Revista de Informática Teórica e Aplicada - RITA - ISSN 2175-2745
Vol. 27, Num. 2 (2020) 42-65

RESEARCH ARTICLE

Hybrid Neural Networks Applied to Brazilian Stock Market

Redes Neurais Híbridas Aplicadas ao Mercado de Ações Brasileiro

Andrey de Aguiar Salvi[1], William Passig de Souza[2], Wilson Castello Branco Neto[3]*

Abstract: The stock market is a stochastic, dynamic environment and is in constant evolution, and its prediction represents a big challenge. Many studies presented in the state of the art are facing this challenge, by making use of Artificial Neural Networks (ANN) as a tool to make such prediction. In this paper a comparative study is made with different methods in order to predict the Brazilian stock market through the Bovespa Index. An ANN was developed and its performance was compared against a hybrid model, in which a Genetic Algorithm (GA) is proposed as an alternative to improve the performance of this ANN. The results obtained were an average accuracy of 55.04% and 55.73% respectively, demonstrating that algorithms such as a GA have the capability of improving the performance of ANN for the stock market prediciton.
Keywords: Artificial Neural Networks — Genetic Algorithms — Stock Market — IBovespa

Resumo: O mercado de ações é um ambiente estocástico, dinâmico e em constante evolução, revelando que a sua previsibilidade é um grande desafio. Diversos trabalhos existentes no estado da arte encaram este desafio, utilizando como ferramenta para a previsão as Redes Neurais Artificiais (RNA). Este trabalho realiza um estudo comparativo de diferentes métodos com o objetivo de prever o mercado de ações brasileiro, por meio do Índice Bovespa. Uma RNA pura foi desenvolvida e seu desempenho foi comparado com um modelo híbrido, no qual um Algoritmo Genético (AG) é proposto como alternativa para melhorar os resultados desta RNA. Como resultados, foram obtidos Taxas de Acerto médias de 55.04% e 55.73% respectivamente, demonstrando que algoritmos como AG possuem a capacidade de aprimorar o desempenho das RNA na previsibilidade do mercado de ações.
Palavras-Chave: Redes Neurais Artificiais — Algoritmos Genéticos — Mercado de Ações — IBovespa

[1] *Computer Science, IFSC Lages, Brazil*
[2] *Computer Science, IFSC Lages, Brazil*
[3] *Computer Science, IFSC Lages, Brazil*
*__Corresponding author__: wilson.castello@ifsc.edu.br
DOI: http://dx.doi.org/10.22456/2175-2745.88911 • **Received:** 13/12/2018 • **Accepted:** 31/01/2020

1. Introdução

Dentre os problemas que se podem solucionar computacionalmente, existem aqueles que possuem um ambiente de tarefas estocástico, ou seja, um ambiente cujo estado não pode ser completamente determinado pelas percepções do agente [1]. Um exemplo é o mercado de ações que, por ser um sistema dinâmico, complexo e que está sempre evoluindo, apresenta movimentos de natureza altamente irregular. Assim, a previsão de retorno de investimento se dá como um desafio significativo devido à grande diversidade de dados desestruturados com um alto grau de instabilidade e interconexões não aparentes [2]. Além disso, vários fatores, como regulamentações governamentais por exemplo, tornam o mercado de ações um ambiente que não é completamente determinado, uma vez que tais fatores não podem ser medidos quantitativamente.

Existem diversos ramos da inteligência artificial que visam resolver problemas como este, como as Redes Neurais Artificiais (RNA). As RNA são sistemas compostos por vários neurônios, que tem inspiração nos neurônios biológicos e no sistema nervoso [3]. A grande importância destes algoritmos é a capacidade de aprenderem padrões, que muitas vezes são difíceis para o ser humano devido ao grande volume de dados, a partir de um conjunto de dados.

Como apresentado em *surveys* como [4] e [5], existem diversos métodos emergentes com o objetivo de realizar a previsão dos valores do mercado de ações e substituir ou complementar métodos clássicos, sendo as RNA e suas variações um dos métodos principais e mais frequentemente utilizados. Alguns dos outros métodos encontrados são ARIMA, ANFIS, Regressão Linear (RL), Máquinas de Vetores de Suporte (MVS), Modelo Oculto de Markov e Árvores de Decisão.

No trabalho de [6] foi realizado um estudo comparativo sobre o desempenho de métodos para a previsão de preços do Bitcoin com base em dados extraídos sobre a *blockchain*, o protocolo usado pelas criptomoedas. Dentre os métodos

comparados, encontram-se Máquina de Vetor de Suporte, Regressão Logística e RNA. A RNA obteve o melhor desempenho, de $55,1\%$ de acurácia, seguido pela RL ($54,3\%$) e pelo MVS ($53,7\%$). Nestas pesquisas, os resultados dos métodos deixam claro que ainda há muito trabalho para ser desenvolvido para alcançar uma previsão boa o suficiente para que seja possível realizar investimentos com segurança.

Existem outros mecanismos que lidam com indeterminismo capazes de aprimorar o desempenho das RNA por meio de algoritmos híbridos. Um exemplo é a Lógica *Fuzzy* (LF), um método de raciocínio para os Conjuntos Fuzzy, os quais são conjuntos que especificam o quanto um objeto satisfaz uma descrição vaga [1]. Outro exemplo são os Algoritmos Genéticos (AG), que constituem uma técnica de busca inspirada no processo de evolução dos seres vivos e baseada na seleção natural de Darwin [3].

Tendo isto em vista, diversos estudos têm sido feitos com a utilização de Redes Neurais Artificiais, e suas hibridizações, para a previsão de valores do mercado de ações, com o intuito de tomar atitudes com um maior grau de certeza [7, 8, 9].

Em [10], foi desenvolvido um método para classificação de dados utilizando conceitos de RNA com *Deep Learning*. Para isso, os dados de entrada são compartilhados em duas redes iniciais, simultaneamente. A primeira rede, uma rede de LF, calcula o grau de pertinência dos dados a um conjunto *fuzzy*. A segunda é uma RNA que transforma a entrada em uma representação de alto nível. Em seguida, a saída das duas redes é unida em uma outra rede que utiliza conceitos de aprendizado multimodal e, finalmente, uma última rede classifica os resultados obtidos em categorias. Este algoritmo foi testado para prever o Índice Financeiro de Shangai, com os testes sendo repetidos 20 vezes. O método desenvolvido superou os seus concorrentes, com $53,2\%(\pm 1,2\%)$ de acurácia contra os $51,1\%(\pm 1,3\%)$ da Rede Neural Fuzzy Auto-Construída e $42,5\%(\pm 1,7\%)$ da Rede Neural Profunda.

Na obra [11] também são apresentadas maneiras de aprimorar RNA com hibridizações baseadas em AG. O autor identificou quatro categorias de como os AG podem melhorar as RNA, sendo elas a pesquisa por um conjunto de pesos das conexões ótimo, pesquisa sobre o espaço de topologia, pesquisa por parâmetros de aprendizado da rede e finalmente abordagens que modificam o algoritmo de aprendizado padrão. O autor ainda propôs uma nova técnica baseada na busca dos melhores pesos trabalhando em conjunto com o algoritmo de aprendizado sem substituí-lo. A aplicação destas técnicas para a identificação de câncer de mama levaram a uma taxa de acerto médio de 85% com RNA puras e 95.6% com o método híbrido.

Como pode ser observado nesses estudos, a união das RNA com outras técnicas tem sido estudada com o objetivo de resolver problemas diversos, dentre eles a previsão do comportamento do mercado de ações, atingindo resultados razoáveis. Porém, muitos desses estudos são realizados no exterior, o que deixa o Brasil com pouco desenvolvimento nesta área.

O estudo de [12] compara o mercado de ações de três países em desenvolvimento, Brasil, China e Índia, tratando principalmente de ofertas públicas iniciais, ou seja, quando uma empresa oferece suas ações para venda pela primeira vez. Neste trabalho, são comparadas diversas características dos mercados desses países e, embora todos estejam em uma situação similar de desenvolvimento, ainda existem muitas diferenças entre suas economias. Portanto, ainda podem ser explorados a pesquisa e o desenvolvimento de técnicas de inteligência artificial para prever o mercado de ações brasileiro. Deve-se considerar que os métodos que podem funcionar melhor em alguns mercados internacionais podem ser incompatíveis com o mercado brasileiro, devido às peculiaridades da economia nacional.

O objetivo deste trabalho foi realizar um estudo sobre algoritmos de RNA híbridas, visando especificamente a previsão do mercado de ações brasileiro, por meio do Índice Bovespa. Para que o objetivo geral fosse alcançado, foram definidos os seguintes objetivos específicos:

- identificar métodos existentes para a previsão dos valores do mercado de ações, utilizando RNA puras e híbridas;
- implementar os métodos escolhidos;
- obter uma base de dados e treinar as RNA;
- comparar os resultados e encontrar o método que melhor se adequou ao mercado de ações brasileiro.

O restante desse artigo está organizado da seguinte maneira: a seção 2 apresenta a base teórica necessária para a realização deste trabalho; a seção 3 descreve as implementações realizadas; a seção 4 apresenta os resultados obtidos neste trabalho; e por fim, a seção 5 apresenta as considerações finais deste trabalho.

2. Referencial Teórico

Para alcançar os objetivos estilpulados neste trabalho foi necessário o levantamento bibliográfico, feito principalmente através de livros e da pesquisa de artigos de periódicos em plataformas como IEEE, ScienceDirect e CAPES, abrangendo trabalhos tanto em inglês quanto em português. Dos artigos encontrados, 31 foram selecionados para leitura e estudo, dos quais 9 foram analisados aprofundadamente devido à metodologia detalhada, técnicas aplicadas e bons resultados obtidos.

Na primeira parte do levantamento bibliográfico estudou-se sobre o mercado de ações: o que são ações, como funciona este mercado e quais são as formas de analisar o seu comportamento. Em seguida, estudou-se sobre Redes Neurais Artificiais: do que são constituídas, como funcionam e como possuem a capacidade de aprendizado. Por fim, foi estudado sobre os algoritmos genéticos: o que são, como funcionam e como podem ser utilizados para aprimorar as Redes Neurais Artificiais. Os resultados do levantamento bibliográfico são descritos nas seções 2.1 a 2.4.

2.1 Mercado de Ações

Ao abrir uma empresa, o empreendedor deve indicar qual é o seu tipo societário, o que define como ela é organizada em relação aos seus sócios, e qual a responsabilidade de cada um deles para com a empresa. Uma das variáveis existentes que enquadram a empresa em um tipo societário é o seu capital social.

Capital social é o montante de capital que o titular, sócios ou acionistas devem vincular à empresa para o início ou manutenção do negócio [13]. Este capital social pode ser de dois tipos: fechado, quando o dono negocia diretamente as partes da empresa; ou aberto, quando o dono negocia indiretamente estas partes.

As empresas de capital aberto (ou sociedades por ações) pertencem ao tipo societário chamado Sociedade Anônima (S.A. ou S/A), e são entidades legalmente independentes, totalmente separadas de seus proprietários. Isto significa que estas empresas assumem a responsabilidade por suas próprias dívidas, de forma que os acionistas têm responsabilidade limitada, isto é, não podem perder mais dinheiro do que originalmente pagaram por suas ações. Por isso, este tipo de empresa arrecada grandes quantias ao vender ações (direito de propriedade) e emprestar dinheiro [14].

De acordo com [15], uma ação é a menor parcela do capital social das S/A. Ao comprar ações de uma S/A, o indivíduo torna-se acionista e participa do lucro da companhia por meio do recebimento de dividendos e de bonificações. Os acionistas podem ganhar também com a possível valorização do preço das ações no mercado. Entretanto, não há garantia nenhuma de valorização e esse resultado depende fundamentalmente da gestão da companhia e das condições gerais da economia. As ações, quanto a classificação dos direitos e das vantagens que conferem aos seus titulares, podem ser de duas espécies: ordinária, que dão direito à voto ao acionista; e preferenciais, que dão prioridades na distribuição de dividendos e reembolso do capital [16].

O investidor possui quatro maneiras de investir em ações, sendo elas: compra direta, na qual a pessoa transmite uma ordem de aquisição para uma corretora, por telefone ou internet; com fundos de índices, que representam o desempenho de setores de mercado; com clubes de investimento, que são um grupos de pessoas que se unem para investir; e com fundos de investimentos em ações, onde o investidor adquire uma cota de um fundo administrado por corretoras ou bancos [17].

As transferências de ações são feitas nas bolsas de valores, que são associações civis, sem fins lucrativos e com funções de interesse público. Seu papel básico é oferecer um mercado para a cotação dos títulos registrados na bolsa e facilitar a divulgação constante de informações sobre as empresas e sobre os negócios que se realizam sob seu controle. As bolsas de valores também propiciam liquidez às aplicações de curto e longo prazos, por intermédio de um mercado contínuo representado por seus pregões diários [13].

O mercado de ações é dinâmico e possui alto grau de incerteza, pois além da oferta e demanda, notícias do mercado, como fusão de empresas ou mudança de tecnologia, determinam o preço das ações [17]. Por isso, com o objetivo de facilitar a análise geral do mercado de ações, foram desenvolvidos índices. O objetivo de um índice de bolsa de valores é ser o indicador de desempenho médio das cotações dos ativos de maior negociabilidade e representatividade dos mercados de ações [18]. Os índices podem ser divididos em categorias para ter maior especialização. Dentre essas categorias, existem os índices amplos e os índices setoriais. Os índices amplos como o Índice da Bolsa de Valores de São Paulo (IBovespa) juntam, por exemplo, as ações mais negociadas, mais líquidas e que mais pagam dividendos. Já os índices setoriais são aqueles que dividem as ações por setor, refletindo o comportamento agregado do segmento econômico considerado.

Conforme [19], a Brasil, Bolsa, Balcão (B3 S.A.) foi criada em 2017 pela união entre a Bolsa de Valores, Mercadorias e Futuros de São Paulo (BM&F-BOVESPA) com a Central de Custódia e de Liquidação Financeira de Títulos (CETIP). Com isto, a B3 é a única bolsa de valores, mercadorias e futuros em operação no Brasil, também sendo a maior da América Latina. Outras das maiores bolsas do mundo são: New York Stock Exchange (NYSE), London Stock Exchange (LSE), Tokyo Stock Exchange (TSE) e Shangai Stock Exchange (SSE) [20].

Apesar dos mercados financeiros do mundo seguirem uma tendência global, existem diversos fatores que caracterizam e diferenciam esses mercados uns dos outros. Desde padrões de listagem, taxas e ambiente regulatório até o método de avaliação de empresas [21]. Esta identidade que cada mercado possui, além da economia em que estão inseridos, é suficiente para alterar de forma significativa seu comportamento. Para fazer uma avaliação deste comportamento, existem diversas metodologias que podem ser agrupadas em duas abordagens, a fundamentalista e a técnica. Na análise fundamentalista considera-se os fatores que afetam a oferta e demanda de um mercado, visando classificar o preço da ação como subvalorizado, justo ou supervalorizado. Estes fatores são internos de empresa, sua rentabilidade e endividamento, e externos, como juros, inflação, poupança, etc. Na análise técnica, analisa-se o comportamento histórico do mercado e suas tendências para prever o seu estado futuro. Para realizar esta previsão, são utilizados gráficos de preços e ferramentas estatísticas [22].

É importante frisar que a análise fundamentalista ajuda o investidor a decidir qual ação deve ser comprada ou vendida, enquanto a análise técnica ajuda a decidir quando realizar estas operações. Por isso, as duas técnicas são complementares. Vale ressaltar também que a análise técnica é estatística, logo, não resulta em valores certos ou errados, mas sim em valores aproximados, que podem ser satisfatórios ou não.

2.1.1 Índices

Os índices são indicadores numéricos do comportamento do mercado. Estes índices são utilizados na análise técnica com a finalidade de indicar ao investidor qual o momento de realizar uma compra ou uma venda de uma ação. Conforme [23], estes índices podem ser classificados em três grupos:

- Rastreadores de Tendência: indicados para se utilizar quando o mercado está seguindo uma tendência de alta ou de baixa. Alguns exemplos de índices rastreadores de tendência são as médias móveis, MACD, on-balance volume, etc.
- Osciladores: sua utilização é indicada nos momentos em que o mercado está na horizontal, ou seja, não seguem uma tendência. Eles captam os pontos de inflexão no mercado horizontal, e também indicam níveis insustentáveis de otimismo e pessimismo. Alguns exemplos de índices osciladores são o estocástico, a taxa de mudança, o índice de força, etc.
- Mistos: proporcionam um discernimento sobre a psicologia da massa, que representa o senso comum e as tendências gerais do mercado. Alguns exemplos de índices mistos são o *put-call ratio*, *bullish consensus*, *commitments of traders*, etc.

Os índices utilizados neste trabalho estão explanados mais detalhadamente nas seções 2.1.1 até 2.1.1.

MACD

Acrônimo de *Moving Average Convergence-Divergence*, o MACD é um rastreador de tendência que ajuda a analizar o comportamento do mercado por meio de um conjunto de Médias Móveis Exponenciais (MME). O cálculo de uma MME é definida por

$$MME_n = X_n * K + MME_{n-1} * (1 - K) \quad (1)$$

onde X_n corresponde ao preço de fechamento no dia n, e K é definido por

$$K = \frac{2}{N+1} \quad (2)$$

onde N é o tamanho do intervalo da média móvel. Para o caso de $n = 0$, a MME será igual à média dos primeiros n-ésimos valores. Para calcular o MACD, primeiramente é necessário obter a MME dos valores de fechamento, no exemplo dado, uma MME com tamanho de 5 dias. Em seguida, calcula-se uma outra MME, com um intervalo maior que o anterior, por exemplo, 9 dias. Em seguida, calcula-se a Linha MACD, através da diferença entre as duas MMEs. Por fim, calcula-se a Linha de Sinal, através de uma MME da Linha MACD, de tamanho menor que as duas MMEs anteriores. A tabela 1 mostra um exemplo de cálculo do índice MACD tendo como base o Índice Bovespa.

A interpretação deste índice se dá a partir da Linha MACD e da Linha de Sinal representadas na figura 1. O cruzamento das linhas identifica mudanças nas tendências do mercado. Quando a Linha MACD cruza de baixo para cima a Linha de Sinal, é um indicativo de que uma tendência de alta está se iniciando e o investidor deve comprar as ações. Já o cruzamento do MACD de cima para baixo indica que o início de uma tendência de baixa foi identificada e o investidor deve vender suas ações.

Tabela 1. Cálculo do Índice MACD

Data	Fechamento	MME 5 Dias	MME 9 Dias	Linha MACD	Linha de Sinal
2006-09-01	37329				
2006-09-04	37693				
2006-09-05	37368				
2006-09-06	36710				
2006-09-08	36558	37131.6			
2006-09-11	35772	36678.4			
2006-09-12	36147	36501.26			
2006-09-13	36550	36517.51			
2006-09-14	36154	36396.34	36697.88	301.54	
2006-09-15	36170	36320.89	36592.31	271.41	
2006-09-18	36483	36374.92	36570.44	195.51	256.16
2006-09-19	35886	36211.95	36433.55	221.60	238.88

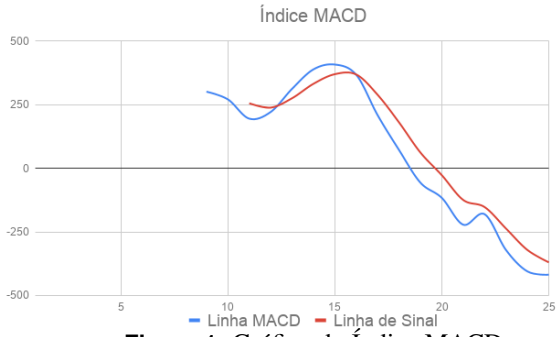


Figura 1. Gráfico do Índice MACD

Sistema Direcional

O Sistema Direcional é um rastreador que auxilia na análise do comportamento do mercado, por meio das movimentações das altas, baixas e fechamentos dos últimos n dias. Neste índice, são utilizadas três curvas para avaliar o mercado: +DI, -DI e ADX.

Para o cálculo do Sistema Direcional, inicialmente deve-se obter o *True Range* (TR), que é o maior valor entre: a distância entre a alta de hoje e a baixa de hoje; a distância entre a alta de hoje e o fechamento de ontem; e a distância entre a baixa de hoje e o fechamento de ontem. Em seguida, obtém-se o *Directional Movement* ($+DM$ e $-DM$) que são as diferenças dos valores de alta de hoje para ontem ($+DM$) e de baixa de ontem para hoje ($-DM$). Por exemplo, se calcularmos a diferença de alta entre o dias $2006-09-04$ e $2006-09-01$ (figura 3), encontraremos o valor 410. Calculando a diferença entre a baixa dos dias $2006-09-01$ e $2006-09-04$, encontraremos o valor -1055. Como a diferença das altas é maior, então $+DM = 410$ e $-DM = 0$. Os cálculos podem ser vistos na tabela 2.

Com esses valores, é possível calcular o *Directional Indicator* (+DI e -DI), através da razão entre +DM e TR e entre -DM e TR. Em seguida, suaviza-se +DI e -DI por médias móveis. Por último, calcula-se DX, por meio da razão entre as diferenças de +DI com -DI e a soma de +DI e -DI. Suavizando esses valores com uma média móvel, gera-se ADX. O cálculo pode ser visto na tabela 3.

O cálculo encontrado nas tabelas 2 e 3 é uma variação encontrada em [24], realizando o cálculo do Sistema Direcional com um n de 5 dias.

Tabela 2. Cálculo do Sistema Direcional - Parte 1

Data	Alta	Baixa	Fechamento	TR	+DM	-DM	TR5
2006-09-01	37329	36232	37329				
2006-09-04	37739	37287	37693	452	410.00	0.00	
2006-09-05	37693	37271	37368	422	0.00	16.00	
2006-09-06	37368	36710	36710	658	0.00	561.00	
2006-09-08	36725	36495	36558	230	0.00	215.00	
2006-09-11	36561	35666	35772	895	0.00	829.00	2,657.00
2006-09-12	36147	35626	36147	521	0.00	40.00	2,646.60
2006-09-13	36726	36034	36550	692	579.00	0.00	2,809.28
2006-09-14	36680	36087	36154	593	0.00	0.00	2,840.42
2006-09-15	36469	35878	36170	591	0.00	209.00	2,863.34

Tabela 3. Cálculo do Sistema Direcional - Parte 2

+DM5	-DM5	+DI5	-DI5	DI5 Dif	DI5 Soma	DX	ADX
410.00	1,621.00	15.43	61.01	45.57	76.43	59.62	
328.00	1,336.80	12.39	50.51	38.11	62.90	60.59	
841.40	1,069.44	29.95	38.07	8.11	68.01	11.93	
673.12	855.55	23.70	30.12	6.42	53.81	11.93	
538.50	893.44	18.81	31.20	12.39	50.00	24.78	33.77
631.80	714.75	22.35	25.29	2.93	47.63	6.160	28.25

Este índice trabalha com a identificação de tendências. Quando as linhas de DI se separam, a linha ADX sobe, indicando uma tendência que continua de maneira vigorosa. ADX declina quando ocorre a reversão de uma tendência ou quando o mercado entra em um impasse.

Este índice também possibilita a identificação de prováveis começos de uma nova tendência. É identificado o começo de uma nova tendência de alta quando +DI está no topo e de baixa quando -DI está no topo, enquanto que ADX sinaliza a intensidade dessa tendência [23]. Conforme pode ser visto na figura 2, logo a partir do décimo dia foi um bom momento de vender as ações, pois -DI e ADX estavam acima de +DI, além de que ADX estava crescente neste momento.

Elder Ray

O *Elder Ray* é um índice oscilador que indica a força dos expeculadores em aumentar o preço das ações, frequentemente chamados de *bull*, contra a força dos expeculadores em diminuir o preço das ações, frequentemente chamados de *bear*. A força destes dois grupos de expeculadores é calculada com base em uma MME e nas variações intradiárias das ações, e é dada pelas equações a seguir:

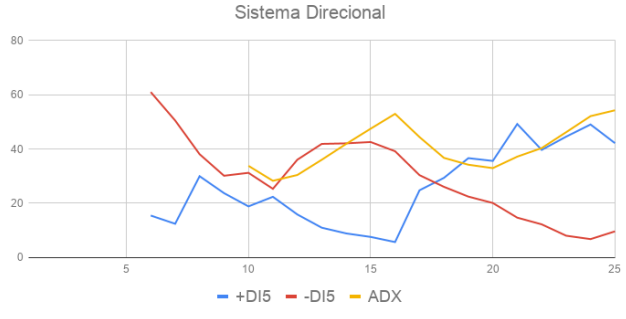


Figura 2. Gráfico do Sistema Direcional

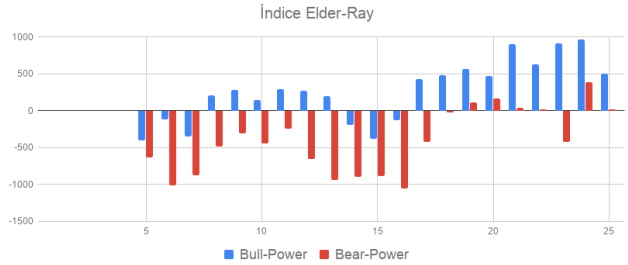


Figura 3. Gráfico do Índice Elder Ray

$$Bull - Power = Maxima_{doDia} - MME \quad (3)$$

$$Bear - Power = Minima_{doDia} - MME \quad (4)$$

A tabela 4 mostra um exemplo do cálculo do índice *Elder Ray* com uma MME de 5 dias e a figura 3 apresenta os valores calculados. O índice pode ser aplicado através da identificação de pontos de entrada e de saída nas operações a partir dos máximos e mínimos dos valores na direção da tendência representada pela inclinação da MME.

Índice de Força

O Índice de Força é um oscilador de cálculo simples que leva em consideração o volume de transações e o valor de fechamento, descrito pela equação 5:

$$IF = Volume_{hoje} * (Fechamento_{hoje} - Fechamento_{ontem}) \quad (5)$$

A tabela 5 traz um exemplo do cálculo do Índice de Força.

O índice de força reflete o grau de comprometimento da massa de participantes no mercado por considerar o volume no cálculo, indicando a força das tendências.

Tabela 4. Cálculo do Índice Elder Ray

Data	Alta	Baixa	Fechamento	MME	Bull	Bear
2006-09-01	37329	36232	37329			
2006-09-04	37739	37287	37693			
2006-09-05	37693	37271	37368			
2006-09-06	37368	36710	36710			
2006-09-08	36725	36495	36558	37131.6	-406.6	-636.6
2006-09-11	36561	35666	35772	36678.4	-117.4	-1012.4
2006-09-12	36147	35626	36147	36501.26	-354.26	-875.26
2006-09-13	36726	36034	36550	36517.51	208.48	-483.51
2006-09-14	36680	36087	36154	36396.34	283.65	-309.34
2006-09-15	36469	35878	36170	36320.89	148.10	-442.89
2006-09-18	36670	36134	36483	36374.92	295.07	-240.92

Tabela 5. Cálculo do Índice de Força

Data	Fechamento	Volume	Índice de Força
2006-09-01	37329	98953400	
2006-09-04	37693	83711000	30470804000
2006-09-05	37368	104148400	-33848230000
2006-09-06	36710	99016400	-65152791200
2006-09-08	36558	80359000	-12214568000

Triple-Screen

Um problema dos índices é que frequentemente eles são contraditórios entre si. Por isso, [23] sugere a técnica *Triple-Screen*. Esta técnica consiste em analisar a tendência do mercado em três crivos. No primeiro crivo analiza-se a tendência de longo prazo do mercado, por meio de rastreadores de tendência, para determinar se o investidor deverá comprar ou vender. No segundo crivo, utiliza-se um oscilador nos valores diários, para identificar o melhor momento de realizar a compra ou venda. Por fim, no terceiro crivo analisa-se os valores intradiários das ações, a fim de otimizar os lucros.

Tendo isto em vista, neste trabalho os dois rastreadores de tendência e os dois osciladores mais indicados por [23] são utilizados para realizar a previsão do Índice Bovespa, de maneira similar à técnica *Triple Screen*.

Para analizar a tendência de longo prazo do mercado, neste trabalho foram empregados como rastreadores de tendência o Índice MACD, utilizando como MMEs os valores de 9, 12 e 26 dias, e o Sistema Direcional, com as MMEs de 13 dias. Já para a análise de curto prazo, foram utilizados os osciladores Elder Ray, com uma MME de 13 dias, e o Índice de Força.

2.2 Redes Neurais Artificiais

Uma RNA é um algoritmo que modela a maneira como o cérebro realiza uma tarefa particular ou função. A motivação para o desenvolvimento das RNA deve-se ao fato do reconhecimento de que o cérebro humano processa informações de uma forma inteiramente diferente do computador digital convencional, pois o cérebro é um computador (sistema de processamento de informação) altamente complexo, não-linear e paralelo [25].

Em linhas gerais, uma RNA constitui-se de inúmeras unidades de processamento simples, chamadas de neurônios, que trabalham paralelamente em grande escala e possuem a capacidade de aprender. Um benefício do processo de aprendizagem de RNAs é a sua característica de generalizar e associar dados além de ter uma tolerância à dados ruidosos [26].

Tradicionalmente, o processo de aprendizagem das RNAs se dá por um algoritmo de aprendizagem, cuja função é modificar os pesos sinápticos. Entretanto, é possível também para uma rede neural modificar sua própria topologia, o que é motivado pelo fato de os neurônios no cérebro humano poderem morrer e que novas conexões sinápticas possam crescer [25].

2.2.1 Neurônio

Os neurônios são as unidades básicas de uma RNA e cada um deles é uma unidade de processamento de informação. O neurônio possui três elementos básicos, como apresentado na figura 4.

- um conjunto de sinapses (ou elos de conexão), em que cada conexão é caracterizada por um peso próprio.
- um somador, para somar os sinais de entradas ponderados pelas respectivas sinapses do neurônio.
- uma função de ativação, para restringir a amplitude da saída de um neurônio.

Conforme a figura 4, o funcionamento de um neurônio é dado da seguinte forma: o neurônio possui um conjunto de sinapses de entrada, que ligam um neurônio i a um neurônio j. Estas sinapses propagam o valor de ativação $y_i(n)$ para j, e são associadas à um peso $w_{ji}(n)$, que determina a intensidade do sinal de conexão. Com isso, é calculada a função de entrada $\upsilon_j(n)$, dada por

$$\upsilon_j(n) = \sum_{i=0}^{n} w_{ji}(n) y_i(n) \tag{6}$$

onde n é a iteração do processo. Em seguida, aplica-se a $\upsilon_j(n)$ a função de ativação $\varphi_j()$. O resultado desta operação é o valor de saída (ativação) do neurônio j, ou y_j, e é dado por

$$y_j(n) = \varphi_j(\upsilon_j(n)) = \varphi_j\left(\sum_{i=0}^{n} w_{ji}(n) y_i\right) \tag{7}$$

[3] apresentam as principais funções para se utilizar como função de ativação:

linear

$$\varphi_j(x) = bx \tag{8}$$

logística, unipolar ou sigmoidal

$$\varphi_j(x) = a * \frac{1}{1+e^{-bx}} \tag{9}$$

e tangente hiperbólica

$$\varphi_j(x) = a * \frac{e^{bx} - e^{-bx}}{e^{bx} + e^{-bx}} \tag{10}$$

Por fim, o valor de ativação $y_j(n)$ é propagado para os próximos neurônios, por meio do conjunto de sinapses de saída.

2.2.2 Arquiteturas

A forma como os neurônios são ligados pelas conexões determina como ocorre a propagação da informação entre os neurônios, definindo a arquitetura da RNA. Esta alimentação pode ser cíclica, sendo chamada de retroalimentada, ou acíclica, podendo conter uma única camada ou múltiplas camadas.

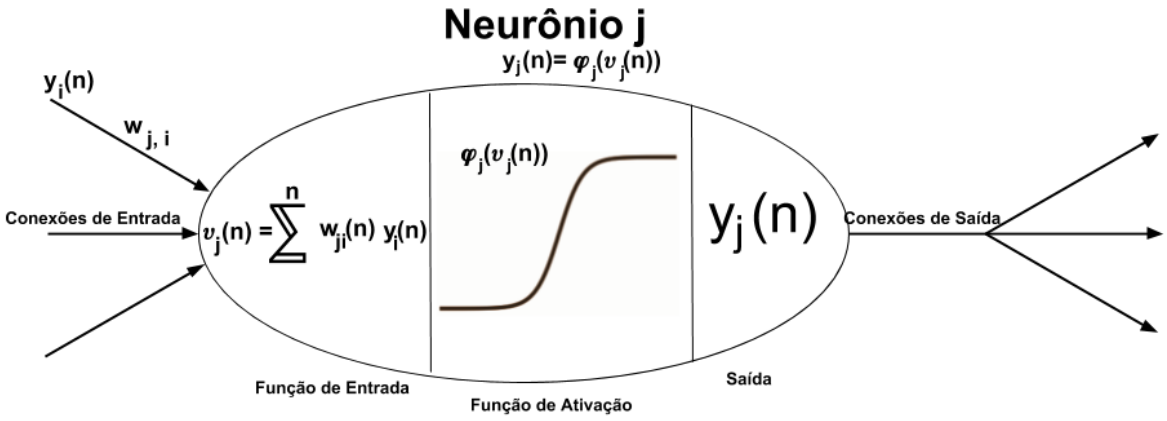


Figura 4. Modelo de um neurônio
Fonte: Adaptado de [1]

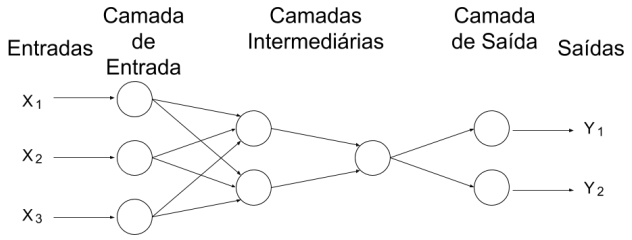


Figura 5. Modelo de uma rede acíclica multi-camadas

Em uma rede acíclica de múltiplas camadas, os neurônios são dispostos em camadas, e a saída de uma camada é a entrada da camada imediatamente posterior [1]. Neste modelo, existem pelo menos duas camadas, a de entrada e a de saída, que dá o retorno da rede neural. Pode haver camadas intermediárias, também conhecidas como camadas ocultas. Segundo [3], estas redes são as mais populares por apresentarem algoritmos de aprendizado bastante difundidos, além de suas capacidades de aproximar, dependendo da quantidade de neurônios, qualquer função não linear. A figura 5 traz um exemplo de RNA com arquitetura acíclica multi-camadas.

2.2.3 Aprendizado

O procedimento utilizado para realizar o processo de aprendizagem é chamado de algoritmo de aprendizagem. Sua função é modificar os pesos sinápticos da rede de uma forma ordenada para alcançar um objetivo de projeto desejado [25]. Os algoritmos de aprendizagem podem ser classificados em três tipos: supervisionado, não-supervisionado e por reforço. O foco deste trabalho é o aprendizado supervisionado. Neste tipo de aprendizagem, é apresentado à rede pares, contendo os valores de entrada e as respectivas saídas esperadas. A resposta da rede é comparada com a saída esperada e calcula-se um sinal de erro, que é utilizado para calcular o ajuste que deve ser feito nos pesos sinápticos da rede [3].

Uma das regras de aprendizado supervisionado é a Regra Delta Generalizada, também conhecida como Algoritmo de Retropropagação ou *Backpropagation*. No Algoritmo de Retropropagação, o tempo é discreto e utiliza-se o método de gradiente descendente para correção do erro. O algoritmo executa um mapeamento entrada-saída através da minimização de uma função de custo. Esta função é minimizada pelo ajuste dos pesos sinápticos de acordo com o erro quadrático acumulado, para todos os padrões do conjunto de treinamento [3].

Em uma RNA que utiliza o algoritmo de retropropagação, o aprendizado resulta das muitas apresentações de um determinado conjunto de exemplos de treinamento (pares de entrada e saída) a ela. Uma apresentação do conjunto de treinamento inteiro é denominada época e o processo de aprendizagem é realizado de época em época até os pesos sinápticos e os níveis de *bias* se estabilizarem e o erro médio quadrado sobre todo o conjunto de treinamento convergir para um valor mínimo [25].

A aprendizagem por retropropagação pode ser compreendida através do Algoritmo 1. Este algoritmo recebe dois parâmetros: *exemplos*, um conjunto de exemplos em que cada item tem um vetor de entrada x_i e um vetor de saída d_i, e *rede*, uma rede multicamadas com L camadas, pesos $w_{ji}(n)$ e função de ativação $\varphi_j()$. Além disso, considera-se a

Algoritmo 1 Pseudocódigo de Retropropagação
Fonte: Adaptado de [1]

1: **função** RETROPROPAGAÇÃO(exemplos, rede)
2: **faça**
3: **para cada** peso w_{ji} na rede **faça**
4: $w_{ji}(n) \leftarrow$ *um numero aleatorio pequeno*
5: **fim para**
6: **para cada** exemplo $(x,d) \in$ exemplos **faça**
▷ Propaga as entradas para frente, para computar as saídas
7: **para cada** nó i na camada de entrada **faça**
8: $y_i(n) \leftarrow x_i$
9: **fim para**
10: **para** $l \leftarrow 2$ **até** L **faça**
11: **para cada** nó j na camada l **faça**
12: $\upsilon_j(n) \leftarrow \sum_{i=0}^{n} w_{ji}(n) \times y_i(n)$
13: $y_j(n) \leftarrow \varphi_j(\upsilon_j(n))$
14: **fim para**
15: **fim para**
▷ Propaga deltas da camada de saída para a de entrada
16: **para cada** nó j da camada de saída **faça**
17: $\delta[j](n) \leftarrow \varphi'_j(\upsilon_j(n)) \times (d_j - y_j)$
18: **fim para**
19: **para** $l \leftarrow L-1$ **até** 1 **faça**
20: **para cada** nó i na camada l **faça**
21: $\delta[i](n) \leftarrow \varphi'_i(\upsilon_i(n)) \times \sum_{j=1}^{n} w_{ji}(n) \times \delta[j](n)$
22: **fim para**
23: **fim para**
▷ Atualiza cada peso na rede usando deltas
24: **para cada** peso $w_{i,j}$ na rede **faça**
25: $w_{ji}(n) \leftarrow w_{ji}(n) + \alpha \times \Delta w_{ji}(n-1) + \eta \times y_i(n) \times \delta[j]$
26: **fim para**
27: **fim para**
28: **enquanto** nenhum critério de parada tenha sido satisfeito
29: **fim função retorna** *rede*

existência da variável local δ que representa o gradiente local de cada neurônio da rede.

No algoritmo 1, pode ser visto nas linhas 12 e 13 o cálculo das variáveis $\upsilon_j(n)$, que é a entrada do neurônio j, e $y_j(n)$, que é a saída do neurônio calculada por meio da função de ativação $\varphi_j()$. Estas instruções do algoritmo implementam as equações 6 e 7, e a função $\varphi_j()$ é qualquer uma das equações 8, 9 ou 10. Já o símbolo φ', que aparece nas linhas 17 e 21, é a derivada da função $\varphi_j()$, utilizado para o cálculo de δ.

Detalhando melhor o algoritmo, $\delta_j(n)$ é o gradiente local do neurônio j na iteração n, e é utilizado para o cálculo da variação Δ que será aplicada ao peso $w_{ji}(n)$. Caso j seja um neurônio de saída, seu cálculo é dado pela linha 17 do algoritmo, conforme a equação

$$\delta_j(n) = e_j(n) * \varphi'_j(\upsilon_j(n)) \tag{11}$$

onde $e_j(n)$ é o erro do neurônio j na iteração n, calculado por $d_j(n) - y_j(n)$. Caso o neurônio j seja um neurônio de camada intermediária, $\delta_j(n)$ é calculado pela linha 21, conforme a equação

$$\delta_j(n) = \varphi'_j(\upsilon_j(n)) * \sum_{k=0}^{K} \delta_k(n) * w_{kj}(n) \tag{12}$$

onde k é o índice dos neurônios da primeira camada à direita de j. Por fim, a variação Δ aplicada ao peso sináptico $w_{ji}(n)$ é dada por

$$\Delta w_{ji} = \alpha * \Delta w_{ji}(n-1) + \eta * \delta_j(n) * y_j(n) \tag{13}$$

onde $\Delta w_{ji}(n-1)$ é a variação de w_{ji} na iteração anterior, para $n \neq 0$. η é a taxa de aprendizagem, e serve para controlar as variações dos pesos sinápticos. Com um valor pequeno, a trajetória no espaço de pesos será suave, porém tornando a aprendizagem lenta. Utilizando valores altos, acelera-se a

aprendizagem, porém com o risco de tornar a rede instável [25]. α é chamado de constante de momento, e o termo $\alpha\Delta w_{ji}(n-1)$ da equação 13 serve para acelerar a aprendizagem da rede evitando o perigo da instabilidade [25].

É mencionado também, na linha 28, o critério de parada, para que a função de retropropagação não execute infinitamente. Os critérios de parada existentes são: uma limitação no número de épocas; quando o erro total $\sum_{j=1}^{N} e_j^2(n)/N$ atinja um valor mínimo; ou ambos os critérios, interrompendo a execução quando qualquer um dos critérios for atingido por primeiro.

2.2.4 RNAs e Séries Temporais

As RNA foram desenvolvidas primariamente para o uso de reconhecimento de padrões (classificação) e não apresentavam bons resultados na modelagem de séries temporais, pois as RNA tradicionais se preocupavam apenas com a detecção de padrões em conjuntos de valores que não mudavam em relação ao tempo. A natureza dinâmica dos dados temporais e espaço-temporais demanda a introdução de novos mecanismos [27].

Para que seja possível o processamento de séries temporais por uma RNA, deve existir algum tipo de memória. Esta memória pode ser dar pelo fornecimento de um intervalo temporal para a RNA, ao usar valores do passado de uma série para um próximo processamento, ou pelo armazenamento de valores passados dos neurônios nas camadas oculta e de saída [27].

Com estas adaptações, as RNA possuem grande poder de decisão em cenários em que se pode identificar séries temporais como na área financeira. Segundo [28], a previsão do valor de ações baseada em séries temporais é um dos campos ativos de pesquisa na área de finanças, matemática, física e aprendizado de máquina. Inicialmente, essa previsão era feita através dos métodos estatísticos de análise, mas nas últimas décadas um grande número de modelos de RNA foram propostos para resolver este problema e o modelo estatístico combinado com RNA híbridas traz melhores resultados quando comparado a um modelo de RNA pura combinada com o um modelo estatístico, como visto em [28].

A memória através da entrada é normalmente implementado nas RNA de múltiplas camadas que se assemelham aos modelos autoregressivos como ARIMA, que são métodos clássicos para processamento de séries temporais no contexto de finanças. Neste tipo de abordagem, a entrada da RNA funciona como uma janela deslizante de tamanho n: se um dado novo entra na janela (X_{t+1}), o valor mais antigo da janela (X_{t-n}) é descartado. O objetivo da RNA é sempre prever o próximo valor que entrará na janela. Quando um novo dado chega (X_{t+2}) ele é adicionado na janela, o último elemento da janela é removido (X_{t-n+1}) e com estes novos n valores a rede tentará prever o valor em $t+3$. Uma vantagem desta abordagem é a capacidade de considerar mais características complexas da série temporal. Como desvantagem há o grande número de dados de amostra requeridos devido ao grande número de parâmetros e o possível problema de *overfitting* [29].

Já o segundo método, que consiste em armazenar valores passados na própria rede, pode ser visto em redes de Jordan e redes de Elman, que nada mais são que RNA que possuem camadas que são utilizadas para realimentação de entradas. Estas camadas podem ser ocultas (camada de estado em redes de Elman) ou a própria camada de saída onde é feita uma retroalimentação para a consideração desses valores no próximo processamento [27]. Embora este tipo de RNA seja utilizado para resolver problemas de séries temporais, um histórico muito grande não é comportado além de necessitar de algoritmos diferentes para o treinamento da rede, uma vez que algoritmos comuns como a retropropagação podem causar problemas [29]. Devido às combinações não-lineares realizadas pela RNA através das funções de ativação, uma mudança infinitesimal de uma entrada temporalmente distante traz efeitos no treinamento que não são mensuráveis pelo gradiente descendente, acarretando no desaparecimento do mesmo, ou seja, os pesos do modelos recebem atualizações virtualmente nulas e a RNA não aprende [25].

Portanto neste trabalho foi escolhido o primeiro modelo em que a memória é uma entrada no sistema para que seja possível utilizar os índices descritos em 2.1.1, que incluem no próprio cálculo para obtê-los dados referentes ao passado. Assim é possível seguir a recomendação de [28] utilizando o modelo estatístico em conjunto com a RNA.

2.3 Algoritmos Genéticos

Os Algoritmos Genéticos (AGs) constituem uma técnica de busca inspirada no processo de evolução dos seres vivos baseada na seleção natural de Darwin. Em uma busca, existe uma série de fatores que influenciam o desempenho de um dado sistema. Esses fatores podem assumir um número limitado ou ilimitado de valores e podem ser sujeitos a certas restrições. O objetivo é encontrar a melhor combinação dos fatores dentro do espaço de busca que é composto de todas as possibilidades de combinações [3].

Considerando os sistemas biológicos como um todo, observa-se que os mesmos desenvolveram ao longo da sua evolução estratégias de adaptação de comportamento que possibilitaram a sua sobrevivência e a perpetuação de suas espécies. As pressões do ambiente fizeram com que estas estratégias tivessem um forte impacto sobre os organismos biológicos, gerando profundas mudanças nos mesmos [3].

Baseado nesta analogia com o processo de evolução biológica das espécies, os AG mantêm a informação sobre o ambiente, acumulando-a durante o período de adaptação. Eles utilizam tal informação acumulada para podar o espaço de busca e gerar novas soluções plausíveis dentro do domínio [3].

Os AGs possuem os seguintes elementos: uma população de cromossomos, seleção através de funções de adaptabilidade, cruzamento para produzir uma nova geração de cromossomos e mutação aleatória dessa nova geração [30].

Dado um problema bem definido a ser resolvido e uma representação das possíveis soluções em sequências de va-

lores, um algoritmo genético funciona conforme os passos descritos na figura 6.

Para ilustrar o funcionamento de um AG é apresentado um problema que consiste de uma caixa preta com cinco interruptores externos que podem estar ligados (1) ou desligados (0). Cada combinação de estados dos interruptores resulta em um valor diferente da função para $f(x) = x^2$, que é o sinal de saída da caixa preta. O objetivo deste problema é encontrar a configuração de interruptores que resulta no maior valor possível da função $f(x)$.

2.3.1 Cromossomo

Um cromossomo se refere a uma possível solução para o problema e é representado por uma sequência de valores, tipicamente *bits* [30]. Cada valor do cromossomo, também chamado de gene, está contido em um conjunto de possíveis números que aquele aspecto da solução pode assumir.

Para preparar o exemplo para uso de um AG, deve-se primeiro identificar o cromossomo. Neste caso, para transformar os interruptores em uma sequência de valores, basta representar seus estados de forma binária: 1 para ligado e 0 para desligado. Assim, uma configuração específica de cinco interruptores pode ser dada pela sequência dessa representação como por exemplo 00000 onde todos os interruptores estão desligados, conforme a figura 7.

2.3.2 Adaptabilidade

Todo cromossomo é avaliado quanto à sua adaptabilidade em cada geração. Essa avaliação é feita a partir da capacidade de resolver o problema que cada cromossomo possui e o resultado é chamado de adaptabilidade. Com um valor de adaptabilidade, torna-se possível classificar cada cromossomo para determinar as melhores soluções para o problema representado por eles.

No problema da caixa preta, cada combinação de estados reflete numa saída diferente da caixa preta e consequentemente de $f(x)$. Como o objetivo é maximizar a saída, quanto maior o valor obtido na saída, melhor é o conjunto de entrada.

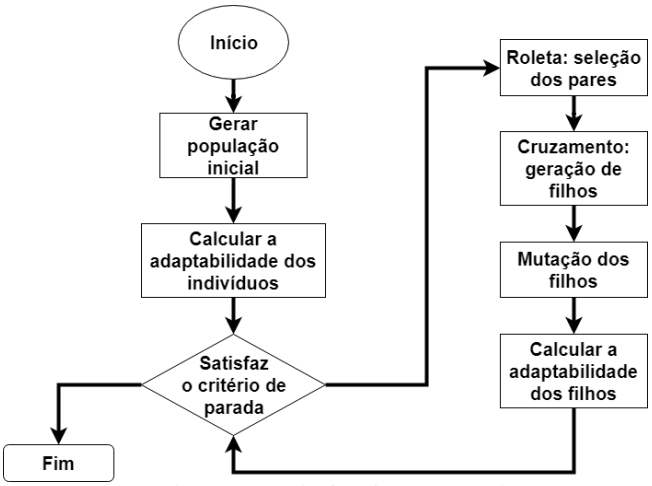


Figura 6. Fluxograma do funcionamento de um AG

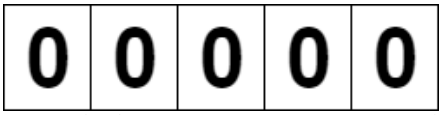


Figura 7. Exemplo de um cromossomo para o problema da caixa preta

Portanto, o sinal de saída da caixa preta serve como uma forma de avaliar a solução proposta.

2.3.3 Seleção

Tendo uma população de cromossomos e uma forma de avaliá-los, a próxima etapa em um AG é selecionar pares de cromossomos para que eles possam se reproduzir.

A seleção é feita a partir dos resultados de uma roleta ponderada que considera o nível de adaptabilidade de cada cromossomo, fazendo com que os cromossomos mais aptos sejam escolhidos com maior frequência sem excluir a possibilidade de que os piores cromossomos também sejam selecionados.

Uma consequência da roleta ponderada é que um cromossomo pode ser escolhido mais de uma vez enquanto outros não são escolhidos [1].

2.3.4 Cruzamento

Para cada par de cromossomos selecionados sorteia-se um valor. Caso esse valor seja maior que a probabilidade predeterminada de cruzamento, é realizado um cruzamento entre o par e por consequência deste processo dois novos cromossomos são gerados. Caso o valor seja inferior à probabilidade, os cromossomos originais não são alterados nessa etapa, sendo simplesmente copiados para a próxima geração.

Para realizar um cruzamento é necessário encontrar pontos de corte, que dividam o cromossomo para que os valores sejam trocados entre o par, assim gerando dois cromossomos novos com partes de cada um dos cromossomos originais.

No exemplo da caixa preta, apenas um ponto de corte é escolhido aleatoriamente e o cruzamento se dá conforme é mostrado na figura 8.

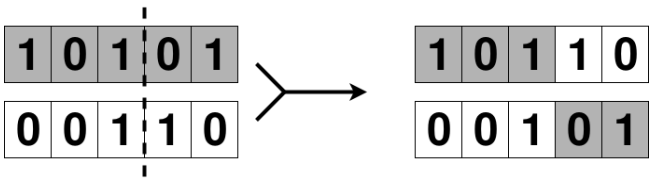


Figura 8. Cruzamento de cromossomos
Fonte: Adaptado de [1]

2.3.5 Mutação

Após o processo de seleção, sorteia-se um novo valor referente à probabilidade de ocorrência de uma mutação em cada cromossomo, que normalmente é muito menor que a chance de cruzamento.

A mutação de um cromossomo acontece através da alteração de apenas um dos valores da sequência do cromos-

somo. No exemplo dos interruptores, basta inverter um bit do cromossomo, ou seja, alterar o estado do interruptor.

2.3.6 Aplicações

A busca e otimização podem ser resolvidas por meio de métodos probabilísticos, numéricos, enumerativos ou por hibridismo destes métodos. Os AGs fazem parte da classe correspondente aos métodos probabilísticos de busca e otimização e apesar de não serem puramente aleatórios, usam conceitos de probabilidade para direcionar a busca para regiões onde é provável que os pontos ótimos estejam [3].

Os AGs podem ser enquadrados em grande parte dos problemas científicos a serem formulados como problemas de busca e otimização [3]. Um exemplo disso é a otimização dos parâmetros de RNAs, tendo a capacidade de melhorar os resultados de previsão de séries temporais na forma de valores do mercado de ações conforme visto em [2].

2.4 Trabalhos Relacionados

Existem inúmeras formas de combinar RNA com AG para se desenvolver um modelo híbrido. [11] agrupou os trabalhos existentes em 4 grandes categorias, descritas a seguir:

- Pesquisar por um conjunto de pesos ótimo: nesta categoria, os pesos w_{ji} são codificados em cromossomos e um AG é utilizado para encontrar a melhor combinação de pesos.
- Pesquisar sobre o espaço de topologia: nesta categoria, o AG é utilizado para encontrar uma quantidade de neurônios na camada intermediária que permita com que a RNA alcance um aprendizado melhor.
- Pesquisar por parâmetros de aprendizado ótimos: nesta categoria, cada gene do cromossomo é um dos parâmetros que afetam a etapa de aprendizado da RNA, como α e η por exemplo.
- Abordagens genéticas que modificam a retropropagação: nesta abordagem, a cada iteração do processo de retropropagação, uma operação de mutação é realizada, realizando um incremento nos pesos w_{ji}.

[11] desenvolveu uma técnica em que o AG é utilizado para encontrar os melhores pesos w_{ji} iniciais, porém não substituindo a retropropagação. Uma população é criada, na qual os cromossomos possuem os pesos iniciais da RNA. Para cada cromossomo, os valores são decodificados para a RNA, que a partir destes pesos iniciais realiza o treinamento (executa do passo 6 em diante do pseudocódigo 1). O desempenho desta RNA em um conjunto de teste é utilizado como valor de adaptabilidade para o respectivo cromossomo. Cabe ressaltar que as alterações realizadas pela retropropagação modificam apenas os pesos que estão na RNA, não alterando os valores originais do cromossomo.

Desta forma, uma pesquisa por pontos ótimos globais é realizada pelo AG, enquanto uma pesquisa por pontos ótimos locais é realizada pela retropropagação, simultaneamente. Além disso, [11] utiliza operações genéticas, como a mutação por exemplo, para incrementar ou decrementar a quantidade de neurônios na camada intermediária. Pode-se dizer que o método de [11] se encaixa no quarto grupo apresentado.

Os testes de [11] foram realizados utilizando uma base de dados contendo dez atributos sobre diagnósticos de câncer de mama. O método desenvolvido foi comparado com RNA puras que obtiveram um erro médio de 15%, contra 4.4% do método desenvolvido.

[31] criou um método híbrido que se encaixa no segundo e no quarto grupo apresentado. No referido trabalho, foi utilizado um AG para gerar os valores iniciais de w_{ji}, selecionar os dados de entrada para a RNA e também definir a quantidade de neurônios intermediários. Esta RNA é utilizada para a detecção de câncer de mama, a partir de uma base de dados que possui 9 atributos para cada instância. Cada atributo é uma *feature* que será utilizada ou não como entrada da RNA.

Neste trabalho, o cromossomo possui valores estritamente binários. Os primeiros P-ésimos bits do cromossomo indicam a utilização ou não de um gerador de pesos iniciais randômico. Assim como no trabalho de [11], a retropropagação não é substituída. Em seguida, os próximos Q-ésimos bits do cromossomo indicam a quantia, em número binário, de neurônios na camada intermediária. Por fim, os últimos R-émos bits indicam quais *features* serão utilizadas.

Após a decodificação dos cromossomos, o trabalho segue de maneira similar ao trabalho anterior. Para cada RNA, a retropropagação é executada a partir da linha 6. Os desempenhos das RNA são utilizados para calcular a adaptabilidade do cromossomo.

Como testes, [31] comparou o seu método utilizando três variações diferentes do algoritmo de retropropagação, alcançando como pior resultado uma taxa de acerto média de 93.14% e um desvio de 4.4%. Já na melhor combinação, foi alcançado uma taxa de acerto média de 98.29% e um desvio de 0.8%.

Em [32] os autores deram continuidade a um trabalho também feito por eles sobre previsão de valores do índice tailandês SET50 com uma RNA pura. A entrada desta RNA pura utiliza combinações de 11 índices diferentes em que cada índice possui 4 possíveis valores referentes a intervalos de previsão diferentes, totalizando assim 44 valores que servem como entrada. No caso da RNA utilizar todos os valores calculados, o custo computacional do treinamento seria muito alto e consequentemente levaria muito tempo.

A melhoria proposta neste trabalho em comparação ao anterior foi a integração de um AG com a RNA para encontrar a melhor entrada da rede composta de um subconjunto dos 44 valores de índices calculados. Para representar este problema, os cromossomos foram codificados por uma sequência de bits e são iniciados aleatoriamente. Cada bit representa a seleção de um determinado valor de índice e o cromossomo como um todo representa a combinação dos valores que servem de entrada para a RNA. Com os cromossomos definidos, a

RNA é treinada e classifica o conjunto de teste, encontrando a adaptabilidade referente à taxa de acerto de predição que cada cromossomo obteve. Após isso é dada continuação às operações genéticas.

Foram realizados testes utilizando dados de um intervalo de seis anos em que cada teste é referente a um intervalo de um ano. A hibridização da RNA através do AG trouxe melhores predições do que a RNA pura não só para cada ano mas também em média, aumentando a taxa de acerto médio de 56.58% para 63.60% melhorando os resultados em 12.4%

No trabalho de [2] foi usado um conjunto de tecnologias para minimizar erros de predições em retornos de investimento a partir de quatro índices europeus, sendo estas tecnologias RNA, LF, AG e combinações entre elas.

Além da RNA de múltiplas camadas com retropropagação foi apresentado um AG e possíveis formas de como modificar a RNA para obter melhorias no processo de previsão, sendo uma das hibridizações a busca da melhor arquitetura de RNA através do AG. Outra combinação que participou do experimento foi a RNA *fuzzy* que trata do uso da lógica *fuzzy* para determinar o intervalo dos pesos das conexões nas camadas ocultas da RNA a partir dos valores das ações de mínimo e de máximo de um determinado dia.

Ao testar os modelos híbridos de RNA com AG e *fuzzy* com os 4 índices e 3 medidas de erro, os resultados da pesquisa indicaram que a RNA com AG teve melhores resultados que a RNA com *fuzzy* tanto em precisão quanto em robustez, sendo sua Média Percentual Absoluta do Erro (MPAE) médio nos testes realizados 0.495, prevalecendo sobre o outro algoritmo que atingiu 0.6025.

Em [33] é explicada a limitação da capacidade que RNA puras tem de encontrar uma solução para o problema, pois a RNA pode demandar muito esforço computacional se sua arquitetura for muito grande ou decorando o conjunto de treinamento se for muito pequena.

Uma das soluções para esse problema é a técnica de poda, que, neste trabalho, foi implementada através de um AG co-evolucionário cooperativo híbrido com poda. Este algoritmo é utilizado para dividir a RNA complexa em várias sub-redes simples. Essas sub-redes formam diversos grupos e cada grupo realiza operações genéticas independentemente, como seleção, cruzamento e mutação. No processo evolucionário, os grupos trocam informação entre si somente durante a avaliação de indivíduos. Assim, somente pela cooperação esses grupos conseguem evoluir e concluir a tarefa de otimização.

Com 8 variáveis de entrada testadas para 2 índices diferentes em um intervalo de 210 dias, três técnicas foram testadas, RNA com AG, algoritmo co-evolucionário cooperativo e o algoritmo co-evolucionário cooperativo híbrido. O último algoritmo foi capaz de atingir o melhor resultado entre os três, atingindo um erro quadrático médio de 0.00237, aproximadamente dez vezes menor que a RNA com AG (0.02429) e mais de duas vezes menor que o algoritmo co-evolucionário cooperativo não-híbrido (0.00549) para uma das bases de dados, com a outra trazendo resultados em proporções similares quanto ao erro quadrático médio.

3. Materiais e Métodos

Neste trabalho, a partir do estudo da seção 2, foi iniciada a etapa experimental do projeto por meio da implementação de uma RNA como um grafo acíclico orientado, conforme descrito por [3]. Esta implementação garante uma maior flexibilidade à RNA que se torna altamente configurável e, diferente dos simuladores, capaz de suportar todas as futuras hibridizações.

No estudo realizado foi observado uma melhoria a partir da adição de AG para auxiliar ou complementar a RNA, como nos trabalhos de [2] e [32]. Assim optou-se por aprimorar o desempenho da RNA através de hibridizações com AG diferentes para que seja possível comparar esses métodos entre si além da comparação com a RNA pura.

3.1 Coleta e Tratamento de Dados

Os valores de entrada para a RNA foram obtidos por meio dos índices, já descritos na seção 2.1.1, com base nos valores históricos do IBOVESPA, a partir da data de 01/09/2006 até 11/04/2018 disponíveis em [34]. Todas as linhas da tabela que não possuíam o valor do volume ou com valores *null* (como feriados, onde a bolsa não abre) foram excluídas da base de dados.

Já os valores utilizados como saída da RNA foram gerados similarmente ao trabalho de [32], conforme a equação 14.

$$SaidaRNA_{hoje} = \begin{cases} 1, & \text{se } Fechamento_{amanha} > Fechamento_{hoje} \\ -1, & \text{se não} \end{cases} \tag{14}$$

Com os valores de entrada e saída obtidos, a base de dados foi montada conforme a técnica utilizada em mineração de dados chamada *Hold Out*. Nesta técnica, divide-se a base de dados em dois grupos: base de treinamento e base de teste. A base de treinamento serve para treinar um modelo que será utilizado para prever/classificar valores, enquanto que a base de teste serve para simular o desempenho que o modelo obteria ao prever/classificar valores inexistentes na base de dados. O *Hold Out* utilizado separou 70% dos dados mais antigos para o treinamento do modelo, enquanto que os 30% dos dados mais recentes foram utilizados para simular o desempenho do modelo em dados não vistos. A ordem de cada instância não foi alterada para que a sequência temporal dos dados não fosse perturbada.

Além disso, para fins de comparação de desempenho, foram criadas duas base de dados. Uma base X_1, que utiliza todos os valores obtidos do IBOVESPA totalizando 12 anos, e outra base X_2, contendo apenas os valores dos últimos três anos. Com isso, testes foram realizados para averiguar se uma base de dados grande garantiria uma taxa de acerto maior ou se prejudicaria o modelo, levando a RNA a uma superadaptação.

Por último, a partir das bases X_1 e X_2 foram geradas outras n bases de dados, X_{1n} e X_{2n}. A diferença destas bases é que, como valores de entrada, a RNA recebe não apenas os índices do dia corrente, mas também dos últimos n-ésimos dias.

3.2 Medidas de Avaliação

Um fator importante para o estudo de previsão de valores e para comparações de trabalhos é determinar a medida de avaliação a ser utilizada. Uma medida de avaliação é uma forma de calcular o desempeho de um modelo ao prever os valores reais. Medidas diferentes possuem significado dos valores e intervalo de imagem diferentes.

Algumas dessas métricas são:

- Raiz do Erro Quadrático Médio (REQM): dado que Y é o valor real obtido de um problema e $\hat{Y}$ é o valor estimado, sua equação é dada por $\sqrt{\frac{\sum_{i=1}^{N}(Y_i-\hat{Y}_i)^2}{N}}$, para uma quantidade de N valores.
- Média Percentual Absoluta do Erro (MPAE): seguindo a mesma notação, sua equação é dada por $\frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i-\hat{Y}_i|}{|Y_i|}$.
- Erro Absoluto Médio (EAM): sua equação é dada por $\frac{1}{N}\sum_{i=1}^{N} Y_i - \hat{Y}_i$.
- Acurácia: também conhecida como taxa de acerto, mede a proporção de exemplos classificados corretamente em relação ao total de exemplos. Considera-se verdadeiro positivo (VP) a quantidade de vezes em que a RNA previu corretamente que o IBOVESPA subiria do dia corrente para o próximo dia, e verdadeiro negativo (VN) a quantidade de vezes em que a RNA previu corretamente que o IBOVESPA cairia, a equação da acurácia é dada por $\frac{VP+VN}{N}$.
- Precisão: consiste na proporção entre as previsões positivas corretas e o total de previsões positivas. Sendo falso positivo (FP) a quantidade de vezes em que a RNA previu erroneamente uma alta no IBOVESPA, sua equação é dada por $\frac{VP}{VP+FP}$.

Nos trabalhos estudados em 2.4 percebeu-se o uso de várias medidas de avaliação diferentes, algumas sendo mais utilizadas do que outras, porém não havendo um padrão no estado da arte. Para o *software* desenvolvido foi então definido o uso de MPAE e a acurácia como métricas, por serem uma das medidas de desempenho mais utilizadas nos trabalhos estudados, e no caso do MPAE também devido à sua representação usual do erro obtido em forma de porcentagem, o que inibe problemas com proporcionalidades.

Como o objetivo deste trabalho é a previsão do crescimento ou decrescimento do IBOVESPA no dia seguinte, os valores de saída da RNA são interpretados de maneira discreta como classes. Por isso, para o cálculo da acurácia, os valores de saída da RNA são arredondados. Caso o número seja positivo, arredonda-se para 1, caso contrário, para -1. Como consequência disto, existe a possibilidade de tanto MPAE quanto acurácia possuírem valores mutuamente altos.

3.3 Implementação da RNA

A implementação da RNA foi baseada em [25], seguindo as fórmulas e recomendações de implementação deste autor. A arquitetura escolhida para a rede foi multi-camadas, a mais comum para resolver problemas envolvendo séries temporais, conforme visto na seção 2.4.

A modelagem da RNA teve início com base em cada componente como neurônios, conexões, camadas, funções de ativação, o algoritmo supervisor e a rede como um todo. Já na modelagem dos neurônios e da conexão foi percebido que para permitir o comportamento desejado de grafo da RNA, as conexões deveriam se comportar como as arestas do grafo, sempre ligando um neurônio à outro em ambos os sentidos. Entre dois neurônios de camadas subsequentes, existe uma conexão.

A inicialização das sinapses é feita a partir de uma distribuição normal com média igual a zero e desvio padrão variando conforme a quantidade de conexões de entrada da última camada, como visto em [25].

O algoritmo 2 traz a implementação do algoritmo de treinamento (pseudocódigo do algoritmo 1). Nela, as instruções *do-while* representam as linhas 6 e 28 do pseudocódigo. Em seguida, todas as instruções do *while* mais interno representam as instruções da linha 7 até a linha 26 do pseudocódigo. Mais especificamente, a função $calcularAj()$ pega o exemplo da base de treinamento e insere na camada de entrada. Em seguida, os valores são propagados para frente, até chegar na camada de saída.

O algoritmo 3 mostra a implementação do gradiente local em um neurônio que se situa na camada de saída, conforme a fórmula 11. O gradiente local é o termo que aponta para as modificações necessárias nos pesos sinápticos e é fortemente influenciado pela diferença entre os pesos das conexões Δw_{ji}.

Algoritmo 2 Implementação em Java do algoritmo de retropropagação (Algoritmo 1)

```
public void treinar() {
   epocas = 0;
   Roleta roleta = new
       Roleta(entrada.size());
   int numeroSorteado;
   inicializarWij();
   do {
      energiaMedia = 0.0;
      energiaInstantanea
            = new double[entrada.size()];
      while (!roleta.isVetorCheio()) {
         numeroSorteado =
             roleta.encontrarProximoValor();
         calcularAj(numeroSorteado);
         propagarDeltas(numeroSorteado);
         atualizarWij();
         for (Neuronio neuronio :
               rna.getUltimaCamada()
                  .getListaNeuronios()) {
            energiaInstantanea[numeroSorteado]
               += neuronio.getEj() *
```

```
                    neuronio.getEj();
            }
            energiaInstantanea[numeroSorteado]
                *= 0.5;
            energiaInstantanea[numeroSorteado]);
        }
        roleta.limparRoleta();
        epocas++;
        for (int i = 0; i <
            energiaInstantanea.length; i++) {
            energiaMedia +=
                energiaInstantanea[i];
        }
        energiaMedia /= entrada.size();
} while (!verificarParada(epocas,
    energiaMedia));
```

Algoritmo 3 Implementação em Java do cálculo do gradiente local na camada de saída

```
for (Neuronio neuronio : neuroniosSaida) {
    int indiceNeuronio =
        neuroniosSaida.indexOf(neuronio);
    dj =
        saida.get(indice).get(indiceNeuronio);
    aj = neuronio.getAj();
    erro = dj - aj;
    neuronio.setEj(erro);
    gLinha = neuronio.getGLinha();
    gradienteLocal = erro * gLinha;
    neuronio.setGradienteLocal(gradienteLocal);
    ajusteEta = 1.0 /
        Math.sqrt(neuronio.getTotalConexaoEntrada
    Delta = eta * ajusteEta *
        neuronio.getGradienteLocal() *
        constanteBias;
    if
        (Double.isFinite(neuronio.getDeltaB()))
        {
       Delta += alpha * neuronio.getDeltaB();
    }
    neuronio.setDeltaB(Delta);
    for (Conexao conexao :
        neuronio.getConexoesEntrada()) {
       Delta = eta * ajusteEta *
           neuronio.getGradienteLocal() *
           conexao.getNeuronioEntrada().getAj();
       if (Double.isFinite(conexao
          .getDeltaWij())) {
          Delta += alpha *
              conexao.getDeltaWij();
       }
       conexao.setDeltaWij(Delta);
    }
}
```

Algoritmo 4 Implementação em Java do cálculo do gradiente local nas camadas intermediárias

```
for (int i = rna.getListaCamadas().size() -
    2; i > 0; i--) {
    for (Neuronio neuronio :
        rna.getCamada(i).getListaNeuronios())
        {
       gLinha = neuronio.getGLinha();
       somatorio = 0.0;
       for (Conexao conexao :
           neuronio.getConexoesSaida()) {
          somatorio +=
              conexao.getNeuronioSaida()
             .getGradienteLocal() *
                 conexao.getWij();
       }
      gradienteLocal = gLinha * somatorio;
       neuronio.setGradienteLocal(gradienteLocal);
       ajusteEta = 1.0 / Math.sqrt(neuronio
          .getTotalConexaoEntrada());
       Delta = eta * ajusteEta *
           neuronio.getGradienteLocal() *
           constanteBias;
       if
           (Double.isFinite(neuronio.getDeltaB()))
           {
          Delta += alpha *
              neuronio.getDeltaB();
       }
       neuronio.setDeltaB(Delta);
       for (Conexao conexao :
           neuronio.getConexoesEntrada()) {
          Delta = eta * ajusteEta *
              neuronio.getGradienteLocal()
             * conexao.getNeuronioEntrada()
             .getAj();
          if (Double.isFinite(conexao
             .getDeltaWij())) {
             Delta += alpha *
                 conexao.getDeltaWij();
          }
          conexao.setDeltaWij(Delta);
       }
    }
}
```

Já o algoritmo 4 traz a implementação que representa a equação 12, que é a fórmula da retropropagação para o gradiente local em neurônios que se situam nas camadas intermediárias. Ambas as figuras são implementações internas da função *propagarDeltas*() para um determinado exemplo sorteado da base de treinamento, que consta na figura 3.3 e que implementa as linhas 16 a 23 do pseudocódigo 1.

Nos dois trechos de código é possível ver o uso da variável *ajusteEta*. Isto advém da recomendação de [25], de que os neurônios precisam aprender por igual em uma mesma velocidade. Como o ajuste do peso das conexões depende do somatório das entradas dos neurônios, neurônios da camada de saída e também neurônios com muitas conexões de entrada tendem a aprender mais rapidamente que os outros. Por isso, um ajuste é realizado, em cada neurônio individualmente, para desacelerar o aprendizado destes neurônios, dado pela

equação

$$AjusteEta = \frac{1}{\sqrt{TotalConexoesEntrada}} \quad (15)$$

As funções de ativação utilizadas pelos neurônios das camadas intermediárias e saída, foram implementadas conforme descrito nas equações 8, 9 e 10 e utilizadas nas implementações das figuras 3.3 e 3.3. As várias opções para a função de ativação, como pode ser visto em emuladores de RNA, contribuíram para que a RNA construída possua uma maior flexibilidade no que diz respeito às áreas de aplicação.

Dentre as funções implementadas, duas foram selecionadas para realizar o experimento, sendo elas a Sigmoidal Simétrica e a Tangente Hiperbólica Sigmoidal. Esta escolha deu-se por dois motivos: pela necessidade do possível intervalo do valor de saída da RNA estar contido no intervalo da função de ativação escolhida, como recomendado em [25], e pela análise de resultados provindos de testes preliminares realizados.

Com a RNA implementada foram realizados alguns testes preliminares e um experimento com a base de dados conforme detalhado na seção 3.1. Os resultados, que serão discutidos na seção 4, possibilitaram uma avaliação da implementação realizada e permitiram com que fosse possível iniciar a etapa seguinte do trabalho com o objetivo de melhorá-los através de hibridizações da RNA.

3.4 Implementação de AG

A modelagem de AG foi separada em uma estrutura genérica que pode ser reaproveitada e uma específica para cada AG específico escolhido para a hibridização da RNA. A partir desta implementação, somente as características específicas de cada AG devem ser construídas, enquanto as operações genéticas comuns são reutilizadas.

O cromossomo, independente das estratégias estudadas, será constituído de uma sequência de números, que podem representar a arquitetura, os pesos e até as melhores entradas para a RNA. Além disso, um cromossomo também sempre precisa ser avaliado pelo AG para que seja possível medir seu grau de adaptabilidade e quão boa é a solução proposta por ele.

A seleção é uma operação genética que deve ser realizada em AG e não é afetada pela especificidade do algoritmo dentro do escopo deste estudo. Nesta etapa, alguns cromossomos são selecionados para participarem, direta ou indiretamente, da nova população de indivíduos. Neste trabalho, duas formas de seleção são utilizadas mutuamente: o elitismo e a roleta ponderada.

Para a seleção dos cromossomos, inicialmente é realizado o cálculo da adaptabilidade de cada cromossomo, como pode ser visto no algoritmo 5. Calcula-se o erro MPAE obtido pela RNA com a base de teste, e em seguida a adaptabilidade do cromossomo é obtida através do inverso do valor MPAE, $1 - MPAE$. Em seguida, o conjunto de cromossomos é ordenado pela a adaptabilidade de maneira crescente.

Para o método do elitismo, basta copiar os n primeiros cromossomos do conjunto e guardá-los para a próxima geração. Esta técnica garante que a próxima população contenha alguns dos melhores cromossomos da geração atual enquanto ainda realiza operações genéticas para buscar novos cromossomos com uma melhor adaptabilidade.

Para o método da roleta ponderada, pares de indivíduos devem ser sorteados, porém aqueles com maior adaptabilidade devem ser selecionados com maior frequência para o cruzamento. O algoritmo 5 apresenta a implementação da seleção de dois cromossomos. Isto é feito simulando uma função de probabilidade.

Algoritmo 5 Implementação em Java da seleção do AG Supervisor

```
for (int i = 0; i < cromossomos.size();
    i++) {
   pontuacoes.add(1 -
       cromossomos.get(i).getPontuacao());
}
pontuacoes = tratamento
   .getNormalizacaoLinearDobrada(pontuacoes);
for (int i = 0; i < pontuacoes.size(); i++)
    {
   pontuacaoTotal += (pontuacoes.get(i));
}
while (totalEscolhidos != 2) {
   somaProbabilidade = pontuacaoTotal;
   valorAleatorio = (Math.random() *
       pontuacaoTotal);
   for (int i = 0; i < pontuacoes.size();
       i++) {
      somaProbabilidade -=
          pontuacoes.get(i);
      if ((valorAleatorio >
          somaProbabilidade) && i !=
          indiceJaEscolhido) {
         sorteados.add(cromossomos.get(i));
         indiceJaEscolhido = i;
         totalEscolhidos++;
         break;
      }
   }
}
```

Inicialmente é realizado o somatório das adaptabilidades de todos os cromossomos. Depois gera-se um valor aleatório proporcional à soma obtida, implementado na figura 3.4 pela variável *valorAleatorio*. Em seguida, iterando cada cromossomo, realiza-se a seguinte comparação: caso *valorAleatorio* for maior que a soma das adaptabilidades decrescida da adaptabilidade do i-ésimo cromossomo, então ele é selecionado; caso contrário, continua-se a iteração.

Para que a roleta ponderada funcione, é importante que haja uma discrepância significativa entre as adaptabilidades dos cromossomos. Em testes realizados preliminarmente, foi observado que os valores de adaptabilidade dos cromossomos foram bastante próximos. Por isso, os valores adaptabilidade

foram ajustados com uma função linear, dada pela equação

$$SerieNormalizada_i = \frac{Serie_i - Minimo(Serie)}{Maximo(Serie) - Minimo(Serie)} \quad (16)$$

O cruzamento é feito após a etapa de seleção. Para cada par de cromossomos selecionados pela roleta ponderada, um número aleatório é gerado. Caso o número seja menor que um valor de probabilidade de cruzamento definido pelo usuário, então o cruzamento é realizado entre este par de cromossomos e os dois filhos gerados são guardados para a próxima geração. Caso contrário, o próprio par de cromossomos então é guardado para a próxima geração.

Quatro formas de cruzamento foram implementadas para representar problemas distintos com maior flexibilidade de forma similar às funções de ativação da RNA. São elas:

- Corte em um ponto: conforme já descrito na seção 2.3.4.
- Corte em dois pontos: funciona de maneira similar à técnica anterior, porém dois pontos de corte diferentes são sorteados.
- Máscara binária aleatória: para cada n-ésimo gene do cromossomo, um número aleatório é sorteado. Caso o número seja 1, então o cromossomo filho F_1 recebe em seu n-ésimo gene o valor do n-ésimo gene do cromossomo pai P_1, enquanto que o cromossomo filho F_2 recebe o n-ésimo gene do cromossomo pai P_2. Caso o número seja 0, o inverso ocorre. O cromossomo filho F_1 recebe o n-ésimo gene do cromossomo pai P_2 e F_2 recebe o n-ésimo gene do cromossomo pai P_1.
- Máscara binária alternada: similar à máscara binária, porém aqui os valores da máscara não são sorteados, mas sim pré-determinados, alternando seus valores entre 0 e 1.

As etapas de mutação e decodificação não são iguais para todos os AG, mudando conforme suas peculiaridades, enquanto todas as etapas descritas anteriormente são similares para as outras técnicas de AG estudadas. As particularidades destas etapas para o algoritmo desenvolvido são explicadas na seção 3.4.1.

3.4.1 Substituição da Retropropagação

Neste modelo de RNA híbrida, o AG foi utilizado para treinar a RNA, substituindo então o algoritmo de retropropagação explicado na seção 2.2.

Para a inicialização dos cromossomos, são gerados valores aleatórios entre -1 e 1 seguindo uma função normal de distribuição de probabilidade com média 0.

Em seguida, é realizada a decodificação, conforme pode ser visto na figura 9. Nesta etapa, os genes do cromossomo são copiados para a RNA. Cada gene representa um peso $w_{j,i}$ das conexões de saída dos neurônios das camadas de entrada e intermediária, ou um *bias* (conexão de entrada individual) de cada neurônio das camadas intermediária e de saída.

O algoritmo 6 traz a implementação da decodificação do cromossomo.

Algoritmo 6 Implementação em Java da decodificação do AG

```
public RNA
    decodificarPesosEBiasParaRNA(Cromossomo
    cromossomo, RNA rna) {
   int indice = 0;
   for (int i = 0; i <
       rna.getNumeroCamadas() - 1; i++) {
      Camada camada = rna.getCamada(i);
      for (Neuronio neuronio :
          camada.getListaNeuronios()) {
         if (i >= 1) {
            neuronio.setB(cromossomo.get(indice));
            indice++;
         }
         for (Conexao conexao :
             neuronio.getConexoesSaida()) {
            conexao.setWij(cromossomo.get(indice));
            indice++;
         }
      }
   }
   for (Neuronio neuronio :
       rna.getUltimaCamada().getListaNeuronios())
       {
      neuronio.setB(cromossomo.get(indice));
      indice++;
   }
   return rna;
}
```

Na operação de cruzamento, optou-se pelo uso de dois dos quatro métodos implementados descritos em 3.4. Devido ao tamanho cromossomo dessa abordagem genética crescer rapidamente com arquiteturas mais complexas, foram escolhidas as estratégias de cruzamento que melhor dividem o cromossomo: o corte em dois pontos e a máscara binária aleatória.

Também foi identificado que, devido à grande quantidade de valores contidos em um único cromossomo, uma vez que cada valor representa um peso de uma conexão incluindo os *bias*, a mutação em apenas um valor é uma mudança muito pequena em relação ao cromossomo inteiro, especialmente em RNA com muitas conexões. Para contornar este problema, a mutação foi implementada para ocorrer em um número variável de genes do cromossomo, seja por substituição ou incremento. Esta técnica trouxe melhores resultados em testes preliminares por causa da maior variedade genética coberta pelo AG ao considerar mais mutações.

Este modelo desenvolvido de RNA híbrida é similar ao trabalho de [35], porém a mutação utilizada neste trabalho realiza a substituição do valor do gene por um novo valor gerado aleatoriamente. As outras etapas do AG ocorrem nor-

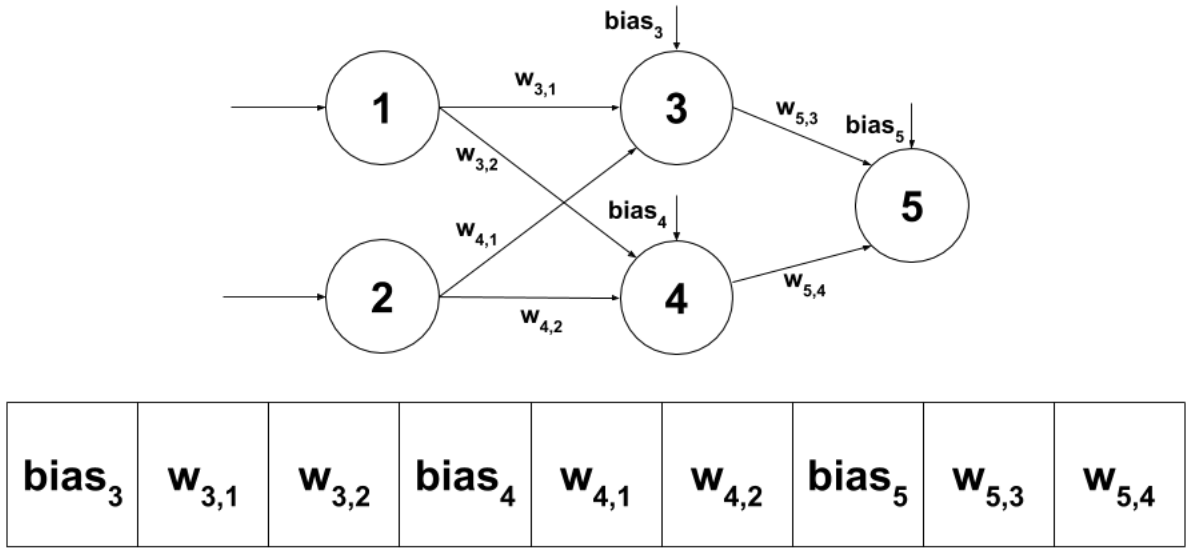


Figura 9. Estrutura do cromossomo e sua relação com os parâmetros da RNA

malmente, conforme já descrito na seção 2.3.

4. Resultados e Discussão

Para avaliar o desempenho obtido pelos algoritmos implementados, foram realizados experimentos para determinar qual combinação dos fatores resulta no melhor desempenho, medido através do MPAE e da taxa de acerto. Como cada algoritmo comparado é dependente de inicialização de variáveis com valores aleatórios, cada teste foi realizado inicialmente com 10 repetições.

Primeiramente foi realizado um teste com a RNA pura, pois esta utiliza um grande número de fatores que também são utilizados nos outros modelos. Por isso, uma vez que a melhor combinação destes fatores é definida, a quantidade de fatores restantes para os testes subsequentes diminui, reduzindo a complexidade dos testes ao focar somente nos fatores que são específicos de cada AG, sendo que um teste específico foi realizado para cada modelo.

Dos fatores que são mais genéricos e que estão presentes em todos os modelos implementados, foram identificados dois tipos: os fatores relacionados ao mercado de ações; e os fatores relacionados à RNA e à retropropagação.

4.1 Resultados da RNA Pura

Os fatores do mercado de ações influenciam a forma com que os valores são apresentados à RNA, são eles: a quantidade de dias que a RNA recebe para cada índice, representado por *Dias*; e o tamanho da base de dados utilizada para treinamento e teste, já citados anteriormente como X_1 e X_2, representado por *Anos*.

Os fatores da RNA são configurações da própria rede ou parâmetros do algoritmo de retropropagação. Estes fatores são: a quantidade máxima de iterações permitidas na etapa de treinamento, representado por *MaxIteracoes*; a quantidade de neurônios na camada intermediária, representado por *qtdNeuronios*; a função de ativação de cada neurônio, representado por fx; os valores de α e η da retropropagação, representados por $Alpha$; e os valores de a e b da função de ativação, também representados por a e b.

Como há um total de oito fatores, um teste do tipo fatorial completo de múltiplos níveis tornou-se inviável devido à quantidade de combinações a serem avaliadas e também ao longo tempo para a execução dos testes. Por isso, o método de avaliação tomado aqui foi o fatorial completo 2^k. Os valores escolhidos para os fatores foram os seguintes:

- *Dias* - a RNA recebeu 1 ou 5 dias de entrada.
- *Anos* - a RNA foi treinada com 3 ou 12 anos.
- *MaxIteracoes* - foi permitido a retropropagação executar 1000 ou 10000 épocas.
- *qtdNeuronios* - 25 ou 75 neurônios na camada intermediária.
- fx - o valor 0 representa a função de ativação Sigmoidal Simétrica, enquanto que o valor 1 representa a Tangente Hiperbólica Sigmoidal.
- $Alpha$ - a fim de reduzir a quantidade de fatores, os valores α e η foram variados aos pares. $Alpha$ 0.5 significa que $\alpha = 0.5$ e $\eta = 0.1$, enquanto que $Alpha$ 0.7 significa que $\alpha = 0.7$ e $\eta = 0.2$.
- a e b - variaram respectivamente entre 1 e 1.7159 e $\frac{2}{3}$ e 1.

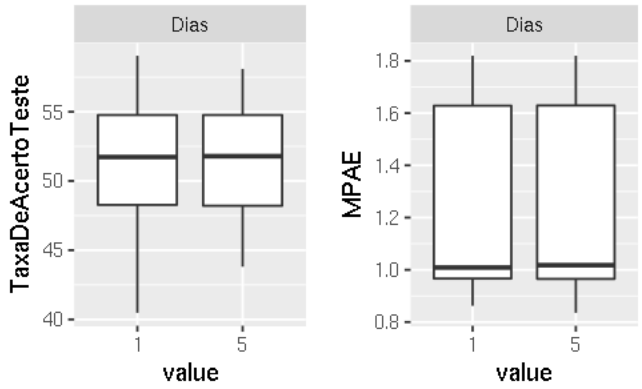
Como variáveis de saída utilizadas para medir a qualidade dos resultados obtidos, foram utilizados a Taxa de Acerto e o MPAE obtidos com a base de teste.

Após a execução do algoritmos com todas as combinações dos fatores, realizou-se um estudo sobre a análise de

variâncias, utilizando a linguagem R, dos resultados obtidos nos testes. As análises de variâncias de ambas as variáveis de saída podem ser vistas nas tabelas 6 e 7.

Como pode ser visto em ambas as tabelas, o cálculo de *TesteF* revelou valores significativos. Conforme a análise de variâncias, ao comparar os valores obtidos com os valores da tabela de distribuição F de Snedecor, foi demonstrado que os valores de *TesteF* possuem 5% de significância, para todos os fatores. Por isso, um estudo mais aprofundado foi realizado sobre cada fator, analisando o desempenho da RNA em Taxa de Acerto e MPAE ao variar os valores de cada fator individualmente. Tal estudo foi realizado a partir dos gráficos do tipo *boxplot* das figuras 10 e 11.

Como pode ser visto nas figuras 10a e 10b, os valores de média e de primeiro desvio padrão, tanto para a métrica de erro MPAE quanto para a Taxa de Acerto, são muito parecidas. Porém, a faixa de valores que estão dois desvios padrão abaixo da média são mais altos para a Taxa de Acerto com 5 Dias, além de que para dois desvios padrão acima da média, 5 Dias possui valores levemente menores. Por isso, para os testes seguintes, o valor de 5 Dias foi definido como padrão.



(a) Dias - Taxa de Acerto **(b)** Dias - MPAE

Figura 10. Desempenho dos testes variando o fator Dias

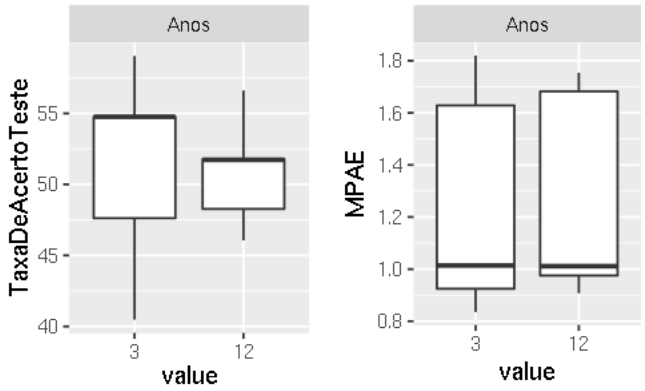
Como também pode ser visto nas figuras 11a e 11b, para o fator *Anos* na métrica MPAE, embora os dois valores possuam médias muito próximas, o valor de 3 Anos possui visivelmente valores menores para um desvio padrão, tanto abaixo quanto acima da méda. Já na Taxa de Acerto, a média para 3 Anos é visivelmente superior. Por isso, o valor de 3 Anos foi definido como padrão para os próximos testes.

Tabela 6. Análise de Variâncias - Taxa de Acerto

FonteDeVariacao	SomaDeQuadrados	GrausDeLiberdade	QuadradosMedios	TesteF
Dias	69.6	1	69.6	7.66
Anos	1337.7	1	1337.7	147.18
MaxIteracoes	164.5	1	164.5	18.095
qtdNeuronios	0.3	1	0.3	0.037
fx	443.5	1	443.5	48.8
Alpha	314.1	1	314.1	34.565
a	559.1	1	559.1	61.517
b	121.2	1	121.2	13.335
Residual	20940	2304	9.1	NA
Total	23950	2312	3019.1	NA

Tabela 7. Análise de Variâncias - MPAE

FonteDeVariacao	SomaDeQuadrados	GrausDeLiberdade	QuadradosMedios	TesteF
Dias	0.86	1	0.86	140.1
Anos	0.13	1	0.13	21.21
MaxIteracoes	0.9	1	0.9	146.82
qtdNeuronios	0.47	1	0.47	77.17
fx	26.88	1	26.88	4403.88
Alpha	14.56	1	14.56	2385.52
a	147.09	1	147.09	24101.09
b	6.21	1	6.21	1017.65
Residual	14.06	2304	0.1	NA
Total	211.16	2312	197.2	NA



(a) Anos - Taxa de Acerto **(b)** Anos - MPAE

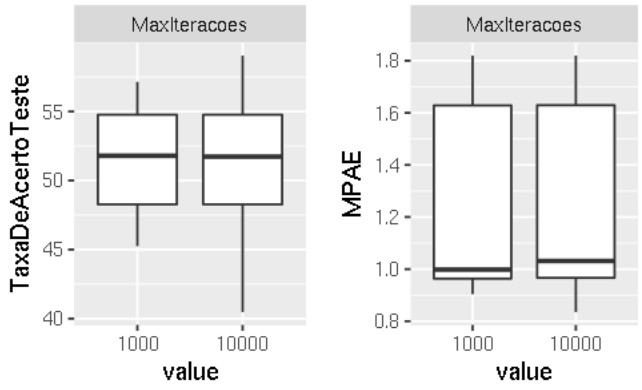
Figura 11. Desempenho dos testes variando o fator Anos

Nas figuras 12a e 12b é possível ver que no fator *MaxIteracoes*, os valores de primeiro desvio padrão acima e abaixo da média foram muito semelhantes, tanto em MPAE quanto em Taxa de Acerto. Já em MPAE é possível ver que 1000 iterações possui uma média levemente menor. Na Taxa de Acerto, 10000 iterações alcançaram valores maiores no segundo desvio padrão acima da média, porém alcançou valores menores em dois desvios padrão abaixo da média, o que demonstra que com esse valor o desempenho da RNA é menos estável. Por isso, 1000 iterações como limite de treinamento foi definido como padrão nos próximos testes.

Para o fator fx, as figuras 13a e 13b revelam uma grande diferença entre as duas funções de ativação no MPAE, com a Sigmoidal Simétrica possuindo média e um desvio padrão acima da média muito menores do que a Tangente Hiperbólica Sigmoidal. Já na Taxa de Acerto, ambas as funções possuem média e um desvio padrão muito próximas. Por isso, a função Sigmoidal Simétrica foi definida como padrão para os testes seguintes.

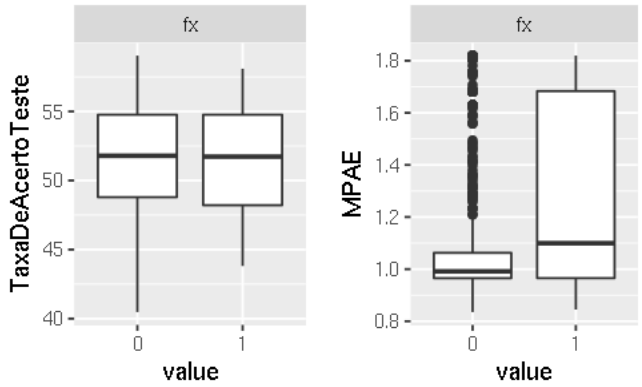
O fator *qtdNeuronios* obteve resultados muito similares ao fator *Dias* e os testes demonstraram que em MPAE os dois valores são muito semelhantes, porém na Taxa de Acerto, em dois desvios padrão, 25 neurônios intermediários demonstraram valores mais altos, definindo então 25 neurônios na camada intermediária como um valor padrão para os testes seguintes.

Os fatores $Alpha$, a e b obtiveram resultados muito se-



(a) MaxIteracoes - Taxa de Acerto **(b)** MaxIteracoes - MPAE

Figura 12. Desempenho dos testes variando o fator MaxIteracoes



(a) fx - Taxa de Acerto **(b)** fx - MPAE

Figura 13. Desempenho dos testes variando o fator fx

melhantes aos já descritos no fator fx das figuras 13a e 13b. Por isso, para os testes seguintes, foi definido como padrão os valores de 0.5 para $Alpha$, 1 para a e $\frac{2}{3}$ para b.

A partir dos resultados obtidos neste experimento foi feita uma avaliação da taxa de acerto, e o melhor resultado teve as seguintes configurações: 1 *Dias*, 3 *Anos*, 10000 *MaxIteracoes*, 25 neurônios intermediários, função de ativação Sigmoidal Simétrica $Alpha$ de 0.5, a 1 e b $\frac{2}{3}$. Uma avaliação do MPAE também foi realizada, e a melhor configuração neste caso foi a de: 5 *Dias*, 3 *Anos*, 10000 *MaxIteracoes*, 75 neurônios intermediários, função de ativação Sigmoidal Simétrica, $Alpha$ 0.5, a 1 e b 1.

Estas configurações diferem em partes da configuração definida anteriormente, após a análise dos fatores individualmente, como sendo a melhor. Isto deve-se ao fato de estes casos pertencerem ao conjunto referente ao segundo desvio padrão ou de *outliers*, conforme já observado nas figuras 10, 11, 12 e 13.

Para confirmar tal hipótese, um segundo teste foi realizado. Desta vez, apenas com três configurações: a configuração que apresentou o melhor MPAE, definida como R_1; a configuração

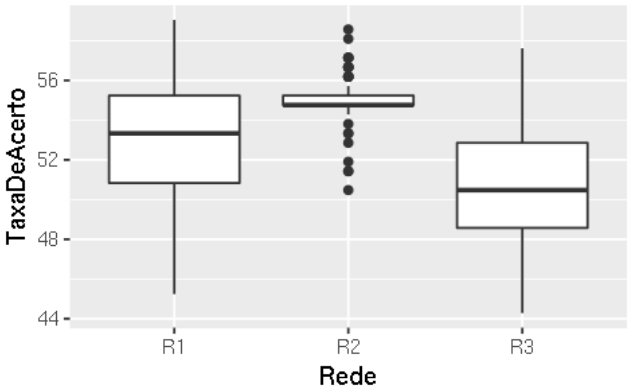


Figura 14. Taxa de Acerto obtida pelas três configurações no segundo teste de RNA pura

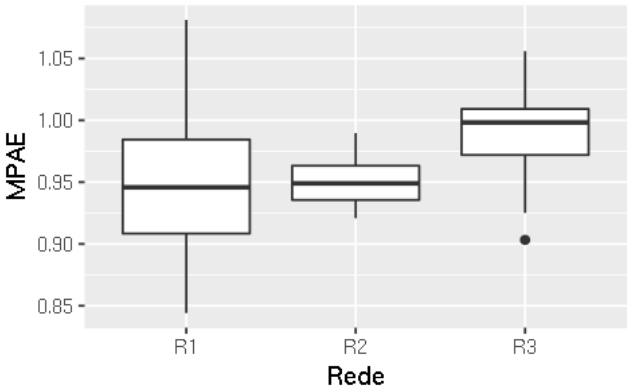


Figura 15. MPAE obtido pelas três configurações no segundo teste de RNA pura

definida neste trabalho como sendo a melhor, definida como R_2; e a configuração que apresentou o melhor desempenho de Taxa de Acerto, definida como R_3. Neste segundo teste com as RNA puras foram realizadas 100 repetições, para que haja uma menor variabilidade nos primeiros desvios padrão e para que os valores *outliers* tenham uma menor influência nos resultados obtidos.

Como pode ser visto na figura 14, a configuração R_2 obteve a maior média, além de possuir primeiro e segundo desvios padrão menores do que as outras configurações, revelando que a rede R_2 possui uma Taxa de Acerto mais alta e de maneira mais concisa. Já na figura 15, pode-se perceber que a rede R_2 possui o segundo menor MPAE médio, levemente maior do que de R_1. Porém, mais uma vez R_2 possui primeiro e segundo desvios padrão visivelmente menores do que de R_1 e R_3, por possuir resultados mais consistentes. Por isso, a configuração R_2 foi mantida como sendo a melhor configuração de fatores de RNA e de mercado e foi mantida (na medida do possível) para os testes seguintes.

Reforçando, a melhor rede é a configuração R_2, que na média das 100 replicações alcançou uma Taxa de Acerto média aproximada de 55.04% e 1.19 de desvio padrão, um MPAE médio aproximado de 0.95 e um desvio padrão

de 0.018, e possui as seguintes configurações: 5 *Dias*, 3 *Anos*, 1000 *MaxIteracoes*, fx Sigmoidal Simétrica, 25 $qtdNeuronios$, $Alpha$ 0.5, a 1 e b $\frac{2}{3}$.

4.2 Resultados da RNA Treinada pelo AG

Para avaliar o desempenho da RNA treinada pelo AG, foram testados apenas os fatores relacionados ao AG. Os fatores de mercado e os fatores de RNA utilizados neste experimento já foram definidos como sendo os de R_2. Assim, o modelo foi testado com 5 dias de índices como entrada, uma base de dados de apenas 3 anos, 25 neurônios na camada intermediária da RNA, função de ativação Sigmoidal Simétrica, a igual a 1 e b igual a $\frac{2}{3}$.

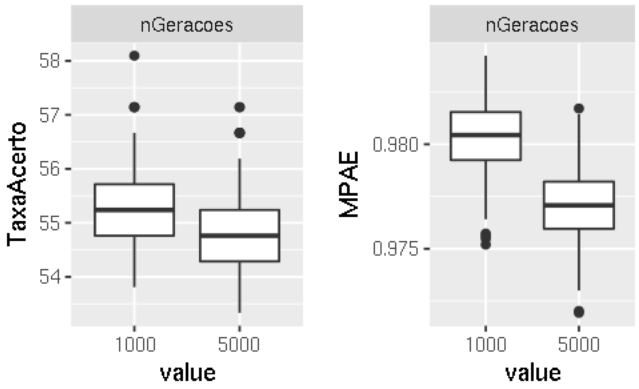
Aqui, também foi realizado um teste do tipo fatorial completo 2^k. Os fatores avaliados neste método híbrido são descritos a seguir:

- *nGeracoes*: o treinamento da RNA pelo AG foi limitado em 1000 gerações ou 5000 gerações.
- *TipoCruzamento*: como a RNA possui ligação entre todos os neurônios da camada n para a camada $n+1$, e como foi definido pelo teste anterior de que a RNA deveria possuir 25 neurônios na camada oculta (representado por X) e 5 dias de índices como entrada (totalizando 45 neurônios de entrada, representados por Y), além de que a rede possui 1 neurônio de saída e que todos os neurônios ocultos e de saída possuem o *bias*, o tamanho do cromossomo é definido por: $X*Y+Y+Y*1+1$, totalizando 1176 genes. Com um cromossomo tão grande, um único ponto de corte pode não trazer uma variabilidade suficiente nos cromossomos da nova geração. Por isso, definiu-se que o cruzamento dos cromossomos pode ser realizado por meio de corte de dois pontos, representado por $CORTE\_2$, ou pela máscara binária aleatória, representado por $MASC\_BIN$.
- *TaxaCruzamento*: a quantidade de pares que realizaram o cruzamento, sendo de 60% ou de 80%.
- *TaxaMutacao*: a probabilidade de um cromossomo sofrer mutação, sendo de 10% ou de 20%.
- *PctGenesMutacao*: conforme já descrito anteriormente, o cromossomo neste AG possui 1176 genes. Por isso uma mutação tradicional, em um único gene, também não traz uma variabilidade suficiente para os cromossomos da próxima geração. Tendo isto em vista, este fator determina a quantidade de genes que sofrerão a mutação, caso sofram, sendo de 1% ou de 3% do total de genes.
- *NumeroDeCromossomos*: o tamanho da população foi definido como 60 ou 100 cromossomos.

Na tabela 8 pode ser visto que o *TesteF* resultou em valores significativos, e a análise de variâncias demonstrou que os fatores *nGeracoes*, *TaxaMutacao* e *NumeroDeCromossomos* obtiveram 0.1% de significância, *PctGenesMutacao* obteve 1% de significância e os fatores *TipoCruzamento* e *TaxaCruzamento* não foram significantes para a variável Taxa de Acerto.

Já na tabela 9 também pode ser visto que o *TesteF* resultou em valores significativos para a variável MPAE, e a análise de variâncias demonstrou que os fatores obtiveram 0.1% de significância, com exceção do fator *TipoCruzamento* que não foi significante.

Como pode ser visto nas figuras 16a e 16b, para o fator *nGeracoes* percebe-se que o valor de 1000 gerações resultou em um média maior do que de 5000 gerações, além de que os valores de primeiro e segundo desvios padrão proporcionarem valores maiores tanto para a Taxa de Acerto quanto para MPAE. Isso deve-se ao fato de a saída da RNA ser considerada como valores discretos ao calcular a Taxa de Acerto, conforme já descrito na seção 3.2. Como o objetivo do estudo é obter a maior Taxa de Acerto, optou-se por 1000 gerações como sendo o valor padrão para testes futuros.



(a) nGeracoes - Taxa de Acerto **(b)** nGeracoes - MPAE

Figura 16. Desempenho dos testes variando o fator nGeracoes

Como *TipoCruzamento* e *TaxaCruzamento* não foram significantes nos testes, optou-se por definir como padrão para testes futuros os valores de $CORTE\_2$ para *TipoCruzamento* e 0.6 de *TaxaCruzamento*, pois estes acarretam em um menor custo computacional, uma vez que com $CORTE\_2$ é necessário o sorteio de apenas 2 números ao contrário da $MASC\_BIN$ que exige um sorteio para cada gene do cromossomo, e que 0.6 proporciona menores processos de cruzamento.

Na figura 17a pode ser visto que para o fator *TaxaMutacao* o valor de 0.1 obteve média maior do que 0.2 na Taxa de Acerto. Além disso, principalmente o segundo desvio padrão abaixo da média para o valor 0.1 possui um tamanho menor, demonstrando valores mais estáveis para 0.1 nesta métrica de desempenho. Já para a métrica MPAE, pode ser visto na figura 17b que o valor de 0.1 possui uma média menor do que 0.2, primeiro desvio padrão acima da média igual mas abaixo da média menor, e um tamanho menor de

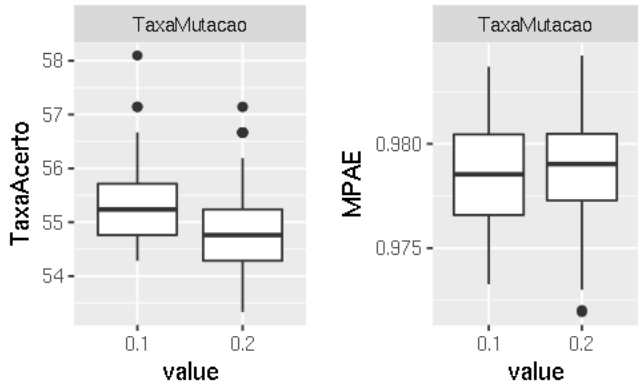
Tabela 8. Análise de Variâncias do AG Supervisor baseado na Taxa de Acerto

Fonte de Variação	Soma de Quadrados	Graus de Liberdade	Quadrados Médios	Teste-F
nGeracoes	36.51	1	36.51	92.101
TipoCruzamento	0.16	1	0.16	0.394
TaxaCruzamento	0.16	1	0.16	0.394
TaxaMutacao	28.38	1	28.38	71.586
PctGenesMutacao	2.81	1	2.81	7.08
NumeroDeCromossomos	5.54	1	5.54	13.966
Residual	228.32	576	0.4	NA
Total	301.88	582	73.96	NA

Tabela 9. Análise de Variâncias do AG Supervisor baseado no MPAE

Fonte de Variação	Soma de Quadrados	Graus de Liberdade	Quadrados Médios	Teste-F
nGeracoes	0.0017806	1	0.0017806	1593.61
TipoCruzamento	0	1	0	0.005
TaxaCruzamento	0.0002579	1	0.0002579	230.813
TaxaMutacao	1.47e-05	1	1.47e-05	13.177
PctGenesMutacao	0.0002488	1	0.0002488	222.678
NumeroDeCromossomos	0.0004609	1	0.0004609	412.465
Residual	0.0006436	576	1.1e-06	NA
Total	0.0034065	582	0.002764	NA

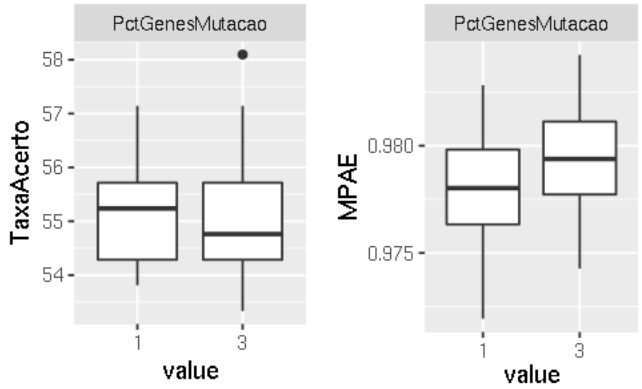
segundo desvio padrão abaixo e acima da média, mostrando assim resultados mais estáveis. Por isso, o valor de 0.1 para *TaxaMutacao* foi definido como sendo o melhor.



(a) TaxaMutacao - Taxa de Acerto **(b)** TaxaMutacao - MPAE

Figura 17. Desempenho dos testes variando o fator TaxaMutacao

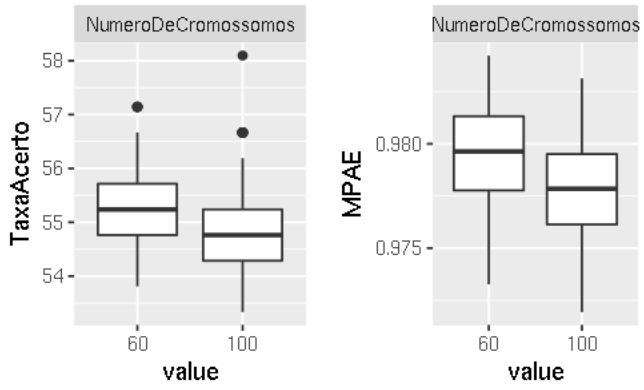
Na figura 18a pode ser visto que para o fator *PctGenesMutacao* o valor de 1% possui uma média visivelmente maior do que 5% em Taxa de Acerto, com primeiro desvio padrão igual, e um segundo desvio padrão abaixo da média visivelmente maior. Já para a métrica MPAE, o valor de 1% possui média, primeiro e segundo desvios padrão visivelmente menores. Por isso, 1% de *PctGenesMutacao* foi definido como o valor padrão.



(a) PctGenesMutacao - Taxa de Acerto **(b)** PctGenesMutacao - MPAE

Figura 18. Desempenho dos testes variando o fator PctGenesMutacao

Na figura 19a, pode ser visto que para o fator *NumeroDeCromossomos* na Taxa de Acerto, é visível uma média, primeiro e segundo desvios padrão maiores para 60 cromossomos. Já na métrica MPAE, pode ser visto o mesmo comportamento para o valor de 60 cromossomos. Mais uma vez a discretização dos valores de saída da RNA para o cálculo da Taxa de Acerto resultaram no comportamento de uma configuração obter maiores Taxa de Acerto e de MPAE, como já explicado em 3.2. Como o objetivo a se alcançar é uma



(a) NumeroDeCromossomos - Taxa de Acerto **(b)** NumeroDeCromossomos - MPAE

Figura 19. Desempenho dos testes variando o fator NumeroDeCromossomos

maior Taxa de Acerto, 100 cromossomos foi definido como valor padrão.

A partir dos resultados obtidos neste experimento foi feita uma avaliação da taxa de acerto, e o melhor resultado teve as seguintes configurações: 1000 *nGeracoes*, *CORTE_2* como *TipoCruzamento*, 0.6 de *TaxaCruzamento*, 0.1 de *TaxaMutacao*, 3% de *PctGenesMutacao* e 100 *NumeroDeCromossomos*. Uma avaliação do MPAE também foi realizada, e a melhor configuração neste caso foi a de: 5000 *nGeracoes*, *MASC_BIN* como *TipoCruzamento*, 0.8 de *TaxaCruzamento*, 0.2 de *TaxaMutacao*, 1% de *PctGenesMutacao* e 100 *NumeroDeCromossomos*.

Novamente, estas configurações diferem em partes da configuração definida neste trabalho como sendo a melhor, devido ao fato de que o resultado obtido por estas configurações situarem-se no segundo desvio padrão acima da média ou a *outliers*.

Para confirmar tal hipótese, um novo teste foi realizado, utilizando apenas três configurações: a configuração que apresentou o melhor MPAE, definida como AG_1; a configuração apontada neste trabalho como sendo a melhor, definida por AG_2; e a configuração que apresentou a melhor Taxa de Acerto, definida como AG_3. Neste segundo teste com os AG Supervisores foram realizadas 100 repetições, para que haja uma menor variabilidade nos primeiros desvios padrão e para que os valores *outliers* tenham uma menor influência nos resultados obtidos.

Como pode ser visto na figura 20, a configuração AG_2 obteve a maior média, além de que os primeiros e segundos desvios padrão obtiveram maiores valores de acerto, superando inclusive a configuração de AG_3, que no teste anterior obteve a maior taxa de acerto, confirmando então a hipótese de que os valores de Taxa de Acerto muito altos obtidos por AG_3 são provenientes do segundo desvio padrão ou de *outliers*.

Já na figura 21, desta vez a configuração que obteve o melhor MPAE no teste anterior, AG_1, realmente obteve a

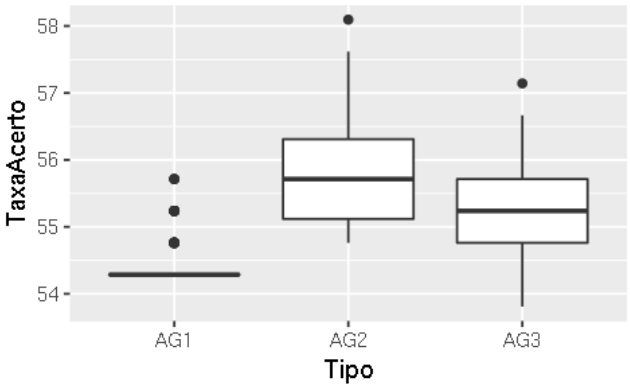


Figura 20. Taxa de Acerto obtida pelas três configurações de AG Supervisora no segundo teste

melhor média de MPAE. Porém, como pode ser visto, AG_1 obteve uma média pouco acima de 0.975, enquanto que AG_2 e AG_3 obtiveram médias abaixo de 0.9825, uma diferença proporcionalmente pequena. Além disso, como o objetivo deste trabalho é alcançar a melhor Taxa de Acerto, reitera-se que AG_2 é a melhor configuração de AG Supervisor.

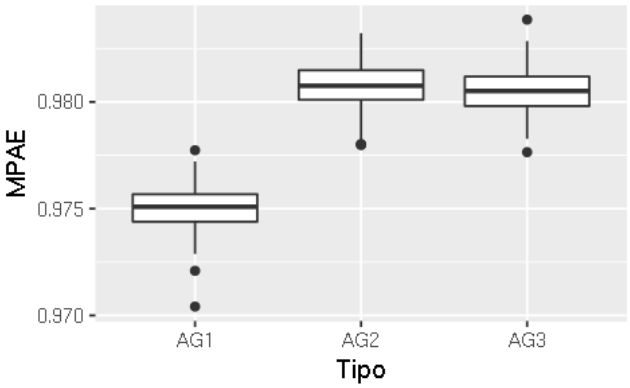


Figura 21. MPAE obtido pelas três configurações de AG Supervisora no segundo teste

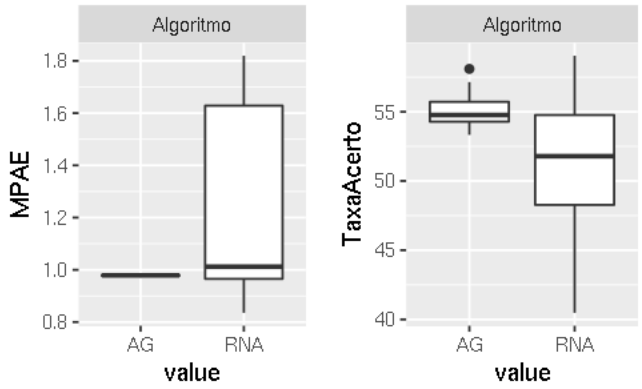
Reforçando, o melhor AG é a configuração AG_2, que na média das 100 replicações alcançou uma Taxa de Acerto média aproximada de 55.73% e um desvio padrão de 0.82, um MPAE médio aproximado de 0.98 com um desvio padrão de 0.001, e possui as seguintes configurações: 1000 *nGeracoes*, *CORTE_2* como *TipoCruzamento*, 0.6 de *TaxaCruzamento*, 0.1 de *TaxaMutacao*, 1% de *PctGenesMutacao* e 100 *NumeroDeCromossomos*.

4.3 Comparação entre os resultados obtidos

Para finalizar os experimentos, foram comparadas a melhor configuração de uma RNA pura, resultante dos experimentos da seção 4.1, com a melhor solução híbrida (RNA + AG), obtida nos experimentos da seção 4.2. Os resultados demonstram uma melhoria na taxa de acerto ao utilizar a configuração descrita como AG2 sobre a melhor configuração de RNA pura.

Para verificar estes resultados uma última análise de

variância foi feita confrontando os resultados dos dois diferentes modelos. Esta análise considera um único fator que é o algoritmo de treinamento da RNA com dois níveis: RNA pura (RNA) e modelo híbrido (AG). A análise dos resultados foi feita a partir do MPAE e taxa de acerto, representados respectivamente pelas figuras 22a e 22b. Com ambas as métricas foi obtida uma diferença estatística significativa entre as implementações, atingindo um valor F de 302,7 quanto ao MPAE e 853,9 quanto à taxa de acerto.



(a) Desempenho - MPAE **(b)** Desempenho - Taxa de Acerto

Figura 22. Comparação das taxas de acerto e MPAE obtidos pelos dois métodos do experimento

Neste estudo não foram realizados experimentos e análises detalhadas sobre o tempo para a execução dos algoritmos. Contudo, uma análise preliminar mostra que a duração da execução da retropropagação na RNA pura, com os melhores parâmetros encontrados neste trabalho, é de aproximadamente 150 segundos, com uma base de dados abrangendo um total de doze anos. Já o tempo de execução do AG para treinar o modelo híbrido, nesta mesma base, leva em torno de 1300 segundos.

As soluções desenvolvidas recebem entradas constituídas por índices que levam em seu cálculo os valores de fechamento das ações, ou seja serão executados apenas uma vez por dia. Por isto, a diferença entre os tempos de execução não é tão relevante, pois em uma aplicação prática, será necessário o fechamento da bolsa para que a inferência do modelo, que é uma estimativa do valor das ações para o próximo dia, seja computada. Como ambas as técnicas de treinamento (Retropropagação e Algoritmo Genético) executam na casa dos minutos, o tempo menor de processamento da Retropropagação não traz vantagem competitiva ao usuário, pois, em ambos os casos, ele deverá esperar a abertura do próximo dia para efetuar uma compra/venda.

5. Conclusão

Neste trabalho, uma RNA foi desenvolvida como modelo para prever o mercado de ações brasileiro. Além disso, um método de RNA híbrido foi desenvolvido, através do treinamento da RNA por meio de um AG. Em testes desenvolvidos, a RNA obteve uma Taxa de Acerto média em uma base de teste de 55.04%. Esta Taxa de Acerto é maior do que a classe majoritária da base de treinamento, que são os valores 1 que representam 52.15% do total da base, demonstrando que o modelo desenvolvido é capaz de aprender sobre o comportamento futuro do mercado de ações. Já com o modelo híbrido desenvolvido, foi obtido uma Taxa de Acerto média de 55.74%, o que comprova que o modelo híbrido desenvolvido com um AG pode incrementar o desempenho da RNA.

Os resultados obtidos comprovam que uma RNA possui a capacidade de prever o mercado de ações, porém as Taxas de Acerto obtidas demonstram que ainda são necessários muitos estudos para que se possa desenvolver um modelo de previsão com uma Taxa de Acerto satisfatória para investidores.

Como trabalhos futuros, podem ser feitos testes mais abrangentes em relação aos parâmetros, tanto dos métodos quanto do mercado e seus fatores; considerar o tempo de processamento dos algoritmos nos experimentos; bem como a implementação de outros modelos híbridos de RNA.

Contribuição dos Autores

Andrey de Aguiar Salvi realizou estudo do referencial teórico, implementação dos códigos das redes neurais artificiais e do algoritmo genético, realização dos experimentos, análise dos resultados e escrita do artigo. William Passig de Souza realizou estudo do referencial teórico, implementação dos códigos das redes neurais artificiais e do algoritmo genético, realização dos experimentos, análise dos resultados e escrita do artigo. Wilson Castello Branco Neto realizou a orientação do trabalho, planejamento dos experimentos, auxílio na análise dos resultados e revisão do artigo.

Referências

[1] NORVIG, P.; RUSSELL, S. *Inteligência Artificial: Tradução da 3a Edição*. São Paulo: Elsevier Brasil, 2013.

[2] RALEVIĆ, N. M. *et al.* The comparative analyses of the nonparametric methods for investment return prediction. In: IEEE. *Intelligent Systems and Informatics (SISY), 2015 IEEE 13th International Symposium on*. [S.l.], 2015. p. 111–115.

[3] AZEVEDO, F. M. D.; BRASIL, L. M.; OLIVEIRA, R. C. L. de. *Redes neurais com aplicações em controle e em sistemas especialistas*. Florianópolis: Visual Books, 2000.

[4] KOLAMBE, M. Survey paper on stock market prediction. *International Journal of Innovative Research in Computer and communication engineering*, v. 4, n. 10, 2016.

[5] SUTKATTI, R.; TORSE, D. Stock market forecasting techniques: A survey. *International Research Journal of Engineering and Technology*, v. 6, n. 5, 2019.

[6] GREAVES, A.; AU, B. Using the bitcoin transaction graph to predict the price of bitcoin. *Quoted*, v. 3, p. 22, 2015.

[7] DEVI, B. U.; SUNDAR, D.; ALLI, P. An optimized approach to predict the stock market behavior and investment decision making using benchmark algorithms for naïve investors. In: IEEE. *Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on*. [S.l.], 2013. p. 1–5.

[8] WU, J.-Y.; LU, C.-J. Computational intelligence approaches for stock price forecasting. In: IEEE. *Computer, Consumer and Control (IS3C), 2012 International Symposium on*. [S.l.], 2012. p. 52–55.

[9] BOYACIOGLU, M. A.; AVCI, D. An adaptive network-based fuzzy inference system (anfis) for the prediction of stock market return: the case of the istanbul stock exchange. *Expert Systems with Applications*, v. 37, n. 12, p. 7908–7912, 2010.

[10] DENG, Y. *et al.* A hierarchical fused fuzzy deep neural network for data classification. *IEEE Transactions on Fuzzy Systems*, v. 25, n. 4, p. 1006–1012, 2017.

[11] CASTILLO, P. A. *et al.* G-prop: Global optimization of multilayer perceptrons using gas. *Neurocomputing*, v. 35, n. 1-4, p. 149–163, 2000.

[12] PROCIANOY, J. L.; CIGERZA, G. C. Ipos in emerging markets: a comparison of brazil, india and china. In: *VIII Encontro Brasileiro de Finanças*. [S.l.: s.n.], 2008.

[13] BOVESPA. *Mercado de Capitais*. São Paulo: Bovespa, 1999.

[14] CORNETT, M.; ADAIR, T.; NOFSINGER, J. *Finanças*. São Paulo: MCGRAW HILL - ARTMED, 2013.

[15] Comissão de Valores Mobiliários. *Portal do Investidor: Ações*. 2018. Disponível em: ⟨http://www.portaldoinvestidor.gov.br/menu/Menu_Investidor/valores_mobiliarios/Acoes/⟩.

[16] CREPALDI, S. A. *Curso Básico de Contabilidade*. São Paulo: Editora Atlas S.A., 2013.

[17] BM&FBOVESPA. *Como investir em ações*. 2016a. Disponível em: ⟨http://www.bmfbovespa.com.br/pt_br/como-investir/como-investir-em-acoes/⟩.

[18] BM&FBOVESPA. *Índice Bovespa*. 2016b. Disponível em: ⟨http://www.bmfbovespa.com.br/pt_br/produtos/indices/indices-amplos/indice-bovespa-ibovespa.htm⟩.

[19] B3. *Perfil e Histórico*. 2017. Disponível em: ⟨http://ri.bmfbovespa.com.br/static/ptb/perfil-historico.asp?idioma=ptb⟩.

[20] VISUAL CAPITALIST. *The 20 Largest Stock Exchanges in the World*. 2017. Disponível em: ⟨http://www.visualcapitalist.com/20-largest-stock-exchanges-world/⟩.

[21] ERNST&YOUNG. *Comparing global stock exchanges*. 2012. Disponível em: ⟨https://www.biva.mx/documents/30877/866719/Comparing+global+stock+exchanges.pdf/8419e06d-5433-8d6b-5034-f94c4cdeaf5e⟩.

[22] CHAVES, D. A. T.; ROCHA, K. C. Análise técnica e fundamentalista: divergências, similaridades e complementariedades. *Monografia (Especialização)-Departamento de Administração da Faculdade de Economia, Administração e Contabilidade, Universidade de São Paulo. São Paulo*, 2004.

[23] ELDER, A. *Como Se Transformar em Um Operador e Investidor de Sucesso*. São Paulo: Elsevier, 2007.

[24] STOCKCHARTS. *Average Directional Index (ADX)*. 2018. Disponível em: ⟨https://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:average_directional_index_adx⟩.

[25] HAYKIN, S. *Redes Neurais - 2ed.* Porto Alegre: BOOKMAN COMPANHIA ED, 2001.

[26] KRIESEL, D. A brief introduction on neural networks. 2007.

[27] PISSARENKO, D. Neural networks for financial time series prediction: Overview over recent research. *University of Derby in Austria*. Disponível em: ⟨http://citeseer.ist.psu.edu/pissarenko02neural.html⟩.

[28] KUMAR, D. A.; MURUGAN, S. Performance analysis of indian stock market index using neural network time series model. In: IEEE. *Pattern Recognition, Informatics and Mobile Engineering (PRIME), 2013 International Conference on*. [S.l.], 2013. p. 72–78.

[29] DORFFNER, G. Neural networks for time series processing. *Neural Network World*, v. 6, p. 447–468, 1996.

[30] MITCHELL, M. *An introduction to genetic algorithms*. Cambridge: MIT press, 1998.

[31] AHMAD, F. *et al.* A ga-based feature selection and parameter optimization of an ann in diagnosing breast cancer. *Pattern Analysis and Applications*, v. 18, n. 4, p. 861–870, 2015.

[32] INTHACHOT, M.; BOONJING, V.; INTAKOSUM, S. Artificial neural network and genetic algorithm hybrid intelligence for predicting thai stock price index trend. *Computational intelligence and neuroscience*, v. 2016, 2016.

[33] PU, X.; LIN, Y.; SUN, P. A pruned cooperative co-evolutionary genetic neural network and its application on stock market forecast. In: IEEE. *The 26th Chinese Control and Decision Conference (2014 CCDC)*. [S.l.], 2014. p. 2344–2349.

[34] Yahoo. *Yahoo Finance*. 2018. Disponível em: ⟨https://finance.yahoo.com/⟩.

[35] JAIN, A.; SRINIVASULU, S. Development of effective and efficient rainfall-runoff models using integration of deterministic, real-coded genetic algorithms and artificial neural network techniques. *Water Resources Research*, v. 40, n. 4, 2004.