UNIVERSIDADE DE SÃO PAULO

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

PÓS-GRADUAÇÃO EM ENGENHARIA FINANCEIRA

JOÃO BERNARDINO DUARTE NAGY

INTRODUÇÃO À VOLATILIDADE EM OPÇÕES E REDES NEURAIS ARTIFICIAIS:

Aplicando um modelo LSTM para previsão da volatilidade do preço futuro de soja.

São Paulo

2024

UNIVERSIDADE DE SÃO PAULO

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO

PÓS-GRADUAÇÃO EM ENGENHARIA FINANCEIRA

JOÃO BERNARDINO DUARTE NAGY

INTRODUÇÃO À VOLATILIDADE EM OPÇÕES E REDES NEURAIS ARTIFICIAIS: Aplicando um modelo LSTM para previsão da volatilidade do preço futuro de soja.

Trabalho de conclusão de curso apresentado ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo no MBA em Engenharia Financeira.

Orientador: Prof. Doutor Bruno Augusto Angélico

São Paulo

2024

Resumo

Este trabalho investiga a aplicação de redes neurais artificiais, especificamente utilizando o modelo Long Short-Term Memory (LSTM), para previsão da volatilidade realizada em ativos financeiros, destacando a relevância desta medida no contexto de opções financeiras. Abordando a complexidade inerente à previsão de volatilidade futura, o estudo demonstra que o modelo LSTM ajuda a complementar métodos tradicionais de previsão, através de uma análise prática que incorpora desafios reais de dados financeiros. Além de desenvolver o modelo preditivo, o trabalho visa detalhar os principais conceitos relacionados à derivativos, precificação de opções, tipos de volatilidades e redes neurais artificiais, necessários para a compreensão integral do problema e da ferramenta.

Palavras-Chave: Derivativos, opções, volatilidade, aprendizado de máquina, redes neurais artificiais, LSTM.

Abstract

This work investigates the application of artificial neural networks, specifically using the Long Short-Term Memory (LSTM) model, for the prediction of realized volatility in financial assets, highlighting the relevance of this measure in the context of financial options. Addressing the inherent complexity of predicting future volatility, the study demonstrates that the LSTM model helps to complement traditional forecasting methods through a practical analysis that incorporates real-world financial data challenges. Beyond developing the predictive model, the work aims to detail the key concepts related to derivatives, option pricing, types of volatilities, and artificial neural networks, necessary for a comprehensive understanding of the problem and the tool.

Keywords: Derivatives, options, volatility, machine learning, artificial neural networks, LSTM.

Sumário

1) Introdução

Este trabalho visa desenvolver um modelo preditivo baseado em inteligência artificial (IA) com o objetivo de estimar com maior acurácia a volatilidade esperada de ativos financeiros. A volatilidade é um componente crucial que influencia o preço dos ativos e, por extensão, afeta as estratégias empregadas em operações com opções. Enquanto um modelo preditivo de volatilidade já seria relevante para o manejo direto de ativos, é nas opções que observamos uma influência mais direta e quantificável sobre os preços.

Optou-se pelo uso de redes neurais artificiais (*RNAs*) como metodologia central para a predição. Esta escolha é justificada pela competência das *RNAs* em identificar padrões complexos de variação, utilizando variáveis exógenas, mas pertinentes, à determinação dos preços dos ativos.

A escolha no estudo da volatilidade futura, em contrapartida ao preço futuro do ativo, decorre do entendimento de que a volatilidade é uma medida de dispersão dos preços e não necessariamente da tendência direcional de seus movimentos. Assim, estima-se que o modelo proposto terá maior aplicabilidade e relevância em contextos práticos.

O escopo deste estudo não se restringe à construção e análise de um modelo de IA genérico; ele se estende à aplicação específica desse modelo no contexto das estratégias operacionais com opções. Para tal, serão introduzidos conceitos fundamentais de derivativos, estatística e aprendizado de máquina, visando elucidar as motivações, implicações e possíveis aplicações do modelo desenvolvido.

Uma abordagem didática será empregada na explicação dos conceitos, abrangendo desde os mais elementares até os mais complexos, com o intuito de tornar o conteúdo acessível até mesmo para leitores que possuam conhecimento limitado nas áreas pertinentes. De qualquer forma, é importante que o leitor tenha um conhecimento básico de cálculo numérico para uma compreensão integral dos métodos e aplicações discutidos neste trabalho.

2) Revisão de Literatura

A previsão de volatilidade apresenta desafios particulares devido à natureza não linear dos dados de volatilidade. Engle (1982)[1] superou os desafios de agrupamento de volatilidade e a curtose da distribuição com o modelo ARCH (do inglês, *AutoRegressive Conditionally Heteroscedastic*), utilizando termos de erro defasados de uma equação média como variáveis independentes. Devido às dificuldades em especificar o comprimento do *lag* no modelo ARCH, Bollerslev (1986)[2] avaliou o modelo GARCH (do inglês, *Generalized ARCH*), que inclui *lags* anteriores da variância, abrangendo efetivamente valores infinitos passados do erro quadrado da equação média. No entanto, o modelo GARCH pressupõe uma variância de retorno simétrica, enquanto observações indicam que retornos negativos impactam mais a variância futura do que positivos.

Além dos modelos do tipo ARCH, vários modelos de volatilidade estocástica foram desenvolvidos, como os propostos por Hull e White (1987)[3]. Andersen, Bollerslev, Diebold e Ebens (2001)[4] argumentaram que ambos os tipos de modelos estavam mal especificados, já que apenas um modelo poderia estar corretamente especificado por vez. Isso os levou a propor uma estimativa de volatilidade livre de erros de estimação. Eles demostraram que é possível fazer previsões de volatilidade usando modelos estatísticos padrão sem fazer suposições sobre o processo subjacente de geração de retorno. Além disso, também demostraram que a volatilidade é altamente correlacionada serialmente em um nível mensal, contradizendo a crença geral da época.

Müller et al. (1993)[5] calcularam a autocorrelação de 20 minutos de retornos absolutos para a taxa de câmbio USD/DEM usando a modelagem fractal de

---

[1] ENGLE, R. F. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. Econometrica: Journal of the Econometric Society, v. 50, n. 4, p. 987, 1982. Disponível em: https://doi.org/10.2307/1912773. Acesso em: 11/11/2023.

[2] ANDERSEN, T. G.; BOLLERSLEV, T.; DIEBOLD, F. X.; EBENS, H. The distribution of realized stock return volatility. Journal of Financial Economics, v. 61, n. 1, p. 43-76, 2001. Disponível em: https://doi.org/10.1016/S0304-405X(01)00055-1. Acesso em: 11/11/2023.

[3] HULL, J.; WHITE, A. The pricing of options on assets with stochastic volatilities. The Journal of Finance, v. 42, n. 2, p. 281-300, 1987. Disponível em: https://doi.org/10.1111/j.1540-6261.1987.tb02568.x. Acesso em: 11/11/2023.

[4] ANDERSEN, T. G.; BOLLERSLEV, T.; DIEBOLD, F. X.; EBENS, H. The distribution of realized stock return volatility. Journal of Financial Economics, v. 61, n. 1, p. 43-76, 2001. Disponível em: https://doi.org/10.1016/S0304-405X(01)00055-1. Acesso em: 11/11/2023.

[5] MÜLLER, U. et al. Fractals and intrinsic time - A challenge to econometricians. UAM, 1993.

Mandelbrot (1963)[6], encontrando anomalias em *lags* de um dia e uma semana. Esses achados apoiaram a Hipótese de Mercado Heterogêneo (HMH), que afirma que os participantes do mercado diferem em horizonte de investimento e, portanto, diferem em seu tempo de reação às notícias. Portanto, a volatilidade consiste em componentes de curto, médio e longo prazo.

Inspirado por Andersen, Bollerslev, Diebold e Ebens (2001) e pela HMH, Corsi (2009)[7] formulou a base para o modelo HAR (do inglês, *Heterogeneous AutoRegressive Realized Volatility*), que inclui apenas três termos: volatilidade diária, média móvel semanal e mensal dessa volatilidade. Ao examinar a volatilidade do retorno da taxa de câmbio USD/CHF, comparando modelos GARCH e fractais ao HAR, concluiu-se que o HAR tinha desempenho superior. O principal "motor" do modelo HAR era a capacidade de modelar dependências de curto e longo prazo em valores passados. Várias variações do modelo HAR foram propostas, e ele é um dos modelos mais utilizados para previsão de volatilidade.

Introduzindo as RNAs para predição de volatilidade, Karaali, Edelberg e Higgins (1996)[8], *Modelling Volatility Derivatives Using Neural Networks* apresenta uma abordagem para prever a volatilidade do marco alemão (antiga moeda da Alemanha, hoje em desuso) e a precificação de opções usando redes neurais. Sua metodologia é dupla, concentrando-se primeiro na criação de um modelo para prever um índice de volatilidade e, em segundo lugar, no desenvolvimento de uma rede neural para precificação de opções com base nesse índice.

Para prever o índice de volatilidade, eles empregam uma rede de *backpropagation* de três camadas com uma estrutura de entrada de Rede Neural de Atraso de Tempo (TDNN, do inglês *Time Delay Neural Net*). Essa rede é treinada com dados históricos de dois meses, compostos por 44 dias de negociação, e incorpora duas camadas ocultas com 20 e 10 neurônios, respectivamente. Enquanto a rede projetada para previsões de um mês mostrou potencial, superando os modelos ARCH

[6] MANDELBROT, B. The variation of certain speculative prices. The Journal of Business, v. 36, n. 4, p. 394, 1963. Disponível em: https://doi.org/10.1086/294632. Acesso em: 11/11/2023.
[7] CORSI, F. A simple approximate long-memory model of realized volatility. Journal of Financial Econometrics, v. 7, n. 2, p. 174-196, 2009. Disponível em: https://doi.org/10.1093/jjfinec/nbp001. Acesso em: 11/11/2023.
[8] KARAALI, O.; EDELBERG, W.; HIGGINS, J. Modelling volatility derivatives using neural networks. In: Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr), New York, NY, USA, 1997. p. 280-286.

tradicionais, as redes de previsão de longo prazo (três e seis meses) demonstraram limitações, produzindo predominantemente valores médios históricos. Esta observação sugere que melhorar os dados de entrada com indicadores econômicos e financeiros adicionais podem melhorar as capacidades preditivas da rede a longo prazo.

Na precificação de opções, os autores desenvolvem uma rede neural modular composta por sete submódulos, projetada para calcular preços de opções de compra e venda para opções de um mês no índice de volatilidade. Esta rede leva em consideração o nível atual do índice, o nível de exercício e a volatilidade histórica de um mês. Notavelmente, a rede neural demonstra uma capacidade notável de prever preços de opções dentro e fora da amostra, depois de ter sido treinada em dados de 1991 a 1994 e testada em dados de 1995. Esta aplicação bem-sucedida de redes neurais na precificação de opções sublinha o potencial de tais modelos em áreas onde os modelos financeiros tradicionais podem ser menos eficazes, marcando um avanço significativo na utilização de técnicas de aprendizagem automática em aplicações financeiras.

Predicting Stock Index Volatility Using Artificial Neural Networks de Ola Johnson (2018)[9], concentra-se na previsão da volatilidade semanal dos principais índices de ações, especificamente os índices de ações suecos (OMXS30), britânicos (FTSE100) e australianos (S&P/ASX200), usando redes neurais artificiais (RNAs) e comparando seu desempenho com modelos tradicionais do tipo ARCH (GARCH, EGARCH, TGARCH).

No desenvolvimento de seu modelo, a tese emprega uma rede neural feed-forward dinâmica de três camadas de *backpropagation* para representar a estrutura da RNA. O objetivo principal do estudo é avaliar se as RNAs podem superar a precisão de previsão dos modelos mais convencionais do tipo ARCH para volatilidade semanal de índices de ações. Para isso, a tese utiliza uma metodologia de teste fora da amostra, aplicando-a aos 20% mais recentes das observações de dados. Esta metodologia permite uma comparação direta dos desempenhos de previsão de volatilidade dos diferentes modelos em exame. Os critérios de avaliação incluem

[9] JOHNSSON, Ola. Predicting Stock Index Volatility Using Artificial Neural Networks: An empirical study of the OMXS30 FTSE100 & S&P/ASX200. Junho de 2018

métricas *como Root Mean Square Error* (RMSE), *Mean Absolute Error* (MAE), *Mean Absolute Percentage Error* (MAPE) e erro amostral fora da amostra.

No entanto, contrariamente às expectativas, os resultados deste estudo abrangente não mostram nenhuma evidência de superioridade das RNAs na previsão da volatilidade sobre os modelos do tipo ARCH para qualquer um dos três índices de ações. Este resultado sugere que, embora as RNAs sejam ferramentas poderosas para reconhecimento de padrões e processamento de dados, sua eficácia na previsão da volatilidade dos índices de ações, especialmente semanalmente, pode não ser tão pronunciada como em outras aplicações financeiras ou pode exigir arquiteturas de modelos e variáveis de entrada mais personalizadas.

Dannström e Broang (2022)[10] investigam a eficácia das redes neurais artificiais (RNAs) na previsão da volatilidade realizada, examinando especificamente dois tipos de RNAs: modelos de redes neurais *feedforward* (FNN) e memória de longo e curto prazo (LSTM). A pesquisa deles compara essas RNAs com o modelo autorregressivo heterogêneo (HAR) em diferentes regimes de volatilidade, com foco em 23 ações do índice sueco OMXS30.

A metodologia do estudo inclui uma análise abrangente dos dados que quase 12 anos, entre 2010 e 2022. Incorpora variáveis de entrada endógenas (como volatilidade histórica) e exógenas (volatilidade implícita de outros índices) para treinar e avaliar os modelos. As RNAs são testadas quanto ao seu desempenho de previsão durante períodos de alta e baixa volatilidade, avaliando sua robustez e comparando suas capacidades preditivas com o modelo HAR.

Dannström e Broang concluem que, embora as RNAs geralmente superem o modelo HAR, seu desempenho não é uniformemente superior em todos os regimes de volatilidade. Esta variação é influenciada pelo regime de regularização empregado nos modelos de RNA. O estudo revela que uma regularização mais baixa ajuda a aumentar a precisão durante os dias de alta volatilidade, enquanto uma regularização mais alta beneficia o desempenho durante os períodos de baixa volatilidade. Além disso, a pesquisa confirma um compromisso entre a complexidade do modelo e o

[10] DANNSTRÖM, C. O.; BROANG, A. Volatility Forecasting with Artificial Neural Networks: Can we trust them? 2022. Dissertation. Disponível em: https://urn.kb.se/resolve?urn=urn:nbn:se:su:diva-218673. Acesso em: 12/11/2023.

desempenho durante cenários de alta versus baixa volatilidade nos modelos LSTM, uma relação que está condicionada à regularização utilizada. Essa compreensão diferenciada do desempenho da RNA na previsão de volatilidade oferece insights valiosos sobre sua aplicação na previsão financeira, destacando a necessidade de ajuste cuidadoso do modelo e consideração das condições de mercado.

Observa-se uma divergência notável entre as conclusões do estudo de Dannström e Broang (2022) e de Johnson (2018). Enquanto Dannström e Broang identificam uma superioridade geral das *RNAs* em relação ao modelo HAR, a tese de 2017 não corrobora a superioridade frente aos modelos do tipo ARCH. Esta discrepância pode ser atribuída às diferenças nos conjuntos de dados utilizados, nas arquiteturas das *RNAs* empregadas e nos diferentes regimes de volatilidade analisados.

Em síntese, os textos abordam que, enquanto as redes neurais possuem um potencial significativo na previsão da volatilidade financeira, até em relação à modelos de regressão, sua aplicação eficaz requer uma análise minuciosa das especificidades do mercado, das complexidades dos modelos e da natureza dos dados financeiros.

Esse trabalho focará direto no modelo de RNA e suas variações, para analisar a volatilidade esperada específica de uma commodity, que até agora, não foi abordada pelos estudos anteriores e comparando com um índice de volatilidade já introduzido no mercado.

3) Fundamentos dos Derivativos, Opções e Volatilidade

A análise sobre a volatilidade esperada de ativos financeiros tem aplicações em diversas situações, desde gestão de riscos a estratégias especulativas, instigando o desenvolvimento e estudo de modelos alternativos de previsão. Este trabalho é concebido a partir da perspectiva da negociação de opções, buscando explorar como um modelo de IA poderia contribuir para a concepção de estratégias de negociação mais eficazes.

O propósito deste primeiro capítulo é fornecer uma exposição detalhada dos conceitos chave sobre derivativos, opções, mecanismos de precificação, volatilidade e métodos estatísticos. Estes conceitos são fundamentais para estabelecer um entendimento integral das bases que motivaram a pesquisa e análise dos modelos de redes neurais artificiais, os quais serão abordados nos capítulos subsequentes.

3.1) Fundamentos dos Derivativos

No contexto financeiro, um derivativo é definido como um instrumento financeiro cujo valor deriva (ou seja, origina-se), de um ativo subjacente. Por sua vez, esse ativo subjacente pode ser ações, títulos de renda fixa, commodities, moedas, índices, taxas de juros e entre outros. (Hull, 2012)[11].

Intrinsecamente, o derivativo não possui valor próprio. Sua natureza é a de um contrato que estabelece um acordo entre duas partes, onde as condições de execução e os resultados são diretamente dependentes das variações de preço do ativo subjacente ao longo do período contratual.

Os derivativos financeiros são empregados primordialmente em estratégias de gestão de riscos, comumente referenciadas pelo termo em inglês *hedge*. Este mecanismo é projetado para mitigar riscos associados às flutuações em variáveis como taxas de juros, taxas de câmbio, preços de ações e de commodities, além de riscos de crédito. Adicionalmente, derivativos são utilizados em estratégias de arbitragem, que visam à obtenção de lucros pela exploração de diferenças no preço de um mesmo ativo em mercados distintos, e em operações de especulação

[11] HULL, John C. Introdução aos mercados futuros e opções. 8. ed. São Paulo: BM&F Bovespa, 2012. xiv, 598 p.

financeira, onde investidores buscam rentabilidade por meio da assunção consciente de riscos.

Os derivativos possibilitam que as partes transfiram riscos financeiros específicos para contrapartes que, seja por disposição ou capacidade, estão mais bem posicionadas para gerenciar tais exposições. Dessa forma, os derivativos constituem uma ferramenta essencial para o equilíbrio e a eficiência do mercado financeiro, contribuindo para a redução da incerteza e a estabilização do sistema econômico.

Embora na maioria das operações com derivativos o resultado ser apenas financeiro, há alguns casos, principalmente envolvendo moedas e commodities, onde há a possibilidade de entrega física do ativo subjacente ao término do contrato.

3.2) Tipos de Derivativos

O trabalho desdobrará em mais detalhes e desenvolverá um modelo a partir do estudo das opções e suas especificidades. Antes disso, a Seção 3.2 introduzirá os demais tipos de derivativos financeiros, para traçar um panorama completo dessa classe de instrumentos e, nas próximas seções, permitir uma comparação com as opções.

3.2.1) Contratos Futuros

Um contrato futuro representa um compromisso contratual padronizado entre duas partes para a compra ou venda de um ativo em uma data futura específica, com o preço acordado antecipadamente. Estes contratos possuem padronização em suas especificações e são comercializados em bolsas de valores regulamentadas. Por exemplo, os contratos futuros de commodities são definidos com especificações padronizadas relativas às características e ao volume da commodity, dispondo de vencimentos distribuídos ao longo dos meses do ano.

É importante destacar que, nestes contratos, com frequência ocorre a entrega física do ativo subjacente. Por isso, especuladores interessados apenas nos ajustes financeiros devem estar cientes das datas de negociação limite, que são estabelecidas e padronizadas nos contratos, para evitar a obrigação de entrega ou recebimento físico do ativo no vencimento.

Uma vantagem significativa dos contratos futuros é a eliminação da necessidade de desembolso imediato de capital ou entrega física do ativo no momento presente. Por exemplo, uma empresa que procura fixar o preço futuro de um insumo negociado em bolsa de valores e não dispõe atualmente dos recursos para aquisição. Os contratos futuros emergem como instrumentos de previsibilidade de custos, sendo particularmente eficientes por permitirem a fixação dos preços sem a necessidade de desembolso inicial.

Contudo, geralmente na prática, é exigida uma margem de garantia com o propósito de mitigar o risco de inadimplência das partes envolvidas. O desembolso financeiro e a entrega do ativo ocorrem unicamente no vencimento do contrato. Caso seja possível reverter antecipadamente a operação, a reversão é feita a partir das condições de mercado do momento da reversão e o ajuste é apenas financeiro.

3.2.2) Contratos a Termo (*Fowards*)

Contratos a termo, também conhecidos pelo termo em inglês "*forwards*", possuem semelhanças com os contratos futuros no sentido de estipular a compra ou venda de um ativo por um preço acordado para uma data futura. Contudo, diferenciam-se pela ausência de padronização e por não serem negociados em mercados organizados, como bolsas de valores. Os *forwards* são acordos privados em que as partes envolvidas têm a liberdade de customizar os termos do contrato conforme suas necessidades específicas.

Dado que os *forwards* são transacionados no mercado de balcão, um ambiente menos regulamentado que permite a negociação mais direta entre as partes, eles carregam um risco inerente conhecido como risco de contraparte. Este risco decorre da possibilidade de uma das partes não cumprir com as obrigações estipuladas no contrato.

Devido à natureza personalizável e ao maior risco associado, os *forwards* são frequentemente utilizados por entidades que necessitam de soluções financeiras mais específicas em relação aos contratos futuros e estão dispostas a aceitar um nível de risco maior em troca dessa flexibilidade.

3.2.3) Swaps

Os contratos de swap, como a própria denominação em inglês já sugere (significa "troca"), constituem um tipo de derivativo financeiro onde duas partes acordam trocar entre elas fluxos de rentabilidade ou variações de preços de determinados ativos ou índices. Por exemplo, uma parte acorda em trocar a rentabilidade advinda de uma taxa de juros totalmente pré-fixada por outra totalmente pós-fixada, ou trocar a variação de valor de uma moeda pôr a de outra.

Em uma operação de swap, existem dois lados (também conhecidos como “pernas”): o lado "ativo" se refere ao ativo subjacente cuja variação de valor é recebida por uma parte, o lado "passivo" corresponde ao ativo subjacente cuja variação de valor deve ser paga. Assim, o que é considerado o lado ativo para uma parte constitui no lado passivo da contraparte, e vice-versa. É importante notar que a variação de valor no lado ativo pode ser negativa e a variação no lado passivo pode ser positiva.

Na execução prática de um swap, não ocorre um ajuste financeiro para cada lado de maneira isolada; ao invés disso, o resultado para cada parte é a sua ponta ativa menos sua ponta passiva. Adicionalmente, o swap pode ser estruturado com troca de fluxos de caixa ao longo da operação, normalmente utilizado para operações envolvendo dívidas ou ativos de renda fixa.

O último tipo de derivativo são as opções, que será estudada em mais detalhes na seção a seguir.

3.3) Opções

3.3.1) Definição

Um contrato de opção caracteriza-se por ser um acordo bilateral, onde as partes se comprometem com a compra ou venda de um ativo subjacente em uma data futura, por um preço previamente estabelecido (denominado preço de exercício ou *strike*). Distintamente dos contratos futuros e a termo, no contrato de opção, o comprador adquire o direito, e não a obrigação, de executar a compra ou venda do ativo.

A aquisição deste direito implica no pagamento de um prêmio pelo comprador ao vendedor da opção. Esse valor não é reembolsável, contudo, o comprador da opção pode recuperar esse custo inicial dependendo da performance do ativo subjacente e,

por extensão, do resultado da operação financeira. O vendedor da opção, ao receber o prêmio, assume o risco de uma possível exigência do exercício deste direito pelo comprador, o que pode implicar em custos maiores que o valor do prêmio, dependendo da variação de valor do ativo subjacente.

3.3.2) Tipos de Contratos de Opção

- *Call* (Contrato de Compra): O contrato de opção de compra confere ao titular (comprador) o direito, mas não a obrigação, de adquirir um ativo subjacente a um preço predeterminado (preço de exercício), dentro de um período especificado. O lançador (vendedor) deste contrato assume a obrigação de vender o ativo subjacente ao titular, caso este decida exercer seu direito. Investidores adquirem contratos de compra com a expectativa de uma valorização do ativo subjacente acima do preço de exercício, buscando lucrar com essa diferença ou proteger-se contra aumentos de preço em posições existentes. Caso o preço do ativo subjacente esteja igual ou inferior ao preço de exercício, exercer a opção de compra torna-se economicamente inviável.
- *Put* (Contrato de Venda): O contrato de opção de venda oferece ao titular o direito, mas não a obrigação, de vender um ativo subjacente a um preço predeterminado (preço de exercício), dentro de um período especificado. O lançador deste contrato tem a obrigação de comprar o ativo subjacente do titular, caso este opte por exercer seu direito. Investidores que adquirem contratos de venda geralmente antecipam uma depreciação no valor do ativo subjacente abaixo do preço de exercício, visando lucrar com a venda ou proteger-se contra quedas de preço em suas posições de ativos. Caso o preço do ativo subjacente esteja igual ou superior ao preço de exercício, exercer a opção de venda torna-se economicamente inviável.

3.3.3) Tipos de Resultados (*Payoff*) de Contratos de Opção

Os resultados financeiros (*payoffs*) de contratos de opção podem ser realizados de duas maneiras principais, dependendo das especificações do contrato e das práticas de mercado:

- Valor de Ajuste (Liquidação em Dinheiro): Na liquidação em dinheiro, o detentor da opção, ao exercê-la, não recebe ou entrega o ativo subjacente. Em vez

disso, é compensado pela diferença entre o preço de exercício da opção e o preço de mercado atual do ativo subjacente. Esse método de liquidação é mais frequente em opções de índices e commodities.

- Entrega do Ativo (Liquidação Física): Por outro lado, nos mercados de opções de ações, é mais comum que a liquidação ocorra por meio da entrega física do ativo subjacente. Se a opção for exercida, o vendedor da opção é obrigado a entregar (no caso de uma *call*) ou comprar (no caso de uma *put*) o ativo subjacente ao preço de exercício.

Independentemente do tipo de opção, seja ela liquidada em dinheiro ou por entrega física, os termos específicos de liquidação dependem das regras da plataforma ou bolsa onde o contrato é negociado. Além disso, certas opções podem oferecer aos detentores a flexibilidade de escolher entre liquidação em dinheiro ou entrega física, proporcionando maior adaptabilidade conforme as preferências ou estratégias do investidor.

3.3.4) *Moneyness*

Outra diferença nas opções para contratos futuros e a termo é a possibilidade de o comprador da opção escolher o preço de exercício, independentemente do preço atual do ativo subjacente, mas impactando o valor do prêmio. Há três termos para a relação entre preço do ativo e de exercício, conhecido pelo termo em inglês, *moneyness*):

- Fora do Dinheiro (em inglês, *Out of The Money,* OTM): Uma opção está fora do dinheiro quando o exercício atual não é economicamente vantajoso com base no preço de exercício em comparação ao preço do ativo subjacente. Para uma *call*, isso ocorre quando o preço de exercício é maior do que o preço atual do ativo. Para uma opção de venda *put*, quando o preço de exercício é menor do que o preço atual do ativo.
- No Dinheiro *(*em inglês*, At the Money,* ATM*)*: Quando o preço de exercício é igual ao preço do ativo subjacente.
- Dentro do Dinheiro *(*em inglês*, In the Money,* ITM*)*: Uma opção está dentro do dinheiro quando o exercício da opção é economicamente vantajoso. Para uma *call*, isto significa que o preço de exercício é menor do que o preço atual do

ativo. Para uma *put*, significa que o preço de exercício é maior do que o preço atual do ativo.

Opções que estão dentro do dinheiro (ITM) possuem prêmios mais altos devido ao seu maior valor intrínseco (mais detalhes na Seção 3.4.2). É importante notar que, ao negociar opções para datas futuras, a referência de preço para determinar se a opção está fora do dinheiro, no dinheiro ou dentro do dinheiro não é o preço à vista atual do ativo, mas sim seu preço futuro (mais detalhes na Seção 3.4.1).

3.3.5) Tipos de Exercícios de Opções

Existem dois principais tipos de opções em relação à quando é possível realizar o exercício do contrato.

- Opções Europeias: O exercício dessas opções é restrito exclusivamente à data de vencimento do contrato. O preço de ajuste do ativo subjacente para o cálculo do valor de liquidação geralmente corresponde ao preço de fechamento do ativo no último dia útil antes da data de vencimento. Esta data específica para a determinação do preço de ajuste é conhecida como "data de *fixing*". Uma limitação para o titular de uma opção europeia é o risco de que o preço do ativo subjacente não seja favorável na data de *fixing*, limitando as oportunidades de lucro ou proteção.
- Opções Americanas: Estas opções permitem que o exercício ocorra a qualquer momento durante a vida do contrato, incluindo a data de *fixing*. Se a opção não for exercida antes do vencimento, o preço de ajuste final do ativo subjacente é determinado da mesma forma que para as opções europeias, ou seja, baseado no preço de *fixing*. A flexibilidade de exercício das opções americanas é uma vantagem significativa para o titular, permitindo a captura de oportunidades favoráveis de mercado antes do vencimento. Por conta disso, o prêmio pago por opções americanas é, normalmente, mais alto em comparação com opções europeias.

3.3.6) Tipos de Estruturas de Opções

Concluindo os conceitos gerais sobre opções, ao analisar suas estruturas, podemos categorizá-las em dois grupos principais com base nas suas características e complexidades.

- Opções Tradicionais (em inglês, *Vanilla*): Este grupo inclui as opções mais básicas e amplamente utilizadas, cujo resultado financeiro (*payoff*) depende diretamente da relação entre o preço do ativo subjacente no momento do exercício (ou na data de *fixing*) e o preço de exercício. As *calls* e *puts*, tanto americanas quanto europeias, se enquadram nesta categoria.
- Opções Exóticas: Este grupo abrange uma ampla gama de instrumentos financeiros que apresentam características mais complexas em comparação com as opções *vanilla*. As opções exóticas incluem condições adicionais que afetam o *payoff*, como barreiras de preço que ativam ou desativam a opção (opções com barreira) ou mecanismos de cálculo do *payoff* baseados em médias de preços durante um período especificado (conhecido como opções asiáticas), entre outras variações. Essas opções permitem aos investidores estruturarem produtos que se adequam melhor a necessidades específicas de risco e retorno, mas também envolvem maior complexidade na avaliação e precificação.

Para delimitar a discussão e permitir uma análise detalhada, especialmente no contexto de incorporar sistemas de inteligência artificial, este trabalho se concentrará exclusivamente em opções tradicionais (*vanilla*). A complexidade adicional e os cenários específicos de uso das opções exóticas exigem considerações detalhadas que vão além do escopo deste estudo. No próximo segmento, exploraremos as metodologias de precificação aplicadas a derivativos e opções, detalhando os fundamentos e as técnicas utilizadas para avaliar opções tradicionais.

3.4) Conceitos Gerais de Precificação

3.4.1) Preço Futuro de um Ativo

Sempre que negociamos um ativo em uma data futura, precisamos corrigir seu preço a partir de alguns fatores. Caso contrário, a negociação futura abriria espaço para arbitragem (ou seja, ganhos sem riscos) e não compensaria para algumas das partes.

- Custo de Carrego: Refere-se aos custos associados à manutenção de um ativo até uma data futura, incluindo armazenamento, transporte e seguro para bens físicos, além de custos financeiros como juros. Para ativos

financeiros, o custo de carrego é influenciado pela taxa de juros livre de risco. Esses custos tendem a aumentar o preço futuro do ativo.

- Custo de Oportunidade: Refere-se aos custos associados a manter liquidez em vez de deter o ativo. Por exemplo, para ações, os dividendos representam um custo de oportunidade, ao rentabilizar o detentor da ação ao longo do tempo. Dessa forma, a expectativa de dividendos (chamada de *dividen yield*) é descontada do preço futuro. No caso de moedas, o custo de oportunidade está relacionado à taxa de juros estrangeira que poderia ser remunerada ao converter e investir em moeda estrangeira. Commodities possuem um custo de oportunidade conhecido como "*convenience yield*", que reflete o benefício de possuí-las no presente devido a fatores de oferta e demanda ou expectativas de custos de carrego menores no futuro.

A Equação 1 demosntra a fórmula para calcular o preço futuro de um ativo, considerando os custos:

$$Preço\ Futuro = Preço\ à\ Vista\ x\ \frac{Custo\ de\ Carrego}{Custo\ de\ Oportunidade} \qquad (1)$$

Para compreender o motivo de qualquer preço diferente da Equação 1 abrir espaço para arbitragem, tem-se o exemplo abaixo:

- Preço à Vista (S_0) = \$100
- Prazo (*T*) = 1 ano
- Taxa de Juros (*r*) = 10%
- Dividend Yield (*d*) = 5%

Calculando o preço futuro (S_1):

$$S_1 = 100 * \frac{(1+10\%)}{(1+5\%)} = \sim\$104{,}76 \qquad (2)$$

Se o preço futuro do ativo estiver sendo negociado por \$106,00, surgiria uma oportunidade de arbitragem. Seria possível:

1. Vender o futuro a \$106,00.
2. Tomar emprestado \$100,00 à taxa de 10% para um custo de \$10,00 ao final do ano.
3. Comprar o ativo com os \$100,00 para receber o *dividend yield*, resultando em \$5,00 no final do prazo.

Caso o preço do ativo, no vencimento, se manter em $100,00, recebemos mais $6,00 do contrato futuro e o resultado fica $5,00+$6,00-$10,00 = $1,00. Se o preço for para $110,00, pagamos $4,00 no contrato futuro, mas ganhamos $10,00 na valorização do próprio ativo, o resultado fica $5,00+$10,00-$4,00-$10,00 = $1,00.

Este exemplo demonstra que, independentemente do valor final do ativo, a estratégia daria lucro, caracterizando uma arbitragem.

3.4.2) Introdução à Precificação de Opções

A precificação de opções representa um dos aspectos mais complexos e debatidos no campo dos derivativos. O desafio central reside em avaliar uma forma justa e livre de arbitragem o valor de um instrumento cujo resultado depende do preço futuro do ativo subjacente, que por sua vez, é impactado por diversas variáveis.

Embora a avaliação de estruturas exóticas ou ativos com tendências específicas de preços, como as taxas de juros, apresente complexidades adicionais, os princípios básicos para a precificação de opções mais simples são bem estabelecidos e amplamente compreendidos.

A Seção 3.4.1 demonstrou como deduzir o preço futuro livre de arbitragem de um ativo, utilizando seus custos de carrego e oportunidade. Com opções, apesar de ser um pouco mais complexo, os fundamentos da precificação são parecidos; o valor do prêmio deve ser o mesmo do custo da estratégia de negociação que, no vencimento, replica seu resultado.

Antes de aprofundar nesse conceito, vamos entender, de maneira probabilística, quais são os principais fatores que influenciam o preço de uma opção:

- *Moneyness* e Valor Intrínseco: Quanto mais dentro do dinheiro, mais provável de a opção ser exercida, logo, maior o custo do prêmio. Essa relação também é descrita como quanto maior o valor intrínseco da opção, maior o valor do seu prêmio. O valor intrínseco de uma opção é o resultado financeiro que seria obtido em caso de exercício naquele momento. Por exemplo, uma *call* de *strike* $100 com preço do ativo à vista em $110, tem o valor intrínseco de $110 – $100 = $10. Caso o ativo esteja negociando a $90, a opção não seria exercida e seu valor intrínseco é zero.

- Vencimento: Quanto maior o prazo, mais chances de o preço do ativo subjacente mover de forma favorável à operação, mais chances da opção ser exercida e maior o custo do prêmio. O “valor tempo” decresce conforme a estrutura se aproxima do vencimento.
- Volatilidade: Quanto maior a expectativa de variação do preço do ativo durante o período da operação, maior a incerteza e risco do preço ir a favor da operação, logo, maior o custo do prêmio.

Outros fatores, como a taxa de juros e o custo de oportunidade, influenciam indiretamente o preço da opção por meio de seu impacto no preço futuro esperado do ativo subjacente. Esses elementos são integrados às fórmulas de precificação de opções, especialmente quando se considera o preço à vista do ativo como base para o cálculo.

3.4.3) Aprofundando Modelos de Não Arbitragem e Precificação Binomial

Assim como preços futuros, o prêmio da opção precisa ser um valor que não permita arbitragem, caso contrário, o mercado de opções não seria sustentável. Isso ocorreria não apenas por conta de uma das partes, sejam os compradores ou lançadores das opções, não desejarem mais realizar negócios, mas também por não ser possível realizar proteções (*hedge*) para as posições em opções.

Boa parte das negociações no mercado de derivativos só ocorrem por conta de instituições que provêm liquidez para as operações, conhecidos pelo termo em inglês, *market maker*s. Um dos principais objetivos dessas instituições é permitir que contrapartes negociem derivativos, cobrando um custo nas operações, mas sem se expor à riscos direcionais de mercado, ou seja, sem se expor à riscos de perda financeira decorrentes de movimentos no preço do ativo subjacente.

Uma forma de se proteger contra os riscos direcionais de uma opção é criando uma estratégia que replique o resultado esperado dessa opção, comprando ou vendendo o ativo subjacente em quantidades específicas. Para que isso seja viável, o custo dessa estratégia (o prêmio da opção) precisa ser igual ao custo de montar essa proteção.

O modelo binomial de precificação de opções, embora não ser mais muito utilizado, é útil para entender a lógica básica de como as opções são precificadas,

além de servir de base para modelos mais avançados e para a gestão de riscos de posições.

Por exemplo, supondo uma ação que custe $100 no instante de tempo *t0*. Ela só possui dois resultados possíveis no instante de tempo *t+1*;

1. Com 40% de chance, ela pode valorizar $2 e ter seu preço final em $102.
2. Com 60% de chance, ela pode desvalorizar $2 e ter seu preço final $ 98.

Considerando uma taxa de juros nula, queremos saber, em *t0,* o preço de uma *call* europeia, com preço de exercício (k) em $101.

Primeiro, é necessário definir os possíveis resultados da operação, que no vencimento, são os valores intrínsecos da opção (V).

1. Se a ação valorizar, o preço final seria $102 e a opção de compra seria exercida, resultando em V=$102-$101= $1.
2. Se a ação desvalorizar, o preço final seria $98, a opção de compra não seria exercida e a *call* teria o valor intrínseco de $0.

Desenhando o modelo binomial desse exemplo, tem-se o diagrama da Figura 1:

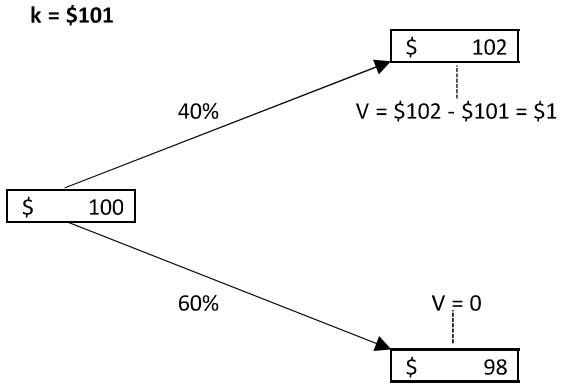

Figura 1 - Árvore Binomial Uniperíodo

Utilizando a medida de probabilidade real - que consiste em encontrar o valor de uma operação a partir da soma dos possíveis cenários e da probabilidade de cada cenário ocorrer - para calcularmos o valor esperado da opção, teríamos:

$$40\% * \$1 + 60\% * \$0 = \$0{,}40 \qquad (3)$$

No entanto, para evitar arbitragem e riscos de mercado, esse não poderia ser o prêmio. Por exemplo, se alguém comprar quatro lotes dessa opção, pagando o

prêmio total de \$0,40*4=\$1,60 e, ao mesmo tempo, vender um lote da ação, recebendo \$100[12], ainda considerando que a taxa de juros do período é 0%, os cenários em *t+1* seriam:

1. Se o preço da ação subir para \$102, haveria um prejuízo de \$2,00 no ativo vendido, mas um lucro de \$1,00 para cada lote da opção, totalizando \$4,00. O resultado total da operação seria a rentabilidade do ativo (-\$2,00) somada à rentabilidade da opção (+\$4,00), descontado o custo inicial das opções (-\$1,60), totalizando +\$0,40 de lucro na operação total.
2. Se o preço da ação cair para \$98, haveria um lucro de \$2,00 no ativo vendido, e nenhum resultado financeiro nas opções. O resultado total da operação seria a rentabilidade do ativo (+\$2,00), descontado o custo inicial das opções (-\$1,60), totalizando +\$0,40 de lucro na operação total.

Assim, conclui-se que se o custo do prêmio fosse o valor probabilístico esperado, haveria espaço para arbitrar a operação. Além disso, seria impossível para o lançador da opção neutralizar o risco da operação, pois se comprar uma ou mais quantidades de ações para cada opção vendida e o preço do ativo cair para \$98, o prejuízo seria de -\$2,00 por ação e o valor que receberia da venda da opção (\$0,40) não seria suficiente para cobrir o prejuízo.

Para solucionar esse problema de maneira correta, é necessário criar um portfólio de valor *P* para o lançador da opção, que consiste no custo do prêmio (*V,* que é negativo para o lançador da opção, pois ele precisa pagar a diferença em caso de alta no preço do ativo), somado às quantidades necessárias de ações (Δ) para neutralizar o risco dos resultados, vezes o preço da ação (*S*). A fórmula geral é:

$$P_t = -V_t + \Delta * S_t \qquad (4)$$

No vencimento da operação, o valor do prêmio da opção é igual ao seu valor intrínseco, logo, sabemos os valores de V.

1. No cenário de alta no preço da ação, temos:

$$P_{t+1} = -1 + 102\Delta \qquad (5)$$

[12] No mercado financeiro, é possível tomar emprestado um ativo para vende-lo no presente, recomprando-o no futuro para devolvê-lo. Isso permite lucrar com uma eventual baixa, mesmo sem possuir o ativo em carteira.

2. No cenário de queda no preço da ação, temos:

$$P_{t+1} = 0 + 98\Delta = 98\Delta \quad (6)$$

Para alguém que busca apenas proteger a posição, os valores absolutos finais desse portifólio não importam, desde sejam iguais entre os dois cenários. Dessa forma, é possível determinar o valor do delta (Δ), ao igualar as equações.

$$98\Delta = -1 + 102\Delta \quad (7)$$

$$1 = 4\Delta \quad (8)$$

$$\Delta = \frac{1}{4}\ ou\ 0{,}25 \quad (9)$$

Substituindo o valor Δ em qualquer equação, descobrimos que o valor dos portifólios em *t+1* deve ser $24,50. Para achar o preço justo do prêmio da opção, deve-se encontrar o valor presente do portifólio, descontado pela taxa livre de risco (*r*) e encontrar V no instante *t0*.

$$\frac{P_{t+1}}{(1+r)^t} = -V_t + \Delta * S_t \quad (10)$$

$$\frac{24{,}5}{(1+0)^1} = -V_{t0} + 0{,}25 * 100 \quad (11)$$

$$V_{t0} = 0{,}50 \quad (12)$$

Resolvendo a equação, calcula-se que o preço justo da opção seria $0,50, maior que o preço calculado com a medida de probabilidade real. Qualquer valor diferente abre espaço para arbitragem; se for menor, como no exemplo anterior, conseguimos arbitrar comprando a opção (que está barata) e vendendo certa quantidade de ações; se for maior, conseguimos arbitrar vendendo a opção (que está cara) e comprando certa quantidade de ações. Assim, foi demonstrado porque o preço de uma opção precisa ser igual ao custo da estratégia que replica seu resultado.

Apesar do exemplo simplificado, o modelo é preciso e possível de estender para casos reais. Ao invés de apenas um período, seriam *n* períodos, estendo a arvore binomial para mais de dois estados e a probabilidade de alta ou baixa poderia ser calculada por um fator derivado da volatilidade do ativo.

Para concluir a seção, é importante indicar que, na prática, o mercado opera com diferentes preços para compra e venda de ativos, opções e contratos de taxa de juros. Essa diferença (conhecida pelo termo inglês de *spread*) que permite aos *market*

makers lucrarem com as negociações sem tomar risco de mercado, além de dificultar o processo de arbitragem, mesmo em casos de preços destoantes.

3.5) Volatilidade e Curvas de Distribuição

Ao compreender os fundamentos da precificação de uma opção, é possível seguir para uma análise das variáveis que impactam seu preço. Essa seção explorará como definir os valores para a variação esperada do preço do ativo e para as probabilidades dessas variações ocorrerem. A volatilidade dos retornos do ativo é o ponto inicial desse tema.

3.5.1) Volatilidade Histórica

A volatilidade é uma medida estatística que representa o grau de dispersão dos retornos de preço de um ativo financeiro em um determinado período. Como quanto mais dispersos os retornos de um ativo, maior seu risco, em outras palavras, a volatilidade é uma forma de quantificar o risco ou a incerteza relacionada a variação do preço de um ativo.

Matematicamente, o cálculo da volatilidade é o desvio padrão da variação de preços, portanto, um número em percentual. Para calcular a volatilidade histórica anual de um ativo, precisa-se calcular primeiro as variações diárias de preço do período desejado:

$$X_i = \frac{Preço_t}{Preço_{t-1}} - 1 \qquad (13)$$

Onde X_i é a variação diária. Depois, calcula-se a média aritmética simples das variações:

$$\overline{X} = \frac{\sum X_i}{Número\ de\ \ Amostras\ (n)} \qquad (14)$$

O próximo passo é calcular a variância amostral[13] σ^2 dos retornos, utilizado para definir o quão distante, em média, os valores do conjunto estão da média de retornos. Calcula-se a diferença de cada elemento para a média, elevando ao quadrado para evitar números negativos e dar mais peso à valores extremos (não será

[13] Para calcular a volatilidade, quase sempre é utilizado um determinado período de observação dos retornos e não todos, logo, precisamos considerar amostras e não a população inteira de retorno para os cálculos de variância e desvio padrão.

feita a demonstração do porquê isso acontece). As diferenças quadráticas são somadas e o resultado é dividido pelo número de amostras (n), menos 1:

$$\sigma^2 = \frac{\sum(X_i - \overline{X})^2}{n-1} \quad (15)$$

Para encontrar o desvio padrão σ, é calculado a raiz quadrada da variância. Esse valor já é definido como a volatilidade diária.

$$\sigma = \sqrt{\frac{\sum(X_i - \overline{X})^2}{n-1}} = Volatilidade\ Diária \quad (16)$$

Para encontrar a volatilidade anualizada, multiplica-se o resultado por raiz quadrada de 252 – utiliza-se esse valor pois, normalmente, é a quantidade de dias que um ativo é negociado ao ano.

$$\sigma\ anualizada = \sqrt{\frac{\sum(X_i - \overline{X})^2}{n-1} * 252} \quad (17)$$

Como exemplo, um ativo que tem a média de retornos 0,10%, variância de 0,0004 e desvio padrão (volatilidade diária) de 2,00%, é um bom indicativo que a maior parte dos retornos diários históricos está entre 0,10% (média) ± 2,00% (desvio padrão) e que, em um dia típico de negociação, o ativo variou 2,00% para cima ou para baixo.

A volatilidade anual do ativo foi de $\sqrt{0{,}0004 * 252} = \sim 31{,}75\%$, ou seja, no decorrer de um ano, espera-se que a maioria dos retornos anuais do ativo esteja dentro de uma faixa de, aproximadamente, 31,75% acima ou abaixo da média anual. Veremos que essa suposição não totalmente precisa, ao não englobar alguns fatores.

Note que os cálculos foram feitos a partir de dados passados. Essa volatilidade é conhecida como volatilidade histórica e não necessariamente refletirá a volatilidade futura. Ao precificar opções, o mercado normalmente utiliza a volatilidade histórica mais como uma referência do que o valor final para a volatilidade esperada.

3.5.2) Distribuições Amostrais

Embora a volatilidade já tenha sido definida, permitindo a compreensão geral para as projeções do modelo proposto, é pertinente introduzir alguns conceitos complementares sobre a distribuição amostral dos retornos, que transcendem a capacidade explicativa da média e do desvio padrão e que também são importantes para modelos de precificação de ativos e opções.

O desvio padrão diário oferece uma estimativa histórica sobre a variação esperada de um ativo em um dia comum. No entanto, a proposição de que a maioria dos retornos diários passados se encontram dentro de um desvio padrão da média não ocorre em todos os casos. A análise das propriedades de uma curva de distribuição nos permite compreender com maior profundidade como os retornos se distribuem em torno da média e do desvio padrão.

Tal curva é essencialmente um gráfico que indica a frequência ou probabilidade de ocorrência de diferentes resultados dentro de um espectro de valores. No eixo horizontal são representados os resultados possíveis — no caso deste estudo, seriam as variações diárias do preço de um ativo — e no eixo vertical são apresentadas a frequência correspondentes a cada resultado. A integral da curva sobre seu domínio (ou seja, a área do gráfico) é igual a 1, refletindo a totalidade do espaço amostral, ou seja, ao considerar todos os retornos passados, temos 100% das amostras de retornos.

Resumindo, a função da curva demonstrar a frequência com que determinados retornos ocorrem. Com a análise histórica dos dados e uma compreensão da forma da curva de distribuição, pode-se melhorar a acuidade das projeções de volatilidade para diferentes cenários futuros.

Antes de aprofundar na análise de uma curva de distribuição, é importante definir outros dois conceitos estatísticos: a mediana, que representa o ponto central de um conjunto de dados, e a moda, que é o valor que aparece com maior frequência. A relação entre esses dois indicadores e a média fornece uma avaliação preliminar da assimetria da distribuição, que será examinada adiante.

3.5.3) Curva de Distribuição Normal

Conhecida também como Distribuição Gaussiana e curva sino, é o modelo distribuição teórica mais utilizado em modelos estatísticos e financeiros. O modelo consiste na curva em que a média, mediana e moda são iguais entre si, refletindo uma distribuição dos resultados simétrica em relação à média. Sua principal propriedade é a relação de distribuição dos resultados:

- 68,3% dos resultados estão a um desvio padrão da média ($\overline{X} \pm \sigma$).
- 95,5% dos resultados estão a dois desvios padrão da média ($\overline{X} \pm 2\sigma$).

- 99,7% dos resultados estão a três desvios padrão da média ($\overline{X} \pm 3\sigma$).

Embora utilizado em muitos modelos estatísticos e até modelos de precificação de opções, como demonstrado por Opdyke (2007)[14], poucos casos de amostras reais de retornos resultam em uma distribuição normal. Segundo Bussab e Morettin (2017) [15], a distribuição normal tem como base o Teorema do Limite Central, que diz ao somar um número alto de amostras aleatórias e independentes, sem que nenhuma seja dominante, deve ter uma distribuição aproximadamente normal.

Não há um número mínimo de amostras para ser considerada grande e, aplicando na prática, vemos algumas divergências do modelo. Ao observar 5 mil retornos do S&P 500 (um dos principais índices de ações do EUA e do mundo), de 14/11/2003 até 27/09/2023, considerando apenas dias com pregão, temos uma média de retornos diária em 0,04% e volatilidade diária em 1,21%. A mediana e moda do modelo são iguais entre si e próximas à média, 0,07%, no entanto, a curva de distribuição não segue a propriedade de resultado da normal (Figura 2).

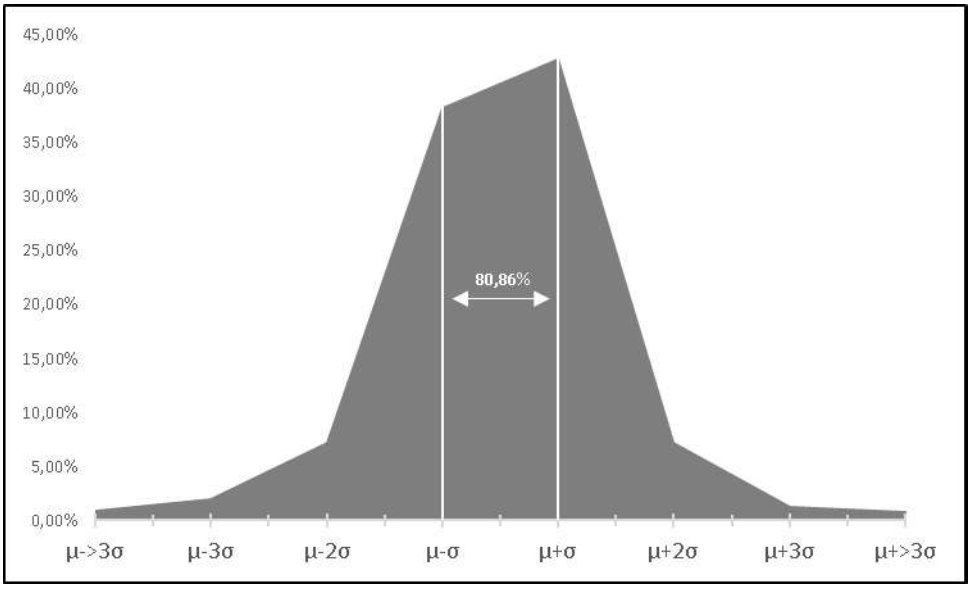

Figura 2 – Curva de Distribuição dos Retornos do S&P 500 (14/11/2003 - 27/09/2023).

Em resumo, considerando um desvio padrão para mais ou menos, a amostra concentrou 80,86% dos resultados, com 42,68% um desvio padrão a mais que a

[14] OPDYKE, J. D. Comparing Sharpe ratios: So where are the p-values?. Journal of Asset Management, v. 8, p. 308-336, 2007. Disponível em: https:// https://papers.ssrn.com/sol3/papers.cfm?abstract_id=886728. Acesso em: 11/10/2023.

[15] BUSSAB, Wilton de O.; MORETTIN, Pedro A. Estatística Básica. 9. ed. São Paulo: Saraiva, 2017. xx, 568 p.

média e 38,18% um desvio padrão a menos que a média. Para dois e três desvios padrão, ficou próximo da distribuição normal, em 95,24% e 98,34%, respectivamente.

Observa-se, então, que há alguns outros fatores a considerar em uma distribuição para analisar o retorno de um ativo.

3.5.4) Assimetria

Segundo Pearson (1895)[16], a assimetria (em inglês, skewness) é uma medida que descreve o grau de distorção da distribuição de uma variável aleatória em relação à distribuição normal. Essencialmente, ela indica em qual direção os dados estão "inclinados".

Fórmula geral da assimetria amostral $(\mathrm{g1})$:

$$\mathrm{g1} = \frac{n}{(n-1)(n-2)} \sum \left(\frac{X_i - \overline{X}}{\sigma}\right)^3 \qquad (18)$$

- Assimetria positiva: a maior parte dos resultados (moda) estão concentrados na parte esquerda da distribuição e cauda aponta para a direita. Isso significa que a média é tipicamente maior que a mediana e moda do conjunto e que a maior parte dos resultados estão a um desvio padrão a menos da média.
- Assimetria negativa: a maior parte dos resultados (moda) estão concentrados na parte direita da distribuição e a cauda aponta para a esquerda. Isso significa que a média é tipicamente menor que a mediana e moda do conjunto e que a maior parte dos resultados estão a um desvio padrão a mais da média.

Em uma distribuição normal, a assimetria é zero. Mesmo que visualmente não pareça, no exemplo do S&P 500 (Figura 2), a assimetria é, aproximadamente, -25,39%.

3.5.5) Curtose

Mesmo que alguns textos definam curtose como a altura do “pico” da curva, na realidade, segundo Oakes (2007)[17], ela é mais uma medida que indica se os dados

[16] PEARSON, K. Contributions to the mathematical theory of evolution.—II. Skew variation in homogeneous material. Philosophical Transactions of the Royal Society of London. Series A, v. 186, p. 343-414, 1895. Disponível em: http://doi.org/10.1098/rsta.1895.0010. Acesso em: 11/10/2023.

[17] OAKES, M. Ord's criterion with word length spectra for the discrimination of texts, music and computer programs. In: Communications in Statistics - Theory and Methods. [S.l.], v. 48, p. 2286-2304, 2007. Disponível em: https://doi.org/10.1515/9783110894219.509. Acesso em: 11/10/2023.

estão concentrados mais próximos ou distantes da média, consequentemente, explica com que frequência as observações do conjunto de dados podem cair nos extremos (caudas) ou no centro da curva.

Fórmula geral da curtose amostral (g2):

$$g2 = \left\{\frac{n(n+1)}{(n-1)(n-2)(n-3)}\sum\left(\frac{X_i-\overline{X}}{\sigma}\right)^4\right\} - \frac{3(n-1)^2}{(n-2)(n-3)} \quad (19)$$

Tipos de curtose:

- Mesocúrtica (curtose = 3): é o tipo da curtose presente na distribuição normal, logo, curvas mesocúrticas possuem certa simetria na distribuição de resultados em relação à média.
- Platicúrtica (curtose < 3): são curvas com picos mais baixos e largos, com a distribuição de resultados mais próximos da média.
- Leptocúrtica (curtose > 3): são curvas com picos mais altos e finos, com a distribuição de resultados mais distantes da média e mais resultados concentrados nas caudas.

Excesso de curtose - ou seja, caso o resultado dela seja maior do que 3 - é interpretado como maior probabilidade de o ativo estar suscetível a uma variação de preço extrema. A curtose sozinha não consegue quantificar esse risco, apenas indicá-lo.

No exemplo do S&P 500, a curtose amostral era de, aproximadamente, 12,90, com excesso alto, conforme já era possível observar no gráfico. Esse valor de curtose faz sentido, ao analisar o índice S&P 500 dentro de um contexto de riscos de mercado. Considerando apenas momentos de caudas, em crises que impactaram a economia, por exemplo, na crise dos *subprime* em 2008 e na pandemia do COVID 19 em 2020, o índice caiu consideravelmente, levando a curtose aumentar.

Não há uma relação quantitativa direta entre a curtose ou assimetria com a volatilidade. Por exemplo, é possível ocorrer casos de ativos com volatilidade baixa, mas curtoses e assimetrias altas, indicando que, na maior parte do tempo, o ativo varia pouco de preço, possui tendencias de alta ou baixa constantemente e, em eventos extremos, o preço muda abruptamente, mas tende a retornar à média.

No entanto, casos reais desse exemplo são pouco prováveis e muitos operadores observam essas duas características para precificar a volatilidade futura de diferentes níveis de preço de exercício, conforme será detalhado na Seção 3.6.2.

3.6) Aprofundando Conceitos de Precificação e Volatilidade

3.6.1) O Método de Monte Carlo

Em modelos de precificação de opções, a volatilidade histórica pode servir como uma referência de cálculo para a projeção de preços futuros, combinando com a média de variações e aplicando-as em métodos estocásticos, que resultaram em números aleatórios, a partir de uma distribuição normal dos retornos e do desvio-padrão.

Um dos mais conhecidos métodos que aplicam essa técnica é o de Monte Carlo, que pode modelar diferentes preços futuros de um ativo (e, consequentemente, o preço de opções), em um processo estocástico que não pode ser facilmente previsto devido à intervenção de variáveis aleatórias.

Segundo Boyle (1977)[18], o conceito por trás do método é criar diferentes cenários (também chamados de "caminhos") para o preço do ativo. Supondo que queiram calcular 100 cenários para o preço de um ativo em 1 ano (252 dias de negociação). Para cada cenário, o método calcula uma variação diária de preço do ativo, a partir da média, volatilidade e um termo de Movimento Browniano - uma variável aleatória normalmente distribuída - até chegar no final do período determinado, no 252º dia. A equação da variação é:

$$\Delta S = \mu S \Delta t + \sigma S \Delta W \qquad (20)$$

Onde:

- ΔS: Variação no preço do ativo para a determinada variação do tempo.
- S: Preço atual do ativo.
- μ: Média de retornos do ativo.
- Δt: Variação do tempo.
- σ: Volatilidade ajustada para o período de Δt.

[18] BOYLE, P. P. Options: A Monte Carlo approach. Journal of Financial Economics, v. 4, n. 3, p. 323-338, 1977. ISSN 0304-405X. Disponível em: https://doi.org/10.1016/0304-405X(77)90005-8. Acesso em: 15/10/2023.

- *ΔW*: Termo de movimento Browniano, que é uma variável aleatória normalmente distribuída com média zero e variação Δt.

A simulação resultaria em 100 caminhos diferentes da variação do preço do ativo até o 252º dia, distribuídos normalmente. Como o preço futuro é o preço atual multiplicado por um termo aleatório, mas positivo, ele nunca assumirá um valor negativo. A média dos valores de cada caminho no dia final pode ser utilizado como uma estimação do modelo para o preço futuro do ativo. Outra utilidade do método refere-se em estabelecer um intervalo de confiança de onde o valor real se encontra entre um intervalo de valores simulados.

Juntando aos conceitos vistos na precificação binomial, o método de Monte Carlo pode ser útil em precificar opções ao simular o valor futuro do ativo e, com isso, podemos calcular o *payoff* das opções e precificá-las.

Para analisar a eficiência de método Monte Carlo, o trabalho de Lerche e Mudford (2005)[19], em uma simulação para preços futuros de ativos de energia, com 1600 caminhos e 99,00% de confiança, a diferença entre o valor real e valor simulado não ficou maior do que 4%, mostrando certa eficiência no modelo.

Estudos como o de McCrary (2015)[20], implementaram ao método as assimetrias e curtoses das distribuições, através de combinações do método de poder de Fleishman (traduzido do inglês, *Fleishman's Power Method*[21]), para a distribuição futura poder ser mais próxima de movimentos passados. Apesar de mostrar instruções de como implementar, o estudo não detalha a eficiência em comparação a modelos tradicionais e foge do escopo desse trabalho explorar esse assunto.

A maior parte das aplicações do método de Monte Carlo em finanças considera a volatilidade como uma variável constante. Trabalhos como o de Sandmann e Koopman (1998)[22], permitem que a volatilidade varie ao longo do tempo de acordo

[19] LERCHE, I.; MUDFORD, B. S. How Many Monte Carlo Simulations Does One Need to Do? Energy Exploration & Exploitation, v. 23, n. 6, p. 405-427, 2005. DOI: 10.1260/014459805776986876.
[20] MCCRARY, S. Implementing a Monte Carlo Simulation: Correlation, Skew, and Kurtosis. Berkeley Research Group [S.l.], 23 set. 2015. Disponível em: https://www.thinkbrg.com/insights/publications/implementing-a-monte-carlo-simulation-correlation-skew-and-kurtosis/. Acesso em: 15/10/2023.
[21] FLEISHMAN, A. I. A Method for Simulating Non-Normal Distributions. Psychometrika, v. 43, n. 4, p. 521-532, 1978.
[22] SANDMANN, G.; KOOPMAN, S. J. Estimation of stochastic volatility models via Monte Carlo maximum likelihood. Journal of Econometrics, v. 87, n. 2, p. 271-301, 1998. ISSN 0304-4076. DOI: https://doi.org/10.1016/S0304-4076(98)00016-5. Acesso em: 15/10/2023.

com um processo estocástico próprio. A aplicação deste modelo também foge do escopo desse trabalho, pois a volatilidade será tratada não como algo latente não diretamente observável, mas sim como uma variável influenciável por demais fatores.

Na precificação de uma opção por Monte Carlo, a volatilidade constante não é exatamente um ponto negativo, pois ela é uma estimativa de variação média diária do ativo da data de precificação até o vencimento. O maior desafio é definir um número que faça sentido para a volatilidade em determinados cenários.

3.6.2) O Modelo de Black-Scholes e a Volatilidade Implícita.

Em 1973, a publicação de *The Pricing of Options and Corporate Liabilities*, por Black e Scholes[23], introduziu uma equação diferencial parcial que permitia calcular o preço de opções europeias. A ideia inicial é similar à explorada no modelo de precificação binomial; uma replicação dinâmica, onde uma carteira é continuamente ajustada para replicar o *payoff* da opção, evitando arbitragem.

Assume-se que o preço do ativo subjacente segue um movimento browniano geométrico com uma taxa de retorno constante e uma volatilidade constante. Também se assume que não há dividendos pagos, os mercados são eficientes e não existem custos de transação.

A fórmula de precificação de uma *call* (*C*) é dada por:

$$C(S,t) = S_t N(d_1) - Ke^{-r(T-t)}N(d_2) \qquad (21)$$

Onde:

- S_t: Preço atual do ativo subjacente.
- *K*: Preço de exercício.
- *T*: Tempo até a expiração da opção.
- *t*: Tempo atual.
- *r:* Taxa de juros livre de risco, anualizada.

$N(d_1)$ e $N(d_2)$ são os valores da função de distribuição cumulativa da distribuição normal padrão para d_1 e d_2, respectivamente:

[23] BLACK, Fischer; SCHOLES, Myron. The Pricing of Options and Corporate Liabilities. Journal of Political Economy, v. 81, n. 3, p. 637-654, 1973. Disponível em: http://www.jstor.org/stable/1831029. Acesso em: 16/10/2023.

$$d_1 = \frac{ln\left(\frac{St}{K}\right)+\left(r+\frac{\sigma^2}{2}\right)(T-t)}{\sigma\sqrt{T-t}} \quad (22)$$

$$d_2 = d_1 - \sigma\sqrt{T-t} \quad (23)$$

Onde:

- σ: Volatilidade anualizada do retorno do ativo subjacente.

A fórmula de precificação de uma *put* (P) é dada por:

$$P(S,t) = Ke^{-r(T-t)}N(-d_2) - S_tN(-d_1) \quad (24)$$

Apesar de sua formulação preceder as primeiras aplicações do método de Monte Carlo na precificação de opções e de apresentar algumas limitações – como a aplicabilidade restrita a opções europeias –, o modelo de Black-Scholes permanece amplamente utilizado no mercado financeiro contemporâneo.

Um conceito central discutido neste trabalho e diretamente abordado pela fórmula de Black-Scholes, é a volatilidade implícita. Considerando opções de mesmo vencimento, o preço do ativo subjacente, o *dividend yield*, a taxa de juros livre de risco e o prazo até o vencimento são constantes entre as opções. No entanto, é comum observar variações nos prêmios, em diferentes preços de exercício.

Devido a estratégias de risco específicas para certos ativos e cenários, os mercados atribuem diferentes níveis de volatilidade a diferentes graus de *moneyness*. Como a equação de Black-Scholes é diferencial, a volatilidade implícita para cada preço de exercício pode ser inferida mantendo-se as demais variáveis constantes, possibilitando a construção da curva de volatilidade, conhecida como *volatility skew*.

Nesta curva, o eixo horizontal representa os preços de exercício e o eixo vertical, os níveis de volatilidade implícita, expressos em percentagem. Uma mesma curva abrange tanto as *calls* quanto as *puts*, de forma que ambas possuem o mesmo preço de exercício ATM; uma *call* de preço de exercício *x* OTM é uma *put* ITM para o mesmo preço de exercício e; uma *put* de preço de exercício *x* OTM é uma *call* ITM para o mesmo preço de exercício.

Partindo de um princípio de não arbitragem, similar ao visto na Seção 3.4.3 de árvores binomiais, para um mesmo preço de exercício, a volatilidade implícita da *call* e da *put* tendem a ser os mesmos. O conceito de *put-call parity*, revela que a diferença

entre o preço da *call* e o preço da *put* é igual à diferença entre o preço do ativo subjacente e o valor presente do preço de exercício.

Em outras palavras, a compra de uma *call* e a venda de uma *put* de mesmo preço de exercício e vencimento, devem ter o mesmo retorno que comprar o ativo subjacente, financiando essa compra através do empréstimo do montante equivalente ao preço de exercício a valor presente. Esse trabalho não demonstrará exemplos desse conceito, de qualquer forma, se essas opções possuem volatilidades implícitas diferentes, a precificação delas são impactadas, *put-call parity* não se aplica e abre espaço para arbitragem

Em alguns casos reais, isso pode ocorrer, principalmente em mercados com menos negociações (ilíquidos), mas a arbitragem é dificultada pelos demais fatores, como menor liquidez no ativo em si, *spread* e custos operacionais.

Voltando às curvas de volatilidade, observa-se uma inclinação positiva na curva quando as *calls* OTM apresentam volatilidades implícitas mais altas que *puts* OTM, indicando que o mercado espera elevação nos preços. Já a inclinação negativa é vista onde as *puts* OTM exibem volatilidades implícitas maiores que *calls* OTM, refletindo preocupações com potenciais quedas nos preços. Relembrando que a volatilidade implícita maior implica em um prêmio mais caro.

Um padrão em forma de sorriso (*smile*) ocorre quando tanto as *calls* quanto as *puts* OTM e ITM apresentam volatilidades implícitas mais altas em comparação com as opções mais próximas do dinheiro (ATM), indicando uma elevada incerteza ou expectativa de movimentos significativos de preço em ambas as direções. Por outro lado, uma curva plana (*flat*) sugere uma volatilidade uniforme para todas as opções, independentemente do preço de exercício, denotando uma expectativa de mercado de movimentos de preço menos expressivos.

Em resumo, a volatilidade implícita é essencialmente uma projeção da volatilidade diária, anualizada, do ativo subjacente, da data de negociação até o vencimento da opção. Ela pode ser negociada no mercado como uma medida de risco, refletindo diferentes expectativas de variação de preços para um ativo específico.

Essa variação pode ser direcional, indicando, por exemplo, uma maior volatilidade implícita para preços de exercício mais distantes do dinheiro, sinalizando

uma tendência do mercado em se proteger ou especular em uma direção específica do preço do ativo.

A volatilidade implícita pode ser diferente em determinados níveis de preços de exercício, no entanto, a volatilidade realizada de um ativo é uma só. Isso torna mais difícil a interpretação de qual nível de volatilidade realizada o mercado espera de fato e dificulta a aplicação de um modelo de RNA para predição de volatilidade.

A penúltima seção do capítulo detalha um método para calcular uma estimativa única de volatilidade, que engloba todos os preços de exercício para determinado vencimento,

3.6.3) Índice de Volatilidade CVOL

Através da volatilidade implícita, as opções se tornaram importantes instrumentos para a avaliação do risco futuro. Contudo, as abordagens convencionais que utilizam preços de opções para mesurar riscos frequentemente não englobam a totalidade das informações disponíveis.

Por exemplo, a volatilidade implícita no dinheiro (ATM), é um indicador comumente utilizado para estimar o nível de risco atual refletido nos preços das opções, entretanto, conforme verificado na Seção 3.6.2, o mercado pode precificar diferentes níveis de volatilidade para um mesmo ativo e vencimento, dependendo do preço de exercício da opção. Para compreender na totalidade o risco precificado de um ativo, é necessário considerar todos os níveis de preços de exercício.

Nesse contexto, a CME Group Inc., grupo norte americano responsável pela operação de um dos principais mercados de derivativos do mundo, elaborou um método de calcular a volatilidade implícita que incorpora um espectro mais amplo de preços de opções e nomeou de CVOL.

A formulação envolve o cálculo da estimativa de variância abrangendo informações de toda a curva de volatilidade, ou seja, obtida considerando as *calls* e *puts* OTM para um vencimento específico. O modelo considera todas as opções OTM até o valor ATM, para montar uma curva na qual o eixo horizontal são os preços de exercício e o eixo vertical o valor do prêmio das opções, conforme demonstrado na Figura 3. A área da curva é a variância implícita para as opções desse ativo e

vencimento, logo, a raiz quadrada dessa variância é o desvio padrão para o período. Ao anualizar esse valor, tem-se a volatilidade implícita para todo o conjunto de dados.

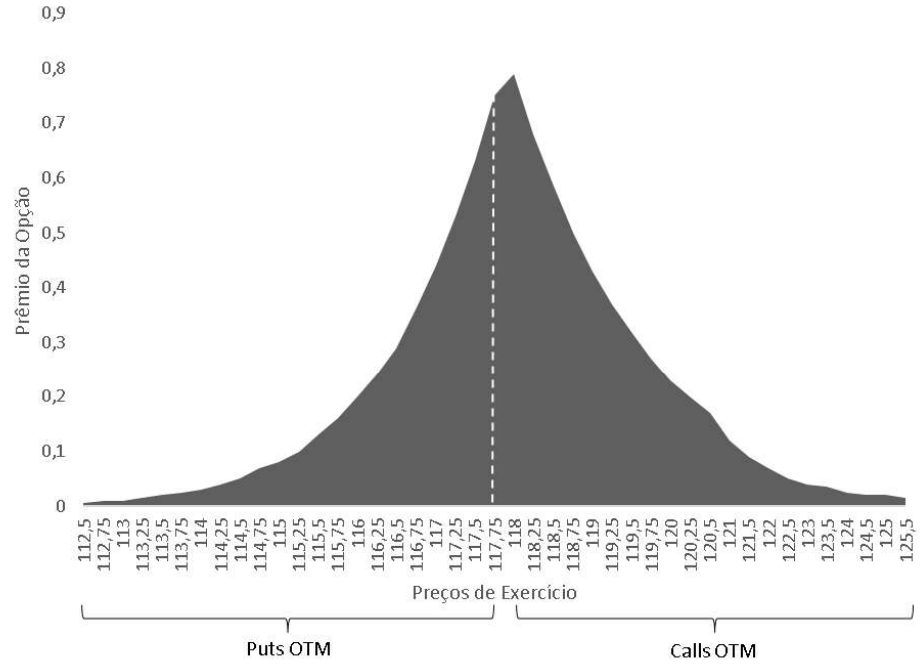

Figura 3 - Exemplo da área a ser calculada para encontrar a variância e, consequentemente, a volatilidade implícita dos preços de exercícios para um mesmo ativo e vencimento. Preço futuro (ATM) em 117,75.

Demonstrar completamente os cálculos do índice foge do escopo desse trabalho, mas resumidamente, as *calls* e *puts* OTM são organizadas em uma tabela de duas colunas (preços de exercício e prêmios). O prêmio utilizado ATM é o menor valor entre a *call* e *put* desse preço de exercício. Encontra-se a área de cada preço de exercício, multiplicando o prêmio (altura) pela variação do preço de exercício (base) e somando todos esses resultados. O resultado é levado a valor futuro pois, como visto na Equação 21 e 24 (Black-Scholes), as opções são trazidas a valor presente, depois, o valor é dividido pelo preço futuro do ativo ao quadrado, encontrando a variância.

A partir desse ponto, é possível aplicar as Equações 16 e 17 para calcular a volatilidade e atualizá-la. O resultado indica, de forma mais abrangente, a expectativa de volatilidade que o mercado precifica implicitamente nas opções. O CME Group

disponibiliza em seu *website*[24] *o* valor atual e histórico do CVOL para diferentes ativos, inclusive, para o ativo será utilizado no modelo prático.

3.7) Considerações Finais do Capítulo 3

Começando de maneira ampla com fundamentos gerais de derivativos e transcorrendo por temas mais específicos, como o índice de volatilidade implícita, o capítulo abordou os conceitos de mercados e derivativos necessários para compreender o objeto de análise do modelo de RNA a ser desenvolvido no capítulo 5. O intuito foi fornecer de maneira completa, até para leitores com menos familiaridade, as ferramentas necessárias para analisar os resultados do modelo de predição da volatilidade realizada.

Considerando um portifólio sem risco, o debate da expectativa da volatilidade pode não ser uma parte crucial da operação. Como no portifólio são negociadas quantidades do ativo subjacente junto às opções, mesmo que a volatilidade realizada seja diferente da volatilidade calculada no início, realizando ajustes nas quantidades do ativo posicionadas, é possível neutralizar esse risco diferencial.

No entanto, para um especulador e até para quem utiliza opções como *hedge* de operações fora do mercado financeiro - por exemplo, um produtor de soja que compra *puts* do contrato futuro de soja, para se proteger da queda do preço -, o valor da volatilidade é importante para decisões de investimento.

Com isso, conclui-se o capítulo 3. A seguir, o trabalho abordará os principais conceitos sobre inteligência artificial, modelos de redes neurais artificiais e como eles podem auxiliar na predição de volatilidade.

[24] CME GROUP. CME Group Volatility Indexes. Disponível em: https://www.cmegroup.com/market-data/cme-group-benchmark-administration/cme-group-volatility-indexes.html. Acesso em: 05/11/2023.

4) Modelos de Redes Neurais Artificiais

Supondo um problema onde há diferentes pontos (x,y) em um plano, mas não se sabe a função *f(x)* que produzem esses pontos. Ao encontrar ou construir uma aproximação dessa função, seria possível calcular o valor de y para determinado valor de x que não estava originalmente na base de dados.

Ainda que não exata, mas com uma boa aproximação da função real, calcular-se-ia, com certa segurança, os valores de y mesmo se a base de dados possua um pouco de ruído (aleatoriedade), pois a função capturaria o padrão geral dos dados, resultando em valores de y que, na maioria, não são perfeitos, mas ainda úteis para aplicação.

Logo, que precisamos para solucionar o problema é de um "aproximador" de funções e, de maneira geral, é esse o objetivo de uma rede neural artificial.

4.1) Fundamentos das Redes Neurais e do Modelo *Feedfoward*

Conforme o nome já sugere, redes neurais artificiais (RNAs) são estruturas inspiradas no cérebro humano, que utilizam algoritmos e funções matemáticas para aprender e reconhecer padrões complexos em dados. Segundo Nielsen (2015)[25], dentro das redes neurais, um neurônio é como uma função matemática, que recebe várias entradas de dados e produz um resultado (saída), comumente referida como "ativação".

Uma rede neural básica possui três tipos principais camadas de neurônios; a de entrada, onde os dados são incluídos; as intermediárias (conhecidas como camadas ocultas), onde ocorrem a maior parte dos cálculos; e a de saída, onde a rede apresenta seu resultado quantitativo para determinado problema.

O estudo sobre RNAs partirá de um dos modelos mais simples, para depois introduzir algumas variações. Conhecido pelo termo inglês *feedfoward,* com os neurônios totalmente conectados entre as camadas, nesse modelo, o neurônio de uma camada produz a saída para todos os outros neurônios da camada seguinte, em

[25] NIELSEN, Michael A. Neural Networks and Deep Learning. Determination Press, 2015. Disponível em: http://neuralnetworksanddeeplearning.com/. Acesso em: 08/10/2023.

um fluxo unidirecional, ou seja, sem quaisquer conexões de feedback para camadas anteriores.

O intuito das camadas é, ao adicionarmos dados dentro da camada de entrada, o modelo realizará cálculos para identificar padrões e apresentar a provável resposta ao problema na camada de saída.

As RNAS são capazes de realizar diferentes tarefas, abrangendo diferentes domínios e complexidades. Alguns principais exemplos dos tipos de tarefas realizadas por uma RNA são:

- Classificação Binária: Realiza a classificação dos dados em duas classes. Utilizado, por exemplo, para classificar e-mails em spam e não spam.
- Classificação Multiclasse: Classifica as entradas em mais de duas classes. Utilizado, por exemplo, para reconhecimento de fotos.
- Regressão: Prevê um valor contínuo de saída para o conjunto de dados. Utilizado, por exemplo, para prever preços de ativos financeiros.

Os neurônios da camada de entrada são as variáveis independentes iniciais do problema, logo, a ativação de um neurônio dessa camada é igual ao valor da variável. Cada neurônio da camada seguinte atribui diferentes pesos (*w*) às ativações da camada anterior (*x*). A soma ponderada das ativações mais um viés técnico (também conhecido como *bias*, *b*) passam por uma função de ativação (*f*), que gera uma saída, a ativação desse neurônio para as camadas seguintes.

As ativações dos neurônios das camadas posteriores a camada de entrada podem ser demostradas matematicamente pela Equação 25 abaixo:

$$a = f[\sum_{i=1}^{n}(x_i * w_i) + b] \quad (25)$$

A partir das ativações da camada intermediária anterior, as ativações dos neurônios da camada de saída apresentam os resultados preditos pelo modelo. Cada etapa do processo será detalhada nas próximas seções.

Abaixo, a Figura 4 refere-se a uma representação de uma rede neural artificial *feedfoward* totalmente conectada.

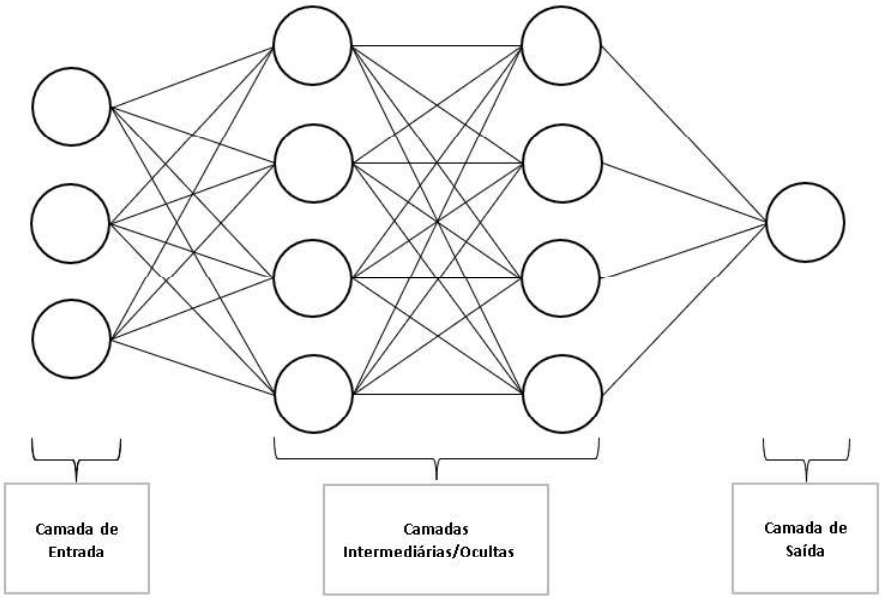

Figura 4 - Exemplo de um modelo de rede neural artificial feedfoward totalmente conectada.

No exemplo da Figura 4, a camada de entrada possuí três neurônios, há duas camadas ocultas com quatro neurônios cada e a camada de saída possui apenas um neurônio. As quantidades de neurônios por camada e de camadas ocultas podem variar em cada modelo, as quantidades demonstradas na Figura 4 foram definidas apenas para exemplificar um modelo.

Como cada neurônio envia uma ativação para cada neurônio da camada seguinte, onde é atribuído um peso, a soma ponderada é somada com um viés e, então, tem-se os parâmetros para a fórmula de ativação, considerando todos os pesos (60) e vieses (9) do exemplo, tem-se um total de 69 “nós” que fazem o modelo se comportar de maneiras diferentes, dependendo dos seus valores. O processo de aprendizagem refere-se a fazer com que a máquina que executa o modelo encontre os valores corretos para os pesos e vieses e consiga solucionar o problema proposto.

Como o processo de ativação, que inclui definir os pesos e vieses, ocorrem em cada neurônio, é possível organizá-los em uma expressão de vetores e matrizes. As ativações iniciais são organizadas em um vetor.

$$\begin{bmatrix} x_0 \\ \vdots \\ x_n \end{bmatrix} \qquad (26)$$

Os pesos para as ativações, definidos em cada neurônio da camada atual, são organizados em uma matriz, em que cada linha corresponde a conexão das ativações iniciais com cada neurônio da camada atual.

$$\begin{bmatrix} w_{0,0} & \cdots & w_{0,n} \\ \vdots & \ddots & \vdots \\ w_{k,0} & \cdots & w_{k,n} \end{bmatrix} \quad (27)$$

Os vieses de cada neurônio da camada atual são organizados em um vetor.

$$\begin{bmatrix} b_0 \\ \vdots \\ b_n \end{bmatrix} \quad (28)$$

Conforme visto na Equação 25, a matriz de pesos é multiplicada pelo vetor das ativações e, depois, somada ao vetor dos vieses.

$$\begin{bmatrix} w_{0,0} & \cdots & w_{0,n} \\ \vdots & \ddots & \vdots \\ w_{k,0} & \cdots & w_{k,n} \end{bmatrix} \begin{bmatrix} x_0 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_0 \\ \vdots \\ b_n \end{bmatrix} \quad (29)$$

Após a aplicação da função de ativação na Equação 29, é possível demonstrar em uma expressão relativamente simples a transição completa de ativações de uma camada para a outra. “Traduzir” o modelo em operações com matrizes facilita até na programação computacional, visto que muitas linguagens de programação possuem ferramentas que trabalham essas operações de maneiras relativamente simples.

$$Ativação\ (a) = f\left(\begin{bmatrix} w_{0,0} & \cdots & w_{0,n} \\ \vdots & \ddots & \vdots \\ w_{k,0} & \cdots & w_{k,n} \end{bmatrix} \begin{bmatrix} x_0 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} b_0 \\ \vdots \\ b_n \end{bmatrix}\right) \quad (30)$$

$$a = f(Wx + b) \quad (31)$$

Onde:

- x: Vetor de entrada/ativações da camada anterior.
- W: Matriz de pesos associados à cada x.
- b: Vetor de viés.
- *f:* Função de ativação.

Em essência, uma rede neural pode ser entendida como um “aproximador” universal de funções ao tentar estimar a relação funcional subjacente entre um

conjunto de entradas e saídas correspondentes, especialmente quando a forma exata dessa relação é complexa e desconhecida.

4.2) Detalhes das Camadas de uma Rede Neural Artificial

4.2.1) Camada de Entrada

A camada de entrada é a primeira etapa no fluxo de informação de uma RNA e, como o nome sugere, é onde os dados "entram" no modelo para serem processados pelas camadas subsequentes. Essa camada que recebe os vetores das características (conhecidos pelo termo em inglês, *features*), que são as variáveis independentes incluídas ao modelo para treinamento e predição.

O número de neurônios nesta camada, geralmente, corresponde à dimensionalidade dos seus dados de entrada. Por exemplo, se cada amostra de dados é um vetor com 10 elementos (10 variáveis independentes), então, a camada de entrada teria 10 neurônios.

Nenhum cálculo é realizado nessa etapa e a camada de entrada não possui uma função de ativação, seus neurônios apenas repassam os dados para a próxima camada. De qualquer forma, para um bom desempenho do modelo, é importante que os dados incluídos tenham passado por algum tipo de normalização ou pré-processamento antes de serem alimentados na rede.

4.2.2) Camadas Intermediárias/Ocultas

São as camadas onde ocorrem a maior parte dos cálculos de ativação, vistos na Seção 4.1, como a equação 30. Um modelo de RNA pode conter uma ou mais camadas ocultas, dependendo do objetivo e dos dados com os quais o modelo está trabalhando.

Em linhas gerais, as camadas intermediárias têm a função de capturar padrões complexos entre cada variável, que não seriam facilmente identificáveis para outros métodos (por exemplo, para uma regressão linear). Elas efetuam transformações nas informações de entrada, com o objetivo de simplificar a tarefa computacional que será realizada pela camada de saída.

Não há um número exato para a quantidade ideal de camadas ocultas e de neurônios em cada uma dessas camadas. Ao estruturar um modelo, normalmente,

são testadas diferentes quantidades, para identificar quais produzem os melhores resultados e se encaixam melhor na capacidade computacional da máquina.

De qualquer forma, observa-se um padrão em diferentes modelos. Quando há mais camadas intermediárias e/ou mais neurônios nas camadas que o necessário para o sistema, o modelo pode ajustar demais aos dados de treinamento, capturando mais um "ruído" do que padrões gerais dos dados, problema chamado pelo termo em inglês "*overfitting",* detalhado na Seção 4.4.5. Além disso, o modelo pode enfrentar dificuldades para atualizar seus pesos, que leva a uma aprendizagem menos eficiente e resultados insatisfatórios.

Quando há menos camadas intermediárias e/ou menos neurônios nas camadas que o necessário, o modelo fica muito simples e tem dificuldades em capturar relações complexas entre as os dados, problema chamado pelo termo em inglês "*underfitting*", detalhado também na Seção 4.4.5.

4.2.3) Camada de Saída

É a camada final do modelo, onde a rede realiza os últimos cálculos de ativação e fornece suas previsões ou classificações. O número de neurônios na camada de saída depende da tarefa específica, por exemplo:

- Classificação Binária: Geralmente, utiliza-se um único neurônio e uma função de ativação que produz uma saída entre 0 e 1. O valor pode ser interpretado como a probabilidade de pertencer a uma das duas classes.
- Classificação Multiclasse: Geralmente, o número de neurônios corresponde ao número de classes. Por exemplo, um modelo que busca classificar imagens em 10 diferentes categorias, ele teria 10 neurônios na camada de saída.
- Regressão: Geralmente, há apenas um neurônio que fornece um valor contínuo.

4.3) Detalhes da Equação de uma Rede Neural Artificial

A Seção 4.1 introduziu a Equação 31. Através dela, os neurônios realizam os cálculos necessários para tenta solucionar o problema proposto. A Seção 4.3 desdobrará cada aspecto da Equação 31.

4.3.1) Pesos

Dentro da arquitetura de uma RNA *feedfoward* totalmente conectada, os pesos sinápticos são parâmetros atribuídos para cada ativação da camada anterior e ajustados iterativamente, com o intuito de identificar e codificar padrões nas variáveis independentes de entrada, que são predicativos para o resultado desejado do problema abordado.

4.3.2) Viés Técnico (*Bias*)

Assim como os pesos, o viés técnico é outro parâmetro do modelo. Seu propósito é conferir um grau adicional de liberdade à função de ativação, facilitando a ativação dos neurônios mesmo na ausência de entradas ou quando as entradas, ponderadas pelos respectivos pesos, não são suficientes para alcançar o limiar requerido para ativação.

Em um plano, o viés possibilita o deslocamento da curva característica da função de ativação ao longo do eixo horizontal (eixo x). Esse deslocamento é fundamental para a capacidade da rede de aprender padrões intrincados, dado que, em muitas situações, é necessário ajustar o ponto onde a função de ativação atinge um valor específico para se adequar melhor à distribuição dos dados.

Por exemplo, em um problema de classificação binária, o modelo trabalha com uma função de ativação que restringe os valores em um intervalo entre 0 e 1 no eixo y. O viés técnico pode determinar o ponto no eixo x no qual a função produzirá um valor de saída específico, como 0,5, ajudando o modelo definir se a ativação está mais próxima da classificação 0 ou 1. Isso beneficia o processo de aprendizado, especialmente em situações em que as classes não estão centradas ao redor do zero ou onde a simetria entre as classes não é desejável.

Em síntese, o viés técnico colabora na modulação da sensibilidade da função de ativação às entradas da rede, permitindo um ajuste mais fino que favorece a aprendizagem de padrões complexos presentes nos dados da entrada.

4.3.3) Funções de Ativação

As funções de ativação introduzem não-linearidades ao modelo. Esta não-linearidade permite que a rede neural aprenda e modele relações complexas e intricadas entre os dados de entrada e de saída.

A linearidade é referente à relação proporcional de variação entre dois dados do modelo, representando por um gráfico de linha reta. Por exemplo, uma relação que, ao dobrar uma variável, a outra dobra junto, o que não ocorre na maior parte dos casos reais.

Sem as funções de ativação, uma rede neural, independentemente de quantas camadas possuísse, funcionaria apenas como um classificador linear, incapaz de resolver problemas além daqueles que são linearmente separáveis, limitando sua aplicação em problemas reais.

Outro objetivo das funções é organizar os dados de uma maneira que torne a aprendizagem do modelo mais fácil e traduza melhor os valores de saída. Por exemplo, para limitar uma ativação a assumir um número dentro um intervalo, como de 0 a 1, utiliza-se uma função específica para isso.

Uma mesma camada pode possuir diferentes funções de ativação entre seus neurônios e, em modelos mais elaborados, até um mesmo neurônio pode possuir diferentes funções de ativação. A Seção 4.5.2 introduz um tipo de RNA que aborda esse conceito.

4.3.4) Principais Funções de Ativação

- Função Linear: A mais simples dentre as alternativas, onde a saída é proporcional à entrada. Esta função não realiza cálculos nos dados de entrada, assim, a saída é exatamente igual à entrada.

$$f(x) = x \tag{32}$$

- Função Sigmoide: Converte valores de entrada em uma faixa entre 0 e 1, com uma curva em forma de “s”. Normalmente utilizada apenas em camadas de saída para problemas de classificação binária e de probabilidades, pois mapeia qualquer intervalo entre 0 e 1. Em camadas ocultas, não é tão utilizada devido ao problema de desaparecimento do gradiente (mais detalhes na Seção 4.4.4).

$$f(x) = \frac{1}{1+e^{-x}} \quad (33)$$

- Função Tanh (Tangente Hiperbólica): Semelhante à sigmoide, mas mapeia os valores de entrada para uma faixa entre -1 e 1. Geralmente utilizada em camadas ocultas quando a centralização (normalização) dos dados é mais crítica.

$$f(x) = tanh\,(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (34)$$

- Função ReLU (Unidade Linear Retificada): Mantém os valores positivos como estão (propriedade conhecida como "não saturação") e converte valores negativos em zero. É mais eficiente na convergência do treinamento e em mitigar o desaparecimento do gradiente, comparado com funções sigmoide e tanh.

$$f(x) = \ max\,(0, x) \quad (35)$$

- Função Leaky ReLU: Semelhante à ReLU, mantendo os valores positivos como estão, mas permite um pequeno gradiente quando x é menor que zero. Normalmente é utilizada para prevenir a morte de neurônios em RNAs mais profundas (ou seja, com mais camadas ocultas) e convolucionais. Um neurônio "morre" quando para de responder a variações nos dados de erro.

$$f(x) = \ max\,(\alpha x, x) \quad (36)$$

Onde α, conhecido como taxa de vazamento, é um pequeno valor constante (por exemplo, 0,01). A eficiência do modelo depende da escolha de um bom parâmetro α, deixando sua aplicação ligeiramente mais complexa.

Há outras funções que são úteis para tarefas específicas, como exemplo, a função softmax, utilizada para problemas de classificação. Devido ao escopo do trabalho, não há necessidade em apresentar essas demais funções.

4.4) Outros Hiperparâmetros

Os hiperparâmetros são todas as configurações ajustáveis em um algoritmo de aprendizado de máquina, definidas antes do processo de treinamento e que governam como o modelo é estruturado e treinado. Diferentemente dos parâmetros do modelo, como os pesos e vieses técnicos, que são ajustados ao longo do treinamento, os

hiperparâmetros devem ser pré-estabelecidos e são mais contantes ao longo do treinamento.

As seções anteriores já apresentaram alguns hiperparâmetros, como as camadas, números de neurônios e as funções de ativação. Essa seção detalhará outros hiperparâmetros, também essenciais para estruturar um modelo de RNA.

4.4.1) Número de Épocas (*Epochs*)

Uma época refere-se a uma passagem completa do algoritmo pelo conjunto de dados de treinamento. Durante cada época, o algoritmo de aprendizado ajusta os pesos da rede com base no erro entre as saídas esperadas e as reais.

Supondo que o modelo tenha 100 conjuntos de dados para treinamento, uma época representa uma passagem completa do modelo por todos esses dados.

Nas primeiras épocas, é comum observar uma melhoria significativa no desempenho do modelo, à medida que ele começa a aprender com os dados. Com o passar das épocas, o modelo tende a convergir para um estado onde faz previsões mais precisas e, idealmente, a diferença entre o erro de treinamento e o erro de teste diminuem (mais detalhes sobre essa dinâmica na Seção 4.4.5).

Não há um número padrão de épocas aplicável em todos os modelos, ele varia dependendo da complexidade do modelo, do tamanho e da natureza do conjunto de dados e do algoritmo específico de aprendizado utilizado. Assim como os demais hiperparâmetros, geralemente, o número de épocas é ajustado conforme experimentação.

Um número de épocas baixo pode levar o modelo a não capturar a complexidade dos dados (*underfitting)*, por outro lado, muitas épocas podem levar o modelo a se ajustar demais aos dados (*overfitting)*.

4.4.2) Tamanho do Lote (*Batch Size*)

Às vezes, uma época pode conter muitos dados para o modelo avaliar de uma vez, então, os dados precisam ser divididos em subgrupos (lotes). O tamanho do lote é um hiperparâmetro que determina a quantidade total de conjunto de dados presentes nos subgrupos de treinamento, que serão utilizados em uma única iteração do algoritmo.

O tamanho máximo do lote é igual ao número de conjunto de dados e seu menor tamanho é igual a um. Quanto maior o lote, o gradiente do modelo (conceito explorado na Seção 4.5.2) é baseado em mais dados, que leva a uma estimativa mais precisa, mas demanda mais trabalho computacional e pode trazer problemas com a generalização do modelo (*overfiting*).

No exemplo de 100 conjuntos de dados, ao dividi-los em 2 lotes, o "tamanho" seria de 50 conjuntos em cada.

4.4.3) Iterações

As iterações são os números de lotes necessários para completar uma época. No exemplo de 100 conjuntos de dados, 2 lotes de 50 conjuntos em cada, o modelo precisaria de 2 iterações para completar uma época.

Há outros importantes hiperparâmetros, como a taxa de aprendizado e a regularização do modelo, que serão introduzidos e explorados ao longo do capítulo.

4.5) Como as Redes Neurais Artificiais Aprendem?

Recapitulando, a aprendizagem em redes neurais refere-se ao processo no qual o modelo busca definir os valores corretos de pesos e viéses para cada ativação, aproximando uma função universal para o problema em questão. Após treinada, a função precisa apresentar um resultado satisfatório de predições em um conjunto de dados não apresentado no treinamento, chamado de conjunto de teste.

Dessa forma, todo modelo necessita de um conjunto de variáveis independentes (*features)* para treino e outro para teste. Além dos vetores de entrada (variáveis independentes), os conjuntos de dados necessitam de um vetor de saída, com os valores esperados que a rede neural indique (variáveis dependentes).

Por exemplo, em um conjunto de dados para um problema de classificação de imagens, além das próprias imagens para treinamento e teste (variáveis independentes), é necessário indicar qual classificação (variável dependente) que o modelo deve responder para cada imagem do conjunto.

Para a predição da volatilidade realizada de um ativo em um determinado período, o conjunto de variáveis independentes - exógenas e endógenas - que o modelo tentará correlacionar com a volatilidade seriam as *features* e o valor da

volatilidade realizada de fato naquele determinado período seria o valor esperado (variável dependente).

Em síntese, após a conclusão do treinamento, é considerado que o modelo aprendeu e elaborou uma função satisfatória quando o algoritmo tem um bom desempenho na predição em novas entradas não vistas anteriormente, conceito conhecido como generalização.

As próximas seções detalharão mais sobre cada etapa desse processo.

4.4.1) Função Erro

Ao iniciar o treinamento de uma rede neural, os pesos e vieses iniciais podem ser atribuídos de maneira aleatória. Esta inicialização aleatória conduz ativações na camada de saída que são igualmente aleatórias e, provavelmente, desvinculadas dos resultados esperados ou da realidade subjacente dos dados.

Para orientar o modelo em direção a estimativas mais precisas, implementa-se uma função de erro (também chamada de função custo ou perda). Esta função mensura o quão distante as predições do modelo estão dos valores reais desejados. Através dela, o modelo realizará ajustes nos parâmetros, com intuito de minimizá-la e, consequentemente, realizar predições melhores (mais detalhes na Seção 4.4.2).

Uma forma comum da função erro é a soma dos erros quadráticos, onde cada erro é a diferença entre o valor previsto pela ativação da camada de saída e o valor real ou esperado. Matematicamente, a função de erro quadrática de um único conjunto ser expressa como:

$$E = \sum_{i=1}^{n} (a_i^{(s)} - \hat{a}_i^{(s)})^2 \qquad (37)$$

Onde $a_i^{(s)}$ é o valor predito pela rede neural e $\hat{a}_i^{(s)}$ é o valor esperado para a i-ésima amostra da camada de saída s. Em outras palavras, a função erro fornece uma medida agregada da diferença entre as predições do modelo e os valores esperados para todas as amostras processadas.

Quando a rede neural tem um único neurônio de saída, o erro é calculado com base na diferença entre a predição desse neurônio e do valor esperado. Se a rede possui múltiplos neurônios de saída, o erro é a soma das diferenças quadráticas individuais para cada neurônio de saída.

Geralmente, o erro será maior quando as predições estão mais distantes dos valores reais. Conforme novas amostras são introduzidas durante o treinamento, o erro médio é recalculado, refletindo a média dos erros quadráticos de todas as amostras processadas até o momento.

Logo, a função de Erro Quadrático Médio (*Mean Squared Error*, MSE) da rede neural é expressa por:

$$MSE = \frac{\sum_{i=1}^{n} E}{n} \quad (38)$$

Onde *n* é o número de conjunto de amostras de dados.

É possível variar a função erro, para se adequar melhor a diferentes problemas. Por exemplo, a função da Raiz Quadrada do Erro-Médio (*Root Mean Squared Error*, RMSE), demonstrada na Equação 39, extrai a raiz quadrada da Equação 38 (MSE). Como o MSE são os erros levados ao quadrado, os resultados ficam fora de escala, podendo dificultar a interpretação. Utilizar a raiz quadrada retorna o erro à escala original.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} E}{n}} \quad (39)$$

Outra opção é utilizar apenas as diferenças simples entre os valores preditos e reais, deixando o erro em valores absolutos. Essa função, demonstrada na Equação 40 abaixo, é chamada de Erro Médio Absoluto (*Mean Absolute Error*, MAE), em contrapartida do MSE e RMSE, que penalizam erros maiores, o MAE penaliza todos os erros na mesma proporção.

$$MAE = \frac{\sum_{i=1}^{n} \left| a_i^{(s)} - \widehat{a_i}^{(s)} \right|}{n} \quad (40)$$

Há uma diferença entre a função erro que o modelo tentará minimizar e as utilizadas como métricas de avaliação. Por exemplo, um modelo pode utilizar a MSE como função erro e a partir dos seus resultados, realizará os ajustes nos parâmetros com intuito de minimizar o erro. No entanto, como o MSE produz resultados fora de escala, utiliza-se as funções RMSE e MAE nas predições finais do modelo apenas para avaliá-lo.

4.4.2) Descida de Gradiente

O papel central inicial do algoritmo de aprendizado é ajustar os pesos e vieses da rede neural de forma a minimizar a função erro. Utilizando algoritmos, como o gradiente descendente, a rede neural é capaz de aprender iterativamente, alterando os pesos e vieses para produzir predições cada vez mais precisas.

O gradiente de uma função qualquer f(x) são as derivadas parciais de cada variável da função, denotado por ∇f(x), indica a direção do maior aumento da função. Por exemplo, para a Equação 41, o gradiente é $\nabla f(x, y) = [2x, 2y]$.

$$f(x, y) = x^2 + y^2 \qquad (41)$$

Na descida do gradiente, o modelo se move na direção oposta ao gradiente para encontrar o ponto crítico, subtraindo um múltiplo do gradiente do ponto atual. Para a Equação 41, no ponto (1,1), o gradiente em *f(1,1)* é [2,2], ou seja, indica que a função crescerá mais rapidamente no ponto (2,2). Dessa forma, para o modelo minimizar o valor, precisa se mover direção oposta.

Para evitar com que o modelo "salte" pelos pontos críticos e deixe a descida mais eficiente, ao invés de subtrair a diferença do gradiente e seu vetor, utiliza-se uma taxa de atualização (aprendizado) η, que é multiplicada ao gradiente. O resultado é subtraído do vetor, conforme a Equação 42.

$$(x', y') = (x, y) - \eta[x, y] \qquad (42)$$

A definição do valor da taxa de aprendizado η pode ser feita por experimentação ou com alguma técnica de otimização (exemplo demonstrado na Seção 4.5.3). Uma taxa de aprendizado alta acelerara o treinamento, mas pode não encontrar pontos críticos ao "saltar" sobre os valores. Uma taxa baixa leva a ajustes mais precisos, mas demanda mais tempo e aumenta o risco do modelo estacionar em um mínimo local (mais detalhes adiante).

No exemplo da Equação 41, utilizando $\eta = 0{,}1$, o modelo subtrairia $0{,}1 * 2 = 0{,}2$ de cada valor, chegando em um novo ponto de $1 - 0{,}2 = 0{,}8$, (0,8; 0,8). O processo seria repetido até encontrar um ponto crítico ou atingir o número máximo de iterações definidas.

Ponto importante, note que o modelo ajusta seus parâmetros a partir das iterações, ou seja, em cada lote do conjunto de treinamento (e não em cada época), o modelo realiza os ajustes e calcula o novo erro de treinamento. Após processar todos os lotes e, consequentemente, todo o conjunto de treinamento, ainda dentro da mesma época, o modelo é então testado com um conjunto separado para essa tarefa. Apenas após essa etapa que se conclui a época vigente e o modelo segue para a próxima época, com os parâmetros utilizados no último lote de treinamento.

Pode ser simples de deduzir o ponto mínimo do exemplo da Equação 41, mas no caso das redes neurais artificiais, que cada variável da equação é um peso ou viés, a tarefa toma dimensões mais desafiadoras.

Os pontos críticos são aqueles onde o gradiente (ou derivada) da função é zero e, na descida de gradiente, podem ser:

- Ponto mínimo local: Onde uma função atinge um valor menor do que todos os outros valores na sua vizinhança imediata, mas não necessariamente o menor de toda a função.
- Ponto mínimo global: Onde a função atinge o seu valor mais baixo em toda a sua extensão.
- Ponto de sela: Posição onde a função tem uma inclinação nula, mas não se trata de um extremo local. É o ponto sobre uma superfície na qual a elevação é máxima numa direção e mínima noutra direção.

A otimização em aprendizado profundo frequentemente envolve funções com muitos mínimos locais e pontos de sela, o que torna a tarefa de encontrar um mínimo global mais trabalhosa.

Aplicando esses conceitos às RNAs, o gradiente é a derivada parcial da função erro em relação a todos os pesos e vieses da rede. Esse é o “gradiente” que o modelo busca minimizar durante o treinamento.

$$\theta^{t+1} = \theta^{t} - \eta \nabla J(\theta) \qquad (43)$$

Onde:

- θ: Parâmetros (pesos e vieses) do modelo. Expressa o conjunto de matrizes de pesos e vetores de vieses. É possível expressar $\theta = \{W1, b1, W2, b2, \dots, Wn, bn\}$.
- η: Taxa de aprendizagem.
- $\nabla J(\theta)$: Gradiente da função erro J em relação aos parâmetros θ. É um vetor que contém as derivadas parciais de J em relação às matrizes de pesos e vetor de vieses.

4.4.3) Retropropagação (*Backpropagation*)

Ao apresentar os fundamentos gerais sobre a função erro e métodos para sua minimização, o trabalho segue com a análise do algoritmo de otimização utilizado no modelo para fazê-lo "aprender". Trata-se do *backpropagation,* cujo objetivo é calcular o gradiente da função erro em relação a cada peso do modelo, fornecendo a direção na qual os pesos devem ser ajustados para minimizar a função de custo.

Em uma rede *feedfoward*, o processo começa com a propagação para a frente, onde as entradas são repassadas para a camada de saída através dos neurônios, gerando as saídas e erros iniciais. A medida de erro é, então, propagada de volta pela rede (por isso o termo "*backpropagation*"), camada por camada, retrocedendo do final para o início da rede. Durante esse processo, os gradientes da função erro em relação a cada peso são calculados utilizando a regra da cadeia.

Em termos gerais, a regra da cadeia no cálculo diferencial é uma ferramenta para calcular a derivada de funções compostas. Se há duas funções, uma composta na outra, a regra da cadeia permite calcular a derivada dessa função composta. Embora os detalhes matemáticos fundamentais da regra não sejam aqui explorados, é importante entender seu papel nas RNAs.

Durante o processo de *backpropagation*, a regra de cadeia é aplicada pois cada neurônio (ou função) não opera isoladamente, mas depende da saída de outros neurônios. Assim, a regra da cadeia é utilizada para calcular as derivadas parciais dos pesos e vieses em relação ao erro, necessárias para ajustá-los de forma eficiente, tendo em vista a natureza interconectada das funções na rede.

Em síntese, um modelo pode alterar a ativação de um neurônio ao alterar os pesos que esse determinado neurônio atribuiu às ativações da camada anterior;

alterando seu viés técnico ou; voltando para as camadas anteriores e ajustando novamente esses fatores, até chegar na camada inicial.

A partir da função erro do conjunto de treinamento e do gradiente descendente, o algoritmo determina o que e quanto alterar, priorizando o ajuste das variáveis que mais impactam o modelo. O modelo "verifica" quais as variáveis de maior peso ao analisar o impacto da alteração delas no valor do erro.

O algoritmo realiza os ajustes até encontrar um ponto crítico ou ao atingir um número máximo de iterações. É importante realizar o treinamento com diferentes exemplos, para o modelo aprender impactos gerais das variáveis independentes e não apenas um único cenário, ficando mais eficiente na predição.

4.4.4) Principais Problemas na Minimização do Erro

Há dois principais problemas em relação aos gradientes, que ocorrem principalmente em redes neurais mais profundas:

- Desaparecimento do Gradiente: ocorre quando os gradientes se aproximam de zero durante o *backpropagation,* sem encontrar um ponto crítico. Isso é mais comum em redes neurais profundas, quando utilizado funções como a sigmoide e tanh, que tendem a produzir gradientes menores.

  À medida que o erro é propagado de volta pela rede, esses pequenos gradientes são multiplicados várias vezes, tornando-se insignificantes. Em outras palavras, o gradiente faz com que o modelo realize passos muito pequenos de ajuste, chegando ao número máximo de iterações antes de encontrar um ponto crítico. Medidas comuns para evitar esse problema são funções como ReLU, ajustes nas iniciações de pesos e outras medidas de normalização.

- Explosão do Gradiente: O oposto do desaparecimento do gradiente, acontece devido à multiplicação do gradiente por valores altos ao longo de várias camadas, o que pode levar a atualizações de peso muito grandes e dificulta o modelo encontrar um ponto crítico. Pode ser evitado limitando o valor máximo de gradientes durante o treinamento, utilizando taxas de aprendizagem adaptativas, ajustes nas iniciações de pesos e outras medidas de normalização.

4.4.5) Avaliando a Capacidade de uma Rede Neural Artificial

Recapitulando alguns conceitos abordados na Seção 4.4, o objetivo geral de um modelo de RNA é encontrar uma função generalizada para determinado conjunto de dados. Esse conjunto, com variáveis independentes e dependentes, é dividido em amostras para treinamento e outras para testes do modelo.

A partir de iterações com o conjunto de treinamento, o método de aprendizagem consiste em minimizar a diferença entre o valor predito e o esperado (chamado de erro), através de ajustes nos pesos e vieses da rede.

Há uma distinção entre erro do conjunto de dados de treinamento e o de teste. O processo de *backpropagation* consiste em minimizar erro de treinamento, no entanto, o modelo torna-se eficiente e aplicável ao minimizar o erro do conjunto de teste, conhecido também como erro de generalização. Goodfellow, Bengio, Courville (2016) [26], discutem o processo de separação de dados e da avaliação da "capacidade" de um modelo.

O erro de teste refere-se à capacidade de um modelo de aprendizagem de máquina ter bons resultados em dados não vistos durante o treinamento. O valor do erro no teste é estimado medindo o desempenho do modelo em um conjunto de valores separados para essa função e não utilizado durante o treinamento.

Por sua vez, a capacidade de um modelo é considerada, informalmente, como a sua habilidade de se adequar a uma ampla variedade de conjunto de dados. Em outras palavras, a capacidade pode ser medida considerando-se a simplicidade e (ênfase no "e") a eficácia com que ele se ajusta aos dados de treinamento e teste.

Técnicas como da validação cruzada podem ser úteis para avaliar a capacidade de generalização de um modelo de aprendizado de máquina. Ela estima como o modelo se comportará em um conjunto de dados independente e ajuda a mitigar problemas de ajuste. O processo envolve dividir os conjuntos de variáveis independentes em múltiplos subconjuntos. Em cada iteração da validação cruzada, diferentes subconjuntos são usados para treinamento e teste. Isso permite que o

[26] GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. Deep Learning. Cambridge: MIT Press, 2016.110.

modelo seja testado em várias configurações de dados, proporcionando uma avaliação mais robusta de sua capacidade de generalização.

Há diferentes maneiras de dividir os subconjuntos de dados, mas os detalhes não serão abordados nesse trabalho. De qualquer forma, essa divisão oferece uma avaliação mais realista de como o modelo se comportará com dados novos e não vistos.

A avaliação pode ser feita a partir da comparação dos desempenhos nos conjuntos de treinamento e teste, ao identificar como o modelo está se ajustando aos dados de treinamento. Para aprofundar nesse conceito, segue um exemplo de conjunto de dados qualquer, para modelo com duas variáveis, conforme apresentado na Figura 5.

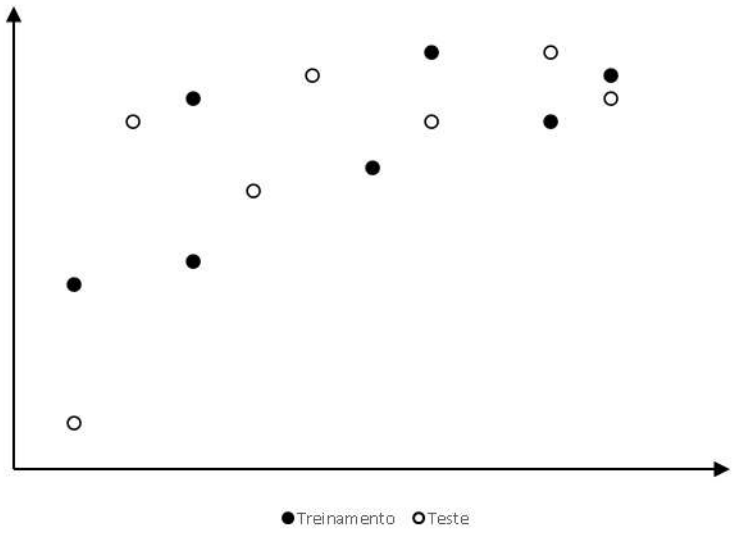

Figura 5 - Exemplo de um conjunto de dados qualquer. Cada ponto é um dado do conjunto.

Um modelo de RNA teria o objetivo de encontrar uma função universal para esse conjunto, que relacione o eixo horizontal com o vertical. Na prática, o modelo possuiria apenas uma variável independente, com os valores no eixo horizontal x, que determinaria o resultado da variável dependente, exposta no eixo y. Supondo que a função desejada seria representada pela curva traçada da Figura 6.

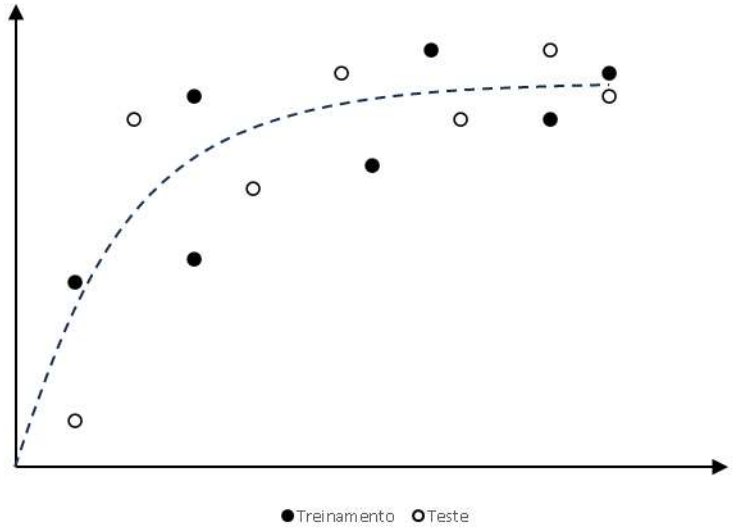

Figura 6 - Exemplo de um conjunto de dados com a curva da função desejada.

Para ilustrar o processo de treinamento do modelo, foram separados apenas os dados do conjunto de treinamento, conforme a Figura 7.

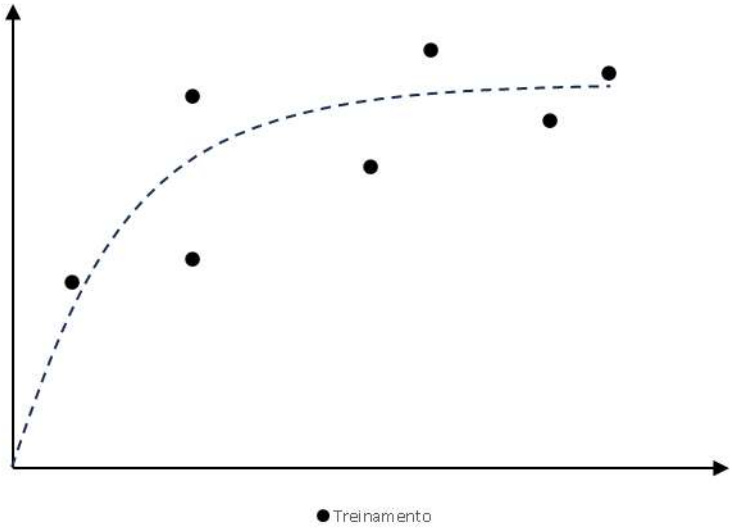

Figura 7 - Exemplo de um conjunto de dados para treinamento com a curva da função desejada.

Supondo que, nas primeiras interações, o modelo resultou em uma regressão linear pelo Método dos Mínimos Quadrados (MMQ), calculando uma reta que minimiza o erro ou, como o nome do método sugere, que minimiza a soma dos quadrados das diferenças entre os valores estimados e reais, conforme a Figura 8.

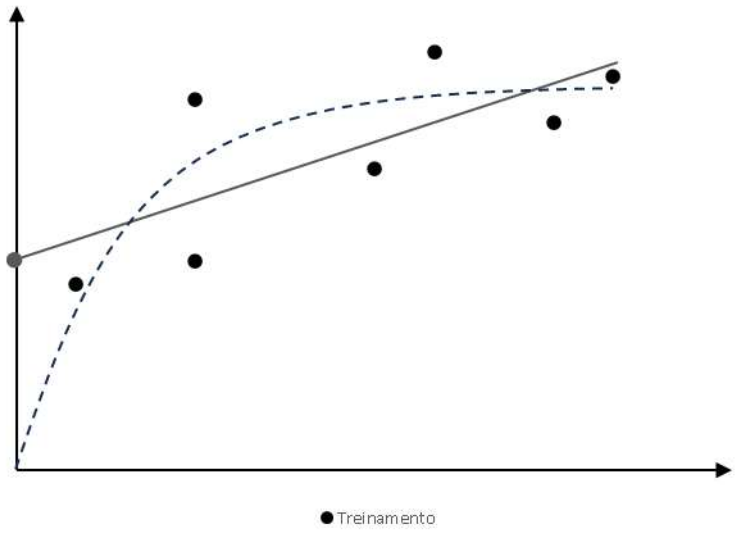

Figura 8 - Exemplo de um conjunto de dados para treinamento com a reta da regressão linear em tom mais claro e a função desejada traçada.

Ao comparar com a curva da função desejada, observa-se que o modelo não conseguiu traçar uma reta que dívida satisfatoriamente os conjuntos de dados. Mesmo alterando a posição e inclinação, não há qualquer reta que aproxime o erro do treinamento à zero.

Esse problema é conhecido como *underfitting*, ou "subajuste", no qual o modelo não consegue capturar adequadamente a complexidade dos dados. As causas incluem, mas não se limitam, à um modelo de RNA muito simples e com hiperparâmetros mal estabelecidos. Por exemplo, um modelo com poucas camadas ocultas, poucos neurônios, restrições de regularização, poucas épocas e com funções de ativação inadequadas.

Também é possível referir-se ao *underfitting* como a incapacidade do modelo em reduzir o viés estatístico (não confundir com o viés técnico, presente na equação de ativação das RNAs). Por conta de a função criada pelo modelo ser uma linha reta que não se "curva" como a função esperada para o conjunto, é dito que ela possui um viés estatístico alto.

Supondo que, após ajustes, o algoritmo diminui o viés estatístico e foi capaz de traçar uma função representada por uma curva ondulada, que passa exatamente sobre todos os pontos, conforme a Figura 9.

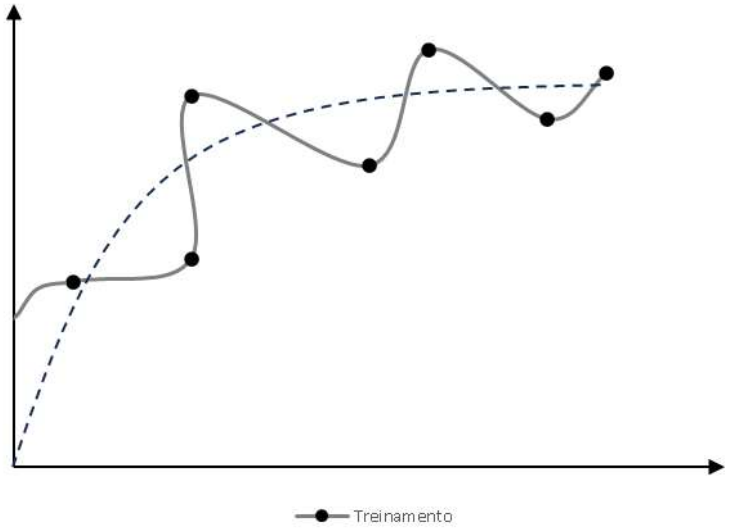

Figura 9 - Exemplo de um conjunto de dados para treinamento com a curva ondulada de exemplo em tom mais claro e a função desejada traçada.

Agora, o erro de treinamento é zero, não houve quaisquer diferenças entre os valores preditos e esperados. No entanto, repara-se que a curva ainda é diferente da esperada para a função. Ao introduzir os conjuntos de dados de teste, fica mais evidente que o resultado ainda não é ideal, conforme a Figura 10.

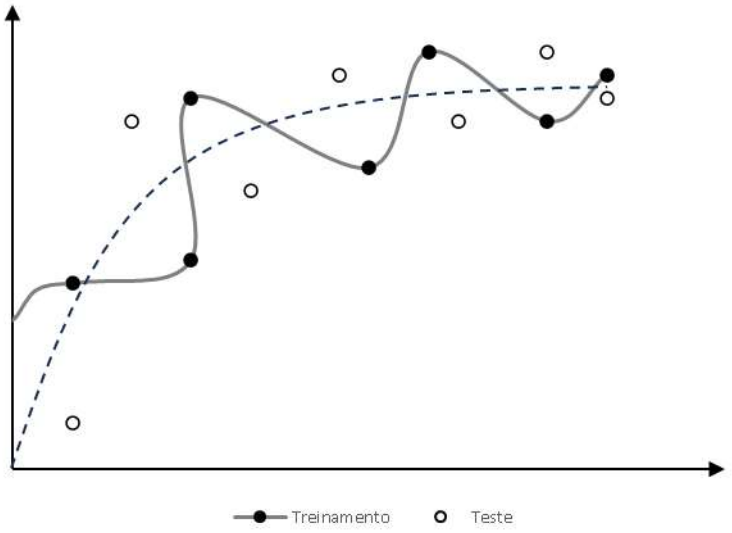

Figura 10 - Exemplo de um conjunto de dados para treinamento e teste com a curva ondulada do treinamento em tom mais claro e a função desejada traçada.

Apesar de o modelo ter diminuído o viés estatístico, se adaptando totalmente à relação entre as variáveis do conjunto de treinamento e, consequentemente, levando

o erro de treinamento a zero, ao introduzi-lo para um conjunto de teste com novos dados, o erro de teste - ou erro de generalização - foi considerável.

Na linguagem de aprendizado de máquina, a diferença entre os erros de treinamento e teste é chamado de variância. No exemplo da Figura 10, quando o modelo se adapta muito ao conjunto de dados de treinamento e falha em capturar generalizações, mantendo a variância alta, é considerado que ele possui um *overfitting* ou “sobreajuste”.

Em contraste com o *underfitting*, o *overfitting* pode ser causado por modelos com muitos parâmetros, tempos de treinamento longos e muitas épocas, ou que possuam características irrelevantes (ruídos) nos dados, que fazem o modelo aprender características não aplicáveis no mundo real.

A Figura 11 é um exemplo de como o comportamento dos erros de treinamento e teste deve seguir através das épocas, em um modelo com o conjunto de dados bem tratados e os hiperparâmetros bem definidos. O erro de teste começa caindo, conforme o modelo vai aprendendo padrões nos dados de treinamento. No entanto, após um determinado número de épocas, ele estabiliza e começa a subir, sinalizando que o modelo está se ajustando muito aos dados de treinamento e perdendo capacidade de generalização.

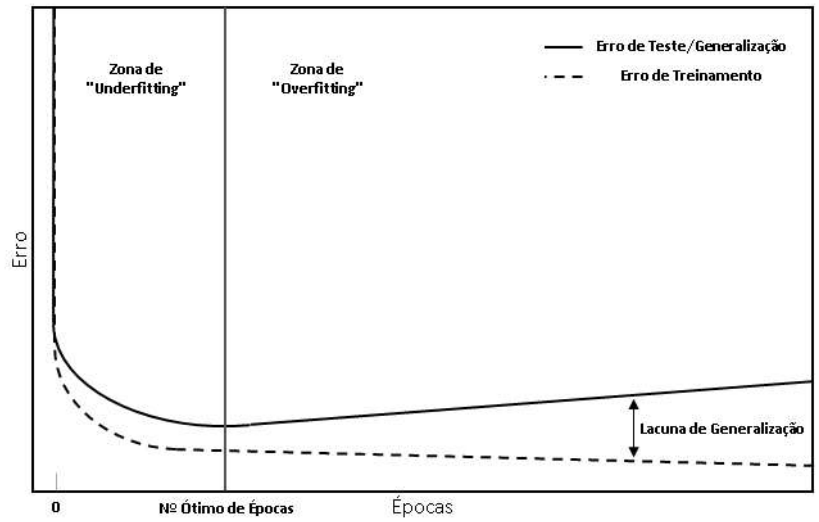

Figura 11 - Exemplo de relação entre o número de épocas e os erros de teste e treinamento de um modelo não regularizado.

Observa-se também que há um certo número ótimo de épocas, que minimiza a variância do modelo. Uma forma de tentar encontrá-la é através da "parada antecipada", que consiste em parar o treinamento ao verificar que o erro de tese estagnou, antes de começar a subir. No entanto, essa tarefa é controversa.

Por exemplo, em ambientes com alta variabilidade de dados ou ruído, o erro de validação pode variar. Isso torna difícil determinar um ponto claro para parar, pois o modelo pode parecer sobreajustado temporariamente devido ao ruído nos dados, mas na verdade, ainda possuí um viés estatístico alto, que será reduzido com mais épocas.

A Figura 11 pode induzir que encontrar o ponto ótimo de épocas é algo simples, mas na prática, a variância não é medida diretamente ao longo das épocas; o que é monitorado são os erros dos conjuntos em cada época. Dessa forma, muitas vezes um aumento na variância inicia em algum lote e só se torna evidente depois de começar a fazer efeito.

A parada antecipada é mais uma solução complementar, ao não abordar as causas principais do *overfitting*, como muitos parâmetros ou dados de treinamento insuficientes. Outros métodos de regularização, abordam diretamente estas questões, impondo restrições à complexidade do modelo ou introduzindo sanções para modelos complexos.

Antes de seguir com a próxima seção, vale concluir que a aprendizagem do modelo só se torna eficiente não ao reduzir apenas o erro de treinamento, mas também ao reduzir o erro de teste. Isso depende dos dados apresentados e na capacidade do modelo em se adequar e generalizar a relação desses dados.

4.4.6) Regularização

A regularização tem o intuito de evitar o *overfitting* do modelo, com métodos que penalizam a complexidade excessiva da rede neural. Isso é realizado através da adição de um termo de penalidade à função erro do modelo ou através de técnicas de treinamento. Para técnicas de penalização, temos:

- L1 - Regularização Lasso (abreviado do inglês *Least Absolute Shrinkage and Selection Operator*): Técnica que consiste em adicionar a soma dos valores absolutos dos pesos do modelo no cálculo da função erro, penalizando o modelo por pesos altos. O método estimula a redução dos

pesos, até desativando alguns neurônios quando sua ativação fica zerada, deixando o modelo esparso. Dessa forma, o método é utilizado em modelos com muitas variáveis, para auxiliar na seleção de apenas as mais importantes.

- L2 - Regularização Ridge: Técnica que consiste em adicionar a soma dos quadrados dos pesos do modelo no cálculo da função erro, penalizando ainda mais os pesos mais altos. O método estimula os pesos ficarem menores e mais distribuídos, sendo eficaz em casos de multicolinearidade, nos quais as variáveis são muito correlacionadas.
- *Elastic Net* (Rede Elástica): Método que combina L1 e L2, proporcionando uma solução intermediária entre os dois.

Os métodos de penalização utilizam um parâmetro lambda (λ, entre 0 e 1) para controlar a força da penalidade. Ao multiplicar o parâmetro na soma dos valores absolutos ou dos quadrados dos pesos (no caso da rede elástica, um parâmetro para cada regularização), busca-se equilibrar a regularização para evitar o *overfitting* e manter um bom nível de generalização.

Um lambda muito alto aumenta o viés estatístico do modelo, mas reduz a variância. Majoritariamente por experimentação, é possível determinar o lambda ótimo do modelo. A Figura 12 ilustra essa relação, ao demonstrar o erro de teste, viés estatístico e variância, através da capacidade (eficiência) do modelo.

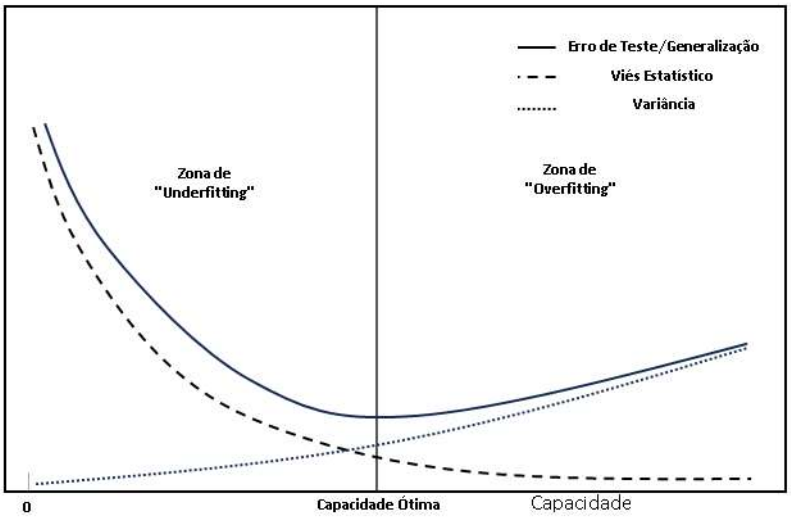

Figura 12 - Exemplo de relação entre erro de teste, viés estatístico e variância de um modelo regularizado.

A última técnica que será apresentada não é relacionada à penalização, mas sim ao treino, como a parada antecipada introduzida na Seção 4.4.5. A técnica em questão chama-se *dropout* (abandono), e similar à L1, busca tirar a dependência do modelo em relação a um neurônio específico. Em cada interação, os neurônios das camadas de entrada e intermediárias tem uma probabilidade p de serem temporariamente abandonados, ou seja, não contribuirão para a propagação direta e *backpropagation* durante a iteração vigente.

Ao utilizar os métodos de penalização de forma eficiente, o *dropout* pode não ser necessário, inclusive, pode conflitar com os demais métodos e aumentar o viés estatístico do modelo.

Esse foi o último conceito da Seção 4.4, que apresentou os principais fundamentos do aprendizado de uma RNA, desde um panorama dos cálculos e processos envolvidos no treinamento do modelo, até maneiras de avaliar seu desempenho. A próxima seção abordará conceitos mais avançados, mas necessários para compreender o modelo prático do próximo capítulo.

4.6) Variações do Modelo Apresentado

O capítulo introduziu, até o momento, os conceitos gerais sobre aprendizado de máquina, que já seriam suficientes para criar um modelo na prática. No entanto, as RNAs *feedfoward*, que o trabalho utilizou como modelo base para apresentar seus fundamentos, não costumam a desempenhar resultados satisfatórios em problemas de séries temporais, como o problema em questão de predição da volatilidade.

As redes *feedfoward* são mais eficientes em tarefas nas quais os dados podem ser analisados sem a necessidade de considerar relações temporais ou espaciais entre eles, como exemplo, alguns problemas de classificação. Além disso, sua estrutura menos elaborada, é útil para introduzir de maneira mais didática os conceitos gerais das RNAs.

Essa seção apresentará duas variações de redes neurais e, por último, um otimizador que colabora para o treinamento e desempenho da rede.

4.6.1) Redes Neurais Recorrentes (RNR)

As Redes Neurais Recorrentes (RNRs) representam um dos principais modelos de aprendizado de máquina, geralmente utilizado em problemas de dados de séries temporais. Estas redes distinguem-se pela sua arquitetura que lhes permite reter e utilizar informações de entradas de dados anteriores, tornando-as particularmente eficazes em cenários onde os dados são inerentemente sequenciais.

A arquitetura de uma RNR envolve neurônios com conexões autorreferenciais, ou seja, as ativações de um conjunto de dados podem ser repassadas para o conjunto da próxima série temporal, possibilitando uma forma de memória no modelo, essencial para tarefas onde o contexto histórico é relevante.

Em uma camada oculta de instante *t*, ao invés de o conjunto de neurônios repassar as ativações para uma camada de saída, eles repassam para cada neurônio da camada oculta do instante seguinte *t+1*, que junto com as próprias novas entradas do conjunto de dados *t+1*, formam ativações que guardam informações temporais, conforme representado na Figura 13.

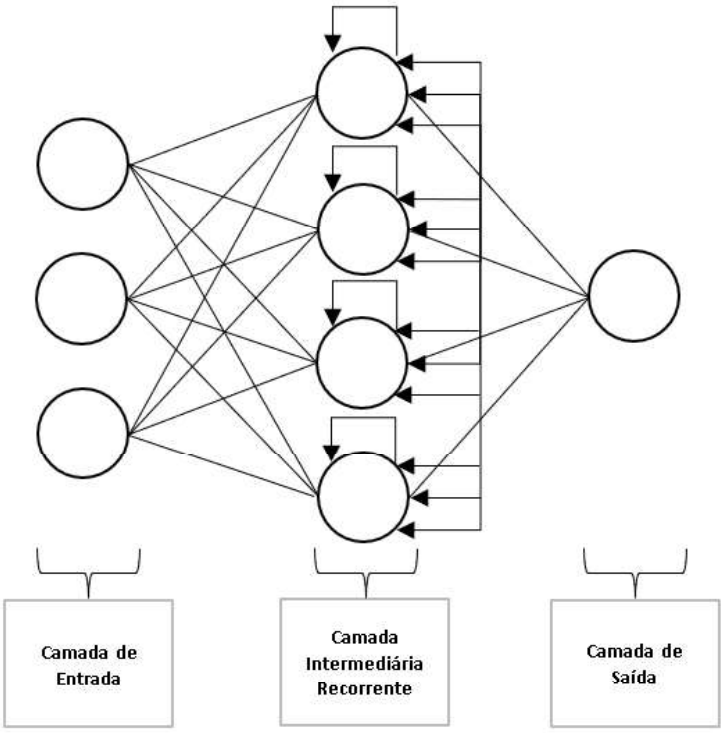

Figura 13 - Exemplo de um modelo de rede neural recorrente, com apenas uma camada intermediária recorrente.

A Figura 13 presume que os neurônios da camada intermediária, ao invés de apenas repassar as ativações para sua camada de saída, repassam também para os neurônios da camada intermediária da próxima série temporal. A Figura 14 complementa esse conceito.

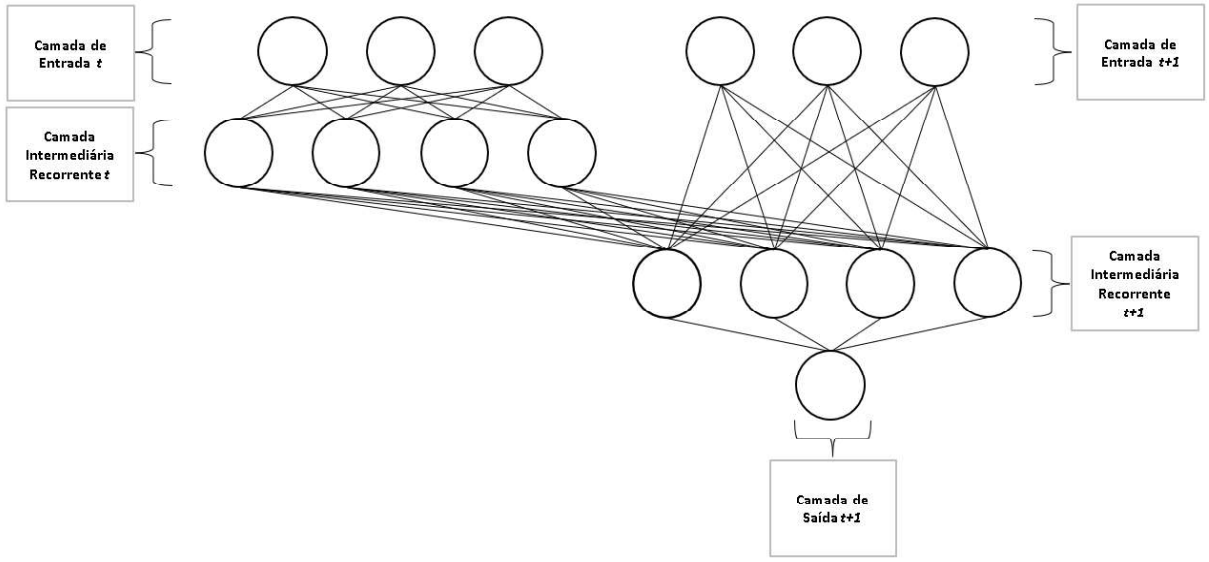

Figura 14 - Exemplo de como os neurônios intermediários repassam as ativações para a próxima série temporal.

Explicando a Figura 14, o modelo busca prever um valor para a variável dependente no instante *t+1*. A camada oculta em *t+1* utiliza as ativações da mesma camada no instante *t*, além das próprias ativações da camada de entrada *t+1*, para calcular suas ativações e repassá-las para a camada de saída *t+1*. Dessa forma, a ativação do neurônio na camada de saída *t+1* seria o resultado do modelo. No instante *t* também há uma camada de saída, no entanto, sua ativação não seria utilizada nesse exemplo.

Caso houvesse mais um instante, *t+2*, as ativações da camada intermediária *t+1* seriam utilizados junto com as ativações da camada de entrada *t+2*, para gerar as ativações da camada intermediária *t+2* e o resultado do modelo seria a ativação do neurônio na camada de saída *t+2* e assim em diante para todos os instantes de tempo. Note que as ativações da camada intermediária *t* já são computadas nas ativações *t+1*.

A equação geral da ativação em um neurônio da camada oculta de um modelo RNR é dada pela Equação 44.

$$a_t = f(W_{xa} * x_t + W_{aa} * a_{t-1} + b_a) \quad (44)$$

Onde:

- a_t: Ativação do neurônio da camada oculta no instante *t*.
- W_{xa}: Matriz de pesos para as conexões do neurônio com as ativações da camada imediatamente anterior de mesmo instante *t*.
- x_t: Vetor das entradas ou ativações das camadas anteriores do mesmo instante de tempo.
- W_{aa}: Matriz de pesos para as conexões recorrentes com as ativações dos neurônios da camada oculta de instante *t-1*.
- a_{t-1}: Vetor das ativações dos neurônios da camada oculta de instante *t-1*.
- b_a: Vetor dos vieses técnicos para aquela camada.
- *f*: Função de ativação.

Ambas as matrizes de pesos e o vetor de viés técnico tem os mesmos valores em todos os instantes de tempo. Isso permite que o modelo aplique os mesmos parâmetros de aprendizado (pesos e vieses técnicos) em diferentes partes da entrada sequencial e crie a "memória" que captura tendencias temporais. Note que os vetores x_t e a_{t-1} são os que possuem valores diferentes em cada instante de tempo.

O algoritmo de *backpropagation* precisa de adaptações específicas para lidar com a natureza sequencial das RNRs. Uma dessas adaptações é o *Backpropagation Through Time* (BPTT), que transforma a RNR em uma representação semelhante a uma rede *feedforward*, onde cada camada oculta corresponde a um passo de tempo específico, similar ao que a Figura 14 ilustra.

Neste processo, o gradiente da função de custo é calculado para cada instante de tempo, levando em consideração não apenas o impacto imediato de um peso na saída atual, mas também como esse peso afeta as saídas em passos de tempo subsequentes. Esta característica do BPTT é essencial para compreender um dos principais desafios das RNRs, especialmente em contextos em que a sequência de dados é extensa.

Devido à natureza da matriz de pesos W_{aa}, que é compartilhada entre diferentes instantes de tempo e mantém seus valores constantes, quando os valores dessa

matriz são consistentemente maiores do que 1, a multiplicação repetida dos pesos ao longo de muitos instantes de tempo pode levar a uma explosão de gradiente (conceito explicado na Seção 4.4.4). Isso ocorre porque os valores das ativações - e consequentemente os gradientes - tendem a aumentar exponencialmente a cada instante de tempo.

Por outro lado, se os valores dos pesos W_{aa} são consistentemente menores que 1, o efeito cumulativo ao longo dos instantes de tempo pode causar um desvanecimento de gradiente (conceito também explicado na Seção 4.4.4). Neste cenário, as ativações - e os gradientes - diminuem progressivamente, tornando-se insignificantes e dificultando o aprendizado de dependências de longo prazo, pois as informações dos passos de tempo anteriores são progressivamente perdidas.

Esses problemas não são causados apenas pela magnitude dos pesos, mas também são influenciados pelas funções de ativação utilizadas nas RNRs, por exemplo, as funções de ativação tanh e sigmoíde. Suas derivadas podem ser pequenas, exacerbando os efeitos de desvanecimento de gradiente.

Dessa forma, modelos de RNR podem ficar menos eficientes em determinadas tarefas, demandando variações no modelo.

4.6.2) *Long Short-Term Memory* (LSTM)

A *Long Short-Term Memory* (LSTM, traduzido livremente como memória de longo e curto prazo) é uma arquitetura de RNR, concebida para superar as limitações dos modelos tradicionais na aprendizagem de dependências temporais extensas. A capacidade das LSTMs em preservar informações por intervalos de tempo prolongados, independentemente da duração destes, as torna mais adequadas para a classificação, processamento e previsão de séries temporais que arquiteturas de RNR mais simples.

A estrutura de uma LSTM é composta por cinco componentes principais — três portões (*forget gate*, *input gate*, e *output gate*) e dois estados (estado da célula e estado oculto) — que juntos, formam os cinco estágios que regulam o fluxo de informação através da unidade, conforme a Figura 15. Essa divisão em estágios que permite o modelo discriminar quais informações devem ser armazenadas, modificadas, ou descartadas ao longo do tempo.

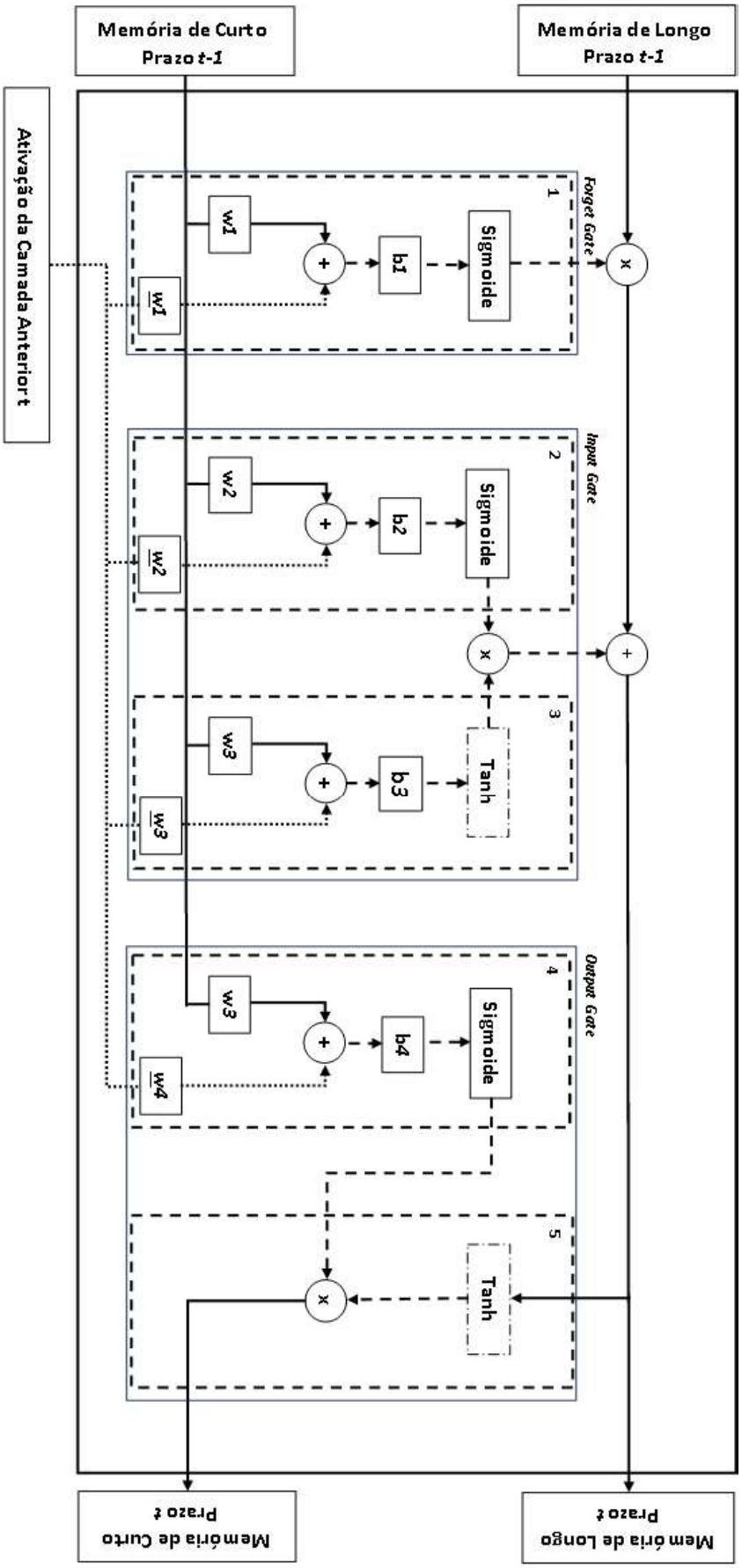

Figura 15 - Exemplo de neurônio com a unidade de LSTM.

A Figura 15 representa um único neurônio com uma unidade de LSTM. O modelo divide a memória em duas partes, de longo e curto prazo. Ao longo dos estágios, os dados de longo prazo são incorporados aos dados de curto prazo.

As linhas contínuas do canto superior, representando e ligando as memórias de longo prazo, são conhecidas como “estado da célula” (*cell state*). Não há pesos e vieses técnicos que modificam diretamente esse estado, apenas uma operação de multiplicação e outra de soma, evitando os problemas das RNRs de explosão ou desvanecimento do gradiente.

As linhas contínuas do canto inferior, representando e ligando as memórias de curto prazo, são conhecidas como “estado oculto” (*hidden state*). Elas atribuem diferentes pesos em cada estágio do modelo, tanto para as memórias de curto prazo de instante *t-1*, quanto para as ativações da camada anterior de instante *t*.

A camada anterior de mesmo instante pode ser tanto uma camada de entrada quanto outra camada intermediária, inclusive, poderia ser até outra camada intermediária com arquitetura LSTM. As ativações da camada anterior de instante *t* são tratadas como um vetor único, assim, a arquitetura atribui pesos específicos para cada estágio, mas dentro de um mesmo estágio, as ativações da camada anterior possuem os mesmos pesos entre elas.

No primeiro estágio, a LSTM atribui pesos específicos para a memória de curto prazo de instante *t-1* e para o vetor de ativação da camada anterior de instante *t.* Os valores ponderados são somados, junto com um viés técnico específico desse estágio. O resultado das somas passa por uma função sigmoide, que por sua vez, resulta em um número de 0 a 1. O resultado dessa função é multiplicado pela memória de longo prazo de instante *t-1*.

Por conta dessa dinâmica que o Estágio 1 é definido como o percentual da memória de longo prazo que o modelo deve esquecer. Por exemplo, se o resultado da função sigmoide desse estágio for zero, o modelo “esquecerá” toda a memória de longo prazo. O Estágio 1 é o único no primeiro “portão” do modelo, conhecido como *Forget Gate.*

O segundo e terceiro estágio formam o *Input Gate*. Esses estágios também atribuem seus pesos para a memória de curto prazo *t-1* e para o vetor de ativação da camada anterior de instante *t.* Os valores ponderados também são somados, junto com um viés técnico, específico para cada estágio.

O Estágio 3, por possuir uma função tanh, resulta em um número entre -1 e 1, que reflete o valor potencial da memória de curto prazo a ser adicionado à memória de longo prazo. O Estágio 2, por possuir novamente a função sigmoide, é responsável por definir o percentual do valor potencial que será de fato adicionado. Os resultados desses estágios são multiplicados e, assim, o modelo define o valor da *Input Gate*, que é somado à memória de longo prazo.

O quarto e quinto estágio formam o *Output Gate*. Este portão controla as informações a serem emitidas a partir do estado da célula atual. O Estágio 5 define, a partir do valor da memória de longo prazo (já incrementada com o valor da *Input Gate*), através de uma função tanh, o novo valor potencial da memória de curto prazo.

O Estágio 4 atribui seus pesos à memória de curto prazo *t-1* e ao vetor de ativação da camada anterior de instante *t*. Também soma os resultados a um viés técnico específico para esse estágio e, através uma função sigmoide, define o percentual do resultado do Estágio 5 que será utilizado como a nova memória de curto prazo. Em outras palavras, o *Output Gate* define o novo valor do estado oculto.

O resultado do estado oculto - ou de curto prazo - no último instante de tempo é a ativação que o modelo repassará para as próximas camadas de mesmo instante de tempo. O estado de célula, normalmente, é utilizado apenas como uma estrutura interna da unidade de LSTM ou entre camadas LSTM.

Uma prática comum em modelos de processamento de sequência temporal é implementar múltiplas camadas LSTM, onde a primeira camada é encarregada de extrair e processar dependências temporais e os padrões ao longo do tempo dos dados. As camadas LSTM posteriores se concentram na agregação e interpretação global dos dados, condensando toda a informação da sequência em um único vetor de características.

Esta abordagem facilita a captura de padrões em múltiplas escalas temporais, aprimorando a capacidade do modelo em realizar predições acuradas e classificações baseadas em sequências completas.

4.6.3) *Adaptive Moment Estimation* (Otimizador Adam)

O otimizador Adam, uma abreviação de "*Adaptive Moment Estimation*", é uma técnica de otimização utilizada no treinamento de RNAs. Desenvolvido por Diederik P. Kingma e Jimmy Ba e introduzido em seu artigo de 2014[27], o Adam é reconhecido por combinar propriedades de outros dois algoritmos, *Adaptive Gradient Algorithm* (AdaGrad) e *Root Mean Square Propagation* (RMSProp), para “otimizar” a taxa de aprendizado do modelo.

O AdaGrad é um otimizador que ajusta a taxa de aprendizagem de cada parâmetro de forma individualizada, aumentando a eficiência em cenários com dados esparsos, ou seja, em cenários que há muitos dados próximos ou iguais a zero. Seu diferencial está na capacidade de realizar passos menores para parâmetros atualizados com mais frequência e passos maiores para parâmetros atualizados com menos frequência, por meio do ajuste da taxa de aprendizagem com base no histórico acumulado a soma dos quadrados dos gradientes para cada parâmetro ao longo do tempo. Isso permite que o AdaGrad se adapte ao comportamento específico de cada parâmetro, otimizando o desempenho do modelo, especialmente em tarefas com gradiente esparso.

Por outro lado, por utilizar o histórico completo do gradiente, o AdaGrad pode sofrer com um problema de taxa de aprendizagem monotonamente decrescente, a ponto de ela ficar muito próxima de zero e interromper o aprendizado. O RMSProp surge como uma extensão do AdaGrad, proposto para contornar esse obstáculo, ajustando a taxa de aprendizagem de uma maneira que ela não diminua tanto.

O RMSProp ajusta a taxa de aprendizagem em cada parâmetro do modelo individualmente. O fator pelo qual a taxa de aprendizagem é ajustada para cada parâmetro é determinado pelo quadrado médio dos gradientes recentes desse parâmetro. Isso significa que o ajuste é feito não com base no histórico completo do

[27] KINGMA, Diederik P.; BA, Jimmy. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

gradiente, mas sim nos gradientes mais recentes, permitindo uma convergência mais rápida e eficiente em muitos casos.

O Adam integra características dessas duas abordagens, para buscar um desempenho mais robustos nos modelos. Ele calcula dois tipos de "*momentums*" para cada parâmetro: o primeiro *momentum* refere-se a média móvel dos gradientes, que ajuda a capturar a direção da tendência dos gradientes ao longo do tempo; o segundo *momentum* refere-se a média móvel dos quadrados dos gradientes, que ajuda a capturar a variabilidade dos gradientes.

As médias móveis são estimativas consideradas viesadas porque são inicializadas em zero e tendem a ser menores durante as primeiras iterações do treinamento. Isso pode levar a uma estimativa inicialmente tendenciosa tanto da direção quanto da variabilidade dos gradientes, que, por sua vez, afeta a aprendizagem.

Para compensar o viés inicial e assegurar que as estimativas dos momentos sejam precisas, o Adam ajusta as estimativas com base no número de iterações já realizadas. Os cálculos fogem do escopo do trabalho, mas em linhas gerais, o ajuste é feito através de fatores de correção que levam em conta quantas vezes o algoritmo atualizou os parâmetros (ou seja, o número de iterações). Os fatores de correção são calculados de tal forma que o viés das estimativas diminua à medida que o número de iterações aumenta.

O objetivo desse ajuste é fazer com que as estimativas corrigidas dos primeiros e segundos momentos se aproximem de seus valores "reais" ou esperados, ou seja, os valores que elas teriam se não houvesse viés inicial. Isso é importante pois permite que o Adam ajuste as taxas de aprendizagem de forma mais precisa e eficaz, levando em conta a magnitude e a variabilidade reais dos gradientes ao atualizar os pesos do modelo.

Ao aproximar as estimativas de seus valores reais, o Adam pode otimizar o processo de treinamento, facilitando a convergência para pontos críticos do gradiente.

Esse é o último conceito introduzido na seção e capítulo. Conclui-se, assim, as apresentações sobre os fundamentos básicos de uma RNA e o trabalho segue para a aplicação prática de um modelo.

5) Modelo de Predição de Volatilidade

Nesse capítulo, será desenvolvido um estudo de caso prático, englobando todos os conceitos previamente discutidos. O objeto de estudo selecionado para esta análise prática será os contratos futuros de soja, com vencimento em novembro de diferentes anos, negociados na Chicago Board of Trade (CBOT). Considerando-se a relevância e o volume de negociação desta commodity, torna-se mais simples a definição de algumas prováveis variáveis que influenciam seu preço e, consequentemente, sua volatilidade.

5.1) Estrutura do Modelo

5.1.1) Características do Ativo e Preparação dos Preços Históricos

A soja é uma planta originária da Ásia, utilizada principalmente para alimentação de animais, mas também na alimentação de humanos, produção de biodiesel, óleo, entre outros derivados. O processo de "esmagamento" ("*crushing*") da soja consiste em remover a casca e enrolá-la em flocos, que são embebidos em solvente e submetidos a um processo de destilação para produzir óleo de soja bruto puro. Após a extração do óleo, os flocos de soja são secos, torrados e moídos em farelo de soja.

É uma das principais commodities agrícolas em áreas plantadas anualmente. Apesar de sua origem, segundo o United States Department of Agriculture (USDA)[28], em 2023, a maior parte da produção do grão de soja é feita, em ordem descendente, no Brasil (40% da produção total do mundo), Estados Unidos (28%), Argentina (12%) e apenas em quarto um país asiático, a China (5%), seguido pela Índia (3%). Já em relação ao consumo, a China fica em primeiro, tanto em consumo doméstico quanto em importações anuais, seja em semente, óleo ou farelo de soja.

Apesar do Brasil ser o principal produtor, por conta de facilidades na coleta de dados, o trabalho analisará os contratos futuros referentes à safra americana. O plantio ocorre normalmente entre os meses de maio e junho, a colheita entre setembro

[28] Soybean 2023World Production. USDA. Disponível em: https://ipad.fas.usda.gov/cropexplorer/cropview/commodityView.aspx?cropid=2222000. Acesso em: 24/12/2023.

e outubro e a maior parte das entregas ocorrem em novembro, mês também referente ao contrato futuro com mais negociações anuais na bolsa.

Dessa forma, o contrato analisado será o negociado na bolsa de Chicago (CBOT), com vencimento em novembro. Conforme introduzido no capítulo 3, todo o contrato futuro de um mesmo ativo possui características padronizadas de negociação, no caso da soja, o contrato é negociado em centavos de dólar por *bushel* - uma medida equivalente à 27,2155 kgs – e cada contrato possui 5 mil bushels.

Cada contrato fica disponível, aproximadamente, um ano e meio antes da data de vencimento e após vencer, não há mais atualização de preços. Para o estudo, será feito uma série diária contínua de preços, de 2002 até 2023. O valor será sempre baseado no preço futuro de vencimento no novembro mais próximo. Após o vencimento do contrato, a série de dados utiliza os valores do contrato de novembro do ano seguinte.

Por exemplo, o primeiro dia de dados da base é 15/11/2002, o primeiro dia útil após o vencimento do contrato de novembro de 2022. Nesse dia, além das demais variáveis independentes, o modelo utilizará o preço de fechamento do contrato com vencimento em novembro de 2003 como referência. No dia 15/11/2003, primeiro dia útil após o vencimento do contrato de novembro de 2003, o modelo começa a utilizar os preços do contrato de vencimento em novembro de 2024 e assim em diante.

Assim, é possível ter uma série contínua de preços de contratos com vencimento em novembro de cada ano, o mês mais utilizado para negociações de preços referentes à safra americana de soja.

5.1.2) Variável Dependente

A variável dependente - ou a variável que o modelo tenta prever - é a volatilidade anualizada (em dias úteis, base 252) do contrato vigente, do dia de referência até o vencimento.

Como exemplo, no primeiro dia de dados, 15/11/2002, o valor da variável dependente será a volatilidade realizada, a partir desse dia até o vencimento do contrato vigente, em 14/11/2003. O valor da volatilidade ao período é transformado em um valor ao ano para padronizar a variável.

Para o último dia de negociação, como não é possível calcular a volatilidade para um único valor, é repetido o valor do dia útil imediatamente anterior.

5.1.3) Definindo as Variáveis Independentes

O modelo constará quatro grupos de variáveis independentes. O primeiro referente às datas de observação e características de prazo dos contratos; o segundo referente aos dados históricos de preços; o terceiro referente à oferta de soja e o último à demanda de soja. Cada grupo constará com diferentes quantidades de variáveis independentes.

As variáveis serão escolhidas buscando independência de relação entre elas, para evitar os erros de colinearidade ou multicolinearidade, que se referem a uma relação linear exata ou muito próxima entre duas ou mais variáveis, respectivamente.

Por exemplo, um fator que impacta consideravelmente a oferta e, por tanto, o preço da soja é a condição das plantas, que está diretamente relacionada às condições climáticas. Utilizar uma variável para a condição da safra e outra para a condição climática pode apresentar uma relação de colinearidade, dificultando a determinação do efeito individual de cada variável independente sobre a variável dependente, porque ambas as variáveis independentes se movem juntas.

Mesmo a colinearidade e multicolinearidade serem problemas mais graves em modelos de regressão linear do que em redes neurais, por conta de características de cada modelo, tratar esses aspectos tornam mais simples a interpretação e ajustes da rede neural, além de melhorar a eficiência computacional e diminuir riscos de *overfitting*.

5.1.4) Variáveis Independentes Temporais

O primeiro grupo tem o intuito organizar os dados, permitir com que o modelo capture tendencias de curto prazo, longo prazo e sazonais, além de identificar a mudança do contrato de referência. Como foi feito uma série contínua de contratos futuros de soja com vencimento em novembro, é fundamental colocar características de prazos de cada contrato para o modelo identificar mudanças de contrato.

As datas das observações foram divididas em dias, meses e anos para facilitar a captura de padrões entre anos e meses, mantendo os dados em sequência. Todas

as demais variáveis independentes, incluindo de outros grupos, tem o valor referente à data exata, ou seja, é o dado mais recente disponível para aquele determinado dia, mês e ano.

Variáveis:

- Dia da Observação.
- Mês da Observação.
- Ano da Observação.
- Dias Passados Desde o Início do Contrato Vigente: Nesse caso, não é referente à emissão ou primeira negociação do contrato no mercado, mas sim ao primeiro dia que o modelo começou utilizar aquele determinado contrato como referência para variações de preço na soja. Por exemplo, como estamos utilizando sempre o contrato de novembro como referência, o dia seguinte ao vencimento de um contrato reinicia esse número para zero.
- Dias até o Vencimento do Contrato Vigente.

5.1.5) Variáveis Independentes de Históricos de Negociação

Esse grupo refere-se aos preços históricos e volumes de negociação do contrato vigente, além das demais medidas possíveis de deduzir observando apenas o preço histórico. Como o modelo não é capaz de realizar cálculos estatísticos, é importante explicitar algumas medidas para compreender possíveis impactos na predição de volatilidade.

Variáveis:

- Preço de Fechamento: Último preço negociado do contrato vigente no dia referente.
- Curtose da Distribuição de Preços Desde o Início do Contrato Vigente: Caso sejam os primeiros três dias de referência de preço do contrato vigente, serão repetidos o último valor da variável do contrato anterior.
- Assimetria da distribuição de Preços Desde o Início do Contrato Atual: Caso sejam os primeiros dois dias de referência de preço do contrato vigente, serão repetidos o último valor da variável do contrato anterior.

- Diferença Entre o Preço de Fechamento e o Preço Médio do Contrato Atual: Caso seja o primeiro dia de negociação do contrato vigente, será repetido o último valor da variável do contrato anterior.
- Volume Negociado do Contrato Vigente no Dia de Referência.
- Volatilidade Desde o Início da Observação do Contrato Vigente: Caso seja o primeiro dia de negociação, será repetido o último valor da variável do contrato anterior.

5.1.6) Variáveis Independentes de Oferta

Em relação às variáveis exógenas ao preço, o primeiro grupo refere-se aos fatores que impactam a oferta de soja. Ele conterá a maior parte das variáveis exógenas, pois uma ideia inicial é que a demanda por soja tende a sempre crescer no longo prazo, devido ao aumento populacional. Dessa forma, a oferta acaba englobando a maior parte das variáveis reais que afetam o preço do ativo.

Variáveis:

- Oceanic Niño Index: O clima é um dos principais fatores que impactam a produção da soja. Se o nível de chuvas e temperatura forem muito inconstantes, a expectativa de produção varia junto.

  O El Niño é um fenômeno relacionado ao aquecimento anormal das águas do Oceano Pacífico em sua porção equatorial, responsável por um clima mais quente e seco no norte dos EUA – parte que possui a maior parte das plantações de soja – sendo menos favorável para a colheita. Já La Niña é o oposto do El Niño, caracterizando-se pelo resfriamento das águas superficiais do Oceano Pacífico equatorial, trazendo climas mais frios e húmidos.

  O índice mede a variação da temperatura no oceano Pacífico acima da média histórica, em graus celsius. Um valor acima de 0,5 sinaliza El Niño e um valor negativo abaixo de -0,5 sinaliza La Niña.

  Os valores do índice foram adaptados em uma média móvel de 90 dias, com o princípio de que os fenômenos demoram um tempo para impactar o clima nos EUA.

- Taxa de Juros Livre De Risco Dos EUA (Fed Funds Rate – Upper Limit): Refere-se ao limite superior da taxa de fundos federais dos EUA, que é a taxa de juros livre de risco do país. O Federal Reserve (Fed), banco central dos Estados Unidos, define essa taxa como parte de sua política monetária.

  Uma taxa mais alta pode refletir em empréstimos mais caros, que por sua vez, podem impactar nas condições de crédito e financiamento das safras de soja e sua negociação. É possível argumentar que essa variável pode impactar tanto a oferta quanto a demanda do ativo.
- Estoques Iniciais de Soja na Safra dos EUA: Refere-se ao volume, em toneladas, de soja já em estoques no início do período da safra nos EUA, em maio de cada ano. Por exemplo, se o número for maior que a média de anos anteriores, a oferta inicial está relativamente mais alta e os preços podem tender a variar mais conforme formem as expectativas da safra atual.
- Estoques Iniciais de Soja na Safra da China: Ideia similar aos estoques iniciais dos EUA, com exceção que na China o início da safra é considerado na metade de abril. Apesar da produção chinesa ser menos expressiva, por eles possuírem a maior demanda, uma maior oferta no próprio país também pode impactar na demanda por soja externa.
- Estoques Finais de Soja na Safra do Brasil: A safra brasileira termina entre maio e junho, próximo do início da safra nos EUA. Como o Brasil é o maior produtor, em toneladas, de soja, a oferta do país pode impactar os preços globais mesmo após sua safra, principalmente se os estoques estiverem altos.

5.1.7) Variáveis Independentes de Demanda

- *Global CIX Soybean Crush Margin*: Refere-se a média da margem de esmagamento global da soja. Quanto menor a margem, assume-se que haverá uma demanda menor por soja. Mais detalhes sobre o esmagamento podem ser encontrados na Seção 5.1.1.
- Balança Comercial da China com os EUA: Como boa parte do volume de soja produzido nos EUA tem o destino a China, utiliza-se essa variável para medir um possível potencial de como a demanda por soja pode comportar.

5.1.8) Divisão de Dados e Hiperparâmetros do Modelo

Essa seção visa descrever as características que estarão presentes em todos os modelos testados.

A estrutura de RNA escolhida foi LSTM com:

- Uma camada de entrada com o número de neurônios iguais ao número de *features*;
- Duas camadas LSTM intermediárias, sendo a primeira processa os dados em sequencias e a segunda condensa esses dados (conforme detalhado na Seção 4.5.2). Os números de neurônios dessas camadas variam ao longo dos testes;
- Uma camada intermediária densa totalmente conectada, utilizada para reduzir a dimensionalidade das características aprendidas, combinar características de maneiras não lineares e preparar os dados para a camada de saída. O número de neurônios e a função de ativação dessa camada variam ao longo dos testes;
- Uma camada de saída, com um único neurônio e uma função de ativação linear (conforme apresentado na seção 4.2.3.

A função erro para treinamento será o erro médio quadrático (MSE), com o intuito de penalizar erros maiores. No entanto, apenas com o intuito de avaliar e interpretar os resultados, serão utilizadas as funções RMSE e MAE nos valores preditos na última iteração do modelo.

Ao invés de utiliza um valor fixo para a taxa de aprendizagem, o modelo constará com o otimizador Adam para ajustar ao longo do treinamento.

A base de dados foi dividida da seguinte forma; os 80% primeiros dados são utilizados para treinamento do modelo e; os 20% dos dados restantes são utilizados para teste do modelo. Dessa forma, do dia 15/11/2002 (primeiro dia da base de dados) até 30/08/2019, as variáveis são utilizadas para treinamento e a partir do dia 03/09/2019 (primeiro dia útil após a data limite de treinamento) até 14/11/2023 (último dia da base de dados), as variáveis são utilizadas para teste.

As variáveis independentes que não possuem um valor entre 0 e 1 passaram por um processo de normalização, através da Equação 45. Isso facilita o treinamento e convergência do modelo. Como a variável dependente é um percentual, com os valores entre 0 e 1, não é necessário normalizá-la.

$$V_{Norm} = \frac{V - V_{Min}}{V_{Max} - V_{Min}} \quad (45)$$

Onde:

- V_{Norm} : Valor normalizado do valor *V.*
- V : Valor a ser normalizado.
- V_{Min} : Valor mínimo dentro do conjunto de dados ao qual *V* pertence.
- V_{Max} : Valor máximo dentro do conjunto de dados ao qual *V* pertence.

O modelo foi programado em ambiente Python. Os códigos do modelo podem ser encontrados nos apêndices do trabalho.

5.2) Aplicações e Ajustes do Modelo

O primeiro teste será feito apenas com as variáveis temporais e o preço histórico. O intuito é testar se o modelo produz resultados satisfatórios em uma estrutura simples e com menos *features*.

O teste inicial foi feito com ambas as camadas LSTM com 25 neurônios (unidades) cada e 25 neurônios na camada intermediária densa. Não houve qualquer tipo de regularização de dados, o treinamento foi feito com 200 épocas – valor normalmente considerado alto, mas definido assim por conta da complexidade baixa do modelo - e tamanho do lote (*batch size*) de 32.

Os resultados foram os demonstrados nas figuras 16 e 17 a seguir.

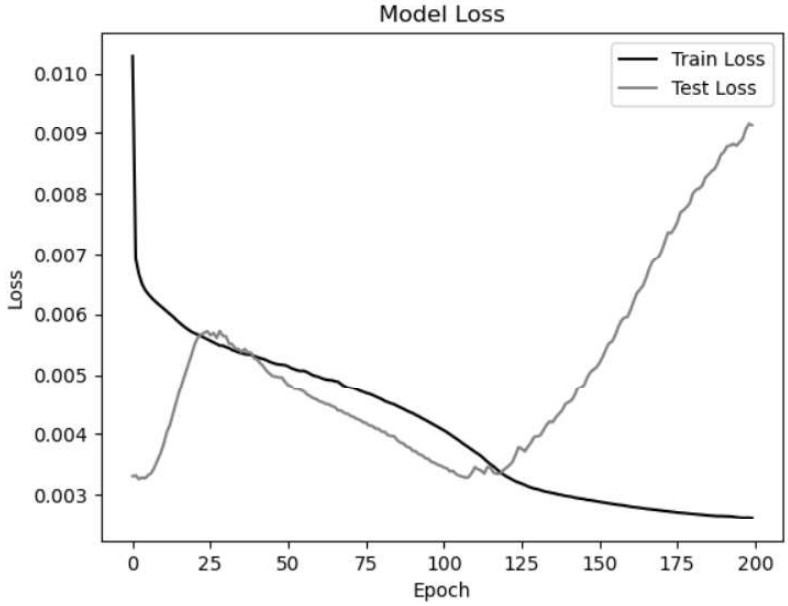

Figura 16 - Teste 1, Base de Dados 1, gráfico da função erro (MSE).

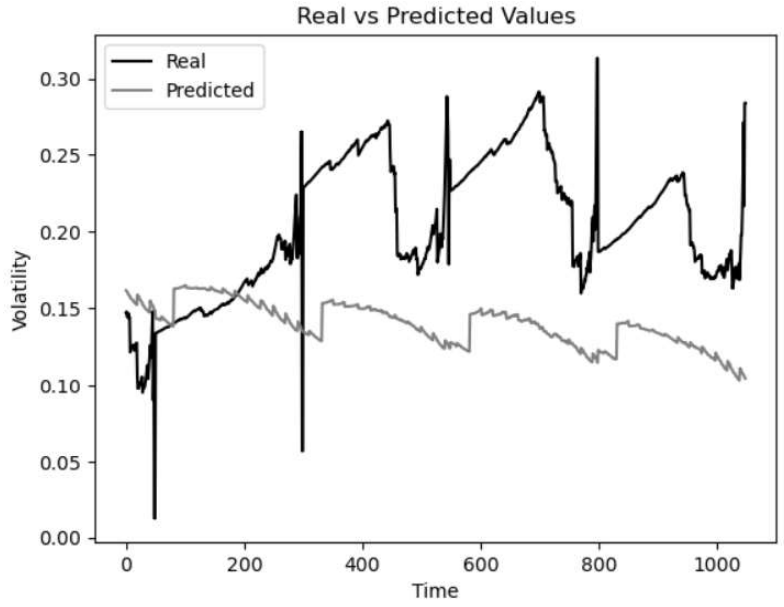

Figura 17 - Teste 1, Base de Dados 1, gráfico de valores reais e valores previstos no teste na última época.

Na Figura 16 e nas demais figuras de gráfico de função erro (*loss*, perda em inglês), o eixo Y refere-se aos valores em percentual do MSE e o eixo X às épocas do modelo. Na Figura 17 e nas demais figuras de gráficos de valores, o eixo Y refere-se à volatilidade anualizada, em percentual, e o eixo X aos dias passados do conjunto de dados separados para teste. Por exemplo, no primeiro dia do conjunto separado para teste, o valor do eixo X é zero. Como o conjunto de dados tenta criar uma série

continua, ou seja, que não possui um vencimento, os valores em dias podem ser generalizados dessa forma.

Ainda sobre a Figura 17, é possível observar momentos de variação abrupta dos valores dos resultados reais. Isso ocorre por conta da preparação feita para criar a série continua, onde a volatilidade realizada dos contratos próxima ao vencimento pode variar muito e, quando a base altera o contrato vigente para o próximo vencimento, ela retorna ao padrão de variação, criando esses choques.

Algumas alternativas para contornar o problema seriam repetir os valores da variável dependente nos últimos dias de negociação do contrato vigente ou mudar para o próximo contrato de referência alguns dias antes do vencimento do contrato vigente. No entanto, a base de dados será mantida dessa forma para avaliar com a rede neural lida com esse problema. Algumas variáveis independentes temporais e de históricos de negociação podem auxiliar o modelo a interpretar esses choques.

Voltando ao teste, o primeiro ponto que chama atenção está na Figura 16, onde o erro de teste do modelo começa baixo, mas aumenta nas primeiras épocas. Isso pode ser causado tanto pelo otimizador *Adam*, que ajusta parâmetros no início, quanto algum ruído na base de dados. Como o erro reduz após algumas épocas, não aparenta ser um problema tão grave ao modelo.

Por outro lado, próximo da época 25, o erro de teste começa a reduzir, indicando que o modelo está conseguindo aprender, mas volta a subir após a época 110, sinalizando *overfitting*.

A Figura 16 demonstra a diferença entre os valores previstos no teste, na última iteração, e os valores reais. A média dessas diferenças (MAE) foi de 7,0494%, ou seja, o resultado previsto pelo modelo foi, em média, 7,0494% maior ou menor que o valor real.

Reforçando que essa medida de erro é para complementar a análise do modelo; não é a medida utilizada para treinamento da RNA, onde o modelo utiliza como referência para tentar reduzir os erros de treinamento (no caso, o MSE é a função erro). O MAE é importante para capturar a dispersão média dos erros, pois todas as diferenças de valores têm o mesmo peso, diferente do MSE, que penaliza mais os erros maiores.

Para comparação, no Teste 2, foi reduzido o número de épocas de 200 para 110. Os resultados são observados nas Figuras 18 e 19

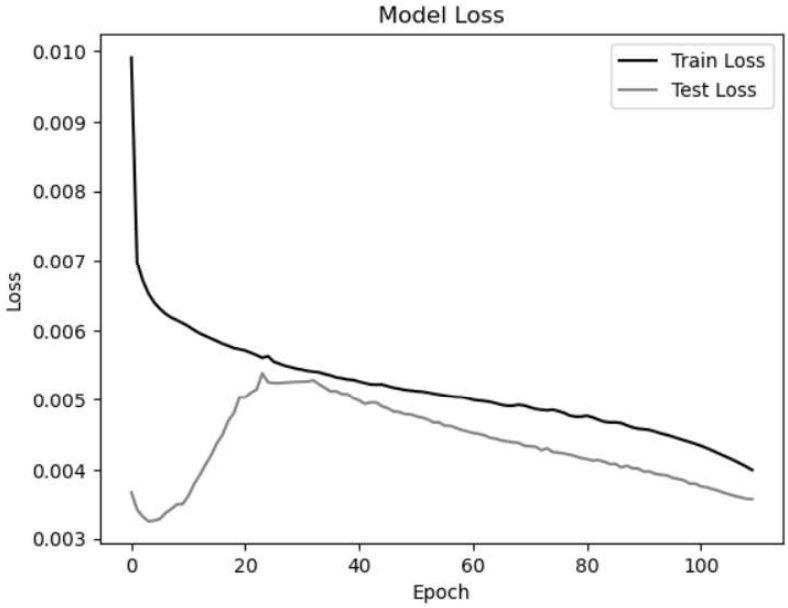

Figura 18 - Teste 2, Base de Dados 1, gráfico da função erro (MSE).

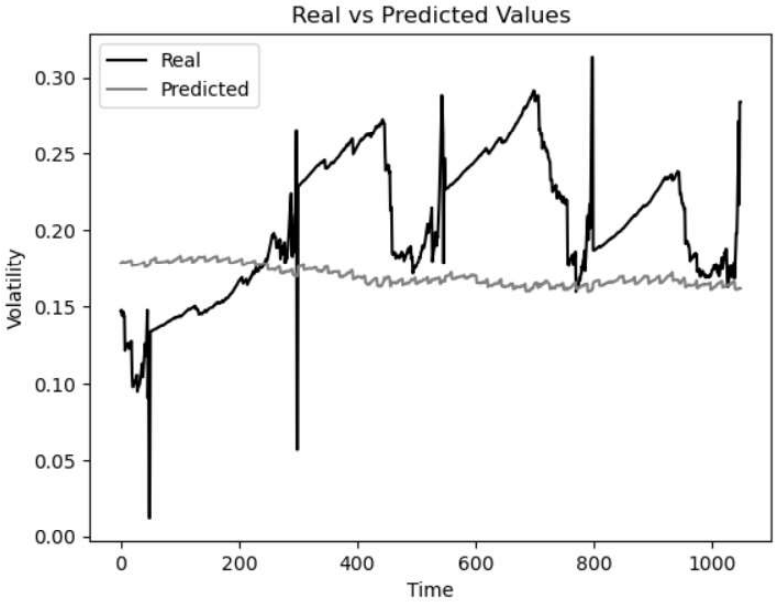

Figura 19 - Teste 2, base de dados 1, gráfico de valores reais e valores previstos no teste na última época.

A diferença média entre os valores previstos de teste e valores reais (MAE) foi de 4,3714%, ou seja, reduziu consideravelmente e a Figura 18 apresenta o erro de teste caindo sem sinais de *overfitting*. No entanto, nota-se na Figura 19 que o modelo perdeu o viés estatístico, ao não conseguir "curvar" a linha de projeção para os valores reais, comparado com a Figura 17.

Esse exemplo demonstra a importância em estruturar melhor o modelo em relação à regularização e complexidade do que apenas reduzir o número de épocas. Por outro lado, aumentar muito a complexidade para um modelo com poucas variáveis – como no exemplo atual – ou com um número baixo de observações, também não traz melhoras significativas.

Para demonstrar essa observação, mantendo as demais características e voltando para 200 épocas, caso seja feita uma regularização L2 em apenas 0,0001 na primeira camada LSTM (que processa os dados em sequência), o modelo aumenta ainda mais o viés estatístico e o *underfitting*.

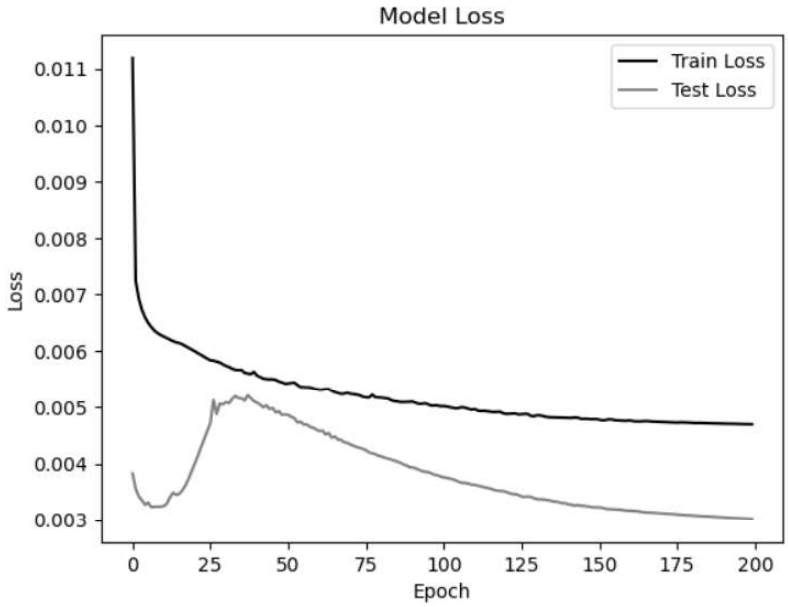

Figura 20 - Teste 3, Base de Dados 1, gráfico da função erro (MSE).

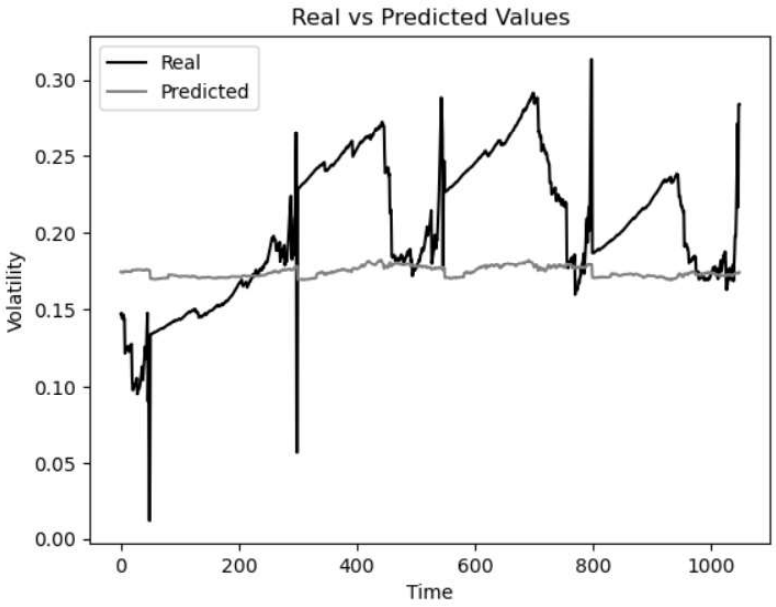

Figura 21 - Teste 3, Base de Dados 1, gráfico de valores reais e valores previstos no teste na última época.

Por conta de poucas quantidades de variáveis independentes diferentes, ao aplicar métricas de regularização, o modelo fica muito simples e não captura a complexidade que afeta a volatilidade do ativo. Ainda com a mesma base de dados, seria ainda possível testar um aumento no número de neurônios nas camadas ou diminuir o tamanho do lote e, assim, aplicar uma regularização. Mesmo com esses ajustes, os resultados não produzem melhorias significativas em relação aos parâmetros iniciais.

Incrementando a complexidade do modelo, foram incluídos na Base de Dados 2 as demais variáveis de preço (diferença para a média, curtose, assimetria, volatilidade histórica desde o início do contrato e volume negociado). No primeiro teste com essa nova base de dados, manteve-se 25 neurônios por camada, 200 épocas e tamanho do lote de 32.

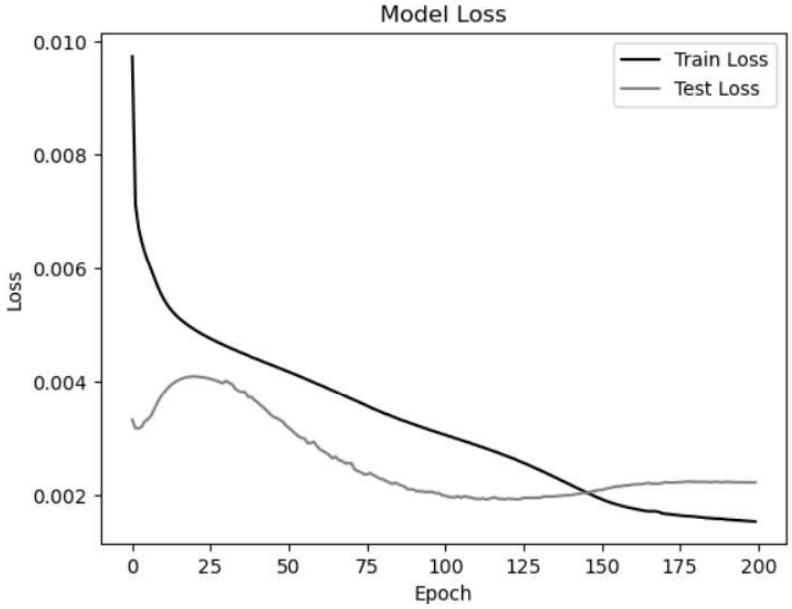

Figura 22 - Teste 4, Base de Dados 2, gráfico da função erro (MSE).

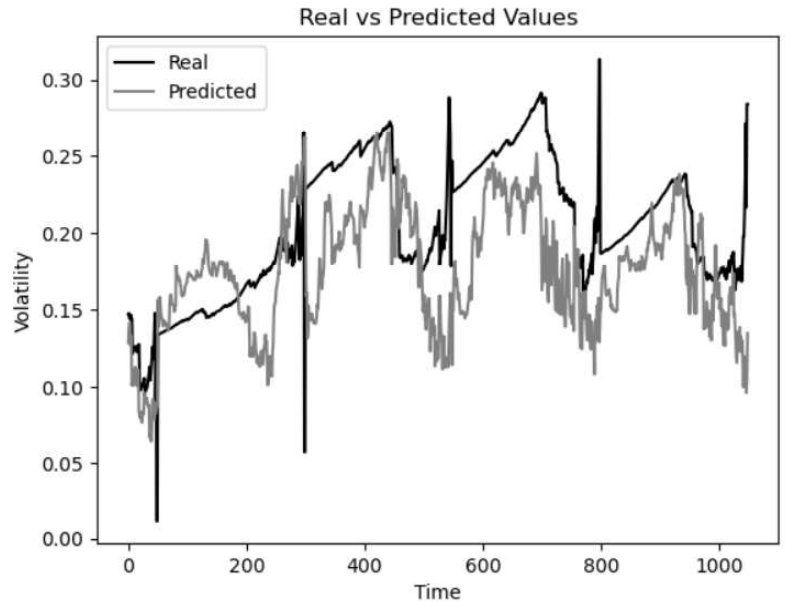

Figura 23 - Teste 4, Base de Dados 2, gráfico de valores reais e valores previstos no teste na última época.

Visualmente, já é possível notar uma melhoria na previsão em relação ao Teste 1, das Figuras 16 e 17. O MAE na Figura 23 foi de 5,3924%, também representando uma melhoria em relação ao Teste 1, mas ainda mais alto que nos testes 2 e 3.

Analisando a Figura 22, percebe-se também que, a partir da época 110, o erro de teste começa aumentar, indicando *overfitting*. Como foi uma característica similar ao Teste 1, para o próximo teste, o número de épocas será reduzido para 110. Os resultados são os demonstrados nas Figuras 24 e 25 abaixo.

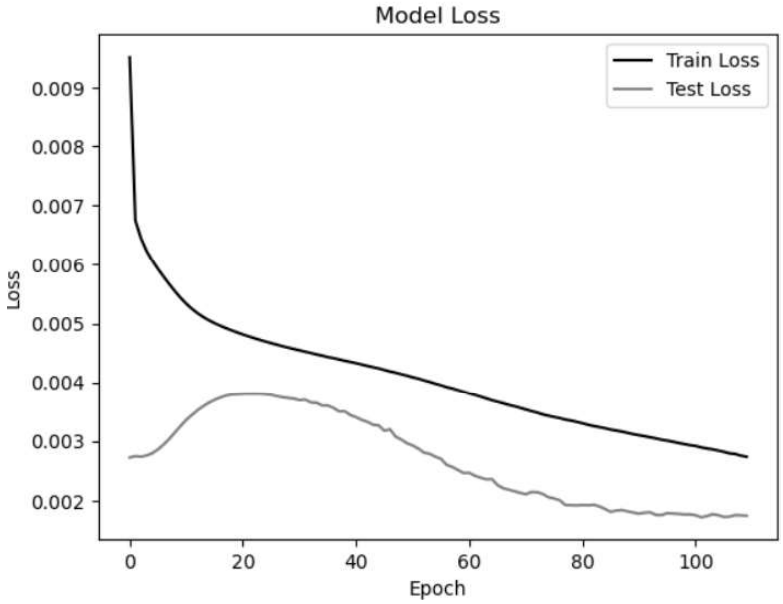

Figura 24 - Teste 5, Base de Dados 2, gráfico da função erro (MSE).

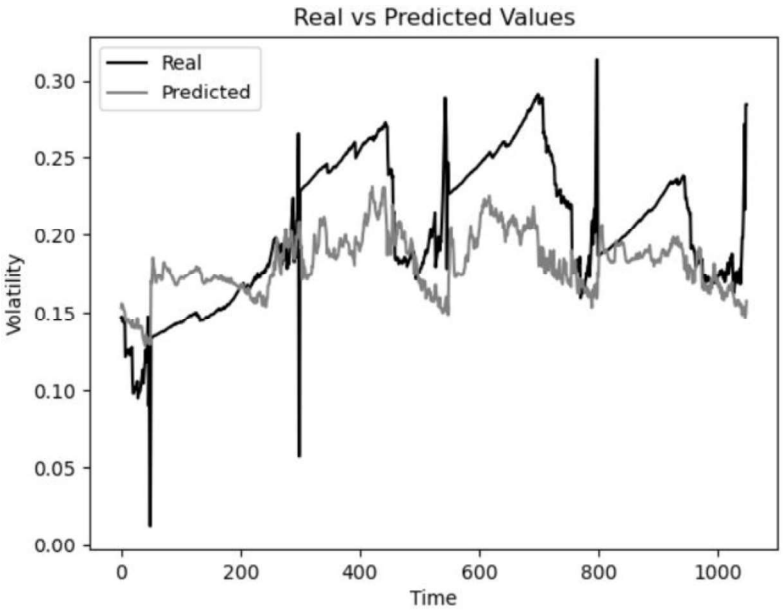

Figura 25 - Teste 5, Base de Dados 2, gráfico de valores reais e valores previstos no teste na última época.

O *overfitting* foi reduzido e o MAE caiu para 4,3549%. Diferentemente do Teste 2, representados nas Figuras 18 e 19, o modelo conseguiu manter o viés estatístico mais baixo, é possível observar essa característica com os valores previstos, que se "curvaram" melhor aos valores reais.

Analisando os números os erros de treinamento e teste, da última época, os valores respectivos em erros quadráticos médios (MSE), foram 0,2773% e 0,1766%. Para facilitar a análise e manter o erro na mesma unidade da volatilidade anualizada,

extrai-se a raiz quadrada desses valores – assim surge a medida raiz quadrada do erro médio (RMSE) - e os números ficam em 5,2658% e 4,2023%.

Relembrando, o MSE aparenta ser mais eficiente na aprendizagem do modelo por penalizar mais os erros de maior magnitude. O RMSE deixa-os na mesma base de comparação da volatilidade anualizada, mas é utilizado apenas na análise dos erros. Já o MAE é mais eficiente como uma medida direta da magnitude do erro. Para avaliar o modelo é necessária uma interpretação de todas as medidas e os próximos testes devem buscar reduzir todas as medidas de análise do erro.

Aumentar as épocas junto com a adição da regularização L1 e/ou L2 não demonstraram resultados melhores, pelo contrário, diminuíram a capacidade do modelo. Alterações no lote também não foram eficientes, tão pouco alterar funções ativação das camadas.

A alteração mais eficiente foi aumentar o número de neurônios apenas da primeira camada LSTM, de 25 para 50. Com o aumento de complexidade, as 110 épocas não sinalizaram *overfitting* e demonstraram o modelo poderia reduzir mais os erros com mais épocas. Dessa forma, aumenta-se o número de épocas para 145 para o Teste 6 e os resultados são demonstrados nas Figuras 26 e 27.

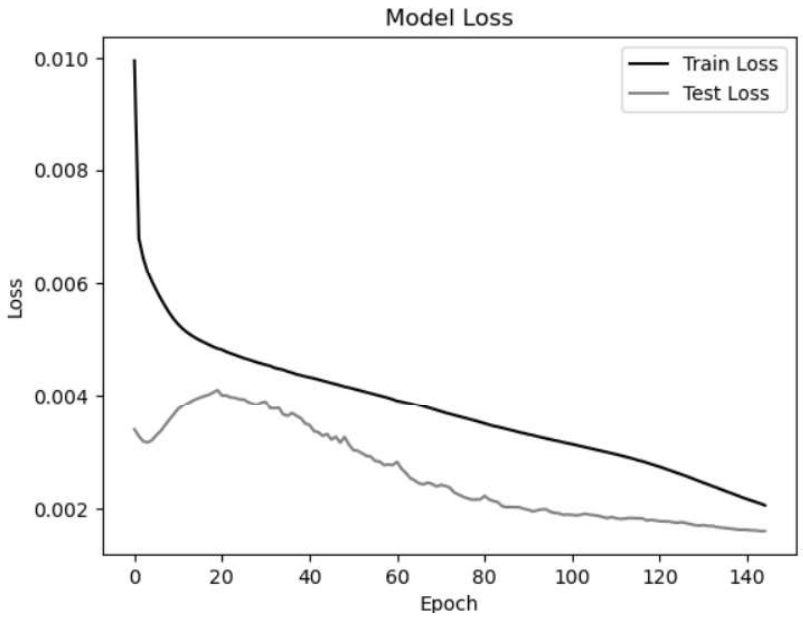

Figura 26 - Teste 6, Base de Dados 2, gráfico da função erro (MSE).

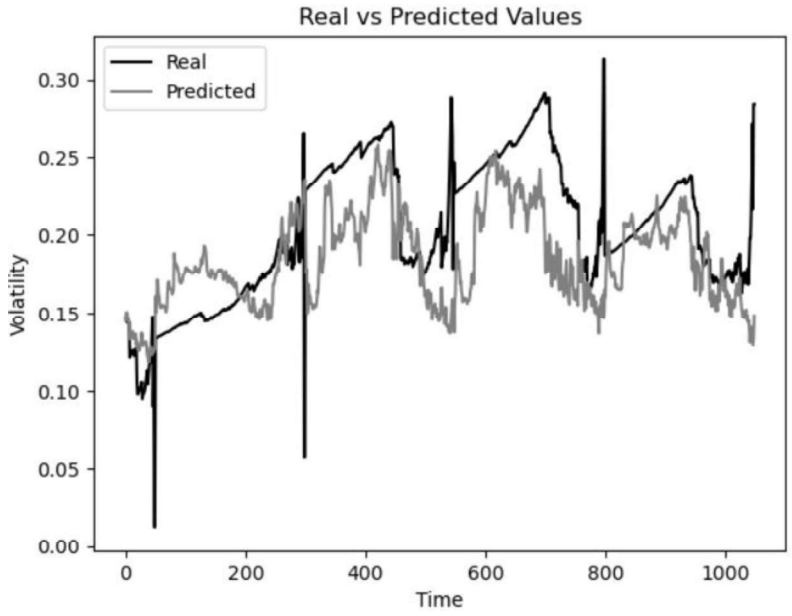

Figura 27 - Teste 6, Base de Dados 2, gráfico de valores reais e valores previstos no teste na última época.

A RMSE de treinamento e teste para a última época foram, respectivamente, 4,5324% (de 5,2658%) e 4,0056% (de 4,2023%), uma redução considerável tanto nos valores dos erros quanto na variância (antes em 1,0635%, agora em 0,5268%). Por outro lado, o MAE aumentou para 4,7190% (de 4,3549%).

A medida de erro MSE – consequentemente, o RMSE também – e variância diminuírem, mas o MAE de teste aumentar, pode indicar uma distribuição diferente dos erros no modelo, no qual ele está errando menos valores altos, mas está desviado mais em valores menores. Isso ocorre, provavelmente, por conta dos ajustes feitos para deixar os contratos como uma série continua e das próprias variações sazonais, que criam picos de variação muito elevados quando há uma mudança de contrato.

Preferir um RMSE ou MAE de teste menor depende da aplicação, objetivos de quem utiliza um modelo e momento de mercado. Por exemplo, alguém que avalia a volatilidade para *hedge* e tem receio de riscos de cauda, pode preferir os parâmetros do Teste 6 (RMSE menor). Por outro lado, alguém que busca especular em operações de curto prazo com volume exposto controlado, no qual o risco de cauda pode não ser tão relevante, o parâmetro anterior (MAE menor) pode ser mais interessante.

Nem sempre será possível reduzir todos os parâmetros de erro do modelo. Como nos testes feitos o RMSE e variância reduziram mais que o aumento do MAE, além do modelo mais complexo ter mais chances de funcionar melhor com o

incremento de mais *features*, o trabalho seguirá com os parâmetros do Teste 6, conforme vistos nas Figuras 26 e 27.

Os próximos testes serão feitos com a adição de variáveis que impactam tanto a oferta de soja– Índice Niño, taxa de juros básica dos EUA, estoques iniciais da safra dos EUA e China. e estoques finais da safra do Brasil –, quanto a demanda de soja – margens de esmagamento, balança comercial entre China e EUA.

A base de dados será avaliada em etapas diferentes. Com exceção do volume de negociação, como o preço histórico dos contratos e as demais variáveis relacionadas aos preços de negociação podem ser influenciadas pelas variáveis de oferta e demanda, utilizar todas as variáveis de preço, oferta e demanda pode não ser a melhor alternativa inicial para testar o modelo.

O principal intuito da divisão em etapas não é contornar uma possível colinearidade com o preço histórico - até porque enquanto as variáveis de preços mudam diariamente, algumas das variáveis de oferta e demanda possuem alterações periódicas -, mas sim para estudar quais *features* são mais eficientes no contexto do problema.

O Teste 7 inclui apenas o volume de negociação e as variáveis de oferta e demanda, excluindo no momento o preço histórico e as demais variáveis relacionadas ao preço de negociação. Os hiperparâmetros do modelo permaneceram iguais ao Teste 6; 50 neurônios na camada LSTM sequencial, 25 na camada LSTM não sequencial e 25 neurônios na camada densa intermediárias; lotes de tamanho 32; 145 épocas e; sem regularização. O resultado está na Figura 28 abaixo.

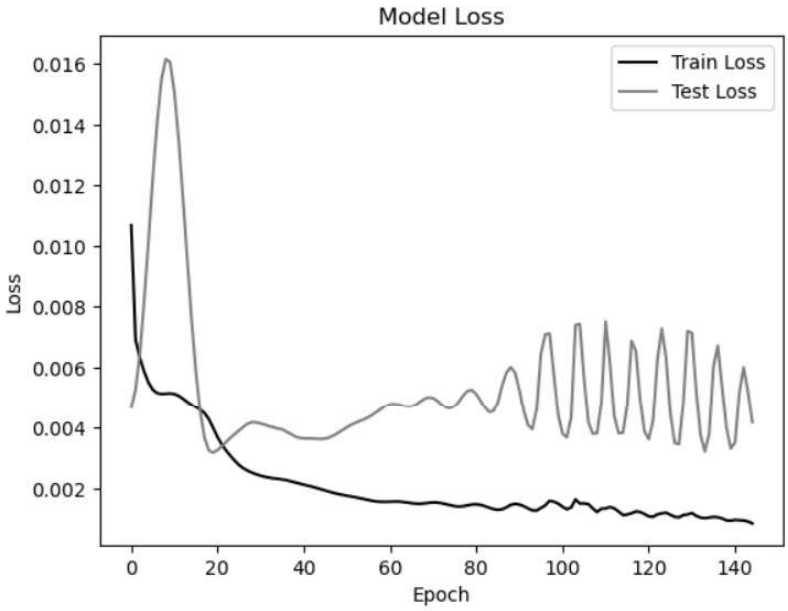

Figura 28 - Teste 7, Base de Dados 3, gráfico da função erro (MSE).

Observa-se que o *overfitting* no modelo foi alto, com o erro de teste maior que o erro de treinamento em quase todas as épocas. O erro de treinamento variar para cima e para baixo no final pode indicar uma complexidade maior no modelo do que o necessário para a base de dados.

Aplicações de regularização e redução na complexidade do modelo, como diminuir o número de neurônios da camada LSTM sequencial, não produziram resultados satisfatórios e o erro de teste continuou elevado. Dessa forma, testa-se uma nova base de dados adicionando apenas o preço histórico do ativo à base vigente. Os hiperparâmetros do modelo permaneceram inalterados.

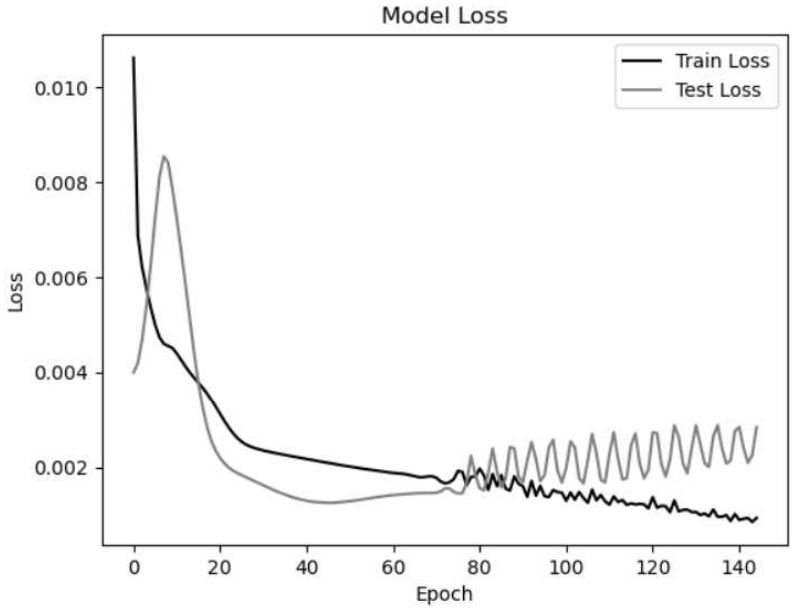

Figura 29 - Teste 8, Base de Dados 4, gráfico da função erro (MSE).

Ao adicionar o preço histórico, houve uma semelhança com o Teste 7 (Figura 28). Após a época 80, os erros de treinamento e, principalmente, de teste começaram a variar muito, indicando novamente certa complexidade do modelo para a base apresentada.

Por outro lado, próximo da época 10 o modelo apresentou uma queda considerável no erro de teste, algo que não ocorreu no modelo da Figura 28. Os valores próximos da época 40 aparentaram ser os melhores no modelo, dessa forma, o próximo teste apenas reduzirá de 145 épocas para 45 épocas.

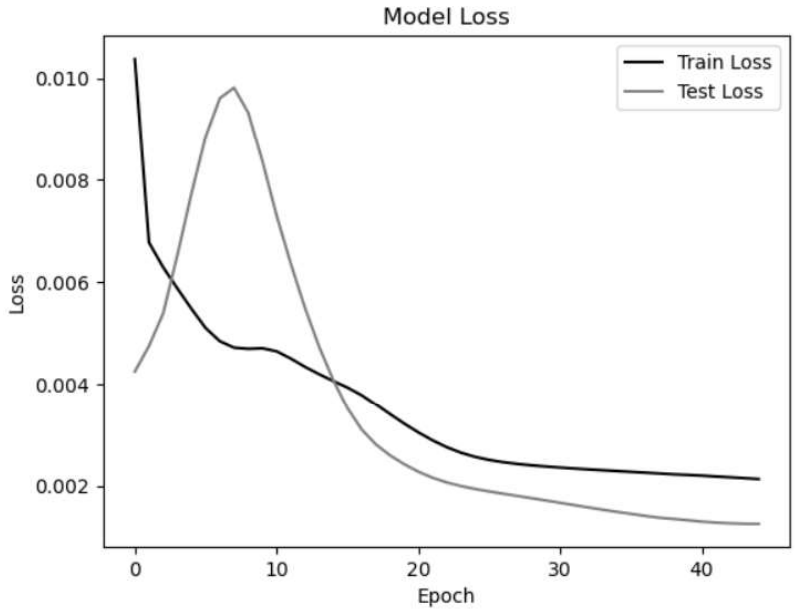

Figura 30 - Teste 9, Base de Dados 4, gráfico da função erro (MSE).

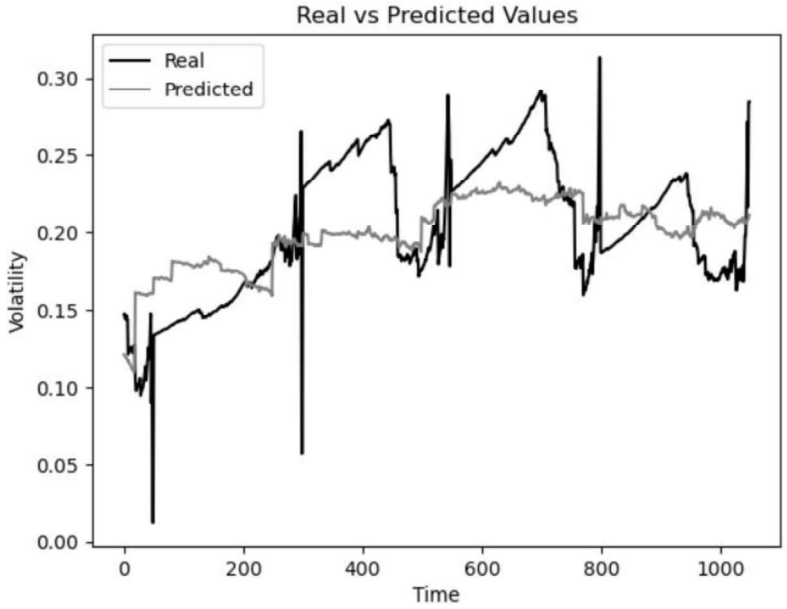

Figura 31 - Teste 9, Base de Dados 4, gráfico de valores reais e valores previstos no teste na última época.

Os resultados iniciais da Figura 30 aparentam ser melhores, com ambos os erros de teste e treinamento diminuindo até a última época. Comparando o Teste 9 com a Base de Dados 2, que continha todas e apenas as variáveis de histórico de negociação, no Teste 6 (Figuras 26 e 27) que possuía os melhores resultados até agora, os valores do RMSE na última época do modelo e teste atual, foram 4,6151% no treinamento (contra 4,5324%) e 3,5470% no teste (contra 4,0056%).

A Figura 31 aparenta ter resultados piores que a Figura 27 – ou pelo menos mostra-se com viés estatístico menor – no entanto, o MAE do Teste 9 foi 4,1932% (contra 4,7190%). Dessa forma, com exceção do erro de treinamento, o modelo atual apresentou os melhores resultados dentre os demais.

O MSE e RMSE maiores no treinamento, comparando com Teste 6, podem indicar principalmente dois fatores; há uma diferença no comportamento dos dados separados para treinamento, o que pode ocorrer por conta das demais variáveis independentes incluídas na Base de Dados 4 e; o modelo com menos épocas tem menos tempo de minimizar o erro de treinamento (em uma linguagem mais didática, tem menos tempo para aprender os dados de treinamento).

De qualquer forma, o Teste 9 foi o melhor em generalizar os dados, sem indicar um *underfitting*, ao manter os erros de teste menores. Além disso, menos épocas podem são uma vantagem por demandar menos esforço computacional. Relembrando a Seção 4.4.5, a capacidade de um modelo refere-se à sua simplicidade e eficiência em trabalhar dados diferentes.

Alguns ajustes nos hiperparâmetros foram testados para tentar melhorar mais a eficiência do modelo, no entanto, nenhum promoveu resultados satisfatórios que compensassem uma mudança. As alterações que mais chegaram próximas foram; aumentar os neurônios das demais camadas intermediárias para 50; aplicar uma regularização L2 de 0,0005 na camada intermediária densa; alterar a função da camada intermediária densa de linear para Leaky ReLu com alpha em 0,01 e; aumentar o número de épocas de 45 para 50.

Os resultados desses ajustes são demonstrados abaixo, nas Figuras 32 e 33.

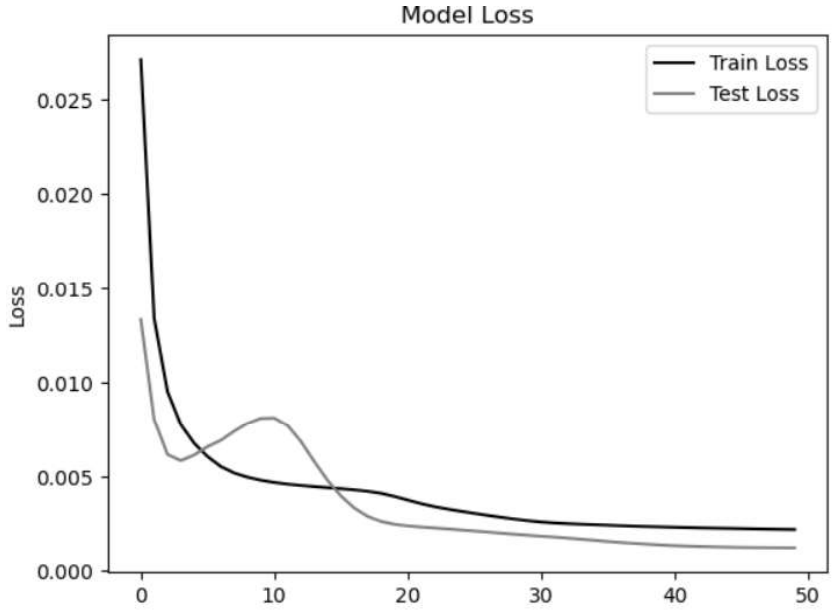

Figura 32 - Teste 10, Base de Dados 4, gráfico da função erro (MSE).

O RMSE diminuiu, no erro de teste foi para 3,4159% (de 3,5470%) e no treinamento foi para 4,6114% (de 4,6151%).

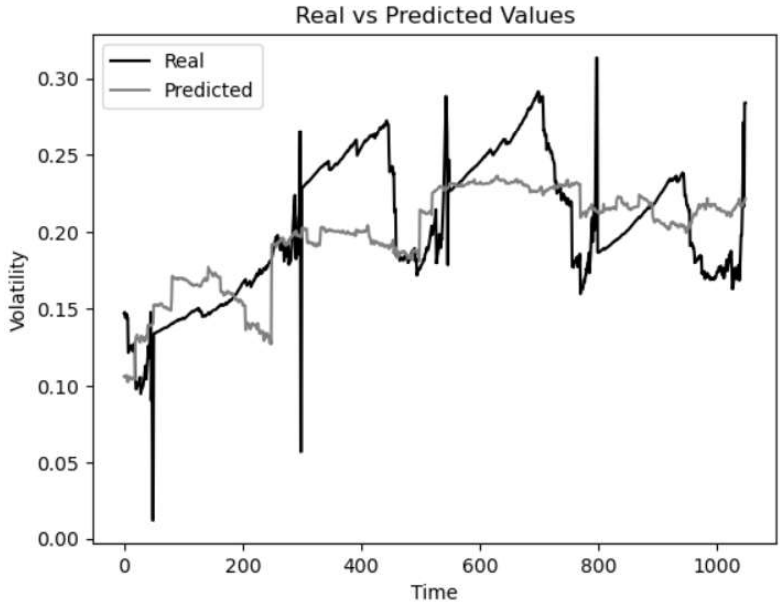

Figura 33 - Teste 10, Base de Dados 4, gráfico de valores reais e valores previstos no teste na última época.

O MAE aumentou para 4,5300% (de 4,1932%). Como o modelo ficou mais complexo e o aumento do MAE foi mais considerável que a queda no RMSE de teste, a próxima base de dados (número 5) utilizará como hiperparâmetros iniciais os mesmos do Teste 9, das Figuras 30 e 31.

Recapitulando esses hiperparâmetros, o modelo inicial para a Base de Dados 5 seguirá com 45 épocas, tamanho dos lotes em 32, uma camada LSTM sequencial com 50 neurônios, uma camada LSTM não sequencial com 25 neurônios, uma camada intermediária densa de 25 neurônios com função linear. Não há qualquer tipo de regularização.

As Bases de Dados 1 e 2 demostraram que o incremento de todas as *features* relacionadas ao histórico de negociação colaboraram para diminuir os erros do modelo. As Bases de Dados 3 e 4 demostraram que as *features* de oferta e demanda, sozinhas, não são satisfatórias para treinar o modelo, mas ao incrementar os dados de preço histórico, já produzem resultados mais promissores.

A Base de Dados 5, a última que será aplicada, constará com todas as variáveis independentes disponíveis - temporais, de histórico de negociação, oferta e demanda. As Figuras 34 e 35 demostram os resultados do primeiro teste com a Base de Dados 5.

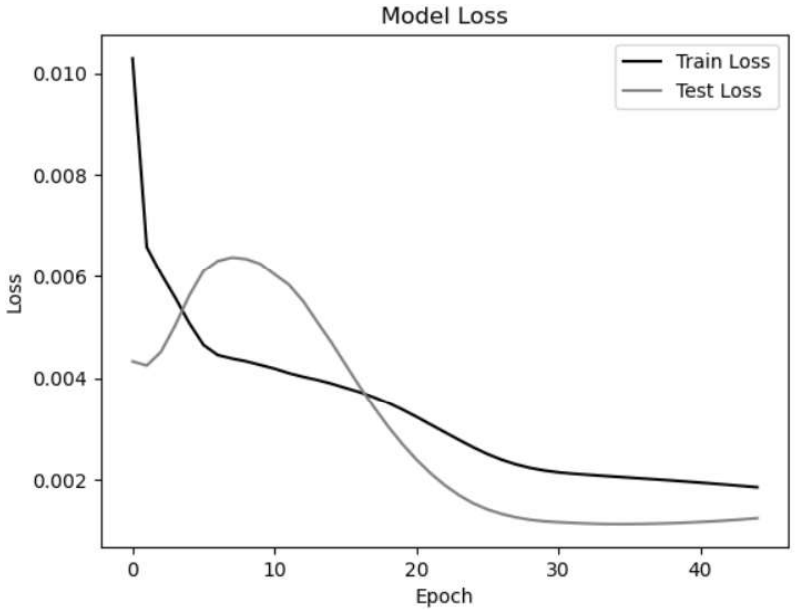

Figura 34 - Teste 11, Base de Dados 5, gráfico da função erro (MSE).

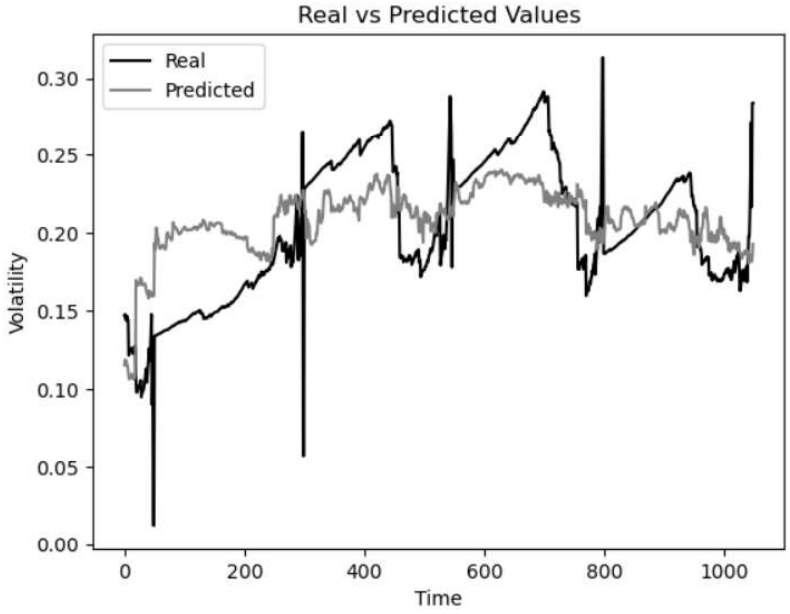

Figura 35 - Teste 11, Base de Dados 5, gráfico de valores reais e valores previstos no teste na última época.

O Teste 11 teve resultados iniciais promissores. Na última época, o RMSE de treinamento caiu para 4,3012% (de 4,6151% no Teste 9), de teste para 3,5283% (de 3,5470% no Teste 9) e o MAE de teste para 4,1620% (de 4,1932% no Teste 9). Além disso, o modelo perde capacidade de generalização próximo da época 35, então, o próximo teste apenas reduzirá as épocas, de 45 para 35. Resultados demonstrados nas Figuras 36 e 37.

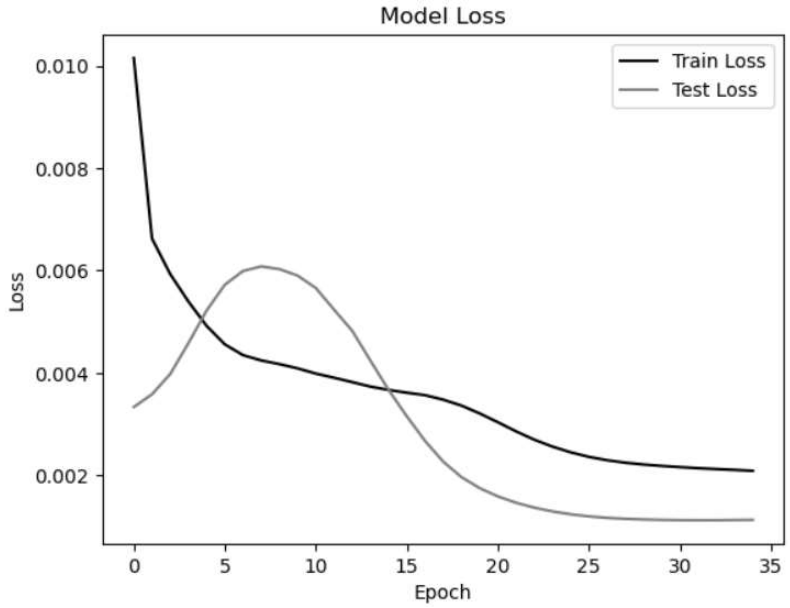

Figura 36 - Teste 12, Base de Dados 5, gráfico da função erro (MSE).

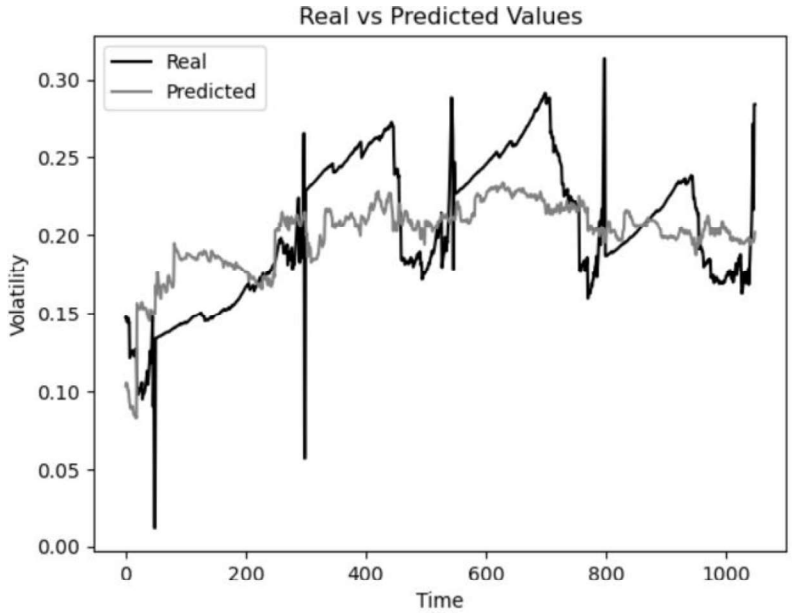

Figura 37 - Teste 12, Base de Dados 5, gráfico de valores reais e valores previstos no teste na última época.

Após o ajuste de épocas, na última, o RMSE de treinamento aumentou para 4,5684% (de 4,3012%), como era possível de se esperar, dado o menor número de épocas. O RMSE de teste reduziu para 3,3594% (de 3,5283%) e o MAE teve uma alta, para 4,2114% (de 4,1620%). Como a redução do RMSE de teste foi proporcionalmente maior que o aumento no MAE, até o momento, o modelo atual demonstra ser o mais efetivo.

No entanto, conforme já visto nessa seção, apenas reduzir as épocas pode não ser a melhor alternativa para o *overffiting*. No caso da Base de Dados 5, como ela já incluí mais variáveis independentes, é importante testar alternativas de regularização do modelo.

O primeiro ajuste foi aumentar para 50 neurônios as demais camadas intermediárias, que estavam em 25. Na segunda camada intermediária LSTM, foi aplicado uma regularização L2 em 0,1% e na camada densa intermediária, L2 em 0,5%. Relembrando, a regularização L2 tem o intuito de reduzir problemas de multicolinearidade. O número de épocas foi aumentado, de 35, para 70.

Por último, a função ativação da camada intermediária densa foi alterada de linerar para Leaky ReLU, com alpha de 0,01. Relembrando, essa função normalmente é eficiente na convergência do treinamento, em mitigar o desaparecimento do

gradiente e em evitar que neurônios parem de responder a variações no treinamento. Resultados das alterações demonstrados nas Figuras 38 e 39 abaixo.

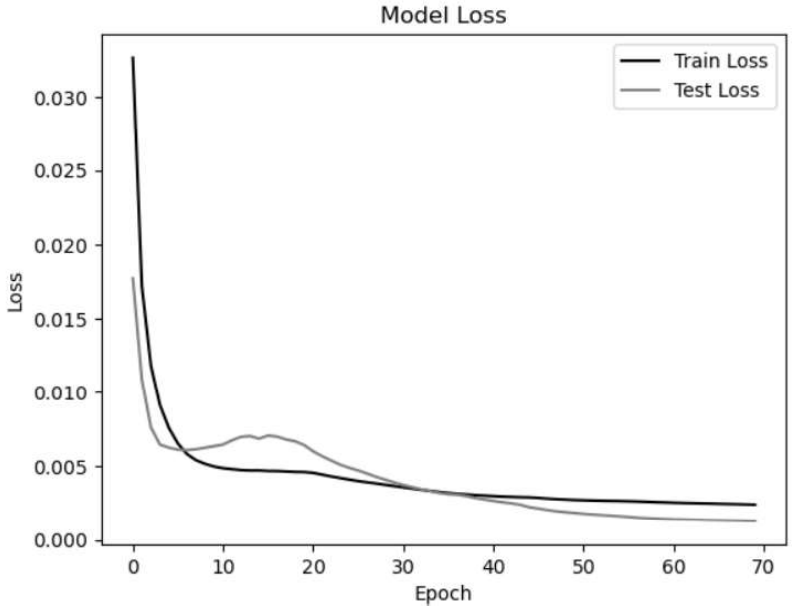

Figura 38 - Teste 13, Base de Dados 5, gráfico da função erro (MSE).

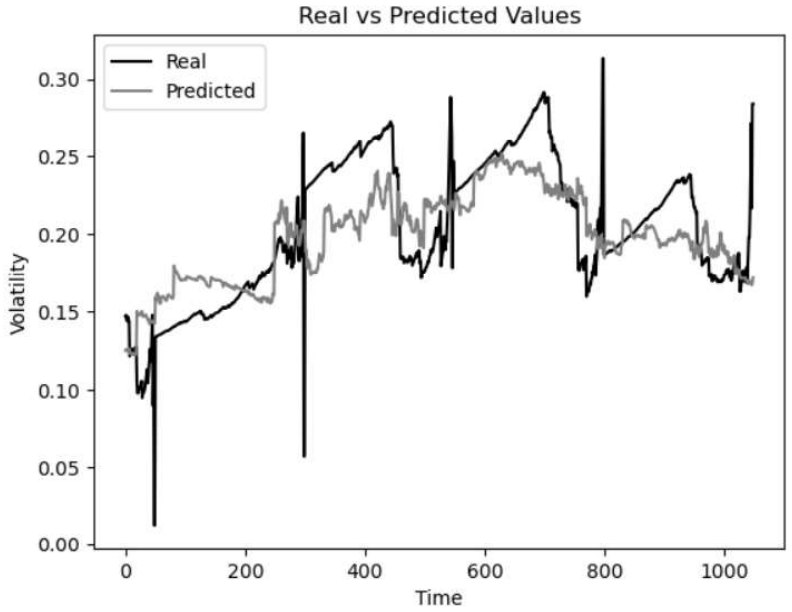

Figura 39 - Teste 13, Base de Dados 5, gráfico de valores reais e valores previstos no teste na última época.

Note-se que se parasse o treinamento na época 45, conforme feito no Teste 11, nas Figuras 34 e 35, o erro ainda seria alto. Incrementando mais épocas no Teste 13, o modelo reduziu ambos RMSE. Para treinamento, na última época, o RMSE foi de 4,4598% (de 4,5684% no Teste 12 e 4,3012% no Teste 11) e o RMSE de teste, para 2,9812% (de 3,3594% no Teste 12).

O MAE aumentou para 4,4148% (de 4,2114% no Teste 12 e 4,1620% no Teste 11), demonstrando novamente que o modelo foi eficiente em reduzir sua função erros para valores mais relevantes, mas aumenta a diferença dos erros menores.

A aplicação de novos ajustes no modelo não produziu resultados satisfatórios. Dessa forma, como toda a base de dados de variáveis disponível para esse trabalho já foi aplicada e há uma quantidade satisfatória de testes de modelos realizados, conclui-se essa seção.

5.3) Analisando o Modelo

O Teste 13 demonstrou ser o mais eficiente, logo, a análise será feita a partir dele.

Ao considerar que a volatilidade realizada anualizada da base de dados, na média, fica em torno de 23%, o erro RMSE no Teste 13 em 2,9812% representa, aproximadamente, 13% do valor médio da volatilidade realizada. Já o MAE em 4,4148% representa quase 19,2% do valor total.

Em outras palavras, esse resultado significa que a dimensão do erro é, na média, a um quinto do valor real. É possível debater se esse valor é um bom resultado ou não. Por um lado, ao utilizar o MAE como referência, o modelo captura um pouco mais de 80% da tendencia de variação real de uma variável impactada por diversos fatores. Por outro lado, a dimensão do erro ainda em 20% do valor médio, pode levar a uma decisão de operação equivocada.

Para aprofundar a análise, compara-se os resultados do modelo com o índice de volatilidade CVOL (explorado na Seção 3.6.3) para o primeiro contrato futuro de soja. Em setembro de 2019, o índice CVOL teve um erro médio absoluto comparado aos valores reais de ~3,26%, menor que o modelo LSTM, de ~3,98% no período. Já em outubro de 2019, por exemplo, o erro absoluto médio do índice CVOL foi ~6,57% e o modelo LSTM em ~4,74%.

A comparação inicial foi feita nos meses de setembro e outubro pois o CVOL utiliza o primeiro vencimento futuro para calcular o índice. Logo, nesses dois meses, o índice considera as opções com vencimento em novembro para cálculos. Ao utilizar o CVOL em um período diferente, por exemplo, em março de 2020, o índice utilizará como referência as opções de vencimento em abril, dificultando a comparação.

De qualquer maneira, a partir dos valores projetados em março de 2020 pelo CVOL, comparando com a volatilidade realizada real para novembro de 2020, a diferença absoluta média foi de ~6,32% contra ~2,84% no modelo LSTM. Esse período foi no início da pandemia de COVID-19, no qual houve um momento de forte aversão a risco e a maioria dos ativos financeiros aumentaram a volatilidade.

Essa comparação reforça a tese de que, devido a função erro da RNA ser MSE, que prioriza diminuir os erros maiores e que, por sua vez, acontecem em momentos de maior volatilidade, o modelo LSTM obteve resultados mais satisfatórios.

Conclui-se que nenhuma das ferramentas – seja o índice CVOL ou o modelo LSTM – sempre produzirá resultados muito próximos da realidade. No entanto, utilizar ambas como referência para tomadas de decisão parece ser uma boa alternativa

Novamente, um possível ajuste na base de dados, principalmente em relação à variável dependente e a preparação feita para deixá-la como uma série continua, poderia aprimorar os resultados do modelo. Os ruídos presentes na base impactaram o desempenho em relação aos erros absolutos. Ainda assim, em algumas situações, o algoritmo já teve melhor performance do que as projeções do mercado, demonstradas pelo índice CVOL.

Isso demonstra um promissor potencial em aplicar modelos LSTM para esse tipo de problema.

6) Conclusão

No terceiro capítulo, foram discutidas as problemáticas inerentes ao objeto de investigação e à questão norteadora desta pesquisa. A volatilidade esperada de um ativo financeiro exerce influência significativa sobre o valor das opções e, por extensão, sobre as estratégias de negociação correspondentes. A determinação de um valor para a volatilidade futura constitui uma tarefa de complexa, dessa forma, ferramentas analíticas capazes de auxiliar nesta definição permitem ao participante de mercado melhorar sua eficiência nas operações com opções.

O capítulo subsequente, o quarto, dedicou-se à exposição detalhada da ferramenta proposta para a resolução do problema em análise, abrangendo desde aspectos gerais de RNAs até conceitos mais específicos e importantes para a plena compreensão do modelo que será desenvolvido na presente seção.

Por fim, no quinto capítulo realizou-se um estudo prático que englobou os conceitos discorridos no terceiro e quarto capítulo. O resultado demonstrou um bom potencial em utilizar modelos LSTM para prever a volatilidade de ativos financeiros. Mesmo em uma tarefa mais complexa, devido à base de dados ter sido manipulada para criar uma série continua, a partir de contratos com vencimentos específicos, o modelo de RNA conseguiu mais acurácia em algumas situações do que ferramentas convencionais.

Ademais, conforme exposto no capítulo introdutório, o escopo deste trabalho transcende a simples apresentação de um modelo preditivo de volatilidade; ele visa a realizar um estudo abrangente sobre a volatilidade no contexto das opções financeiras e elucidar como um modelo baseado em técnicas de aprendizado de máquina pode ser estruturado para auxiliar na predição da volatilidade de ativos.

O propósito, especialmente no que tange às RNAs, consistiu não em discutir conceitos de alta complexidade, mas sim em possibilitar que até leitores com pouca familiaridade com os temas possam adquirir uma base de conhecimento robusta, suficiente para compreender integralmente o modelo proposto e, potencialmente, aplicar estes conhecimentos em contextos diversos.

Um bom caminho de investigação para trabalhos futuros seria se aprofundar em variações dos modelos RNAs; testar a capacidade do modelo apresentado no

Capítulo 5 em diferentes ativos ou com uma diferente manipulação da base de dados e; comparar os resultados com mais ferramentas, por exemplo, com regressões lineares.

Referências

- ENGLE, R. F. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. Econometrica: Journal of the Econometric Society, v. 50, n. 4, p. 987, 1982. Disponível em: https://doi.org/10.2307/1912773. Acesso em: 11/11/2023.
- ANDERSEN, T. G.; BOLLERSLEV, T.; DIEBOLD, F. X.; EBENS, H. The distribution of realized stock return volatility. Journal of Financial Economics, v. 61, n. 1, p. 43-76, 2001. Disponível em: https://doi.org/10.1016/S0304-405X(01)00055-1. Acesso em: 11/11/2023.
- HULL, J.; WHITE, A. The pricing of options on assets with stochastic volatilities. The Journal of Finance, v. 42, n. 2, p. 281-300, 1987. Disponível em: https://doi.org/10.1111/j.1540-6261.1987.tb02568.x. Acesso em: 11/11/2023.
- ANDERSEN, T. G.; BOLLERSLEV, T.; DIEBOLD, F. X.; EBENS, H. The distribution of realized stock return volatility. Journal of Financial Economics, v. 61, n. 1, p. 43-76, 2001. Disponível em: https://doi.org/10.1016/S0304-405X(01)00055-1. Acesso em: 11/11/2023.
- MÜLLER, U. et al. Fractals and intrinsic time - A challenge to econometricians. UAM, 1993.
- MANDELBROT, B. The variation of certain speculative prices. The Journal of Business, v. 36, n. 4, p. 394, 1963. Disponível em: https://doi.org/10.1086/294632. Acesso em: 11/11/2023.
- CORSI, F. A simple approximate long-memory model of realized volatility. Journal of Financial Econometrics, v. 7, n. 2, p. 174-196, 2009. Disponível em: https://doi.org/10.1093/jjfinec/nbp001. Acesso em: 11/11/2023.
- KARAALI, O.; EDELBERG, W.; HIGGINS, J. Modelling volatility derivatives using neural networks. In: Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr), New York, NY, USA, 1997. p. 280-286.
- JOHNSSON, Ola. Predicting Stock Index Volatility Using Artificial Neural Networks: An empirical study of the OMXS30 FTSE100 & S&P/ASX200. Junho de 2018.
- DANNSTRÖM, C. O.; BROANG, A. Volatility Forecasting with Artificial Neural Networks: Can we trust them? 2022. Dissertation. Disponível em: https://urn.kb.se/resolve?urn=urn:nbn:se:su:diva-218673. Acesso em: 12/11/2023.

- HULL, John C. Introdução aos mercados futuros e opções. 8. ed. São Paulo: BM&F Bovespa, 2012. xiv, 598 p.
- OPDYKE, J. D. Comparing Sharpe ratios: So where are the p-values?. Journal of Asset Management, v. 8, p. 308-336, 2007. Disponível em: https:// https://papers.ssrn.com/sol3/papers.cfm?abstract_id=886728. Acesso em: 11/10/2023.
- BUSSAB, Wilton de O.; MORETTIN, Pedro A. Estatística Básica. 9. ed. São Paulo: Saraiva, 2017. xx, 568 p.
- PEARSON, K. Contributions to the mathematical theory of evolution.—II. Skew variation in homogeneous material. Philosophical Transactions of the Royal Society of London. Series A, v. 186, p. 343-414, 1895. Disponível em: http://doi.org/10.1098/rsta.1895.0010. Acesso em: 11/10/2023.
- OAKES, M. Ord's criterion with word length spectra for the discrimination of texts, music and computer programs. In: Communications in Statistics - Theory and Methods. [S.l.], v. 48, p. 2286-2304, 2007. Disponível em: https://doi.org/10.1515/9783110894219.509. Acesso em: 11/10/2023.
- BOYLE, P. P. Options: A Monte Carlo approach. Journal of Financial Economics, v. 4, n. 3, p. 323-338, 1977. ISSN 0304-405X. Disponível em: https://doi.org/10.1016/0304-405X(77)90005-8. Acesso em: 15/10/2023.
- LERCHE, I.; MUDFORD, B. S. How Many Monte Carlo Simulations Does One Need to Do? Energy Exploration & Exploitation, v. 23, n. 6, p. 405-427, 2005. DOI: 10.1260/014459805776986876.
- MCCRARY, S. Implementing a Monte Carlo Simulation: Correlation, Skew, and Kurtosis. Berkeley Research Group [S.l.], 23 set. 2015. Disponível em: https://www.thinkbrg.com/insights/publications/implementing-a-monte-carlo-simulation-correlation-skew-and-kurtosis/. Acesso em: 15/10/2023.
- FLEISHMAN, A. I. A Method for Simulating Non-Normal Distributions. Psychometrika, v. 43, n. 4, p. 521-532, 1978.
- SANDMANN, G.; KOOPMAN, S. J. Estimation of stochastic volatility models via Monte Carlo maximum likelihood. Journal of Econometrics, v. 87, n. 2, p. 271-301, 1998. ISSN 0304-4076. DOI: https://doi.org/10.1016/S0304-4076(98)00016-5. Acesso em: 15/10/2023.

- BLACK, Fischer; SCHOLES, Myron. The Pricing of Options and Corporate Liabilities. Journal of Political Economy, v. 81, n. 3, p. 637-654, 1973. Disponível em: http://www.jstor.org/stable/1831029. Acesso em: 16/10/2023.
- CME GROUP. CME Group Volatility Indexes. Disponível em: https://www.cmegroup.com/market-data/cme-group-benchmark-administration/cme-group-volatility-indexes.html. Acesso em: 05/11/2023.
- NIELSEN, Michael A. Neural Networks and Deep Learning. Determination Press, 2015. Disponível em: http://neuralnetworksanddeeplearning.com/. Acesso em: 08/10/2023.
- GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. Deep Learning. Cambridge: MIT Press, 2016.110.
- KINGMA, Diederik P.; BA, Jimmy. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
- Soybean 2023 World Production. USDA. Disponível em: https://ipad.fas.usda.gov/cropexplorer/cropview/commodityView.aspx?cropid=2222000. Acesso em: 24/12/2023.

Apêndice I – Código Geral Utilizado Para Criar o Modelo LSTM em Python

O código abaixo pode ser copiado e colado em um *notebook* em Python. A biblioteca "Tensor Flow" possui estruturas prontas para executar modelos de RNAs. Os parâmetros do modelo, que podem ser alteradas conforme experimentação, estão identificadas como #*Variável* no código.

```
# Importando as bibliotecas necessárias. Pode ser necessário a instalação prévia.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout, LeakyReLU
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import regularizers
from tensorflow.keras.metrics import RootMeanSquaredError
from sklearn.metrics import mean_squared_error, mean_absolute_percentage_error, r2_score
from sklearn.model_selection import train_test_split
from math import sqrt

# Inicialização aleatória.
np.random.seed(1)
tf.random.set_seed(1)

# Carregando o arquivo da base de dados.
data = pd.read_csv(#NOME DO ARQUIVO EM .CSV, sep=';', encoding='ISO-8859-1')

#  Separando os valores das variáveis independentes (Features) e da variável dependente, assumindo que todas as colunas, com exceção da última, são features.
X = data.iloc[:, :-1].values
y = data.iloc[:, -1].values
```

```
# Dividindo os dados em treinamento e teste. No exemplo abaixo, 20% (0.20) dos dados são para teste e o resto (80%) para treinamento.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42, shuffle=False)

# Reestruturando a entrada de dados para ser [amostra, passo temporal, variável independente].
X_train = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
X_test = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))

# Criando o modelo LSTM
# Definindo Hiperparâmetros
batch_size= 32 #Variável, tamanho do lote.
epochs= 50 #Variável, número de épocas.
loss_function='mean_squared_error' #A função de erro deve ser incluída a partir das funções disponíveis na biblioteca Tensor Flow. No caso, utilizou-se erros quadrados médios.
optimizer='adam' #Os otimizadores, caso utilizados, devem também ser incluídos conforme disponível na biblioteca Tensor Flow.

# Definindo parâmetros do modelo. Cada 'model.add' é uma adição de camada após a camada de entrada, que podem ser LSTM, densa ou outra categoria disponível no Tensor Flow.
#Variáveis de cada camada; 'units', número de neurônios; 'return_sequences', determina se a camada LSTM deve processar os dados sequencialmente; 'dropout', taxa de dropout; 'l1' e 'l2', taxa para cada tipo de regularização.
def create_model():
    model = Sequential()
    model.add(LSTM(units= 50, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2]), dropout=0.0000, kernel_regularizer=regularizers.l1_l2(l1=0.000, l2=0.000)))
    model.add(LSTM(units= 50, return_sequences=False, kernel_regularizer=regularizers.l1_l2(l1=0.000, l2=0.000)))
```

```
    model.add(Dense(units=25))
    #Caso deseje adicionar a função erro Leaky ReLU na camada densa, utilizar a linha de código abaixo.
    #model.add(LeakyReLU(alpha=0.01))
    model.add(Dense(units=1)) #Camada de saída.
    model.compile(optimizer=optimizer, loss=loss_function, metrics=[RootMeanSquaredError()]) #Cálcula também o RMSE em cada época, apenas como uma métrica adicional, não de treinamento.
    return model

# Nomeando o modelo.
model = create_model()

# Treinando o Modelo
history = model.fit(X_train, y_train, epochs=epochs, batch_size=batch_size, validation_data=(X_test, y_test), shuffle=False, verbose=0)

# Avaliação do Modelo
history_df = pd.DataFrame(history.history, columns=['loss', 'val_loss', 'root_mean_squared_error', 'val_root_mean_squared_error'])

# Exibir a tabela de dados
print(history_df)

# Exibir os gráficos da função erro de treinamento e teste em relação às épocas.
plt.plot(history_df['loss'], label='Train Loss', color='black')
plt.plot(history_df['val_loss'], label='Test Loss', color='grey')
plt.title('Model Loss', color='black')
plt.ylabel('Loss', color='black')
plt.xlabel('Epoch', color='black')
plt.legend()
plt.show()
```

```
# Organizando a variável de valores previstos pelo modelo.
y_pred = model.predict(X_test, verbose=0)]

# Criando gráfico dos valores reais e previstos na última época.
plt.plot(y_test, label='Real', color='black')
plt.plot(y_pred, label='Predicted', color='grey')
plt.title('Real vs Predicted Values', color='black')
plt.xlabel('Time', color='black')
plt.ylabel('Volatility', color='black')
plt.legend()
plt.show()

# Calcula e apresenta a diferença absoluta média.
average_difference = np.mean(np.abs(y_pred - y_test))
print('Average Difference:', average_difference)
plt.scatter(y_test, y_pred, alpha=0.5,color='grey')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], '--k')  # Line y = x
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Actual vs. Predicted')
plt.show()
```

Apêndice II – Tabela Consolidada de Testes e Resultados do Modelo

Teste	Base de Dados	Figuras	Variáveis Utilizadas	Nº Neurônios 1ª Camada LSTM	Nº Neurônios 2ª Camada LSTM	Nº Neurônios Camada Densa	Função Ativação Camada Densa	Tamanho do Lote	Épocas	Regularização	RMSE (Teste)	MAE (Teste)
1	1	16, 17	Temporais (Todas); Históricos de Negociação (Preço Histórico).	25	25	25	Linear	32	200	N/A	9.70%	7.05%
2	1	18, 19	Temporais (Todas); Históricos de Negociação (Preço Histórico).	25	25	25	Linear	32	110	N/A	6.16%	4.37%
3	1	20, 21	Temporais (Todas); Históricos de Negociação (Preço Histórico).	25	25	25	Linear	32	200	LSTM 1: L2 (0.0001)	5.48%	4.32%
4	2	22, 23	Temporais (Todas); Históricos de Negociação (Todas).	25	25	25	Linear	32	200	N/A	4.90%	5.39%
5	2	24, 25	Temporais (Todas); Históricos de Negociação (Todas).	25	25	25	Linear	32	110	N/A	4.20%	4.35%
6	2	26, 27	Temporais (Todas); Históricos de Negociação (Todas).	50	25	25	Linear	32	145	N/A	4.01%	4.72%
7	3	28	Temporais (Todas); Históricos de Negociação (Volume de Negociação); Oferta (Todas); Demanda (Todas).	50	25	25	Linear	32	145	N/A	6.93%	5.18%
8	4	29	Temporais (Todas); Históricos de Negociação (Volume de Negociação, Preço de Negociação); Oferta (Todas); Demanda (Todas).	50	25	25	Linear	32	145	N/A	5.29%	4.87%
9	4	30, 31	Temporais (Todas); Históricos de Negociação (Volume de Negociação, Preço de Negociação); Oferta (Todas); Demanda (Todas).	50	25	25	Linear	32	45	N/A	3.55%	4.19%
10	4	32, 33	Temporais (Todas); Históricos de Negociação (Volume de Negociação, Preço de Negociação); Oferta (Todas); Demanda (Todas).	50	50	50	LeakyReLu (0.01)	32	50	Densa: L2 (0.0005)	3.42%	4.53%
11	5	34, 35	Temporais (Todas); Históricos de Negociação (Todas); Oferta (Todas); Demanda (Todas).	50	25	25	Linear	32	45	N/A	3.53%	4.16%
12	5	36, 37	Temporais (Todas); Históricos de Negociação (Todas); Oferta (Todas); Demanda (Todas).	50	25	25	Linear	32	35	N/A	3.36%	4.21%
13	5	38, 39	Temporais (Todas); Históricos de Negociação (Todas); Oferta (Todas); Demanda (Todas).	50	50	50	LeakyReLu (0.01)	32	70	LSTM 2: L2 (0.0001); Densa: L2 (0.0005)	2.98%	4.41%