RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

PREDIÇÃO DE CRIPTOMOEDAS UTILIZANDO REDES NEURAIS ARTIFICIAIS RECORRENTES

CRYPTOCURRENCY PREDICTION USING RECURRENT ARTIFICIAL NEURAL NETWORKS

PREDICCIÓN DE CRIPTOMONEDAS UTILIZANDO REDES NEURONALES ARTIFICIALES RECURRENTES

Matheus Alves Coelho Ramazza[1], João Henrique Gião Borges[2], Fabiana Florian[3]

e463378
https://doi.org/10.47820/recima21.v4i6.3378

PUBLICADO: 06/2023

RESUMO

O presente estudo demonstra a criação de um *software* que possui a capacidade de prever a oscilação da criptomoeda *Bitcoin* através da aprendizagem profunda, por meio da rede neural recorrente do tipo Memória de Longo-Curto Prazo (LSTM), que manipula os valores de fechamento da criptomoeda como dados sequenciais temporais. O desenvolvimento do *software* foi baseado em programação *Python*, utilizando bibliotecas como *Pandas, TensorFlow e Numpy*, que são comumente utilizadas para visualização e análise de dados. O processo de aprendizagem do programa foi baseado nos valores da *Bitcoin* no período de janeiro de 2016 a janeiro de 2022. A partir da leitura dos dados, o *software* gerou um gráfico final de previsão, que demonstrou ser capaz de prever a oscilação de uma criptomoeda, apesar de alguma divergência do valor real e do valor previsto. É possível otimizar o *software* por meio do refinamento do número de testes realizados. Contudo, os resultados obtidos não podem ser confiados ou utilizados como ferramenta de investimento.

PALAVRAS-CHAVE: *Bitcoin.* Aprendizagem Profunda. Rede Neural Recorrente. Pandas. TensorFlow.

ABSTRACT

The present study demonstrates the creation of a software that has the ability to predict the oscillation of the Bitcoin cryptocurrency through deep learning and through the recurrent neural network of the Long-Short-Term Memory (LSTM) type, which manipulates the closing values of the cryptocurrency as temporal sequential data. The Software development was based on the Python programming language, using libraries such as Pandas, TensorFlow and Numpy, which are commonly used for data visualization and analysis. The program learning process was based on Bitcoin values in the period from January 2016 to January 2022. After reading the data, the software generated a final prediction graph, which proved to be able to predict the oscillation of a cryptocurrency, despite some divergence of the actual value and the predicted value. It is possible to optimize the software by refining the number of tests performed. However, the results obtained cannot be trusted or used as an investment tool.

KEYWORDS*: Bitcoin. Deep Learning. Recurrent Neural Network. Pandas. TensorFlow.*

RESUMEN

El presente estudio demuestra la creación de software que tiene la capacidad de predecir la oscilación de la criptomoneda Bitcoin a través del aprendizaje profundo, a través de la red neuronal recurrente del tipo Long-Term Memory (LSTM), que manipula los valores de cierre de la criptomoneda como datos secuenciales temporales. El desarrollo del software se basó en la programación Python, utilizando librerías como Pandas, TensorFlow y Numpy, que se utilizan comúnmente para la visualización y análisis de datos. El proceso de aprendizaje del programa se basó en los valores de Bitcoin en el período comprendido entre enero de 2016 y enero de 2022. A partir de la lectura de los datos, el software generó un gráfico de predicción final, que demostró ser capaz de predecir la oscilación de una criptomoneda, a pesar de cierta divergencia del valor real y el valor previsto. Puede optimizar el software

[1] Uniara - Universidade de Araraquara.
[2] Uniara - Universidade de Araraquara.
[3] Uniara - Universidade de Araraquara.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

refinando el número de pruebas realizadas. Sin embargo, los resultados obtenidos no pueden ser confiables o utilizados como una herramienta de inversión.

***PALABRAS CLAVE**: Bitcoin. Pandas. TensorFlow. Red Neuronal Recurrente. Aprendizaje Profundo.*

INTRODUÇÃO

A *Bitcoin* foi a primeira criptomoeda a ser criada, no início de 2008, por Satoshi Nakamoto, desse modo, possibilitando um novo método de investimento no mercado financeiro (NAKAMOTO, 2008). De acordo com Mattos; Abouchedid; Silva (2020), as criptomoedas são um tipo de moeda descentralizada, com o objetivo de retirar o controle da moeda estatal. As criptomoedas são transacionadas em um sistema descentralizado, sem a presença de intermediários financeiros, e sua validação é feita por um computador conectado à rede da moeda em questão.

Há no mercado financeiro várias criptomoedas, sendo *Bitcoin* (BTC), *Ethereum* (ETH), *Weth* (WETH) e *Binance* (BNB), as criptomoedas mais valorizadas atualmente. A popularidade das criptomoedas tem se ampliado de forma significativa desde a sua criação, aumentando o seu valor monetário consideravelmente.

Mesmo com o crescimento astronômico das criptomoedas nos últimos anos, ainda assim, não é um investimento seguro, por causa de sua volatilidade. O valor de uma criptomoeda oscila consideravelmente em um dia e ainda mais em meses, dificultando o estudo e a análise do mercado financeiro, e é quase impossível prever sua oscilação para um investimento seguro.

Este trabalho teve como objetivo geral, desenvolver um *software* para prever a oscilação da *Bitcoin* (BTC) no mercado financeiro, observando as altas e baixas e o volume transacionado da moeda em um determinado período. Desta maneira o *software* auxiliaria um investimento mais seguro e consciente sobre uma determinada criptomoeda.

O *software* desenvolvido utilizou uma rede neural artificial (RNA), que segundo Haykin (1998), uma rede neural é um processador distribuído composto de simples unidades de processamento, os quais armazenam conhecimento e disponibiliza-os para o uso futuro. Uma RNA assemelha-se com o cérebro, pois o conhecimento é adquirido pela rede através de aprendizado.

Para fazer a predição do mercado da criptomoeda foi utilizado um tipo específico de RNA, uma rede neural artificial recorrente (RNN), um tipo de RNA. De acordo com Koutník; Greff; Gomez; Schmidhuber (2014), RNN's são adequadas para lidar com problemas sequenciais como classificação de fala, predição e generalização.

A metodologia adotada neste trabalho é uma abordagem descritiva de cunho qualitativo-quantitativa e iniciou-se a partir de um referencial bibliográfico sobre redes neurais artificiais recorrentes, composto de livros, artigos, teses de mestrado e doutorado, e com o auxílio da internet. O levantamento de dados foi feito em bases de consulta como SCIELO (*A Scientific Electronic Library Online*), CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) e Google Acadêmico.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

REFERENCIAL TEÓRICO

Para descrever sobre o tema de redes neurais artificiais recorrentes, é preciso apresentar alguns conceitos sobre criptomoedas, inteligência artificial, redes neurais artificiais, LSTM e redes neurais recorrentes.

Criptomoedas e Conceitos Correlatos

O *Bitcoin*, desenvolvido por Satoshi Nakamoto em 2008, se destacou ao se propagar rapidamente não apenas entre membros e simpatizantes dos movimentos pró-criptografia, mas também entre os agentes dos mercados financeiros (MATTOS; ABOUCHEDID; SILVA, 2020).

A identidade real de Satoshi Nakamoto, mais de 15 anos após a criação da *Bitcoin*, ainda continua uma incógnita, pois o nome do criador é apenas um pseudônimo (SENA; DIAN, 2020).

Segundo Tredinnick (2019), a criptomoeda é como as outras moedas, não possuem um valor intrínseco, mas sim, o valor proveniente de transações feitas. Como uma consequência, as criptomoedas tendem a ser voláteis.

Ainda segundo Tredinnick (2019), as criptomoedas não são controladas por nenhuma autoridade central, como um banco nacional. Nenhuma organização ou agência consegue influenciar o valor da moeda digital ou emitir mais moedas no sistema, assim tornando-as em criptomoedas descentralizadas e independentes do Estado.

Inteligência Artificial

A inteligência artificial é uma área de estudo da ciência da computação que herdou várias técnicas, ideias e pontos de vista de outras disciplinas (RUSSEL; NORVIG, 2010). A palavra “inteligência” vem do latim *inter* (entre) e *legere* (escolher), portanto a inteligência é aquilo que permite o ser humano escolher entre uma coisa e outra, executar uma determinada tarefa eficientemente (FERNANDES, 2003).

Portanto, a inteligência artificial é um tipo de inteligência produzida pelo homem para que as máquinas tenham algum tipo de habilidade para simular a inteligência natural do ser humano. Contudo, o termo “inteligência artificial” é definido de inúmeras outras formas, baseando-se na racionalidade e em como os seres humanos pensam e agem, conforme pode ser verificado na figura 1.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 1 - Algumas definições de inteligência artificial, organizadas em quatro categorias

Pensando como um humano	**Pensando racionalmente**
"O novo e interessante esforço para fazer os computadores pensarem (...) *máquinas com mentes*, no sentido total e literal." (Haugeland, 1985) "[Automatização de] atividades que associamos ao pensamento humano, atividades como a tomada de decisões, a resolução de problemas, o aprendizado..." (Bellman, 1978)	"O estudo das faculdades mentais pelo uso de modelos computacionais." (Charniak e McDermott, 1985) "O estudo das computações que tornam possível perceber, raciocinar e agir." (Winston, 1992)
Agindo como seres humanos	**Agindo racionalmente**
"A arte de criar máquinas que executam funções que exigem inteligência quando executadas por pessoas." (Kurzweil, 1990) "O estudo de como os computadores podem fazer tarefas que hoje são melhor desempenhadas pelas pessoas." (Rich and Knight, 1991)	"Inteligência Computacional é o estudo do projeto de agentes inteligentes." (Poole *et al*., 1998) "AI... está relacionada a um desempenho inteligente de artefatos." (Nilsson, 1998)

Fonte: Russel; Norvig, (2010)

Aprendizado de Máquina

Aprendizado de máquina, que também é conhecido como *machine learning*, tem o objetivo de aprender automaticamente relacionamentos e padrões significativos a partir de exemplos e observações (BISHOP, 2006).

De acordo com Janiesch; Zschech; Heinrich (2021), o aprendizado de máquina é alcançado aplicando algoritmos que aprendem iterativamente com dados de treinamento de um problema específico, permitindo computadores encontrarem padrões ocultos e mais complexos, sem serem explicitamente programados para esta tarefa.

Ainda de acordo com Janiesch; Zschech; Heinrich (2021), o algoritmo de aprendizado de máquina, ao aprender com cálculos anteriores e extrair regularidades de bancos de dados, pode ajudar a tomar decisões confiáveis, por isso tais algoritmos são usados em áreas como detecção de fraude, processamento de linguagem natural e reconhecimento de voz e imagem.

Redes Neurais Artificiais

As Redes Neurais Artificiais (RNA) são técnicas computacionais do ramo da inteligência artificial, que usam como base a biologia e o estudo do cérebro para criar estruturas de processamento. Para Haykin (1998), uma rede neural é um processador distribuído e massivamente paralelo, composto por unidades de processamento simples que tem a propensão natural para armazenar conhecimento experiencial e disponibilizá-lo para uso. Isto se assemelha-se ao cérebro em dois aspectos:

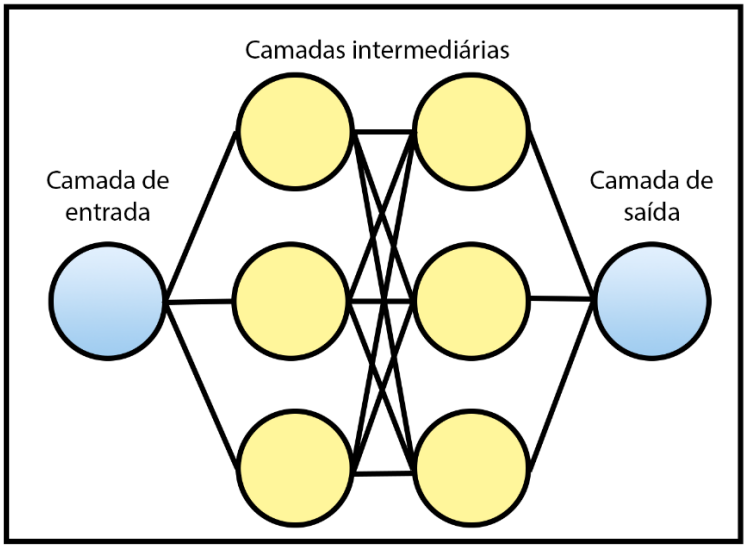
1- O conhecimento é adquirido pela rede a partir de seu ambiente, através de um processo de aprendizagem.
2- As forças de conexão interneurônio são conhecidas como pesos sinápticos e são usadas para armazenar o conhecimento adquirido.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Segundo Rosa (2011), outra razão para estudar modelos parecidos com o cérebro é seu paralelismo, pois para que o cérebro trabalhe o mais rápido possível, muitos neurônios devem trabalhar em paralelo. Entretanto, muitos programas de inteligência artificial possuem uma má performance, pois são simulados em sistema com um processador único e sem paralelismo.

A arquitetura de um RNA, pode possuir uma única camada de neurônios ou até mesmo diversas camadas, conforme pode ser visto na figura 2. A definição de um RNA é baseada nos seguintes parâmetros: os números de camadas da rede; números de neurônios por camada; e a topologia da rede (BRAGA; LUDERMIR; CARVALHO, 2000).

Figura 2 – Exemplo de um RNA de duas camadas



Fonte: Autor

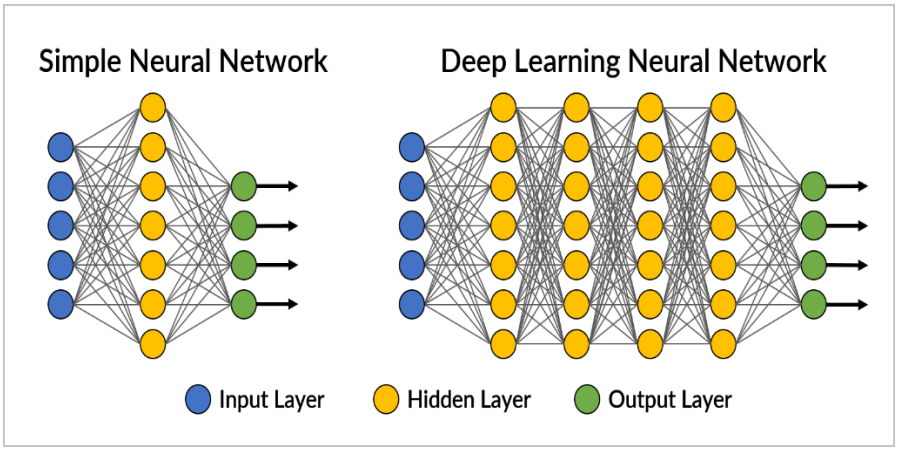
Aprendizagem Profunda

Quando falamos em redes neurais artificiais, normalmente consideramos a conexão entre dois ou mais neurônios, utilizando os dados de entrada e os pesos sinápticos para obter o que chamamos de dados de saída, por meio de uma dada função de ativação. Cada problema a ser resolvido envolve, potencialmente, uma rede neural diferente e pode exigir as chamadas camadas ocultas, através das quais pode-se fazer o envio de um dado tratado por seus pesos a um outro neurônio, refinando assim o possível resultado do processo.

Alguns problemas possuem complexidade elevada e podem exigir muitas dessas camadas ocultas para obter um dado de saída no neurônio da camada correspondente, e uma rede neural que usa muitas dessas camadas realizam o que se chama de *deep learning*, ou seja, aprendizado profundo, caracterizado por muitas camadas ocultas entre a camada de entrada e a camada de saída. Conforme a figura 3:

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 3 – Exemplo de aprendizagem profunda



Fonte: Autor

Naturalmente, *deep learning* demanda por mais recursos computacionais quando comparado com uma rede neural comum, uma vez que cada camada oculta é uma peça de *software* que precisa ser processada para oferecer algum resultado.

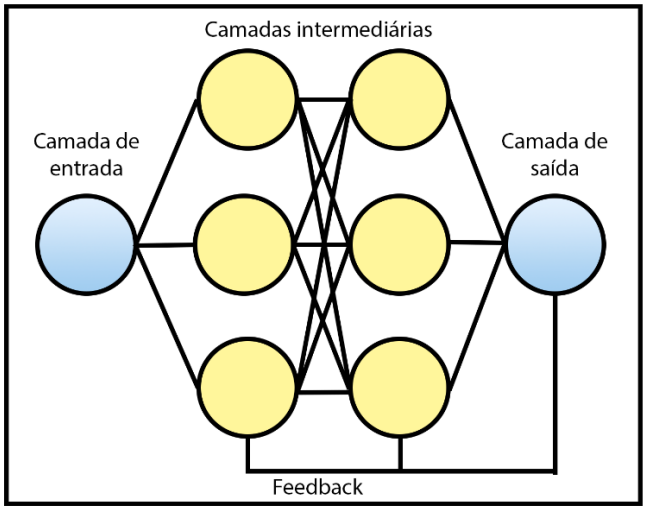
Redes Neurais Artificiais Recorrentes

Segundo Koutník; Greff; Gomez; Schmidhuber (2014), Redes Neurais Recorrentes são uma classe de modelos baseados em conexões, que possuem um estado interno ou memória de curto prazo devido a conexões recorrentes de *feedback,* conforme pode ser visto na figura 3, que as tornam adequadas para lidar com problemas sequenciais.

De acordo com Braga; Ludermir; Carvalho (2000), as redes neurais recorrentes (RNN) são caracterizadas por suas conexões de realimentação que proporcionam um comportamento dinâmico. As RNN's são classificadas em dois tipos, a padrão, em que a entrada de dados é sempre fixa e a baseada no tempo, que tanto a entrada quanto a saída de dados variam com o tempo. As RNN's baseadas no tempo são mais comumente utilizadas.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 4 – Exemplo de uma RNN



Fonte: Autor

Memória de Longo-Curto Prazo

Um simples RNN tem dificuldade em aprender dependências de longo termo nas sequências de entrada devido ao gradiente de desaparecimento. Este problema foi solucionado usando uma estrutura neural especializada em Memória de Longo-Curto Prazo (LSTM) que mantem um fluxo retrógrado constante (KOUTNÍK *et al.,* 2014).

Segundo Greff *et al.,* (2017), a ideia central por trás da arquitetura do LSTM é uma célula de memória, que mantém informações durante um determinado período, a mesma célula consegue regular a passagem de dados pela célula.

BIBLIOTECAS UTILIZADAS

Para facilitar o desenvolvimento de *softwares,* são utilizadas diversas bibliotecas, que são códigos desenvolvidos por terceiros e disponibilizados com livre acesso ou disponibilizados gratuitamente, com intuito de facilitar a execução de atividades de outros usuários. Para a criação de um agente inteligente em Python, algumas bibliotecas específicas vão ser utilizadas, com finalidade de simplificar e facilitar a criação do agente.

NumPy

NumPy é uma biblioteca fundamental para a computação científica em *Python*, pois fornece um objeto de matriz multidimensional e uma variedade de rotinas para operações rápidas em matrizes (NUMPY, 2023).

Segundo Harris *et al*., (2020), Numpy sustenta quase todas as bibliotecas *Python* que fazem computação científica ou numérica, incluindo *SciPy, Matplotlib, Pandas, Scikit-learn*. O *NumPy* é uma biblioteca de código aberto desenvolvida pela comunidade, que fornece um objeto de matriz multidimensional, junto com funções de reconhecimento de *arrays*. A matriz *NumPy* é uma estrutura de

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

dados que armazena e acessa eficientemente matrizes multidimensionais e permite uma ampla variedade de cálculos científicos.

Pandas

Pandas é uma biblioteca do *Python* que fornece estruturas de dados eficientes e flexíveis para trabalhar com dados rotulados ou relacionais. Ele visa ser a ferramenta ideal para a análise de dados e se esforça para se tornar a ferramenta de manipulação de dados de código aberto mais poderosa de todas as linguagens. A biblioteca é adequada para vários tipos de dados, incluindo dados tabulares, dados de séries temporais, dados de matrizes e quaisquer outros conjuntos de dados observacionais ou estatísticos. O *Pandas* é construído sobre o *Numpy* e se integra perfeitamente com outras bibliotecas de terceiros, que são comumente usadas em ambientes de computação científica (PANDAS, 2023).

TensorFlow

O *TensorFlow* é uma plataforma de código aberto voltado ao aprendizado de máquina. Ele tem um ecossistema abrangente e flexível de ferramentas, bibliotecas e recursos da comunidade que permite a fácil criação e implementação de aplicativos alimentados por aprendizado de máquina (TENSORFLOW, 2023).

Segundo Abadi *et al.,* (2016), o *TensorFlow* é uma interface para expressão de algoritmos de aprendizado máquina e sua implementação para execução desses algoritmos. O *TensorFlow* é flexível e pode ser usado para expressar uma ampla variedade de algoritmos, incluindo os de treinamento e inferência para modelos de redes neurais profundas, e tem sido utilizado para a condução de pesquisa e implementação de sistemas de aprendizado de máquina em inúmeras áreas da ciência da computação.

Scikit-learn

Scikit-learn é uma biblioteca *Python* que integra uma ampla gama de algoritmos de aprendizado de máquina para problemas supervisionados e não supervisionados de média escala. Esta biblioteca se concentra em levar o aprendizado de máquina para não especialistas, usando uma linguagem de alto nível de uso geral. A ênfase é colocada na facilidade de uso, desempenho, documentação, consistência da API e dependências mínimas (PEDREGOSA *et al*., 2011).

Matplotlib

Matplotlib é uma biblioteca para criar visualizações estáticas, animadas e interativas em *Python.* O *Matplotlib* produz figuras de qualidade de publicação em uma variedade de formatos impressos e ambientes interativos em todas as plataformas (MATPLOTLIB, 2023).

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Yfinance

Yfinance é uma ferramenta de código aberto que usa as APIs publicamente disponíveis do *Yahoo* e destina-se a fins educacionais e de pesquisa. O *Yfinance* oferece uma maneira encadeada para baixar dados de mercado do *Yahoo*! Finanças (YFINANCE, 2023).

MÉTODO

Esta pesquisa se baseou na documentação citada na revisão de literatura para criar um agente inteligente, com o objetivo de prever a oscilação da *Bitcoin* (BTC). A RNN do tipo LSTM utilizada no *software*, manipula os valores de fechamento da criptomoeda em questão, como dados sequenciais temporais.

O *software* foi desenvolvido na linguagem de programação *Python*, utilizando bibliotecas adicionais para auxiliar no projeto, como *Pandas, Tensorflow, Numpy*, entre outras. O desenvolvimento foi feito no laboratório da Universidade de Araraquara (UNIARA), onde foram utilizadas as ferramentas do laboratório para a analisar e testar o *software.*

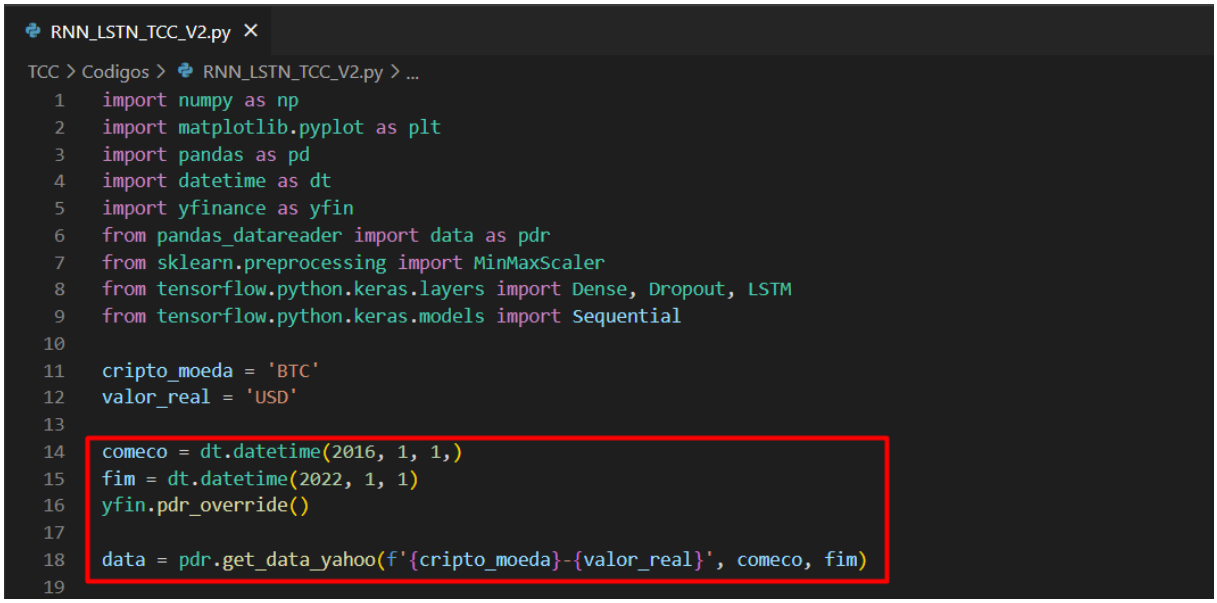
Na criação do *software,* primeiramente, várias bibliotecas *Python* são importadas, como por exemplo o *Numpy, Matplotlib, Yfinance* e *Pandas*. Essas bibliotecas são comumente utilizadas na computação científica, para visualização e análise de dados, acesso a dados financeiros, entre outras funções.

Em seguida, duas variáveis são definidas: “cripto_moeda” e “valor_real”, essas variáveis representam a criptomoeda *(Bitcoin)* e a moeda na qual seu preço é medido (dólares americanos). Logo após, a biblioteca datetime é utilizada para definir mais duas variáveis: começo e fim, essas variáveis representam as datas de início e término dos dados históricos que serão usados para treinar a rede LSTM.

Com essas variáveis definidas, o *software* busca os dados históricos da *Bitcoin* e do dólar, utilizando a função “*pdr.get_data_yahoo()”.* Os dados são obtidos a partir do *Yahoo Finances*, entre a data de início e a data de término, que são definidas para 1º de janeiro de 2016 e 1º de janeiro de 2022, respectivamente. Conforme a figura 5:

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 5 – Obtenção dos dados históricos



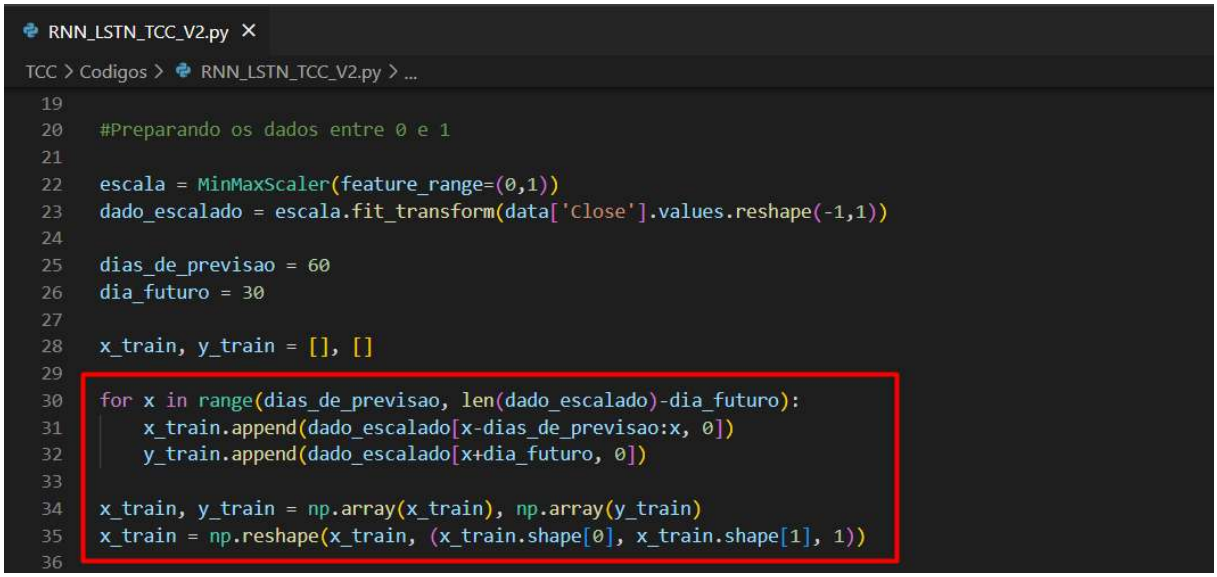
Fonte: Autor

A próxima etapa é utilizar a função "*MinMaxScaler()"* da biblioteca *Scikit-learn,* isso é feito para transformar os dados históricos, normalizá-los. Esta é uma etapa comum de pré-processamento no aprendizado de máquina, pois ajuda aprimorar a sua performance. Em seguida, a função *"fit_transform()"* é utilizada para aplicar o dimensionamento entre 0 e 1, aos dados de preços "*Close*" da criptomoeda. Os novos dados de preços "*Close*" escalados são armazenados na variável "dado_escalado", como um *numpy array*.

Após escalar os dados, é preciso prepará-los para treinamento do modelo. Para isso, são criadas duas variáveis: "dias_de_previsao" e "dia_futuro". A primeira variável recebe o valor de 60 e a segunda recebe o valor de 30, que significa que o *software* utilizará os dados dos últimos 60 dias para prever o preço da criptomoeda 30 dias no futuro. Os dados são divididos nos *arrays* "x_*train*" e "*y_train",* onde "*x_train* contém os dados dos 60 dias anteriores e *"y_train"* contém o preço do 30º dia depois disso. Conforme a figura 6:

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 6 – Preparação dos dados



Fonte: Autor

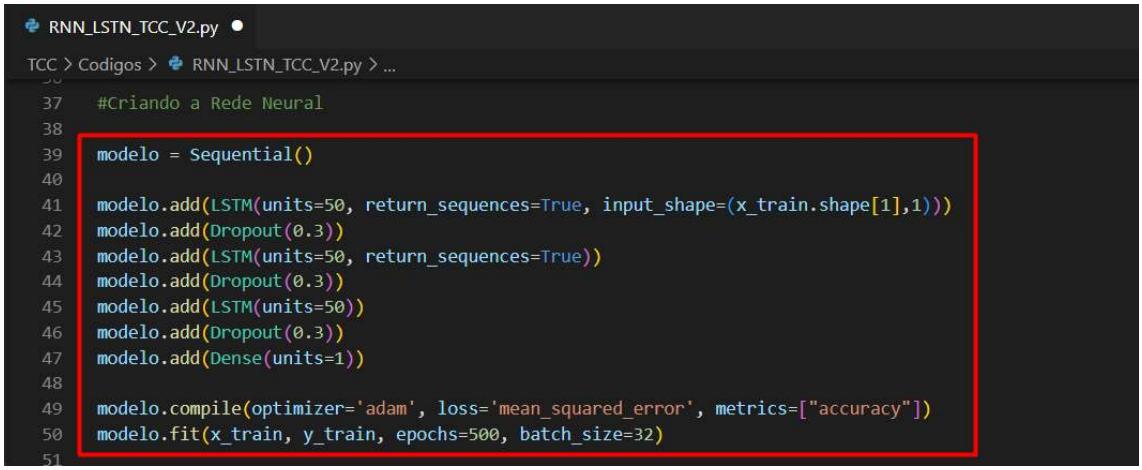
Depois de preparar os dados de treinamento, o *software* cria o modelo LSTM usando a classe Sequential da biblioteca do *TensorFlow*. O modelo é criado com a adição de camadas, começando com uma camada LSTM que possui 50 unidades, uma camada de *Dropout* para evitar o *overfitting* e, em seguida, esse processo é repetido mais duas vezes. A camada final é uma camada densa com 1 unidade, para fazer a previsão final.

Após a criação do modelo, é preciso compilar o modelo utilizando a função "*compile()"* da biblioteca *Tensorflow.* O modelo é então copilado utilizando o otimizador "*adam*", que é um algoritmo que atualiza os pesos da rede neural durante o treinamento, e também é utilizado a perda "mean_squared_error", esse argumento tem a função de medir quão bem o modelo se ajusta aos dados de treinamento.

A próxima etapa é treinar o modelo, durante este processo, o modelo fará previsões sobre os dados de entrada e comparará a saída com a saída real. Para fazer o treinamento é utilizado a função "fit()" da biblioteca da *Scikit-learn*, que recebe "*x_train*" e "*y_train*" como dados de entrada e saída, que são utilizados para treinar o modelo. Além do argumento de entrada e saída, é utilizado mais dois argumentos: *epochs* e *batch_size*. O argumento *epochs* especifica o número de vezes que o processo de treinamento iterará pelo conjunto de dados, neste caso, o modelo será iterado 500 vezes. Já o argumento *batch_size* especifica o número de amostras por atualização de gradiente, separando os dados de treinamento em lotes e o modelo é atualizado após cada lote, neste caso, o modelo é atualizado após 32 amostras. Conforme a figura 7:

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 7 – Criação da rede neural



Fonte: Autor

Após o modelo ser criado e treinado, ele precisa ser testado. Para isso, é novamente utilizado a função "*pdr.get_data_yahoo()"* para obter os dados de treinamento, mas desta vez, só são obtidos os dados de 1º de janeiro de 2021 até o dia atual. Os dados históricos de preços dos períodos de treinamento e teste são armazenados em uma variável chamada "data_total". Em seguida os dados para o modelo treinado são preparados, selecionando "n" dias do *dataframe*, onde "n" é igual a variável "dias_de_previsão" (ou seja, o número de dias usados para previsão).

Os dados precisam ser normalizados, o que é necessário para o modelo funcionar corretamente e para isso é utilizado novamente a função "*MinMaxScaler()"* da biblioteca *Scikit-learn.* Em seguida, um loop é utilizado para preencher a lista "x_teste" com os dados para o teste e os dados são convertidos em uma matriz 3D, utilizando a função "*reshape()"* da biblioteca *NumPy.* Finalmente, os preços previstos são obtidos usando a função "*predict()"* do modelo treinado, e o dimensionamento é revertido usando a função "*inverse_transform()"* da biblioteca do *Scikit-learn*. Conforme a figura 8:

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 8 – Testando o modelo

```python
#Testando o modelo

teste_comeco = dt.datetime(2021,1 ,1)
teste_fim =dt.datetime.now()

teste_data = pdr.get_data_yahoo(f'{cripto_moeda}-{valor_real}', teste_comeco, teste_fim)

preco_real = teste_data['Close'].values

data_total = pd.concat((data['Close'], teste_data['Close']), axis=0)

modelo_input = data_total[len(data_total) - len(teste_data) - dias_de_previsao:].values
modelo_input = modelo_input.reshape(-1,1)
modelo_input = escala.fit_transform(modelo_input)

x_teste = []

for x in range(dias_de_previsao, len(modelo_input)):
    x_teste.append(modelo_input[x-dias_de_previsao:x, 0])

x_teste = np.array(x_teste)
x_teste = np.reshape(x_teste, (x_teste.shape[0], x_teste.shape[1], 1))

preco_previsto = modelo.predict(x_teste)
preco_previsto = escala.inverse_transform(preco_previsto)
```

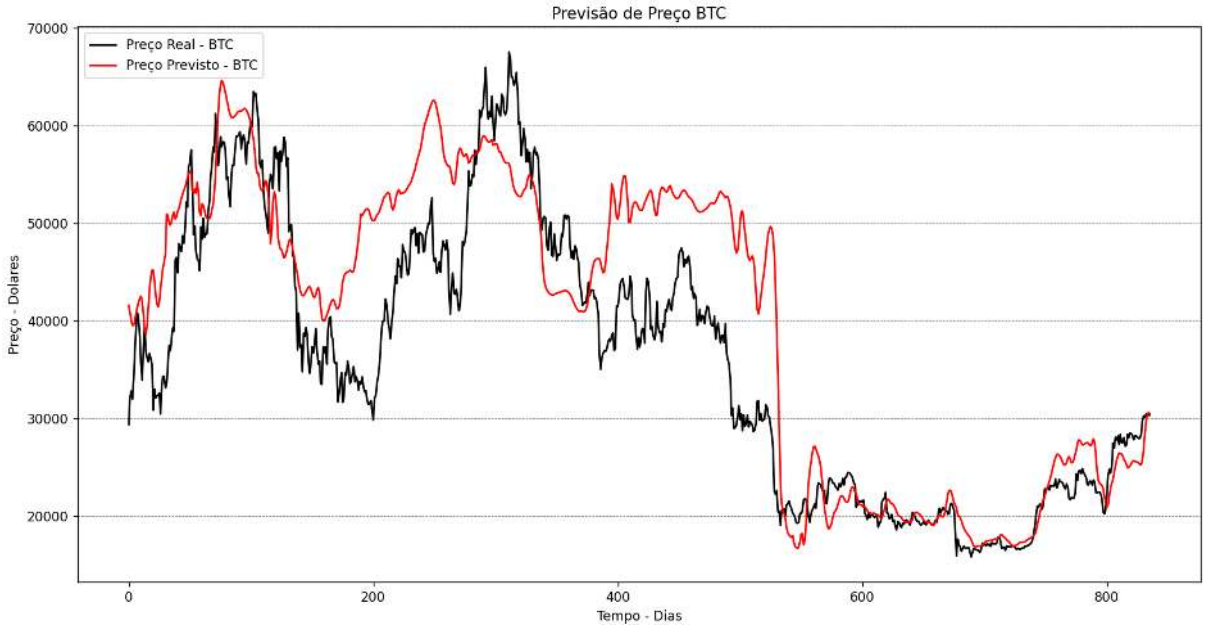
Fonte: Autor

Por último, é utilizada a biblioteca do *Matplotlib* para mostrar os dados obtidos após o treinamento e o teste do modelo. O preço real e o preço previsto são exibidos em um gráfico para uma melhor análise dos resultados.

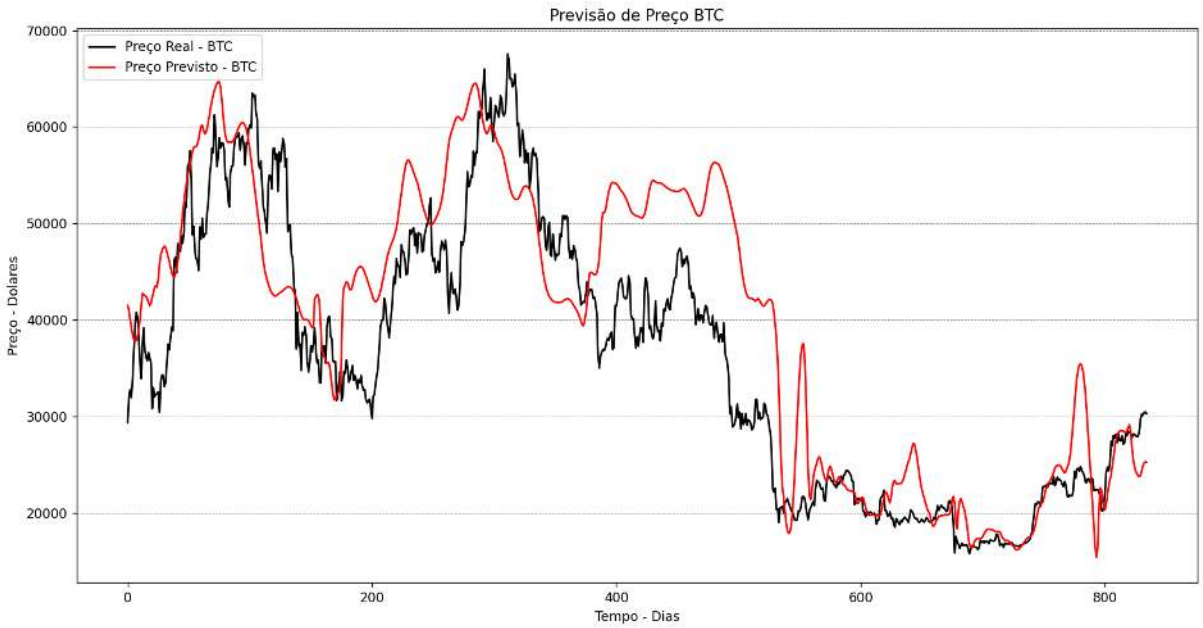
RESULTADOS

Com o gráfico gerado no final do *software*, é possível analisar os seus resultados, onde o valor real da *Bitcoin* é representado pela linha preta e o valor previsto pelo *software* é representado pela linha vermelha. Conforme as figuras 9 e 10:

Figura 9 e 10 – Gráficos gerados



RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218



Fonte: Autor

É possível observar que a previsão da criptomoeda acompanha a variação real da *Bitcoin*, mas com dois gráficos gerados, se é capaz de notar que o *software* possui uma baixa precisão, pois toda vez que ele é executado, são obtidos resultados diferentes. Mesmo tendo uma baixa precisão e sempre gerar resultados diferentes, o *software* consegue prever a oscilação da criptomoeda.

Apesar de seus bons resultados, o *software* não é perfeito, pois há momentos que a divergência entre o valor real da *Bitcoin* e o valor previsto é muito grande. Para obter resultados melhores e mais precisos, é possível modificar alguns parâmetros do *software*, como a quantidades de camadas do modelo, a quantidade de *epochs,* o *batch_size* etc.

CONCLUSÃO

O presente trabalho teve como objetivo desenvolver um *software* que fosse capaz de prever a oscilação da criptomoeda *Bitcoin*. Com cunho bibliográfico das fontes de pesquisa, foi possível alcançar parcialmente o objetivo proposto. Em seguida foi realizada uma descrição detalhada do projeto, com as bibliotecas utilizadas, como *Tensorflow* e *Pandas*.

É possível concluir que um *software* é capaz de prever a oscilação de uma criptomoeda, mas os resultados obtidos não podem ser confiados ou utilizados como uma ferramenta de investimento. Porque para gerar boas previsões é preciso fazer inúmeros testes para obter um software optimizado e ainda assim, fatores externos que o *software* não consegue prever podem influenciar uma criptomoeda.

REFERÊNCIAS

ABADI, M.; AGARWAL, A.; BARHAM, P. *et al*. **TensorFlow**: Large-Scale Machine Learning on Heterogeneous Distributed Systens, Estados Unidos: [s. n.], 2016.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

BISHOP, C. M. **Neural Networks for Pattern Recognition**. Oxônia: Oxford University Press, 1995, 504p.

BISHOP, C. M. **Pattern recognition and machine learning**. Nova Iorque: Springer, 2006. 738 p.

BRAGA, A.; LUDERMIR, T.; CARVALHO, A. **Redes Neurais Artificiais**: Teoria e aplicações. Rio de Janeiro: LTC, 2000. 262 p.

FERNANDES, A. **Inteligência artificial**: noções gerais. Florianópolis: Visual Books, 2003. 160p.

GREFF, K.; SRIVASTAVA, R. K.; KOUNTNÍK, J. *et al*. LSTM: A Search Space Odyssey. Institute of Electrical and Electronics Engineers. **Piscataway**, v. 28, n. 10, p. 2222-2232, out. 2017.

HARRIS, C.; MILLMAN, K.; WALT, S. *et al*. Array programming with NumPy. **Nature**, v. 585, p 357-362, set. 2020. DOI: https://doi.org/10.1038/s41586-020-2649-2

HAYKIN, S. **Neural Networks**: A Comprehensive Foundation. Michigan: Pearson, 1998. 842p.

JANIESCH, C.; ZSCHECH, P.; HEINRICH, K. Machine Learning and Deep Learning. **Electronic Markets**, Alemanha, v. 31, p 685-695, abr. 2021. DOI: https://doi.org/10.1007/s12525-021-00475-2

KARPATHY, A. **The Unreasonable Effectiveness of Recurrent Neural Networks**. [*S. l.: s. n.*], 2015. Disponível em: https://karpathy.github.io/2015/05/21/rnn-effectiveness/ Acesso em: 20 mar. 2022.

KOUTNÍK, J.; GREFF, K.; GOMEZ, F.; SCHMIDHUBER, J. A Clockwork RNN. Proceedings of the 31st International Conference on Machine Learning. **PMLR**, v. 32, n. 2, p. 1863-1871, 2014. DOI: https://doi.org/10.48550/arXiv.1402.3511

MATPLOTLIB. 2023. Disponível em: https://github.com/matplotlib/matplotlib Acesso em: 10 mar. 2023

MATTOS, O. B.; ABOUCHEDID, S.; SILVA, L. A. As criptomoedas e os novos desafios ao sistema monetário: uma abordagem pós-keynesiana. **Economia e Sociedade**, Campinas, v. 29, n. 3, p. 761-778, dez. 2020. DOI: https://doi.org/10.1590/1982-3533.2020v29n3art04

NAKAMOTO, Satoshi. **Bitcoin**: A Peer-to-Peer Electronic Cash System. [*S. l.: s. n.*], 2008. Disponível em: https://bitcoin.org/bitcoin.pdf Acesso em: 10 jul. 2022.

NUMPY. **Numpy Documentation**. [S. l.]: Numpy, 2023. Disponível em: https://numpy.org/doc/stable/ Acesso em: 10 mar. 2023

PANDAS. **Packge Overview**. [S. l.]: Pandas, 2023. Disponível em: https://pandas.pydata.org/docs/getting_started/overview.html Acesso em: 10 mar. 2023

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A. *et al.* Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, Estados Unidos, v. 12, p 2825-2830, 2011.

PHI, M. **Illustrated Guide to Recurrent Neural Networks**. [*S. l.: s. n.*], 2018. Disponível em: https://towardsdatascience.com/illustrated-guide-to-recurrent-neural-networks-79e5eb8049c9 Acesso em: 9 abr. 2022.

ROSA, J. **Fundamentos da Inteligência Artificial**. Rio de Janeiro: LTC, 2011. 228 p.

RUSSEL, S.; NORVIG, P. **Inteligência Artificial**. Rio de Janeiro: Elsevier, 2010. 1136 p.

SENA, L.; DIAN, M. Criptomoeda: Como obtê-la através da mineração. **Revista Interface Tecnológica**, Taquaritinga, v. 17, n. 2, p. 364-375, dez. 2020. DOI: https://doi.org/10.31510/infa.v17i2.1053

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

SUTSKEVER, I. **Training Recurrent Neural Networks**. 2013. 101f. Dissertação (Doutorado em Ciência da Computação) – Universidade de Toronto, Toronto, 2013.

TENSORFLOW, 2023. Disponível em: https://github.com/tensorflow/tensorflow Acesso em: 10 mar. 2023

TREDINNICK, L. Cryptocurrencies and the Blockchain. **Business Information Review**, Reino Unido, v. 36, p 39-44, mar. 2019. DOI: https://doi.org/10.1177/0266382119836314

YFINANCE, 2023. Disponível em: https://github.com/ranaroussi/yfinance Acesso em: 10 mar. 2023.