UNIVERSIDADE FEDERAL DO PAMPA
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

THAMIRES SAMPAIO PONTES DE MATOS

USO DE ALGORITMOS PARA PREDIÇÃO DE SÉRIES TEMPORAIS: UM ESTUDO DAS PLATAFORMAS *FACEBOOK'S PROPHET* E *AMAZON DEEPAR*

2022

THAMIRES SAMPAIO PONTES DE MATOS

USO DE ALGORITMOS PARA PREDIÇÃO DE SÉRIES TEMPORAIS: UM ESTUDO DAS PLATAFORMAS *FACEBOOK'S PROPHET* E *AMAZON DEEPAR*

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Engenharia de Computação como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Gerson Alberto Leiria Nunes
Coorientador: Fábio Luis Livi Ramos

2022

Ficha catalográfica elaborada automaticamente com os dados fornecidos
pelo(a) autor(a) através do Módulo de Biblioteca do
Sistema GURI (Gestão Unificada de Recursos Institucionais) .

M192u Matos, Thamires Sampaio Pontes de
Uso de algoritmos para predição de séries temporais: um estudo das plataformas Facebook's Prophet e Amazon DeepAR / Thamires Sampaio Pontes de Matos.
83 p.

Trabalho de Conclusão de Curso(Graduação)-- Universidade Federal do Pampa, ENGENHARIA DE COMPUTAÇÃO, 2022.
"Orientação: Gerson Alberto Leiria Nunes".

1. Inteligência Artificial. 2. Aprendizado de Máquina. 3. Séries Temporais. 4. Facebook's Prophet. 5. Amazon DeepAR. I. Título.

SERVIÇO PÚBLICO FEDERAL

MINISTÉRIO DA EDUCAÇÃO

Universidade Federal do Pampa

THAMIRES SAMPAIO PONTES DE MATOS

USO DE ALGORITMOS PARA PREDIÇÃO DE SÉRIES TEMPORAIS: UM ESTUDO DAS PLATAFORMAS FACEBOOK'S PROPHET E AMAZON DEEPAR

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Engenharia de Computação da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Trabalho de Conclusão de Curso defendido e aprovado em: 12 de Agosto de 2022.

Banca examinadora:

_____

Prof. Dr. Gerson Alberto Leiria Nunes
Orientador
UNIPAMPA

_____

Prof. Erico Marcelo Hoff do Amaral
UNIPAMPA

_____

Prof. Prof. Dr. Milton Roberto Heinen
UNIPAMPA

Assinado eletronicamente por **GERSON ALBERTO LEIRIA NUNES**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 19/08/2022, às 18:04, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **MILTON ROBERTO HEINEN**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 19/08/2022, às 19:26, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **ERICO MARCELO HOFF DO AMARAL**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 20/08/2022, às 01:21, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0903629** e o código CRC **7D928F00**.

Referência: Processo nº 23100.017425/2022-49 SEI nº 0903629

Dedico este trabalho ao meu pai Ítalo José Sampaio (in memoriam), que muito antes de me ensinar a tabuada e as coisas da vida me ensinou a amar.

AGRADECIMENTO

Agradeço à Deus por não ter me deixado desistir em momento nenhum.

À minha mãe, Ilana Abrantes Sampaio e à minha avó, Maria Luiza Abrantes Sampaio por todo o apoio, confiança e amor incondicional durante esse período tão intenso de faculdade.

À meu orientador, Gerson Alberto Leiria Nunes pela dedicação extraordinária, paciência e parceria ao longo do curso.

À meus amigos que estiveram presentes em todos os momentos e me apoiaram desde o início, deixando a caminhada mais leve até aqui.

"If I am more fortunate than others I need to build a longer table not a higher fence."

RESUMO

Atualmente com avanços da Inteligência Artificial, e o aumento da capacidade de processamento, o aprendizado de máquina, as redes neurais e o aprendizado profundo possibilitam previsões das mais variadas séries temporais. Este trabalho aborda conceitos de Inteligência Artificial, *Machine Learning* e principalmente as plataformas de previsões de séries temporais amplamente utilizadas no momento: o *Facebook's Prophet* e a *Amazon DeepAR*. Neste estudo, são criados *softwares* das ambas plataformas citadas com o objetivo de por meio destas, prever o valor do ouro(oz/USD). Apresenta-se um referencial estado-da-arte e das principais tecnologias utilizadas, trabalhos correlatos ao tema, o trabalho realizado em cada plataforma bem como seus resultados: utilizando a métrica de erro MAPE, a ferramenta *Amazon DeepAR* obteve uma taxa de erro de 0,19% e o *Facebook's Prophet* de 0,006%, demostrando ser mais adequada para esse tipo de previsão. Por fim, apresenta-se as considerações relativas aos resultados encontrados juntamente com ideias de futuros trabalhos a serem realizados.

Palavras-chave: Inteligência Artificial; Aprendizado de Máquina; Séries Temporais; Facebook's Prophet. Amazon DeepAR. Preço do Ouro. Predição.

ABSTRACT

Nowadays, with advances in Artificial Intelligence, and the increase in processing capacity, machine learning, neural networks and deep learning have become able to make predictions of the most varied time series. This work presents concepts of Artificial Intelligence, Machine Learning and especially the time series prediction platforms widely used at the moment: Facebook's Prophet and Amazon DeepAR. In this paper, *software* of both aforementioned platforms are created in order to predict the value of gold (oz/USD) through them. It presents a state-of-the-art reference and the main technologies used, works related to the theme, the paper carried out on each platform as well as it's results: using the MAPE error metric, the *Amazon DeepAR* software obtained a rate error of 0.19% and *Facebook's Prophet* of 0.006%, this one proving to be more suitable for this type of prediction. Finally, considerations relating to the results found are presented with ideas for future work to be carried out.

Keywords: Artificial Intelligence, Machine Learning, Time Series, Facebook's Prophet, Amazon DeepAR, Gold Price, Prediction.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ARCH	Autoregressive Conditional Heteroskedasticity
ARFIMA	Autoregressive Fractionally integrated Moving Average
ARIMA	Autoregressive Integrated Moving Average
AWS	Amazon Web Services
Bi-LSTM	Bi-directional Long Short Term Memory
CPU	Central Processing Units
ECG	Eletrocardiograma
ETH	Ethereum
FB	Facebook
GARCH	Generalized Autoregressive Conditional Heteroskedasticity
GPU	Graphics Processing Units
GRU	Gated Recurrent Unit
IA	Inteligência Artificial
IDE	Integrated Development Environment
LSTM	Long Short Term Memory
ML	Machine Learning
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
NN	Neural Networks
RMSE	Root Mean Squared Error
RN	Redes Neurais
RNN	Recurrent Neural Network
SARIMA	Seasonal Autoregressive Integrated Moving Average
SARIMAX	Seasonal Autoregressive Integrated Moving Average with eXogenous factors

TCC	Trabalho de conclusão de curso

SUMÁRIO

1 INTRODUÇÃO

A palavra previsão, esteve presente na sociedade humana desde o início dos tempos e é definida como a ação de prever o futuro. Estas previsões podem ocorrer por meio de previsões analíticas, regressões lineares ou não lineares, algoritmos baseados em inteligência artificial, previsões climáticas ou negociação de ações algorítmicas, o homem sempre esteve interessado em prever o que o futuro detém. Muitos tentam prever o futuro, tais como: os especialistas do mercado de ações e os meteorologistas, porém poucos conseguem. Além disso, para aqueles que de fato o conseguem, algumas vezes não há como saber se foi habilidade de previsão ou sorte.

Nos últimos anos, computadores de alta capacidade de processamento tornaram-se mais comuns, do que, digamos, 30 anos atrás. Este fato gerou uma tendência de uso de ferramentas de inteligência artificial, dentre essas ferramentas de IA, o aprendizado de máquina (*machine learning*) pode ser usado em uma variedade de tarefas incluindo a robótica, carros autônomos, previsões de séries temporais etc, séries temporais é o principal tópico tratado neste referido estudo.

No decorrer deste trabalho, serão apresentados alguns conceitos como Inteligência Artificial para previsão de séries temporais, Aprendizagem de Máquina (*machine learning*), Séries Temporais, e dentro desta, aprofundar a fim de esclarecer plenamente o uso de técnicas modernas de aprendizado de máquina que são relevantes para a previsão. Ademais são apresentados dois grandes modelos de aprendizado de máquina, *Facebook Prophet* e *Amazon DeepAR*, focando junto com uma explicação teórica do modelo, sua matemática e um uso aplicado.

Segundo Korstanje (2021), o modelo de séries temporais é um modelo que faz uma previsão da variável em si apenas olhando para o desenvolvimento da própria variável ao longo do tempo. Isto significa que as séries temporais ao invés de outros modelos de séries, não tentam descrever um comportamento lógico entre as variáveis. Elas não tentam explicar o motivo das tendências das sazonalidades, elas simplesmente tornam o que há de histórico à respeito das mesmas, em fórmula matemática e tentam projetar o futuro com base nisso.

O ouro é usado em várias indústrias, incluindo eletrônica, aeroespacial, medicina, joalheria e como ferramenta de investimento. O ouro é considerado um metal precioso tanto por sua estabilidade e qualidades monetárias, portanto este é, simultaneamente, uma mercadoria, um metal precioso e uma moeda. No mercado financeiro, o mercado de ouro

atrai grande atenção de indivíduos, instituições investidoras e governos. Como um metal precioso, o ouro sempre foi considerado de valor.

O ouro tem como característica uma extraordinária capacidade de manter seu valor, mesmo durante os períodos de crise econômica, financeira e política. Os bancos centrais em todo o mundo mantêm reservas de ouro para garantir o dinheiro dos depositantes, credores da dívida externa e detentores de divisas. Os bancos centrais atualmente usam reservas de ouro para controlar a inflação. Embora o ouro não seja mais usado como dinheiro, é uma das moedas mais importantes comercializadas no mundo.

Por exemplo, o ouro hoje é a melhor ferramenta de proteção de variação de preço, especialmente para governos e bancos centrais. De acordo com Gaspareniene et al. (2018), o ouro mantém seu valor em tempos de crise econômica, quando o valor de outros ativos cai significativamente. A previsão do preço do onça do ouro é importante não só para economistas, mas também para unidades de tesouraria do governo, bancos centrais, instituições financeiras e pessoas físicas.

1.1 Problema de pesquisa

A metodologia de pesquisa se trata de uma abordagem de pesquisa explanatória, iniciando por um estudo teórico de pesquisa bibliográfica, norteado pela seguinte interrogação: Há a possibilidade de realizar uma previsão do valor do onça do ouro com as ferramentas *Amazon DeepAR* e *Facebook's Prophet*?

1.2 Importância e motivação da pesquisa

Atualmente investidores do mundo todo procuram formas seguras e confiáveis de investir a fim de ter maior segurança e ganhos financeiros. Sabendo da constante instabilidade do mercado financeiro, tornam-se necessários métodos de previsão de mercado.

Vista a necessidade de aplicações para a predição de séries temporais, especialmente no mercado financeiro, criou-se as aplicações *Facebook's Prophet*[1] e *Amazon DeepAR*[2] que são comumente utilizadas na previsão de séries temporais dentro da economia. A finalidade desta pesquisa é avaliar se é possível prever o valor do onça do

[1] https://pypi.org/project/fbprophet/
[2] https://docs.aws.amazon.com/sagemaker/latest/dg/deepar.html

ouro com as ferramentas de previsões de séries temporais citadas acima.

1.3 Objetivos

O objetivo deste é identificar se é possível ou não prever o valor do onça do ouro (ativo financeiro) utilizando ferramentas de IA relativas a predição de séries temporais tais como *Facebook's Prophet* e *Amazon DeepAR*.

1.3.1 Objetivos Específicos

- Formar uma habilidade mais sólida de programação utilizando Pyton;
- Estudar IA e ML afim de criar um referencial teórico consistente;
- Realizar uma revisão bibliográfica acerca do tema afim de, também, aumentar conhecimento;
- Buscar artigos correlatos ao tema a fim de também treinar a leitura acadêmica;
- Revisar conceitos relativos conforme as técnicas utilizadas no trabalho (Inteligência Artificial, *Machine Learning*, séries temporais etc..);
- Estudar a respeito da lógica e funcionamento de cada uma das duas ferramentas a serem analisadas (*Facebook's Prophet* e *Amazon DeepAR*);
- Iniciar codificação nas linguagens de programação de cada ferramenta;
- Desenvolver primeiros testes nas referidas linguagens;
- Analisar os resultados obtidos;
- Escrita do TCC;

1.4 Organização do texto

Nos capítulos que compõem esta referida monografia, são abordadas definições dos conceitos utilizados pelos principais autores da área, assim como trabalhos relacionados, artigos, monografias, dissertações e teses, neste estudo é abordado o desenvolvimento dos *softwares* das plataformas *Facebook's Prophet* e *DeepAR*, bem como seus resultados, conclusões à respeito do tema e trabalhos futuros.

2 METODOLOGIA

2.1 Caracterização de Pesquisa

A pesquisa científica pode ser definida como um estudo planejado, uma vez que esta tem como característica do aspecto cientifico da investigação o método de abordagem do problema. A pesquisa cientifica sempre parte de um problema que geralmente é dado por uma pergunta, e para que chegue-se à resposta dessa pergunta existem diversos métodos. (PRODANOV; FREITAS, 2013)

De acordo com Prodanov e Freitas (2013): "A pesquisa exploratória tem como objetivo buscar mais informações a respeito do assunto que será investigado, auxiliando na definição e delimitação do tema da pesquisa. Em geral está relacionada com a forma de pesquisa bibliográfica e estudos de caso, proporcionando uma maior familiaridade com o problema."

A pesquisa exploratória se dá após a delimitação do tema e sua amplitude, após isso, o pesquisador tem o objetivo de refazer as condições de um fato a ser estudado, neste caso em um ambiente controlado com a finalidade de demonstrar como um fato é produzido. Esta pesquisa é caracterizada pela manipulação das variáveis relacionadas com o objetivo de estudo, neste caso a variável em si é o preço do onça do ouro, estudando as causas e os efeitos da variação do preço do ouro, através da criação de situações de controle, de forma que não haja interferência de variáveis. Interfere-se diretamente na realidade, manipulando a variável independente, a fim de observar o que acontece com a dependente (PRODANOV; FREITAS, 2013).

2.2 Métodos

Após finalização da pesquisa experimental buscou-se trabalhos correlatos ao tema tratado neste referido estudo. Segundo Neiva e Silva (2016), existem alguns passos para se realizar uma revisão sistemática, sendo eles:

1. A definição das questões de pesquisa, sendo a principal intenção o uso de inteligência artificial e o uso de *machine learning* para predição do valor do onça do ouro.

2. A definição das palavras-chave, sendo elas: inteligência artificial, *machine learning*, *forcasting, gold price, Facebook's Prophet, Amazon DeepAR.*

3. A definição de *strings* de busca, que consiste em separar as palavras-chave,

concatená-las com seus sinônimos utilizando o conector *OR*, separá-las grupos de sinônimos, e conectá-las com *AND*, como é mostrado na figura 1.

4. Definir a bases de busca, que foi o *Google Scholar*[3] por ser uma plataforma que consegue retornar resultado de diversos congressos e eventos.

5. Houve a tentativa de incluir na *string* de busca os termos *"Facebook's prophet"e "Amazon DeepAR"* porém quando buscada a *string* tornou-se muito grande (como mostra a figura 1 abaixo).

6. Por a segunda *string* ter se tornado muito longa, houve uma seleção apenas nas quinze primeiras páginas de resultado do *Google Scholar* buscando sempre pelos artigos com mais menções e publicados mais recentemente.

7. Na última fase de seleção, realizando a leitura completa, restaram quatro publicações, que serão utilizadas para embasar, justificar e responder o problema de pesquisa. Estes trabalhos selecionados são citados no capítulo 4.

Figura 1 – *String* de busca

("INTELIGÊNCIA ARTIFICIAL" OR "ARTIFICIAL INTELLIGENCE") AND ("APRENDIZADO DE MÁQUINA" OR "MACHINE LEARNING") AND ("PREVISÃO" OR "FORECASTING") AND ("PREÇO DO OURO" OR "GOLD PRICE") AND (FACEBOOK'S PROPHET) AND (AMAZON DEEPAR)

Fonte:Próprio Autor.

2.3 Procedimentos Adotados

O procedimento adotado para a pesquisa foi primeiramente a leitura de livros, teses, trabalhos correlatos, e demais artigos relacionados à este referido estudo. Em seguida, junto a escrita da monografia foi desenvolvido o código de programação exposto no capítulo 5, este código tem por objetivo utilizar as bibliotecas do *Facebook's Prophet* e realizar simulações de previsões do valor da criptomoeda *Ether*[4]. Ademais, realizou-se um projeto de engenharia de *software* a fim de desenvolver os códigos necessário para a previsão do valor do ouro em ambas plataformas: *Facebook's Prophet* e *Amazon DeepAR*, além disso, o *dataset* do onça do ouro foi refinado para melhor se adequar ao uso nos

[3] https://scholar.google.com.br
[4] *Token* ativo do *blockchain* Ethereum

softwares. No capitulo de desenvolvimento é exposto o passo a passo da codificação em cada uma das ferramentas e os resultados gerados pelas mesmas. Por fim, o capítulo de considerações aborda as considerações acerca do trabalho desenvolvido e futuros trabalhos.

3 REVISÃO BIBLIOGRÁFICA

O presente capítulo aborda o levantamento de dados bibliográficos que alicerçaram a construção desta pesquisa, sendo estes: inteligência artificial, aprendizado de máquina, séries temporais, *Amazon DeepAR* e *Facebook's Prophet*

3.1 Inteligência Artificial

Por milhares de anos o homem estudou a fim de compreender o fenômeno do pensamento humano, como e porquê pensamos. O pensamento humano é capaz de compreender, perceber e prever ações, sentimentos, movimentos etc, já a inteligência artificial vai ainda além do pensamento humano, esta não tenta apenas compreender algo mas também construir mecanismos pensantes.

A inteligência artificial nos dias de hoje abrange diversos campos de outras ciências como a computação, a física, a química e até mesmo a medicina. Além disso, ela é usada em sistemas complexos, como por exemplo, para explorar o espaço, exames médicos e até um simples jogo de xadrez num computador. Este ramo da ciência é capaz de automatizar e sistematizar tarefas intelectuais e, portanto, é potencialmente relevante para qualquer esfera da atividade intelectual humana. Nesse sentido, ela é um campo universal (NORVIG; RUSSELL, 2004).

3.1.1 O que é inteligência artificial?

Definir a inteligência artificial é difícil mas ao longo do tempo seguem-se quatro linhas importantes de pensamento a respeito da mesma:

I. Sistemas que pensam como seres humanos: "O novo e interessante esforço para fazer os computadores pensarem... máquinas com mentes, no sentido total e literal". (HAUGELAND, 1989).

II. Sistemas que atuam como seres humanos: "A arte de criar máquinas que executam funções que exigem inteligência quando executadas por pessoas." (KURZWEIL et al., 1990).

III. Sistemas que pensam racionalmente: "O estudo das faculdades mentais pelo seu uso de modelos computacionais." (CHARNIAK; MCDERMOTT, 1985).

IV. Sistemas que atuam racionalmente: “A Inteligência Computacional é o estudo do projeto de agentes inteligentes.” (POOLE; MACKWORTH; GOEBEL, 1998).

De forma geral, as linhas de pensamento II e IV referem-se diretamente ao comportamento, tendo a inteligência artificial como uma entidade só. Já as linhas de pensamento I e III referem-se ao processo de raciocínio dentro da inteligência artificial. Ressalta-se que as linhas de pensamento I e II consideram, para termos de inteligência artificial a fidelidade do pensamento humano, enquanto que as linhas III e IV consideram de fato o avanço tecnológico em si.

3.2 Aprendizado de Máquina

Machine Learning (ML) ou Aprendizado de Máquina é o termo que define uma área da inteligência artificial responsável pelo desenvolvimento de técnicas computacionais assim como a construção de sistemas capazes de aprender algo, isto é, adquirir conhecimento de forma automática.

Este sistema de aprendizado se baseia em soluções encontradas previamente, que ficam retidas na história do *software* e com isso este reproduz novos padrões baseados nos padrões antigos que agora funcionam como um *background* de aprendizagem.

Existem diversos sistemas de aprendizado de máquina, cada um deles com suas particularidades e capacidades, estas possibilitam sua classificação quanto a linguagem de descrição, modo, paradigma e forma de aprendizado utilizada.

Atualmente, a ML é considerada a melhor forma de aprendizagem automática, porém é importante frisar que não existe um algoritmo que apresente uma ótima solução e desempenho para todos os problemas. Isto nos leva a necessidade de conhecer as habilidades e as limitações de cada um dos algoritmos de aprendizagem de máquina. A capacidade dos algoritmos de aprendizado de máquina de aprender com o contexto atual e generalizar tarefas não vistas permite melhorias tanto na segurança quanto na eficácia da prática de tais atividades, levando a melhores resultados.

O papel da ciência da computação é duplo: primeiro, no treinamento, precisa-se de algoritmos eficientes para resolver o problema de otimização de tempo de CPU, bem como para armazenar e processar a grande quantidade de dados que geralmente temos. Em segundo lugar, uma vez que um modelo é aprendido, sua representação e solução algorítmica para inferência também precisam ser eficientes de acordo com a representatividade das amostras. Em certas aplicações, a eficiência do algoritmo de

aprendizagem ou interferência, nomeadamente, a sua complexidade no espaço e no tempo, pode ser tão importante quanto a sua precisão preditiva. (ALPAYDIN, 2021)

3.3 Séries Temporais

Dados de séries temporais e suas análises têm aumentado em função do crescimento de dados em escala mundial. Nos próximos anos, em decorrência do aumento da necessidade de analises para provisões, espera-se que a quantidade e qualidade dos dados de série cresçam rapidamente.

Segundo Wei (2006), à medida que o monitoramento contínuo e a coleta de dados se tornam mais comuns, a necessidade de análises de séries temporais com técnicas estatísticas e de aprendizado de máquina aumentará. De fato, os novos modelos mais promissores combinam essas duas metodologias.

Há uma ampla gama de técnicas de séries temporais úteis para analisar e prever o comportamento humano, fenômenos científicos e dados do setor privado, pois todas essas áreas oferecem ricas matrizes de dados de séries temporais, neste estudo estudaremos o comportamento do valor do ouro.

De acordo com Nielsen (2019), "A análise de séries temporais é o esforço de extrair um resumo significativo e informações estatísticas de pontos organizados em ordem cronológica, isso é feito para diagnosticar o comportamento passado, bem como para prever o comportamento futuro."A análise de séries temporais geralmente se resume à questão da causalidade: como o passado influenciou o futuro? Às vezes, tais questões (e suas respostas) são tratadas estritamente dentro de sua questão e não como parte da questão geral de análise de séries temporais. Como resultado, uma variedade de questões e variáveis contribuiu com novas maneiras de pensar sobre conjuntos de dados de séries temporais.

Os indicadores de produção e eficiência nos mercados há muito tempo fornecem dados interessantes para serem estudados a partir de uma análise de séries temporais. Mais interessante e urgente tem sido a questão de prever futuros estados econômicos com base no passado. Essas previsões não são apenas úteis apenas para investir a fim de para obter lucros com o mercado financeiro – elas também ajudam a promover a prosperidade e evitar catástrofes sociais.

As previsões econômicas surgiram da ansiedade desencadeada por crises bancárias episódicas nos Estados Unidos e na Europa no final do século XIX e início

do século XX. Naquela época, empresários e pesquisadores se inspiraram na ideia de que a economia poderia ser comparada a um sistema cíclico, assim como se pensava que o clima se comportava. Usando métricas corretas, pensava-se, que as previsões feitas poderiam auxiliar à evitar problemas financeiros. Até a linguagem das primeiras previsões econômicas espelhava a linguagem da previsão do tempo. Isso foi involuntariamente adequado. No início do século 20, as previsões econômicas e meteorológicas eram de fato parecidas: ambas eram terríveis. Porém as aspirações dos economistas criaram um ambiente no qual, pelo menos, o progresso poderia ser esperado e assim uma variedade de instituições públicas e privadas foram formadas para rastrear dados econômicos. Os primeiros esforços de previsão econômica levaram à criação de indicadores econômicos e históricos tabulados e disponíveis ao público, indicadores estes que ainda estão em uso hoje.

Hoje em dia, os Estados Unidos e a maioria das outras nações têm milhares de pesquisadores e arquivistas governamentais cujo trabalho é registrar os dados com a maior precisão possível e disponibilizá-los ao público. Esta prática provou ser inestimável para o crescimento econômico, para evitar catástrofes econômicas e ciclos dolorosos de expansão e recessão. Além disso, as empresas se beneficiam de uma atmosfera rica em dados, pois esses conjuntos de dados públicos permitem que fornecedores de transporte, fabricantes, proprietários de pequenas empresas e até agricultores antecipem as prováveis condições de mercado futuras. Tudo isso surgiu da tentativa de identificar "ciclos de negócios" que se pensava serem as causas de falências bancárias cíclicas, uma forma inicial de análise de séries temporais em economia.

À medida que os esforços do governo obtiveram grande sucesso na coleta de dados, as organizações privadas começaram a copiar os registros governamentais. Com o tempo, as finanças mundiais e as bolsas de valores tornaram-se cada vez mais acessíveis. Os almanaques financeiros também se tornaram populares. Isso aconteceu porque os participantes do mercado se tornaram mais sofisticados e porque as tecnologias emergentes permitiram maior automação e novas formas de competir e pensar sobre preços. Todo esse registro minucioso deu origem à busca de se obter lucros fora dos mercados via matemática em vez de intuição, de uma forma inteiramente impulsionada pela estatística (e, mais recentemente, pelo aprendizado de máquina). Os primeiros pioneiros fizeram esse trabalho matemático à mão, enquanto os "quants" atuais fazem isso por métodos analíticos de séries temporais.

Um dos pioneiros da negociação mecânica, ou previsão de séries temporais

via algoritmo, foi Richard Dennis (DENNIS, 2007). Dennis era um milionário que se tornou famoso por transformar pessoas comuns, em comerciantes estelares, ensinando-lhes algumas regras específicas sobre como e quando negociar. Essas regras foram desenvolvidas nas décadas de 1970 e 1980 e espelhavam o pensamento de "IA" da década de 1980, em que a heurística ainda governava fortemente o paradigma de como construir máquinas inteligentes para trabalhar no mundo real.

Desde então, muitos negociadores pragmáticos adaptaram essas regras, que, como resultado, se tornaram mais lucrativas em um mercado automatizado, eles continuam a crescer em número e riqueza, estes estão continuamente em busca do próximo melhor lançamento tecnológico porque há muita concorrência.

George Box, (BOX, 2013), um estatístico pioneiro que ajudou a desenvolver um modelo popular de séries temporais, era um grande pragmático. Segundo Box: "Todos os modelos estão errados, mas alguns são úteis". Box (2013) fez essa declaração em resposta a uma atitude comum de que a modelagem adequada de séries temporais era uma questão de encontrar o melhor modelo para ajustar os dados. Segundo Box, a ideia de que qualquer modelo possa descrever o mundo real é muito improvável. Box fez esse pronunciamento em 1978, que parece tardio na história de um campo tão importante quanto a análise de séries temporais, entretanto, a IA era surpreendentemente jovem. Por exemplo, uma das conquistas que tornaram George Box famoso, o método Box-Jenkins descrito por Naylor, Seaks e Wichern (1972) considerado uma contribuição fundamental para a análise de séries temporais – apareceu apenas em 1970. *Time Series Analysis: Forecasting and Control (Wiley)* este livro continua popular e está agora em sua quinta edição.

O modelo original de Box-Jenkins (BOX, 2013) foi aplicado a um conjunto de dados de níveis de dióxido de carbono emitidos por um forno a gás. Embora não haja nada de estranho em um forno a gás, o conjunto de dados de 300 pontos que foi usado para demonstrar o método parece um pouco fora de moda. Certamente, conjuntos de dados maiores estavam disponíveis na década de 1970, mas lembra-se que era excepcionalmente difícil trabalhar com eles na época. Esta foi uma época que antecedeu conveniências como *R, Python* e até *C++*. Os pesquisadores tinham boas razões para se concentrar em pequenos conjuntos de dados e métodos que minimizavam os recursos de computação.

Análise e previsão de séries temporais foram desenvolvidas com os computadores, com conjuntos de dados maiores e ferramentas de codificação mais fáceis, abrindo caminho para mais experimentação e a capacidade de responder a perguntas mais

interessantes. A história de competições de previsão do professor Rob Hyndman fornece exemplos adequados de como as competições de previsão de séries temporais se desenvolveram a uma taxa paralela à dos computadores. O professor Hyndman coloca o "primeiro estudo não trivial de precisão de previsão de séries temporais" como ocorrendo em uma dissertação de doutorado de 1969 na Universidade de Nottingham, apenas um ano antes da publicação do método Box-Jenkins. Esse primeiro esforço foi logo seguido por competições organizadas de previsão de séries temporais, as primeiras apresentando cerca de 100 conjuntos de dados no início da década de 1970. Nada mal, mas certamente algo que poderia ser feito à mão, se fosse absolutamente necessário.

No final da década de 1970, os pesquisadores montaram uma competição com cerca de 1.000 conjuntos de dados, uma escala impressionante. Aliás, essa era também foi marcada pelo primeiro microprocessador comercial, o desenvolvimento de disquetes, os primeiros computadores pessoais da *Apple* e a linguagem de computador *Pascal*[5]. É provável que algumas dessas inovações tenham sido úteis. Uma competição de previsão de séries temporais do final da década de 1990 incluiu 3.000 conjuntos de dados. Embora essas coleções de conjuntos de dados fossem substanciais e, sem dúvida, refletissem uma enorme quantidade de trabalho e engenhosidade para coletar e organizar, elas são ofuscadas pela quantidade de dados agora disponíveis. Os dados de séries temporais estão em toda parte, e em breve tudo será uma série temporal.

Esse rápido crescimento no tamanho e na qualidade dos conjuntos de dados deve suas origens aos enormes avanços que foram feitos na computação nas últimas décadas. Os engenheiros de *hardware* conseguiram continuar a tendência descrita pela Lei de Moore - uma previsão de crescimento exponencial na capacidade de computação - (CHIEN; KARAMCHETI, 2013) durante esse período. À medida que o *hardware* se tornava menor, mais poderoso e mais eficiente, era fácil ter muito mais dele, de forma acessível – para criar tudo, desde computadores portáteis em miniatura com sensores conectados até grandes *data centers* que alimentam a internet. Mais recentemente, *wearables*, técnicas de aprendizado de máquina e *GPU*s revolucionaram a quantidade e a qualidade dos dados disponíveis para estudo.

Sem dúvida as séries temporais serão beneficiadas à medida que o poder de computação aumenta, porque muitos aspectos dos dados de séries temporais são computacionalmente exigentes. Devido à aceleração dos recursos computacionais e a disponibilidade de dados, pode-se esperar que a análise de séries temporais continue em

[5]Pascal é uma linguagem de programação orientada à objetos, que recebeu este nome em homenagem ao matemático e físico Blaise Pascal. A linguagem foi criada por Niklaus Wirth

seu ritmo acelerado de desenvolvimento, além disso, a estatística é uma ciência muito jovem. O progresso em estatísticas, análise de dados e séries temporais sempre dependeu fortemente de quando, onde e como os dados estavam disponíveis e em que quantidade. O surgimento da análise de séries temporais está ligado não apenas aos desenvolvimentos na teoria das probabilidades, assim como se relaciona com o desenvolvimento na capacidade de armazenamento de dados por parte dos países.

Uma referência para o início da análise de séries temporais é a aplicação de modelos autorregressivos a dados reais. Isso não aconteceu até a década de 1920. Udny Yule (YULE, 1927), um físico experimental que se tornou professor de estatística na Universidade de Cambridge, aplicou um modelo autorregressivo aos dados de manchas solares, oferecendo uma nova maneira de pensar sobre os dados em contraste com métodos projetados para ajustar a frequência de uma oscilação.

Yule (1927) apontou que um modelo autorregressivo não começou com um modelo que assumiu periodicidade, Segundo Yule (1927), "Quando a análise do periodograma é aplicada a dados que respeitam qualquer fenômeno físico na expectativa de eliciar uma ou mais periodicidades verdadeiras, geralmente há, como me parece, uma tendência a partir da hipótese inicial de que a periodicidade ou periodicidades são mascaradas apenas por tais flutuações sobrepostas mais ou menos aleatórias – flutuações que não perturbam de forma alguma o curso constante da função ou funções periódicas subjacentes... não parece haver razão para supor que seja a hipótese mais provável a priori."

À medida que o mundo se tornou um lugar mais ordenado, registrado e previsível, os primeiros problemas na análise prática de séries temporais foram apresentados pelo setor empresarial. Os problemas de séries temporais orientados para negócios eram importantes e não excessivamente teóricos em suas origens. Estes incluíam a previsão de demanda, estimativa de preços futuros de matérias-primas e cobertura de custos de fabricação. Nesses casos de uso industrial, as técnicas eram adotadas quando funcionavam e rejeitadas quando não funcionavam. Provavelmente ajudou que os trabalhadores industriais tivessem acesso a conjuntos de dados maiores do que os disponíveis para os acadêmicos na época (como continua a ser o caso agora). Isso significava que, às vezes, técnicas práticas, mas teoricamente pouco exploradas, passaram a ser amplamente utilizadas antes de serem bem compreendidas.
(NIELSEN, 2019)

O aprendizado de máquina inicial na análise de séries temporais remonta a muitas

décadas. Um artigo muito citado de 1969, "*The Combination of Forecasts*" (BATES; GRANGER, 1969), analisou a ideia de combinar previsões em vez de escolher uma "melhor" como forma de melhorar o desempenho da previsão. Essa ideia era, a princípio, abominável para os estatísticos tradicionais, mas os métodos de conjunto tornaram-se a melhor opção em muitos problemas de previsão.

Mais recentemente, usos práticos para análise de séries temporais e aprendizado de máquina surgiram na década de 1980 e incluíam uma ampla variedade de cenários:

• Especialistas em segurança de computadores propuseram a detecção de anomalias como método de identificação de intrusões.

• A distorção do tempo dinâmico, um dos métodos dominantes para calcular a similaridade de séries temporais, entrou em uso porque o poder de computação finalmente permitiria o cálculo razoavelmente rápido de correlações, digamos, entre diferentes gravações de áudio.

• As redes neurais recursivas foram inventadas e mostraram-se úteis para extrair padrões de dados corrompidos.

A análise e a previsão de séries temporais ainda não chegaram em seus melhores métodos, até o momento, a análise de séries temporais permanece dominada por métodos estatísticos tradicionais, bem como por técnicas de aprendizado de máquina mais simples, como conjuntos de árvores e ajustes lineares. Ainda estamos à espera de um grande salto em frente para prever o futuro.

3.3.1 Lidando com séries temporais

Nesta seção, discute-se problemas que podem surgir durante o pré-processamento de dados de séries temporais. Alguns desses problemas são familiares para analistas de dados experientes, mas existem dificuldades específicas impostas por registros de data e horário dos dados. Como em qualquer tarefa de análise de dados, o refinamento e o pré processamento adequado dos dados geralmente é a etapa mais importante de um *pipeline* de registros de data e horário. Técnicas avançadas não podem corrigir dados confusos.

A maioria dos analistas de dados precisará encontrar, alinhar, refinar, suavizar e pré processar seus próprios dados para aprender a análise de séries temporais ou para realizar um trabalho significativo em suas organizações. Ao preparar os dados, é preciso realizar várias tarefas, desde unir colunas diferentes até remostrar dados irregulares ou ausentes e alinhar séries temporais com diferentes eixos de tempo.

Discute-se as seguintes habilidades úteis para encontrar e refinar dados de séries temporais:

• Encontrar dados de séries temporais de repositórios *online*;

• Descobrir e preparar dados de séries temporais de fontes não originalmente destinadas a séries temporais;

• Resolver problemas comuns que são encontrados com dados de séries temporais;

Se estivermos interessados em onde encontrar dados de séries temporais e como refiná-los, o melhor recurso dependerá de qual deles é o objetivo principal:

• Encontrar um conjunto de dados apropriado para fins de aprendizado ou experimentação;

• Criação de um conjunto de dados de série temporal a partir de dados existentes que não são armazenados em um formato explicitamente orientado ao tempo;

No primeiro caso, deve-se encontrar conjuntos de dados existentes com referências conhecidas para poder verificar se a análise está sendo feita corretamente. Nesses casos, provavelmente será necessário refinar os dados para uma finalidade específica, mesmo que algum trabalho preliminar já tenha sido feito.

No segundo caso, deve-se pensar em maneiras eficazes de identificar dados com registro de data e hora interessantes, transformá-los em uma série, refiná-los e alinhá-los com outros dados com registro de data e hora para criar dados de séries temporais válidas e interessantes.

A melhor maneira de aprender uma técnica analítica ou de modelagem é executá-la em uma variedade de conjuntos de dados e ver como aplicá-la e se ela o ajuda a atingir uma meta concreta. Nesses casos, é útil ter várias opções de dados preparadas.

3.3.2 Analisando conjuntos de séries temporais

Anteriormente neste capítulo, foi discutido conceito de uma série temporal encontrada, que são dados de séries temporais que foram reunidas a partir de fontes de dados em estado desorganizado.

Reunir uma série temporal de transações para um cliente específico a partir de um banco de dados SQL que armazena as transações de uma empresa é um exemplo claro. Nesse caso, a série temporal poderia ser construída desde que um registro de data e horário fosse salvo no banco de dados. Também pode-se imaginar outras séries temporais construídas a partir dos mesmos dados, como uma série temporal de volume

total de transações por dia para a empresa ou volume total em dólares para clientes do sexo feminino por semana. Pode-se até imaginar a geração de dados de séries temporais multivariadas, como uma série temporal que indicaria separadamente o volume total de todos os clientes com menos de 18 anos por semana, o total de dólares gastos por mulheres com mais de 65 anos por semana e o total de gastos com anúncios da empresa por semana. Isso resultaria em três indicadores em cada passo de tempo, uma série temporal multivariada.

Encontrar dados de séries temporais em dados estruturados não armazenados explicitamente como uma série temporal pode ser fácil, no sentido de que o registro de data e hora é onipresente. Aqui estão alguns exemplos de onde pode haver registros de tempo (dia e hora) em um banco de dados:

Se houver um registro de tempo no banco de dados, há o potencial de construir uma série temporal. Mesmo que tudo o que se faça seja registrar a hora em que um arquivo foi acessado sem outras informações, criará uma série temporal. Por exemplo, nesse caso, seria possível modelar o tempo médio entre os registros de tempo com cada um marcado de acordo com seu registro posterior, de modo que sua série temporal consistiria em tempo em seu eixo temporal e tempo delta em seu eixo de valor. Pode-se ir além, agregando esses tempos delta como médias ou totais em períodos maiores, ou pode mantê-los registrados individualmente.

Em alguns casos, o tempo não é explícito nos dados, mas é contabilizado na lógica subjacente do conjunto de dados. Por exemplo, pode-se pensar em seus dados como "distância versus valor" quando a distância está sendo causada por um parâmetro experimental conhecido, como retrair um sensor de uma posição à uma taxa conhecida. Se for possível mapear uma de suas variáveis para o tempo, terá uma série temporal. Como alternativa, se um de seus eixos tiver uma distância conhecida e uma relação de ordenação (como comprimento de onda), também é considerado como dados de séries temporais. Até este ponto, foi apresentado onde encontrar dados de séries temporais e como processá-los. Agora será apresentado como criar dados de séries temporais por meio de simulação.

Essa discussão prossegue em três partes. Primeiro, comparar simulações de dados de séries temporais com outros tipos de simulações de dados, observando quais novas áreas de interesse particular emergem quando temos que explicar a passagem do tempo. Em segundo lugar, examinar algumas simulações baseadas em código. Em terceiro lugar, discutir algumas tendências gerais na simulação de séries temporais.

A simulação de dados é uma área da ciência de dados que raramente é ensinada, mas que é uma habilidade particularmente útil para dados de séries temporais. Isso decorre de uma das desvantagens de ter dados temporais: dois pontos de dados na mesma série temporal não são exatamente comparáveis, pois ocorrem em momentos diferentes. Se quisermos pensar no que poderia ter acontecido em um determinado momento, entramos no mundo da simulação.

As simulações podem ser simples ou complexas. No lado mais simples, encontra-se dados sintéticos em qualquer livro de estatística sobre séries temporais, como na forma de uma jogada aleatória. Estes são geralmente gerados como somas cumulativas de um processo aleatório (como a *rnorm* de *R*) ou por uma função periódica (como uma curva senoidal). No lado mais complexo, muitos cientistas e engenheiros fazem suas carreiras simulando séries temporais. As simulações de séries temporais continuam sendo uma área ativa de pesquisa - e computacionalmente exigente - em muitos campos, incluindo:

- Meteorologia
- Finança
- Epidemiologia
- Química quântica
- Física de plasma

Em alguns desses casos, as regras fundamentais de comportamento são bem compreendidas, mas ainda pode ser difícil explicar tudo o que pode acontecer devido à complexidade das equações (meteorologia, química quântica, física de plasma). Em outros casos, nem todas as variáveis preditivas podem ser conhecidas, e os especialistas nem têm certeza de que as previsões perfeitas podem ser feitas devido à natureza não linear estocástica dos sistemas estudados (finanças, epidemiologia etc.)

Uma série temporal é um conjunto de dados coletados sequencialmente ao longo do tempo. Por exemplo, pensemos em qualquer gráfico em que o eixo x seja alguma medida de tempo – qualquer coisa desde o número de estrelas no Universo desde o *Big Bang* até hoje ou a quantidade de energia liberada a cada nanossegundo de uma reação nuclear. Os dados por trás de ambos são séries temporais. O gráfico no aplicativo de clima de um telefone mostrando a temperatura esperada para os próximos 7 dias? Esse também é o enredo de uma série temporal.

Mas os dados de séries temporais não são exclusivos do clima. O campo da medicina adotou técnicas de análise de séries temporais com a invenção em 1901

do primeiro eletrocardiograma prático pelo médico holandês Willem Einthoven. O eletrocardiograma, (ECG) como é comumente conhecido, produz o padrão familiar de batimentos cardíacos que vemos agora na máquina ao lado da cama de um paciente em todos os dramas médicos.

Hoje, um dos campos de previsão mais discutidos é a economia. Existem canais de televisão inteiros dedicados a analisar as tendências do mercado de ações. Os governos usam a previsão econômica para aconselhar a política do banco central, os políticos usam a previsão econômica para desenvolver suas plataformas e os líderes empresariais usam a previsão econômica para orientar suas decisões.

3.3.3 O problema com dados dependentes

Então, por que a previsão de séries temporais exige sua própria abordagem exclusiva? De uma perspectiva estatística, pode-se ver um gráfico de dispersão de séries temporais com uma tendência relativamente clara e tentar ajustar uma linha usando regressão padrão – a técnica para ajustar uma linha reta através dos dados. O problema é que isso viola a suposição de independência exigida pela regressão linear. Para ilustrar a dependência de séries temporais com um exemplo, supomos que um jogador está jogando um dado imparcial. Ele acabou de rolar um dois e perguntar qual será o próximo valor. Esses dados são independentes; rolagens anteriores não têm efeito em rolagens futuras, portanto, saber que o resultado anterior foi um dois não fornece nenhuma informação sobre o próximo lançamento.

No entanto, em uma situação diferente, suponha que seja feita uma ligação para alguém de um local não revelado em algum lugar da Terra e tenta-se adivinhar a temperatura nesta localização. A melhor aposta seria adivinhar alguma temperatura global média para aquele dia. Mas agora, suponha que seja sabido que a temperatura de ontem em determinada localização foi de 30°C. Isso fornece uma grande quantidade de informações, porque sabe-se intuitivamente que a temperatura de ontem e a temperatura de hoje estão ligadas de alguma forma, eles não são dados independentes.

Usando dados de séries temporais, não pode-se embaralhar aleatoriamente a ordem dos dados sem perturbar as tendências, dentro de uma margem de erro razoável. A ordem dos dados é importante; não é independente. Quando os dados são dependentes assim, um modelo de regressão pode mostrar significância estatística por acaso, mesmo quando não há correlação verdadeira, com muito mais frequência do que o nível de

confiança escolhido sugeriria.

Valores altos tendem a seguir valores altos e valores baixos tendem a seguir valores baixos, é mais provável que um conjunto de dados de série temporal mostre mais valores agregados de valores altos ou baixos do que estariam presentes, e isso, por sua vez, pode levar ao aparecimento de mais correlações do que estariam presentes.

3.3.4 Média móvel e suavização exponencial

Possivelmente, a forma mais simples de previsão é a média móvel, conhecida por MA *(*moving average) demonstrada na equação 1

$$M_\tau(t) = \sum_{i=0}^{\tau-1} \frac{x_{t-i}}{\tau} \tag{1}$$

Sendo:

- M é a média móvel;
- x é a série temporal (ativo observado);
- τ é o período utilizado para o cálculo da média móvel;
- t é o instante observado; e
- $\sum$ é o somatório (de soma).

Muitas vezes, uma média móvel é usada como uma técnica de suavização para encontrar uma linha mais reta através de dados com muita variação. Cada ponto de dados é ajustado ao valor da média de n pontos de dados circundantes, com n sendo referido como o tamanho da janela. Um tamanho de janela de dez, por exemplo, ajustaria-se um ponto de dados para ser a média dos cinco valores anteriores e dos cinco valores posteriores. Em uma configuração de previsão, os valores futuros são calculados como a média dos n valores anteriores, então, novamente, com um tamanho de janela de 10, isso significa a média dos dez valores anteriores.

O ato de equilíbrio com uma média móvel é que deseja-se uma amplitude de projeção grande para suavizar o "ruído" e capturar a tendência real, mas com um tamanho de amplitude maior, as previsões ficarão significativamente atrasadas em relação à tendência à medida que voltar mais e mais para calcular a média. A ideia por trás da suavização exponencial, EMA (*exponential moving average*) disposta na equação 2 é aplicar pesos exponencialmente decrescentes aos valores que estão sendo calculados ao

longo do tempo, dando mais peso aos valores recentes e menos aos valores mais antigos. Isso permite que a previsão seja mais reativa às mudanças, enquanto ainda ignora uma boa quantidade de ruído.

$$E_\alpha(t) = \alpha \sum_{i=0}^{\tau} (1-\alpha)^i x_{(i+1)} \tag{2}$$

Sendo α o peso atribuído aos instantes de tempo, conforme α, pode ser dado um peso maior ou menor aos valores mais recentes, tornando essa média muito mais flexível ao se adaptar às necessidades de análise. Geralmente α assume o valor 2/(n+1).

A suavização exponencial originou-se na década de 1950 com a suavização exponencial simples, que não permite tendência ou sazonalidade. Holt (2004) avançou a técnica em 1957 para permitir uma tendência com o que chamou de suavização exponencial dupla; e em colaboração com Peter Winters, (HOLT, 2004) adicionou suporte à sazonalidade em 1960, no que é comumente chamado de suavização exponencial de Holt-Winters.(CHATFIELD, 1978)

A desvantagem desses métodos de previsão é que eles podem ser lentos para se ajustar a novas tendências e, portanto, os valores previstos ficam aquém da realidade - eles não se sustentam bem em prazos de previsão mais longos e há muitos hiperparâmetros para ajustar, o que pode ser difícil e processo muito demorado.

3.3.5 Modelo ARIMA

Em 1970, os matemáticos Box (2013) e Jenkins (1961) publicaram *Time Series: Forecasting and Control*, que descrevia o que hoje é conhecido como modelo Box-Jenkins. Essa metodologia levou a ideia da média móvel adiante com o desenvolvimento do ARIMA (*autoregressive integrated moving average*). ARIMA é uma generalização do modelo auto-regressivo de médias móveis (ARMA). Estes os modelos são ajustados aos dados da série temporal para entender melhor os dados ou para prever pontos futuros na série. Como termo ARIMA é frequentemente usado de forma intercambiável com (JENKINS, 1961), embora tecnicamente, Box-Jenkins se refira a um método de otimização de parâmetros para um modelo ARIMA.

ARIMA é um acrônimo de três conceitos: Auto regressivo (AR), Integrado (I) e Média Móvel (MA). Já foi demonstrada a parte da média móvel. Auto regressivo significa que o modelo usa a relação dependente entre um ponto de dados e algum número de pontos de dados defasados. Ou seja, o modelo prevê valores futuros com base em valores

anteriores. Isso é semelhante a prever que amanhã estará quente porque tem estado quente durante toda a semana até agora.

A parte integrada significa que, em vez de usar qualquer ponto de dados brutos, é usada a diferença entre esse ponto de dados e algum ponto de dados anterior. Essencialmente, isso significa que converte-se uma série de valores em uma série de mudanças nos valores. Intuitivamente, isso sugere que amanhã será mais ou menos a mesma temperatura de hoje porque a temperatura durante toda a semana não variou muito.

Cada um dos componentes AR, I e MA de um modelo ARIMA é explicitamente especificado como um parâmetro no modelo. Tradicionalmente, p é usado como o número de observações de atraso a serem usadas, também conhecido como ordem de atraso. O número de vezes que uma observação bruta é diferenciada, ou o grau de diferenciação, é conhecido como d, e q representa o tamanho da janela de média móvel. Assim surge a notação padrão para um modelo ARIMA de ARIMA(p, d, q), onde p, d, e q são todos inteiros não negativos.

Um problema com os modelos ARIMA é que eles não suportam sazonalidade ou dados com ciclos repetidos, como temperatura subindo durante o dia e caindo à noite ou subindo no verão e caindo no inverno.

SARIMA, ou *Seasonal* ARIMA, foi desenvolvido para superar essa desvantagem do ARIMA de não suportar a sazonalidade. Semelhante à notação ARIMA, a notação para um modelo SARIMA é SARIMA(p, d, q)(P, D, Q)m, com P sendo a ordem autorregressiva sazonal, D a ordem de diferença sazonal, Q a ordem média móvel sazonal, e m o número de instantes de tempo para um único período sazonal.

Encontra-se na literatura outras variações de modelos ARIMA, incluindo VARIMA (Vector ARIMA, para casos com várias séries temporais como vetores); FARIMA (*Fractional* ARIMA) ou ARFIMA (*Fractionally Integrated* ARMA), ambos os quais incluem um grau de diferenciação fracionária permitindo uma memória longa no sentido de que observações distantes no tempo podem ter dependências não desprezíveis; e SARIMAX, um modelo sazonal ARIMA onde o X representa variáveis exógenas ou adicionais adicionadas ao modelo, como adicionar uma previsão de chuva a um modelo de temperatura.

ARIMA normalmente exibe resultados muito bons, mas a desvantagem é a complexidade. Ajustar e otimizar modelos ARIMA é muitas vezes computacionalmente caro em questões de complexidade e desempenho ademais os resultados bem-sucedidos podem depender da habilidade e experiência do previsor. Não é um processo escalável,

mas mais adequado para análises, caso necessário, por profissionais qualificados.

3.3.6 Modelo ARCO/GARCH

Quando a variação de um conjunto de dados não é constante ao longo do tempo, os modelos ARIMA enfrentam problemas ao modelá-lo. Em economia e finanças, em particular, isso pode ser comum. Em uma série temporal financeira, grandes retornos tendem a ser seguidos por grandes retornos e pequenos retornos tendem a ser seguidos por pequenos retornos. O primeiro é chamado de alta volatilidade e o segundo de baixa volatilidade.

Modelos de Heterocedasticidade Condicional Autorregressiva (ARCH) foram desenvolvidos para resolver este problema. A heterocedasticidade é uma maneira elegante de dizer que a variação ou dispersão dos dados não é constante, com o termo oposto sendo homocedasticidade.

A diferença é visualizada aqui: Engle (1995) introduziu o primeiro modelo ARCH em 1982, descrevendo a variância condicional em função dos valores anteriores. Por exemplo, há muito mais incerteza sobre o uso diurno de eletricidade do que sobre o uso noturno. Em um modelo de uso de eletricidade, então, podemos supor que as horas diurnas têm uma variação específica, e o uso durante a noite teria uma variação menor.

Bollerslev, Chou e Kroner (1992) introduziram um componente de média móvel no modelo em 1986 com seu modelo ARCH Generalizado, ou GARCH. No exemplo da eletricidade, a variação no uso foi uma função da hora do dia. Mas talvez as oscilações na volatilidade não ocorram necessariamente em horários específicos do dia, mas as oscilações são aleatórias. Isto é quando GARCH é útil.

Ambos os modelos ARCH e GARCH não podem lidar com tendência nem sazonalidade embora, muitas vezes, na prática, um modelo ARIMA possa primeiro ser construído para extrair a variação sazonal e a tendência de uma série temporal, e então um modelo ARCH pode ser usado para modelar a variação esperada.

3.3.7 Método LSTM, biLSTM e GRU

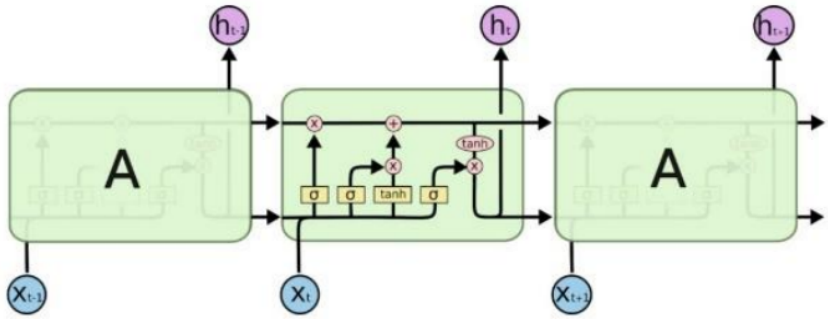
Os métodos LSTM, BiLSTM e GRU são amplamente utilizados para prever o preço do ouro. O algoritmo de aprendizado profundo LSTM é um algoritmo neural

recorrente de rede, criado por Hochreiter e Schmidhuber em 1997 para eliminar as desvantagens da arquitetura RNN. No RN, cada item nos dados de entrada é examinado iterativamente considerando o valor da saída anterior. Em geral, o RNN está em desvantagem porque os gradientes são perdidos ao aprender longas sequências de dados. O LSTM resolve este problema determinando quando certas informações são usadas ou não.

Segundo YURTSEVER (2021) o modelo LSTM consiste em camada de entrada, camadas ocultas e camada de saída. Cada bloco tem várias células de memória ligadas a ele, três unidades multiplicadoras, entradas, saídas e portas de esquecimento. Dependendo desses componentes, o bloco de células LSTM contém três portas e uma unidade de célula de memória capaz de esquecer ou memorizar informações para determinar quanta informação deve ser transferida para a próxima célula.

A Figura 2 demonstra a arquitetura LSTM.

Figura 2 – Arquitetura LSTM



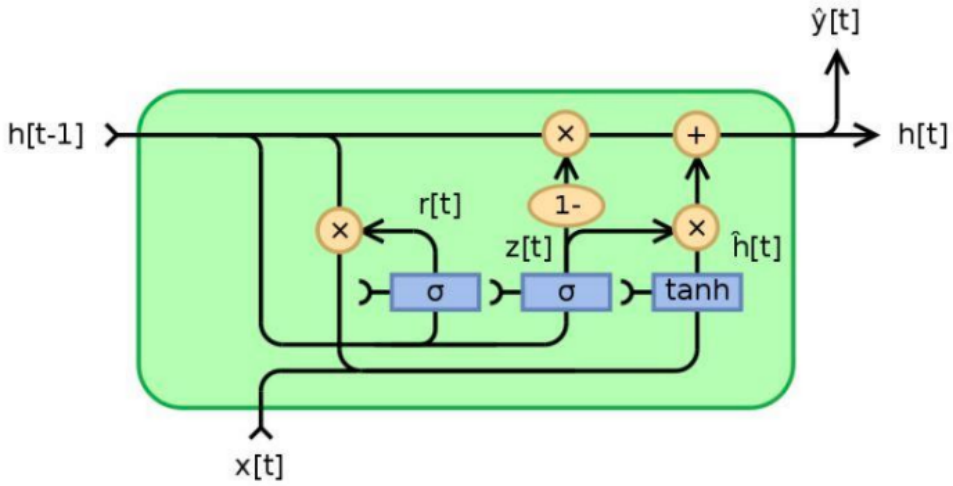
Fonte: (YURTSEVER, 2021)

Para superar as limitações da célula LSTM, que pode trabalhar no conteúdo anterior, mas não pode usar o futuro, Schuster e Paliwal (1997) apresentaram redes que consistem em duas camadas ocultas de LSTM diferentes com saídas semelhantes em direções opostas. Com essa arquitetura, informações anteriores e futuras são usadas na camada de saída. O BiLSTM, em particular, integra o conceito de ordem temporal realizando o processo de aprendizagem em duas direções, do passado para o futuro e do futuro para o passado.

O modelo GRU, segundo YURTSEVER (2021), possui um mecanismo de porta para regular o fluxo de informações para lembrar o contexto em várias etapas de tempo (Cho et al., 2014). Este modelo usa um portão de atualização e um portão de redefinição

para determinar quais informações passadas podem ser mantidas ou esquecidas. GRU é semelhante ao LSTM, combina o esquecimento e a entrada do LSTM em um único portão de estado. O portão de estado decide quais informações de entrada ficam enquanto o portão de *reset* decide quais são descartadas. A Figura abaixo mostra a estrutura de uma unidade de GRU. O modelo GRU supera o modelo LSTM em termos de tempo de treinamento e precisão de previsão devido à sua estrutura muito simples (Jianwei et al., 2019).

Figura 3 – Arquitetura GRU



Fonte: (YURTSEVER, 2021)

Existem duas portas no GRU, como demonstrado na figura 3: as portas de atualização e de reinicialização. Esses dois portões decidem quais informações são transmitidas para a saída e quais as informações a serem excluídas e transferidas.

3.4 Redes Neurais Recorrentes - RNN

Um desenvolvimento relativamente recente na previsão de séries temporais é o uso de Redes Neurais Recorrentes, RNNs (*Recurrent Neural Networks*). Isso foi possível com o desenvolvimento da unidade de *Long Short-Term Memory* (LSTM), por Hochreiter e Schmidhuber (1997). Essencialmente, uma unidade LSTM permite que uma rede neural processe uma sequência de dados, como fala ou vídeo, em vez de um único ponto de dados, como uma imagem.
(HOCHREITER; SCHMIDHUBER, 1997)

Uma RNN padrão é chamada de recorrente porque possui *loops* embutidos nela,

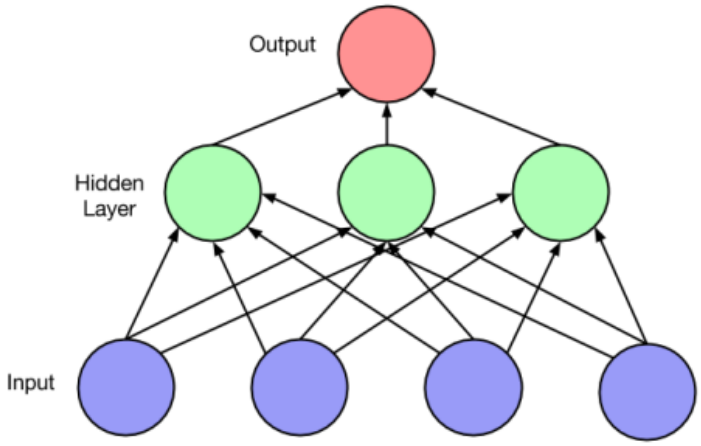
que é o que lhe dá memória, ou seja, dá acesso às informações anteriores. A composição de uma RNN é feita de *inputs* que são as entradas da rede neural, *hidden layers* que são as camadas ocultas da RNN, onde ficam as células LSTM, e o *output* que é a saída da RNN. A figura 4 extraída do artigo de Lipton, Berkowitz e Elkan (2015) demonstra o a arquitetura explicada previamente. Uma rede neural básica pode ser treinada para reconhecer uma imagem de um pedestre em uma rua aprendendo como um pedestre se parece a partir de imagens anteriores, mas não pode ser treinada para identificar que um pedestre em um vídeo em breve estará atravessando a rua com base na aproximação do pedestre observada em *frames* anteriores do vídeo. Não tem conhecimento da sequência de imagens que leva o pedestre a sair para a estrada. A memória de curto prazo é o que a rede precisa temporariamente para fornecer contexto, mas essa memória se degrada rapidamente.

Os primeiros RNNs tinham um problema de memória: não era muito longo. Na frase aviões voam no ..., um simples RNN pode adivinhar que a próxima palavra será céu. Mas com eu fui para a França para férias no verão passado. Por isso passei minha primavera aprendendo a falar..., não é tão fácil para o RNN adivinhar que o francês vem depois; ele entende que a palavra para uma língua deve vir em seguida, mas esqueceu que a frase começou mencionando a França. Um LSTM, no entanto, fornece esse contexto necessário. Dá mais longevidade à memória de curto prazo da rede. No caso de dados de séries temporais em que os padrões podem ocorrer novamente em longas escalas de tempo, os LSTMs podem ter um desempenho muito bom.

A previsão de séries temporais com LSTMs ainda está em sua infância quando comparada com os outros métodos de previsão discutidos aqui; no entanto, está se mostrando promissor. Uma forte vantagem sobre outras técnicas de previsão é a capacidade das redes neurais de capturar relacionamentos não lineares. Mas, como acontece com qualquer problema de aprendizado profundo, a previsão LSTM requer uma grande quantidade de dados, poder de computação e tempo de processamento.

Além disso, há muitas decisões a serem tomadas em relação à arquitetura do modelo e aos hiperparâmetros a serem utilizados, o que exige um previsor muito experiente. Na maioria dos problemas práticos, onde o orçamento e os prazos devem ser considerados, um modelo ARIMA costuma ser a melhor escolha. No caso deste referido estudo, o modelo ARIMA é apresentado como referencial teórico e utilizado em parte pela ferramenta do *Facebook's Prophet*.

Figura 4 – Arquitetura de uma RNN



Fonte: (LIPTON; BERKOWITZ; ELKAN, 2015)

3.5 Amazon DeepAR

DeepAR[6] é um modelo desenvolvido por pesquisadores da *Amazon*. *DeepAR* fornece uma interface para construir modelos de séries temporais usando uma arquitetura de aprendizado profundo baseada em RNNs. A vantagem de usar o *DeepAR* é que ele vem com uma interface mais fácil de usar para construção de modelo de previsão quando comparado ao Keras. O *DeepAR* pode ser considerado um concorrente do *Facebook's Prophet*, ambas plataformas tentam fornecer uma interface de modelo que pode auxiliar sua utilização.

A plataforma *DeepAR* prevê séries temporais univariadas ou multivariadas usando RNNs. O grande diferencial do *DeepAR* é que ele compreende, em um único modelo, diversas séries temporais ao mesmo tempo. Ele é projetado para se beneficiar de correlações entre várias séries temporais e portanto, é um ótimo modelo para previsão multivariada. No entanto, também pode ser aplicado a uma série temporal única.

O *DeepAR* modela vários tipos de variáveis de sazonalidade. Também cria variáveis para o dia do mês, dia do ano e outras variáveis derivadas. Diferentemente do *Facebook's Prophet*, o *Amazon DeepAR* não dá ao usuário muito controle ou informações sobre o funcionamento interno do algoritmo, o que faz com que este se torne uma caixa

[6]https://docs.aws.amazon.com/sagemaker/latest/dg/deepar.html

preta para o usuário, com exceção de alguns hiperparâmetros que serão demonstrados nesta subseção. Na plataforma *DeepAR*, o usuário tem a opção de entrar com alguns dados, sendo eles os seguintes:

– A tamanho da previsão, que é a janela de tempo que deseja-se prever de uma determinada série temporal.

– O CTX, que é um controle onde pode se especificar se deseja executar o programa na CPU ou na GPU. (A execução na GPU é muito mais rápida, mas isso só é possível se o computador em questão possui uma GPU disponível em seu *hardware*.)

– O número de épocas é o quantidade de séries de dados à serem passados pela rede neural profunda subjacente.

– A taxa de aprendizado é o tamanho do passo para o otimizador do sistema de rede neural, ou seja, uma taxa de aprendizado maior permite que o usuário faça sua previsão de forma mais rápida, porém corre-se o risco de perder informações importantes.

Ao fazer a previsão, após a inserção dos dados, a plataforma realiza diversas - e aqui fala-se em grandes quantidades de previsões - o que para um usuário comum pode parecer inútil, mas a plataforma compreende que é necessário ter diversas previsões para se ter uma boa margem de erro.

Observa-se algo difícil no *DeepAR*: é difícil controlar a aleatoriedade neste modelo. Ao fazer a entrada de dados com a série temporal afim de fazer a previsão, o *DeepAR* faz um grande número de potenciais trajetórias para a previsão final, e a mediana delas pode ser usada como previsão final. Ademais, se executar o código uma segunda vez, essas trajetórias não são as mesmas e, portanto, sua mediana será um pouco diferente também. Isso pode ser considerado uma desvantagem do *DeepAR*.
(KORSTANJE, 2021)

3.6 Facebook's Prophet

O *Facebook's Prophet*[7] é uma ferramenta poderosa para criar, visualizar e otimizar previsões. O *Prophet*, é capaz de entender quais fatores irão conduzir seus resultados futuros de acordo com os dados passados e permitir que decisões sejam tomadas de forma mais confiantes. O *Facebook's Prophet* realiza tarefas e objetivos por meio de uma interface de programação muito flexível, projetada tanto para iniciantes quanto para especialistas. Para usar o *Prophet* não precisa de um conhecimento profundo da

[7] https://facebook.github.io/prophet/

matemática ou estatística por trás das técnicas de previsão de séries temporais, embora a plataforma inclua um rico conjunto de recursos que permite por em prática a análise de dados com grande efeito. A plataforma trabalha em um paradigma estruturado em que cada problema segue o mesmo padrão, permitindo que gaste-se menos tempo descobrindo como otimizar previsões e mais tempo melhorando decisões.
(RAFFERTY, 2021)

Esta seção apresenta as ideias fundamentais por trás da previsão de séries temporais e discute algumas das principais iterações do modelo que eventualmente levaram ao desenvolvimento do *Prophet*. Aqui é apresentado o que são dados de séries temporais e por que eles devem ser tratados de forma diferente dos dados de séries não temporais.

O *Prophet* foi desenvolvido internamente no *Facebook* por Sean J. Taylor e Ben Letham para superar dois problemas frequentemente encontrados em outras metodologias de previsão: as ferramentas de previsão mais automáticas disponíveis tendiam a ser muito inflexíveis e incapazes de acomodar suposições adicionais, e a previsão mais robusta ferramentas exigiriam um analista experiente com habilidades especializadas em ciência de dados. O *Facebook* estava experimentando muita demanda por previsões de negócios de alta qualidade do que seus analistas eram capazes de fornecer. Em 2017, o *Facebook* lançou o *Prophet* ao público como *software* de código aberto.

O *Prophet* foi projetado para lidar de maneira ideal com tarefas de previsão de negócios, que normalmente apresentam qualquer um destes atributos:

• Dados de séries temporais capturados no nível horário, diário ou semanal com, idealmente, pelo menos um ano inteiro de dados históricos;

• Fortes efeitos de sazonalidade ocorrendo diariamente, semanalmente e/ou anualmente;

• Feriados e outros eventos especiais que não seguem necessariamente os padrões de sazonalidade, mas ocorrem de forma irregular;

• Dados ausentes e valores discrepantes;

• Mudanças significativas de tendência que podem ocorrer com o lançamento de novos recursos ou produtos, por exemplo;

• Tendências que se aproximam assintoticamente de um limite superior ou inferior.

Ademais o *Prophet* normalmente produz previsões de alta qualidade. Mas também é muito personalizável e acessível por analistas de dados sem experiência prévia em dados de séries temporais.

Essencialmente, o *Prophet* é um modelo de regressão aditiva. Isso significa que o modelo é simplesmente a soma de vários componentes (opcionais), como os seguintes:

• Uma curva de tendência de crescimento linear ou logístico;

• Uma curva de sazonalidade anual;

• Uma curva de sazonalidade semanal;

• Uma curva de sazonalidade diária;

• Feriados e outros eventos especiais;

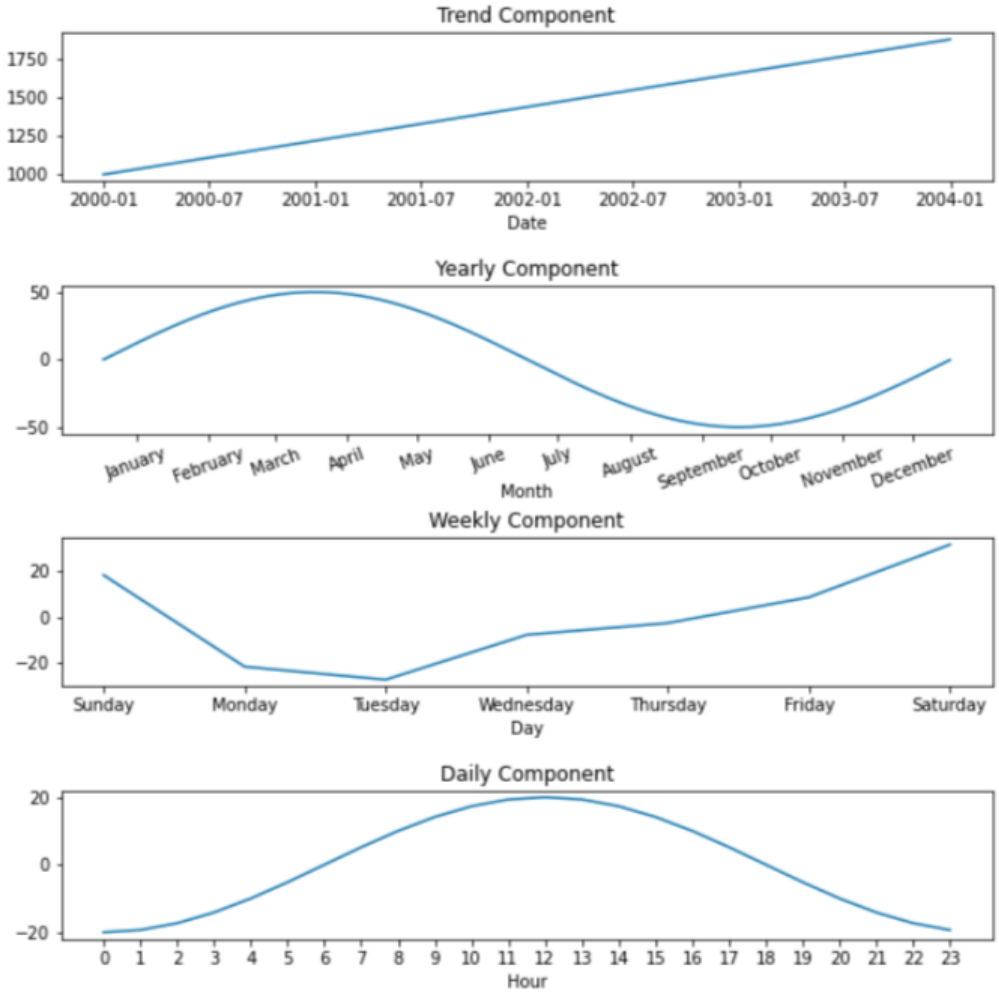
• Curvas de sazonalidade adicionais especificadas pelo usuário, como horária ou trimestral, por exemplo. (RAFFERTY, 2021)

Para dar um exemplo concreto, suponha que modele-se as vendas de uma pequena loja de varejo *online* ao longo de quatro anos, de 1º de janeiro de 2000 até o final de 2003. Observamos que a tendência geral está aumentando constantemente ao longo do tempo, de 1.000 vendas por dia para cerca de 1800 no final do período de tempo. Vê-se também que as vendas na primavera estão cerca de 50 unidades acima da média e as vendas no outono estão cerca de 50 unidades abaixo da média. Semanalmente, as vendas tendem a ser mais baixas na terça-feira e aumentam ao longo da semana, com pico no sábado. Finalmente, ao longo do dia, as vendas atingem o pico ao meio-dia e caem suavemente para o menor à meia-noite. A figura 5 demonstra a aparência dessas curvas individuais (observe as diferentes escalas do eixo x em cada gráfico):

O Prophet é escrito em *Stan*[8], isso tem várias vantagens. Ele permite que o *Prophet* otimize o processo de ajuste para que ele seja concluído em menos de um segundo. *Stan* também é compatível com *Python* e *R*, portanto, a equipe do *Prophet* pode compartilhar o mesmo procedimento de ajuste principal entre as duas implementações de linguagem. Além disso, usando estatísticas *Bayesianas*, *Stan* permite que o previsor crie intervalos de incerteza para previsões futuras para adicionar uma estimativa de risco de previsão baseada em dados. O *Prophet* consegue obter resultados típicos como as técnicas de previsão mais complicadas, mas com apenas uma fração do esforço. Ele tem algo para todos. O iniciante pode construir um modelo inicialmente satisfatório sem necessariamente entender os detalhes de como tudo funciona, enquanto o especialista pode se aprofundar no modelo, adicionando mais recursos e ajustando hiperparâmetros para obter um desempenho cada vez melhor. No caso de estudantes de engenharia de computação a plataforma se torna um grande objeto de estudo e de prática de programação em Python[9].

[8]Linguagem de programação probabilística desenvolvida pelo grupo do professor Andrew Gelman
[9]https://www.python.org/

Figura 5 – Gráficos de comportamento de previsões do *Facebook's Prophet*



Fonte: (RAFFERTY, 2021)

O *Prophet* é um *software* de código aberto, o que significa que a totalidade do código subjacente está disponível gratuitamente para qualquer pessoa inspecionar e modificar. Isso dá ao previsor uma grande quantidade de poder, pois qualquer usuário pode adicionar recursos ou corrigir *bugs*, mas também tem suas desvantagens. Muitos pacotes de *software* de código fechado, como o *Microsoft Word*[10], vêm empacotados em seu próprio arquivo de instalação independente com uma interface gráfica de usuário organizada para não apenas orientar os usuários durante a instalação, mas também permitir que eles interajam com o *software* após a instalação. O *Prophet*, por outro lado, é acessado por meio das linguagens de programação *Python* ou *R* e depende de muitas bibliotecas de código aberto adicionais. - o que é possivél mesmo o profeta sendo programado em Stan, uma vez que a linguagem de entrada de codigo não tem,

[10] *Software* desenvolvido pela empresa *Microsoft*

necessariamente, a ver com a linguagem de sua implementação. - Isso oferece grande flexibilidade, pois os usuários podem ajustar recursos ou até mesmo adicionar recursos totalmente novos para atender ao seu problema específico, mas vem com a desvantagem da usabilidade potencialmente difícil.

O *Prophet Forecasting Model*, criado pela equipe *Core Data Science do Facebook* é um *software* de código aberto para prever dados de séries temporais. O algoritmo do *Prophet* funciona bem no caso de conjuntos de dados com várias estações e descreve qualitativamente a sazonalidade nos dados. Taylor e Letham (TAYLOR; LETHAM, 2018) concluíram que o Modelo de Previsão do *Prophet* é projetado para permitir ajuste de parâmetro intuitivo sem o conhecimento necessário dos detalhes do modelo subjacente. Para criar o modelo, um modelo de série temporal decomponível é usado (HARVEY; PETERS, 1990). O modelo baseia-se na análise de três componentes. É baseado em um modelo aditivo onde o primeiro componente é a tendência do comportamento dos dados. O comportamento de dados não lineares é descrito pela sazonalidade em diária, semanal ou anual. O algoritmo observa os efeitos dos feriados, o que aumenta sua precisão. A estimativa final é formada conforme a equação 3 a seguir:

$$Y(t) = g(t) + s(t) + h(t) + Xt \tag{3}$$

Sendo que *g(t)* representa a tendência, *s(t)* representa a sazonalidade, enquanto *h(t)* representa o feriado efeito sobre o comportamento dos dados. O valor de Xt é um erro ou alteração nos dados que não estão contidos no modelo. A versão proposta do Modelo de Previsão do *Prophet* usa um modelo de crescimento saturante e um modelo linear por partes para a análise de tendência. A sazonalidade é baseada na série de *Fourier,* os feriados afetam significativamente o comportamento dos dados. Além das férias que repetido no mesmo período do ano, há um grande número de eventos que afetam o comportamento dos dados, e não há data fixa no ano. Além disso, os feriados variam de acordo com o país e são muitas vezes baseado no calendário lunar. O modelo do *Prophet* analisa feriados e feriados globais por país, e assume que os eventos são independentes. A implementação final é semelhante à análise de sazonalidade.

4 TRABALHOS CORRELATOS

Neste capitulo apresenta-se os trabalhos correlatos à este que foram escolhidos de acordo com a *string* de busca descrita no capítulo de metodologia. O principal objetivo dos trabalhos correlatos é ter uma visão geral de como o tópico aqui abordado é visto, discutido e analisados por demais pesquisadores, além de servir como fonte de conhecimento e embasamento para a escrita deste estudo.

4.1 Comparison analysis of Facebook's Prophet, Amazon DeepAR+ and CNN-QR algorithms for sucessful real-world sales forecasting

Desde a criação do *Facebook's Prophet* e do *DeepAR*, os algoritmos de predição têm atraído muita atenção. Zunic et al. (2021) apresenta a aplicação e comparação dos algoritmos para previsão de vendas em empresas de distribuição. é realizada uma comparação detalhada do desempenho de algoritmos sobre dados reais com diferentes comprimentos de histórico de vendas. Os resultados mostram que o *Facebook's Prophet* obtém melhores resultados para itens com histórico mais longo e vendas frequentes, enquanto o algoritmo da *Amazon* mostra superioridade para itens sem um longo histórico que raramente são vendidos.

No artigo de Zunic os autores explicam que existem muitos métodos de previsão baseados em dados de séries temporais, e este artigo, apresenta uma análise comparativa de três algoritmos modernos: *Facebook's Prophet, DeepAR+* da *Amazon* e *CNN-QR* da *Amazon* (Rede Neural Convolucional - Regressão Quantílica). A análise é baseada na comparação do desempenho dos algoritmos fornecidos, dependendo dos dados históricos de vendas disponíveis bem como na duração do próprio histórico para cada um dos itens analisados.

Os parâmetros usados, métricas, e o método de *backtesting* estão descritos no artigo. O artigo de (ZUNIC, et al, 2021) lista todas as etapas para a implementação bem-sucedida desses algoritmos em uso prático além de comparações entre os mesmos. No artigo os dados de vendas são observados e organizados de forma que sejam agregados mensalmente, devido à possibilidade de obtenção de previsões mais precisas e devido a necessidades práticas a previsão completa é feita no nível de item e organização do mesmo.

A *Amazon Web Services* (AWS) tem a capacidade de encontrar conexões entre

dados semelhantes. Com a estrutura de dados observada no artigo de Zunic, essa possibilidade é aumentada. Portanto, o algoritmo pode melhorar os resultados de previsão em uma loja com base nos resultados de previsão do mesmo item em outra loja. O algoritmo de previsão do *Facebook's Prophet* observa apenas a loja em si sem procurar conexões adicionais.

Durante a preparação do conjunto de dados, foi feito o pré-processamento e filtragem dos dados. Os últimos sete anos de vendas foram observados (quando possível). As vendas diárias foram agregadas, a média mensal e o preço foram calculados para cada tipo de item. Além disso, uma quantidade unitária foi determinada para cada item, e um preço adequado foi calculado. O preço de um item pode mudar ao longo dos meses, pode depender da demanda do mesmo, e de fatores externos, como Zunic apresenta no artigo. O conjunto de dados criado contém dados sobre quantidades e preços de itens. Os dados são divididos em dois grupos, onde o primeiro contém as seguintes colunas descritas na figura 6

- item - número de identificação do item;
- org - número de identificação da organização;
- data - a data de venda apresentada como o primeiro dia do mês;
- quantidade - quantidade de itens vendidos.

O segundo grupo contém dados sobre preços de itens. Este contém as seguintes informações:

- item - número de identificação do item;
- org - número de identificação da organização;
- data - a data de venda apresentada como o primeiro dia do mês;
- preço unitário - o preço de um item individual.

Um exemplo de dados em ambos os grupos é dado na figura a seguir:

Figura 6 – Exemplo dos dados de ambos grupos

Quantidade target_ts.csv	**item**	**Org**	Data	Quantidade
	3959294	1617388	2021-01-01	3718
Preço related_ts.csv	**item**	**Org**	Data	Preço Unitário
	3959294	1617388	2021-01-01	0.611830413

Fonte: (ZUNIC et al., 2021)

O conjunto de dados analisado contém os 50 itens mais significativos. Os dados observados representam 99 por cento das vendas da marca analisada. Quando determinada a significância de um item, observa-se o retorno financeiro total após a venda por item. Para a implementação do *backtesting*, é determinado o histórico de dados mínimos necessários. Com base na análise empírica, Zunic concluiu que 18 meses de história são suficientes para observar as mudanças necessárias nos dados (sazonalidade e tendência). Devido à necessidade de dados adicionais para testes, foram observados itens com história de pelo menos 24 meses. Assim, a quantidade de dados necessários em relação ao conjunto de dados descrito anteriormente foi reduzida, e o número de itens para teste foi aumentado.

O objetivo principal do artigo de Zunic é fornecer uma comparação dos três algoritmos de previsão mais populares *Facebook 's Prophet, DeepAR+* e *CNN-QR*. Devido à abordagem diferente na criação dos modelos mencionados, a comparação foi feita em duas fases. Na primeira fase, foram observados artigos com histórico mais longo, enquanto na segunda fase foram comparados algoritmos para artigos com histórico mais curto. Os itens são classificados como um histórico mais longo se o conjunto de dados contiver pelo menos 24 meses de dados de vendas.

A primeira fase visa testar a possibilidade de melhorar a aplicação do *Facebook's Prophet*, algoritmo descrito no capítulo 3. Na segunda fase de testes, o objetivo é ampliar o conjunto de itens com itens com um histórico mais curto. Assim, uma possível combinação dos métodos propostos alcançaria um resultado de qualidade para uma ampla gama de itens.

Zunic dividiu os itens em "itens com uma curta história de dados" e "itens com uma grande história de dados". Os resultados obtidos confirmam as vantagens de cada algoritmo. Os benefícios de cada uma das abordagens propostas é levantada, Zunic relata a diferença essencial na forma como o *Facebook 's Prophet* funciona comparado aos algoritmos da *Amazon*. Como observado, o *Facebook 's Prophet* analisa cada sinal de forma independente, enquanto os algoritmos da *AWS* criam um modelo único para todos os sinais e tentam encontrar interdependências. Portanto, os algoritmos da *AWS* mostram superioridade sobre os métodos clássicos apenas quando possuem um grande número de sinais sobre os quais quer se criar um modelo e no caso de artigos com um histórico curto.

Zunic observou que o modelo do *Facebook* apresenta superioridade no caso de itens que são vendidos frequentemente, em grandes quantidades e têm um longo histórico de vendas. Ao mesmo tempo, os algoritmos da *AWS* mostraram dominância em outros

casos.

Zunic conclui que isso leva à possibilidade de combinar esses métodos para criar previsões de vendas para uma ampla classe de itens. Com o uso combinado de algoritmos, é possível fazer previsões para itens com vendas frequentes e infrequentes, bem como para itens com uma longa e curta história, e em todos os casos para obter resultados de previsão satisfatórios.

Para Zunic, o conceito descrito baseado no uso dos algoritmos acima para previsão de vendas pode ser de grande benefício para aplicação no mundo real. O conceito pode lidar com uma gama diversificada de itens, itens com vendas frequentes e infrequentes e um histórico cada vez mais curto, o que não é o caso dos clássicos abordagens de previsão. O trabalho de Zunic tem por objetivo comprar as plataformas de previsão tal qual este referido estudo, assim, torna-se de alta relevância para este projeto uma vez que os objetivos dos artigos são os mesmos. Entretanto, a analise de projeção é sobre uma série temporal de valores de itens de loja, enquanto que o deste estudo é do valor do ouro. Mesmo com as diferenças em termos de projeção, ainda é um artigo correlato ao tema.

4.2 NeuralProphet: Explainable Forecasting at Scale

O presente artigo de Triebe et al. (2021) apresenta o *NeuralProphet*, um sucessor do *Facebook's Prophet*, que estabeleceu um padrão na indústria para estruturas de previsão explicáveis, escaláveis e fáceis de usar. Com a proliferação de dados de séries temporais, uma previsão confiável continua sendo uma tarefa desafiadora para decisões de negócios financeiros, principalmente. No artigo de Triebe et al. (2021), soluções híbridas são necessárias para preencher a lacuna entre os métodos clássicos e modelos de aprendizado profundo. Triebe et al. (2021) apresenta o *NeuralProphet* como um precursor de tal solução. O algoritmo *NeuralProphet* é uma estrutura de previsão híbrida baseada em *PyTorch* e treinada com métodos de aprendizado profundo, tornando mais fácil para os desenvolvedores estenderem a estrutura. Os dados são introduzidos com módulos de auto-regressão e covariáveis, que podem ser configurados como clássicos de regressão linear ou como Redes Neurais. Caso contrário, o *NeuralProphet* mantém a filosofia de design do *Facebook's Prophet* e fornece os mesmos componentes básicos deste modelo.

Um precursor de métodos híbridos, *Facebook's Prophet*, fornece um modelo interpretável que pode ser dimensionado para muitas aplicações de previsão. A previsão fornece automação completa para iniciantes e recursos de ajuste fino para especialistas.

Para Triebe et al. (2021) o *Facebook's Prophet* continua sendo um dos primeiros pacotes de previsão que um cientista de dados usará, e também um dos poucos que muitas vezes continuará a ser usado à medida que as necessidades do analista crescem. Ele fez séries temporais clássicas se tornarem previsões acessíveis e úteis para um amplo grupo demográfico. No entanto, suas limitações em torno dos principais recursos, como a falta de contexto local e extensibilidade, apresentou desafios para os usuários. A falta de contexto local, que é essencial para prever o futuro de curto prazo, restringiu a utilidade do *Facebook's Prophet* em aplicações industriais, isto porque o mesmo foi construído em cima de *Stan*, uma linguagem de programação probabilística, tornando difícil estender a original biblioteca de previsões. Este novo modelo de previsão proposto por Triebe et al. (2021), quando testado com grandes e defasados dados, tem desempenho melhor de 55 a 97 por cento, de acordo com o artigo aqui tratado.

Não se faz grande necessidade trazer mais detalhes sobre este estudo (soluções para problemas encontrados com os profetas já usados) no momento, entretanto este trabalho contribuiu para a construção do referencial teórico deste referido estudo. Ademais, é valioso conhecer estudos futuros como este em relação ao tópico de previsão de séries temporais.

4.3 Gold Price Forecasting Using LSTM, Bi-LSTM and GRU

Segundo YURTSEVER (2021) devido à natureza multifatorial e não linear do mercado de ouro, é difícil prever o preço do ouro. O preço do ouro é afetado por muitos fatores externos, como ambiente de mercado, crises econômicas, aumento do preço do petróleo, vantagens fiscais e taxas de juros. Portanto, modelos multivariados podem prever melhor o preço do ouro do que modelos univariados. Este estudo de YURTSEVER (2021) investigou os efeitos do preço do ouro, preço do petróleo bruto, índice de preços ao consumidor, índice cambial, índice da bolsa e indicadores de juros entre 2001 e 2021. Este artigo fornece algumas novas informações para previsão do preço do ouro com base em indicadores econômicos.

Segundo YURTSEVER (2021), as aplicações de aprendizado profundo têm sido amplamente utilizadas em várias áreas da ciência nos últimos anos. Nesse contexto, a área de finanças também começou a usar aplicativos de aprendizado profundo. A principal razão para isso é a natureza altamente volátil dos mercados financeiros e sua capacidade de serem afetados por muitas variáveis. Para as partes interessadas do mercado financeiro,

essa variabilidade é uma oportunidade e também um grande risco. Aplicativos de aprendizado profundo desenvolvidos nesse sentido devem proporcionar aos investidores a oportunidade de capturar oportunidades de mercado, por um lado, e, por outro lado, fornecer proteção contra riscos que possam surgir no mercado. Isso é importante porque o mercado de ouro está seguindo por governos e investidores individuais em todo o mundo.

Os resultados do artigo de YURTSEVER (2021) sugerem que LSTM, Bi-LSTM e GRU são todos estimadores satisfatórios para os dados ordinais utilizados neste estudo. Para YURTSEVER (2021) estudos futuros podem usar os modelos aqui avaliados para prever o preço do ouro, incluindo outras variáveis externas. Modelos de previsão podem ser feitos para preço do ouro, bem como outros metais preciosos, moedas e demais "valores" de séries temporais.

O artigo de YURTSEVER (2021) faz projeção do valor do ouro, o que é condizente com este estudo porém não utiliza as plataformas nem do *Facebook* nem da *Amazon*. Entretanto ilustra conceitos de métodos de projeção de série, o que faz que este seja um trabalho correlato muito útil para este estudo.

4.4 Gold Price Forecasting Using ARIMA Model

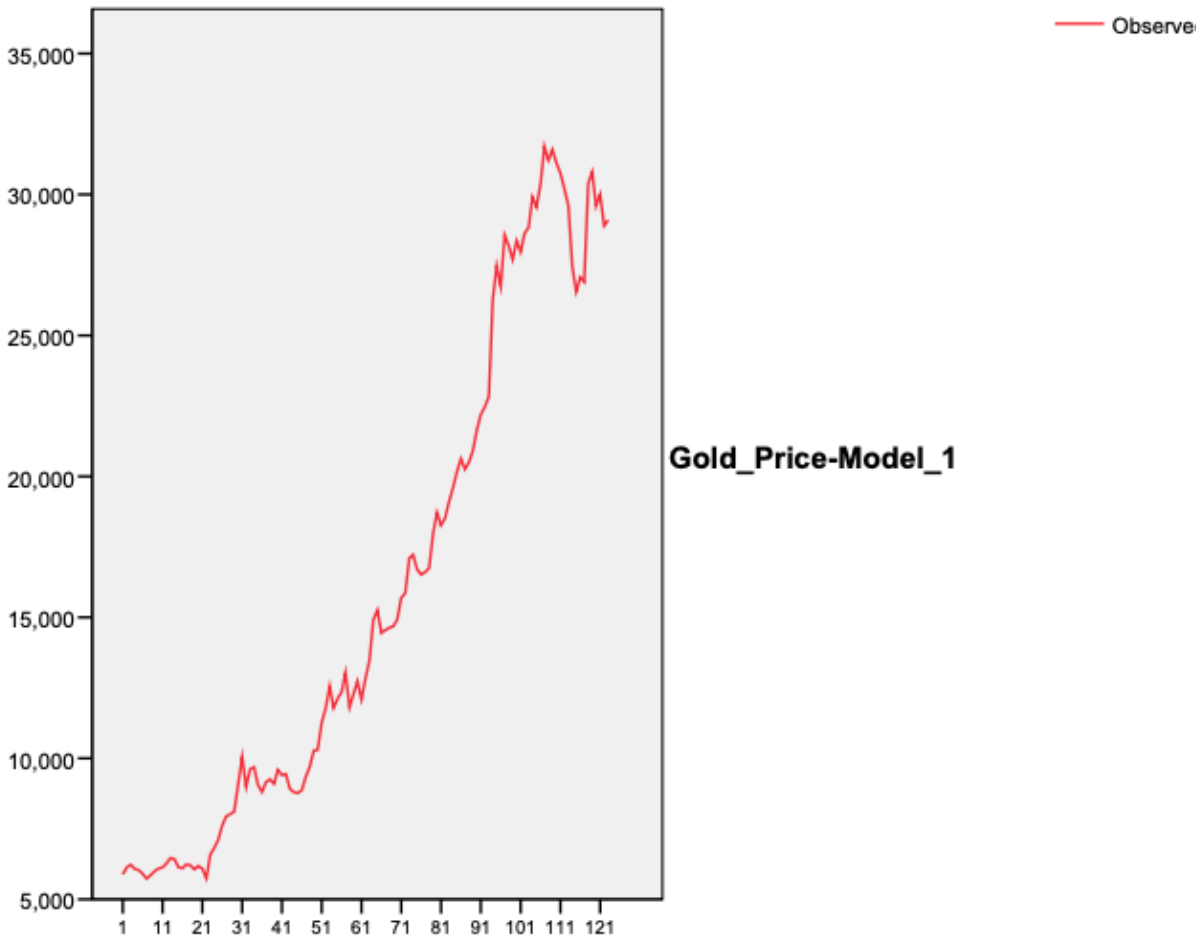
O estudo de Guha e Bandyopadhyay (2016) é baseado em dados mensais do preço do ouro a serem previstos usando o método ARIMA, Guha tem seu dataset retirado do MCX India (*Multi Commodity Exchange of India*) que é um banco de dados para operações financeiras situado na Índia, local da universidade do autor do estudo.

Após a coleta de dados no MCX, Guha testa se os dados se encaixam na descrição de uma série linear que é um dos requisitos para qualquer tipo de previsão. Guha demonstra por meio de gráficos a capacidade do modelo ARIMA na previsão de valores do ouro, como pode-se ver na figura 7, um gráfico de linha onde o eixo y se refere ao valor do ouro e o eixo x se refere ao tempo.

Guha conclui então que, com dez anos de dados do preço do onça do ouro, o modelo ARIMA traz resultados confiáveis para a predição do mesmo.

O artigo de Guha é extremamente correlato a este referido estudo uma vez que aborda conceitos de séries temporais e predições das mesmas usando o modelo ARIMA tal qual é abordado aqui, ademais, Guha utiliza um grande *dataset* do valor do ouro como este utilizado neste referido estudo.

Figura 7 – Resultados da predição com modelo ARIMA



Fonte: (GUHA; BANDYOPADHYAY, 2016)

4.5 Forecasting Crude Oil Price Using ARIMA and Facebook Prophet Within Machine Learning

Segundo (AZIZ; BARAWI,) a previsão do preço do petróleo tem recebido muita atenção dos praticantes e pesquisadores, mas continua sendo um tópico difícil por causa de sua dependência de uma variedade de fatores, incluindo o ciclo econômico, relações internacionais, geopolítica e assim por diante. Os autores apresentam previsões para o preço do petróleo utilizando o modelo ARIMA e o Facebook's Prophet além de aplicar a suavização de Savitzky Golay para um resultado mais livre de ruído, isto é, um resultado mais linear e acurado da previsão do valor do ouro. Este trabalho apresenta vasta revisão de literatura além de grande revisão teórica a respeito do tema, e ainda que utilize outro ativo financeiro como valor de previsão, é um trabalho extremamente correlato por utilizar técnicas de *machine lerning* e a plataforma *Facebook's Prophet*.

Tabela 1 – Análise comparativa entre os trabalhos correlatos

Proposta	**Característica**	**Este estudo**
Zunic et al. (2021)	- Uso de redes neurais; - Plataforma *DeepAR*.	- Uso de redes neurais; - Plataforma *DeepAR*; - *Machine Learning*; - *Facebook's Prophet*.
Triebe et al. (2021)	- Redes neurais; - Modelo ARIMA; - União de plataformas.	- Redes neurais; - Modelo ARIMA.
YURTSEVER (2021)	- Predição do ouro; - Redes Neurais; - LSTM, Bi-LSTM, GRU.	- Predição do ouro; - Redes Neurais.
Guha e Bandyopadhyay (2016)	- Predição do ouro; - Modelo ARIMA.	- Predição do ouro; - Modelo ARIMA; - Redes Neurais.
Aziz e Barawi (2021)	- Modelo ARIMA; - *Facebook's Prophet*; - *Machine Learning*.	- Modelo ARIMA; - *Facebook's Prophet*; - Machine Learning; - Redes Neurais.

Fonte: Autora (2022).

A tabela 1 traz a comparação entre os trabalhos apresentados nessa seção com o que é desenvolvido nesse referido estudo.

5 DESENVOLVIMENTO

Neste capítulo serão apresentados etapas do desenvolvimento do projeto proposto, os *softwares* necessários para a instalação do *Facebook's Prophet* e do *Amazon DeepAR* são apresentados. Além disso, com base nos capítulos 3 e 4 expõe-se o processo de tratamento de dados feito para a plataforma.

A fim de cumprir a proposta principal do projeto, que é prever o valor do onça do ouro por meio de ferramentas de IA relativas a predição de séries temporais, as ferramentas *Facebook's Prophet* e *Amazon DeepAR* foram instalados. Primeiramente foi treinada a programação na linguagem *python* uma vez que esta é utilizada pelas bibliotecas de previsão disponibilizadas pelas empresas. Logo após foi instalada a IDE *Anaconda*, para codificar em *python* e usar as bibliotecas das plataformas de predição. Ademais, foi instalado o *PyCharm*, como IDE de codificação.

Ao finalizar as instalações dos programas necessários iniciou-se o processo de programação, com o propósito de teste da plataforma do *Prophet*. Neste processo devido ao tempo excessivo de execução, decidiu-se usar o *Google Collaboratory*[11], uma IDE *online* de *python* e demais linguagens que não leva o computador à um extremo de sobrecarga e desempenha de forma satisfatória - realizando as mesmas atividades que a plataforma Anaconda - as atividades necessárias para os testes deste trabalho.

Já no *Google Collaboratory*, foram instaladas as bibliotecas referentes ao *Facebook's Prophet* no código e logo em seguida foi passado o arquivo com os dados de valor e data da criptomoeda ETH (*Ethereum*)[12]. Neste primeiro momento, está sendo utilizada esta moeda para testes de código e bibliotecas. A figura 8 mostra o trecho de código da importação das bibliotecas e dos dados da ETH. Os relativos parâmetros das função da biblioteca do *Prophet* são passados, em sequência na figura 9, logo em seguida, conforme a figura 10 está o código de impressão dos gráficos de previsões desejadas.

[11] Disponível em: https://colab.research.google.com/
[12] *Token* ativo do *blockchain* Ethereum

Figura 8 – Passagem da bibliotecas do *Facebook's Prophet* e do arquivo com os dados

```
import pandas as pd
import numpy as np
from fbprophet import Prophet
from fbprophet.diagnostics import cross_validation
from fbprophet.diagnostics import performance_metrics
from fbprophet.plot import plot_cross_validation_metric
from fbprophet.plot import add_changepoints_to_plot
import matplotlib.pyplot as plt


df = pd.read_csv('eth.csv')
```

Fonte: Próprio Autor.

Figura 9 – Passagem dos parâmetros para as funções do *Prophet*

```
df.plot()
plt.savefig('eth_pred0.jpg')
plt.show()
df['y'] = np.log(df['y'])

m = Prophet(growth='linear', daily_seasonality=True)
m.add_seasonality(name='monthly', period=30.4, fourier_order=7)
m.fit(df)

future = m.make_future_dataframe(periods=12, freq='M')

fcst = m.predict(future)
```

Fonte: Próprio Autor.

Figura 10 – Código da impressão dos gráficos de previsões

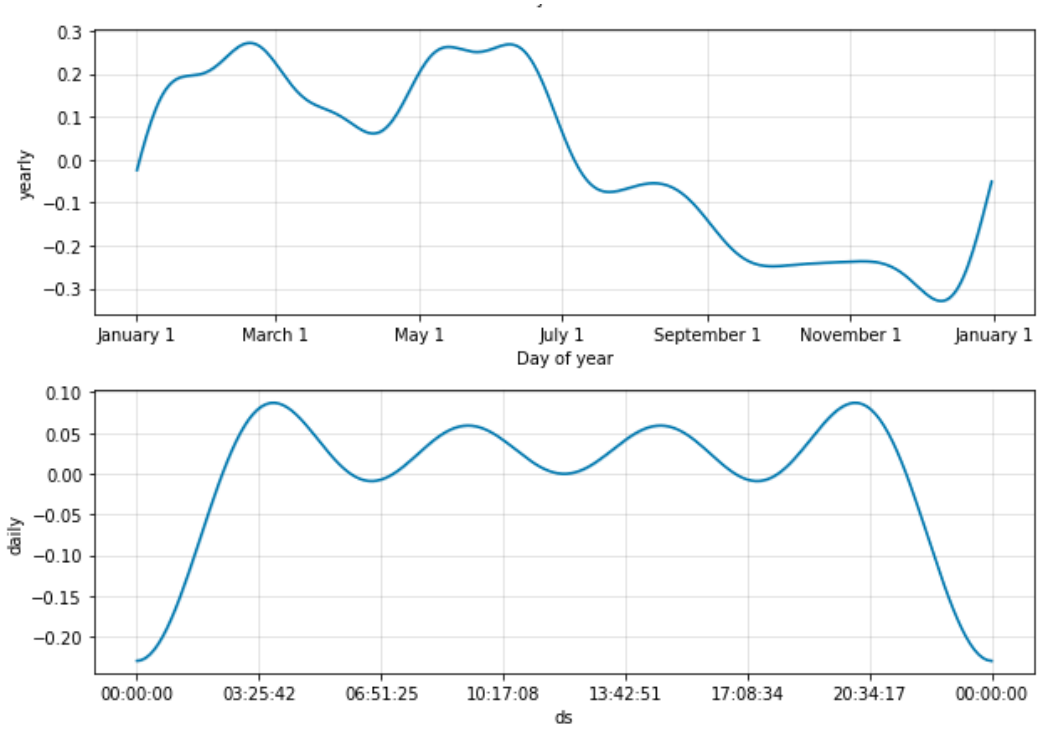
```
m.plot(fcst);
plt.savefig('eth_pred1.jpg')
m.plot_components(fcst);
plt.savefig('eth_pred2.jpg')
plt.show()
```

Fonte: Próprio Autor.

Nos dois primeiros gráficos apresentados, figura 11, que foram impressos pelo *Prophet* consegue-se visualizar como se comporta a previsão do valor do ETH anual e diária.

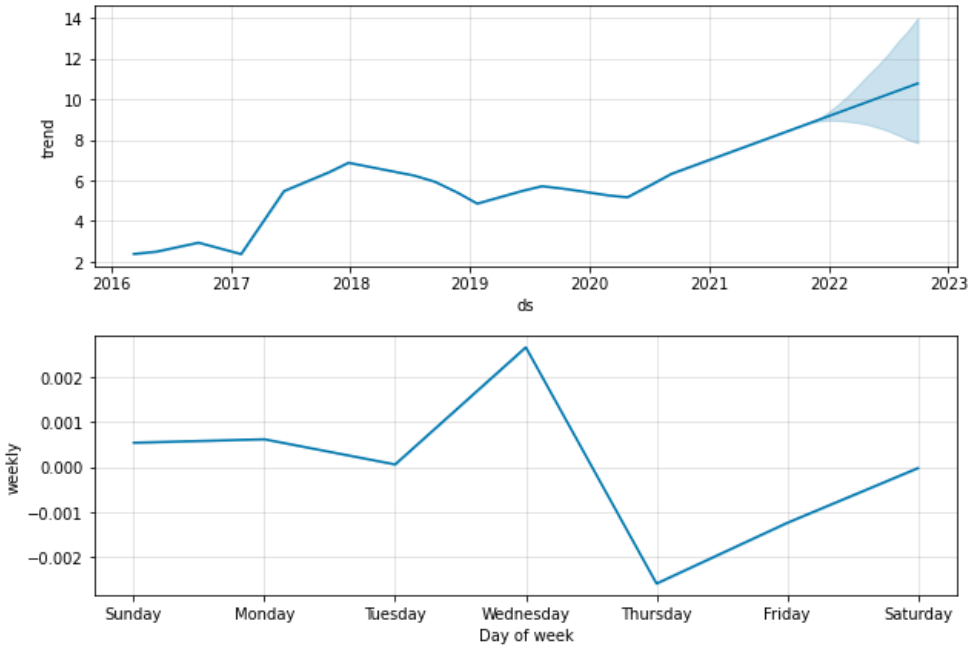
Ademais, nos dois outros gráficos da figura 12, consegue-se visualizar como se comporta a previsão do valor do ETH na curva de tendência e durante a semana.

Figura 11 – Gráfico da previsão anual e diária da ETH



Fonte:Próprio Autor.

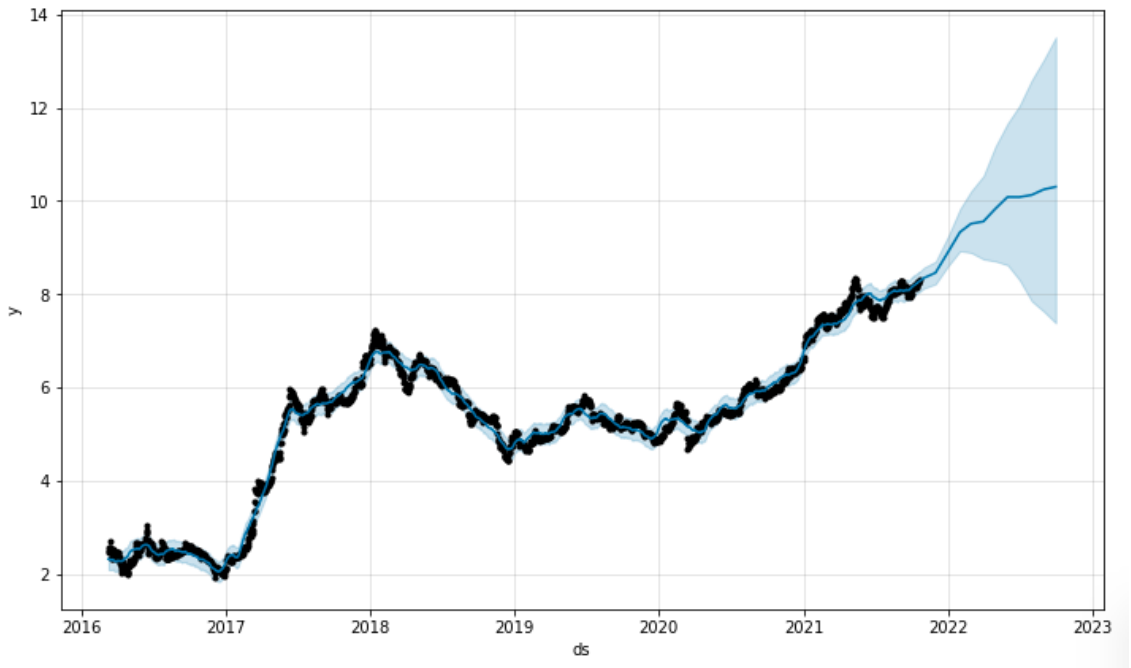
Figura 12 – Gráfico da comportamento da previsão para *trend* e semanal da ETH



Fonte:Próprio Autor.

Por ultimo, pode-se observar na figura 13 a regressão do preditor em preto, a curva real em azul e no final da curva a previsão para 2022 e 2023.

Figura 13 – Gráfico da comportamento da moeda ETH de 2016 à 2023



Fonte: Próprio Autor.

5.1 Previsão do valor do ouro

Por outro lado, seguindo a proposta inicial deste referido estudo, deu-se inicio à previsão do valor ouro, que para tal meta foram desenvolvidas as seguintes atividades. Para que seja possível prever o valor de qualquer *token* ativo é necessário, como dito anteriormente nessa sessão, um *dataset* mais completo deste *token*. Para o ouro, uma vez que é um dos mais antigos ativos financeiros, existe diversos *datasets* sendo a maioria extremamente vasta.

5.1.1 Desenvolvimento dos *softwares* utilizados

Quando fala-se em desenvolvimento de *software* existem diversas metodologias renomadas que podem ser adotas, o fator de escolha de qual metodologia adotar para o desenvolvimento do *software* é compreender claramente qual objetivo o cliente quer atingir com aquela aplicação, isto é, o que o cliente espera ser capaz de realizar, compreender ou até mesmo quantificar com aquele *software*. Neste referido estudo, o objetivo principal é identificar se é possível ou não prever o valor do ouro por meio das aplicações *Facebook's Prophet* e *Amazon DeepAR*.

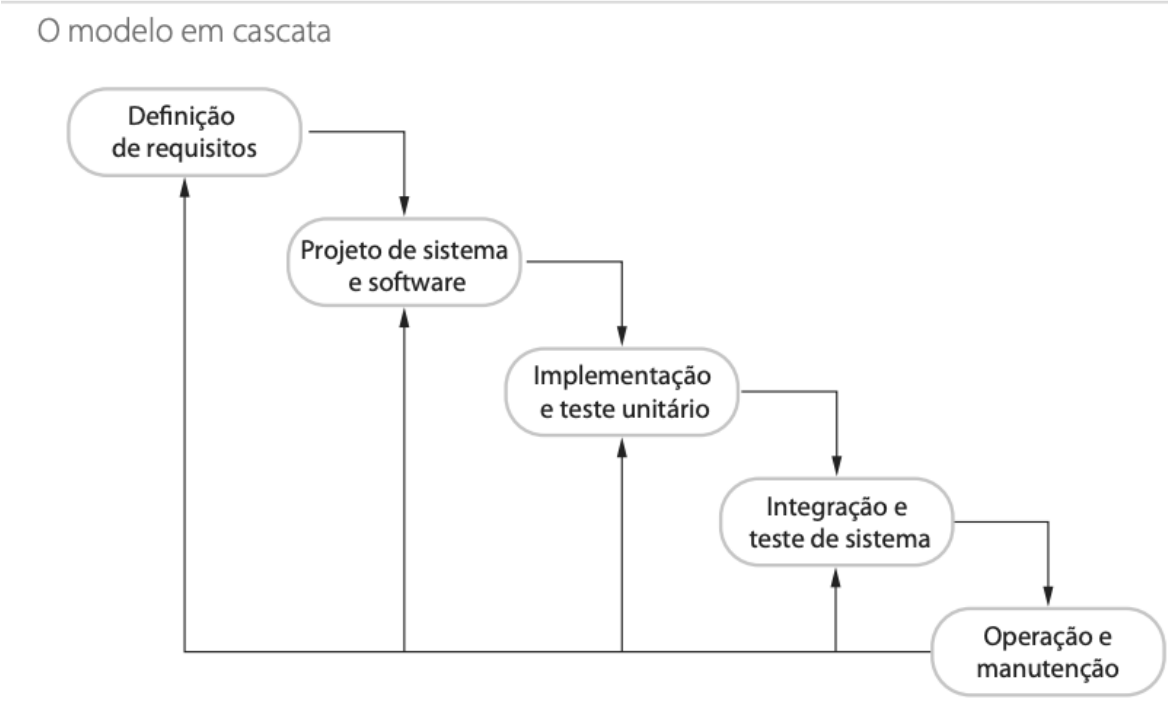
Para começar a planejar o desenvolvimento dos programas aqui utilizados, seguindo os ideais da engenharia de *software*, primeiramente pensou-se a respeito de qual modelo de *software* utilizar. Um modelo de *software* diz respeito ao processo desta aplicação, cada modelo tem suas particularidades representando uma perspectiva particular dos processos daquela aplicação assim fornecendo informações parciais sobre o mesmo, como suas atividades e as sequencias das mesmas.

Neste referido estudo, optou-se pelo modelo em cascata para o desenvolvimento dos *softwares*, este modelo considera cada etapa de desenvolvimento como uma etapa individual do desenvolvimento, geralmente esse modelo apresenta três principais etapas, sendo elas: a análise e definição de requisitos, projetos de sistema e *software* e, implementação e teste unitário. Na etapa de análise e definição de requisitos os serviços que o *software* oferecerá são definidos de acordo com o que o usuário explicita que necessita. Já na etapa de projetos de sistemas e *softwares* são definidas as necessidades de *hardware* e *software* para a execução do projeto, essa etapa envolve a identificação e descrição das tarefas fundamentais esperadas do *software*. Na etapa implementação e teste unitário o *software* é de fato desenvolvido como um todo e os testes unitários existem para garantir que este atenda, de fato, os objetivos do cliente. Já na etapa de integração e testes de sistema cada unidade do programa é executada e testada juntas como um programa completo, para que se tenha a certeza que os requisitos sejam atendidos. Por fim, a etapa de operação e manutenção tende, na maioria das vezes, a ser a etapa mais longa, uma vez que todo *software* tem necessidade constante de evolução e manutenção. A figura 14 é extraida do livro Engenharia de *Software* de Sommerville (2011), que traz, em forma de diagrama a ideia do modelo em cascata, já na figura 15 é demonstrado como o modelo em cascata foi implementado no desenvolvimento dos *softwares* desde referido estudo.

Logo após a definição do modelo de *software* a ser utilizado, definiu-se as especificações de *software*, também conhecidas como engenharia de requisitos, esse processo é a compreensão dos serviços que o *software* deve oferecer bom como os requisitos que este deve cumprir para sua total funcionalidade. O processo de engenharia de requisitos possui 4 fases, sendo estas: estudo de viabilidade, elicitação e análise de requisitos, especificação de requisitos, validação de requisitos.

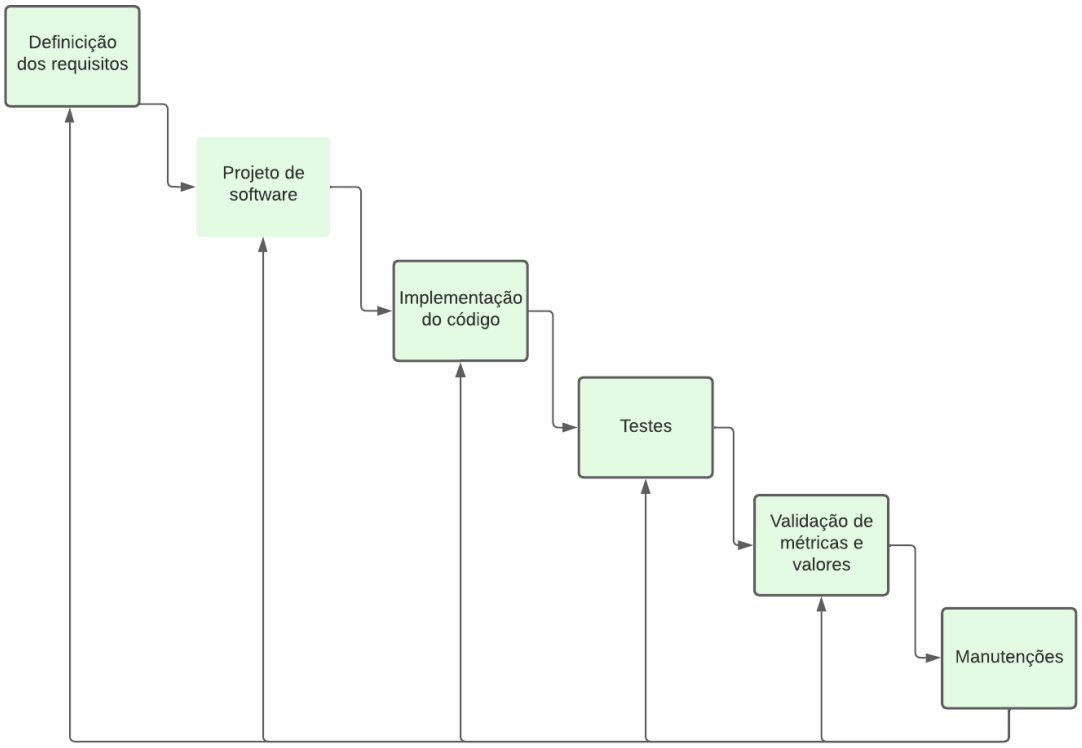
O processo de estudo de viabilidade é, tal como o nome já indica, um estudo se o *software* a ser desenvolvido é viável em termos de *hardware* e *software* para satisfazer a necessidade do cliente. O resultado desta etapa indica se é possível ou não avançar para as demais etapas. A fase de elicitação e análise de requisitos é dada via

Figura 14 – Modelo em cascata exposto no livro Engenharia de *software* de Sommerville (2011)



Fonte: (SOMMERVILLE, 2011)

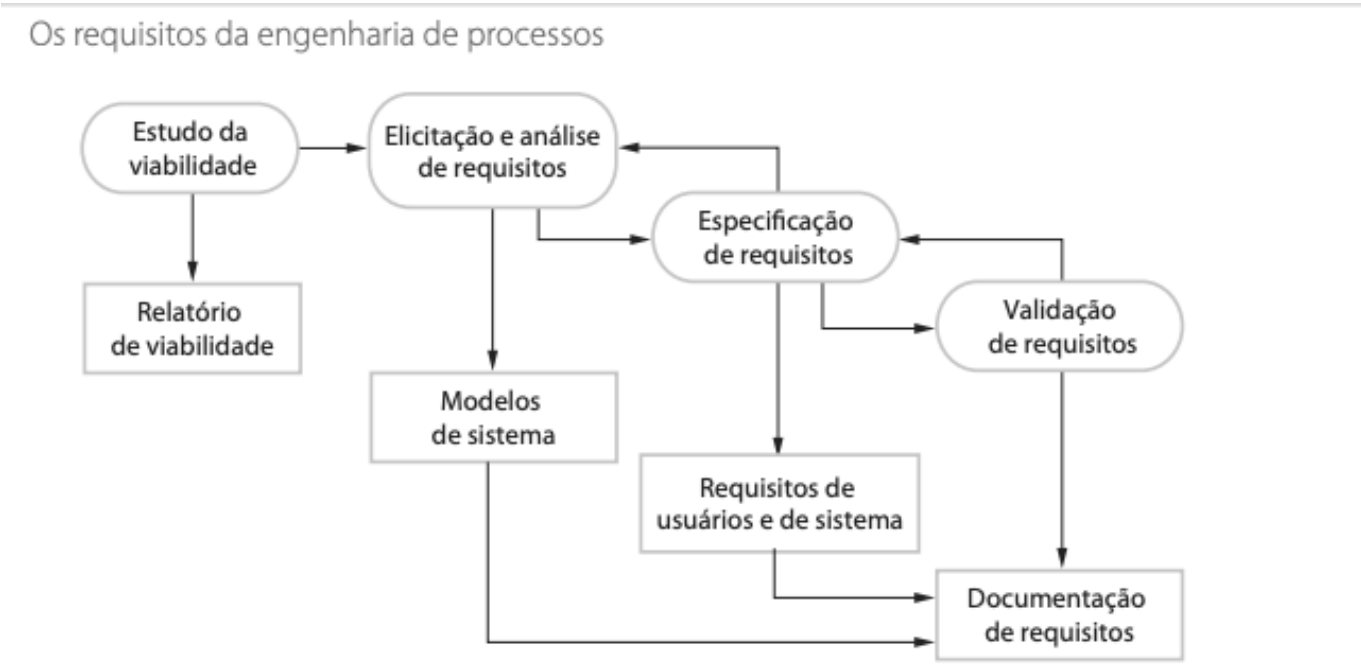
Figura 15 – Modelo em cascata



Fonte: Próprio Autor.

observação de sistemas já existentes similares ao que o cliente deseja, com o intuito de desenvolver protótipos para se ter uma ideia do que será feito para o cliente. Já na etapa de especificação de requisitos é onde ocorre a "tradução" do que o cliente espera que o *software* faça para requisitos que devem ser cumpridos durante a elaboração do *software*. Por fim, a etapa de validação de requisitos é quando a equipe avalia os requisitos a fim de entender se estes são realistas à realidade da capacidade de desenvolvimento e suficientes para atingir o objetivo proposto pelo cliente. Sommerville (2011) traz em seu livro a imagem 16 que exemplifica um processo de elaboração de engenharia de requisitos, o processo descrito na imagem foi o processo utilizado na engenharia de requisitos deste estudo.

Figura 16 – Engenharia de requisitos

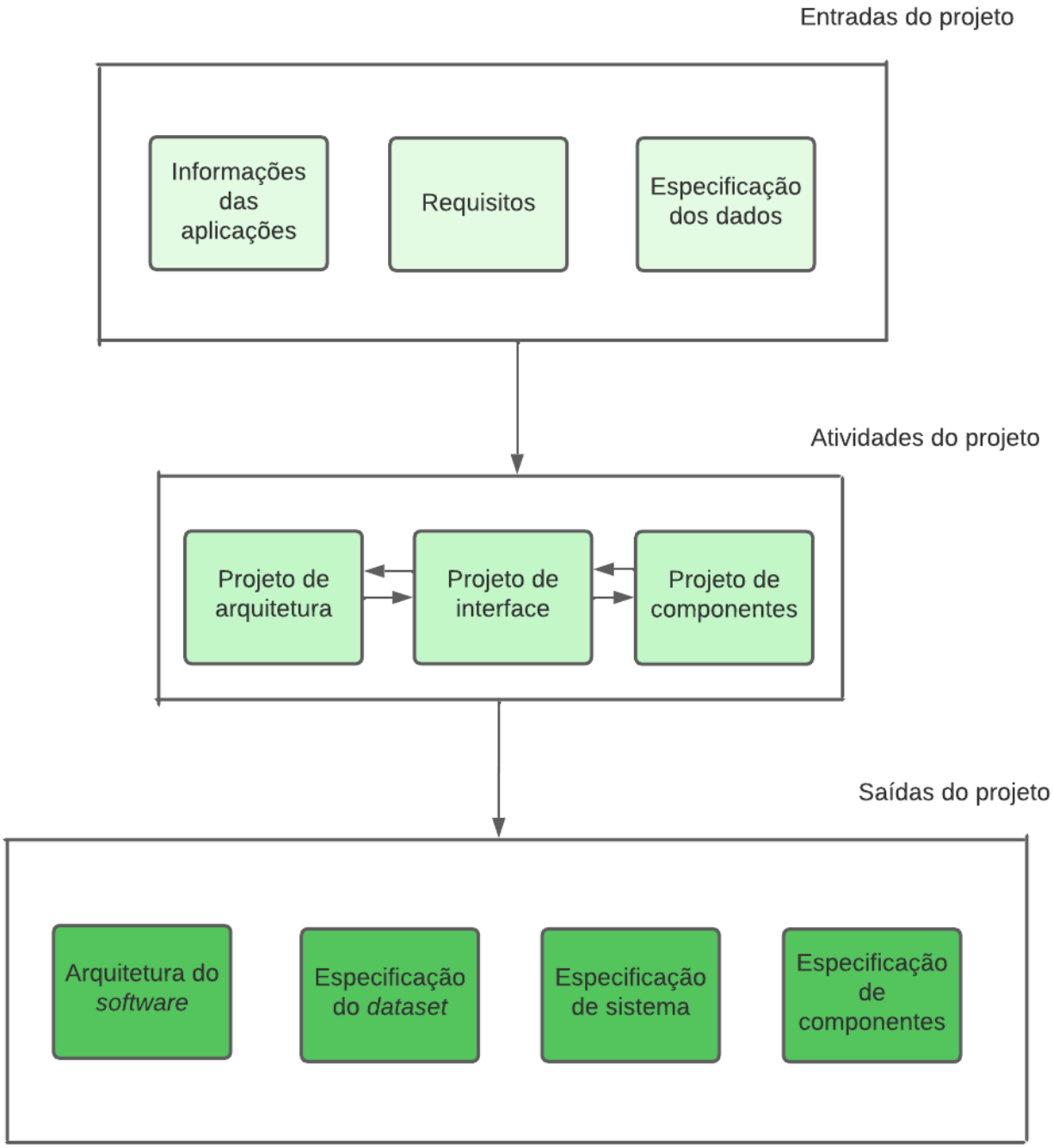


Fonte: (SOMMERVILLE, 2011).

Após os dois processos descritos acima, passou-se para a implementação do *software*, este foi o momento em que todas as etapas e processos trazidos anteriormente agiram para a melhor execução do desenvolvimento do *software*. Sendo o objetivo deste estudo a implementação de dois *softwares* distintos, porém ambos com o mesmo objetivo, optou-se por traçar um processo de projeto básico para ambos *softwares* primeiramente e depois um processo de projeto individual para cada aplicação baseado nas necessidades, requisitos e tecnologias utilizadas por cada uma destas. A imagem 17 demonstra como foi realizado o projeto de *software* abstrato para ambas aplicações.

Ademais, criou-se então a tabela de requisitos para o *dataset* a ser utilizado neste

Figura 17 – Processo de projeto



Fonte: Próprio Autor.

estudo e de cada uma das aplicações a serem desenvolvidas para a predição do valor do ouro que será demonstrada nas próximas subseções deste projeto. Foi definida a formatação da tabela de requisitos dos sistemas, esta contém o *id* do requisito, a descrição do requisito e a prioridade do mesmo. as tabelas de cada um dos *softwares* e do *dataset* foram baseados no processo de projeto de cada uma das aplicações. O exemplo de

formatação da tabela se encontra na tabela 2.

ID do requisito	**Descrição do requisito**	**Prioridade**
Aqui o *id* do requisito	Aqui uma breve descrição do requisito	Aqui a prioridade do requisito

Tabela 2 – Exemplo da formatação da tabela de requisitos de sistema

Fonte: Próprio Autor.

Por fim, com o intuito de validar os resultados encontrados usou-se métricas de erros. Métricas de erro são uma forma de quantificar o desempenho de um modelo e fornecem uma maneira para o usuário comparar quantitativamente diferentes modelos, estas são comumente utilizadas em previsões de séries temporais para qualificar a acurácia da previsão. Existem diversas métricas de erro, neste estudo optou-se por usar três das mais conhecidas, sendo elas MAPE (erro percentual absoluto médio), MAE (erro absoluto médio) e RMSE (raiz quadrática média dos erros entre valores observados e as predições).

A métrica de erro MAPE representada na equação 4 foi fundamental para a analise desse trabalho uma vez que esta é usada quando diferentes modelos de previsão ou conjuntos de dados precisam ser comparados porque seu resultado é o valor percentual do erro da ferramenta de predição. Já o MAE demonstrada na equação 5 é o cálculo do erro absoluto médio, que calcula o residual de cada ponto, e residuais negativos e positivos não se anulam, após obter-se o somatório, faz-se a média dos residuais. Ademais o RMSE representado na equação 6 é útil quando se tem uma ampla gama de valores, tal como se tem no *dataset* utilizado neste estudo.

$$MAPE = \frac{1}{n} = \sum_{t=1}^{t=n} \frac{|y' - y''|}{y} * 100\% \tag{4}$$

Sendo y' o valor previsto, y'' o valor real e y o valor total de registros de valores.

$$MAE = \sum_{i=1}^{N} \frac{\rho i - \tau i}{n} \tag{5}$$

Sendo ρi o valor real, τi o valor encontrado pelo previsor e n o valor total de registros de valores.

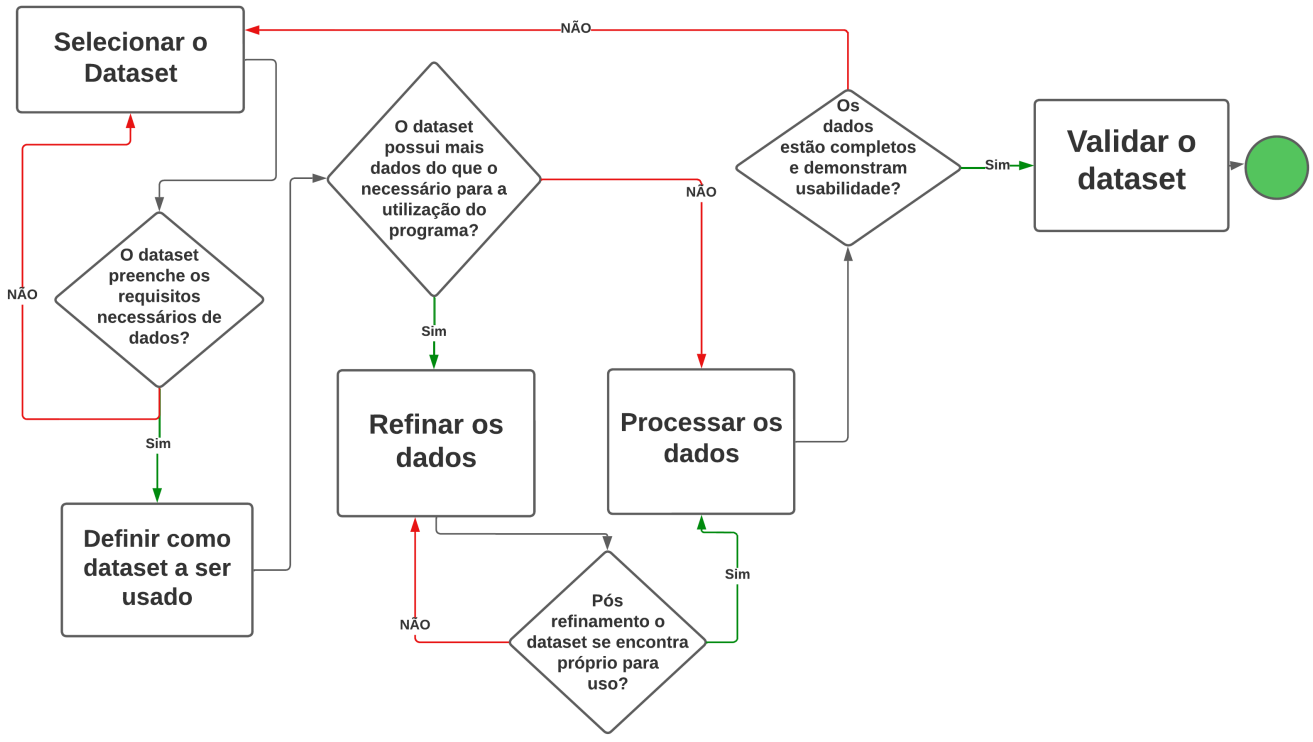
$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (\alpha - \beta)^2}{N}} \tag{6}$$

Sendo α o valor encontrado pelo previsor, β o valor real do moeda e n o valor total de registros de valores.

5.1.2 *Dataset*

A escolha do *dataset* é extremamente importante uma vez que este é a principal fonte de dados pro trabalho que foi desenvolvido, o conjunto de dados traz diversas informações históricas de determinado objeto de estudo, no caso deste referido trabalho, o *dataset* escolhido traz informações a respeito da variação do valor do ouro ao longo do tempo. Uma vez que estes grandes acervos de dados são produzidos por diversas pessoas, empresas, bancos etc, não há um padrão em conjuntos de dados, alguns podem conter somente a informações triviais e relativas a busca feita pelo usuário na hora da escolha do *dataset*, como outros podem ter diversas informações extras, conversões, métricas e valores que se tornam dispensáveis para a pessoa que vai trabalhar com o *dataset*.

Figura 18 – Fluxograma referente ao refinamento do *dataset*



Fonte: Próprio Autor.

Após o levantamento dos requisitos e a criação da tabela de requisitos do *dataset*, partiu-se para a etapa de, dentre os vários conjuntos de dados encontrados selecionar o

Figura 19 – *Dataset* filtrado

ds	y
2000-01-04	281.0
2000-01-05	283.2
2000-01-06	281.4
2000-01-07	281.9
2000-01-10	281.7
2000-01-11	283.4
2000-01-12	282.7
2000-01-13	284.1
2000-01-14	283.9
2000-01-18	288.6
2000-01-19	289.8
2000-01-20	288.8
2000-01-21	289.3
2000-01-24	288.0
2000-01-25	286.3
2000-01-26	286.3
2000-01-27	287.1
2000-01-28	287.1

Fonte: Próprio Autor.

Figura 20 – *Dataset* bruto

Date	# Open	# High	# Low	# Close
2000-01-04	281.0	281.0	281.0	282.7
2000-01-05	283.2	283.2	283.2	281.1
2000-01-06	281.4	281.4	281.4	281.4
2000-01-07	281.9	281.9	281.9	281.9
2000-01-10	281.7	281.7	281.7	281.7
2000-01-11	283.4	283.4	283.4	283.4
2000-01-12	282.7	282.7	282.7	282.7
2000-01-13	284.1	284.1	284.1	284.1
2000-01-14	283.9	283.9	283.9	283.9

Fonte: Próprio Autor.

que melhor atenderia às necessidades do trabalho. Portanto, aquele que preencheu todos os requisitos e foi escolhido para o trabalho passou por uma filtragem, refinamento e processamento de dados, este processo se deu de acordo com o fluxograma demonstrado na figura 18. O fluxograma de refinamento apresenta processos e critérios de decisões a

serem tomados a fim de chegar numa coleção de dados mais limpa possível e totalmente útil ao trabalho.

Os dois primeiros processos do fluxograma de refinamento do *dataset* diz respeito diretamente à tabela de requisitos exposta previamente nesta seção. Ademais, após a definição de qual *dataset* utilizar, verificou-se se este possuía mais informações do que as necessárias para que os modelos de previsão. No caso do conjunto de dados escolhido, demonstrado na figura 20 havia dados que não eram necessários para o trabalho, como os dados de abertura da cotação do dia, demonstrados na tabela na coluna " Open", o valor mais alto de cotação que foi atingido naquela data, demonstrado na coluna " High", a cotação mais baixa do dia, demonstrada na coluna " Low", e o valor de fechamento da cotação diária, demonstrada na coluna " Close".

De acordo com o fluxograma de refinamento, por esse conjunto de dados ser mais abundante que o necessário, este passou pelo processo de refinar os dados para uso, em que passou-se a considerar do conjunto de dados apenas as datas e os valores de abertura da cotação. Ademais, processou-se os dados a fim de formatar da forma necessária para os modelos de previsão lerem o *dataset* e por fim, validou-se o mesmo. O *dataset* filtrado e pronto para uso é exposto na figura 19. Para a escolha do *dataset* deste estudo criou-se critérios de escolha que o conjunto de dados deveria cumprir para ser aceito como um possível conjunto de dados a ser utilizado. Para isso, foi levantado requisitos que este *dataset* deveria cumprir, como mostra a tabela 3.

Requisito	**Requisição**	**Importância**
Requisito 01	O *dataset* possui o valor do ouro em função do tempo?	Alta
Requisito 02	A variação de tempo no *dataset* é diária?	Alta
Requisito 03	Os valores relacionados ao preço do ouro são consistentes?	Alta
Requisito 04	Há espaços em branco ou dias sem registro (datas faltando)?	Alta
Requisito 05	O *dataset* vem de uma fonte confiável?	Alta

Tabela 3 – Tabela de requisitos do *dataset*

Fonte: Próprio Autor.

5.1.3 Configuração do modelo *DeepAR*

O modelo da *Amazon Web Services*, *Amazon DeepAR*, como dito no capítulo de referencial teórico deste estudo, é um modelo mais complexo comparado ao *Facebook's Prophet* quando se trata de usabilidade da plataforma e modelos de previsão. O *DeepAR* é um modelo de previsão de séries temporais programável na linguagem *Python*, sua maneira de realizar a previsão da série temporal que é integrada ao modelo via *dataset* é por meio de RNN (redes neurais recorrentes), isto significa que é necessário, durante a programação do código criar as redes neurais bem como seus parâmetros de funcionamento e logo após treiná-las a fim de conseguir extrair dalí a melhor previsão possível.

Unindo conceitos do capítulo de Referencial Teórico, da subseção de Desenvolvimento de *Software* Utilizados e modelo da *Amazon Web Services*, *Amazon DeepAR* sendo bem conhecido e entendido, criou-se o processo de projeto do modelo e a tabela de requisitos do mesmo. A figura 21 ilustra o projeto de processo de *software* referente à aplicação *DeepAR*.

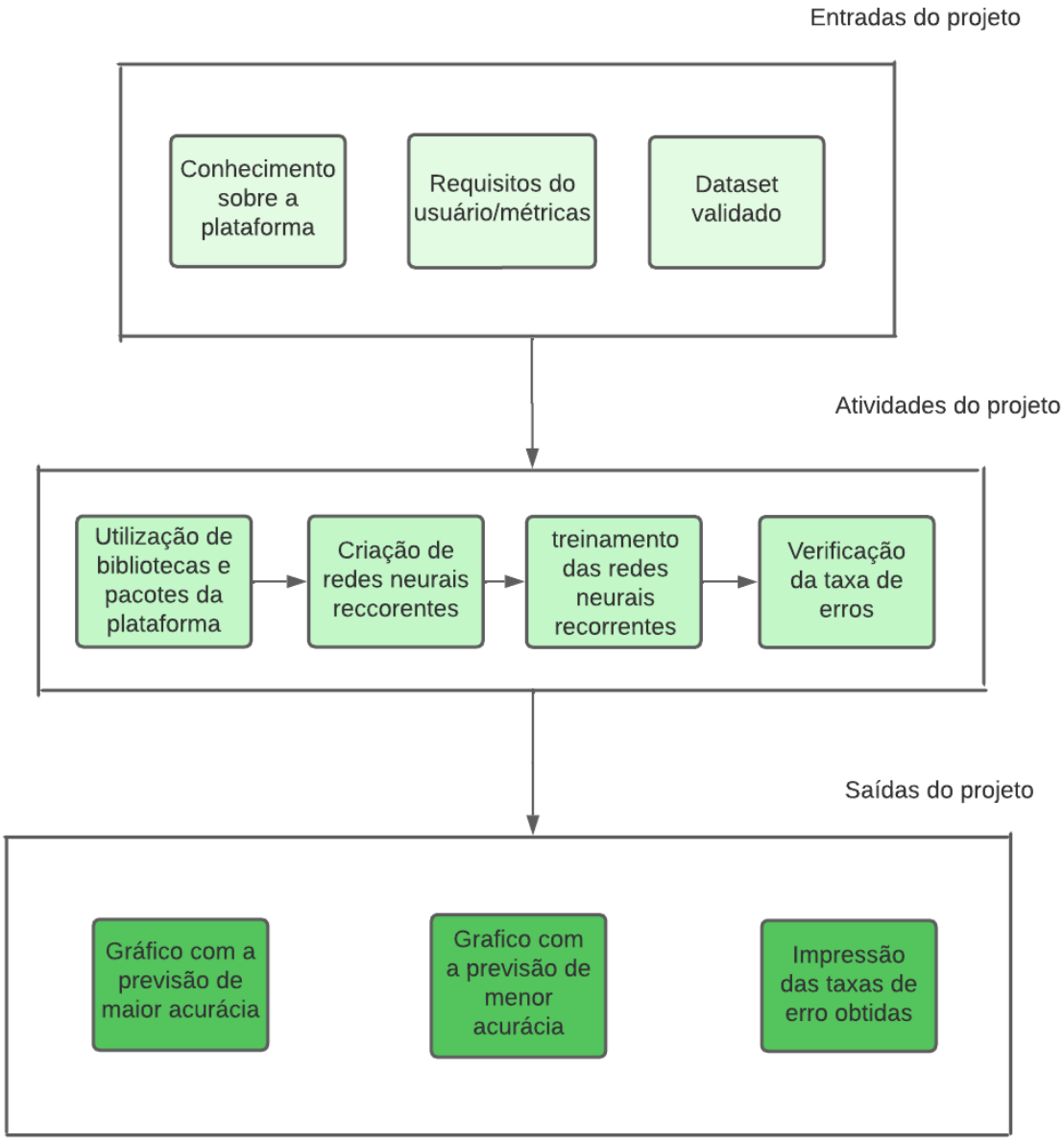
Tendo o projeto de de processo de *software* pronto, criou-se então a tabela de requisitos, esta é demonstrada na tabela 4

Para a codificação em *Python*[13], foi utilizada a *API Apache MXNet*[14], está é uma *API* bem flexivel e vasta, esta contem o principal pacote que foi utilizado além de suas respectivas bibliotecas. Diversas bibliotecas e pacotes da linguagem *Python* foram utilizadas na construção do algoritmo *DeepAR*, destaca-se, dentre estes, o uso do pacote *GluonTS*[15]. *GluonTS* é um pacote do *Python* para modelagem probabilística de séries temporais, focado em modelos de aprendizado profundo. *GluonTS* é instalado via terminal e requer minimamente a versão 3.6 do *Python* ou mais recentes, dentro deste pacote existe diversas bibliotecas que são essenciais para a predição de séries temporais, tais quais foram utilizadas neste estudo: "to_pandas"[16], "PandasDataset", "DeepAREstimator" e "Trainer". Ademais, a biblioteca "json" foi utilizada para calculo de erro do algoritmo de predição. Estas são expostas na figura 22.
Alexandrov et al. (2019)

Grande parte da programação do algoritmo do modelo *DeepAR* girou em torno de

[13]https://www.python.org/
[14]https://mxnet.apache.org/versions/1.9.1/
[15]https://ts.gluon.ai/stable/
[16]https://pandas.pydata.org/

Figura 21 – Projeto de *software* da plataforma *DeepAR*.



Fonte: Próprio Autor.

criar e trabalhar as redes neurais recorrentes, estas tem alguns requisitos específicos que precisam ser preenchidos para um funcionamento adequado do algoritmo, esses requisitos são conhecidos como Hiperparâmetros do *DeepAR* demonstrados na tabela 5.

A figura 23 demonstra como estes valores foram utilizados dentro do código desenvolvido para este estudo. Na figura apresenta-se o valor de *context_length* igual a 720, este valor foi escolhido por estar acima de 500 que é o valor padrão e também abaixo de 1000 que é o valor máximo. Já o valor de *Epochs* foi configurado em 1000 a fim de se ter a maior passagem possível pelos dados de treinamento. Ademais, o valor

ID do requisito	Descrição do requisito	Prioridade
01	O sistema deve ter, em sua composição, os pacotes e bibliotecas necessários para seu funcionamento.	Alta
02	O usuário deve ser capaz de entrar com um *dataset* no sistema.	Alta
03	O sistema deve ser capaz de reconhecer e fazer a leitura do *dataset*.	Alta
04	O sistema deve ser adaptável para que o usuário configure os hiperparâmetros da RNN.	Alta
05	O sistema deve ser capaz de reconhecer os hiperparâmetros e construir a RNN de acordo com tais.	Alta
06	O sistema deve ser capaz de aceitar uma configuração, por parte do usuário, de período de observação de dados do *dataset*.	Alta
07	O sistema deve ser capaz de processar os dados dentro da configuração de previsão do usuário.	Alta
08	O sistema deve ser capaz de, por meio de suas funções de previsão, realizar a previsão desejada do cliente.	Alta
09	O sistema deve ser capaz de salvar os valores de previsão em uma lista e imprimir a mesma para o cliente.	Alta
10	O sistema deve ser capaz de imprimir os gráficos referentes às previsões requeridas pelo cliente.	Alta
11	O sistema deve ser capaz de imprimir as métricas de erro obtidas durante o processo de previsão.	Alta

Tabela 4 – Tabela de requisitos do *DeepAR*

Fonte: Próprio Autor.

Figura 22 – Pacote *GluonTS* e bibliotecas utilizadas

```
!pip install --upgrade mxnet-cu101==1.6.0.post0
!pip install --upgrade mxnet==1.6.0
!pip install gluonts
!pip install mxnet-cu101 gluonts ujson # GPU

import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt

from gluonts.model.deepar import DeepAREstimator
from gluonts.mx.trainer import Trainer
from gluonts.evaluation.backtest import make_evaluation_predictions
from gluonts.evaluation import Evaluator
import numpy as np
import pandas as pd
import json
```

Fonte: Próprio Autor.

de *num_cells* foi configurado em 100, um pouco maior que o desejável uma vez que o *dataset* usado neste estudo compreende 19 anos de dados, sendo muito vasto. O tipo da célula da RNN é LSTM pois é o mais indicado com menor custo computacional e grade desempenho. O valor de *num_layers* foi configurado em 2 como o recomendável do *DeepAR* para que resultado satisfatório uma vez que 2 foi o modelo de rede neural

Figura 23 – Hiperparâmetros do *DeepAR* no código

```
estimator = DeepAREstimator(freq="1440min",
                            context_length=720,
                            num_layers=2,
                            num_cells=100,
                            cell_type='lstm',
                            trainer=Trainer(epochs=1000))
```

Fonte: Próprio Autor.

Nome do parâmetro	**Descrição**
"context_length"	Context Length se refere ao número de pontos no tempo que o modelo lê antes de fazer a previsão, isto é, número de períodos de tempo que o modelo levará em consideração para sua previsão. Esse valor é indicado pelo usuário e, seguindo a melhor prática de programação do modelo, varia de 100 à 1000.
"epochs"	Epochs se refere ao número máximo de passagens para examinar os dados de treinamento, o valor ideial de epochs depende, diretamente, do tamanho do *dataset* que está sendo utilizado, o valor de Epochs é sempre positivo e por padrão tende a variar de 100 à 1000, sendo 500 o valor considerado o valor padrão.
"num_cells"	Num Cells se refere ao número de células que serão usadas dentro da RNN (rede neural recorrente), esse valor é sempre um inteiro positivos, variando de 30 à 100 a fim de obter um resultado satisfatório.
"num_layers"	Num Layers se refere ao número de camadas ocultas da RNN, é sempre um valor inteiro positivo e varia de 1 à 4, tendo como padrão o valor 2.
"cell_type"	Cell Type se refere ao tipo de célula que se encontra nas camadas ocultas da RNN.

Tabela 5 – Hiperparâmetros do *DeepAR*

Fonte: Próprio Autor.

que obteve melhor resultado e que não teve um custo computacional tão alto, por fim, na figura aparece o parâmetro "freq" que se refere a frequência da série temporal, ou seja, a frequência que os dados variam no *dataset*, este parâmetro foi configurado como 1440 minutos uma vez que os dados variavam diariamente.

Após feita a configuração da RNN dentro do código do *DeepAR*, configurou-se as métricas para a previsão dos valores, isto é, os períodos que serviriam de treinamento da RNN e o período que se desejava a previsão. Como já explicado nessa seção, em ambos

modelos de previsão, foram previstos valores já conhecidos, (tempo já passado) para que pudesse ser feita uma comparação com os valores reais e o previsto pelas plataformas, tendo assim mais uma forma de medida de erro do programa. O período de observação foi de 06 de junho de 2000 à 27 de dezembro de 2019, gerando assim um período de observação de 7143 dias, como demonstra a figura 24.

Figura 24 – Configuração do tempo de observação

```
test_data = ListDataset(
    [
        {"start": df_input.index[0], "target": df_input.y[:"2001-06-06"]},
        {"start": df_input.index[0], "target": df_input.y[:"2019-12-27"]}
    ],
    freq = "1440min"
)
```

Fonte: Próprio Autor.

Ademais, o resultado do algoritmo de previsão configurado como descrito nessa sessão, imprimiu-se dois gráficos mostrando os resultados das previsões, o primeiro gráfico na figura 25 demonstra a previsão com maior acurácia que o algoritmo foi capaz de fazer, isto é, das 1000 *epochs* de redes neurais recorrentes treinadas, o resultado com menor taxa de erro encontrado foi este.

No gráfico, a linha azul demonstra os valores reais do ouro naquele período, a linha verde o valor da previsão, e as sombras verde escuro e verde claro 95% e 80%, respectivamente, de previsão para as datas demonstradas no gráfico.

Em termos de margens de erro, o algoritmo imprimiu o valor das métricas MAPE, MAE e RMSE, e estes demonstraram um baixo valor de erro, menor que 1, fazendo com que seja possível considerar o algoritmo altamente capaz e acurado para as previsões

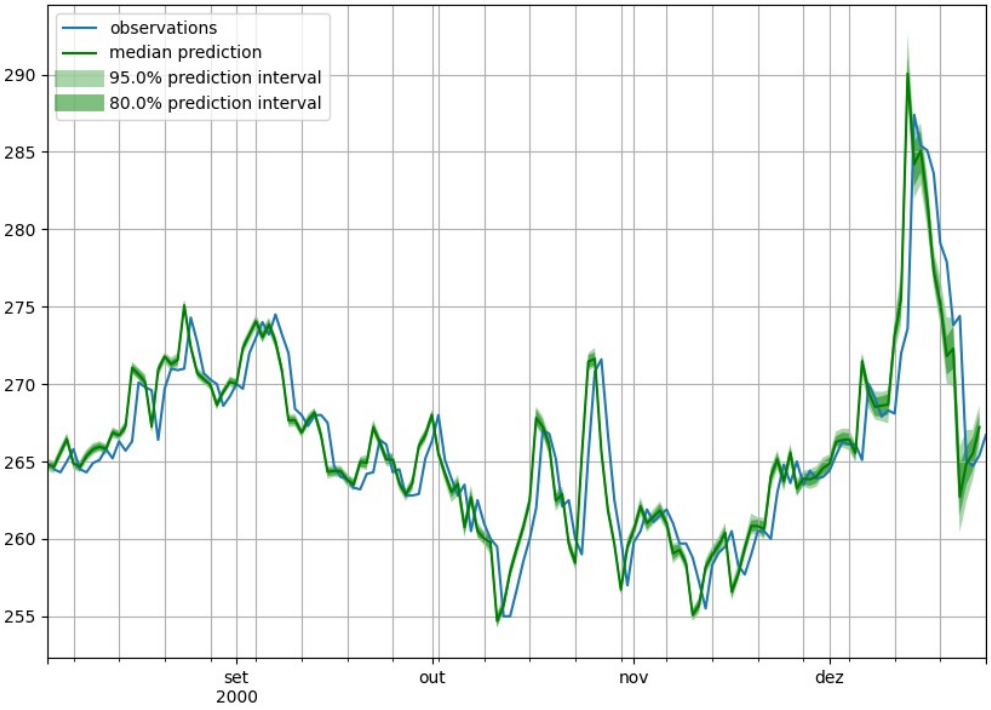
Métrica de erro	**Valor**
MAE	0.25393518518518515
MAPE	0.19324233496768606%
RMSE	0.4745893264972914

Tabela 6 – Métricas e valores

Fonte: Próprio Autor.

feitas. Esses valores não se referem a nenhuma parte específica da previsão, mas sim a média dela como um todo. A tabela 6 traz o valor das métricas. Um adendo à essa conclusão satisfatória do *DeepAR* se da pelo fato do valor do RMSE - que é a distância entre o valor real e o valor previsto pelo algoritmo - ter sido menor que 0,5.

Figura 25 – Previsão de maior acurácia



Fonte: Próprio Autor.

5.1.4 Configuração do modelo *Facebook's Prophet*

O *Facebook's Prophet* ou só *Prophet* além de ser considerado um dos melhores previsores da atualidade traz o grande diferencial que é a possibilidade de sazonalidade, isto é, prever valores para pequenos intervalos de tempo, como para os dias da semana, os finais de semana e até mesmo observar como se comporta a previsão ao longo de um único dia. Uma plataforma de previsão, como dito no capítulo de referencial teórico, de alta acurácia e grande facilidade de configuração para a obtenção de resultados, onde desde o programador com anos de experiência até alguém que tenha noções apenas de probabilidade e estatística conseguem trabalhar e obter previsões.

O *Prophet*, tal qual o *DeepAR* é uma ferramenta que sua codificação pode ser realizada em *Python* ou em R, este consegue realizar a previsão de valores de séries temporais por meio de um modelo linear, como é explicado no capitulo de referencial teórico deste estudo.

Para a construção do algorítmo do *Prophet* os mesmos princípios de engenharia de *software* já descritos aqui foram utilizados, uma vez que trabalhou-se utilizando o modelo

em cascata, foi criado o projeto de processo do modelo a fim de compreender melhor quais passos seguir durante a programação deste, bem como configurar sua variável de entrada que é o *dataset* e ter em mente o que era esperado como saída deste algorítimo. A figura 26 demonstra o projeto de processo da criação da aplicação.

ID do requisito	**Descrição do requisito**	**Prioridade**
01	O sistema deve ter, em sua composição, os pacotes e bibliotecas necessários para seu funcionamento.	Alta
02	O usuário deve ser capaz de entrar com um *dataset* no sistema.	Alta
03	O sistema deve ser capaz de reconhecer e fazer a leitura do *dataset*.	Alta
04	O sistema deve ser adaptável para que o usuário configure as sazonalidades que deseja prever	Alta
05	O sistema deve ser capaz de reconhecer as sazonalidades desejadas do usuário	Alta
06	O sistema deve ser capaz de aceitar uma configuração, por parte do usuário, de período de observação de dados do *dataset*.	Alta
07	O sistema deve ser capaz de processar os dados dentro da configuração de previsão do usuário.	Alta
08	O sistema deve ser capaz de, por meio de suas funções de previsão, realizar a previsão desejada do cliente.	Alta
09	O sistema deve ser capaz de imprimir os gráficos referentes às previsões requeridas pelo cliente.	Alta
10	O sistema deve ser capaz de imprimir as métricas de erro obtidas durante o processo de previsão.	Alta

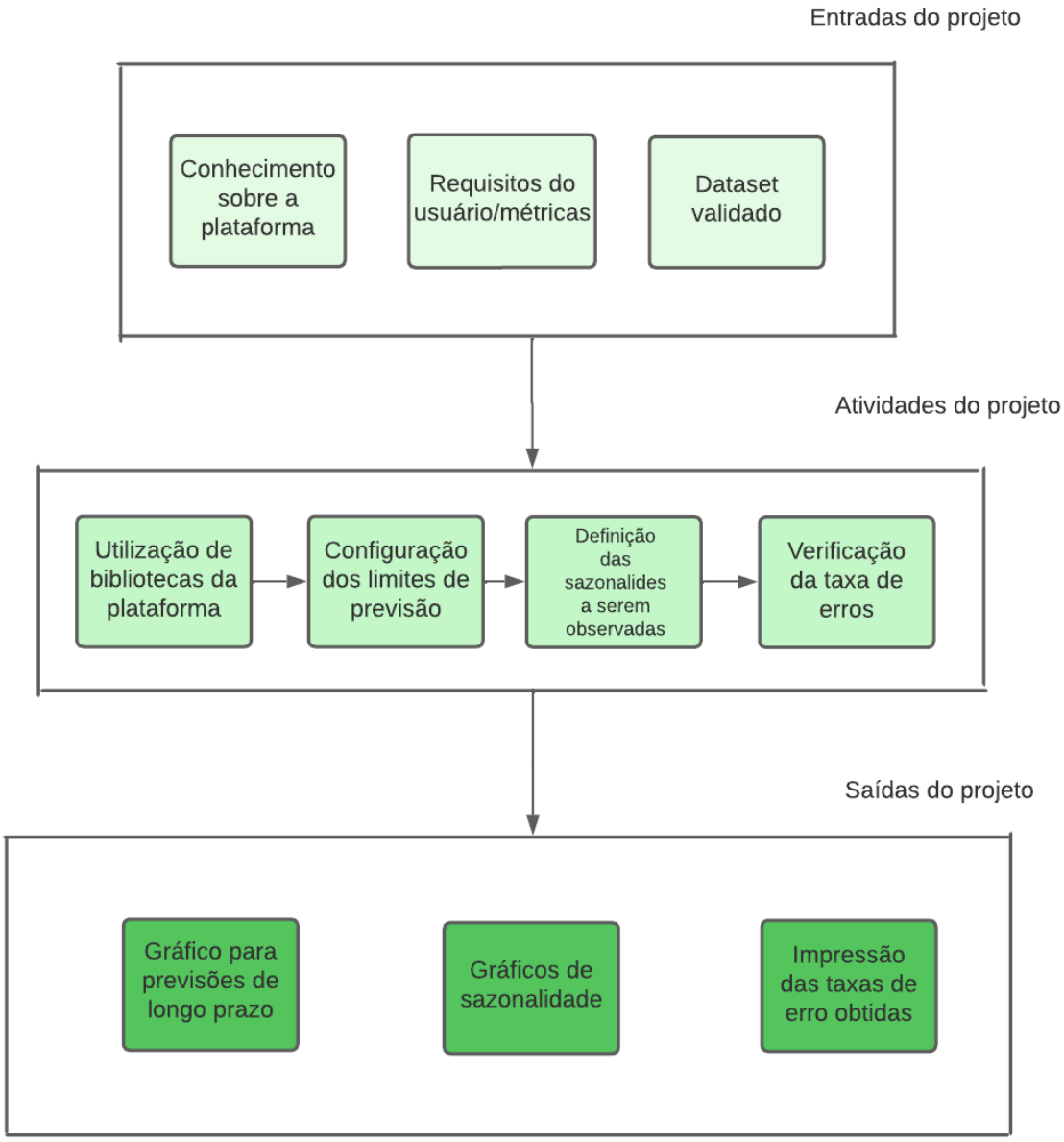
Tabela 7 – Tabela de requisitos do *Facebook's Prophet*

Fonte: Próprio Autor.

O segundo passo após ter o projeto de processo pronto, é criar a tabela de requisitos do *software*, onde, como já dito nessa sessão, contém todos os requisitos que o *software* deve atender para atingir o objetivo proposto pelo cliente, esta tabela está exposta na tabela 7.

Ademais, com todo o planejamento do *software* pronto e o *dataset* refinado e pronto para uso, começou-se então a codificação na plataforma. Primeiramente definiu-se as bibliotecas a serem utilizadas, estas são responsáveis aqui por trazer funções do *Prophet* para o algoritmo que está sendo criado, funcionam como um grande acervo de funções que, quando trazidas para o código são o que fazem as funções criadas pelo programador de fato funcionar e elaborar resultados. As bibliotecas utilizadas foram: Pandas e NumPY, Pandas é uma biblioteca para análise e manipulação de dados que contém diversas extensões que também foram utilizadas a fim de manipular corretamente os dados, validar

Figura 26 – Projeto de processo do *Prophet*



Fonte: Próprio Autor.

métricas de erro e imprimir gráficos. Já a biblioteca NumPY pode ser chamada também de *Numerical Python*, ou *Python* numérico, é responsável por ter a capacidade de realizar operações em *arrays* multidimensionais, o que é muito útil para trabalhar com o *dataset* aqui utilizado uma vez que este é uma tabela (o que pode ser considerado como um *array* multidimensional). As bibliotecas estão expostas na figura 27

Após a configuração das bibliotecas, partiu-se para a entrada dos valores no programa, que é dada pela entrada do *dataset* contendo datas e valores do ouro. Junto à entrada do *dataset* há também o uso da NumPY para tratar o *array* multidimensional

Figura 27 – Bibliotecas utilizadas no *Prophet*

```
import pandas as pd
import numpy as np
from fbprophet import Prophet
from fbprophet.diagnostics import cross_validation
from fbprophet.diagnostics import performance_metrics
from fbprophet.plot import plot_cross_validation_metric
from fbprophet.plot import add_changepoints_to_plot
import matplotlib.pyplot as plt
```

Fonte: Próprio Autor.

que o *dataset* é. Esse processo é demonstrado na figura 28.

Figura 28 – Entrada do *dataset*

```
df = pd.read_csv('goldinho.csv')
df.plot()
plt.savefig('goldinho.jpg')
plt.show()
df['y'] = np.log(df['y'])
```

Fonte: Próprio Autor.

O próximo passo após a passagem do *dataset* é configurar o tipo de previsão que deseja obter, e para isso, utilizou-se as funções relativas à predição com seus parâmetros: tipo de crescimento, tipo de sazonalidade, período médio, e ordem de Fourier. O tipo de crescimento é necessário para que os gráficos sejam obtidos corretamente, o tipo de sazonalidade traz mais opções de observação das previsões, o período médio é configurado em 30.4 uma vez que essa é a média de dias que um mês tem. A figura 29 demonstra as funções no código.

Figura 29 – Funções de predição

```
m = Prophet(growth='linear', daily_seasonality=True)
m.add_seasonality(name='monthly', period=30.4, fourier_order=7
m.fit(df)

future = m.make_future_dataframe(periods=12, freq='M')
future1 = m.make_future_dataframe(periods=3, freq='M')

fcst = m.predict(future)
```

Fonte: Próprio Autor.

Após a configuração da previsão que se deseja, utilizou-se funções para a exposição dos graficos relativos às previsões, e passou-se para a configuração dos parâmetros de métricas de erro, nesta plataforma também foram utilizadas as métricas

MAE, MAPE e RMSE. A função *cross_validation*, responsável pela configuração das métricas de erros, recebeu como parâmetros o tempo inicial do *dataset* que é de X dias, o perído de um ano para erros, e a observação de 730 dias. Ademais, imprimiu-se os gráficos referentes a cada métrica de erro. Essas funções são expostas na figura 30

Figura 30 – Funções de métricas de erros e impressão de seus gráficos.

```
df_cv = cross_validation(m, initial='6570 days', period='365 days', horizon='730 days')
df_p = performance_metrics(df_cv)


print(df_p.head());
fig = plot_cross_validation_metric(df_cv, metric='mape')
fig.savefig('gold4.jpg')
fig.show()
fig = plot_cross_validation_metric(df_cv, metric='rmse')
fig.savefig('gold5.jpg')
fig.show()
fig = plot_cross_validation_metric(df_cv, metric='mae')
fig.savefig('gold6.jpg')
fig.show()
```

Fonte: Próprio Autor.

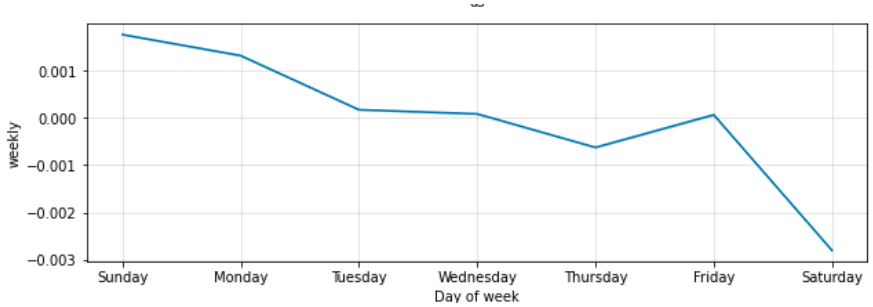
O algoritmo do *Prophet* imprimiu as seguintes previsões com as configurações aqui descritas:

- Previsão semanal, descritas na figura 31;
- Previsão de mensal, descritas na figura 32;
- Previsão anual, descrita na figura 33;
- Previsão de sazonalidade, descrita na figura 34;
- Previsão geral, onde os pontos pretos são os valores reais, a linha azul a previsão gerada e o sombreado azul claro a margem de erro. Na figura 35.
- Previsão geral suavizada, onde os pontos pretos são os valores reais, a linha azul a previsão gerada, o sombreado azul claro a margem de erro e a linha vermelha a suavização exponencial. Na figura 36.

Por fim, o algoritmo imprimiu os gráficos relativos as mesmas métricas já explicadas na sub sessão anterior. Nos gráficos do *Prophet* os pontos são os valores reais e a linha azul os valores previstos. Foram impressas as métricas MAE na figura 37, a

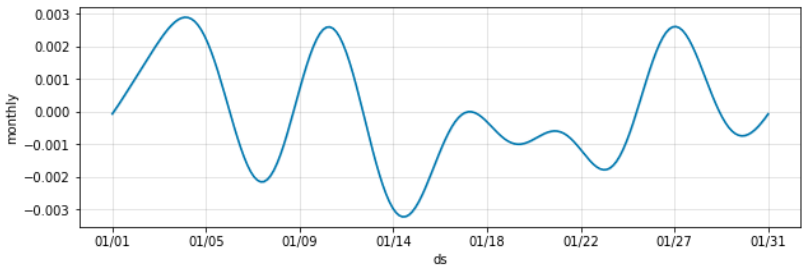
MAPE na figura 38 e por fim a RMSE na figura 39,.

Figura 31 – Previsão semanal.



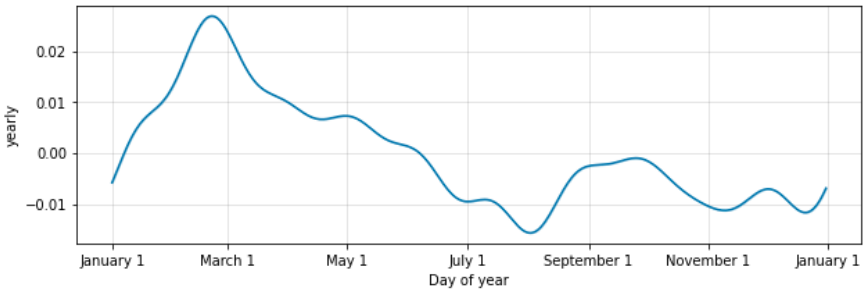
Fonte: Próprio Autor.

Figura 32 – Previsão dentro de um mês.



Fonte: Próprio Autor.

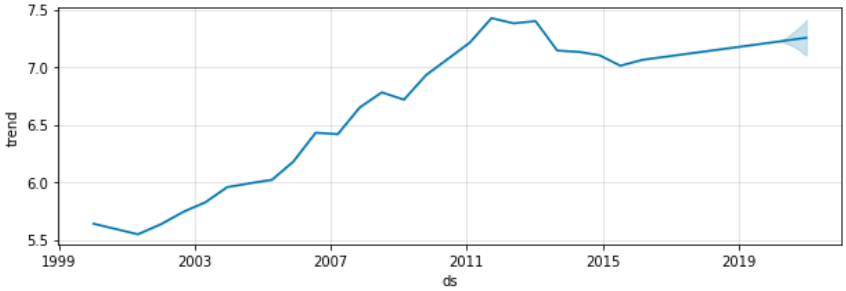
Figura 33 – Previsão anual.



Fonte: Próprio Autor.

Os valores das métricas de erro são expostos na tabela 8 afim de demonstrar as métricas numericamente. Com essa tabela é possível perceber que os valores são extremamente baixos, chegando até um mínimo de 0,006. Isso se dá devido ao vasto *dataset* de entrada e a capacidade de acurácia do *software* desenvolvido.

Figura 34 – Previsão de sazonalidade.



Fonte: Próprio Autor.

Figura 35 – Previsão geral.

Fonte: Próprio Autor.

Métrica de erro	**Valor**
MAE	0.046968
MAPE	0.006528%
RMSE	0.048185

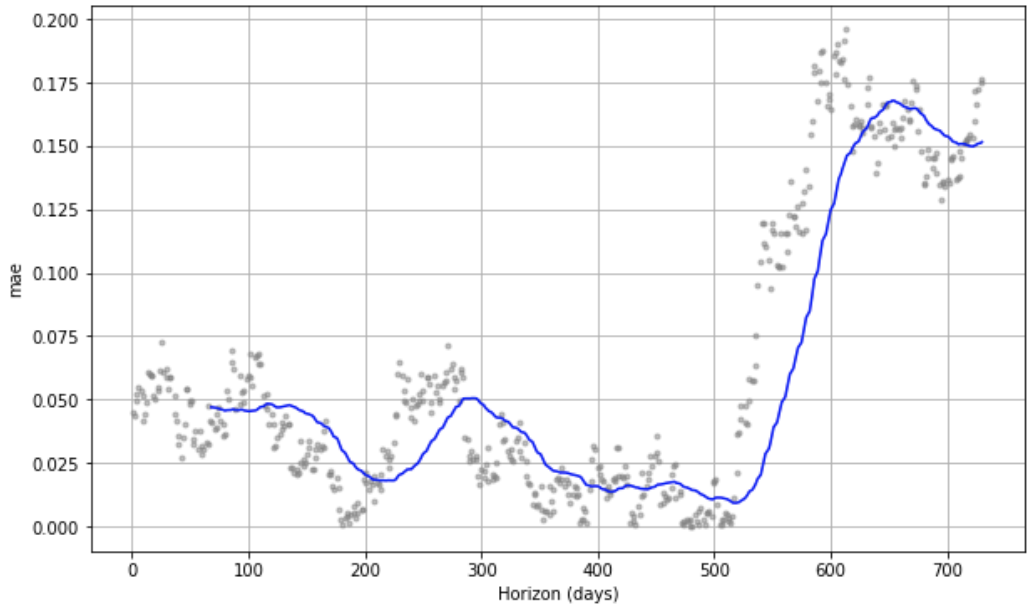
Tabela 8 – Métricas e valores

Fonte: Próprio Autor.

Figura 36 – Previsão geral suavizada.

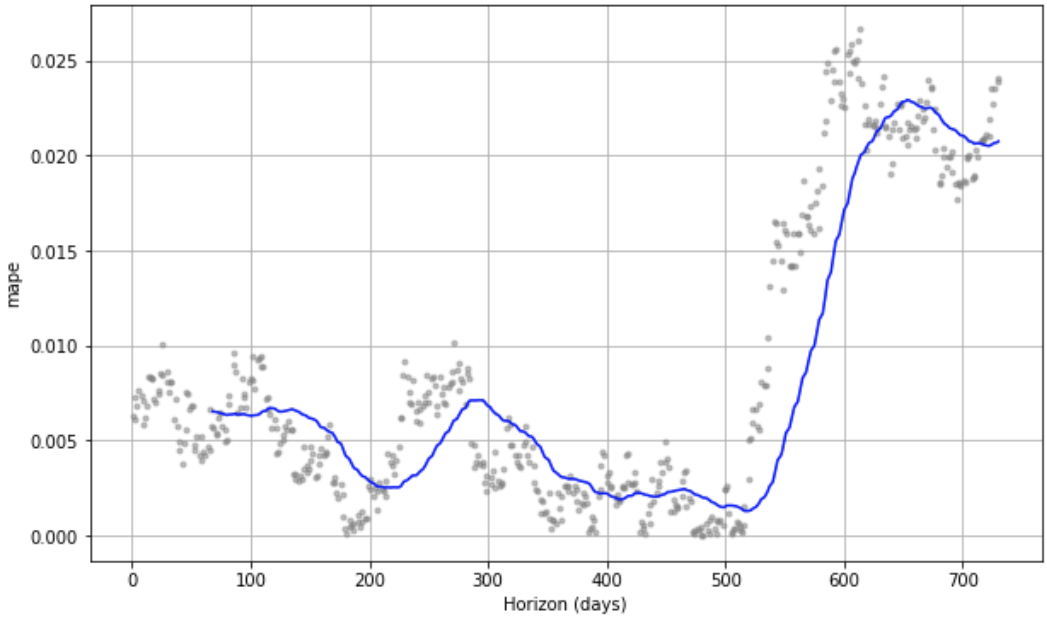
Fonte: Próprio Autor.

Figura 37 – Métrica de erro MAE.



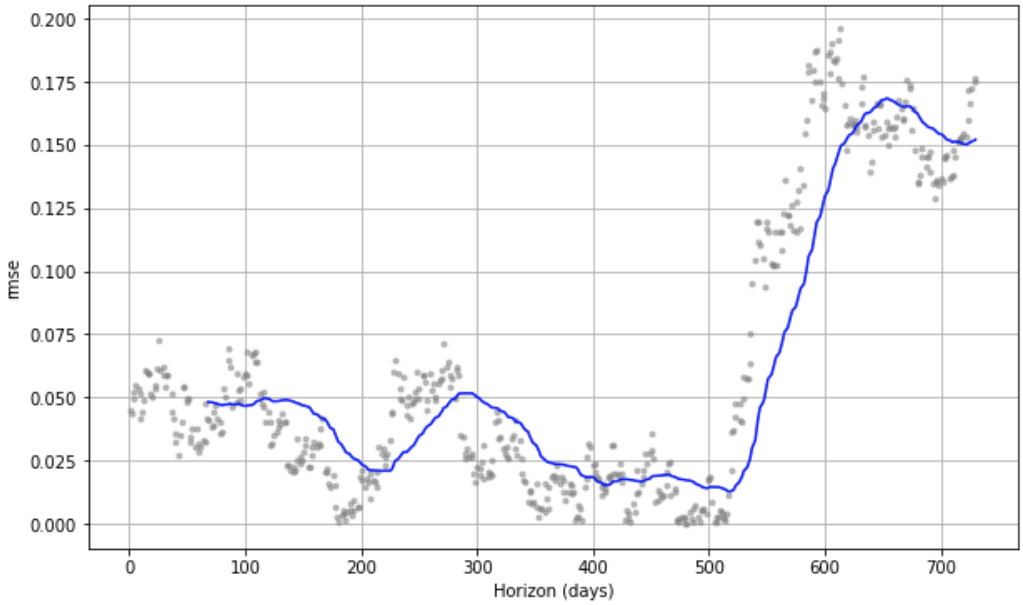
Fonte: Próprio Autor.

Figura 38 – Métrica de erro MAPE.



Fonte: Próprio Autor.

Figura 39 – Métrica de erro RMSE.



Fonte: Próprio Autor.

6 CONSIDERAÇÕES FINAIS

Neste capítulo são abordadas as conclusões do referido trabalho. Ademais, apresenta-se os objetivos realizados, bem como trabalhos futuros.

Ambas ferramentas mostraram-se suficientemente capazes de realizar predições de séries temporais. As predições demonstradas no capitulo de desenvolvimento além das métricas de erros exibidas dão confiabilidade aos resultados encontrados e aos algoritmos desenvolvidos. As métricas de erro tiveram um papel essencial para a validação dos resultados. Ainda que não seja o objetivo deste trabalho, quando trabalha-se com mais de uma ferramente tende-se a comparar as mesmas entre si, e levando em consideração essa comparação, o *Facebok's Prophet* demonstrou resultados mais acurados, maior possibilidades de previsões além de uma configuração mais simples por não fazer uso de RNNs. Por outro lado, o algoritmo da *Amazon*, *DeepAR* demosntra um código muito mais robusto e integral quando comparado em teoria, o que, para engenheiros de computação, é algo satisfatório e mais desafiador que uma plataforma mais simples.

6.1 Trabalhos futuros

Dos objetivos específicos anteriormente mencionados no capítulo de introdução, todos foram atingidos de forma satisfatória. O trabalho aqui desenvolvido, ainda que já vasto, tem espaço para outras várias ramificações, que são expostas abaixo:

- Realizar a predição de outros tipos de ativo financeiro com ambas plataformas;
- Realizar a predição de valores desconsiderando épocas de feriados (Natal, dia das mães, dia das crianças) com a plataforma *Prophet*;
- Realizar uma predição somente com os dados do ouro no período da pandemia do COVID-19 a fim de perceber como este se comportou e como era previsto que se comportasse sem pandemia;
- Incluir novas funções ao *software* do *DeepAR* a fim de projetar outros tipos de previsões.

Uma vez que o trabalho aborda as limitações e capacidades de cada uma de ambas ferramentas de previsão, bem como o referencial teórico de cada uma, este trabalho abre espaço para questionamentos como: "como se comportaria um código que trabalhasse com as mesmos métodos de previsão e funções do *Prophet* e do *DeepAR* de forma

combinada?". Aqui traz-se a ideia de trabalhar com RNN juntamente com o modelo ARIMA e suas ramificações. Esse questionamento é suficiente para ser a justificativa de um novo trabalho de pesquisa que seja a continuação deste.

REFERÊNCIAS

ALEXANDROV, A. et al. Gluonts: Probabilistic time series modeling in python. **arXiv preprint arXiv:1906.05264**, 2019.

ALPAYDIN, E. **Machine learning**. [S.l.]: MIT Press, 2021.

AZIZ, M. I. A.; BARAWI, M. H. Forecasting crude oil price using arima and facebook prophet withi machine learning.

BATES, J. M.; GRANGER, C. W. The combination of forecasts. **Journal of the Operational Research Society**, Taylor & Francis, v. 20, n. 4, p. 451–468, 1969.

BOLLERSLEV, T.; CHOU, R. Y.; KRONER, K. F. Arch modeling in finance: A review of the theory and empirical evidence. **Journal of econometrics**, Elsevier, v. 52, n. 1-2, p. 5–59, 1992.

BOX, G. Box and jenkins: time series analysis, forecasting and control. In: **A Very British Affair**. [S.l.]: Springer, 2013. p. 161–215.

CHARNIAK, E.; MCDERMOTT, D. Introduction to artificial. **Intelligence (Addison Wesley, Reading, MA, 1984)**, 1985.

CHATFIELD, C. The holt-winters forecasting procedure. **Journal of the Royal Statistical Society: Series C (Applied Statistics)**, Wiley Online Library, v. 27, n. 3, p. 264–279, 1978.

CHIEN, A. A.; KARAMCHETI, V. Moore's law: The first ending and a new beginning. **Computer**, IEEE, v. 46, n. 12, p. 48–53, 2013.

DENNIS, R. Optimal policy in rational expectations models: New solution algorithms. **Macroeconomic Dynamics**, Cambridge University Press, v. 11, n. 1, p. 31–55, 2007.

ENGLE, R. **ARCH: selected readings**. [S.l.]: Oxford University Press, 1995.

GASPARENIENE, L. et al. The main gold price determinants and the forecast of gold price future trends. **Economics & Sociology**, Centre of Sociological Research (NGO), v. 11, n. 3, p. 248–264, 2018.

GUHA, B.; BANDYOPADHYAY, G. Gold price forecasting using arima model. **Journal of Advanced Management Science**, v. 4, n. 2, 2016.

HARVEY, A. C.; PETERS, S. Estimation procedures for structural time series models. **Journal of forecasting**, Wiley Online Library, v. 9, n. 2, p. 89–108, 1990.

HAUGELAND, J. **Artificial intelligence: The very idea**. [S.l.]: MIT press, 1989.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HOLT, C. C. Forecasting seasonals and trends by exponentially weighted moving averages. **International journal of forecasting**, Elsevier, v. 20, n. 1, p. 5–10, 2004.

JENKINS, G. M. General considerations in the analysis of spectra. **Technometrics**, Taylor & Francis, v. 3, n. 2, p. 133–166, 1961.

KORSTANJE, J. **Advanced Forecasting with Python**. [S.l.]: Springer, 2021.

KURZWEIL, R. et al. **The age of intelligent machines**. [S.l.]: MIT press Cambridge, 1990. v. 580.

LIPTON, Z. C.; BERKOWITZ, J.; ELKAN, C. A critical review of recurrent neural networks for sequence learning. **arXiv preprint arXiv:1506.00019**, 2015.

NAYLOR, T. H.; SEAKS, T. G.; WICHERN, D. W. Box-jenkins methods: An alternative to econometric models. **International Statistical Review/Revue Internationale de Statistique**, JSTOR, p. 123–137, 1972.

NEIVA, F. W.; SILVA, R. d. S. da. Revisao sistemática da literatura em ciência da computaçao um guia prático. **Universidade Federal de Juiz de Fora**, 2016.

NIELSEN, A. **Practical time series analysis: Prediction with statistics and machine learning**. [S.l.]: O'Reilly Media, 2019.

NORVIG, P.; RUSSELL, S. Inteligencia artificial. **Editora Campus**, v. 20, 2004.

POOLE, D.; MACKWORTH, A.; GOEBEL, R. Computational intelligence. 1998.

PRODANOV, C. C.; FREITAS, E. C. D. **Metodologia do trabalho científico: métodos e técnicas da pesquisa e do trabalho acadêmico-2ª Edição**. [S.l.]: Editora Feevale, 2013.

RAFFERTY, G. **Forecasting Time Series Data with Facebook Prophet**. [S.l.]: Packt Publishing, 2021.

SOMMERVILLE, I. Sommerville: Software engineering. **Monthly Notices of.... https://doi. org/10.1111/j**, p. 1365–2362, 2011.

TAYLOR, S. J.; LETHAM, B. Forecasting at scale. **The American Statistician**, Taylor Francis, v. 72, n. 1, p. 37–45, 2018.

TRIEBE, O. et al. Neuralprophet: Explainable forecasting at scale. **arXiv preprint arXiv:2111.15397**, 2021.

WEI, W. W. Time series analysis. In: **The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2**. [S.l.: s.n.], 2006.

YULE, G. U. On a method of investigating periodicities in disturbed series, with special reference to wolfer's sunspot numbers. **Philosophical Transactions of the Royal Society of London Series A**, v. 226, p. 267–298, 1927.

YURTSEVER, M. Gold price forecasting using lstm, bi-lstm and gru. **Avrupa Bilim ve Teknoloji Dergisi**, n. 31, p. 341–347, 2021.

ZUNIC, E. et al. Comparison analysis of facebook's prophet, amazon's deepar+ and cnn-qr algorithms for successful real-world sales forecasting. **arXiv preprint arXiv:2105.00694**, 2021.