master's Degree in Finance from Faculdade de Economia da Universidade Nova de Lisboa (Nova School of Business and Economics)

NOVA SCHOOL OF BUSINESS & ECONOMICS

Automated AI Trading System

Diogo Miguel Fontes Raposo Baltazar

Student number 29134

A Project carried out on the International Master's in Finance degree, with the supervision of:

Professor Leid Zejnilovic

16-12-2021

Abstract

This Work Project is an empirical investigation and a prototype development of an automated AI trading system. The framework system is a fully automated pipeline of data processing, data analytics, and signals interpretation that ends with an action to buy, hold, or sell an asset. The data analytics segment represents a deployment of state-of-the-art AI models developed to predict future cryptocurrency prices while accounting for risk and order management. The framework proposed predicted the next 15-minute close price of Bitcoin achieving an RMSE value of 167 during the period of 11^{th} - 15^{th} December. After accounting for fees and commissions, the system would have yielded a return of (1.11%) with a Sharpe Ratio of **(**0.04) against a B&H strategy with (3.92%) and (0.09) Sharpe.

Keywords:

Ensembled Systems; Cryptocurrencies; Machine Learning; Algorithmic Trading.

This work used infrastructure and resources funded by Fundação para a Ciência e a Tecnologia (UID/ECO/00124/2013, UID/ECO/00124/2019 and Social Sciences DataLab, Project 22209), POR Lisboa (LISBOA-01-0145-FEDER-007722 and Social Sciences DataLab, Project 22209) and POR Norte (Social Sciences DataLab, Project 22209).

Table of Content

1. Introduction

Automated trading systems can go from simple scripts with a few lines of code to fairly complex digital platforms that provide algorithm-driven decisions and/or recommendations with little to no human intervention/supervision. Thus, the motivation for this project arises from the lack of research and open-source platforms and the possibility of generating excess returns without any human intervention. The system presented, which has its codebase available on GitHub, proposes a full automated pipeline, from data acquisition and pre-processing to order management and portfolio risk. By leveraging big amounts of data stored in a SQL database, powerful machine learning techniques and NLP features in a sequential and seamlessly assembled process, it is able to continuously look for any indication of alpha or superior asset performance. In the same web page, a read.me file can be found, helping the user to run the system on their end. There is also a config. file where it is possible to replace the respective API keys to gain access to different exchanges (Binance, Coinbase, Kraken, FTX) and change other parameters. Additionally, a simple GUI (Guided User Interface) is being designed to aid the user on the system's performance and will be released in a future date. The same will happen in terms of configuring trade alerts via Telegram.

Regarding the investment universe, the commonly observable high levels of volatility in cryptocurrencies markets, which are not found in any other asset class, represent one of the many inefficiencies found in the space. As such, and due to the fact that the underlying technology, blockchain in most cases, is still in early stages of development, the system will focus on trading this asset class. A diagram can be found on the next page giving an overview of the framework implemented and a detailed explanation on section 3.

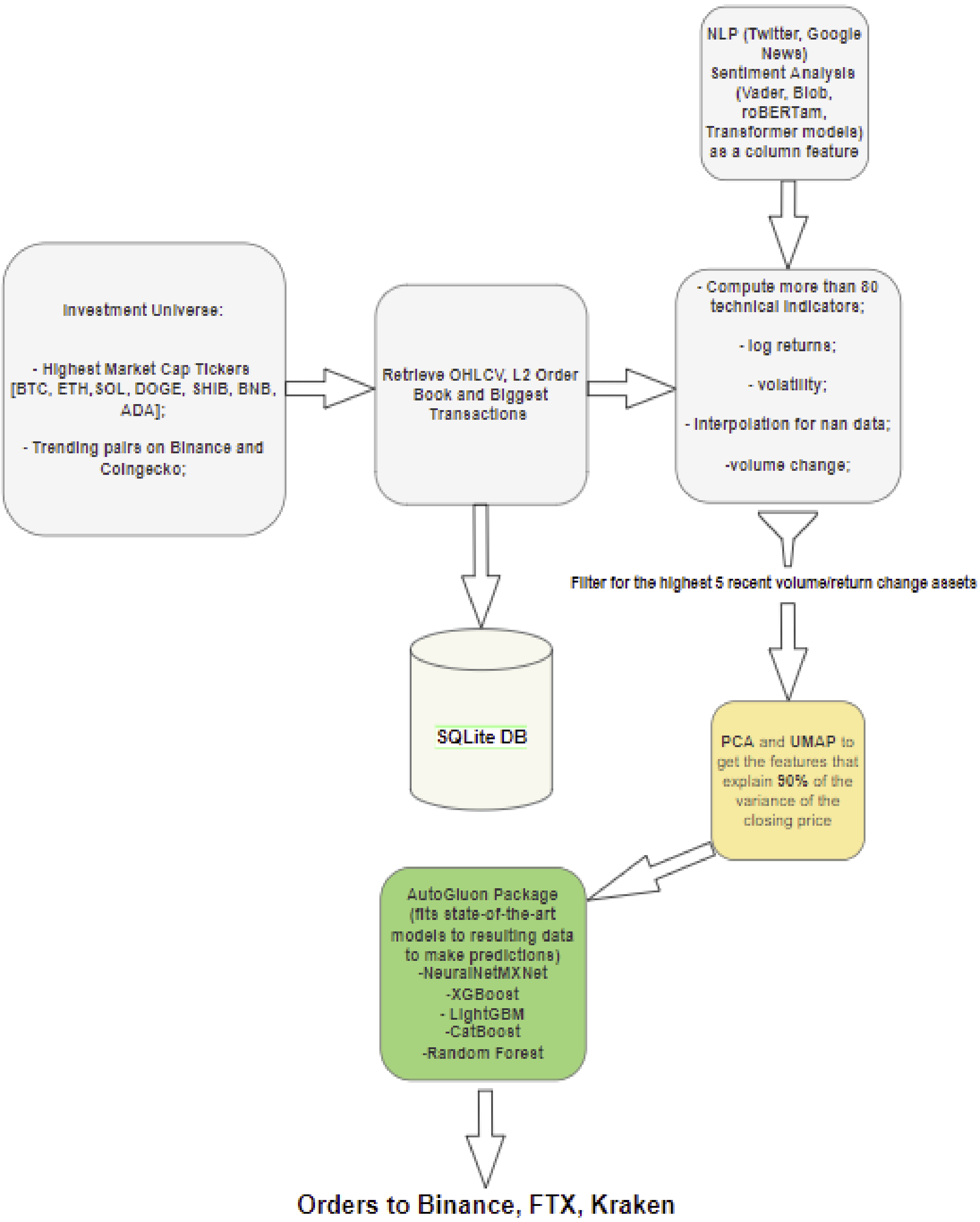


Figure 1) Current workflow

2. Literature Review

Inspiration to build an automated trading system has come from different fields ranging from fundamental analysis, econometric modelling to machine-learning. However, from inspiration to execution there is a long path in between. This section will explain the main obstacles faced when building an automated trading system. The first one and analogously one of the most important, is the existence of powerful and dedicated hardware infrastructure able to run many computationally expensive processes, often concurrently. A good GPU and large amounts of RAM memory will allow to meet most of the requirements needed for parallel computing and model processing. (Zhuo Zou, 2018)

The dependability and performance of such a system is also related to the efficiency of big data analysis and modellings (Szu-Hao Huang, 2020). To tackle this, a SQL database is an almost mandatory component for any system. The speed efficiency and performance have no comparison with other traditional databases. In this particular case, SQLite was used due to its stability, reliability and high efficiency as Chunyue Bi (2009) mentions in his paper. Acquiring the data to populate this database is another mandatory step which is not an easy task by any means. Your API connection may be terminated, and because the entire process is based on the processing of incoming data, things might quickly go wrong. In order to handle this, appropriate redistribution of your API requests across several exchanges and proper error and exception handling code is crucial.

Having this established, profitable strategies are now necessary, which commonly can include trend-following, mean-reversion, pairs-trading (statistical arbitrage) or machine learning models. These, can include neural networks, such as LSTM (Long-Short Term

Memory) or CNN (Convolutional Neural Network). When using models like this, the biggest issue is the unavoidable overfitting to the noisy data that is OHLCV (Open-High-Low-Close-Volume). Common RMSE values for models like this tend to be between 190 and 410. (S. M. Raju, 2020 and Yousef Owda, 2021) for the 30 min interval time period. Since the current strategy applied also falls in the space of auto machine learning, these figures can also serve as a model benchmark.

As for another quantitative solutions, OLPS (Li et al., 2016) was one of the first approaches to portfolio selection making use of machine learning algorithm as benchmarks. Quantopian, that has been terminated while the design of this system was taking place, also offered free available strategies and the respective code and reasoning behind. Another very interesting yet exponentially more complicated project named QLib, built by a team of developers at Microsoft (Qlib, 2020). It also employs the latest state-of-the-art AI algorithms. Although, this system is already extremely advanced populated by some of the latest state-of-the-art algorithms and data management engines, it lacks understandability and versatility. An open-source ensembled strategy has also been designed by a group of students and collaborators (AI4Finance Foundation, 2020). This strategy makes use of Deep Reinforcement Learning Agents that will keep learning from the actions that takes, buying selling or holding according to the portfolio turnover. This system aims to be a much simpler solution to these projects, yet equally powerful and with particular focus for cryptocurrencies.

Also, the possibility of being easily capable of adding a new strategy to the system is a huge plus for users that are experienced with programming or just prefer running their personalized strategies. For that, users just have to create a class with the structure present in figure 2 and add it to the “Strategies” folder.

```python
import sys
sys.path.insert(0, r'F:\Ensembled-AI-Trading-System')

from Utilities.config import *
from Utilities.util_funcs  import *
from Utilities.Database.Database_funcs import *

class NewStrat(object):

    def __init__(self):

        pass

    def Strategy(self, data):

        data = 'retrieved from SQLite DB'

        forecast = 'your code here'

        return forecast
```

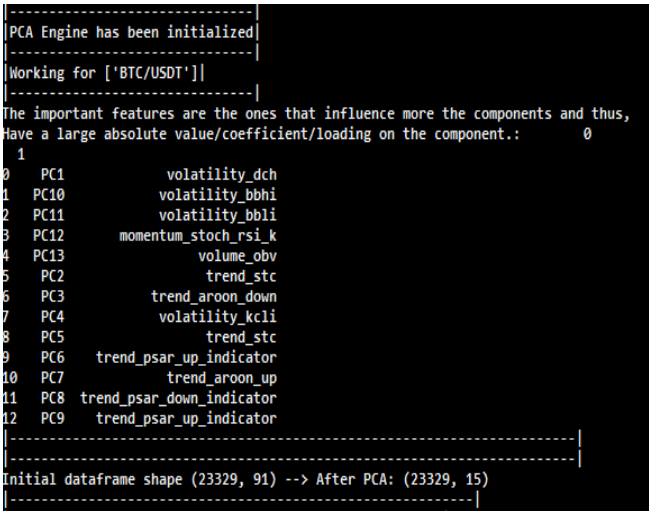
Figure 2) Example strategy design

As for the controversial incumbent cryptocurrencies market, pump-and-dump schemes and high volatility events have been more frequent than never, exhibiting a growth in frequency for the past two years. This happened mainly due to the entrance of new market participants, in some cases being large financial institutions. Another consequence was the increase in number of Twitter accounts, Telegram and Discord groups that either due to their large number of followers or participants have become able to influence the price of a coin with limited supply. As such, there are patterns associated with these schemes that might also provide significant alpha (Jiahua Xu, 2018). To track these patterns, Natural Language Processing is one of the best tools to use. It is also one of the areas where the most interesting developments in Artificial Intelligence have been taking place. Sentiment classification applied to financial related data dealt has proved to be a crucial element that any trading system should have (Michael Jermann, 2017). By using the advanced BERT model, one of the four implemented in the NLP module (Figure 4), was able to improve the prediction of Bitcoin price compared to only using bitcoin's previous day closing price (Ashrit Deebadi, 2020). As a summary, this work project reports on the development of a prototype of an automated trading system that has the following components: 1. Data acquisition, 2. Data processing, 3. Data Analytics, NLP component and Price predictor powered by ML models, 4. Trading strategy maker & risk manager.

3. Context, Data & Methods

Data acquisition and pre-processing

The system was designed in such a way that would be easy for the user to understand the various working parts during the trading period by looking to the command line. It begins by making API calls to the different cryptocurrency exchanges in order to scan for assets that have either recently experienced high periods of volatility or volume and return changes. Once this investment universe list is configured, it continues by retrieving OHLCV & L2 Order Book data. These requests make use of the *ccxt* library, which is considered to be one of the best in providing quick and free access to market data on several venues. The granularity of this data is configured within the system for one minute tick bars but can also be changed. According to this same granularity, some features are drawn from this data, such as Order Flow Imbalance (OFI), technical indicators and common metrics (volatility, volume change, log return, etc.). More than 80 technical indicators are calculated and added to the initial dataframe by using the ta-lib package. Then, as the dataset can quickly grow, consequentially increasing the overall computation time of the system, it is of extreme



```
|------------------------------|
|PCA Engine has been initialized|
|------------------------------|
|Working for ['BTC/USDT']|
|------------------------------|
The important features are the ones that influence more the components and thus,
Have a large absolute value/coefficient/loading on the component.:        0
  1
0    PC1            volatility_dch
1   PC10           volatility_bbhi
2   PC11           volatility_bbli
3   PC12      momentum_stoch_rsi_k
4   PC13                volume_obv
5    PC2                 trend_stc
6    PC3           trend_aroon_down
7    PC4           volatility_kcli
8    PC5                 trend_stc
9    PC6    trend_psar_up_indicator
10   PC7            trend_aroon_up
11   PC8  trend_psar_down_indicator
12   PC9    trend_psar_up_indicator
|----------------------------------------------------------------------|
|----------------------------------------------------------------------|
Initial dataframe shape (23329, 91) --> After PCA: (23329, 15)
|---------------------------------------------------------|
```

Figure 3) PCA engine

importance to infer the statistical importance of these features. By using dimensionality reduction techniques, such as Principal Component Analysis (PCA) and UMAP (Figure 3) the system is able to capture the few components that explain at least 90% of the closing price variance. These, that suffered an orthogonal linear transformation (PCA) and a nonlinear dimension reduction (UMAP), should exhibit no correlation with each other and retain most of the content of the original values. Furthermore, it is also configured to retrieve the largest transactions (refer to Figure 12 of appendix**)** in the crypto space since this has been proven to be linked with Bitcoin price movements due to the restricted token availability and the lack of a more homogeneous distribution of this existing supply. All this data is stored in the SQLite database mentioned earlier so that can be later queried to begin model training and execution. In order to create the database, there is a dedicated script with different queries for each table. Once these are created, the user can look at the "Database_funcs.py" file where different functions with multiple uses are defined. Some of the connection paths might have to be changed in the beginning of the scripts.

The Natural Language Processing module, incorporates sentiment analysis from Twitter, Reddit, and Google News. To achieve this, three packages (*tweepy*, *praw* and *google_news_api*) and four models were used. By specifying a list of relevant twitter usernames or cryptocurrencies related keywords, the system will continuously look for new information available, which can be tweets, posts or news. Following that, the text must be cleaned and processed in order for the models to make more educated conclusions. Regarding these, two are considered to be simpler, *Blob* and *Vader* while the remaining two, *roBERTa* and a *Transformers* neural network are considered to be the current state-of-the-art. As such, they have a higher weight calculating the final sentiment.

Final Sentiment = (0.125 * Blob + 0.125 * Blob + 0.35 * BERT + 0.4. * Transformers) / 1

	Date	User	Ticker	Tweet	Likes	Retweets	Blob sentiment	Vader sentiment	roBERTa sentiment	Transformers sentiment	Total Sentiment
0	2021-12-16 17:44:10	Seeking Alpha	[$ACN]	$ - ( ) Sweet on Q1 2022 - Earnings Call Trans...	0	0	0.350000	0.4588	0.841423	-0.972632	0.006545
1	2021-12-16 17:42:17	Seeking Alpha	[$VICR]	$ - ' s Big Q3 Miss Had Best Not Be Repeated I...	0	0	0.500000	0.5574	0.561793	-0.998768	-0.070705
2	2021-12-16 17:40:30	CtheLightTrading	[$TGT]	$ little gap - fill here	3	0	-0.187500	0.0000	0.836684	-0.996326	-0.129129
3	2021-12-16 17:39:43	CtheLightTrading	[$TGT]	$ support retest in progress today	5	0	0.000000	0.6705	0.934897	0.693884	0.688580
4	2021-12-16 17:39:15	Nathan Michaud	[$F]	$ F slip	2	0	0.000000	0.0000	0.491674	-0.997161	-0.226778
5	2021-12-16 17:35:55	Nathan Michaud	[$FBRX, $ADGI]	$ steady guess they ' re going back for any w ...	2	0	0.083333	-0.3400	0.755910	-0.992987	-0.164709
6	2021-12-16 17:34:10	Seeking Alpha	[$ADPT]	$ - Our First Assessment On Adaptive . finance	0	0	0.250000	0.0000	0.902697	0.735536	0.641408
7	2021-12-16 17:28:44	Seeking Alpha	[$PRGO]	$ - : In On A Long Term Bottom . trading finance	0	0	-0.050000	0.0000	0.692116	-0.999082	-0.163642
8	2021-12-16 17:25:37	CtheLightTrading	[$AAPL]	: * APPLE BUILDING OUT NEW OFFICE TO BRING WIR...	0	26	0.136364	0.0000	0.886019	0.930074	0.699182
9	2021-12-16 17:24:33	Brad Mullins	[$ADGI, $CABA, $APTO, $NISN]	$ $ CABA $ $ all gap down getting action today	5	1	-0.027778	0.0000	0.898424	-0.995937	-0.087398

Figure 4) Twitter Dataframe

These sentiments are then associated with the respective cryptocurrency dataset (through the ticker and timeframe) as a column feature.

Modelling and Ensembling

This idea of ensembling different strategies arises from the fact that different market regimes ask for different approaches by each market participant. This last segment of section 3, will introduce the current strategy applied and future possibilities. After having obtained the features that mostly contribute to the closing price of the asset, the data follows to be processed by *TabularPredictor*, an object within the Autogluon package. This package was developed by Amazon developer's and allows fitting state-of-the-art machine learning models such as LightGBM, XGBoost, RandomForest and Weighted_Ensemble_L2 **(**Figure 14 of appendix**)**. Since model fitting can easily become a very lengthy task, two hypotheses are provided:

- Do not limit the training time and improve prediction's accuracy with increased computational cost, which is advisable only for high-end machines.
- Retrain the models for x minutes every time the system runs at the cost of predictive accuracy.

Once the training is completed and after assessing the model with the best performance and metrics, a prediction for the next x minutes and the estimated return are calculated. If this return is greater than a certain amount enough to cover fees and commissions, the module will print a buy signal.

Another range of strategies that can be applied in the future involve mean-reverting indicators which state when an asset's price will tend to converge to its average price. Similarly, trend following indicators represent the likelihood for an asset to continue its current price movement. These, can be custom indicators or one of the common features that remain after the dimensionality reduction techniques are used (for instance, ichimoku trend). Also, by making use of an established automatic machine learning framework, a suite of statistical arbitrage or pairs trading strategies can also be configured. The main reasoning here, would be to continuously compare asset prices across venues and infer any profit opportunities (through the machine learning models) that may appear. One thing that all these strategies should have in common is a stochastic parameter optimization. Ideally this would be implemented through genetic algorithms, which tend to be computationally expensive yet very powerful. If that's not the case, it could also be done through a Bayesian approach.

Regarding the ensemble module, due to the stochastic nature of the markets, by ensembling the different strategies, the system will be able to reduce the generalization error and increase the signal-noise ration and overall reliability. Its functioning will be structured as follows, once all models have finished training, the system will assess which model's actions have been yielding the highest profit and attribute a weight to each for the overall ensemble. It will write the position value or profit obtained for a given model after a certain action (buy, sell or hold) has been taken. For example, if Autogluon (the present strategy) predicts a return of 0,4% and it's having a good momentum, it will have a bigger overall contribution to the final signal than the Facebook Neural Prophet model (which only forecasts a return of 0,2%). Thus, this will be a weighted average. For values between (-0.5; 0.5), rounding will be applied and interpreted as a signal to hold position. The same reasoning is applied for sell signal with -1 floor and buy signal with 1 cap.

Order and Risk Management

Handling open and closed orders is pivotal to keep track of the trading process and to avoid incurring in any unnecessary costs. Regarding order types, there are three main ones (Market, Normal Limit, Limit with Stop Loss) that have been configured. Market orders are executed according to the best available price. These are optimal when our goal is speed and trading immediately. As for the Limit type, there is a restriction on the maximum price to be paid or minimum to be received. If we include a Stop Loss, besides paying higher fees we are also introducing a safety buffer. This type of position was designed to limit an investor's loss by liquidating a position immediately once the asset's price reaches a certain threshold. As such, it tends to be the safer option when trading algorithmically.

For position sizing, Kelly Criterion was applied. The formula calculates the proportion of available capital to trade in order to maximize returns. According to existent literature, trading full Kelly can become very volatile. As such, I found it optimal to trade only 40% of the given output.

$$K\ \% = \frac{(bp - q)}{b} \quad (2)$$

K % --> Percentage of portfolio capital

p --> Probability of winning

b --> Decimal odds that are always equal to one

b --> Probability of losing (1 – p)

Figure 5) Kelly Criterion formula

Regarding risk control, certain rules were imposed such as the use of Stop-Loss Orders and the portfolio exposure to a certain pair can never be greater than 60% spread across exchanges.

Additionally, the system will trade with reduced size on low-volume days or when it exhibits a diminished degree of confidence.

Overview of the framework rules:

(Currently only signals from Autogluon are being considered)

1. Entry rules:
    - Go long/short if the sum of the weighted signals is equal to 1/ -1 and estimated return is enough to cover fees.
2. Exit rules:
    - Exit long/short positions once the signal reverses or a certain percentage of profit is obtained.
3. Instrument rule:
    - Trade only assets available on Binance and with USDT pairs (332).
4. Position sizing rule:
    - A position is sized according to Kelly Criterion or the target risk times your trading capital divided by the instrument risk (volatility).

```
|--------------------------------------------------------------------------------|
BTC/USDT 15 min prediction: 46414.54296875 with an estimated return of 0.037 %
|--------------------------------------------------------------------------------|
AutoGluon signal: 0
This script took %s 3:00:10.101058 seconds to run
-------------------------------------------------------
Finished running at 2021-12-13 21:07:52.790181
-------------------------------------------------------
```

Figure 6) Example prediction output

4. Results

The models were trained using more than thirty thousand rows of data and stored in a local folder for later usage. By loading existing predictors and only allowing for full training of the models once per week when the market exhibits low levels of volatility, the system is able to make faster predictions and ease on the computational calculations. The best model was Ensemble_NET with a RMSE of 167.34 and an overall return of (1.11%) for a period where the bitcoin price depreciated (3.92%) with a sample size of 98 observations. This showed that the strategy, even not fully optimized, to trade slightly better when compared to a buy-and-hold strategy than what would be expected of an algorithmic trading in such a highly volatile space. Despite the fact that the approach's ROI was negative, it was 2.81 basis points greater than a buy-and-hold strategy.

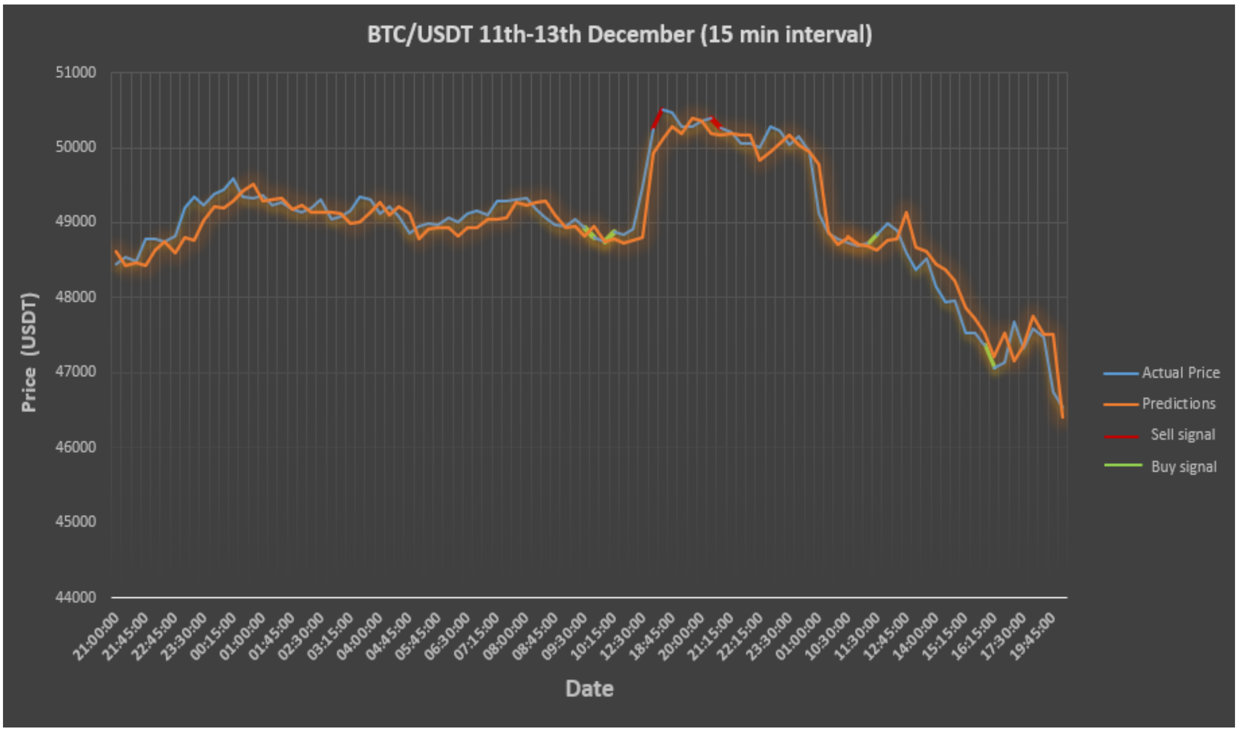


Figure 7) Bitcoin predictions & signals 15 min interval 11^{th}-13^{th} December

	Return (%)	*Sharpe Ratio*	*Max Drawdown*
Buy and Hold	(3.92%)	(0.09)	**(7.86%)**
Autogluon	**(1.11%)**	**(0.04)**	(7.90%)

Figure 8) Metrics Table

5. Discussion

These statistics may not appear to be impressive since they are negative, but they do serve as proof-of-concept that this project can be a viable trading framework if further refined. The current established pipeline, already enhances the baseline prediction from models like LSTM, ARIMA or CNN's. Moreover, once the NLP features from Twitter and Google News start being taken into account as well as Order Flow Imbalances and other strategies like the ones mentioned before, the metrics will improve substantially more. Furthermore, longer evaluation period and alternative time data intervals could have been explored for more accurate results.

Results different to those presented in this paper may be obtained when the models are run using the code from GitHub. This stems from the fact that the derived results presented here were obtained during the period of 11th to 15th December, after which some changes were made concerning the data used to train the models among other minor procedural changes.

6. Conclusion

This paper sets out to clear up much of the existing literature by proposing a new benchmark bitcoin price forecasting process that future studies can build on.

There are more plans for this project in the future involving doing part of the data collection and model training in the cloud; telegram configuration for buy and sell orders and finish the GUI. By establishing a rigorous machine learning pipeline to ensure the system's higher accuracy, I pretended to alleviate the existing issues faced in the data science business and algorithmic trading. The provisional results show that there is potential for further development as there are still many important parts that need to be configured and taken into account. I appreciate that the approach presented in this paper is unlikely to be without error and in the spirit of openness and transparency, the code and data is available through the GitHub repository (https://github.com/DiogoMBaltazar/Ensembled-AI-Trading-System) to allow full replication by others. I more than welcome any extensions, corrections, and improvements to the models presented.

7. References

Yang, Xiao. 2020. “Qlib: An AI-oriented Quantitative Investment Platform.”

Yang, Hongyang. 2020. “Deep Reinforcement Learning for Automated Stock Trading: An Ensemble Strategy”

M. A. H. Dempster and V. Leemans. Cambridge 2004. “An Automated FX Trading System Using Adaptive Reinforcement Learning”

Crone, Nathan, 2021. “Exploration of Algorithmic Trading Strategies for the Bitcoin market”

Pauna, Cristian. 2018. “Automated Trading Software. Design and Integration in Business Intelligence Systems”

Hu, Wei. 2020. “Practical Meta-Reinforcement Learning of Evolutionary Strategy with Quantum Neural Networks for Stock Trading”

Davda, Atish. 2018. “NLP and Sentiment Driven Automated Trading”

Jermann, Michael. Stanford. “Predicting Stock Movement through Executive Tweets”

Deebadi, Ashrit. 2020. “Understanding Impact of Twitter Feed on Bitcoin price and trading patterns”

Zou, Zhuo. 2018. “Automated trading systems statistical and machine learning methods and hardware implementation”

8. Appendix

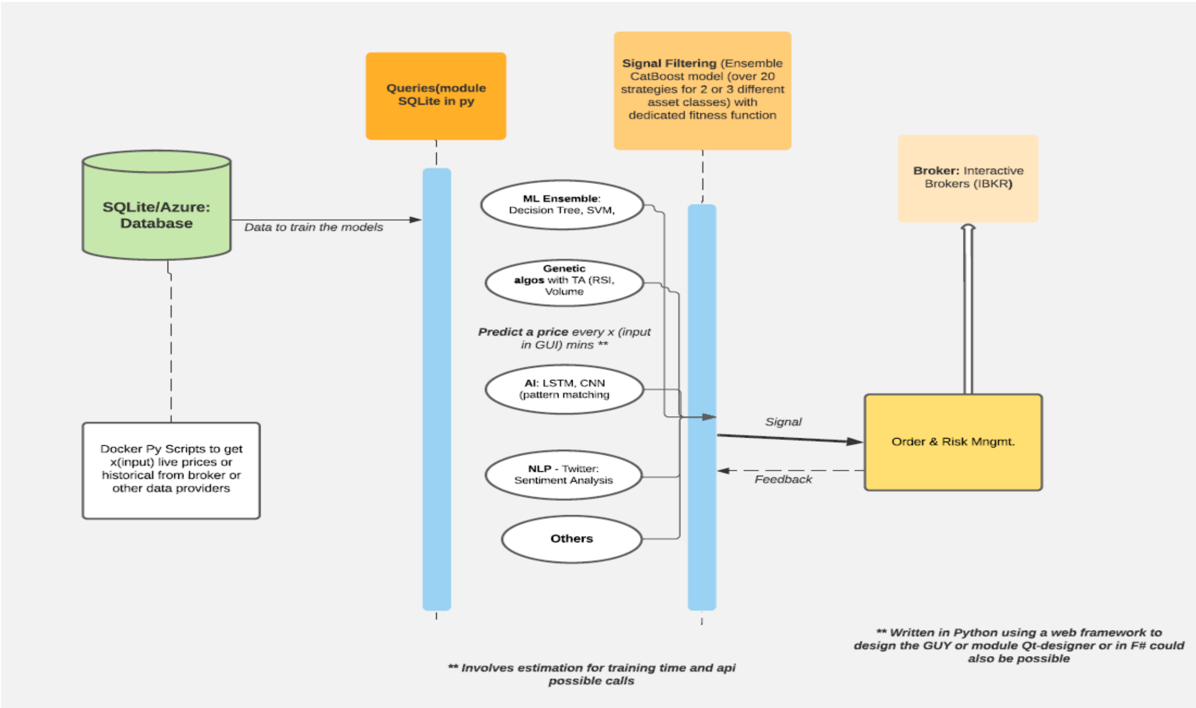


Figure 9) Initial framework proposed

Figure 10) Current state of the GUI

Table: Crypto_L2_Order_Book

	id	date	crypto_id	Bid_price	Bid_size	Ask_price	Ask_size	Spread	OFI
1	1	2021-09-12 15:11:10.744087	SOL/USDT	178.28	29.18	178.29	0.11	0.00999999999999091	0.931238248724147
2	2	2021-09-12 15:11:11.032033	SOL/USDT	178.28	4.98	178.29	0.11	0.00999999999999091	0.811385853939045
3	3	2021-09-12 15:11:11.538975	SOL/USDT	178.28	12.98	178.29	0.11	0.00999999999999091	0.846657316503039
4	4	2021-09-12 15:11:12.046949	SOL/USDT	178.28	30.19	178.31	17.83	0.0300000000000011	-0.0291050583657587
5	5	2021-09-12 15:11:12.550611	SOL/USDT	178.3	8.0	178.31	17.78	0.00999999999999091	0.317578057616008

Figure 11) OHLCV table

Table: Crypto_Prices

	id	crypto_id	date	open	high	low	close	volume
1	1	ETH/BTC	2021-06-18 04:26:00.000000	0.061827	0.061841	0.061816	0.061822	17.667
2	2	ETH/BTC	2021-06-18 04:27:00.000000	0.061827	0.061842	0.061816	0.061832	17.204
3	3	ETH/BTC	2021-06-18 04:28:00.000000	0.061827	0.061832	0.061801	0.061819	14.79
4	4	ETH/BTC	2021-06-18 04:29:00.000000	0.061818	0.061822	0.061803	0.061818	37.492
5	5	ETH/BTC	2021-06-18 04:30:00.000000	0.061819	0.061823	0.061802	0.06181	34.767

Figure 12) L2 order book table

Table: Crypto_Whale_Transactions

	id	date	Owner	Owner_Type	Amount	crypto_id	Amount_Dollars	Recipient	Recipient_Type
1	1	1970-01-19 21:06:35.527000	binance	exchange	3038	BNB	1267996.8		unknown
2	2	1970-01-19 21:06:35.541000		unknown	1000000	USDT	1000000		unknown
3	3	1970-01-19 21:06:35.577000	binance	exchange	2911066	USDT	2911066	ftx	exchange
4	4	1970-01-19 21:06:35.577000	binance	exchange	2130994	BUSD	2130994		unknown
5	5	1970-01-19 21:06:35.592000	gemini	exchange	500	ETH	1768982.1		unknown

Figure 13) Whale transactions table

```
AutoGluon Models Performance:                    model  score_test   score_val  _val      fit_time
0    KNeighborsDist_BAG_L1    -0.001150 -229.019629     9.905777    9.964999   9.905777
1           XGBoost_BAG_L1   -24.259467 -207.400436     6.132381    0.189782   6.132381
2     LightGBMLarge_BAG_L1   -31.131893 -226.039841    43.743001    4.139955  43.743001
3          CatBoost_BAG_L1   -34.070566 -167.934857     0.265623    0.058975   0.265623
4          LightGBM_BAG_L1   -36.130627 -224.630127    22.518044    2.195034  22.518044
5    NeuralNetMXNet_BAG_L2   -52.605848 -144.167984   145.356712   33.054787   7.336230
6           XGBoost_BAG_L2   -56.020373 -142.269131   141.492530   32.440632   3.472048
7        LightGBMXT_BAG_L1   -56.529665 -331.903613    34.541687    3.383074  34.541687
8     LightGBMLarge_BAG_L2   -61.951080 -139.932960   145.617920   33.058029   7.597438
9     ExtraTreesMSE_BAG_L1   -62.039668 -166.141678     0.652758    0.642999   0.652758
10  RandomForestMSE_BAG_L1   -69.997468 -186.610011     0.685737    0.670973   0.685737
11         LightGBM_BAG_L2   -87.998898 -139.526632   145.710353   33.147774   7.689871
12     WeightedEnsemble_L3   -92.500436 -126.766761   176.949007   45.225094   0.000000
13   KNeighborsDist_BAG_L2  -100.960526 -148.170219   148.225208   42.505150  10.204726
14   KNeighborsUnif_BAG_L1  -202.921193 -260.845803     9.951841   10.042480   9.951841
15  NeuralNetFastAI_BAG_L2  -219.421888 -151.947751   140.648695   32.753532   2.628213
16   NeuralNetMXNet_BAG_L1  -698.347982 -698.468620     7.152747    0.702784   7.152747
17  NeuralNetFastAI_BAG_L1  -842.457744 -619.169668     2.470887    0.355908   2.470887
```

```
AutoGluon Models Feature Importance:
volatility_dch                    7815.969518
volume_obv                        1282.634309
5_min_log_returns                  181.437521
15 min volume                      127.187628
volatility_kcli                     20.371367
trend_stc                           19.237724
momentum_stoch_rsi_k                14.735880
trend_aroon_up                      11.283620
trend_aroon_down                    10.530725
log_returns                         10.221876
volatility_bbli                      5.188467
Volume                               4.212574
TARGET                               1.107281
volatility_bbhi                      0.764389
trend_psar_up_indicator              0.257356
trend_psar_down_indicator            0.039607
```

Figure 14) Autogluon models scores for 3 hours of training and according feature importance