

**PROGRAMA DE EDUCAÇÃO CONTINUADA –
ESCOLA POLITÉCNICA DA USP**
MASTER OF BUSINESS ADMINISTRATION EM ENGENHARIA FINANCEIRA

LUCAS EDUARDO LIMA ORSI

**APREÇAMENTO DE OPÇÕES DE COMPRA VIA INTELIGÊNCIA
ARTIFICIAL – ESTUDO COMPARATIVO COM O MODELO
BLACK-SCHOLES-MERTON**

São Paulo

2022

LUCAS EDUARDO LIMA ORSI

**APREÇAMENTO DE OPÇÕES DE COMPRA VIA INTELIGÊNCIA
ARTIFICIAL – ESTUDO COMPARATIVO COM O MODELO
BLACK-SCHOLES-MERTON**

Monografia apresentada à Escola Politécnica da
Universidade de São Paulo como requisito para a
conclusão do curso de *Master of Business
Administration* em Engenharia Financeira.

Orientador: Prof. Dr. Márcio Menezes

São Paulo

2022

LUCAS EDUARDO LIMA ORSI

**APREÇAMENTO DE OPÇÕES DE COMPRA VIA INTELIGÊNCIA
ARTIFICIAL – ESTUDO COMPARATIVO COM O MODELO DE
BLACK-SCHOLES-MERTON**

Monografia apresentada à Escola Politécnica da Universidade de São Paulo como requisito para a conclusão do curso de *Master of Business Administration* em Engenharia Financeira

COMISSÃO JULGADORA:

Prof. Dr. Oswaldo Luiz do Valle Costa
Escola Politécnica da Universidade de São Paulo

Prof. Dr. Roberto Moura Sales
Escola Politécnica da Universidade de São Paulo

Prof. Dr. Márcio Menezes
Escola Politécnica da Universidade de São Paulo
Professor Orientador – Presidente da Banca Examinadora

São Paulo, 17 de fevereiro de 2022

Agradecimentos

Agradeço primeiramente a Deus, por me proporcionar saúde, vitalidade e oportunidades que culminaram na realização deste trabalho.

À PECE, seu corpo docente, direção e administração por me proporcionarem o conhecimento, as ferramentas e o caminho para conclusão do curso.

Ao meu orientador, professor doutor Márcio Menezes por todo apoio, prontidão e exemplo de conhecimento.

À família, pais, irmãos e namorada pela paciência, compreensão e amor dos quais sem, não seria capaz de percorrer esta longa jornada: esta conquista é nossa.

Aos amigos de classe, que forneceram a cumplicidade e a união que tanto precisamos em determinados momentos do curso, pessoas que levarei, sem dúvida alguma, para sempre em meus pensamentos e coração.

E a todas as pessoas que, direta ou indiretamente, fizeram parte da realização deste sonho, a todos vocês, o meu muito obrigado.

Apreçamento de Opções de Compra Via Inteligência Artificial – Estudo Comparativo com o Modelo Black-Scholes-Merton

Resumo

O presente estudo procura entender, através de abordagem empírica, se um modelo de rede neural (inteligência artificial) produz um apreçamento de opções de maneira mais eficaz do que o modelo padrão de Black-Scholes-Merton (“BSM”). Com base em uma amostra de opções de compra de ações preferenciais da Petróleo Brasileiro S.A. (“Petrobras”) entre os meses de agosto e outubro de 2020, uma rede neural com os mesmos *inputs* do modelo de Black-Scholes-Merton foi treinada e a raiz quadrada do Erro Quadrático Médio gerado pelo conjunto teste foi confrontado com o gerado via BSM.

Os resultados sugerem que, quando realizado o apreçamento via rede neural, os apreçamentos são mais próximos das cotações de mercado para as opções do que os apreçamentos gerados através da fórmula de Black-Scholes-Merton, apontando que a rede neural possui uma capacidade de geração de resultado mais próxima dos preços praticados pelo mercado do que quando utilizado o modelo original de maneira direta.

Palavras-chave: Black-Scholes. Opções. Inteligência Artificial. Redes Neurais.

Abstract

This study aims to understand, through an empirical approach, whether a neural network model (artificial intelligence) produces option pricing more effectively than the standard Black-Scholes-Merton (“BSM”) model. Based on a sample of call options of Petróleo Brasileiro SA (“Petrobras”) preferred shares between August and October 2020, a neural network with the same inputs as the Black-Scholes-Merton model was trained and the square root of the Mean Square Error generated by the test set was compared with the one generated via the standard BSM model.

The results suggest that when pricing via the neural network, the outputs are closer to the market option quotes than the results generated through the Black-Scholes-Merton formula, indicating that the neural network has a pricing capacity closer to market prices than when using the original model directly.

Keywords: Black-Scholes. Options. Artificial Intelligence. Neural Network.

Índice

1. Introdução.....	7
2. Revisão bibliográfica.....	10
3. Metodologia.....	14
3a. Modelo de apreçamento de opções Black-Scholes-Merton.....	14
3b. Rede Neural Artificial.....	22
3c. Dados do modelo.....	27
3d. Modelo de Rede Neural.....	29
3e. Erro Quadrático Médio (EQM).....	30
4. Resultado do estudo.....	31
5. Conclusão do estudo.....	35
Referências.....	36
Apêndice.....	38

1 Introdução

O mercado de futuros e opções, com as características que conhecemos hoje, tem suas origens no século 19, com a padronização de contratos agrícolas na *Chicago Board Trade* (CBOT) como uma tentativa de reduzir as ineficiências da precificação de *commodities* agrícolas que possuíam colheitas curtas e necessitavam de estocagem. Já nos anos de 1960, esse mercado foi estendido às *commodities* que não necessitavam de estocagem e, nos anos de 1970, para *commodities* minerais, como ouro (Carter, 2007). Ainda nos anos 80, a CBOT conseguiu colocar ordem no mercado de opções para ativos financeiros, inaugurando as negociações de opções de compra de 16 ações negociadas em bolsa e, em 1977, inaugurou as negociações de contratos de opções de venda sobre ações (Hull, 2014).

A classe de ativos que engloba opções é a dos derivativos, ou seja, derivados de outros ativos – o ativo objeto (*underlying asset*) – e sua existência deriva da dinâmica e especificidades desses ativos, cuja participação das opções constitui importante ferramenta para a realização de estratégias de *hedge*, especulações e arbitragem de preços¹. Os mecanismos utilizados nas estratégias de negociação envolvendo opções são muitos, mas algo que é de comum interesse para todos os participantes de mercado é o apreçamento, da maneira mais precisa possível, dessas opções.

As décadas do pós Segunda Guerra Mundial foram campo fértil para acadêmicos desenvolverem modelos de apreçamento de opções e aperfeiçoamentos nesse segmento através da sofisticação matemática. Temos, em 1951, o surgimento do Lema de Itô, demonstrado através de Equação Diferencial Estocástica (EDS) pelo matemático Kiyoshi Ito em seu artigo *On Stochastic Differential Equations* (Hull, 2014) e utilizado posteriormente para realizar o apreçamento de opções.

Outra contribuição para o campo foi o advento da Árvore Binomial, em 1979, cuja metodologia utilizada no apreçamento foi desenvolvida por Cox, Ross e Rubinstein (1979), consistindo em um apreçamento via tomada de decisão. A Árvore Binomial é um diagrama que estima o preço das opções através do tempo ao simular a decisão de executar (ou não) a compra da opção, utilizando como parâmetro o preço

1 Para um detalhamento dos objetivos de estratégias com opções, ver Hull (2014).

do ativo subjacente. A decisão é então tomada através da comparação de duas estratégias: (i) desembolso de caixa para compra direta da opção e (ii) replicar o efeito da opção através da compra do ativo subjacente e tomada de empréstimo.

Outros modelos de apreçamento de derivativos foram desenvolvidos nos anos 70 e nas décadas seguintes, mas o modelo dominante do mercado de opções e utilizado como parâmetro de preço justo é o modelo Black-Scholes, desenvolvido por Fischer Black e Myron Scholes (1973). O modelo de apreçamento de opções Black-Scholes utiliza como base o cálculo diferencial e um processo estocástico de evolução dos preços, fornecendo assim o preço teórico de uma opção de ação, seja ela de compra ou de venda, em que o exercício do direito sobre o contrato possa ser realizado apenas no vencimento da opção (opção estilo europeia).

Apesar da criação, nas décadas seguintes, de modelos de apreçamento de opções mais sofisticados, o modelo Black-Scholes - em sua versão de 1976, Black-Scholes-Merton (Merton, 1976) - ganhou popularidade no mercado por apresentar variáveis relativamente simples de se obter, passando a ser utilizado como parâmetro para estratégias com opções. A versão Black-Scholes-Merton² do modelo leva em consideração cinco variáveis: (i) preço (usualmente de fechamento) do ativo objeto, (ii) o *preço de exercício*³ da opção, (iii) o tempo do momento atual até o vencimento da opção (usualmente expresso em base anual), (iv) a volatilidade do preço do ativo-objeto e (v) uma taxa de referência de risco de mercado (*risk-free rate*).

O modelo Black-Scholes-Merton rendeu, nas décadas seguintes, uma série de modificações e aperfeiçoamentos por parte de outros acadêmicos matemáticos e financistas, como é o caso do modelo apresentado por Gong, Thavaneswaran e Sing (2010), que substituiu a medida de risco do ativo-objeto, expressa pela volatilidade histórica ou implícita, por um modelo de volatilidade autorregressivo GARCH (*General Autoregressive Conditional Heteroskedasticity*), adicionando uma camada de complexidade ao modelo padrão de Black-Scholes-Merton.

² Em 1976 o economista estadunidense Robert Merton desenvolveu uma forma aperfeiçoada do modelo original. Essa abordagem será tratada em seção posterior.

³ Preço pré-determinado pelo qual o detentor da opção tem o direito de adquirir/vender o ativo objeto.

Uma das mais recentes contribuições do modelo de apreçamento Black-Scholes-Merton está no campo de *Machine Learning*⁴ (Aprendizado de Máquina, em tradução livre), sendo utilizado através de sua subcategoria conhecida como Redes Neurais, para gerar preços teóricos que possuem um menor erro de apreçamento em relação ao preço de mercado, quando comparados ao modelo original. Essa associação é utilizada no trabalho de Mitra (2012), que aborda o modelo de Black-Scholes-Merton como modelo de entrada de uma Rede Neural, com o objetivo de trazer maior acurácia ao estimar o preço teórico de opções de compra do índice de ações da bolsa indiana - Nifty 50 - com prazos de vencimento entre julho de 2008 e junho de 2011. Mitra (2012) encontra indícios de que a utilização de *Machine Learning* aperfeiçoa o modelo de Black-Scholes-Merton e ajuda a reduzir o erro de apreçamento, quando comparado ao modelo original.

Utilizando como princípio o escopo de *Machine Learning*, o objetivo do presente trabalho é realizar uma regressão, através de uma Rede Neural e os parâmetros de entrada do modelo Black-Scholes-Merton, para uma série de opções de compra de ações da Petróleo Brasileiro S.A. ("Petrobras") negociadas diariamente na bolsa brasileira entre os meses de agosto e outubro de 2020. O objetivo da regressão é conseguir treinar uma Rede Neural para realizar o apreçamento das opções de compra da Petrobras. Após a simulação, propõe-se a comparação dos erros de apreçamento via Redes Neurais e o modelo original Black-Scholes-Merton em relação aos preços de mercado, verificando assim qual modelo reflete melhor os preços de mercado através do cálculo da raiz quadrada do erro quadrático médio.

4 Uma das áreas de desenvolvimento do conceito de Inteligência Artificial.

2 Revisão bibliográfica

A utilização de Redes Neurais para precificação de derivativos e instrumentos futuros é um campo já há algum tempo explorado por acadêmicos do mercado financeiro, tanto no que tange o refino de métodos de apreçamento quanto na utilização do poder preditivo de uma Rede Neural treinada.

No campo da evolução de modelos de apreçamento, Tseng, Cheng, Wang e Peng (2008) encontraram a combinação entre Redes Neurais e o apreçamento de ativos financeiros com ruído na distribuição dos dados, criando uma rede com capacidade de assimilar um modelo híbrido que considera um padrão de volatilidade não simétrico, partindo do modelo de volatilidade EGARCH (*Exponential Generalized Autoregressive Conditional Heteroscedastic*), utilizado em modelos que ferem a premissa de volatilidade constante e introduzindo o conceito de persistência da volatilidade assimétrica. O processo foi aplicado para reduzir o erro e reagir com a capacidade de previsibilidade do modelo, a Rede Neural foi então utilizada para dar maleabilidade ao modelo ao introduzirem dados de opções de compra do índice da bolsa de Taiwan (TAIEX).

Tseng *et al.* (2008) encontraram indícios que opções dentro do dinheiro (*in the money options*) podem ser mais bem precificadas a partir de uma abordagem utilizando um modelo de Rede Neural em conjunto com uma perspectiva de volatilidade a partir do modelo EGARCH, quando comparado com o modelo de apreçamento original Black-Scholes-Merton.

A utilização de aspectos de Inteligência Artificial para o estudo de volatilidade no mercado financeiro é sumariamente aplicada como ferramenta nos objetivos acadêmicos, mas o trabalho de Horvath, Muguruza e Tomas (2019) utiliza a modelagem de volatilidade como objetivo final no contexto de apreçamento de opções, ao apresentarem um modelo de *Deep Learning*⁵ que realiza a calibragem da volatilidade implícita de toda a superfície de volatilidade. O modelo apresentado por Horvath *et al.* (2019) possui robustez ao utilizar uma abordagem generalista, abrangendo

⁵ Subtipo de Rede Neural. Para maiores referências sobre o conceito de *Deep Learning*, ver Vargas, Mosavi e Ruiz (2017).

uma gama de modelos de volatilidade e sendo aplicável a diversos contratos derivativos, debatendo o tema sob a perspectiva da acuracidade dos resultados, velocidade de processamento da rede, robustez do modelo e generalização da aplicação dos resultados.

As conclusões encontradas por Horvath *et al.* (2019) apontam para uma solução universalizada - no que tange a aplicação de Redes Neurais - para estimadores de modelos de volatilidade. Os autores realizam experimentos com o modelo padrão de Black-Scholes-Merton, com o modelo de apreçamento de Heston e com o modelo de Bergomi, mostrando a resiliência dos resultados frente às diversas abordagens matemáticas de apreçamento.

Já para temas correlatos com o presente estudo, o uso de *Machine Learning* com ênfase em índices financeiros de ações também possui grande espaço no meio acadêmico, seja no formato preditivo, seja para treinar a máquina para sugerir calibrações mais precisas e, assim, reduzir o erro amostral.

Zhong e Enke (2019) aproveitam esse tópico para gerar uma base de dados com os retornos diários do SPDR S&P 500 ETF - um fundo de ações negociado na bolsa (*Exchange-Traded Fund - ETF*) que replica o índice S&P 500 da Standard & Poor's - para alimentar seu algoritmo com base em DNN (*Deep Neural Network*) e gerar previsões em relação aos futuros retornos do índice.

Os autores aplicaram inteligência artificial com o objetivo de realizar a previsão da direção dos retornos diários da ETF, ou seja, não estavam necessariamente interessados na magnitude dos retornos, apenas em deixar a rede treinada para prever o direcionamento (retornos positivos ou negativos no futuro). Zhong e Enke (2019) utilizaram um procedimento híbrido entre Redes Neurais e PCA⁶ com base em uma amostra de 2.518 dias de negociação do fundo e levando em consideração 60 fatores como *inputs* do modelo (retorno histórico, médias móveis, referência de ativos livres de risco, diferença cambial entre o dólar e outras quatro moedas, etc.).

⁶ Zhong e Enke (2019) utilizam a técnica de transformação de dados brutos conhecida como PCA (*Principal Component Analysis*) para chegarem nos resultados apresentados.

Zhon e Enke (2019) encontram indícios de que a abordagem híbrida possui acurácia significativa na previsão da direção dos retornos diários do fundo SPDR S&P 500 ETF. Já no que tange às informações apresentadas, os resultados encontrados nesse tipo de estudo (e em outros modelos apontados na literatura) podem, inclusive, servir como possíveis instrumentos para investidores realizarem suas estratégias de alocação de recursos e *hedge* de posição de fundos (fundos quantitativos utilizam esse tipo de análise, por exemplo, para traçarem suas estratégias de alocação de recursos e tomadas de posição, principalmente *hedge funds* quantitativos⁷).

Outro campo de estudo no meio acadêmico envolvendo *Machine Learning* e mercado financeiro e que extrapola as análises quantitativas é a utilização de dados brutos gerados por redes sociais como *proxy* para sentimento político ou de comportamento social que, por vezes, enviesa os rumos da economia e mercado financeiro. Pimprikar, Ramachandran e Senthilkumar (2017) analisaram a fundo essa correlação: através da utilização de algoritmos via *Machine Learning*, os autores captaram padrões de comportamento social no Twitter que ajudam a prever movimentos no mercado de ações. Pimprikar *et al.* (2017) coletaram dados dos retornos diários do índice da bolsa norte-americana Dow Jones (*Dow Jones Industrial Average*) e utilizaram em modelos de Regressão Linear, *Support Vector Machine* (SVM)⁸ e Redes Neurais em conjunto com o *Twitter Sentiment Analysis*, espécie de algoritmo de classificação de texto que utiliza a rede social Twitter como base de dados.

Esse tipo de ferramenta é utilizada, no campo de ciência computacional, para identificar e classificar o sentimento presente nos textos publicados em redes sociais, e compõe os recursos usados por empresas e agências de *marketing* para entender melhor o comportamento do consumidor. Pimprikar *et al.* (2017) aproveitam a enorme quantidade de dados presente nesse tipo de plataforma para treinar a Rede Neural. A ideia dos autores foi classificar entre positivo, negativo ou neutro os textos que eram postados na plataforma sobre as empresas listados no índice. Com os resultados, os

7 Para um detalhamento da utilização de Inteligência Artificial na gestão de ativos financeiros, ver Bartram, Branke e Motahari (2020).

8 Metodologia de aprendizado supervisionado capaz de realizar análise de dados brutos, encontrar padrões associados aos dados e classificá-los de acordo com esses padrões encontrados.

autores concluem que a análise via sentimento apenas influencia o preço das ações quando há uma determinada polarização entorno da notícia, ou seja, quando mais de 80% das notícias estão mostrando sentimentos positivos em relação às empresas das ações analisadas.

Com inspiração na literatura já existente sobre o tema, na próxima seção iremos explorar a metodologia utilizada no presente trabalho na tentativa de melhorar a acuracidade do apreçamento de opções de compra de ações da Petrobrás. Através da abordagem do tema via Redes Neurais, serão aproveitadas as mesmas fontes de informações utilizadas no modelo de apreçamento Black-Scholes-Merton. Exploraremos também as configurações do modelo, bem como as características quantitativas e critérios utilizados para segregação do conjunto de treino e do conjunto de teste da rede.

3 Metodologia

A construção do modelo proposto no presente estudo possui dois grandes campos de análise: (i) a utilização de metodologia de apreçamento de opções via modelo de Black-Scholes-Merton e (ii) a aplicação desse *framework* em uma segunda base metodológica, que é a Rede Neural Artificial (“Rede Neural” ou “ANN”). Será apresentada, nos subtópicos seguintes, a abordagem teórica do modelo Black-Scholes-Merton, seu desdobramento matemático e a derivação da fórmula final, traçando um paralelo com o uso do modelo no estudo. Em um segundo momento, será apresentada a topologia de utilização da Rede Neural escolhida, as entradas de dados e, sequencialmente, os dados utilizados e suas características.

a. Modelo de apreçamento de opções Black-Scholes-Merton

Em 1973, Black e Scholes (1973) propuseram um modelo que calculava o preço justo para opções de instrumentos financeiros. Desde os anos de 1970, estudiosos financeiros e agentes do mercado utilizam o modelo dos autores como referência de preço justo dos ativos derivativos por conta de sua praticidade e fácil implementação computacional, mas o modelo original proposto recebe críticas pela sua simplificação e por não representar, com certa acuracidade, as dinâmicas de mercado, sendo utilizado como referencial teórico e instrumento balizador de estratégias no mercado financeiro.

Segundo Hull (2016), Black e Scholes partiram do modelo de CAPM (*Capital Asset Pricing Model*) para estabelecer uma relação entre o que o mercado exige como preço da opção e o retorno exigido, sob a perspectiva de custo de oportunidade, do ativo subjacente. Posteriormente, em 1976, Merton (1976) complementou a visão de Black e Scholes a partir do CAPM, introduzindo a comparação com um portfólio livre de risco e afirmando que, por um curto período, a opção e o ativo subjacente deveriam produzir retornos livres de risco.

Hull (2016) aponta que, uma das formas de derivar a fórmula de Black-Scholes-Merton é através da valoração, via árvore binomial, de uma opção europeia de uma

ação que não realiza pagamento de dividendos, em que o número de etapas na árvore tende ao infinito. Supondo que essa valoração possua preço de exercício K e tempo T , cada etapa do processo teria comprimento T/n e, se tivermos j movimentos para cima e $n - j$ movimentos para baixo na árvore binomial, o preço final da ação seria:

$$S_0 u^j d^{n-j} \quad (1)$$

Em que:

- u é o proporcional de movimentos para cima;
- d é o proporcional de movimentos para baixo;
- S_0 é o preço da ação.

Portanto, o *payoff*⁹ de uma operação realizada com uma opção de compra (*call option*) europeia será:

$$\max(S_0 u^j d^{n-j} - K, 0) \quad (2)$$

De acordo com as propriedades da distribuição binomial, sabemos que a probabilidade de realizarmos j movimentos para cima e $n - j$ movimentos para baixo na árvore binomial é (Hull, 2016):

$$\frac{n!}{(n-j)!j!} p^j (1-p)^{n-j} \quad (3)$$

⁹ Medida de ganho ou perda com a operação.

Podemos agora estipular o *payoff* esperado da operação:

$$\sum_{j=0}^n \max(S_0 u^j d^{n-j} - K, 0) \frac{n!}{(n-j)!j!} p^j (1-p)^{n-j} \quad (4)$$

De acordo com os movimentos da árvore binomial, representando a topologia de um ambiente neutro ao risco, podemos descontar o *payoff* esperado à taxa livre de risco r , assim obtemos o preço da opção de compra:

$$c = e^{-rT} \sum_{j=0}^n \max(S_0 u^j d^{n-j} - K, 0) \frac{n!}{(n-j)!j!} p^j (1-p)^{n-j} \quad (5)$$

Como estamos derivando uma opção de compra, o preço da opção é não zero quando o preço do ativo subjacente é maior do que o preço de exercício, ou seja:

$$S_0 u^j d^{n-j} > K \quad (6.1)$$

ou

$$\ln\left(\frac{S_0}{K}\right) > -j \ln(u) - (n-j) \ln(d) \quad (6.2)$$

Como temos $u = e^{\sigma\sqrt{T/n}}$ e $d = e^{-\sigma\sqrt{T/n}}$, a equação 6.2 fica:

$$\ln\left(\frac{S_0}{K}\right) > n\sigma\sqrt{T/n} - 2j\sigma\sqrt{T/n} \quad (7.1)$$

ou

$$j > \frac{n}{2} - \frac{\ln\left(\frac{S_0}{K}\right)}{2\sigma\sqrt{\frac{T}{n}}} \quad (7.2)$$

Com essas aberturas, podemos reescrever a equação 5:

$$c = e^{-rT} \sum_{j>\alpha} \frac{n!}{(n-j)!j!} p^j (1-p)^{n-j} \max(S_0 u^j d^{n-j} - K, 0) \quad (8)$$

Considerando a equação 7.2, temos que:

$$\alpha = \frac{n}{2} - \frac{\ln\left(\frac{S_0}{K}\right)}{2\sigma\sqrt{\frac{T}{n}}} \quad (9)$$

Definições:

$$U_1 = \sum_{j>\alpha} \frac{n!}{(n-j)!j!} p^j (1-p)^{n-j} u^j d^{n-j} \quad (10.1)$$

$$U_2 = \sum_{j>\alpha} \frac{n!}{(n-j)!j!} p^j (1-p)^{n-j} \quad (10.2)$$

Substituindo as equações 10.1 e 10.2 na equação 8, temos que:

$$c = e^{-rT} (S_0 U_1 - K U_2) \quad (11)$$

Considerações sobre a equação 10.2: Hull (2016) aponta que, admitindo que a árvore binomial possui uma distribuição normal conforme o número de tentativas tende ao infinito, quando existem n tentativas e a probabilidade de sucesso é p , a distribuição se aproxima de uma normal quando possui média np e desvio padrão $\sqrt{np(1-p)}$, portanto, a equação 10.2 representa a probabilidade do número de sucessos ser maior que α , ou seja, maior que $\frac{n}{2} - \frac{\ln\left(\frac{S_0}{K}\right)}{2\sigma\sqrt{\frac{T}{n}}}$.

Portanto, podemos reescrever a equação 10.2 da seguinte maneira:

$$U_2 = N\left(\frac{np - \alpha}{\sqrt{np(1-p)}}\right) \quad (12)$$

Em que N é a função de distribuição acumulada para uma variável normal padrão. Substituindo a equação 9 na equação 12, temos:

$$U_2 = N\left(\frac{\ln\left(\frac{S_0}{K}\right)}{2\sigma\sqrt{T}\sqrt{p(1-p)}} + \frac{\sqrt{n}\left(p - \frac{1}{2}\right)}{\sqrt{p(1-p)}}\right) \quad (13)$$

Ao expandirmos p em função da taxa livre de risco e do tempo transcorrido, temos que:

$$p = \frac{e^{rT/n} - e^{-\sigma\sqrt{T/n}}}{e^{\sigma\sqrt{T/n}} - e^{-\sigma\sqrt{T/n}}} \quad (14)$$

Como n tende ao infinito, $\sqrt{n}\left(p - \frac{1}{2}\right)$ tende a $\frac{(r - \frac{\sigma^2}{2})\sqrt{T}}{2\sigma}$. Portanto:

$$U_2 = N\left(\frac{\ln\left(\frac{S_0}{K}\right) + \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}}\right) \quad (15)$$

Rearranjando a equação 10.1, temos:

$$U_1 = \sum_{j>\alpha} \frac{n!}{(n-j)!j!} (pu)^j [(1-p)d]^{n-j} \quad (16)$$

Definição:

$$p^* = \frac{pu}{pu+(1-p)d} \quad (17.1)$$

e

$$1 - p^* = \frac{(1-p)d}{pu+(1-p)d} \quad (17.2)$$

Partindo das definições de p^* e $1 - p^*$, podemos reescrever a equação 16 como:

$$U_1 = [pu + (1-p)d]^n \sum_{j>\alpha} \frac{n!}{(n-j)!j!} (p^*)^j (1 - p^*)^{n-j} \quad (18)$$

Considerando a premissa adotada por Merton (1976), de que o retorno esperado para o ativo subjacente em um cenário neutro ao risco é a própria taxa livre de risco r , temos que:

$$pu + (1-p)d = e^{\frac{rT}{n}} \quad (19)$$

Portanto:

$$U_1 = e^{rT} \sum_{j>\alpha} \frac{n!}{(n-j)!j!} (p^*)^j (1-p^*)^{n-j} \quad (20)$$

Considerando que U_1 assume uma distribuição binomial em que a probabilidade de movimentos para cima (valorização da opção) é p^* ao invés de p , ao aproximarmos uma distribuição binomial de uma distribuição normal, temos:

$$U_1 = e^{rT} N\left(\frac{np^* - \alpha}{\sqrt{np^*(1-p^*)}}\right) \quad (21)$$

Substituindo α pela equação 9, temos:

$$U_1 = e^{rT} N\left(\frac{\ln\left(\frac{S_0}{K}\right)}{2\sigma\sqrt{T}\sqrt{p^*(1-p^*)}} + \frac{\sqrt{n}\left(p^* - \frac{1}{2}\right)}{\sqrt{p^*(1-p^*)}}\right) \quad (22)$$

Substituindo u e d na equação 17.1, temos:

$$p^* = \left(\frac{e^{\frac{rT}{n}} - e^{-\sigma\sqrt{\frac{T}{n}}}}{e^{\sigma\sqrt{\frac{T}{n}}} - e^{-\sigma\sqrt{\frac{T}{n}}}}\right) \left(\frac{e^{\sigma\sqrt{\frac{T}{n}}}}{e^{\frac{rT}{n}}}\right) \quad (23)$$

Como n tende ao infinito, temos que $\sqrt{n} \left(p^* - \frac{1}{2} \right)$ tende a:

$$\frac{\left(r + \frac{\sigma^2}{2}\right)\sqrt{T}}{2\sigma} \quad (24)$$

Substituindo na equação 22, temos que:

$$U_1 = e^{rT} N \left(\frac{\ln\left(\frac{S_0}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \right) \quad (25)$$

Finalizando, partindo das equações 11, 15 e 25, temos que a fórmula de Black-Scholes-Merton para apreçamento de uma opção de compra europeia é dada por:

$$c = S_0 N(d_1) - K e^{-rT} N(d_2) \quad (26)$$

Detalhando d_1 e d_2 , temos que:

$$d_1 = \frac{\ln\left(\frac{S_0}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \quad (27)$$

e

$$d_2 = \frac{\ln\left(\frac{S_0}{K}\right) + \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} = d_1 - \sigma\sqrt{T} \quad (28)$$

Em que:

- S_0 → Preço do ativo subjacente;
- K → Preço de exercício da opção;
- r → Taxa Livre de Risco;
- σ^2 → Variância dos retornos contínuos do ativo subjacente;
- T → Tempo decorrido entre o apreçamento e o vencimento do contrato derivativo.

O modelo padrão de Black-Scholes-Merton possui algumas limitações em relação à aderência com a dinâmica real do mercado de opções, apontadas por Leuthold, Junkus e Cordier (1989):

- Não assume custos transacionais;
- Assume que os preços dos ativos subjacentes seguem um movimento browniano geométrico, ou seja, não consideram movimentos de tendência de preços;
- Assume retornos do ativo livre de risco e volatilidade constantes ao longo do tempo;
- Não leva em consideração os pagamentos de dividendos das ações.

Apesar das restrições quanto à utilização do modelo, os agentes financeiros continuam a empregar essa abordagem por conta de sua influência nas estratégias de delta *hedge* no mercado financeiro e, em decorrência disso, por conta da influência do modelo na calibração dos preços de mercado, tornando o modelo de Black-Scholes-Merton, apesar das décadas que já se passaram desde sua concepção, ainda bem contemporâneo.

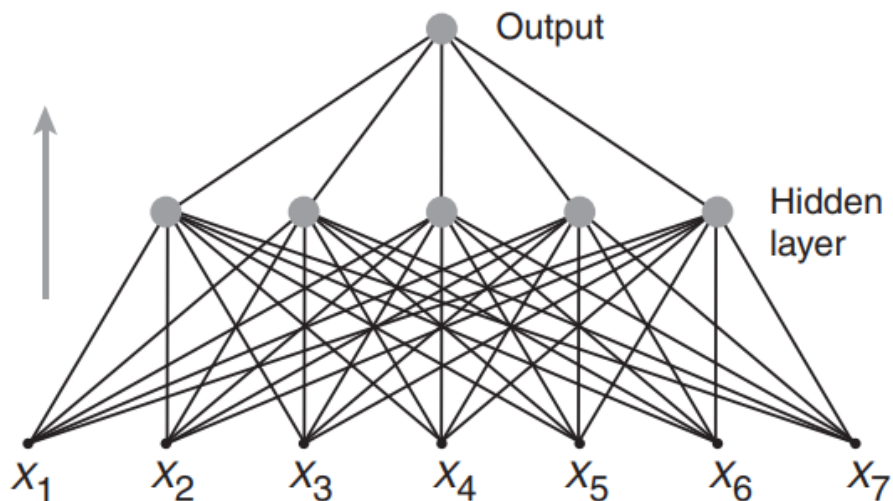
b. Rede Neural Artificial

A presente seção é baseada no trabalho de Krogh (2008). O conceito de Rede Neural Artificial (*Artificial Neural Network*, em inglês, ou “ANN”) tem como inspiração

a biologia e na forma como um cérebro biológico processa as informações recebidas e as transmite, em formato de pulsos elétricos, para os demais órgãos do corpo (Krogh, 2008). Essas tarefas intelectuais e o constante aprendizado (é necessário primeiro aprender uma tarefa, aperfeiçoá-la, e depois utilizá-la no dia a dia) traçam o paralelo entre o *modus operandi* cognitivo e o processo de aprendizagem de máquinas, do qual as ANNs fazem parte.

As unidades básicas de processamento de uma ANN são os neurônios, que de maneira similar aos neurônios biológicos, exercem a função de filtrar e passar a informação adiante. Através da compilação de neurônios, é formada a rede: um emaranhado de neurônios artificiais que exercem a função de receber informações e repassar ao neurônio seguinte, e cada neurônio carrega consigo um fator de ajuste, chamado de peso sináptico, que é multiplicado então pela entrada do neurônio e exerce a função de calibrador da ANN.

Figura 1 – Diagrama simplificado das camadas de uma Rede Neural Artificial.



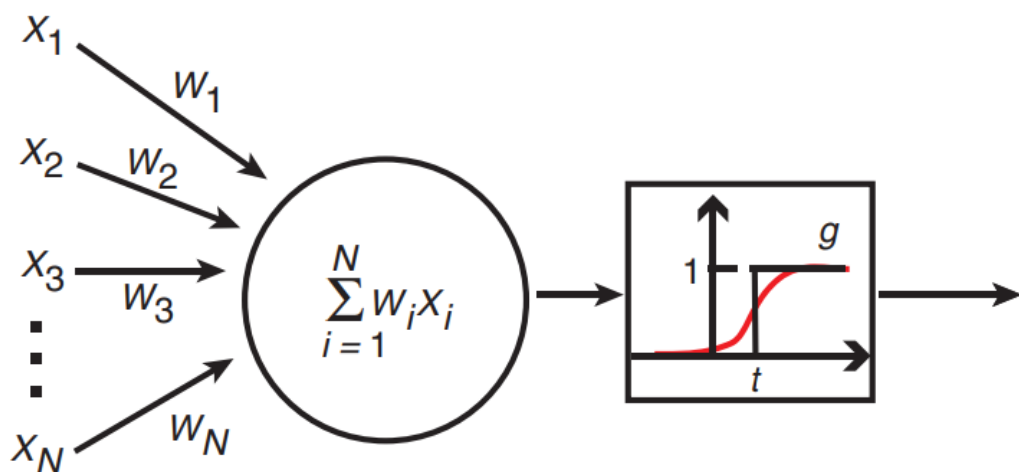
Fonte: Krogh (2008).

Na figura 1 temos a demonstração do diagrama (simplificado) de uma Rede Neural. A seta cinza à esquerda representa a direção da informação, que flui desde a entrada dos dados (base do diagrama) até a camada de saída da ANN, refletindo a informação processada (Krogh, 2008).

As ANNs então são construídas e distribuídas em camadas, cada camada recebe informações da cama anterior, processa, e repassa para a camada seguinte, através de uma função algébrica chamada de função de ativação, e existem funções de ativação padrão para as ANNs (ex.: Linear, Sigmóide, ReLu, etc.). A depender do tipo de dado, do tipo de Rede Neural construída e do objetivo, recomenda-se a utilização de uma função específica. Algumas bibliotecas de linguagem computacional permitem o desenvolvimento de funções de ativação personalizadas, como é o caso das bibliotecas Pytorch e Keras.

Antes das informações passarem pela função de ativação, são atribuídos os pesos sinápticos, que desempenham o papel de calibração da ANN. Os pesos sinápticos servem como calibradores da entrada das informações na rede, contribuindo para que ela aprenda com os erros gerados no processo de treino e adapte os resultados calibrando esses pesos.

Figura 2 – Diagrama representando a entrada das informações (x_n) multiplicadas pelos pesos sinápticos (w_n) e a subsequente aplicação de uma função de ativação.



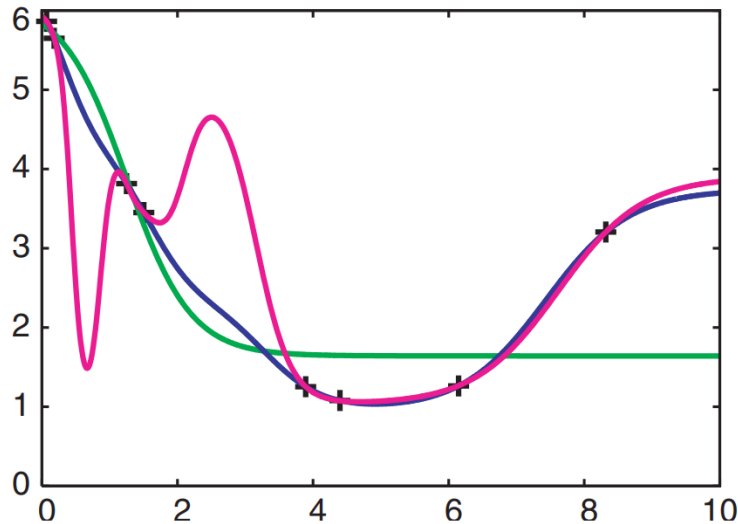
Fonte: Krogh (2008).

Na Figura 2 temos a representação de um diagrama em que podemos ver a demonstração conceitual completa de uma Rede Neural Artificial. As entradas de informação (X_n) são multiplicadas pelos pesos sinápticos (W_n) e somadas. O resultado dessa soma entra como informação na função de ativação e, no processo de treino da Rede Neural, é comparada ao resultado real. De acordo com o erro gerado entre as saídas da rede e o resultado real, os pesos sinápticos são alterados para gerarem erros cada vez menores e, por sua vez, melhorarem a ANN criada.

A ANN é desenvolvida em camadas ocultas (*hidden layers*) que utilizam as informações geradas pela camada anterior e processam a informação. São denominadas camadas ocultas porque não carregam as informações originais fornecidas à rede: elas são camadas processadoras de informação, cujo objetivo é prover à camada de saída (*output*) a informação mais próxima possível do dado real fornecido. O número de camadas ocultas varia de modelo para modelo, e existem bibliotecas de Inteligência Artificial que auxiliam na escolha do número de camadas ocultas. Já as informações fornecidas à rede estão muito mais ligadas à teoria por trás do objeto de análise do que da matemática envolvida nas Redes Neurais Artificiais.

Há ainda o problema da introdução de muitas informações na rede neural (*overfitting*), o que pode gerar um problema de estimação. A ANN tentará encontrar uma relação entre as informações utilizando a função de ativação escolhida, o que muitas vezes, matematicamente, pode ser viável, mas o conjunto solução pode não ter aderência à teoria, tornando a rede inutilizada.

Figura 3 – Representação de três redes neurais distintas e o problema de *over-fitting*.



Fonte: Krogh (2008).

Na Figura 3 podemos ver o esquema gráfico presente em Krogh (2008) da representação de três redes neurais. Existem oito pontos (representados por cruzes) que as redes neurais foram treinadas para perseguir, e foram treinadas fornecendo apenas um parâmetro de entrada de informação.

A curva verde representa a Rede Neural com apenas um neurônio na camada oculta, já a curva azul representa uma Rede Neural com 10 neurônios na camada oculta e a curva rosa representa uma Rede Neural com 20 neurônios na camada oculta. A Rede Neural representada pela curva verde não fez um bom trabalho ao aproximar a função dos pontos fornecidos, o que ajuda a explicar isso é a falta de capacidade processual da rede (ao optarmos por apenas um neurônio na camada oculta, limitamos as transformações das variáveis e a capacidade de alterar os pesos sinápticos).

A rede neural representada pela curva azul fez um bom trabalho ao aproximar a função, já a Rede Neural representada pela curva rosa, apesar de passar pelos pontos fornecidos, causa algumas distorções nos resultados que, em condições normais, não são desejados, evidenciando o problema de *over-fitting*.

As informações fornecidas nos parâmetros de entrada do modelo são então divididas em dois grupos: conjunto treino e conjunto teste. O conjunto treino exerce a função de aprendizado da Rede Neural, ele será responsável pela calibragem dos pesos sinápticos e pela aderência dos resultados obtidos na camada de saída da ANN com os resultados reais: esse processo de iteração ocorre até que os erros quadráticos produzidos entre os resultados obtidos pela Rede Neural e os resultados reais sejam minimizados.

Já o conjunto teste tem a função de estimar, com base nas informações aprendidas pelo conjunto treino e na calibração dos pesos sinápticos, a configuração otimizada dos parâmetros de entrada - em conjunto com as camadas ocultas e os pesos sinápticos - que fornece as saídas de dados mais próximas do que seria uma saída de dado original do modelo, ou seja: primeiro a Rede Neural aprende como aquele modelo se comporta no mundo real com base nas informações fornecidas no treino e, após esse aprendizado ser otimizado, a ANN tenta reproduzir seu comportamento, gerando novos dados de saída.

c. Dados do modelo

A Rede Neural segue a teoria apresentada nas seções de Revisão Bibliográfica e Metodologia, utilizando as variáveis do modelo de apreçamento de opções Black-Scholes-Merton. Em seguida, as variáveis foram utilizadas como parâmetros de entrada da ANN construída e então as informações produzidas pela ANN são comparadas com os preços de mercado das opções de compra da ação PETR4 entre os meses de agosto e outubro de 2020. Ao todo, a amostra apresentada à Rede Neural possui 4.085 observações, passando por um filtro de volume de transações realizadas no dia. A amostragem foi dividida em 4 quartis e as observações que estavam posicionadas no primeiro quartil (baixo nível de volume transacionado) foram descartadas, evitando assim possíveis distorções nos resultados obtidos por falta de liquidez. Abaixo, o detalhamento das variáveis de entrada do modelo:

$PETR4_{Fech.}$: Preço de fechamento ajustado¹⁰ das ações preferenciais da Petróleo Brasileiro S.A. As cotações foram compiladas entre as datas de 28/08/2021 e 28/10/2021. As cotações foram obtidas através da base de dados históricos da B3¹¹.

$Strike$: Cotação de fechamento do preço de exercício negociado para a opções de compra das ações preferenciais da Petróleo Brasileiro S.A. As cotações foram compiladas entre as datas de 28/08/2021 e 28/10/2021. Os dados foram obtidos através da base de dados históricos da B3.

Rf : Taxa Livre de Risco do mercado brasileiro. Adotou-se a taxa básica de juros do mercado brasileiro (Sistema Especial de Liquidação e Custódia - Selic) como taxa livre de risco. As cotações foram compiladas entre as datas de 28/08/2021 e 28/10/2021 e obtidas através da base de dados históricos do Banco Central do Brasil.

TtM_{anos} : Tempo até o vencimento da opção, calculado em termos anuais (frações de um ano). Calcula-se o tempo, em dias úteis, entre a data da cotação do preço da opção e seu vencimento, o resultado é então dividido por 252 para se obter o tempo até o vencimento em termos anuais.

$\sigma_{60 dias}$: Desvio padrão dos retornos diários anualizados para 60 pregões de PETR4. O desvio padrão é calculado considerando-se 60 dias móveis.

10 Considera ajustes nos preços por dividendos, recompras e outros mecanismos de ajuste de cotação.

11 Disponível em https://www.b3.com.br/pt_br/market-data-e-indices/servicos-de-dados/market-data/historico/mercado-a-vista/cotacoes-historicas.

d. Modelo de Rede Neural

A Rede Neural foi construída considerando cinco parâmetros de entrada (dados da subseção anterior) e 20 neurônios¹² dispostos em uma única camada oculta, trabalhada pela função de ativação Unidade Linear Retificada (ReLU). Essa função tem como característica apresentar resultados no intervalo $[0, \infty[$, retornando zero para resultados negativos e o próprio valor do parâmetro quando positivo. A escolha pela função de ativação ReLU foi feita por ser uma função de ativação versátil, com rápido treinamento e necessidade de capacidade computacional considerada leve (Schmidt-Hieber, 2020).

No apêndice A podemos ver a representação do diagrama, simplificado, da Rede Neural utilizada no presente estudo. A topologia da ANN utilizada é composta por (i) cinco parâmetros de entrada e seus respectivos pesos sinápticos, (ii) a representação do processo matemático da somatória do produto dos pesos sinápticos pelos parâmetros de entrada, (iii) a função de ativação utilizada na Rede Neural (ReLU), (iv) a camada oculta com 20 neurônios, (v) o mesmo processo utilizado na entrada da camada oculta (somatória do produtos dos pesos sinápticos pelos neurônios da camada oculta e função de ativação) e (vi) a camada de saída da Rede Neural com a informação desejada.

Da amostra de dados de entrada do modelo, 80% das observações foram separadas para o conjunto de treino da Rede Neural e 20% dos dados que compõe os parâmetros de entrada foram destinados ao conjunto teste. A ideia de fornecer grande parte dos dados disponíveis para o conjunto de treinamento é justamente entregar à Rede Neural uma variedade de situações representativas do conjunto amostral, o que aumenta a capacidade de calibração dos parâmetros da ANN.

¹² Foram também performados testes empíricos com configurações diferentes dos 20 neurônios apresentados e o modelo que apresentou os melhores resultados em termos de redução do erro foi o que considera 20 neurônios na camada oculta.

e. Erro Quadrático Médio (EQM)

Após realizada a configuração da ANN, a calibração dos parâmetros, o treinamento e o teste, a camada de saída do modelo apresentado (*output*) expõe os dados obtidos. Os resultados são então submetidos ao cálculo do Erro Quadrático Médio (um EQM para o conjunto de treino e um para o conjunto teste), que considera a diferença entre o apreçamento da opção feita pela Rede Neural (\hat{Y}_i) e a cotação de fechamento de mercado para aquela opção (Y_i). Essa diferença é então elevada ao quadrado e somada para todas as observações, dividindo-se o resultado da somatória pelo número total de observações N :

$$EQM = \frac{1}{N} \sum_{i=1}^N (Y_i - \hat{Y}_i)^2 \quad (29)$$

O código implementado também realiza o cálculo do apreçamento das opções de PETR4 via modelo padrão de Black-Scholes-Merton descrito na seção 3.a, através da equação 28. Com o resultado obtido via modelo padrão, encontra-se o EQM para o conjunto amostral pertencente ao conjunto de teste (20% da amostra) e os resultados são comparados, através da raiz quadrada do Erro Quadrático Médio para cada conjunto amostral, a fim de testar empiricamente se a Rede Neural conseguiu reduzir o erro da amostra em comparação com o cálculo padrão de Black-Scholes-Merton.

A raiz quadrada do Erro Quadrático Médio é utilizada por fornecer a unidade de medida em BRL, ou seja, o resultado está diretamente relacionado ao valor na moeda local brasileira. Se a comparação fosse realizada diretamente via Erro Quadrático Médio, a unidade de medida seria em Reais ao quadrado, o que dificulta a comparação. Portanto, a equação 29 sofre uma alteração para se chegar na tabela comparativa final:

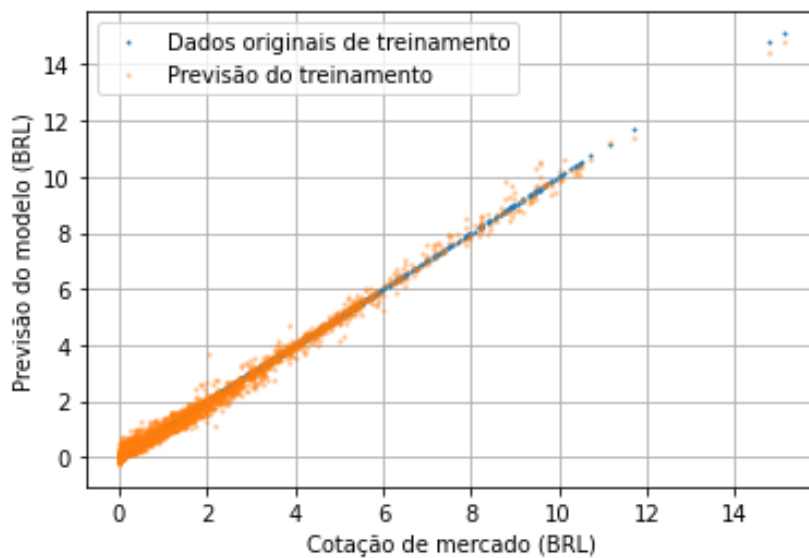
$$EQM_{\text{raiz quadrada}} = \sqrt{\frac{1}{N} \sum_{i=1}^N (Y_i - \hat{Y}_i)^2} \quad (30)$$

Note que a equação 30 possui a mesma estrutura apresentada na equação 29, com a diferença de que extraímos a raiz quadrada do Erro Quadrático Médio.

4 Resultados do estudo

Após a calibração dos pesos sinápticos através do treinamento, podemos organizar os dados e comparar os resultados:

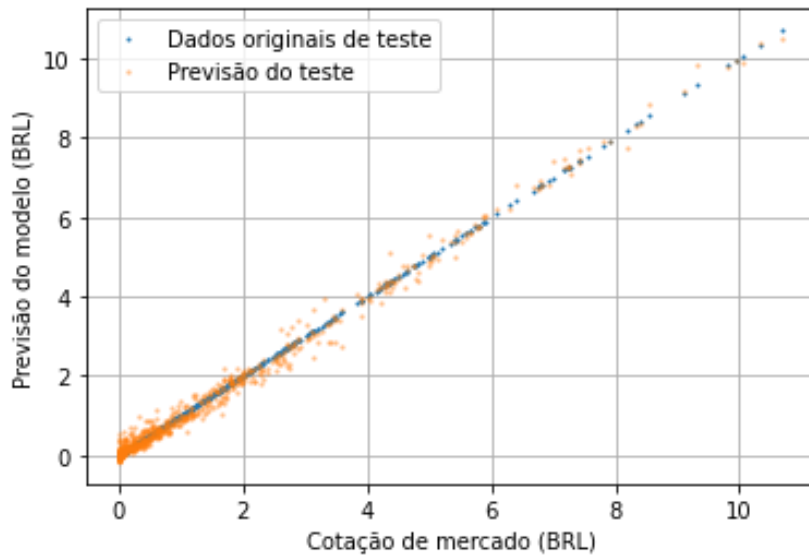
Figura 5 – Gráfico de dispersão de pontos do Conjunto de Treinamento



Fonte: Elaboração própria.

Na figura 5 pode notar que foi inserida, para cada linha de observação do conjunto de treinamento, a comparação entre os dados de saída da ANN e a cotação de mercado das respectivas observações. Podemos notar, visualmente, que boa parte dos dados originais (cotação de mercado) estão abaixo dos BRL 6,00, possuindo uma concentração representativa entre BRL 0,00 e BRL 2,00.

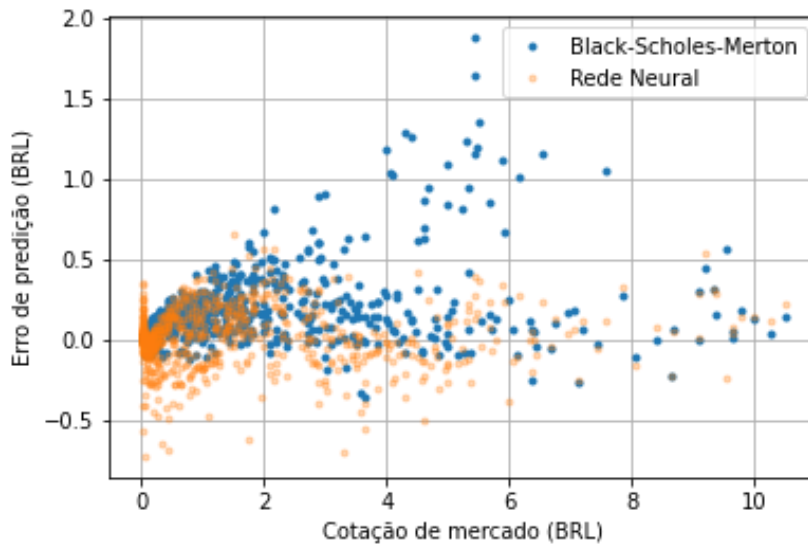
Figura 6 – Gráfico de dispersão de pontos do Conjunto de Teste



Fonte: *Elaboração própria.*

Na Figura 6 podemos ver o gráfico que representa a comparação dos dados originais para o conjunto de teste em contraste com os dados gerados pela ANN. Podemos notar, através da análise visual do gráfico, que boa parte do conjunto amostral se concentra nas cotações de opções de compra de PETR4 de até BRL 4,00, além de possuir, visualmente, as mesmas características presentes no gráfico da Figura 5, em que se analisa o Conjunto Treinamento.

Figura 7 – Gráfico de dispersão de pontos Black-Scholes-Merton VS. Rede Neural



Fonte: Elaboração própria.

No gráfico plotado na Figura 7, podemos ver a comparação entre o erro de predição gerado pela Rede Neural e o gerado através do cálculo padrão de Black-Scholes-Merton, ambos colocados sob a perspectiva da cotação de mercado. O erro de predição é baseado na equação 29, ou seja, $(Y_i - \hat{Y}_i)$ para cada observação. É possível notar, visualmente, que a nuvem de pontos gerada pelos resultados via Black-Scholes-Merton possui um erro de predição, em termos absolutos, mais amplo (como é possível observar empiricamente na tabela 1 abaixo). Já a Rede Neural, para a mesma observação, apresenta erros de predição mais próximos de zero.

Após o levantamento empírico dos erros de predição para cada uma das observações, podemos calcular a raiz quadrada do Erro Quadrático Médio do conjunto de treinamento, do conjunto de teste e do modelo padrão Black-Scholes-Merton:

Tabela 1 – Tabela comparativa da Raiz Quadrada do EQM

Conjunto Amostral	Raiz Quadrada do EQM (Termos absolutos)
Conjunto Treinamento (ANN)	BRL 0,154765
Conjunto Teste (ANN)	BRL 0,169822
Black-Scholes-Merton (Conjunto Teste)	BRL 0,292314

Fonte: Elaboração própria.

A Tabela 1 apresenta os resultados do estudo. Podemos ver o resultado da raiz quadrada do EQM para cada conjunto de amostras. Os resultados são obtidos a partir da fórmula presente na equação 30, e apresentados em termos absolutos, ou seja, em BRL.

Apesar do conjunto de treinamento apresentar uma raiz quadrada do EQM superior ao apresentado pelo conjunto teste, o objetivo do presente estudo se concentra na comparação quantitativa da raiz quadrada do EQM entre o Conjunto Teste (Rede Neural) e o erro gerado, utilizando a mesma amostra do Conjunto Teste, via modelo de Black-Scholes-Merton. Quando colocamos esses dois resultados sob perspectiva, notamos que o erro gerado via ANN é menor do que o erro gerado via Black-Scholes-Merton diretamente.

Enquanto a Rede Neural gera um erro de aproximadamente BRL 0,17, o modelo Black-Scholes-Merton gera um erro de aproximadamente BRL 0,29, ou seja, para o conjunto amostral proposto (Conjunto Teste), existem evidências empíricas que a Rede Neural consegue melhorar o erro em aproximadamente BRL 0,12, sugerindo que o apreçamento de opções via Rede Neurais, que utiliza os parâmetros de entrada derivados do modelo padrão Black-Scholes-Merton, é mais eficiente do que o cálculo realizado diretamente via Black-Scholes-Merton.

5 Conclusão do estudo

O avanço do poder computacional nas últimas décadas do século XX possibilitou à humanidade a criação de máquinas e mecanismos artificiais dotados da capacidade de aperfeiçoamento e aprendizagem de informações, criando um campo da ciência que passou a ter interação com as diversas áreas do conhecimento: seja para detectar, através de inteligência artificial, vestígios de doenças que podem se manifestar no futuro, ou no mercado financeiro, ajudando acadêmicos e agentes do mercado no apreçamento de ativos e até mesmo na tentativa de predição deles.

O presente estudo procurou responder se existe uma forma, utilizando como base de informações os dados do modelo de Black-Scholes-Merton, de realizar o apreçamento de opções de compra de PETR4 de modo alternativo, tentando também diminuir o viés presente no modelo padrão, reduzindo a raiz do Erro Quadrático Médio, o que tornaria o apreçamento mais preciso e coerente com os preços de mercado.

Os resultados apontam que a Rede Neural desenvolvida foi responsável pela redução da raiz do EQM em comparação ao cálculo realizado via Black-Scholes-Merton, tornando o modelo desenvolvido mais aderente às cotações de mercado para a amostra utilizada. Para futuros trabalhos, recomenda-se realizar o cálculo de Erro Quadrático Médio percentual, ou seja, comparando a média percentual do erro entre a cotação de mercado e o resultado da Rede Neural em contraste via Black-Scholes-Merton, o que pode fornecer uma perspectiva de escala ao erro encontrado no modelo.

Outra recomendação para futuras análises é segregar, na base de dados, as opções de compra de PETR4 que são do tipo europeia e americana. Atualmente, a base de dados fornecida pela B3 não fornece esse filtro, o que impossibilita a segregação da amostra entre os dois tipos de opções.

Referências

- Bartram, S. M., Branke, J. & Motahari, M. (2020). *Artificial intelligence in asset management*. CFA Institute Research Foundation. Recuperado de <https://rflr-artificial-intelligence-in-asset-management.ashx> (cfainstitute.org).
- Black, F., Scholes, M. (1973). *The Pricing of Options and Corporate Liabilities*. *Journal of Political Economy*, University of Chicago Press, 81(3), 637-654. Recuperado de https://www.cs.princeton.edu/courses/archive/fall09/cos323/papers/black_scholes73.pdf.
- Carter, C. A. (2007). *Commodities*. In Carter, C., A. (Waveland Press). *Futures and options markets: An Introduction*. California: Waveland Press.
- Cox, J. C. (1979). *Option pricing: A simplified approach*. *Journal of financial economics*, (7), 229-263. Recuperado de <https://www.financeinvest.at/wp-content/uploads/2020/08/Cox-Ross-Rubinstein-1979.pdf>.
- Gong, H., Thavaneswaran, A. & Singh, J. (2010). *A Black-Scholes model with garch volatility*. *Math Scientist*, (35), 37-42. Recuperado de https://www.researchgate.net/profile/Aera-Thavaneswaran/publication/265442526_A_Black-Scholes_model_with_GARCH_volatility/links/5be3486c4585150b2ba6c271/A-Black-Scholes-model-with-GARCH-volatility.pdf.
- Horvath, B., Muguruna, A. & Tomas, M. (2019). *Deep Learning volatility: A deep neural network perspective on pricing and calibration in (rough) volatility models*. *Quantitative Finance*, (21), 1-17. Recuperado de <https://arxiv.org/pdf/1901.09647.pdf>.
- Hull, J., C. (2014). *The Black-Scholes-Merton model*. In Hull, J., C. (Person). *Options, futures, and other derivatives* (9^o edição). New Jersey: Person.
- Hull, J., C. (2016). *Valuing stock options: The Black--Scholes--Merton model*. In Hull, J., C. (Person). *Fundamentals of Futures and Options Markets* (9^o edição). New Jersey: Person.
- Krogh, A. (2008). *What are artificial neural networks?*. *Nature Biotechnology*, 26(2), 195-197. Disponível em <https://people.binf.ku.dk/~krogh/publications/pdf/Krogh08.pdf>.
- Leuthold, R., M., Junkus, J., C., Cordier, J., E. (1989). (Stipes Publishing). *The theory and practice of futures markets*. Stipes Publishing.
- Merton, R., C. (1976). *Option pricing when underlying stock returns are discontinuous*. *Journal of Financial Economics*, 3(1-2), 125-144. Disponível em [https://doi.org/10.1016/0304-405X\(76\)90022-2](https://doi.org/10.1016/0304-405X(76)90022-2).
- Mitra, S., K. (2012). *An option pricing model that combines neural network approach and Black-Scholes formula*. *Global journal of computer science and technology*,

12 (4), 6-15. Disponível em <https://computerresearch.org/index.php/computer/article/download/455/455>.

Pimprikar, R., Ramachandran, S. & Senthilkumar, K. (2017). *Use of machine learning algorithms and Twitter sentiment analysis for stock market prediction*. *International Journal of Pure and Applied Mathematics*, 115 (6), 521-526. Disponível em <https://www.acadpubl.eu/jsi/2017-115-6-7/articles/6/69.pdf>.

Schmidt-Hieber, J. (2020). *Nonparametric regression using deep neural networks with ReLU activation function*. *The Annals of Statistics*, 48(4), 1875-1897. Disponível em <https://arxiv.org/pdf/1708.06633.pdf>

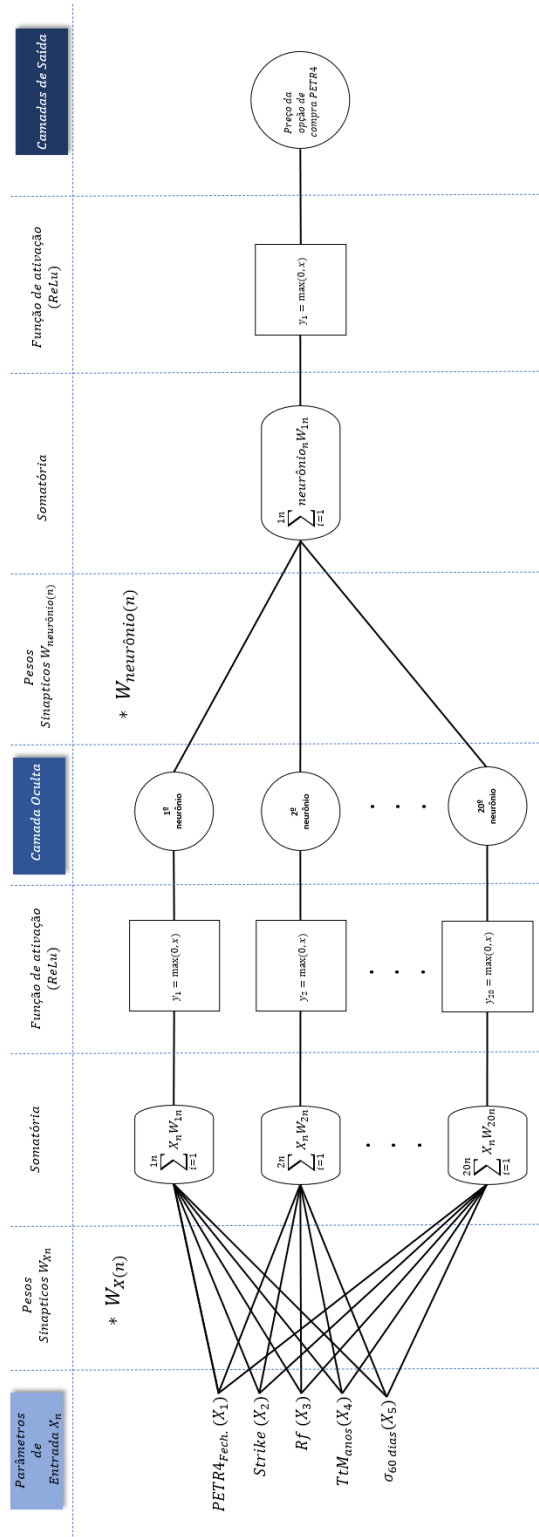
Tseng, C., Cheng, S., Wang, Y. & Peng, J. (2008). *Artificial neural network model of the hybrid EGARCH volatility of the Taiwan stock index option prices*. *Physica A*, 387, 3192-3200. Disponível em https://www.researchgate.net/publication/222364321_Artificial_neural_network_model_of_the_Hybrid_EGARCH_volatility_of_the_Taiwan_stock_index_option_prices.

Vargas, R., Mosavi, A. & Ruiz, L. (2017). *Deep learning: A review*. (Working paper nº 127354). Recuperado de <https://eprints.qut.edu.au/127354/>.

Zhong, X. & Enke, D. (2019). *Predicting the daily return direction of the stock market using hybrid machine learning algorithms*. *Financial Innovation*, 5 (10), 5-24. Recuperado de <https://jfin-swufe.springeropen.com/track/pdf/10.1186/s40854-019-0138-0.pdf>.

Apêndice

Apêndice A – Diagrama simplificado da Rede Neural utilizada



Fonte: Elaboração própria.

Apêndice B – Código da Rede Neural desenvolvido em linguagem Python

▼ Carregando dados do Google Drive

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

```
from google.colab import drive
drive.mount('/content/drive')
```

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.r



```
dados = pd.read_csv('/content/drive/My Drive/Monografia/Inputs_Modelo.csv')
```

▼ Observando os dados

```
dados.head()
```

	Preco_fechamento_ativo_objeto	Strike	Preco_da_opcao	Risk_free_Selic	Time_to
0	22.97	28.53	0.01	0.02	
1	22.97	18.53	4.69	0.02	
2	22.97	18.61	5.03	0.02	
3	22.97	22.61	2.30	0.02	
4	22.97	23.61	1.66	0.02	

```
dados.dtypes
```

```
Preco_fechamento_ativo_objeto    float64
Strike                            float64
Preco_da_opcao                    float64
Risk_free_Selic                   float64
Time_to_maturity_years            float64
Desv.Pad_60_dias_anualizado_(2)  float64
dtype: object
```

▼ Criando variáveis para modelagem

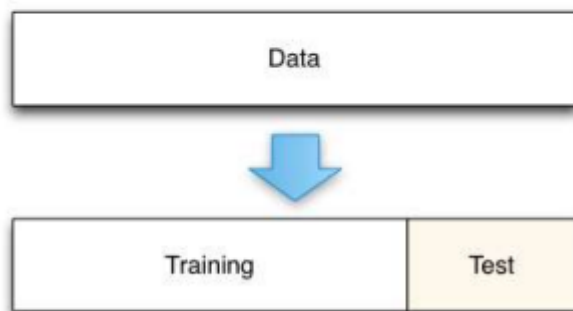
$$y = f(x)$$

```
# cria variáveis X e y
```

```
# X: dados de entrada do modelo
# y: dados de saída do modelo
target = 'Preco_da_opcao'

X = dados.drop(target, axis=1)
y = dados[target]
```

▼ Dividindo os dados em dois conjuntos



```
from sklearn.model_selection import train_test_split
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, train_size=0.8)

# dividindo em dois conjuntos
# * conjunto de treinamento
# * conjunto de teste
# tamanho_total = len(dados)
# tamanho_treino = int(tamanho_total * 0.8)

# 0 conjunto de teste representa 20% da amostragem total

# Xtrain = X[:tamanho_treino]
# ytrain = y[:tamanho_treino]

# Xtest = X[tamanho_treino:]
# ytest = y[tamanho_treino:]

print(Xtrain.shape)
print(ytrain.shape)
print(Xtest.shape)
print(ytest.shape)

(3268, 5)
(3268,)
(817, 5)
(817,)
```

▼ Modelagem da Rede Neural

```
# Carregando a biblioteca
from sklearn.neural_network import MLPRegressor

# Cria a rede neural definindo:
# * o tamanho das camadas ocultas,
# * a função de ativação
# * número máximo de iterações
modelo = MLPRegressor(max_iter=10000, hidden_layer_sizes=(20,10))

# Treina o modelo
modelo.fit(Xtrain, ytrain)

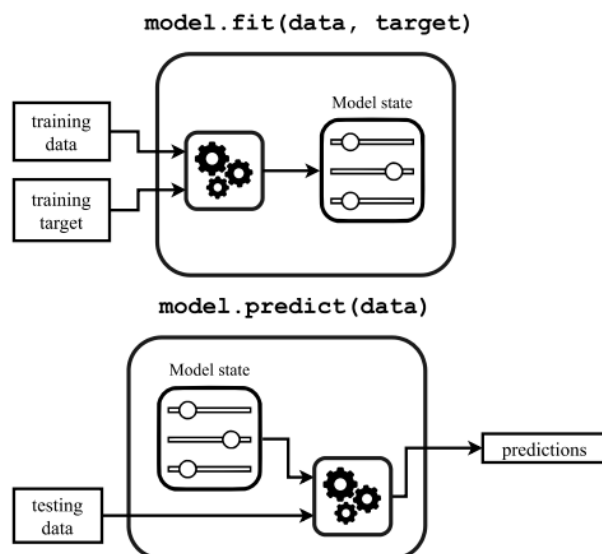
        MLPRegressor(hidden_layer_sizes=(20, 10), max_iter=10000)

# Criação do output da rede neural para o conjunto de treinamento
ypred = modelo.predict(Xtrain)
ytrain_pred = pd.DataFrame(ypred, index = ytrain.index)

# Criação do output da rede neural para o conjunto de teste
ypred = modelo.predict(Xtest)
ytest_pred = pd.DataFrame(ypred, index = ytest.index)
```

▼ Revisão do modelo

1. Criar o modelo
2. Fazer o ajuste (fit) do modelo aos dados
3. Fazer a predição



▾ Implementação do modelo Black-Scholes-Merton

```

import numpy as np
import scipy.stats as si
import sympy as sy
from scipy.stats import norm
from sympy.stats import Normal, cdf
from sympy import init_printing
init_printing()

def bs(S, r, K, sigma, prazo, put=False):
    d1 = (np.log(S/K) + (r + 0.5*sigma**2) * (prazo)) / (sigma * np.sqrt(prazo))
    d2 = d1 - sigma * np.sqrt(prazo)
    y = S * norm.cdf(d1) - np.exp(-r*prazo) * K * norm.cdf(d2)
    if put:
        y = y - S + K*np.exp(-r*prazo)

    return y

#Inserindo o cálculo de Black-Scholes-Merton no arquivo .csv
dados["B&S"]= bs(dados['Preco_fechamento_ativo_objeto'],dados['Risk_free_Selic'],dados['St
dados.head()

```

	Preco_fechamento_ativo_objeto	Strike	Preco_da_opcao	Risk_free_Selic	Time_to
0	22.97	28.53	0.01	0.02	
1	22.97	18.53	4.69	0.02	
2	22.97	18.61	5.03	0.02	
3	22.97	22.61	2.30	0.02	
4	22.97	23.61	1.66	0.02	

```

#Inserindo o cálculo de Black-Scholes-Merton na amostra de base destinada ao teste da rede
Xtest["B&S"]= bs(Xtest['Preco_fechamento_ativo_objeto'],Xtest['Risk_free_Selic'],Xtest['St
Xtest.head()

```

	Preco_fechamento_ativo_objeto	Strike	Risk_free_Selic	Time_to_maturity_ye
3083	20.13	18.20	0.02	0.095
3757	20.57	27.47	0.02	0.468
2609	20.03	35.61	0.02	0.214
2056	20.13	26.53	0.02	0.063
3023	19.80	22.00	0.02	0.746

▼ Cálculo do erro de predição ($Y_i - \hat{Y}_i$) e do Erro Quadrático Médio (EQM)

```
from sklearn.metrics import mean_squared_error

#Cálculo do erro quadrático na base de treino
erro_train= np.sqrt(mean_squared_error(ytrain, ytrain_pred))
print(f'{erro_train:.6f}')

0.154765

#Cálculo do erro quadrático na base de teste
erro_test= np.sqrt(mean_squared_error(ytest, ytest_pred))
print(f'{erro_test:.6f}')

0.169822

#Cálculo do erro quadrático na base de teste via Black Scholes
erro_BS= np.sqrt(mean_squared_error(ytest, Xtest['B&S']))
print(f'{erro_BS:.6f}')

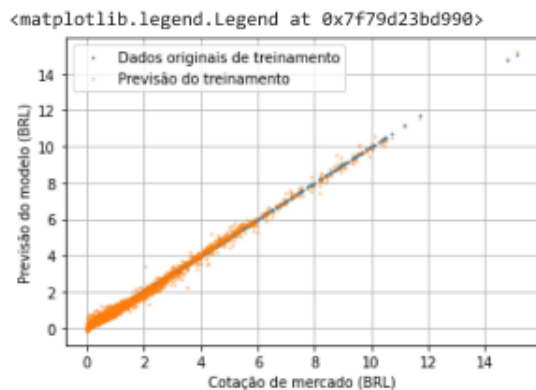
0.292314
```

▼ Gráficos

```
ytrain.shape

(3268,)
```

```
plt.plot(ytrain, ytrain, '.', markersize=2, label='Dados originais de treinamento')
plt.plot(ytrain, ytrain_pred.values.reshape(-1), '.', markersize=3, label='Previsão do tre
plt.xlabel('Cotação de mercado (BRL)')
plt.ylabel('Previsão do modelo (BRL)')
plt.grid(True)
plt.legend()
```



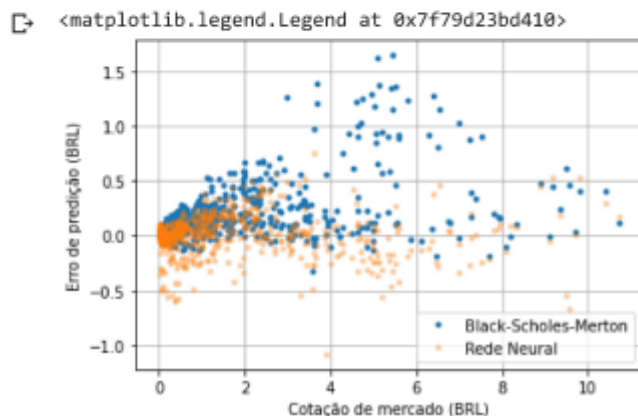
```
ytrain_pred.to_numpy().reshape(len(ytrain_pred))

array([ 1.25266899, -0.00272327,  1.0213003 , ...,  0.15180673,
        0.02950493,  0.02870817])
```

```
plt.plot(ytest, ytest, '.', markersize=2, label='Dados originais de teste')
plt.plot(ytest, ytest_pred.values.reshape(-1), '.', markersize=3, label='Previsão do teste')
plt.xlabel('Cotação de mercado (BRL)')
plt.ylabel('Previsão do modelo (BRL)')
plt.grid(True)
plt.legend()
```



```
plt.plot(ytest, ytest-Xtest['B&S'].values.reshape(-1), '.', label='Black-Scholes-Merton')
plt.plot(ytest, ytest-ytest_pred.to_numpy().reshape(len(ytest_pred)), '.', label='Rede Neural')
plt.ylabel('Erro de predição (BRL)')
plt.xlabel('Cotação de mercado (BRL)')
plt.grid(True)
plt.legend()
```



Parâmetros do modelo

1. activation = 'relu'
2. alpha = 0.0001
3. batch_size = 'auto'
4. beta_1 = 0.9
5. beta_2 = 0.999
6. early_stopping = False
7. epsilon = 1e-08
8. hidden_layer_sizes = (20, 10)
9. learning_rate = 'constant'
10. learning_rate_init = 0.001
11. max_fun = 15000

```
12. max_iter = 200
13. momentum = 0.9
14. n_iter_no_change = 10
15. nesterovs_momentum = True
16. power_t = 0.5
17. random_state = None
18. shuffle = True
19. solver = 'adam'
20. tol = 0.0001
21. validation_fraction = 0.1
22. verbose = False
23. warm_start = False
```