Received: 24 November 2021 | Revised: 6 March 2023 | Accepted: 29 May 2023

DOI: 10.1002/isaf.1538

RESEARCH ARTICLE

WILEY

Evaluating interpretable machine learning predictions for cryptocurrencies

Ahmad El Majzoub [1] | Fethi A. Rabhi [1] | Walayat Hussain [2]

[1]School of Computer Science and Engineering, University of New South Wales (UNSW), Kensington, Australia

[2]Peter Faber Business School, Australian Catholic University, North Sydney, Australia

Correspondence
Walayat Hussain, Peter Faber Business School, Australian Catholic University, North Sydney, Australia.
Email: walayat.hussain@acu.edu.au

Summary

This study explores various machine learning and deep learning applications on financial data modelling, analysis and prediction processes. The main focus is to test the prediction accuracy of cryptocurrency hourly returns and to explore, analyse and showcase the various interpretability features of the ML models. The study considers the six most dominant cryptocurrencies in the market: Bitcoin, Ethereum, Binance Coin, Cardano, Ripple and Litecoin. The experimental settings explore the formation of the corresponding datasets from technical, fundamental and statistical analysis. The paper compares various existing and enhanced algorithms and explains their results, features and limitations. The algorithms include decision trees, random forests and ensemble methods, SVM, neural networks, single and multiple features N-BEATS, ARIMA and Google AutoML. From experimental results, we see that predicting cryptocurrency returns is possible. However, prediction algorithms may not generalise for different assets and markets over long periods. There is no clear winner that satisfies all requirements, and the main choice of algorithm will be tied to the user needs and provided resources.

KEYWORDS
artificial intelligence, cryptocurrency, deep learning, interpretability, machine learning, technical indicators, time series forecasting

1 | INTRODUCTION

The application of artificial intelligence (AI) has been taking various forms within different industries. In finance, numerous firms and banks have been gradually integrating a variety of AI applications into their workflows and processes. These may include automation, credit decisions (Dumitrescu et al., 2022), algorithmic and high-frequency trading, risk management (Hussain, Raza, et al., 2022), fraud detection and prevention (Khan et al., 2022), personalised banking (Cao, 2022) and many others. Despite the inevitable difficulties that companies will be faced with when transitioning into new systems, the potential of AI in transforming the financial sector could not be matched by traditional pipelines. These difficulties may include the large costs associated with R&D and implementation, the business's unrealistic expectations, the shortage of specialised engineers, interpretability and the lack of agility within huge corporations (Dixon et al., 2020). However, with the exponential increase of computational power and data abundance, the shift into automated *intelligent* structures powered by machine learning is a crucial step that could determine the survival of many existing corporations (Hussain, Gao, et al., 2022).

Machine learning (ML) algorithms are widely used by different investing firms to analyse the pattern of data and infer important

information from it. These algorithms enable the decision-maker to identify various nonlinear data patterns that other linear algorithms cannot detect. The performance of each ML algorithm varies depending on selecting suitable parameters and the nature of a dataset (Hussain & Sohaib, 2019). The speed and accuracy with which some ML algorithms can analyse massive amounts of historical data are unparalleled. Different algorithms perform altered results on different datasets. The accuracy of algorithms also varies on structured and unstructured data. There are various situations where the decision-maker needs to tailor and customise the dataset to accommodate the user's special needs and priorities. For example, Hussain et al. (2021) and Hussain, Merigó, et al. (2022) introduced an Induced Ordered Weighted Averaging (IOWA) operator in Artificial Neuro-Fuzzy Inference Systems (ANFIS) to prioritise a certain set of data over others to handle complex nonlinear predictions. The approach handles the complexity of prediction by assigning variable weights using the inducing variable of IOWA for nonlinear stock market predictions. Although such approaches can handle complex nonlinear predictions, there are still many gaps in making such ML algorithms interpretable for humans to better manage the analysis of data such as cryptocurrency market data. This is exacerbated by the fact that most traders and investors do not disclose information about their in-house designed algorithms and techniques to guarantee their advantage and dominance over a highly competitive and merciless market and to secure a technological edge over their competitors.

This paper aims to explore and examine the application of various machine learning methods on financial markets with a focus on interpretability. Following this main aim, our research question is as follows:

> Which machine learning algorithm, given adequate data, can have relative robust predictions of the cryptocurrency market directions, while providing interpretable results?

The goal is to provide a comprehensive comparative study on financial time series forecasting methods by selecting various approaches from different categories and testing them in a homogenous environment. The approach is not only focused on the overall accuracy achieved by the algorithm but also considers other important factors, including interpretability, user expertise, computational requirements and related costs. To achieve the objective, the paper first explores the variety and diversity of the available data that could help in predicting cryptocurrency market trends. The aim is to incorporate different types of indicators and their impact on prediction accuracy. The paper then analyses and compare the prediction accuracy of selected existing machine learning algorithms in a homogenous environment with clearly defined variables and testing measures. The analysis includes the interpretability of each prediction result that assists the decision-maker in adopting an optimal algorithm in a real-world problem.

The rest of the paper is organised as follows: Section 2 discusses related literature. Section 3 describes the proposed approach and different component of the approach. Section 4 presents analysis results and findings, and, finally, Section 5 concludes the paper with future work.

2 | LITERATURE REVIEW

The section presents related studies that highlight the use of AI techniques in predicting financial data. The section discusses the approaches and how they are related to predicting financial data. Even though the boundaries are often blurred as many approaches combine different algorithms and techniques, the division was beneficial to understand the evolution of such processes and the effectiveness of each type in treating the problem at hand.

Thakkar and Chaudhari (2021) analysed different neural network approaches for stock market data. The authors took nine best-performing algorithms and compared their results. The authors found that deep Q-network (DQN) performed better than other deep neural approaches for a dataset of 5-day stock trends. Henrique et al. (2019) reviewed and compared 57 of the most cited papers in the field. The authors classified the studies according to the corresponding markets, assets, predictive variables, predictions, main methods and performance measures. Even though it is almost impossible to compare performances when the actual studies target different variables, markets and error measures, the authors concluded that there is still high activity and interest in the subject. The study found that one of the most commonly used algorithms is support vector machines (SVM) and that there was a high concentration of studies on the North American markets. Fischer (2018) used a more generalised methodology to analyse the use of various technologies in financial markets. The author divided the relevant literature based on critic-only, actor-only, actor-critic, the number of citations and the number of citations per year. The approaches were then compared based on their intended usage, including high-frequency trading, optimising execution, enhancing existing trading strategies and others. The study found that the true potential of reinforcement learning lies in the combination of their predictive strength and portfolio construction.

Considering the highly complex nature of the problem at hand and the limitless possibilities and combinations of algorithms that could be explored, many approaches attempted to exploit the advantages of several techniques. Kumar and Thenmozhi (2014) investigated several hybrid methods, including ARIMA-SVM, ARIMA-ANN and ARIMA-Random Forest methods. The study found that ARIMA-SVM outperformed the other methods by achieving the best forecasting accuracy that would translate into better returns. Kim et al. (2004) proposed a hybrid knowledge integration approach using a fuzzy genetic algorithm to integrate knowledge from multiple sources to predict the Korean price index. The study found that the hybrid integration of knowledge approach performed better than other approaches. Rabhi et al. (2020) surveyed several machine-learning algorithms in electronic financial market trading. The study found a mismatch between existing academic literature, which tends

to concentrate on asset price prediction and certain areas in electronic trading, for example, smart order routing, that need more attention.

Time series forecasting could be implemented using various approaches and techniques. These include traditional statistical systems using conventional methods like AR, MA, ARIMA, Machine learning and deep learning algorithms, and unique hybrid approaches. The M4 competition could be the most influential time series forecasting competition that is done yearly by comparing various submissions from individuals, academics and institutions (Darin & Stellwagen, 2020). Makridakis et al. (2020) performed an M4 competition on 100,000-time series data and assessed 61 forecasting methods. The time series data span across various industries, with almost 25% of data focused on the financial sector. The study found that most approaches used a combination of statistical and ML methods, while other submissions were mostly pure statistical methods. Few approaches are built exclusively on machine learning algorithms. Smyl (2020) presented the winning submission of the M4 prediction competition. The author used a hybrid approach that combined exponential smoothing with advanced long short-term memory (LSTM) neural network. The study found that the approach performed better for monthly, yearly and quarterly datasets. Oreshkin et al. (2019) proposed a neural basis expansion analysis for the interpretable time series (N-BEATS) forecasting method. The study found that N-BEATS proved to be highly effective in time series forecasting and outperformed ES-RNN when it was run on the M4 datasets. Furthermore, one main focus of N-BEATS is to provide forecasting practitioners with the trend and seasonality decomposition. This is usually overlooked in competitions where the emphasis might be solely on the accuracy of the algorithm. Still, as mentioned previously, interpretability is a major requirement when forecasting financial markets.

Based on the requirements of the situation, the user might use different AI and/or statistical techniques. This could vary according to the computational requirements, required outcome and specialty of the user. Lara-Benítez et al. (2021) analysed seven deep learning algorithms in time series forecasting. The authors execute MLP, ERNN, LSTM, GRU, ESN, CNN and TCN algorithms on over 50,000 time-series data. The study found that LSTM achieved the best weighted absolute percentage error (WAPE). Convolutional neural network (CNN) had the better mean and standard deviation of WAPE while maintaining the best speed and accuracy balance. All other methods, except Multilayer perceptron (MLP), achieved comparable results when it was hyper-tuned accordingly.

Although the discussed approach have attempted to optimally predict financial time-series data, however, there are still many gaps and shortcoming as listed below:

1. The authors could not make conclusive findings unless the study used the same dataset and error measures. It is needed to create a unified pipeline that could test various algorithms in the same environment and highlight its feature and limitations.
2. Most of the discussed approaches used American equity markets and very few have focused on cryptocurrency prediction. Considering the distinct nature of the crypto market due to its highly volatile environment and the huge interests of various stakeholders, it is imperative to see the behaviour of AI in crypto trading.
3. Very limited literature has focused on the interpretability of the forecasting process and results, despite its major importance in institutional trading. Prediction accuracy might be the top priority in automated trading. However, an interpretable outcome that could be explained to non-technical managers should also be aimed for. This will facilitate and expedite the mass adoption process, which will only reflect positively on the evolution of the whole industry.

Considering the discussed gaps, this paper tries to bridge gaps between various approaches. The paper provides a comprehensive comparative analysis of several prediction approaches by defining different metrics involved in the process. Section 3 discusses the proposed approach.

3 | PROPOSED APPROACH

This section proposes a corresponding methodology, which will be broken down into multiple steps. Each will require an independent literature review to benefit the most from the current state-of-the-art methods. Since we will be developing our modules for this study, the pipeline will be divided into the following three main parts:

1. The dataset formation: The first step is gathering all related information for maximising forecasting accuracy and performance. These include technical and fundamental indicators. The different features will be explained in the dataset chapter, along with their corresponding extraction and formatting methods.
2. Applying the models: One of the faced issues in the reviewed literature was the limited scope of each paper. It is impossible to compare algorithms that are being tested on distinct datasets and performance measures. This chapter will examine and compare the commonly used algorithms, the state-of-the-art forecasting approaches and our hybrid method.
3. Analysing the results: Our evaluation method will mainly focus on backtesting all the individual approaches and comparing their predictions to try to understand the pros and cons of each. We will expand on the performance measures in the corresponding chapter. Furthermore, we will be examining the interpretability of each algorithm's outcome. Even though this feature might have been overlooked in most of the literature, it is a crucial element that might dictate algorithm adoption in the real world. The analysis result is presented in Section 4.

Figure 1 shows a graphical representation of our proposed pipeline. Each section will have its own chapter to investigate the current literature and explain the process behind the selected modules.

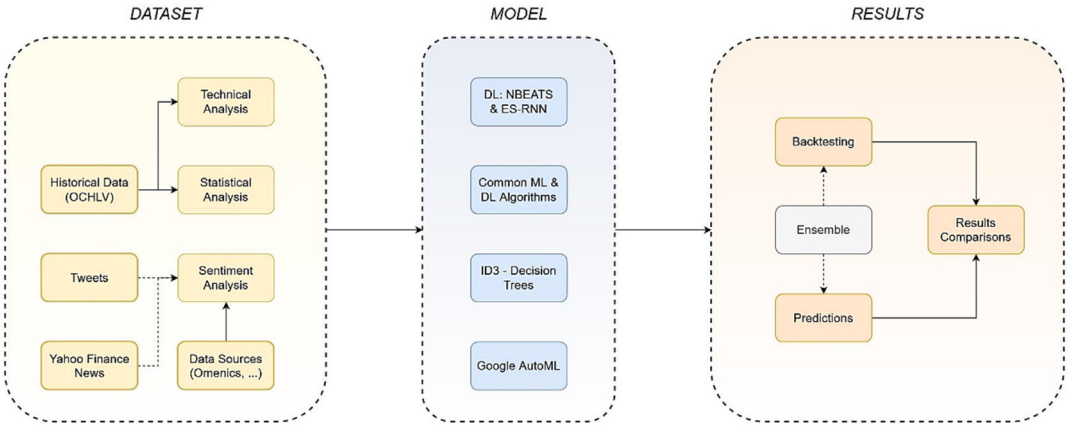

FIGURE 1 Three phases of the proposed approach.

3.1 | Dataset formation

Datasets play a major part in the success or failure of any machine learning algorithm. The cleanliness, relevance and statistical significance of the features will directly dictate the outcome (Cruz et al., 2022). Because we are dealing with a highly complex nonlinear forecasting task, we should try to benefit from all available related information to construct the datasets. This will only reflect in the objectivity of the comparative study, which is a necessary element in this situation.

Numerous events are dictating the evolution of any financial asset. As mentioned in the previous section, most of the reviewed approaches depended on a single type of feature. In this study, we aim to collect and combine various types of features to ensure the algorithms benefit from their true potential. The failure or success of any prediction job could be attributed to the dataset, the algorithm or the special combination of both, along with the related hyperparameters (Gogas & Papadimitriou, 2021). To avoid this and ensure that each algorithm has a fairground, we will invest in developing comprehensive datasets and conduct the appropriate tests and trials to ensure that the approach is hyper-tuned efficiently and effectively.

First, we start extracting the hourly data for the intended cryptocurrencies. These usually include the open, close, high, low and volume metrics (OCHLV). Even though cryptocurrencies do not technically have an open or close price, as the market is always open, these usually indicate the start and end price of the intervals (i.e., a granularity of the dataset). From these metrics, we calculated a number of technical indicators. Many retail and institutional traders depend on technical analysis as part of their prediction process. All cryptocurrency transactions are recorded on a public ledger. These records could be accessed, analysed and used in the prediction process of certain metrics, including the projected price of the asset. However, accessing such information will require significant time and computational resources. The paper also tried to access third-party data analytics tools and providers that could help in enriching the datasets. After investigating several sources, the cryptocurrency analytical data provider Omenics (https://omenics.com/) was selected. A graphical presentation of the Omenics features used in the final dataset is presented in Figure 2.

Finally, we combined the various features in six different datasets, one for each cryptocurrency. The final data frame dimensions are 25,560 rows representing 3 years of hourly data (August 2, 2018 to July 1, 2021). A section of the dataset is presented in Figure 3.

3.2 | Applying target models

The following are the competing algorithms used in this paper:

1. ES-RNN: The winner of the M4 competition
2. N-BEATS: An approach that outperformed ES-RNN and could provide the trend and seasonality decomposition
3. SVM: Was highlighted as one of the best machine learning algorithms in the related literature
4. Decision Trees: A very common machine learning approach that might be considered as one of the most interpretable algorithms.
5. Random Forests: An ensemble method built on decision trees that could achieve more generalisable results with the capabilities of extracting feature's importance.
6. LSTM (Long Short Term Memory): A deep learning algorithm that was able to outperform other DL approaches in time series forecasting.
7. CNN (Convolutional neural networks): A deep learning approach that achieved the best mean and standard deviation in the related literature and the best speed.
8. ARIMA: A conventional statistical approach that statisticians and data scientists commonly use in various forecasting tasks.
9. Google AutoML: A fully automated approach that could cater for non-technical users.


```
[
  {
    "timestamp": "2019-04-21T04:00:00+00:00",
    "ticker": "BTC",
    "price_usd": 4128.96,
    "market_cap_usd": 71209871860,
    "volume_usd": 9101654235,
    "overall_score": 7.25360,
    "fundamental": 8.71034,
    "technical": 7.75621,
    "news_sentiment": 6.10119,
    "news_volume": 1072,
    "twitter_sentiment": 6.75134,
    "twitter_volume": 5765,
    "reddit_sentiment": 5.76544,
    "reddit_volume": 8764,
    "buzz": 5.12011,
    "buzz_raw": 5.12011,
  }
]
```


FIGURE 2 Omenics extracted features.

df - DataFrame

Index	returns	log returns	log close	change %	price usd	market cap usd	volume usd	overall score	fundamental	technical	news sentiment	news volume	twitter sentiment	twitter volume	reddit sentiment	reddit volume	buzz
2018-08-02 03:00:00	-26.55…	-0.00347	8.94146	-0.34620	7695.39625	13213100362…	4749584641…	6.05259	9.98700	4.12100	6.35930	447.00000	4.46410	334.00000	3.02201	524.00000	4.24520
2018-08-02 04:00:00	-15.24…	-0.00200	8.93946	-0.19942	7672.43118	13223220230…	4596814364…	6.06634	9.98700	4.13800	6.36747	438.00000	4.56906	343.00000	2.97674	517.00000	4.25126
2018-08-02 05:00:00	29.870…	0.00391	8.94337	0.39163	7672.43118	13183758687…	4563010252…	6.09146	9.98600	4.16100	6.37512	438.00000	4.73587	351.00000	2.95157	509.00000	4.25634
2018-08-02 06:00:00	-11.91…	-0.00156	8.94181	-0.15554	7682.99811	13167064483…	4510145161…	6.11863	9.98600	4.18800	6.38795	401.00000	4.89394	364.00000	2.93124	505.00000	4.25973
2018-08-02 07:00:00	13.950…	0.00182	8.94364	0.18247	7669.13822	13201916158…	4410131320…	6.14484	9.98600	4.21800	6.39995	401.00000	5.02953	365.00000	2.92127	513.00000	4.26298
2018-08-02 08:00:00	-47.78…	-0.00626	8.93738	-0.62384	7688.36457	13179567414…	4339660962…	6.17242	9.98600	4.25000	6.41605	416.00000	5.16608	367.00000	2.90918	518.00000	4.27372
2018-08-02 09:00:00	2.81000	0.00037	8.93775	0.03692	7668.96046	13212608280…	4349499503…	6.19879	9.98600	4.28400	6.43111	416.00000	5.29168	368.00000	2.89646	510.00000	4.28542
2018-08-02 10:00:00	-40.86…	-0.00538	8.93237	-0.53664	7655.22659	13179261922…	4242711385…	6.22763	9.98600	4.31900	6.45682	404.00000	5.41837	359.00000	2.87902	508.00000	4.29381
2018-08-02 11:00:00	-13.12…	-0.00173	8.93063	-0.17324	7603.20539	13155660004…	4273891004…	6.25673	9.98600	4.35400	6.48088	404.00000	5.53916	363.00000	2.87293	501.00000	4.29511
2018-08-02 12:00:00	-50.05…	-0.00664	8.92399	-0.66203	7592.12669	13066260527…	4289310109…	6.28167	9.98500	4.38300	6.50272	402.00000	5.65421	376.00000	2.85771	502.00000	4.29683
2018-08-02 13:00:00	-9.680…	-0.00129	8.92270	-0.12889	7568.14454	13047221562…	4321405460…	6.30673	9.98500	4.40700	6.52316	402.00000	5.76959	378.00000	2.85396	504.00000	4.31025
2018-08-02 14:00:00	41.270…	0.00549	8.92819	0.55024	7538.54253	13006007759…	4346834006…	6.32691	9.98500	4.42300	6.54217	400.00000	5.86554	370.00000	2.84937	494.00000	4.32237
2018-08-02 15:00:00	-16.80…	-0.00223	8.92596	-0.22276	7576.00955	12955136116…	4398613555…	6.34673	9.98500	4.43300	6.55996	400.00000	5.96740	367.00000	2.85049	505.00000	4.33117
2018-08-02 16:00:00	42.230…	0.00560	8.93155	0.56121	7545.94100	13019523932…	4431489097…	6.36710	9.98500	4.44100	6.57601	392.00000	6.07757	360.00000	2.85518	516.00000	4.34121
2018-08-02 17:00:00	-60.98…	-0.00809	8.92346	-0.80587	7585.92753	12967850524…	4442238751…	6.38361	9.98500	4.44700	6.59104	392.00000	6.16819	365.00000	2.85462	515.00000	4.34946
2018-08-02 18:00:00	1.59000	0.00021	8.92368	0.02118	7555.92563	13036568175…	4485642984…	6.39331	9.98500	4.45000	6.59534	389.00000	6.23023	364.00000	2.85560	524.00000	4.35620
2018-08-02 19:00:00	29.800…	0.00396	8.92764	0.39693	7543.57641	12985009310…	4478771105…	6.40370	9.98500	4.45300	6.59937	389.00000	6.29012	354.00000	2.86477	550.00000	4.35909
2018-08-02 20:00:00	-3.700…	-0.00049	8.92715	-0.04909	7561.83770	12963786926…	4512365034…	6.40367	9.98400	4.45100	6.59860	391.00000	6.30720	351.00000	2.85498	554.00000	4.36096
2018-08-02 21:00:00	-4.630…	-0.00061	8.92653	-0.06146	7578.66187	12995169322…	4530337366…	6.40349	9.98400	4.44600	6.59788	391.00000	6.30303	355.00000	2.86915	547.00000	4.36242
2018-08-02 22:00:00	17.380…	0.00231	8.92884	0.23084	7555.08448	13024081999…	4462373479…	6.40215	9.98400	4.44100	6.59557	389.00000	6.27024	348.00000	2.90580	527.00000	4.36011
2018-08-02 23:00:00	-20.77…	-0.00276	8.92608	-0.27523	7575.64838	12983563784…	4388002635…	6.39818	9.98400	4.43500	6.59341	389.00000	6.23319	359.00000	2.92739	520.00000	4.35720
2018-08-03 00:00:00	-140.5…	-0.01885	8.90723	-1.86760	7567.15160	13018903252…	4253208560…	6.39629	9.98400	4.42900	6.59239	380.00000	6.21158	355.00000	2.94797	516.00000	4.35039
2018-08-03 01:00:00	-45.16…	-0.00613	8.90109	-0.61150	7419.26836	13004301371…	4214110858…	6.39339	9.98400	4.42400	6.59143	380.00000	6.16668	349.00000	2.98157	514.00000	4.34041
2018-08-03 02:00:00	0.00000	0.00000	8.90109	0.00000	7383.14517	12750161068…	4234157728…	6.39065	9.98400	4.41200	6.59756	375.00000	6.11181	375.00000	3.02623	512.00000	4.32770
2018-08-03 03:00:00	-14.09…	-0.00192	8.89917	-0.19196	7379.54356	12688082638…	4288373705…	6.38647	9.98400	4.39400	6.60330	375.00000	6.06193	400.00000	3.06719	513.00000	4.31106
2018-08-03 04:00:00	37.480…	0.00510	8.90428	0.51161	7369.77345	12681893195…	4282360904…	6.37892	9.98400	4.37200	6.61016	372.00000	5.99461	409.00000	3.10446	513.00000	4.29594
2018-08-03 05:00:00	-38.39…	-0.00523	8.89905	-0.52136	7342.31937	12665103064…	4269848952…	6.36389	9.98400	4.34500	6.61657	372.00000	5.89080	402.00000	3.12916	525.00000	4.28063
2018-08-03 06:00:00	33.000…	0.00450	8.90354	0.45051	7358.67582	12617922691…	4349575426…	6.34595	9.98400	4.31600	6.62742	393.00000	5.76798	387.00000	3.14477	534.00000	4.26716
2018-08-03 07:00:00	40.680…	0.00551	8.90906	0.55287	7383.61827	12646031570…	4435002240…	6.32472	9.98400	4.28300	6.63756	393.00000	5.64320	384.00000	3.14543	547.00000	4.25365
2018-08-03 08:00:00	-22.66…	-0.00307	8.90599	-0.30627	7378.45625	12693418129…	4485003843…	6.30408	9.98400	4.24900	6.65421	374.00000	5.51009	382.00000	3.14812	550.00000	4.23943
2018-08-03 09:00:00	-16.13…	-0.00219	8.90380	-0.21868	7409.66202	12684543939…	4544784051…	6.28519	9.98300	4.21200	6.66979	374.00000	5.39768	395.00000	3.15428	550.00000	4.22611
2018-08-03 10:00:00	-6.890…	-0.00094	8.90286	-0.09362	7381.21422	12738190792…	4636071745…	6.26431	9.98300	4.17500	6.67825	382.00000	5.28782	398.00000	3.15418	551.00000	4.21324

FIGURE 3 Sample dataset used for the study.

3.2.1 | Models hyperparameters and meta-data

Figure 4 shows the python libraries used to implement each of the above models and the hyper tuning methods and corresponding links. Some of the models were hyper-tuned using automated functions implemented within the same python libraries (GridSearchCV and auto_arima) or based on current similar implementations of the models (Keras LSTM and CNN). The rest were run on default parameters.

Model	Python Library	Parameters	Library Link
ES-RNN	ESRNN	default	https://pypi.org/project/ESRNN/
N-BEATS	NBEATS	default	https://pypi.org/project/NBEATS/
SVM	SKLEARN	hypertuned using sklearn.model_selection.GridSearchCV	https://pypi.org/project/scikit-learn/
Decistion Trees	SKLEARN	hypertuned using sklearn.model_selection.GridSearchCV	https://pypi.org/project/scikit-learn/
Random Forests	SKLEARN	hypertuned using sklearn.model_selection.GridSearchCV	https://pypi.org/project/scikit-learn/
LSTM	KERAS	based on the following implementation: https://colab.research.google.com/drive/18WiSw1K0BW3jOKO56vxn11Fo9lyOuRjh	https://pypi.org/project/keras/
CNN	KERAS	based on the following implementation: https://github.com/hoseinzadeehsan/CNNpred-Keras	https://pypi.org/project/keras/
ARIMA	PMDARIMA	hypertuned using pmdarima.arima.auto_arima	https://pypi.org/project/pmdarima/
Google AutoML	Cloud Implementation	default	https://cloud.google.com/automl/

FIGURE 4 Source for selected models.

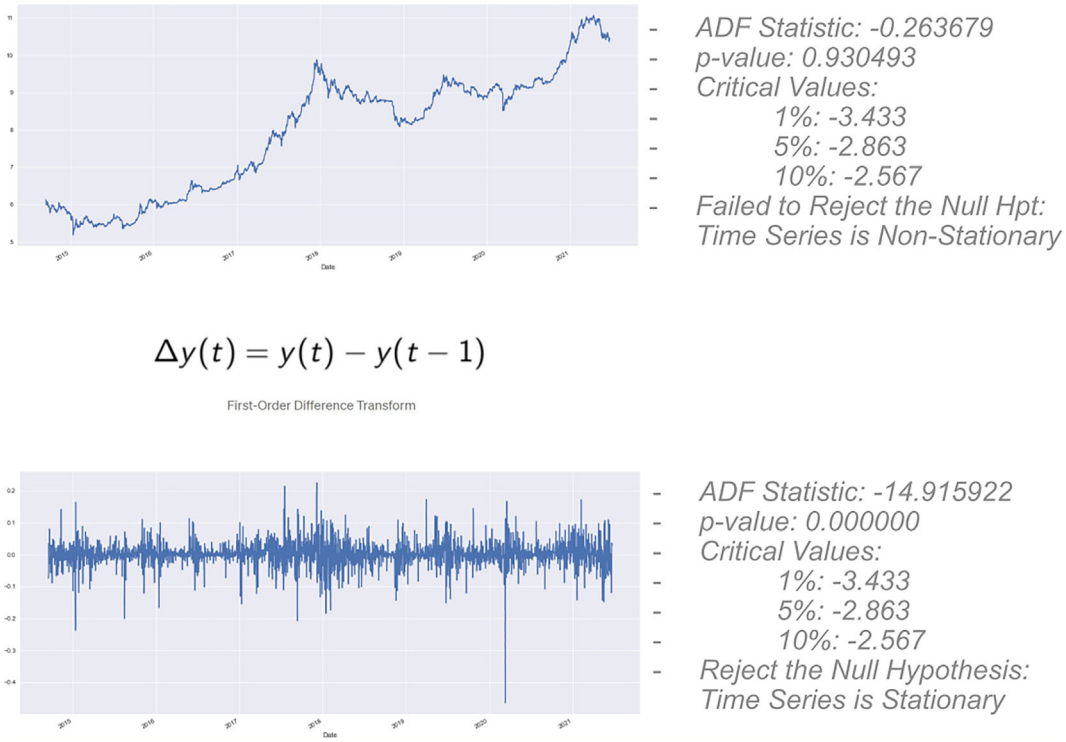

FIGURE 5 ADF stationary tests for prices versus returns.

3.2.2 | Applying the models

The datasets are divided into 60% training, 10% validation and 30% testing sets. All methods are backtested on six datasets representing six distinct cryptos. We should expect a high correlation between the results of the various datasets, as the crypto movement is usually related to market makers. We transform the close price to the hourly returns by taking the first order difference. This will ensure the stationarity of the dependent variable for some algorithms, especially statistical models, which require the variables to be stationary, as presented in Figure 5. We could then transform the returns to positive and negative returns when we are running a binary classification.

The buy and sell decisions were made based on the outcome of each of the models on the hourly dataset. If the predicted return of the next hour was positive, the model was executing a buy order and vice versa. These actions are automated using the backtesting library in python. We could specify the threshold for each action (i.e., if the result is above a certain figure buy, under a certain figure sell, otherwise hold). The following are the performance measures that will be taken into consideration when backtesting the algorithms:

1. F1 Score: The F1 score is one of the most commonly used metrics for classification models. It calculates the harmonic mean of the classifier precision and recall. This score will be the main score used in the classification tasks.
2. R-Squared: The R-squared, or the coefficient of determination, represents the proportion of the variation for the dependent variable, which is predicted by the independent variable. This score will be one of the scores used in the regression tasks.

3. Final Equity: The total final equity when we backtest the trained model. The model will be executing orders solely based on the projected direction of the market with an initial equity of 1,000$.
4. Return (%): The same metric as above but in percentages.
5. Buy & Hold Return (%): The return in case of the buy and hold strategy.
6. Sharpe Ratio: This score will take into consideration the volatility of the model by dividing by the standard deviation of the results. The Sharpe ratio calculates the risk-adjusted return of an asset.
7. *P* Value: The *P* value was calculated from the win rate percentages compared to a random guessing strategy. This score will determine the statistical significance of the classification results.
8. Interpretability: The capability of the algorithm to deliver interpretable results. Although this might be considered a subjective qualitative measure, we will list each algorithm's interpretability features and state our opinion on which might provide the most interpretable and informative outcomes based on the needs of the industry.
9. User Expertise: The expertise required by the user to train and test the models.
10. Computational Requirements: Most of the models were trained and tested using a laptop computer running on i7-8550u and 16GB of ram. Few of the models, especially the deep learning algorithms, were run on Google colab free service, and Google AutoML was running on google dedicated servers.
11. Related Costs: This will be specifically related to running models on Google AutoML as all other models were free of charge, not considering the cost of the user computer, electricity and depreciation.

3.2.3 | Interpretability features

Interpretability is a crucial feature in machine learning that is often overlooked. It can either facilitate or prevent the mass adoption of these technologies, especially in institutions. The demand for it is usually subjective as everyone possesses various levels of technical experience. In this section, we will explore the various interpretability features of the selected algorithms. We will focus primarily on the winner of this category, which was the decision trees algorithm as shown in the corresponding Tables 1 and 2 (The text in bold highlights the best scores in each category).

Decision trees and random forests can utilise various algorithms for their classification (ID3, CART, etc.). However, most of these methods are built on calculating information gain. Hence, ranking related features and extracting their importance is extremely easy.

TABLE 1 Quantitative performance measures.

Algorithm	F1 score	R-squared	Final equity	Return (%)	B&H return (%)	Sharpe ratio	*P* value
ES-RNN	0.71	0.22	23,782$	137%	212%	0.37	0.28
N-BEATS	0.75	0.31	25,857$	158%	212%	0.42	0.09
SVM	0.72	0.28	16,333$	63%	212%	0.23	0.22
Decition Trees	0.78	0.35	16,968$	69%	212%	0.58	0.25
Random Forest	0.81	0.31	19,482$	94%	212%	0.62	0.18
LSTM	0.79	0.25	21,468$	114%	212%	0.78	0.21
CNN	0.65	NA	18,953$	89%	212%	0.02	0.24
ARIMA	0.54	0.16	12,320$	23%	212%	0.17	0.32
Google AutoML	**0.85**	0.42	**32,657$**	**226%**	212%	0.52	0.09
Our proposed hybrid approach	0.83	0.39	29,674$	196%	212%	**0.81**	0.12

TABLE 2 Qualitative performance measures.

Algorithm	Interpretability	User expertise	Computational req.	Related costs
ES-RNN	NA	5	4	Run locally
N-BEATS	Trend-seasonality decomposition	5	3	Run locally
SVM	Per feature visualization	3	2	Run locally
Decition Trees	Features importance & tree visualization	2	2	Run locally
Random Forest	Features importance	3	3	Run locally
LSTM	NA	5	5	Free cloud services
CNN	Heatmaps	5	4	Free cloud services
ARIMA	Time series decomposition	4	1	Run locally
Google AutoML	Features importance	**1**	**1**	21.252/node hour

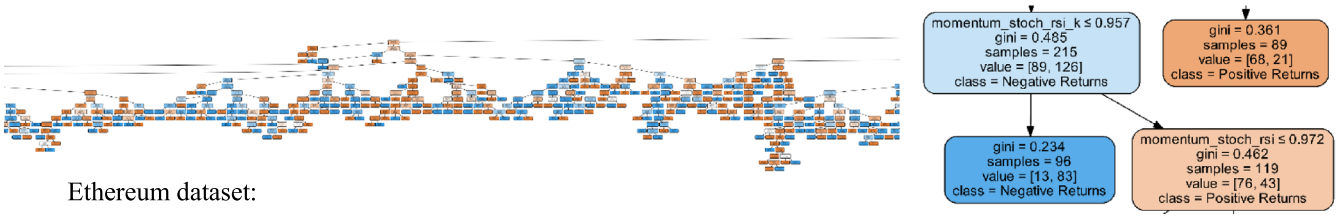

FIGURE 6 Part of the full decision tree visualisation (with zoomed-in portion).

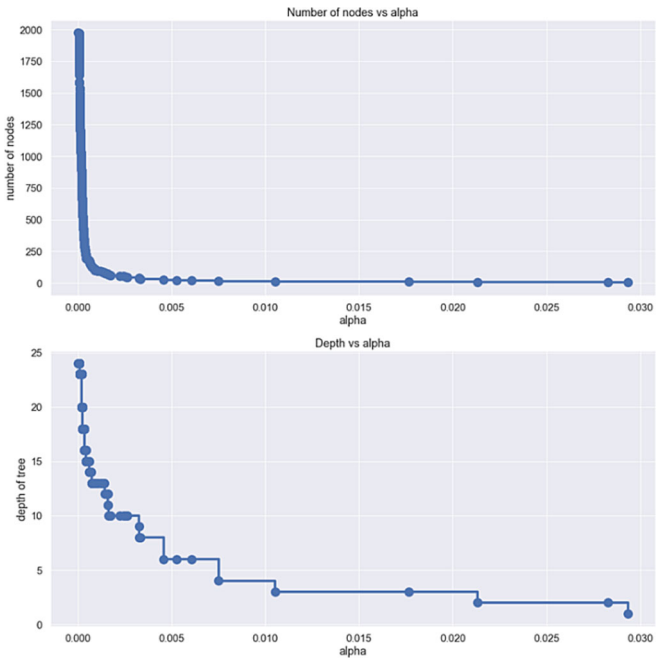

FIGURE 7 The effect of ccp|_alpha on the number of nodes and tree depth.

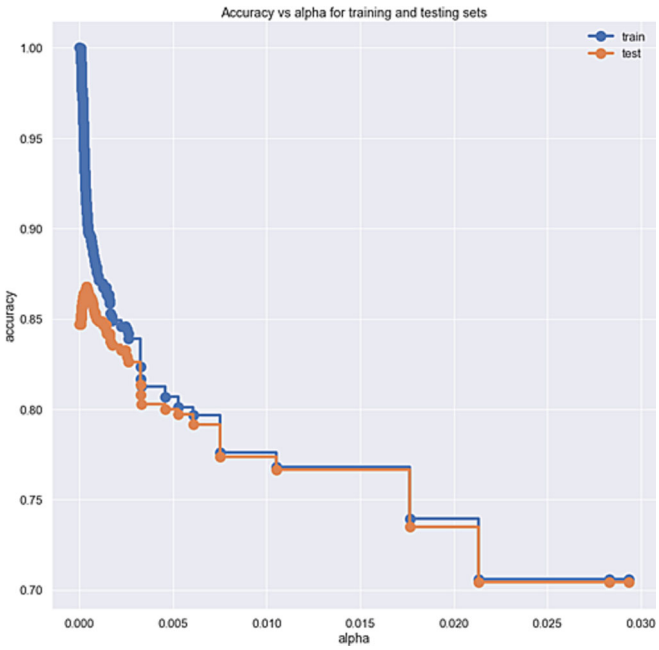

FIGURE 8 The effect of ccp|_alpha on the training and testing accuracy.

Furthermore, it is quite efficient to export the visualisation of the whole decision tree (Figure 6), which makes understanding the decision-making process intuitive for non-technical users. Figure 6 shows the complete decision tree, as well as, a very small zoomed-in section for the classification model run on the Ethereum dataset.

The algorithm keeps splitting the branches of the unpruned tree until it reaches 100% accuracy on the training set. Often, this leads to a drop in the testing accuracy and a relatively large tree that could not be intuitively understood. Luckily, numerous hyperparameters could be easily tweaked to prune the tree. This will reduce the tree's size and enhance the testing accuracy by preventing or reducing the overfitting of the model. From the various pruning parameters in the Sklearn python implementation of decision trees, ccp alpha is one of the most effective and efficient parameters. It is defined in the scikit-learn documentation (https://scikit-learn.org) as the 'Complexity parameter used for Minimal Cost-Complexity Pruning. The subtree with the largest cost complexity that is smaller than ccp_alpha will be chosen. By default, no pruning is performed'. The pruning parameter ccp_alpha has a major effect on the size of the tree. We could clearly notice the exponential decrease in the number of nodes and the depth of the corresponding tree, as presented in Figure 7.

Furthermore, we could visualise the direct effect of altering ccp_alpha on the training and testing accuracy. This will help us choose the ideal value considering the tree's desired size, as presented in Figure 8. After finding the ideal ccp alpha parameter, we could now prune the tree, ensuring the best results possible and a human-readable visualisation of the algorithm decision-making process, as presented in Figure 9.

We focused on describing the pruning process in detail to highlight its flexibility to cater for every user's needs. Even though decision trees might not be considered the best performer in accuracy, their capacity to deliver interpretable outcomes might make them an attractive solution to large businesses and teams. Figure 10 shows a zoomed-in portion of the previous tree in Figure 9.

The highlighted nodes explain how the algorithm is making every single branch with enough details yet a simple representation. This could be utilised in numerous ways, from gaining domain expertise to developing a better trading strategy that could be combined with

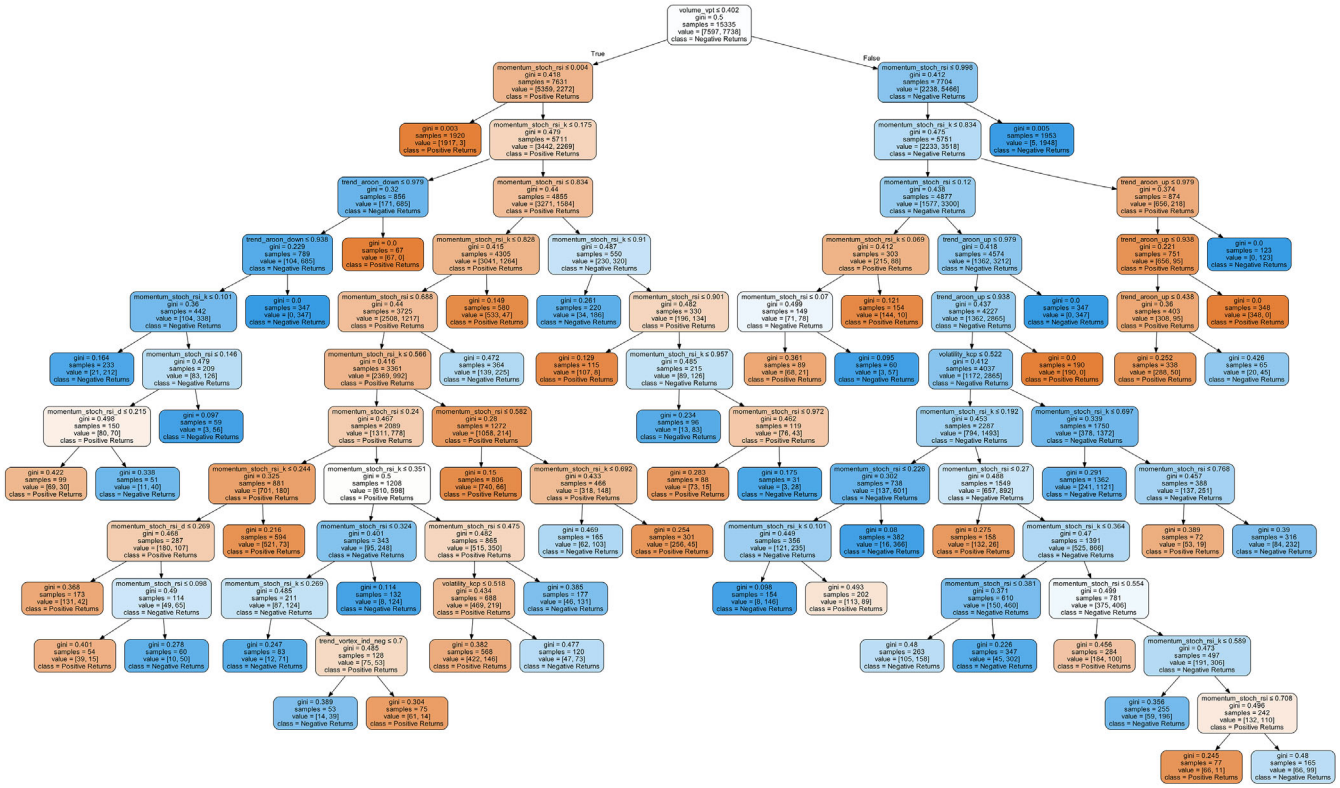

FIGURE 9 Pruned tree—classification.

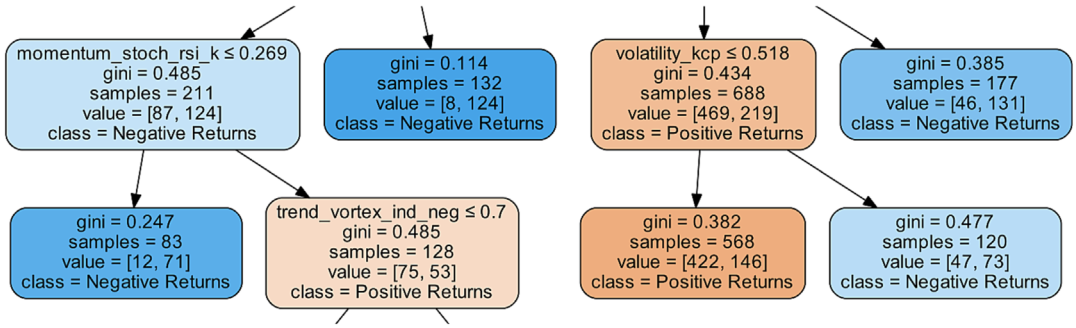

FIGURE 10 Decision tree zoom-in.

other better-performing algorithms and methods. The same process could also be applied to decision tree regressors. The corresponding tree visualisation presented in Figure 11 shows the pruned decision tree when it is run on the same dataset but as a regressor.

Furthermore, we could easily extract the feature's importance from each tree. Even though both trees were trained on the same dataset, we could notice some differences in the top 10 important features, as presented in Figure 12a,b.

Figure 12 shows that the lists are fairly different. However, we can notice the dominance of technical indicators, which indicates that they are more contributing to the formation of the tree than other types of indicators. The same is also observed when we extract the features importance from the Google AutoML algorithm as presented in Figure 13. This is the only interpretable outcome the algorithm can provide.

N-BEATS, on the other hand, can provide, when it is run on a single feature, with the trend-seasonality decomposition, as presented in Figure 14. This could be utilised in almost similar ways as the seasonality-trend-level conventional decomposition method by traders and practitioners (Robert et al., 1990).

Finally, although some algorithms will not be able to deliver interpretable outcomes, we could still visualise the actual trades and extract numerous relevant metrics when back-testing any method, as presented in Figure 15. These values could explain the automated trades but not the process that the algorithm has depended on executing those trades.

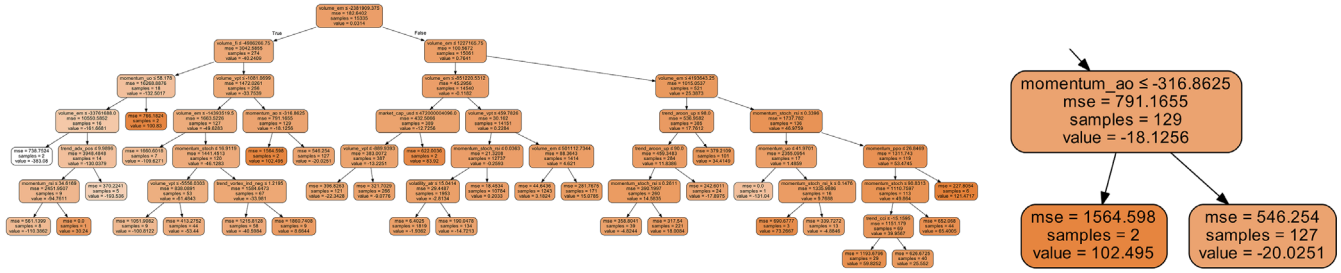

FIGURE 11 Pruned Tree – regression (with Zoomed-In Portion).

(a)

momentum_stoch_rsi	0.25705
momentum_stoch_rsi_k	0.19551
volume_vpt	0.18237
trend_aroon_up	0.06594
momentum_stoch_rsi_d	0.02327
trend_aroon_down	0.01876
volatility_kcp	0.01019
volatility_bbp	0.00974
trend_adx	0.00886
volatility_dcp	0.00870

(b)

volume_em	0.48963
volume_fi	0.07712
volume_vpt	0.06752
momentum_uo	0.05306
momentum_stoch_rsi	0.04178
trend_aroon_up	0.03485
momentum_stoch	0.02181
momentum_stoch_rsi_k	0.01988
momentum_ao	0.01855
momentum_ppo	0.01801

FIGURE 12 (a) Classifier features importance and (b) regressor features importance.

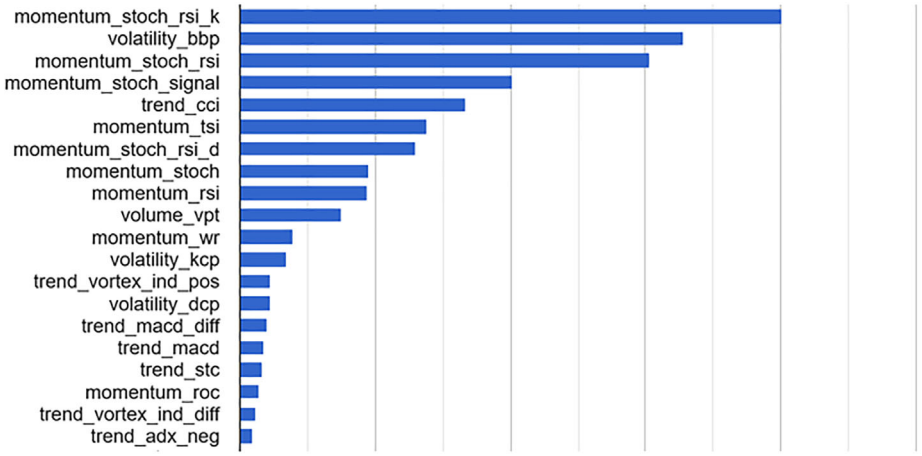

FIGURE 13 Google AutoML classifier features importance.

4 | ANALYSIS RESULTS AND FINDINGS

4.1 | Analysis results

Considering that some algorithms may have stochastic variables, all the models, except for Google AutoML, were trained and tested 10 times for each dataset. The figures in the tables below are the mean values of the corresponding results. Furthermore, to generalise the outcomes, we have taken the averages of the six cryptos and combined them into a single score for each method. The primary aim of this study is to compare various algorithms and approaches; hence, individual scores may not be that helpful. The

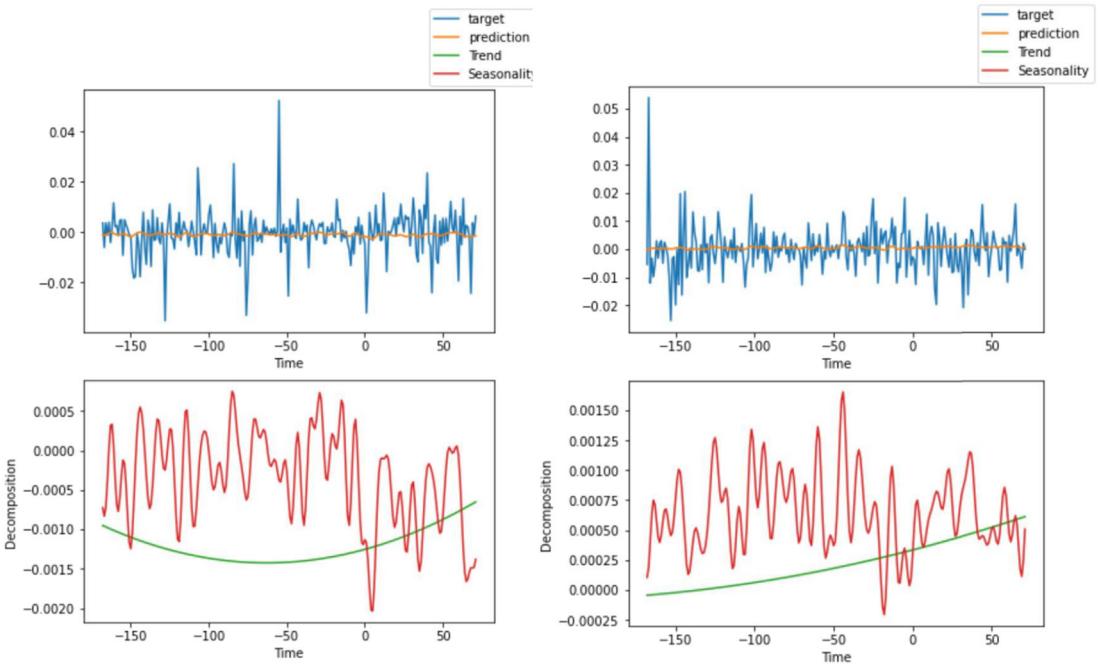

FIGURE 14 N-BEATS trend seasonality decomposition.

Start	2020-12-01 00:00:00
End	2021-07-01 23:00:00
Duration	212 days 23:00:00
Exposure Time [%]	98.572
Equity Final [$]	16225.6
Equity Peak [$]	26149.5
Return [%]	62.256
Buy & Hold Return [%]	71.244
Return (Ann.) [%]	129.194
Volatility (Ann.) [%]	268.906
Sharpe Ratio	0.480444
Sortino Ratio	2.14993
Calmar Ratio	2.54205

Max. Drawdown [%]	-50.8227
Avg. Drawdown [%]	-3.99993
Max. Drawdown Duration	77 days 02:00:00
Avg. Drawdown Duration	3 days 06:00:00
# Trades	529
Win Rate [%]	51.2287
Best Trade [%]	20.3106
Worst Trade [%]	-22.0158
Avg. Trade [%]	0.0915362
Max. Trade Duration	3 days 08:00:00
Avg. Trade Duration	0 days 10:00:00
Profit Factor	1.14974
Expectancy [%]	0.149105
SQN	0.447797

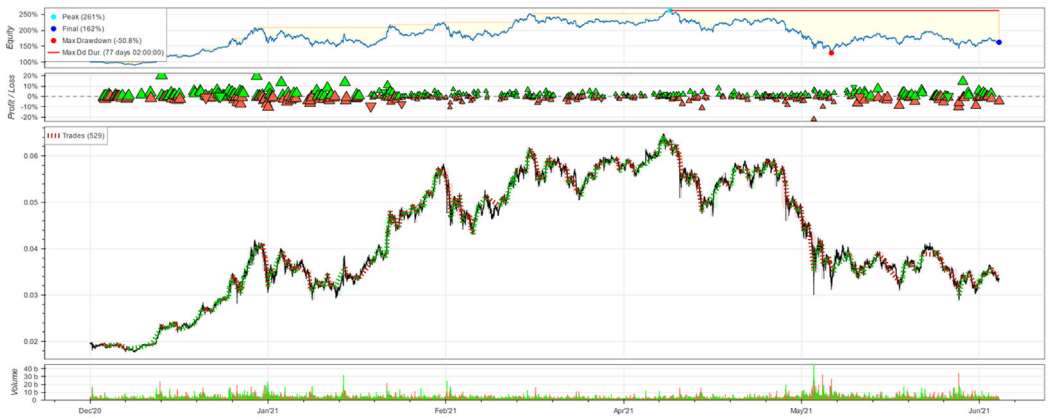

FIGURE 15 Trades metrics and visualization.

best result in each category is highlighted in bold font. The *User Expertise* and *Computational Req.* values range from 1 to 5, where 1 is the lowest and 5 the highest. The performance measures are presented in Tables 1 and 2. Table 1 shows the actual scores for the regression and classification models. In contrast, Table 2 shows the other factors that may not be directly related to the overall accuracy of the model but could be well considered when choosing a method.

4.2 | Findings

The following are the main findings of this evaluation study:

1. The final accuracy scores of the tested algorithms provided enough evidence to conclude that using machine learning in automated trading could be effective and efficient. However, the results may not be generalisable. This could limit the algorithm's effectiveness when the scope is increased to target long periods and/or multiple currencies and financial assets.
2. The *P*-value results proved to be statistically insignificant. This is due to the fact that the win rates were very close to the random guessing results. Furthermore, considering the limitations of our computing resources, we could not run the experiment a large number of times in order to stabilise the results and decrease the final *P* value. However, considering the difficulties in predicting the market, even win rates that hover around 50% could be used to formulate a winning trading strategy.
3. We could notice from the feature's importance that almost all the algorithms, when they are capable of handling multiple features, have benefited from the variety of the features. An algorithm that is trained solely on a single type of feature may have limited relative performance.
4. Furthermore, technical analysis has dominated the feature importance lists.
5. Google AutoML could be a very attractive solution to individuals or businesses that are looking for fast deployment with an emphasis on superior results. However, this would come with a financial cost and other disadvantages that could include limited customisation and no interpretable outcomes.
6. Even though they are considered relatively basic algorithms, decision tree and conventional ensembling methods like random forests were able to provide competing results when it comes to accuracy and the best interpretable outcomes. The tree visualisation and pruning are flexible yet crucial elements to institutional automated trading that may require interpretability without a major sacrifice to the overall performance.
7. Pruning the tree is a major step that will ensure the best performance by preventing overfitting and making the algorithm more generalisable over the testing dataset. Another overlooked benefit is the massive decrease in the number of nodes, which will render the tree human readable.
8. Creating a successful trading strategy will always be dependent on the relevant circumstances. There are numerous ways of executing the trades. The variety of metrics, visualisations and techniques would require extensive time, effort and money to reach optimality. The solutions have to be updated regularly to maintain good performance.

5 | CONCLUSION AND FUTURE WORK

The use of machine learning algorithms to predict financial markets is a highly complex task, and the paper aims to review and compare the effectiveness of selected algorithms from the existing literature with a focus on cryptocurrency markets. The main finding is that most were designed based on their capability of delivering an interpretable outcome. The results might have been slightly different if the sole focus was the overall accuracy. However, interpretability is a major component in algorithmic trading that might dictate the adoption and use of such technology. As for the specific approaches, Google AutoML outperformed other approaches in the accuracy department and was the only method to surpass the buy and hold returns, but it was very limited in the customisation, required considerable investment, and results are hard to interpret. Decision trees showed a good balance between interpretability and accuracy. N-BEATS scored an average accuracy but was the only algorithm that could provide the trend and seasonality decomposition of the time series. All algorithms emphasised the feature importance of technical indicators over fundamental ones and statistical analysis. From the comparative analysis of existing approaches, we found that it is feasible for machine learning algorithms to predict, with relatively high accuracy, the trend and direction of a cryptocurrency market. Future studies could include the optimisation of certain approaches. These include altering the dimensions of the input and output—the backcast and forecast length, testing various unique combinations of diverse features, adjusting the numerous hyperparameters of the algorithms in order to find the optimal configurations, and modifying the trades execution condition rules and thresholds. Moreover, a hybrid approach could be considered to combine the benefits of various algorithms.

ACKNOWLEDGEMENTS

Open access publishing facilitated by Australian Catholic University, as part of the Wiley - Australian Catholic University agreement via the Council of Australian University Librarians.

CONFLICT OF INTEREST STATEMENT

All authors confirm that there is no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from Omenics. Restrictions apply to the availability of these data, which were used under license for this study. Data are available from https://omenics.com/ with the permission of Omenics.

REFERENCES

Cao, L. (2022). AI in finance: Challenges, techniques, and opportunities. *ACM Computing Surveys (CSUR)*, *55*(3), 1–38. https://doi.org/10.1145/3502289

Cruz, B. G. S., Bossa, M. N., Soelter, J., Hertel, F., & Husch, A. (2022). The importance of dataset choice lessons learned from COVID-19 X-ray imaging models. In *Bildverarbeitung für die medizin* (Vol. 2022, p. 114). Springer. 10.1007/978-3-658-36932-3_24

Darin, S. G., & Stellwagen, E. (2020). Forecasting the M4 competition weekly data: Forecast Pro's winning approach. *International Journal of Forecasting*, *36*(1), 135–141. https://doi.org/10.1016/j.ijforecast.2019.03.018

Dixon, M. F., Halperin, I., & Bilokon, P. (2020). *Machine learning in finance*. Springer. 10.1007/978-3-030-41068-1

Dumitrescu, E., Hue, S., Hurlin, C., & Tokpavi, S. (2022). Machine learning for credit scoring: Improving logistic regression with nonlinear decision-tree effects. *European Journal of Operational Research*, *297*(3), 1178–1192. https://doi.org/10.1016/j.ejor.2021.06.053

Fischer, T. G. (2018). Reinforcement learning in financial markets-a survey. FAU Discussion Papers in Economics.

Gogas, P., & Papadimitriou, T. (2021). Machine learning in economics and finance. *Computational Economics*, *57*(1), 1–4. https://doi.org/10.1007/s10614-021-10094-w

Henrique, B. M., Sobreiro, V. A., & Kimura, H. (2019). Literature review: Machine learning techniques applied to financial market prediction. *Expert Systems with Applications*, *124*, 226–251. https://doi.org/10.1016/j.eswa.2019.01.012

Hussain, W., Gao, H., Raza, M. R., Rabhi, F. A., & Merigó, J. M. (2022). Assessing cloud QoS predictions using OWA in neural network methods. *Neural Computing and Applications*, *34*, 1–18. https://doi.org/10.1007/s00521-022-07297-z

Hussain, W., Merigó, J. M., & Raza, M. R. (2021). Predictive intelligence using ANFIS-induced OWAWA for complex stock market prediction. *International Journal of Intelligent Systems*, *37*, 4586–4611. https://doi.org/10.1002/int.22732

Hussain, W., Merigó, J. M., Raza, M. R., & Gao, H. (2022). A new QoS prediction model using hybrid IOWA-ANFIS with fuzzy C-means, subtractive clustering and grid partitioning. *Information Sciences*, *584*, 280–300. https://doi.org/10.1016/j.ins.2021.10.054

Hussain, W., Raza, M. R., Jan, M. A., Merigo, J. M., & Gao, H. (2022). Cloud risk management with OWA-LSTM predictive intelligence and fuzzy linguistic decision making. *IEEE Transactions on Fuzzy Systems*, *30*, 4657–4666. https://doi.org/10.1109/TFUZZ.2022.3157951

Hussain, W., & Sohaib, O. (2019). Analysing cloud QoS prediction approaches and its control parameters: Considering overall accuracy and freshness of a dataset. *IEEE Access*, *7*, 82649–82671. https://doi.org/10.1109/ACCESS.2019.2923706

Khan, A. T., Cao, X., Li, S., Katsikis, V. N., Brajevic, I., & Stanimirovic, P. S. (2022). Fraud detection in publicly traded US firms using Beetle Antennae Search: A machine learning approach. *Expert Systems with Applications*, *191*, 116148. https://doi.org/10.1016/j.eswa.2021.116148

Kim, M. J., Han, I., & Lee, K. C. (2004). Hybrid knowledge integration using the fuzzy genetic algorithm: prediction of the Korea stock price index. *Intelligent Systems in Accounting, Finance & Management: International Journal*, *12*(1), 43–60. https://doi.org/10.1002/isaf.240

Kumar, M., & Thenmozhi, M. (2014). Forecasting stock index returns using ARIMA-SVM, ARIMA-ANN, and ARIMA-random forest hybrid models. *International Journal of Banking, Accounting and Finance*, *5*(3), 284–308. https://doi.org/10.1504/IJBAAF.2014.064307

Lara-Benítez, P., Carranza-García, M., & Riquelme, J. C. (2021). An experimental review on deep learning architectures for time series forecasting. *International Journal of Neural Systems*, *31*(03), 2130001. https://doi.org/10.1142/S0129065721300011

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, *36*(1), 54–74. https://doi.org/10.1016/j.ijforecast.2019.04.014

Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2019). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. arXiv preprint arXiv:1905.10437.

Rabhi, F. A., Mehandjiev, N., & Baghdadi, A. (2020). State-of-the-art in applying machine learning to electronic trading. In *International workshop on enterprise applications, markets and services in the finance industry* (pp. 3–20). Springer. 10.1007/978-3-030-64466-6_1

Robert, C., William, C., & Irma, T. (1990). STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, *6*(1), 3–73.

Smyl, S. (2020). A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, *36*(1), 75–85. https://doi.org/10.1016/j.ijforecast.2019.03.017

Thakkar, A., & Chaudhari, K. (2021). A comprehensive survey on deep neural networks for stock market: the need, challenges, and future directions. *Expert Systems with Applications*, *177*, 114800. https://doi.org/10.1016/j.eswa.2021.114800

How to cite this article: El Majzoub, A., Rabhi, F. A., & Hussain, W. (2023). Evaluating interpretable machine learning predictions for cryptocurrencies. *Intelligent Systems in Accounting, Finance and Management*, *30*(3), 137–149. https://doi.org/10.1002/isaf.1538