Ethereum (ETH) Time Series Analysis and Predictions in Python

João Oliveira

[1] ISCTE – Instituto Universitário de Lisboa, Lisboa, Portugal
jbboa@iscte-iul.pt

Abstract. In recent years, cryptocurrencies have experienced a remarkable surge in popularity and adoption, thanks to groundbreaking advancements in blockchain technology that have facilitated fully digital and anonymous transactions with strengthened security measures.

However, the volatile nature of cryptocurrency prices has posed a significant challenge, as they exhibit rapid and often unpredictable fluctuations across trading platforms. As a result, accurately predicting cryptocurrency prices has emerged as a crucial and highly sought-after research domain.

Given the current setting, in our study, we employ various models (Traditional, Facebook Prophet and Deep Learning neural networks) under the CRISP-DM methodology to analyze and predict the daily prices of Ethereum which is the second-largest cryptocurrency in terms of market capitalization.

Our findings suggest that using both traditional ARIMA or GRU neural networks are the best approaches to make predictions on Ethereum daily prices, although LSTM´s also yielded satisfactory results.

Keywords: Séries Temporais, Ethereum, Previsões de preços, Criptomoedas.

1 Introdução

Com o aparecimento da *Decentralized Finance* (DeFi) houve uma transformação dos sistemas financeiros tradicionais, procurando eliminar intermediários, aumentar a transparência e fornecer serviços constantemente abertos e disponíveis a qualquer pessoa com uma ligação à Internet [1]. Esta transformação deu origem às criptomoedas. Estas são moedas digitais ou virtuais que utilizam a criptografia para realizarem transações seguras e controlarem a criação de novas unidades. Ao contrário das moedas tradicionais emitidas por bancos centrais, as criptomoedas operam em redes descentralizadas chamadas *blockchains*, nas quais os registos são distribuídos por uma rede de computadores (nós).

O *Ethereum*, criptomeda alvo de estudo, é a segunda maior criptomoeda em termos de capitalização de mercado [2] e corresponde a uma plataforma de *blockchain* que permite o desenvolvimento de contratos inteligentes e aplicações descentralizadas (*DApps*). Este tem possibilitado avanços significativos na indústria ao permitir que programadores criem e implementem as suas próprias aplicações na rede [1].

A análise de séries temporais fornece um conjunto de ferramentas poderosas para modelar e efetuar previsões de dados sequenciais, como é o caso dos preços diários de

criptomoedas. É neste âmbito que se desenvolve o presente relatório, que exibe enquanto principal foco a obtenção das previsões mais precisas quanto possível para os valores diários do *Ethereum* através da exploração de uma série de técnicas de previsão, incluindo métodos estatísticos clássicos como os métodos ARIMA (*AutoRegressive Integrated Moving Average*), modelos de aprendizagem profunda/*deep learning* como as redes neuronais recorrentes (RNN) e as redes de memória de longo-curto prazo (LSTM) e ainda o *Facebook Prophet*. Para cumprir esta tarefa, os modelos clássicos foram utilizados para servir de referência para a comparação/*benchmark* enquanto os modelos de aprendizagem profunda e o *Facebook Prophet* foram investigados quanto à sua capacidade de captar padrões temporais mais complexos nos dados de preços do *Ethereum*.

Desta forma, o estudo tem início com uma breve descrição da metodologia CRISP-DM (secção 2) com o objetivo de compreender os ganhos gerados pela utilização desta metodologia e melhor enquadrar a abordagem seguida no estudo. Em seguida, na secção 3, é efetuada uma análise da problemática adjacente aos dados. Na secção 4, é efetuada uma análise exploratória dos dados usados no estudo. A secção 5 aborda a preparação dos dados focando-se nas alterações efetuadas aos mesmos. A secção 6 corresponde à modelação e nela é realizado um breve enquadramento teórico seguid das principais notas da aplicação desse tipo de modelos e dos procedimentos efetuados. Por fim, na secção 7 observam-se os resultados obtidos para cada um dos modelos a nível do conjunto de treino e conjunto de teste com a sua respetiva discussão sendo efetuada uma avaliação dos modelos e na secção 8 são apresentadas as conclusões e as disposições finais.

Toda o trabalho foi efetuado recorrendo ao *Python* e às suas bibliotecas em ambiente local *Jupyter Notebooks*.

2 Metodologia CRISP-DM

A metodologia CRISP-DM (Anexo 1) foi a metodologia selecionada não apenas por ser o *standard* da indústria, mas também por permitir ganhos em diversas vertentes. Efetivamente, as vantagens referidas enumeram-se em seguida:

- Processo claro e estruturado: oferece um processo claro, estruturado e bem definido que ajuda a garantir que todos os passos necessários são seguidos, reduzindo as hipóteses de aspetos importantes serem negligenciados.
- Flexibilidade: permite personalização com base nas características dos dados e nos objetivos e necessidades específicos da análise.
- Foco no negócio: enfatiza a importância de compreender os objetivos e requisitos do negócio antes de mergulhar na análise dos dados, assegurando que os conhecimentos obtidos são relevantes.
- Natureza iterativa: suporta uma abordagem iterativa, permitindo a melhoria contínua através da incorporação de novos dados, do refinamento de modelos ou do ajuste de estratégias utilizadas.
- Avaliação e validação: enfatiza a avaliação e validação de modelos para garantir a sua eficácia e exatidão, utilizando métricas de avaliação adequadas.

3 Business Understanding

Para compreender adequadamente o problema em questão e poder convertê-lo num problema analítico bem definido foi crucial compreender o significado dos valores da nossa série temporal *Coinbase Ethereum*. A série foi obtida a partir do website da *Federal Reserve Bank of St. Louis* disponível no seguinte link: https://fred.stlouisfed.org/series/CBETHUSD.

A série representa os valores diários da criptomoeda *Ethereum* em dólares americanos (USD). É importante referir que o preço do *Ethereum* é determinado por um mercado global de oferta e procura. Consequentemente, este pode ser volátil a curto prazo, uma vez que a procura tende a superar a oferta e vice-versa [2].

Desta forma, a análise e previsão dos valores diários do *Ethereum* podem proporcionar um valor significativo para os principais interessados, nomeadamente, os investidores, *traders* e investigadores do mercado de criptomedas dado que os insights poderão auxiliar no processo de tomada de decisões relacionadas com investimentos ou pesquisas futuras nesta área que se encontra em constante evolução.

4 Data Understanding

De maneira a se poder obter uma melhor compreensão dos dados foi levada a cabo uma análise exploratória dos mesmos que é possível desdobrar nas seguintes etapas: informação geral, estatísticas descritivas, distribuição e análise de *outliers*, decomposição da série temporal e logaritmação da série.

4.1 Informação geral do dataset

Inicialmente foi observada a informação geral do *dataset* através da qual foi possível perceber que a série temporal a ser estudada detinha uma periodicidade diária e um total de 2.581 entradas correspondentes ao período compreendido entre os dias 18 de maio de 2016 e 11 de junho de 2023. Destas entradas três apresentavam valores omissos que obrigaram à imputação desses valores (explicação do procedimento realizado disponível no capítulo 5).

4.2 Estatísticas descritivas, distribuição e análise de *outliers*

Após a imputação foram observados o histograma e o gráfico de bigodes (presentes nos Anexos 2 e 3) assim como as estatísticas descritivas da série temporal que se apresentam na tabela em seguida:

Tabela 1. Estatísticas descritivas da série *Coinbase Ethereum* (após imputação)

count	mean	std	min	25%	50%	75%	max	kurtosis	skewness
2.581	951,502	1.115,689	6,75	170,16	361,51	1.593,93	4.805,95	1,032	1,382

Dos resultados podemos claramente compreender que a distribuição é enviesada para a direita, com uma elevada concentração de valores na extremidade inferior e um conjunto de noventa e dois (92) *outliers* na extremidade superior (quando os preços se encontravam nos seus valores mais elevados). Este facto é validado pelo valor de *skewness* positivo de 1,382 que foi obtido. Também podemos ver que a distribuição é ligeiramente leptocúrtica uma vez que se obteve um valor de *kurtosis* na ordem dos 1,032.

4.3 Decomposição da série temporal

Em termos da série temporal em si, foi possível ver que não há uma tendência linear evidente, nem uma sazonalidade óbvia, nem valores negativos, embora haja muitos altos e baixos ao longo dos anos (Anexos 4 e 5).

É importante notar o grande salto nos preços que aconteceu em 2021 e que a variância não parece ser constante, como foi através da divisão da série em duas metades e da comparação dos valores da média e da variância de cada uma das metades comprovado. Por conseguinte, a série cronológica não poderá ser estacionária. No entanto, os testes de raiz unitária e outros testes de estacionariedade continuarão a ser efectuados de modo a se poder afirmar tal com certeza.

Tabela 2. Valores da média e da variância para cada metade da série *Coinbase Ethereum*

Média	248,74 (1ª metade)	1.653,72 (2ª metade)
Variância	61.785,66 (1ª metade)	1.439.259,39 (2ª metade)

4.4 Aplicação da função logarítmica

Tal como anteriormente identificado, uma vez que a série *Coinbase Ethereum* não apresentava quaisquer valores negativos foi testada a aplicação de uma transformação logarítmica. Esta transformação permitiu aproximar a distribuição da série temporal à distribuição normal (Anexo 6) e também resultou na desconsideração dos noventa e dois outliers anteriormente identificados, não sendo agora encontrado qualquer *outlier* (Anexo 7).

5 Data Preparation

No que concerne à preparação de dados foi realizada a renomeação de colunas, a imputação de valores omissos, a divisão dos dados em conjuntos de treino e de teste, a estacionarização da série e a formatação dos dados na forma requerida pelos modelos utilizados, nomeadamente, pelas redes neuronais e pelo *Facebook Prophet.*

A nível das renomeações a série *Coinbase Ethereum* teve a sua variável '*CBETHUSD*' renomeada para '*Ethereum_Prices*' a fim de ser mais fácil compreensão.

A nível da imputação dos valores omissos da série *Coinbase Ethereum*, foram testados vários métodos de imputação (média, mediana, interpolação linear e *forward*

fill) tendo as estatísticas descritivas da série após cada um desses mesmos métodos de imputação sido comparadas com as da série temporal antes de qualquer imputação ser realizada com vista a ser observado a imputação que manteria o mais possível as propriedades originais dos dados. Foi possível compreender que o método de imputação *forward fill* foi o que nesse respeito obteve melhores resultados, tendo sido aquele que foi selecionado para realizar a imputação.

A nível da divisão em conjunto de treino e de teste foi, de acordo com o requerido, criado um conjunto de teste com cem observações e um conjunto de treino com as remanescentes (Anexo 8).

A nível da estacionariedade, da aplicação dos testes ADF, PP e KPSS e dos seus resultados foi possível, através de uma decisão unânime dos mesmos, compreender, tal como era inicialmente suspeitado, que tanto a série temporal original como a obtida da transformação logarítmica não eram estacionárias e que dado que a transformação logarítmica por si só não foi tinha sido suficiente para que a série atingisse a estacionariedade, seria necessário proceder-se à diferenciação de primeira ordem de ambas as séries e aplicar novamente os testes ADF, PP e KPSS. Tal foi feito e, desta vez, tinha sido alcançada a estacionariedade uma vez que os três testes para tal apontavam.

As formatações serão descritas no capítulo posterior da modelação.

6 Modelling

Dado tratar-se de um estudo comparativo de diferentes métodos de previsão de preços da criptomoeda *Ethereum* foram consideradas metodologias diversas, de forma a se obter uma *overview* geral das abordagens possíveis.

6.1 Métodos estatísticos clássicos – Modelos ARIMA/SARIMA

Enquadramento teórico

Os modelos ARIMA são utilizados para a análise de séries temporais e incluem componentes autorregressivos (AR), de diferenciação (I) e de média móvel (MA).

A componente AR capta a dependência do valor atual em relação aos seus próprios valores anteriores, a componente I representa a diferenciação aplicada à série temporal para assegurar a estacionariedade e a componente MA modela os choques aleatórios (dependências entre a observação atual e uma combinação linear de termos de erro passados/resíduos).

Os parâmetros de ordem (p, d, q) determinam o número de observações e de resíduos desfasados a serem considerados no modelo e o número de vezes que foi necessária a aplicação da diferenciação para alcançar a estacionariedade.

Já o modelo SARIMA é uma extensão do ARIMA que incorpora componentes sazonais (P, D, Q) para ter em conta padrões sazonais claros nos dados que precisam de ser tidos em conta com vista a melhorar as previsões.

A componente sazonal do SARIMA é indicada pelos parâmetros adicionais P, D e Q, que representam as ordens AR sazonal, diferenciação sazonal e MA sazonal, respetivamente.

Notas da aplicação dos modelos

Os modelos clássicos foram aplicados a dois conjuntos de dados distintos, ao original e ao resultante da transformação logarítmica. Foram criados quatro modelos com base nas seguintes iterações:

- Iteração 1 – SARIMA (1,1,0) (0,0,0) [0] – utilizando os dados originais;
- Iteração 2 – SARIMA (1,1,0) (1,0,0) [6] – utilizando os dados originais;
- Iteração 3 – SARIMA (1,1,0) (1,0,0) [5] – utilizando os dados originais;
- Iteração 4 – SARIMA (2,1,1) (0,0,0) [0] –utilizando os dados transformados.

Descrição do processo

O processo teve início com a utilização do *auto-arima* de maneira a serem alcançadas as ordens do modelo ótimas a utilizar com base no critério de informação AIC (*Akaike Information Criterion*). O *auto-arima* foi inicializado tendo por base as visualizações dos gráficos ACF e PACF (Anexo 9), tendo do mesmo sido obtido um modelo SARIMA (1,1,0) (0,0,0,0).

O modelo foi depois treinado com o conjunto de treino e calculadas tanto as previsões para o conjunto de treino como para o de teste. (Anexos 10 e 11)

Contudo, e uma vez que no *Correlogram* apresentado aparentavam existir dois *lags* significativos para os valores cinco e seis (Anexo 12) foram testados também modelos que tentassem incorporar esses padrões contidos nos resíduos e calculadas as suas previsões para os conjuntos de treino e teste (Anexos 13 e 15). No entanto, para cada um dos modelos referidos continuou-se a observar a existência de um *lag* significativo (Anexos 14 e 16).

A última iteração deveu-se à tentativa de utilização do conjunto de treino transformado pela função logarítmica, para o qual à semelhança da primeira iteração foi utilizado o *auto-arima*. Os parâmetros deste foram inicializados tendo em consideração as visualizações dos gráficos ACF e PACF (Anexo 19).

À semelhança do já realizado foram calculadas as previsões para o conjunto de treino e para o conjunto de teste (Anexo 17). Relativamente a este modelo é ainda possível referir que o mesmo não apresenta qualquer *lag* significativo no seu *Correlogram* (Anexo 18).

6.2 Facebook Prophet

Enquadramento teórico

O *Facebook Prophet* é uma biblioteca de previsão de séries temporais *open-source* desenvolvida pelo *Facebook*.

O *Prophet* consiste num modelo de regressão aditiva ou multiplicativa que combina três componentes principais: tendência, sazonalidade e efeitos de dias feriados.

Este é um modelo que, acima de tudo, oferece simplicidade e facilidade de utilização aliada de flexibilidade uma vez que permite aos utilizadores ajustar o modelo de previsão de acordo com as suas necessidades específicas através da especificação de padrões de sazonalidade, do tratamento de *outliers* e da incorporação de regressores adicionais, se disponíveis.

Notas da aplicação dos modelos

O *Facebook Prophet*, à semelhança dos métodos clássicos foi aplicado ao conjunto de dados original e ao resultante da transformação logarítmica. Foram deste modo criados três modelos com base nas seguintes iterações:

- Iteração 1 – *Facebook Prophet* – utilizando os dados originais;
- Iteração 2 – *Facebook Prophet* com *fine tuning* – utilizando os dados originais;
- Iteração 3 – *Facebook Prophet* –utilizando os dados transformados.

Descrição do processo

O processo teve início com a alteração da formatação na qual se encontravam os dados visto este ser um requerimento para a utilização do modelo. Assim, antes de se fazer o *fit* do mesmo utilizando o conjunto de treino procedeu-se à criação de uma cópia do *dataset* original (*eth*) na qual se efetuou a reposição do índice (onde se encontravam as datas) e a renomeação da coluna com a variável alvo e com as datas. Estas viram os seus nomes serem alterados de '*Ethereum_Prices*' e '*DATE*' para '*y*' e "*ds*", respetivamente.

Após a formatação inicial foi criado o modelo e calculadas as suas previsões para o conjunto de treino e de teste (Anexos 20 e 21). Dado os resultados obtidos terem sido considerados como pouco satisfatórios tentou-se numa segunda iteração optar pela realização do *fine tuning* do modelo.

Na segunda iteração para se realizar o *fine tuning* foi criada uma *grid* com várias combinações possíveis de hiperparâmetros apresentados na seguinte tabela:

Tabela 3. Hiper parâmetros utilizados no tuning do modelo *Facebook Prophet*

'changepoint_prior_scale'	'seasonality_prior_scale'	'seasonality_mode'	'changepoint_range'
0.01	0.1	multiplicative	0.8
0.1	1.0	additive	0.9
0.5	10.0	----------	0.95

Foi feita a busca pela melhor combinação com base no critério RMSE e uma vez encontrada a mesma foi construído o modelo e calculadas as previsões para o conjunto de treino e de teste (Anexos 22 e 23).

Por fim, para a terceira iteração foi testado o modelo para os dados transformados. Para isto foi realizada novamente uma formatação dos dados à semelhança do que foi feito para os originais e depois foi realizado o *fit* do modelo para o conjunto de treino. Foram novamente calculadas as previsões para os conjuntos de treino e de teste (Anexos 24 e 25) só que desta vez de maneira a serem calculadas as métricas foi necessária a utilização da função exponencial de maneira que os valores preditos fossem colocados de novo na escala original da variável alvo.

6.3 Modelos de aprendizagem profunda/*Deep learning*

Enquadramento teórico

As redes neuronais recorrentes (RNN) são um tipo de rede neuronal artificial concebida especificamente para o processamento sequencial de dados, sendo, portanto, especialmente adequadas a análise de séries temporais.

Ao contrário das redes neuronais *feed-forward* tradicionais, as RNNs têm conexões recorrentes que formam um ciclo de *feedback*. Esta característica permite-lhes manter uma memória interna de informações relativas a momentos anteriores que fornecem ao modelo o histórico associado ao fenómeno em estudo.

Dentro das redes neuronais recorrentes existem duas variantes comumente utilizadas para lidar com dependências de longo prazo: a *Long Short-Term Memory* (LSTM) e a *Gated Recurrent Unit* (GRU).

As LSTM foram introduzidas para superar o problema de explosão do gradiente, comum nas RNN tradicionais. Elas possuem mecanismos denominados de "portões" (*gates*) que controlam o fluxo de informações e a atualização do estado celular, tornando-as muito eficazes.

Por outro lado, as GRUs são uma versão simplificada das LSTM possuindo uma arquitetura mais compacta, com menos componentes. Tal foi possível através da combinação do estado oculto e do estado de memória das LSTMs num único estado chamado "estado de atualização".

Desta forma, as GRU capturem informações relevantes semelhantes às LSTM, mas com menos parâmetros, sendo, portanto, computacionalmente mais eficientes.

Notas da aplicação do modelo

Em relação às Redes Neuronais Recorrentes foram testados um modelo LSTM e um modelo GRU. Os modelos criados apenas foram aplicados ao conjunto de dados original. Assim, foram criados três modelos com base nas seguintes iterações:

- Iteração 1 – *Long Short-Term Memory* – utilizando os dados originais;
- Iteração 2 – *Gated Recurrent Unit* –utilizando os dados originais.

Descrição do processo

O processo teve início com a formatação dos dados necessária à utilização das redes. Assim, foi um criado um novo *dataframe* denominado *df_redes_lags* ao qual foram adicionados para cada entrada os seus dez valores desfasados. Em seguida, foi feita a separação dos dados entre conjuntos X, Y, treino e teste. Posteriormente, para a primeira iteração foi feito o *fit* de uma rede neuronal com uma camada LSTM e calculadas as previsões para os conjuntos de treino e de teste (Anexos 26 e 27).

Por fim, na última iteração foi criado uma rede neuronal GRU também com uma camada LSTM e calculadas as previsões para os conjuntos de treino e de teste (Anexos 28 e 29).

7 Resultados e Discussão/*Evaluation*

A avaliação da qualidade dos modelos foi feita com base na combinação de dois critérios: em primeiro lugar a magnitude do erro apresentado pelas previsões e em segundo lugar, o ajustamento do modelo aos dados/série temporal.

Tal ordem foi decidida uma vez que foi considerado como principal objetivo do estudo a minimização dos erros de previsão e a obtenção das previsões mais exatas para os valores diários do Ethereum. Assim, foram preferidos os modelos que demonstraram as métricas de erro mais baixas e não necessariamente aqueles com o maior ajustamento sendo métricas de erro baixas e ajustamento elevado a combinação ótima.
Tendo isso em consideração foram selecionadas as seguintes métricas para a avaliação dos resultados:

- *Root Mean Squared Error* (RMSE): é o desvio médio entre os valores previstos (Y_pred) e os valores reais (Y_true), fornecendo uma indicação da exatidão com que o modelo prevê a variável-alvo, sendo que valores mais baixos indicam um melhor desempenho. ex: RMSE = sqrt(Σ((Y_true - Y_pred)^2) / n)
- *Mean Absolute Error* (MAE): é a diferença média absoluta entre os valores previstos (Y_pred) e os valores reais (Y_true), representando a magnitude média dos erros cometidos pelo modelo, sem considerar a sua direção. ex: MAE = (1 / n) * Σ(|Y_true - Y_pred|)
- *Mean Absolute Percentage Error* (MAPE): é a diferença percentual média entre os valores previstos (Y_pred) e os valores reais (Y_true), relativamente aos valores reais, representando o erro relativo médio das previsões do modelo, expresso em percentagem. ex: MAPE = MAPE = (1 / n) * Σ(|(Y_true - Y_pred) / Y_true|) * 100
- *R-squared* (R^2): é uma medida que mede a proporção da variação na variável-alvo (Y_true) que é explicada pelas previsões do modelo (Y_pred), fornecendo uma indicação de quão bem o modelo se ajusta aos dados. Os valores variam entre 0 e 1, sendo que 1 representa um ajuste perfeito. ex: R^2 = 1 - (SSR / SST) = 1 - (Σ((Y_true - Y_pred)^2) / Σ((Y_true - Y_mean)^2))

Na seguinte tabela colocámos os modelos que obtiveram melhores resultados acompanhados das suas respetivas métricas:

Tabela 3. Métricas para os conjuntos de treino (A) e teste (B) de cada um dos modelos testados

Modelo	**RMSE**	**MAE**	**MAPE**	$\boldsymbol{R^2}$
Baseline (ARIMA (1,1,0) (0,0,0) [0]) A)	69,187	33,081	3,711	0,996
Baseline (ARIMA (1,1,0) (0,0,0) [0]) B)	47,432	34,360	1,894	0,857
ARIMA (1,1,0) (1,0,0) [6] A)	68,809	32,944	3,726	0,996
ARIMA (1,1,0) (1,0,0) [6] B)	47,803	35,053	1,931	0,855
ARIMA (1,1,0) (1,0,0) [5] A)	69,089	33,070	3,725	0,996
ARIMA (1,1,0) (1,0,0) [5] B)	47,671	34,683	1,912	0,856
ARIMA (2,1,0) (0,0,0) [0] A)	69,195	33,168	3,718	0,996
ARIMA (2,1,0) (0,0,0) [0] B)	163,525	51,856	3,000	-0,695
Facebook Prophet A)	283,608	192,454	119,260	0,936

Facebook Prophet B)	1.411,043	1.385,755	75,742	-125,203
Facebook Prophet (tuned) A)	235,702	155,360	78,668	0,956
Facebook Prophet (tuned) B)	380,268	360,227	19,543	-8,166
Facebook Prophet – transformado A)	275,576	141,511	15,078	0,940
Facebook Prophet – transformado B)	890,251	875,316	47,679	-49,236
Long Short-Term Memory A)	77,529	48,455	5,266	0,995
Long Short-Term Memory B)	70,614	58,384	3,207	0,684
Gated Recurrent Unit A)	71,792	38,318	4,164	0,996
Gated Recurrent Unit B)	47,085	34,403	1,890	0,859

*__NOTA__: Embora os modelos tenham tido um bom desempenho no conjunto de treino, o seu desempenho no conjunto de teste proporcionou uma avaliação mais realista das suas capacidades de previsão.

Ao analisar e comparar os diferentes modelos, apresentados na tabela, é possível retirar várias conclusões importantes:

Os modelos ARIMA, apresentaram um bom desempenho em termos de precisão e capacidade de captar a variação presente nos dados tendo fornecido uma linha de base sólida para efetuar posteriores comparações.

Os modelos criados com o *Facebook Prophet* ficaram aquém dos demais em termos dos resultados apresentados, tendo obtido valores bastante mais altos para RMSE, MAE e MAPE indicando erros de previsão maiores em comparação com os modelos ARIMA e LSTM/GRU. Embora tenha havido uma tentativa de melhorar o desempenho através do *fine tuning*, estes continuaram a não corresponder à eficácia dos outros modelos mesmo tendo apresentado uma melhoria significativa face à iteração prévia.

Os modelos LSTM e GRU, por outro lado, superaram os modelos do *Facebook Prophet* ao apresentarem valores RMSE e MAE mais baixos e demostraram a sua habilidade para capturar as dependências temporais presentes nos dados, resultando em valores menores de MAPE.

Assim, os modelos ARIMA e GRU devem ser considerados enquanto os melhores no desempenho de tarefas de previsão deste âmbito.

No entanto, além do já referido, é importante destacar o impacto do *fine tuning* no desempenho dos modelos, uma vez que a versão aprimorada do modelo *Facebook Prophet* apresentou, em geral, uma melhor precisão em comparação com os seus homólogos de base. Tal vem sugerir que uma otimização cuidadosa dos parâmetros conduzirá a ainda melhores resultados de previsão.

8 Conclusões

Com base nos resultados observados conclui-se que os modelos clássicos e os mais complexos, como as redes neuronais profundas, conseguiram melhor capturar a natureza das relações entre os preços diários do *Ethereum* obtendo, consequentemente, melhores resultados de precisão.

Desta forma a ser realizado o *Deployment* de um modelo este poderia ser o GRU uma vez que este foi aquele que conseguiu uma melhor representação dos dados que, no final, se refletiu nas métricas apresentadas pelo mesmo.

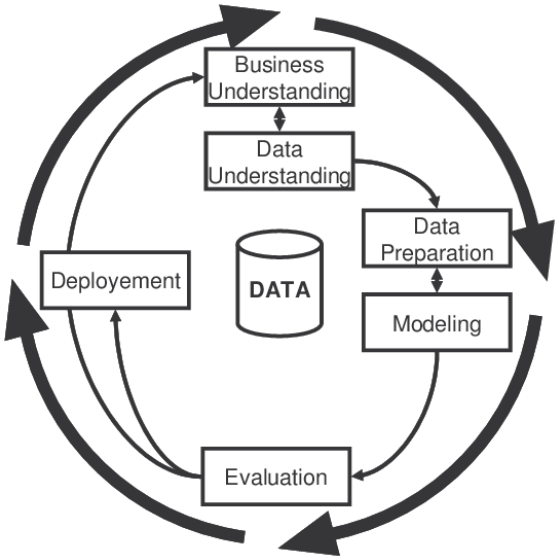
Todavia, poderia ser interessante explorar abordagens distintas como a utilização de outro tipo de transformações aos dados ou otimizações dos modelos já desenvolvidos por forma a se poderem alcançar resultados mais sólidos.

Ainda assim, acredito que o presente trabalho tenha sido bastante frutífero e enriquecedor dado os modelos e a análise descrita e apresentada ao longo do relatório.

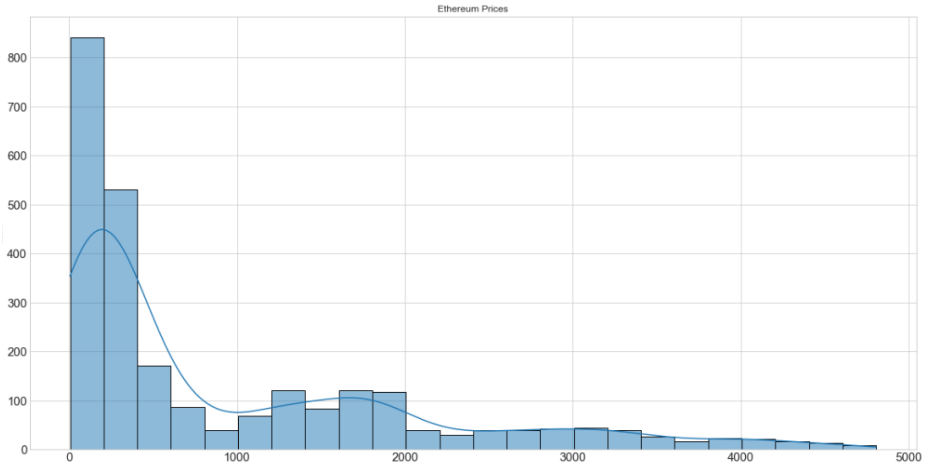
References

1. Ethereum Decentralized Finance (DeFi). Retrieved June 20, 2023, from https://ethereum.org/en/defi/#bitcoin
2. Coinbase Ethereum Prices. Retrieved June 20, 2023, from https://www.coinbase.com/pt-PT/price/ethereum

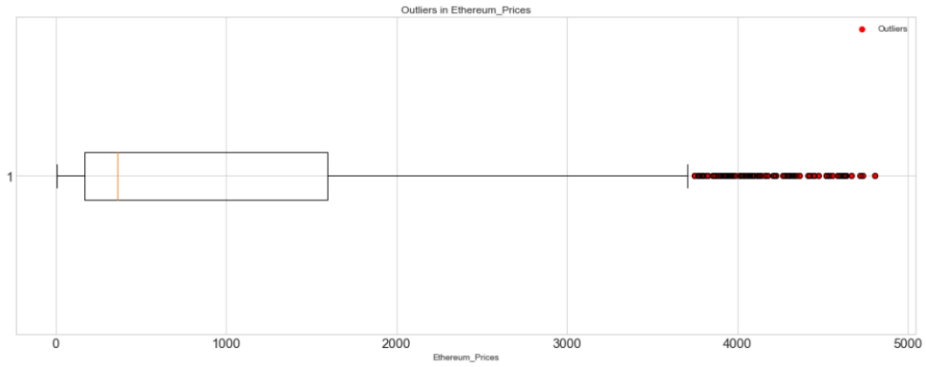
Anexos



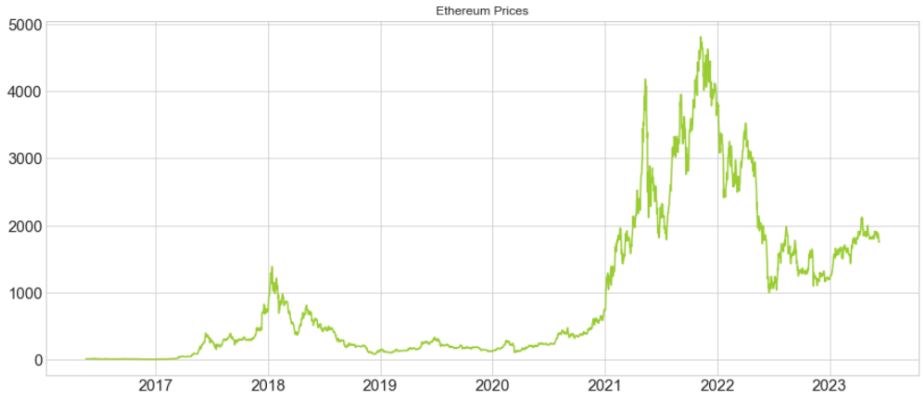
Anexo 1. Representação gráfica da metodologia CRISP-DM.



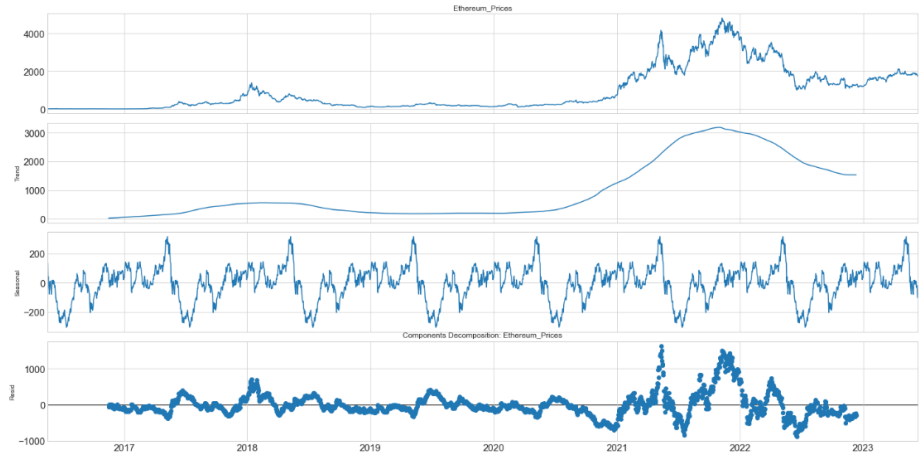
Anexo 2. Histograma da série temporal *Coinbase Ethereum* (após imputação)



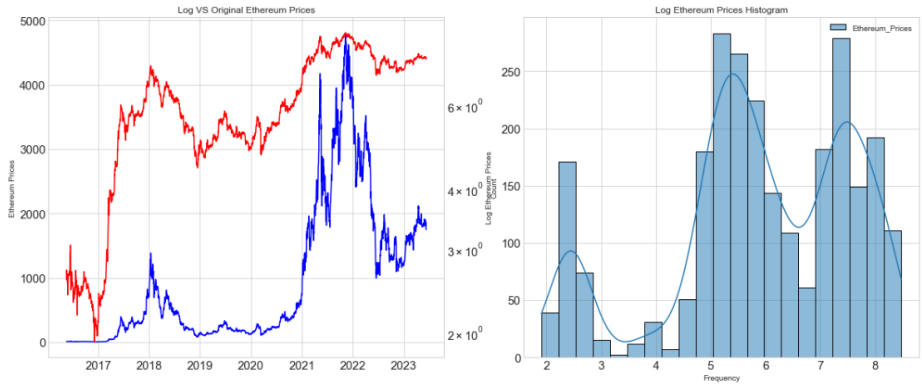
Anexo 3. Gráfico de bigodes da série *Coinbase Ethereum* (após imputação)



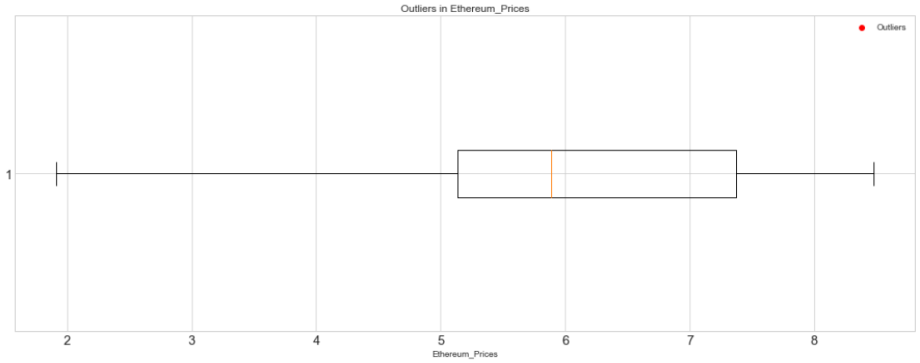
Anexo 4. Representação gráfica da série *Coinbase Ethereum* (após imputação)



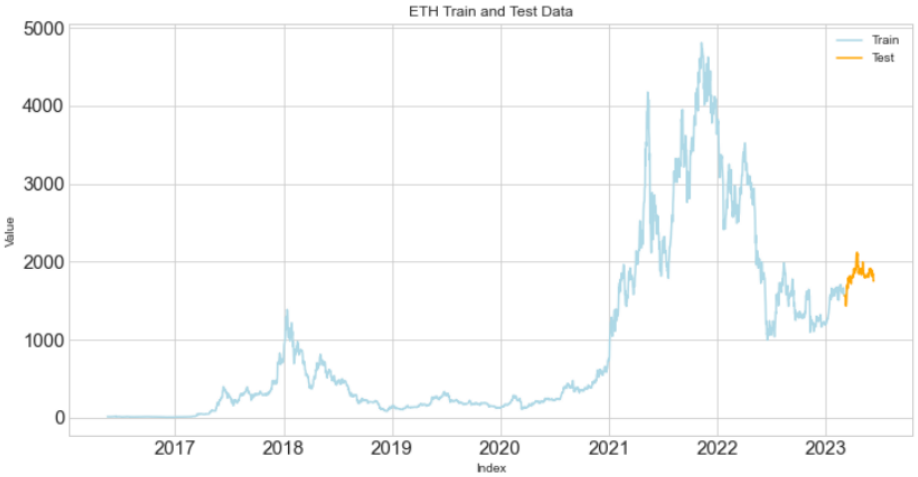
Anexo 5. Decomposição da série *Coinbase Ethereum* (após imputação)



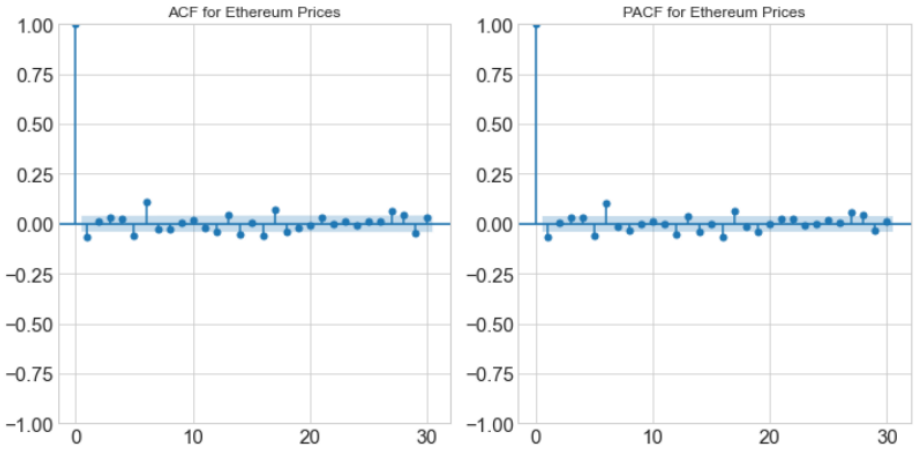
Anexo 6. Comparação da série temporal original com a transformada e histograma da mesma



Anexo 7. Gráfico de bigodes da série Coinbase Ethereum (após imputação e transformação logarítmica)

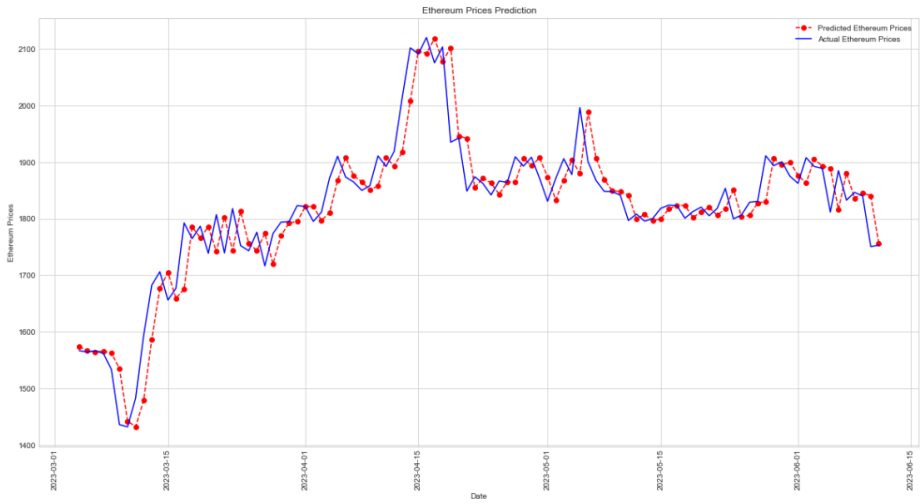


Anexo 8. Observação da divisão da série Coinbase Ethereum em conjunto de treino e teste

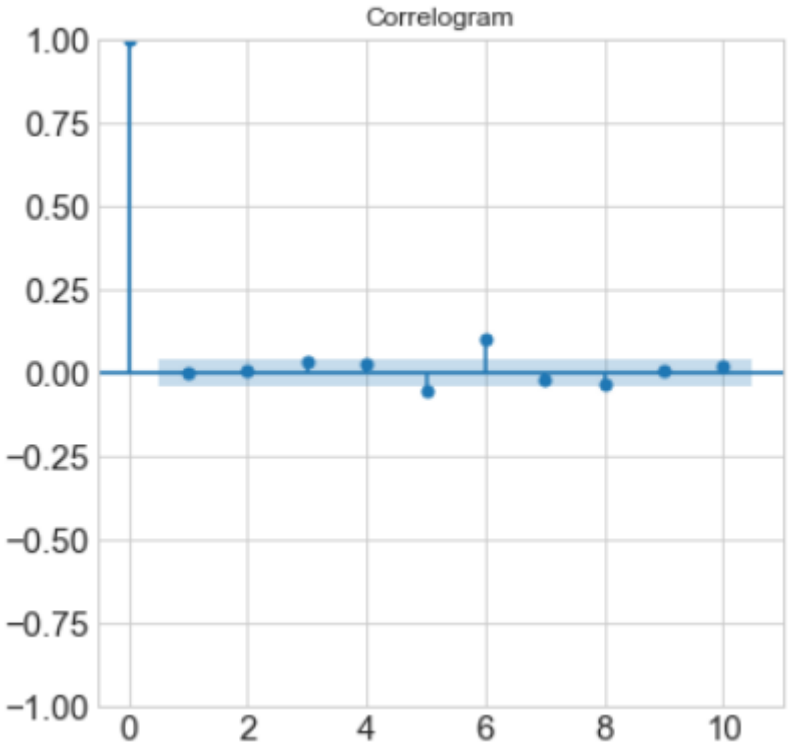


Anexo 9. Gráficos ACF e PACF da série *Coinbase Ethereum*

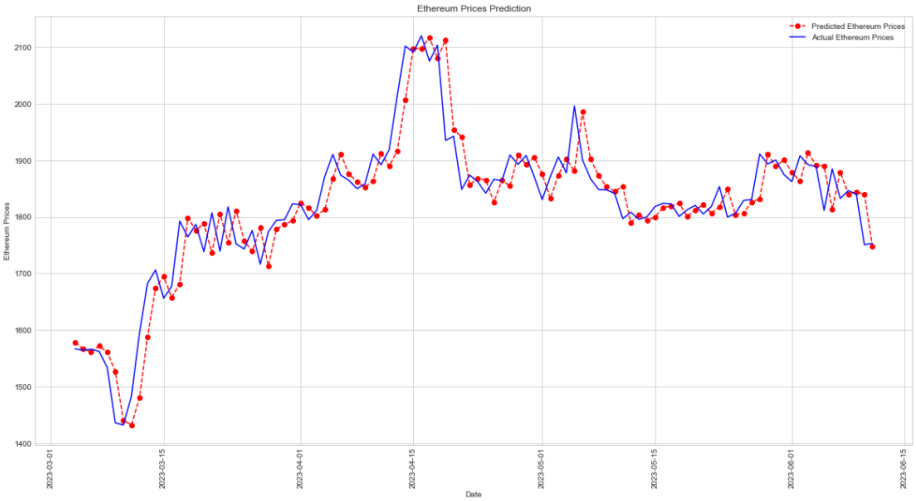
Anexo 10. Previsões para o conjunto de treino – SARIMA (1,1,0) (0,0,0) [0]



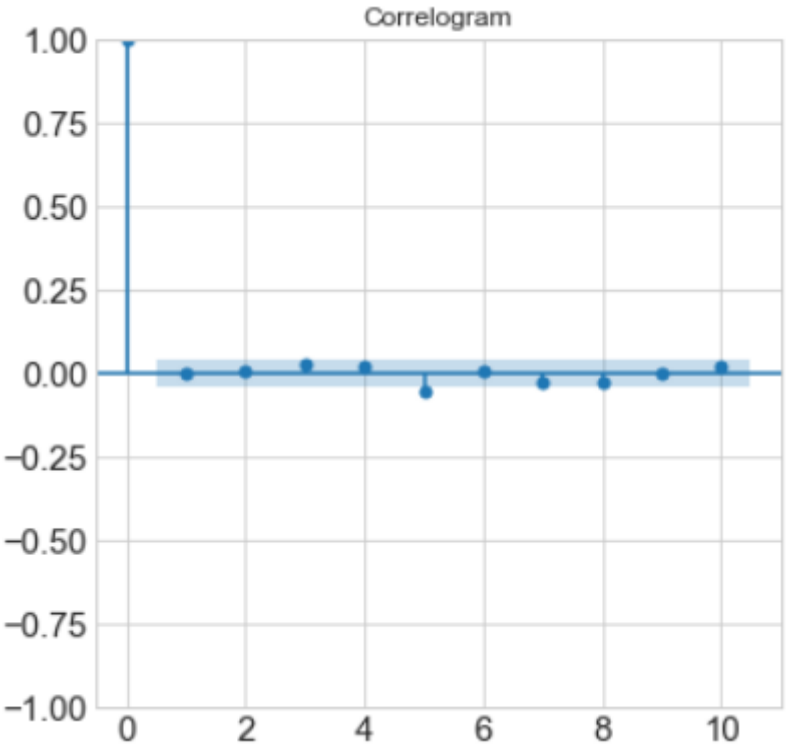
Anexo 11. Previsões para o conjunto de teste – SARIMA (1,1,0) (0,0,0) [0]



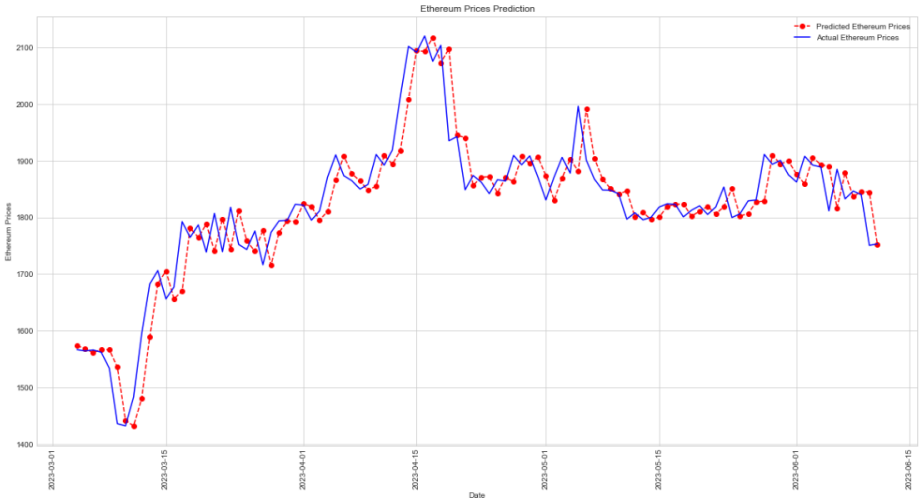
Anexo 12. *Correlogram* do modelo SARIMA (1,1,0) (0,0,0) [0] construído



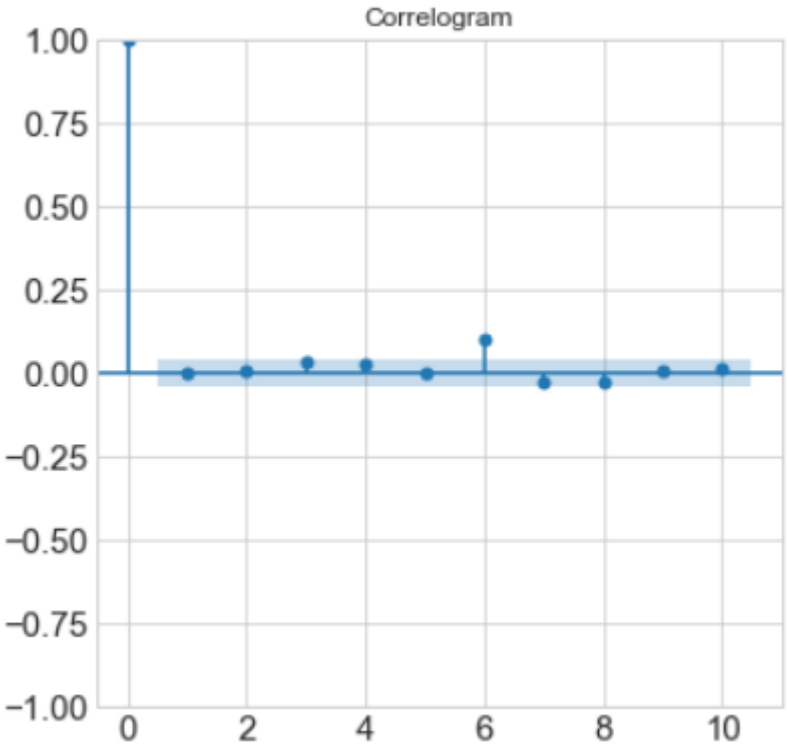
Anexo 13. Previsões para o conjunto de teste – SARIMA (1,1,0) (1,0,0) [6]



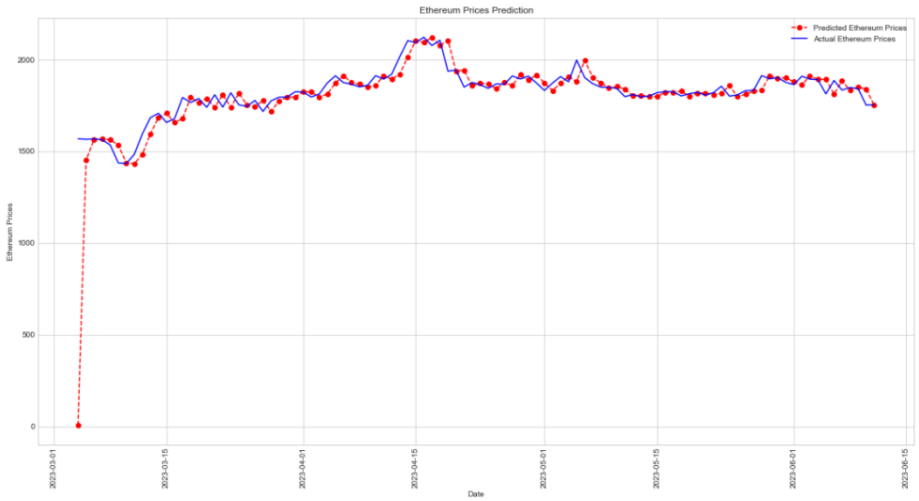
Anexo 14. *Correlogram* do modelo SARIMA (1,1,0) (1,0,0) [6] construído



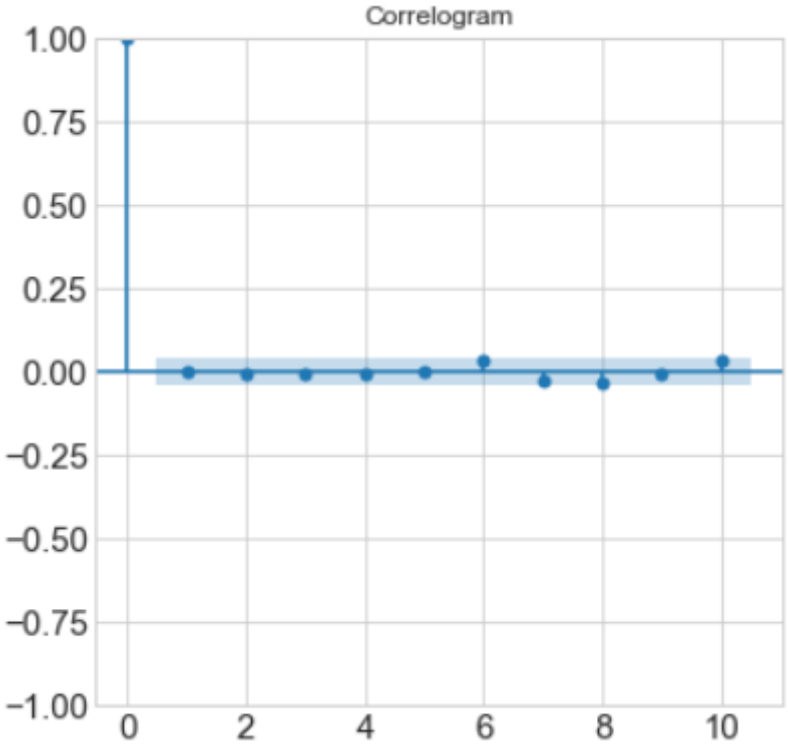
Anexo 15. Previsões para o conjunto de teste – SARIMA (1,1,0) (1,0,0) [5]



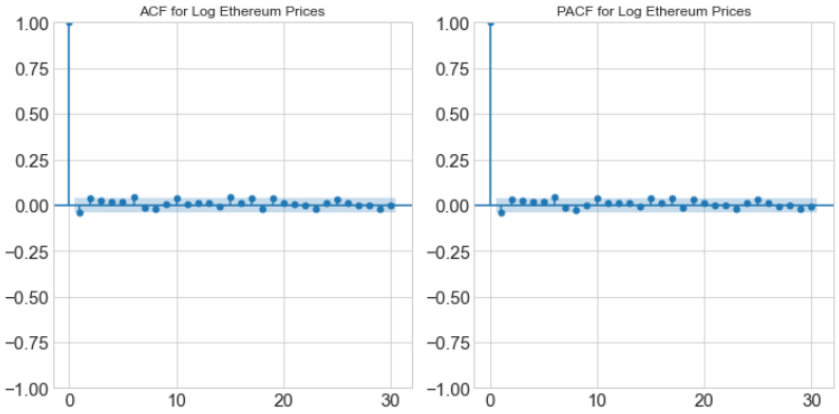
Anexo 16. *Correlogram* do modelo SARIMA (1,1,0) (1,0,0) [5] construído



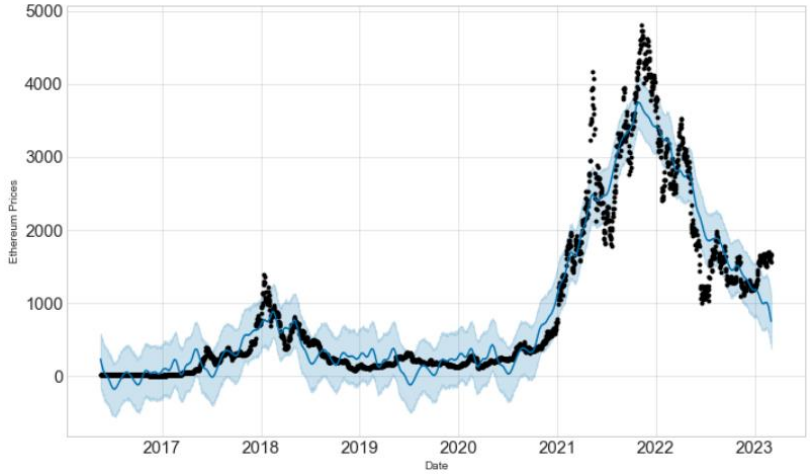
Anexo 17. Previsões para o conjunto de teste – SARIMA (2,1,1) (0,0,0) [0]



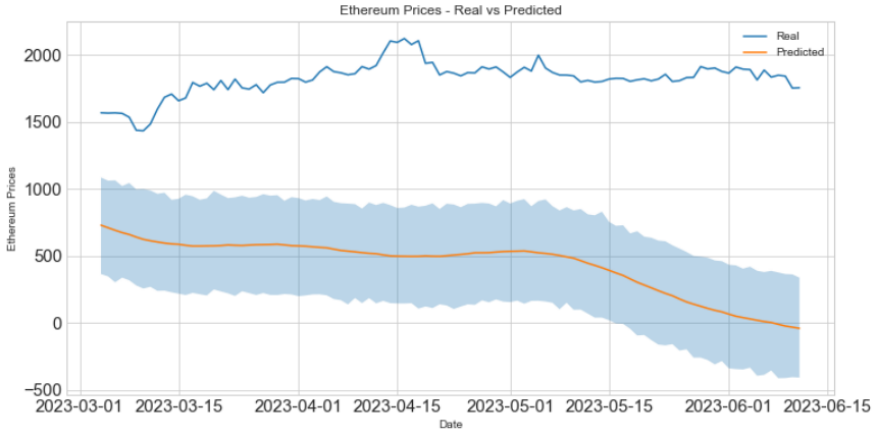
Anexo 18. *Correlogram* do modelo SARIMA (2,1,1) 01,0,0) [0] constr



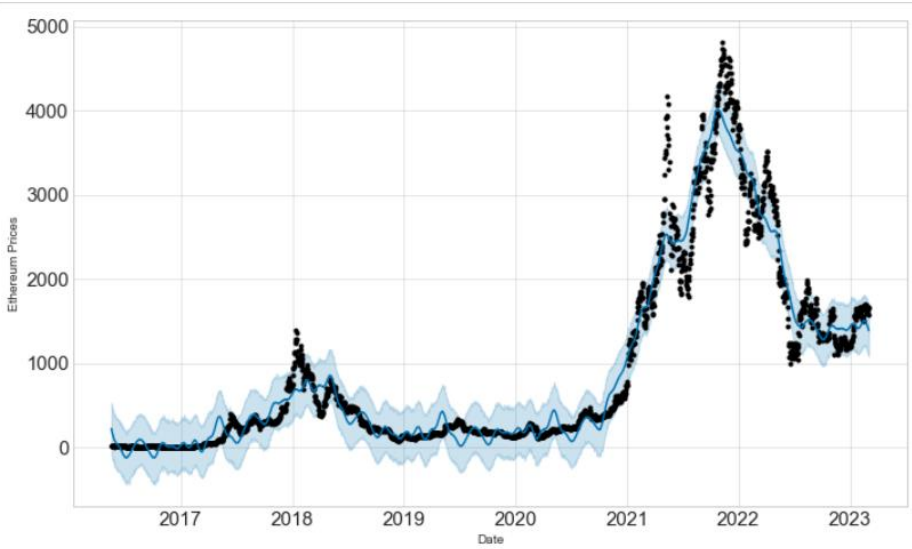
Anexo 19. Gráficos ACF e PACF da série *Coinbase Ethereum* transformada



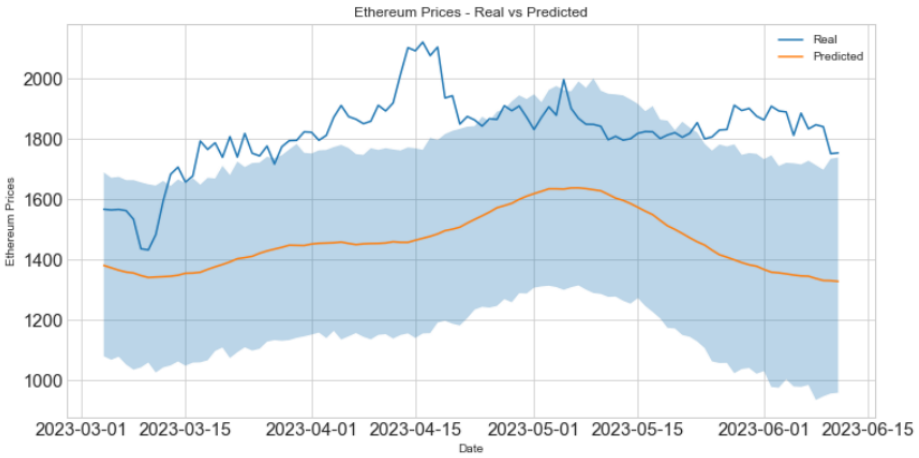
Anexo 20. Modelo *Facebook Prophet* – 1ª iteração, previsões para o conjunto de treino



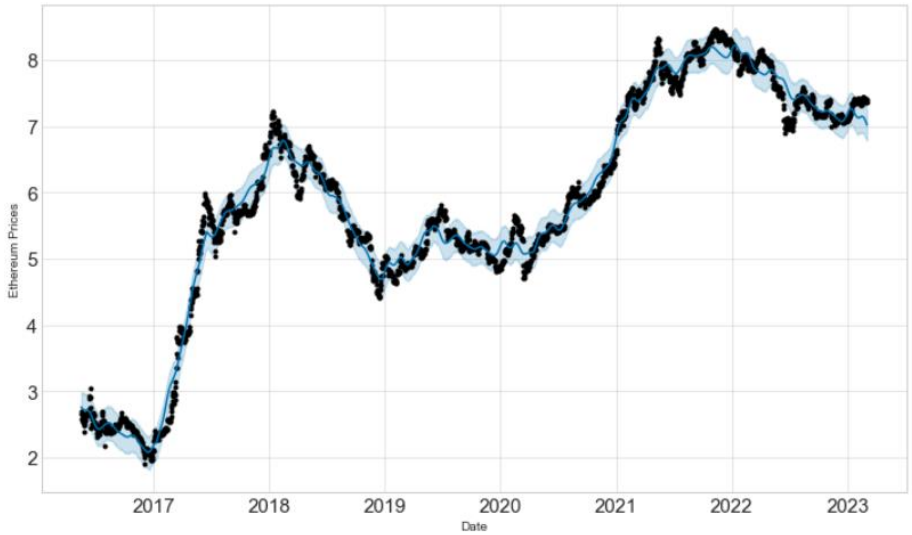
Anexo 21. Modelo *Facebook Prophet* – 1ª iteração, previsões para o conjunto de teste



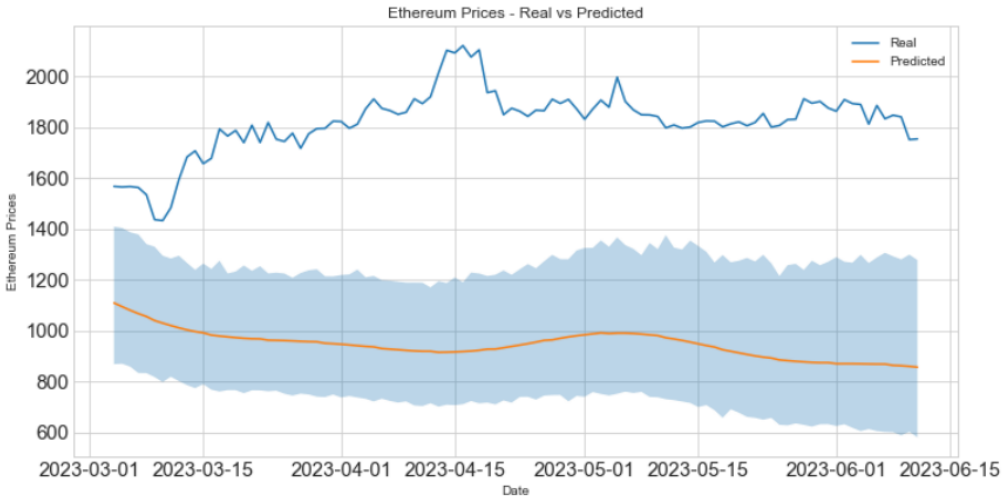
Anexo 22. Modelo *Facebook Prophet* – 2ª iteração, previsões para o conjunto de treino



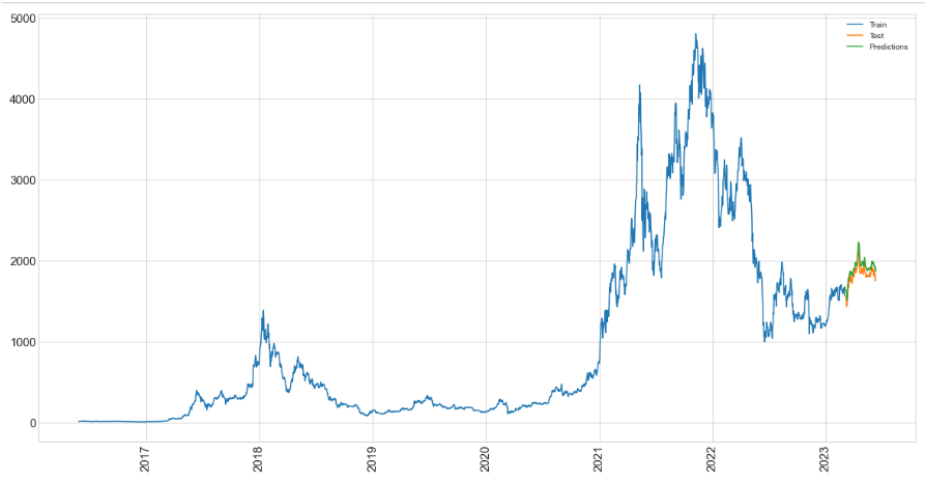
Anexo 23. Modelo *Facebook Prophet* – 2ª iteração, previsões para o conjunto de teste



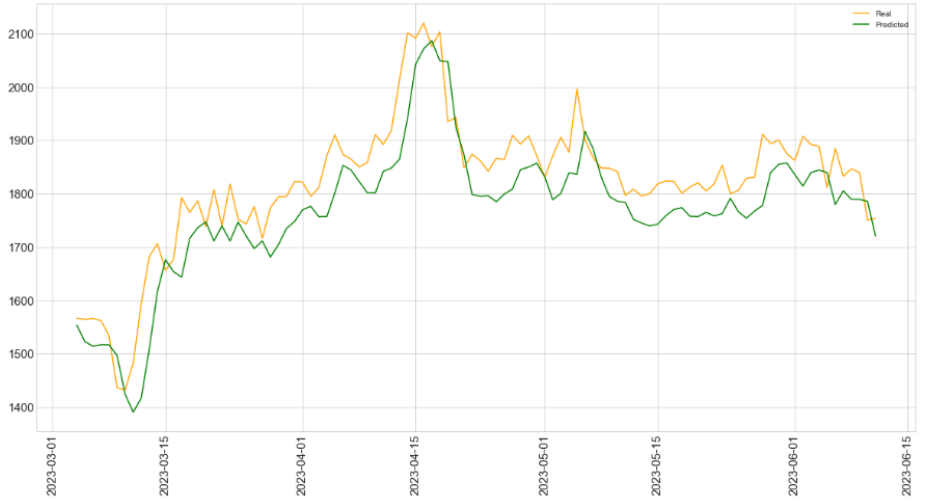
Anexo 24. Modelo *Facebook Prophet* – 3ª iteração, previsões para o conjunto de treino



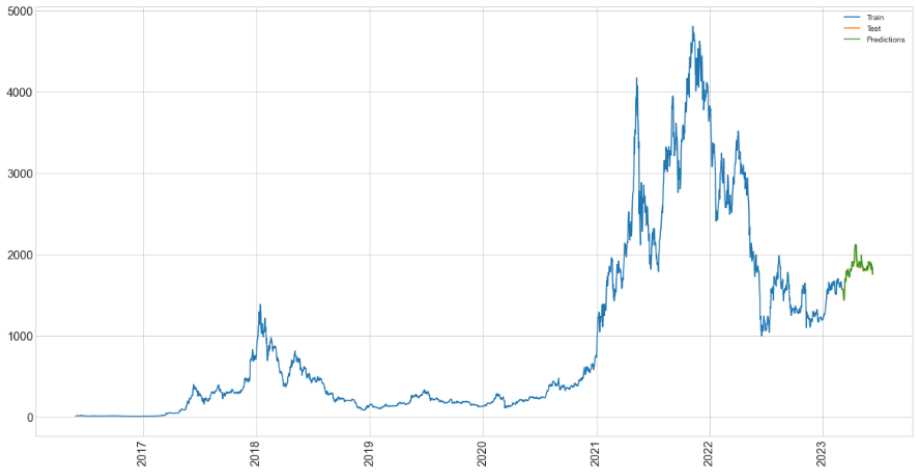
Anexo 25. Modelo *Facebook Prophet* – 3ª iteração, previsões para o conjunto de teste



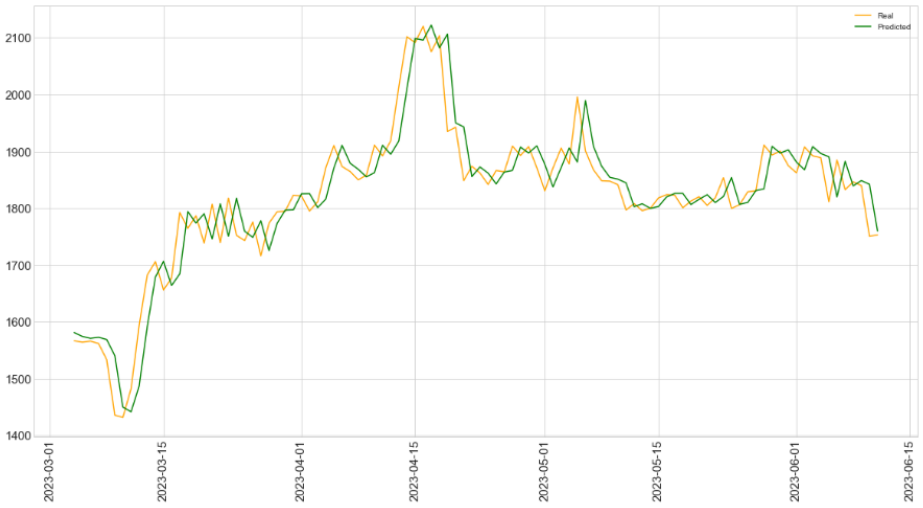
Anexo 26. Modelo *LSTM* – 1ª iteração, previsões para o conjunto de teste - geral



Anexo 27. Modelo *LSTM* – 1ª iteração, previsões para o conjunto de teste VS valores reais.



Anexo 28. Modelo *GRU* – 2ª iteração, previsões para o conjunto de teste - geral



Anexo 29. Modelo *GRU* – 2ª iteração, previsões para o conjunto de teste VS valores reais.