Ministério da Educação
Secretaria de Educação Profissional e Tecnológica

Instituto Federal Catarinense

Campus Rio do Sul

RODRIGO DE MORAES

PREVISÃO DE ÍNDICES DA BOLSA DE VALORES ATRAVÉS DE REDES NEURAIS ARTIFICIAIS: UMA COMPARAÇÃO COM O MODELO DE PREDIÇÃO BASEADO NA LÓGICA PARACONSISTENTE

Rio do Sul

2017

PREVISÃO DE ÍNDICES DA BOLSA DE VALORES ATRAVÉS DE REDES NEURAIS ARTIFICIAIS: UMA COMPARAÇÃO COM O MODELO DE PREDIÇÃO BASEADO NA LÓGICA PARACONSISTENTE

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia Catarinense – Campus Rio do Sul para a obtenção do título de bacharel em Ciência da Computação.

Orientador: Prof. Daniel Gomes Soares, Msc.
Co-Orientador: Prof. Guilherme Bitencourt Martins, Msc.

Rio do Sul

2017

RODRIGO DE MORAES

PREVISÃO DE ÍNDICES DA BOLSA DE VALORES ATRAVÉS DE REDES NEURAIS ARTIFICIAIS: UMA COMPARAÇÃO COM O MODELO DE PREDIÇÃO BASEADO NA LÓGICA PARACONSISTENTE

Este Trabalho de Curso foi julgado adequado para a obtenção do título de Bacharel em Ciência da Computação e aprovado em sua forma final pelo curso de Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia Catarinense – Campus Rio do Sul.

Rio do Sul (SC), 22 de novembro de 2017

_____

Prof. e Orientador Daniel Gomes Soares, Msc.

Instituto Federal de Educação, Ciência e Tecnologia Catarinense – Campus Rio do Sul

_____

Prof. e Co-Orientador Guilherme Bitencourt Martins, Msc.

Instituto Federal de Educação, Ciência e Tecnologia Catarinense – Campus Rio do Sul

_____

Prof. Juliano Tonizetti Brignoli, Dr.

Instituto Federal de Educação, Ciência e Tecnologia Catarinense – Campus Rio do Sul

_____

Prof. André Alessandro Stein, Msc.

Instituto Federal de Educação, Ciência e Tecnologia Catarinense – Campus Rio do Sul

AGRADECIMENTOS

Agradeço à minha família Atair de Moraes, Erli de Moraes e Adriana de Moraes pelo apoio e incentivo prestados durante os períodos do curso.

Aos meus chefes e grandes amigos Fabio Baldo e Adriano Baldo por me proporcionarem horários flexíveis de trabalho para que assim eu possa ter realizado os anos letivos no período matutino.

Principalmente a meu orientador Daniel Gomes Soares e Co-Orientador Guilherme Bitencourt Martins pelos conselhos e orientações prestados com muita dedicação durante o desenvolvimento deste trabalho.

E por fim, a todos os professores e colegas do curso, sem os quais muitos dos conhecimentos deixariam de ser adquiridos durante esta caminhada.

RESUMO

A predição de índices e/ou valores da bolsa de valores, bem como futuras cotações das ações, é um grande desafio até mesmo para especialistas com anos de experiência em técnicas de investimento. A previsão de tais séries temporais é um problema que tem recebido atenção dos pesquisadores nos últimos anos, na tentativa de encontrar um método preciso para prever tais valores. Prever o futuro, e em especial o comportamento de séries temporais é considerado um grande desafio, tanto para a estatística como para a computação. Dentro deste contexto, este trabalho teve como objetivo a construção de uma modelo baseado em Redes Neurais Artificiais (RNA) para a predição do índice S&P 500 da bolsa de valores de Nova Iorque. RNAs são modelos computacionais baseados no sistema nervoso dos seres vivos e têm como principal característica o poder de generalização e reconhecimento de padrões, se tornando ideal para o previsão de séries temporais. Para a realização deste trabalho, foram utilizadas RNAs do tipo Multilayer Perceptron (MLP) e como complemento, os resultados obtidos através da melhor RNA modelada, foram comparados com os resultados obtidos pelo modelo de predição de índices da bolsa, baseados na Lógica Paraconsistente, estipulado por Martins (2012). Os resultados mostram que é possível fazer previsão de índices da bolsa de valores, com erro e antecedência aceitáveis, utilizando-se RNAs e tendo como entrada dados históricos e estatísticos do índice. Também verifica-se que as RNAs se sobressaem em relação ao modelo paraconsistente, pelo fato de que uma Rede Neural Artificial irá realizar a predição para todos os dias, enquanto o modelo paraconsistente poderá não prever algum dia, por se tornar uma previsão inconclusiva para tal dia.

Palavras-chave: Previsão de Índices da Bolsa de Valores, Redes Neurais Artificiais, Inteligência Artificial, Lógica Paraconsistente.

ABSTRACT

The prediction of stock market indexes or values, as well as future stock quotes, is a great challenge even for experts with years of experience in investment techniques. The prediction of such time series is a problem that has received attention from researchers in recent years in an attempt to find a precise method to predict such values. Forecasting the future, and especially the time-series behavior, is considered a major challenge for both statistics and computing. In this context, the objective of this work was the construction of a model based on Artificial Neural Networks (RNA) for the prediction of the S & P 500 index of the New York stock exchange. RNAs are computational models based on the nervous system of living beings and have as main characteristic the power of generalization and pattern recognition, becoming ideal for the prediction of time series. In order to perform this work, RNAs of the Multilayer Perceptron type (MLP) were used and, as a complement, the results obtained through the best RNA modeling were compared with the results obtained by the prediction model of stock indexes, based on the Paraconsistent Logic, stipulated by Martins (2012). The results show that it is possible to forecast stock market indexes, with acceptable error and advance, using RNAs and having as input historical and statistical index data. It is also verified that ANNs stand out in relation to the paraconsistent model, because an Artificial Neural Network will predict every day, whereas the paraconsistent model may not predict someday, since it becomes an inconclusive prediction for such day.

Keywords: Stock Market Indexes Prediction, Artificial Neural Network, Artificial Intelligence, Paraconsistent logic.

LISTA DE FIGURAS

Figura 1 – Neurônio em um sistema biológico vivo 29
Figura 2 - Representação de um neurônio artificial 30
Figura 3 - Rede MLP com duas camadas intermediárias 34
Figura 4 - Fases de treinamento de uma rede MLP 35
Figura 5 - Amostra de dados disponíveis 44
Figura 6 - Função de ativação linear 49
Figura 7 - Função de ativação sigmoide tangente hiperbólica 49
Figura 8 - Função de ativação logística 50
Figura 9 - Modelo 01 de predição do índice da bolsa de valores 52
Figura 10 - Modelo 02 de predição do índice da bolsa de valores 52
Figura 11 - Modelo 03 de predição do índice da bolsa de valores 53
Figura 12 - Modelo 04 de predição do índice da bolsa de valores 53

LISTA DE GRÁFICOS

Gráfico 1 – Evolução do erro da MLP com menor RMSE para 6 meses de dados. 59
Gráfico 2 - Evolução do erro da MLP com menor MAPE para 6 meses de dados. 59
Gráfico 3 - Evolução do erro da MLP com menor RMSE e MAPE para 1 ano de dados. 60
Gráfico 4 - Evolução do erro da MLP com menor RMSE para 3 anos de dados. 61
Gráfico 5 - Evolução do erro da MLP com menor MAPE para 3 anos de dados 62
Gráfico 6 - Evolução do erro da MLP com menor RMSE para 5 anos de dados. 63
Gráfico 7 - Evolução do erro da MLP com menor MAPE para 5 anos de dados 64
Gráfico 8 – Valores reais e previstos pela Rede 1 (ver tabela 2) 66
Gráfico 9 - Valores reais e previstos pela Rede 2 (ver tabela 2) 66
Gráfico 10 - Valores reais e previstos pela Rede 3 (ver tabela 2) 67
Gráfico 11 - Valores reais e previstos pela Rede 4 (ver tabela 2) 67
Gráfico 12 - Valores reais e previstos pela Rede 5 (ver tabela 2) 68
Gráfico 13 - Valores reais e previstos pela Rede 6 (ver tabela 2) 68
Gráfico 14 - Valores reais e previstos pela Rede 7 (ver tabela 2) 69

LISTA DE TABELAS

Tabela 1 – Trabalhos Correlatos.....40
Tabela 2 – Sete melhores combinações de RNAs.....64
Tabela 3 – Resultados da aplicação das melhores RNAs.....70
Tabela 4 – Resultados para a estratégia paraconsistente. 73

LISTA DE ABREVIATURAS E SIGLAS

BM&FBovespa - Bolsa de Valores, Mercadorias e Futuros de São Paulo

CVM - Comissão de Valores Mobiliários

DAN - *Dynamic artificial network*

IA - Inteligência Artificial

IBOVESPA - Índice Bovespa

IBX - Índice Brasil

IPO - *Initial Public Offering*

LOGSIG – Função de ativação logística

MAD - *Mean Absolute Deviate*

MAE - *Mean Absolute Error*

MAPE - *Mean Absolute Percentage Error*

MLP - *Multilayer Perceptron*

MSE - *Mean Square Error*

NYSE - *New York Stock Exchange*

NASDAQ - *National Association of Securities Dealers Automated Quotations*

POCID - *Percentage of Correct Directional Prediction*

RNA - Redes Neurais Artificiais

ROC - *Rate of Change*

RSI - *Relative Strenght Index*

RMSE - *Root Mean Square Error*

S&P500 - *Standard and Poor's 500*

TANSIG – Função de ativação tangente hiperbólica

TLFN - *Time Lagged Feed-forward Network*

TrainBR – Algoritmo de Regularização Bayesiana

TrainLM – Algoritmo de Levenberg-Marquardt

WDBP - *Wavelet De-noising-based Back Propagation*

SUMÁRIO

1. INTRODUÇÃO **14**

1.1. PROBLEMATIZAÇÃO 15

1.1.1. Solução proposta **16**

1.1.2. Delimitação do escopo **16**

1.1.3. Justificativa **17**

1.2. OBJETIVOS 18

1.2.1. Objetivo geral **18**

1.2.2. Objetivos específicos **18**

1.3 METODOLOGIA 18

2. FUNDAMENTAÇÃO TEÓRICA **20**

2.1 MERCADO DE CAPITAIS 20

2.1.1 Ações **21**

2.1.2 Os valores das ações **22**

2.1.3 Mercado de bolsa e mercado de balcão organizado **23**

2.1.4 Índices de mercado **24**

2.1.4.1 Índice IBOVESPA 24

2.1.4.2 Índice IBX 24

2.1.4.3 Índice Standard & Poors's 500 (S&P500) 25

2.1.5 Técnicas de análise de investimento **26**

2.1.5.1 Análise fundamentalista 26

2.1.5.2 Análise técnica 27

2.2 REDES NEURAIS ARTIFICIAIS 28

2.2.1 Neurônios Artificiais **29**

2.2.2 Arquiteturas de Redes Neurais Artificiais **31**

2.2.3 Métodos de treinamento **32**

2.2.3.1 Aprendizado supervisionado 32

2.2.3.2 Aprendizado não supervisionado 33
2.2.4 Redes Multilayer Perceptron (MLP) 33
2.2.4.1 Método de treinamento de uma MLP 35
2.3 LÓGICA PARACONSISTENTE 37
2.3.1 Lógica Clássica e Não Clássicas 38
2.3.2 Imprecisão e Inconsistência 38
2.3.3 Lógica Paraconsistente Anotada 39
3. TRABALHOS CORRELATOS 40
3.1 COMPARAÇÃO ENTRE TRABALHOS CORRELATOS 40
3.1.1 Uso de Redes Neurais Artificiais para Predição da Bolsa de Valores 40
3.1.2 Applying Artificial Neural Networks to prediction of stock price and improvement of the directional prediction index – Case study of PETR4, Petrobras, Brazil 41
3.1.3 Using artificial neural network models in stock market index prediction 42
3.1.4 Forecasting stock indices with back propagation neural network 43
4. DESENVOLVIMENTO 44
4.1 DADOS DISPONÍVEIS 44
4.2 TRATAMENTO DOS DADOS 45
4.2.1 Normalização dos dados 45
4.3 VALIDAÇÃO CRUZADA 47
4.4 RECURSOS UTILIZADOS 48
4.4.1 Funções de Ativação 48
4.4.2 Algoritmos de treinamento 50
4.5 MODELOS DE RNAS 51
4.6 ÍNDICES PARA ANÁLISE DE QUALIDADE DA PREVISÃO 54
4.7 ESCOLHA DA MELHOR CONFIGURAÇÃO DE RNA 55
4.8 COMPARAÇÃO COM O MODELO PARACONSISTENTE 56
5. RESULTADOS 57

5.1 TREINAMENTO E RESULTADOS DAS RNAS 57
5.1.1 Treinamento 57
5.1.1.1 Dados de 6 meses 58
5.1.1.2 Dados de 1 ano 60
5.1.1.3 Dados de 3 anos 61
5.1.1.4 Dados de 5 anos 62
5.1.2 Resultados 64
5.2 ESTRATÉGIA PARACONSISTENTE 70
5.2.1 Estratégia e treinamento 71
5.2.2 Resultados 73
5.3 COMPARAÇÃO DOS RESULTADOS 74
6. CONCLUSÕES 76
REFERÊNCIAS 78
APÊNDICE A – ALGORITMO PARA MODELAGEM DAS REDES MLP 81
APÊNDICE B – ALGORITMO PARA ESTRATÉGIA PARACONSISTENTE 86

1. INTRODUÇÃO

Investir no mercado de ações é tido como um desafio para muitos, visto que os resultados para este tipo de investimento podem ser de certa maneira, muito difíceis de prever. De acordo com Wang et al (2011), “os dados históricos sobre os preços das ações são uma das informações mais importantes para investidores na bolsa de valores, porém, infelizmente, estes dados são dinâmicos, não lineares, não parametrizados e caóticos por natureza. Isso implica que os investidores têm que lidar com séries cronológicas não estacionárias e com frequentes quebras estruturais, dificultando a previsibilidade dos dados futuros.”

Ainda para Wang et al (2011), os preços das ações são afetados por diversos fatores macroeconômicos diferentes, dentre estes fatores se destacam, eventos políticos, taxas bancárias, expectativas dos investidores, condições financeiras em geral e até mesmo fatores psicológicos dos investidores. Prever a variação desses preços com precisão pode ser bastante desafiador, porém, de certa maneira, é de grande interesse para os investidores.

Existem vários estudos relativos a modelos de predição de índices da bolsa, tais como, o IBOVESPA da bolsa de valores brasileira como pode ser visto nos trabalhos de Rêgo e Mussa (2008) e Roque (2009), e o índice S&P500 da bolsa americana, conforme pode ser observado em Tsaih et al (1998) e Yudong e Lenan (2009). Naturalmente a predição dos índices através destes modelos se dá por indicadores de análise técnica, visto que este tipo de análise baseia-se na observação de gráficos e índices em busca de prever o comportamento futuro desses valores em função de uma série histórica de cotações.

Outro exemplo de modelo de predição é o proposto por Martins (2012) que usa premissas da Lógica Paraconsistente para avaliação destes indicadores técnicos e tenta sinalizar a direção do sentido dos índices da bolsa. De maneira breve, segundo Varela (2012), "a Lógica Paraconsistente é uma técnica que nos permite lidar com sistemas inconsistentes, porém não triviais. Isto é, as contradições não necessariamente trivializam uma argumentação baseada em Lógica Paraconsistente." Embora, seja um lógica não clássica, a paraconsistente é baseada nos princípios dela. Na lógica clássica, existem dois estado lógicos, verdadeiro ou falso. Ou seja, possui propriedades binárias, ou de ser ou não ser. Este fato faz com que ela tenha problemas com fatos do mundo real como com inconsistências, ambiguidades, paradoxos e indefinições.

A Lógica Paraconsistente se difere neste ponto. Nesta teoria, existem graus de certeza e incerteza, ou seja, uma proposição pode ter um nível de verdade e outro de falsidade. As informações se baseiam em graus de crença ou de evidências relativos a uma proposição. Em outras palavras a proposição e sua contradição podem ser ambas verdadeiras.

De maneira paralela, outra técnica que vem obtendo sucesso na predição de preços das ações ou índices da bolsa de valores, são as Redes Neurais Artificiais (RNAs), como é o caso do trabalho de Krieger (2012). Em geral, as RNAs possuem atributos de aprendizagem, generalização, processamento paralelo e resistência a erros (Wang et al, 2011). Outro aspecto importante, é o que mencionam Yudong e Lenan (2009), afirmando que a maioria dos modelos baseados em RNAs usam os dados históricos e atuais do índice de ações para prever os preços futuros, além disto as RNAs têm ganhado popularidade devido às suas capacidades inerentes para aproximar qualquer função não linear com um alto grau de precisão.

Dentro deste contexto, pelo fato das RNAs terem a capacidade de identificação de padrões, realizar previsões e de terem sido utilizadas com sucesso nesse tipo de problema, o objetivo deste trabalho foi utilizar a técnica de Redes Neurais Artificiais para, a partir da série histórica do índice de ações S&P500, prever a direção do sentido deste índice, ou seja, prever se o índice irá subir ou descer.

De forma complementar, foi realizado uma comparação entre os resultados obtidos através do modelo baseado em RNA e os resultados obtidos pelo modelo paraconsistente apresentado por Martins (2012), utilizando como base a mesma série temporal.

1.1. PROBLEMATIZAÇÃO

Como aponta Krieger (2012), fazer previsões no mercado de ações é um grande desafio, principalmente pelo fato de que este mercado é influenciado por diversos fatores. Dentre estes fatores podem ser citados, situações políticas e econômicas de companhias ou até mesmo países, comportamento histórico dos preços, questões sociais, dentre outros. Porém, mesmo com esse nível de imprevisibilidade, é possível a realização de uma análise técnica dos dados a fim de prever tendências no preço das ações. Entretanto, pelo grande número de variáveis envolvidas, este tipo de previsão torna-se muito complexa, inclusive para investidores qualificados[1] e com anos de experiência.

Muitos investidores apostam em sistemas automatizados, bem como robôs investidores, para predição de índices da bolsa, porém muitas vezes estes sistemas utilizam apenas regras de análise técnica para a decisão de compra ou venda das ações, métodos esses que muitas vezes não provêm de técnicas de Inteligência Artificial em sua composição, que

[1] De acordo com a regra criada pela Comissão de Valores Mobiliários (CVM), um investidor qualificado é qualquer pessoa física ou jurídica que possui investimentos em valor igual ou superior a 300 mil reais, e que atestem esta condição por escrito.

trariam, pelo menos em teoria, maior precisão quando comparados com os modelos tradicionais baseados em análise técnica.

Além disto, como muitos trabalhos têm sido feitos em torno deste tema e muitas técnicas foram desenvolvidas, torna-se difícil a escolha de qual método é o melhor e mais preciso. É possível encontrar na literatura modelos baseados na Lógica Paraconsistente para predição e também modelos que utilizam RNAs. Contudo, não se tem uma comparação entre ambos os modelos a fim de averiguar se uma Rede Neural, treinada com certa série temporal, é capaz de trazer resultados tão ou mais satisfatórios quanto os obtidos com a Lógica Paraconsistente ou vice-versa.

1.1.1. Solução proposta

Com o crescimento dos estudos na área de Tecnologia de Informação, bem como o avanço das técnicas de Inteligência Artificial, se tornou possível criar técnicas mais consistentes para previsões e identificações de padrões. A partir disto, a solução proposta por este trabalho é utilizar de técnicas da Inteligência Artificial conexionista para modelar uma Rede Neural Artificial (RNA) que seja capaz de prever o preço das ações da bolsa de valores e que venha a facilitar na tomada de decisões e na predição de índices da bolsa de valores.

Depois de modelada a RNA, foi realizado uma comparação entre a solução proposta por Martins (2012) e a rede criada neste trabalho. Esta comparação tem como objetivo identificar qual das duas técnicas analisadas produz previsões com menor erro.

1.1.2. Delimitação do escopo

Este trabalho se propõem a criar uma Rede Neural Artificial (RNA) para predição do índice S&P500 da bolsa de valores com um dia de antecedência. A Rede Neural criada é do tipo MultiLayer Perceptron (MLP), e para sua construção, treinamento e validação, foi utilizada a *toolbox* de Redes Neurais do software Matlab.

Durante a modelagem, vários testes foram feitos em diversas configurações de redes do tipo MLP, criando combinações entre o número de neurônios da camada oculta, o algoritmo de treinamento e a função de ativação. Os algoritmos de treinamento foram variados entre o Levenberg-Marquardt (TrainLM) e Regularização Bayesiana (TrainBR). Sobre as funções de ativação, foi usada a função linear (purelin) para a camada de saída, e na camada

oculta foram utilizadas as funções, sigmoide tangente hiperbólica (tansig) e sigmoide logística (logsig).

Os dados para treinamento, validação e testes da Rede Neural, são os dados da série histórica do índice S&P500 entre os períodos de setembro de 1997 e Outubro de 2016.

Além disto, também na ferramenta Matlab, foi criado um script para automatização dos processos descritos por Martins (2012) para predição dos índices através do modelo baseado na Lógica Paraconsistente. O trabalho visa também, realizar uma comparação entre este modelo paraconsistente e o melhor resultado obtido através da Rede Neural criada.

1.1.3. Justificativa

O crescente avanço do mercado de capitais, através principalmente dos IPOs[2] e consequentemente, do aumento de investidores, torna essa, uma área de estudo bastante visada por diversos pesquisadores de distintas áreas, desde economistas e engenheiros à cientistas da computação. Porém mesmo com vários estudos já realizados, a predição automática do mercado de ações ainda gera muitas dúvidas e opiniões contraditórias, e por isso há a necessidade de se criar modelos que validem a previsão desses índices de forma automática e com o menor erro possível

Além disto, um dos assuntos mais importantes dentro da teoria das finanças segundo Rêgo e Mussa (2008), é a hipótese de mercados eficientes, que diz que o mercado é considerado eficiente ao ponto que, nenhum agente ou investidor tenha retornos anormais e/ou superiores à média do mercado, apenas por obter informações atuais sobre o ativo no momento em que se está realizando o investimento. Isto significaria que a posse de informações sobre este mercado não alteraria o retorno esperado. Esta hipótese tem provocado diversos debates a respeito de sua validade, o que torna importante o estudo de técnicas de predição de índices da bolsa, que vão contra a teoria do mercado eficiente, ou seja, comprovando que é possível ter ganhos superiores à média do mercado, apenas com dados históricos sobre o ativo financeiro.

Também pelo fato da grande concentração de investimentos no mercado de capitais, e a grande oferta de trabalhos/estudos na área, bem como novas técnicas de investimento, cada

[2] IPO é a sigla para “Initial Public Offering” ou “Oferta Pública Inicial”, ou seja, é primeira vez que os proprietários de uma empresa renunciam parte da propriedade em favor de acionistas em geral. Este ato também é conhecido por “abertura de capital”. Em outras palavras, é quando a empresa vende pela primeira vez ações ao público.

vez tornam-se mais necessários estudos que validem e comparem estas técnicas criadas, a ponto de demonstrar e qualificar tais métodos.

1.2. OBJETIVOS

Esta seção descreve os objetivos geral e específicos do trabalho.

1.2.1. Objetivo geral

Realizar um estudo comparativo entre a técnica de Redes Neurais Artificiais e a estratégia baseada na Lógica Paraconsistente para realizar a previsão dos valores do índice S&P 500 da bolsa de valores.

1.2.2. Objetivos específicos

a. Definir as principais variáveis que possam auxiliar na previsão de índices da bolsa;
b. Modelar e treinar as RNAs para previsão do índice S&P500 da bolsa de valores;
c. Comparar as previsões feitas pelas RNAs com dados reais e identificar a configuração que gerou o menor erro;
d. Comparar os resultados obtidos pela melhor RNA com os da Lógica Paraconsistente.

1.3 METODOLOGIA

Esta seção apresenta a metodologia empregada para o desenvolvimento do trabalho.

a. Revisão Bibliográfica: Esta etapa objetivou o aprofundamento teórico acerca de previsão de índices da bolsa de valores e RNAs aplicadas a este problema. A pesquisa bibliográfica foi realizada em livros, monografias e artigos de periódicos.
b. Estudo do mercado financeiro bem como entendimento de seu funcionamento: Com base na revisão bibliográfica, foram identificadas as melhores variáveis para a previsão de índices da bolsa de valores, bem como o horizonte de previsão mais adequado.
c. Coletar o conjunto de dados da série temporal para treinamento e teste das RNAs.
d. Modelagem das Redes Neurais Artificiais: Modelar e treinar diferentes configurações de RNAs para a previsão de índices da bolsa de valores.

e. Validação das RNAs: Avaliar, através de métricas de erro, o desempenho das RNAs criadas a fim de selecionar as redes com menor erro. Estas redes pré selecionadas serão testadas com dados reais da bolsa de valores, para assim identificar a melhor configuração de RNA para predição do índice S&P500.
f. Comparação com o modelo baseado na Lógica Paraconsistente: Os resultados obtidos pela melhor RNA encontrada foram comparados, através de métricas de erro, com os resultados da Lógica Paraconsistente, para assim estabelecer qual das duas técnicas obtém melhor desempenho.

2. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são abordados conceitos fundamentais para a realização e entendimento deste trabalho. Conceitos estes, referentes às áreas de Mercado de Capitais, Redes Neurais Artificiais e Lógica Paraconsistente.

No contexto de Mercado de Capitais, são apresentados os temas relacionados com Bolsa de Valores e Mercado de Ações, bem como apresentação do índice S&P500. Após isto, é explicado o funcionamento de Redes Neurais Artificiais, incluindo suas características, arquiteturas, métodos de treinamento e, por fim, é detalhada a RNA do tipo MLP, utilizada neste trabalho. Além disto, também é apresentado o conceito de Lógica Paraconsistente e sua aplicação.

2.1 MERCADO DE CAPITAIS

Segundo a BM&FBovespa (2016), o Mercado de Capitais "tem como objetivo principal canalizar recursos dos agentes econômicos para a capitalização das empresas de capital aberto, por meio de operações com títulos e valores mobiliários em mercado de bolsa ou balcão. A compra e a venda de títulos e valores mobiliários pelos diversos agentes econômicos (indivíduos, empresas, fundos e instituições financeiras, por exemplo) é o que define a liquidez e o valor de mercado de tais ativos e, portanto, da empresa emissora."

De acordo com Cavalcante (2002), os principais títulos negociados no mercado de capitais são os representativos do capital de empresas (ações) ou empréstimos tomados, via mercado, por empresas que permitem a circulação para custear o desenvolvimento econômico. Desta forma, uma empresa não precisará ficar restrita a sua geração de lucros ou novos aportes de seus acionistas para financiar seus planos de investimento, podendo captar recursos, junto a novos investidores.

"A principal função do mercado de capitais é, portanto, possibilitar que empresas, com o intuito de viabilizar projetos de investimento, captem recursos diretamente do público investidor em condições mais vantajosas do que as oferecidas pelos empréstimos e financiamentos bancários. Quando as companhias decidem levantar recursos dessa forma, realizam uma nova emissão de valores mobiliários no mercado." (BM&FBovespa, 2016)

Lagioia (2007) aponta que as operações de abertura de capital precisam ter autorização da Comissão de Valores Mobiliários (CVM), que é o órgão fiscalizador do mercado de capitais brasileiro, o qual também registra e autoriza a emissão dos valores mobiliários para

distribuição pública. Em outras palavras, a CVM é o principal órgão responsável pelo controle, normatização e fiscalização do Mercado de Capitais, além de ser responsável também pela regulação e supervisão dos emissores de valores mobiliários, das entidades administradoras de mercado de bolsa e balcão e instituições integrantes do sistema de distribuição de valores mobiliários.

Além disto, como é apontado pela BM&FBovespa (2016), o Mercado de Capitais, pode ser dividido em mercado primário e secundário.

O mercado primário é o ambiente em que as empresas emitem títulos e valores mobiliários para captar novos recursos diretamente de investidores. Os valores mobiliários de uma nova emissão da empresa são negociados diretamente entre esta e os investidores, e os recursos são destinados para projetos de investimento da empresa ou para o caixa.

O mercado secundário é o ambiente em que os investidores negociam e transferem entre si os valores mobiliários emitidos pelas empresas. Neste caso, ocorre apenas a transferência de propriedade e de recursos entre investidores, ou seja, a empresa não tem participação neste mercado. O mercado secundário pode ser considerado como o oferecedor da liquidez aos títulos emitidos no mercado primário.

2.1.1 Ações

De acordo com a BM&FBovespa (2016) “ações são valores mobiliários, quando ofertados publicamente, quaisquer títulos ou contratos de investimento coletivo que gerem direito de participação, de parceria ou remuneração, inclusive resultante da prestação de serviços, cujos rendimentos advêm do esforço do empreendedor ou de terceiros”.

Outra definição é a de Lagioia (2007), que aponta uma ação como sendo a menor fração do capital social de uma empresa de sociedade anônima, elas são títulos de renda variável, nominativos e negociáveis que não possuem prazo de resgate, podendo ser negociadas no mercado em bolsa de valores.

Ainda de acordo com Lagioia (2007), as ações se classificam quanto à sua forma de emissão e quanto à sua espécie. Referente à forma de emissão, existem dois tipos de ações:

- Nominativas (N): Quando as ações são emitidas sob a forma física de cautelas ou certificados, que comprovam a existência e a posse de certa quantidade especificada de ações.
- Escriturais (E): Quando as ações são emitidas sob a forma de registro eletrônico, no qual os valores são lançados a débito ou a crédito dos acionistas, não havendo movimentação

física de documentos. Neste caso, o controle fica a cargo de uma instituição depositária. Atualmente, as ações são representadas predominantemente em formato escritural, através de extrato bancário comprovador da propriedade.

Quanto às espécies das ações, elas são divididas em:

- Ordinárias (ON): Proporcional participação nos resultados da empresa e conferem ao acionista o direito a voto nas assembleias gerais.
- Preferenciais (PN): Garantem ao acionista a prioridade na distribuição de dividendos (geralmente um percentual mais elevado do que o das ações ordinárias) e no reembolso de capitais.

2.1.2 Os valores das ações

Um dos aspectos mais importante do mercado de ações, de acordo com Lagioia (2007), é a formação dos preços, ou a cotação das ações. A cotação da ação é o preço pelo qual a ação é negociada, e tal cotação pode ser acompanhada por meio de jornais, publicações especializadas, serviços de difusões de cotações e pela internet.

Segundo Cavalcante (2002), os preços das ações refletem as expectativas dos agentes econômicos (compradores e vendedores) com relação às perspectivas do país e ao desempenho das empresas. As mudanças que ocorrem no país aumentam ou diminuem a confiança nas empresas abertas e influenciam no preço de suas ações. Porém, as ações podem apresentar valores monetários de referência diferentes, dependendo do critério utilizado na sua conceituação. De acordo com Lagioia (2007), estes valores podem ser: valor de mercado, nominal, patrimonial, contábil, de liquidação, de subscrição e intrínseco, conforme descritos abaixo:

- Valor de Mercado: É o preço pelo qual a ação está sendo negociada, ou seja, é o valor que os compradores estão aceitando pagar e os vendedores estão aceitando receber para fazer o negócio.
- Valor Nominal: Este valor é aquele estabelecido pelos estatutos da companhia e que vem impresso na ação. Ele é correspondente ao capital social da sociedade anônima dividido pela quantidade de ações.
- Valor Patrimonial: Diferentemente do valor nominal que é o capital social da empresa dividido pelo número de ações, o valor patrimonial representa o patrimônio líquido da empresa dividido pelo número de ações.

- Valor Contábil: Corresponde ao valor lançado no estatuto ou nos livros da companhia. Pode ser explícito (valor nominal e preço de emissão) ou indiscriminado (sem valor).
- Valor de Liquidação: É o valor da ação avaliado em caso de encerramento das atividades da companhia.
- Valor de Subscrição: Preço da emissão fixado em subscrições para aumento de capital. Em tese, o valor de subscrição não pode ser menor que o valor nominal contábil.
- Valor intrínseco: Valor avaliado no processo fundamentalista, que é uma metodologia de análise que determina o preço justo da ação, fundamentada na expectativa de resultados futuros.

2.1.3 Mercado de bolsa e mercado de balcão organizado

De acordo com a BM&FBovespa (2016), o mercado de capitais é compreendido tanto pelo mercado de bolsa quanto pelo mercado de balcão.

No mercado de bolsa, as negociações são abertas e realizadas regularmente por meio de sistemas centralizados e multilaterais de negociação, possibilitando o encontro e a interação de ofertas de compra e de venda de valores mobiliários, permitindo a execução de negócios tendo como contraparte a instituição responsável por este ambiente, respeitadas algumas condições estabelecidas em suas normas.

Já o mercado de balcão, pode ser considerado organizado, quando existe fiscalização governamental, ou não organizado nos demais casos. Neste caso, os valores são negociados apenas entre as partes envolvidas, mas devem, necessariamente, ser registrados em sistemas centralizados.

De acordo com Lagioia (2007), existem duas formas de negociar ações, uma por meio de viva voz e outro através do sistema eletrônico de negociação (Mega Bolsa).

No modo através de viva voz, o operador apregoa sua intenção de negocia, especificando o nome da empresa, o tipo de ação, a quantidade e o preço de compra ou de venda. Já através do sistema eletrônico de negociação, o fechamento de negócios é realizado automaticamente, funcionando de forma contínua.

2.1.4 Índices de mercado

Segundo Lagioia (2007), um índice de mercado, ou índice de ações é um composto estatístico que registra mudanças nas cotações das ações e mede as altas e baixas das ações numa determinada bolsa de valores.

Outra definição é a dada por BM&FBovespa (2016), apontando que os índices acionários resume a evolução dos preços de um conjunto de ações em um único indicador. São números absolutos utilizados para representar o valor de mercado de uma carteira teórica de ações e para observar sua evolução temporal.

Cavalcante (2002) aponta que o critério de seleção das ações, o tipo de média utilizado para combinar as oscilações das diferentes ações que integram um índice, e a forma de ponderação utilizada em sua geração, personalizam os índices. Dessa forma, a performance dos diferentes índices geralmente é próxima, mas não idêntica.

2.1.4.1 Índice IBOVESPA

Segundo Cavalcante (2002) o IBOVESPA é o índice de ações mais conhecido do Brasil. Calculado ininterruptamente desde 4 de junho de 1968, o IBOVESPA é um importante indicador do desempenho médio das cotações do mercado de ações brasileiro.

Ele retrata o comportamento das principais ações negociadas na bolsa, representando a variação do valor de uma carteira teórica, constituída pelas ações que em conjunto, representam 80% do volume transacionado no mercado a vista, nos doze meses anteriores à formação desta carteira. Essa carteira é reavaliada a cada 4 meses, onde é refeita à sua composição com base no último ano.

De acordo com a BM&FBovespa (2017), em abril de 2017, algumas das empresas que compõem o índice IBOV, são Ambev (ABEV3), Bradesco (BBDC3), Embraer (EMBR3), Lojas Americanas (LAME4) e Pão de Açúcar (PCAR4).

2.1.4.2 Índice IBX

De acordo com Cavalcante (2002), o IBX (Índice Brasil) é um índice de lucratividade também calculado pela Bovespa. Teve início em 02 de janeiro de 1997 e seu rebalanceamento é feito a cada 4 meses.

O critério de escolha para as ações negociadas é o número de negócios e o volume financeiro apurados nos 12 meses anteriores à reavaliação. As ações escolhidas são ponderadas no índice pelo seu respectivo número de ações disponíveis para negociação no mercado.

2.1.4.3 Índice Standard & Poors's 500 (S&P500)

Segundo Siegel (2015), a *Standard Statistics Co.* foi criada em 1906 e em 1918, começou a publicar o primeiro índice de valor das ações com base no desempenho ponderado pela capitalização ou pelo valor de mercado de cada empresa, sendo diferente de outros índices que tinham sua base pelo preço das ações.

Ainda segundo Siegel (2015), a ponderação por capitalização, é considerada a que melhor oferece a indicação de retorno no mercado em geral, é utilizada quase que universalmente para estabelecer referências de mercado.

"O índice de preços de ações *Standard & Poors's*, criado em 1923, passou a ser chamado *Standard & Poor's Composite Index* em 1926 e continha 90 empresas. Em 4 de março de 1957, ele foi ampliado para 500 empresas e tornou-se o índice S&P 500." (SIEGEL, 2015).

De acordo com Siegel (2015), na sua criação, o valor do índice compunha aproximadamente 90% do valor de todas as ações listadas na NYSE. Dentre as 500 empresas, haviam exatamente 425 empresas industriais, 25 ferrovias e 50 empresas de serviços de utilidade pública. Até 1988 era restringido o número de empresas de cada setor a estes descritos em sua criação, porém após 1988 não há mais nenhuma restrição setorial para as empresas selecionadas.

O índice S&P 500 é composto por quinhentas ações selecionadas de acordo com o seu tamanho de mercado, liquidez e sua representação de grupo industrial. Os ativos deste índice estão presentes para negociação nas bolsas dos Estados Unidos, NYSE (*New York Stock Exchange*) e NASDAQ.

Segundo o site oficial do índice[3], o S&P 500 faz parte de uma família de índices denominada *S&P Global*, onde todos os índices dessa família são regidos pelo comitê do *Standard and Poors (S&P)*. Dentro desta família destacam-se como populares os índices *S&P MidCap 400*, que representa o mercado de companhias de capital médio, e o *S&P SmallCap 600*, que representam as empresas de capital pequeno. Por sua vez, o S&P 500, é considerado

[3] Site oficial do índice S&P 500: http://www.standardandpoors.com

como medidor do desempenho das empresas de grande capitalização do mercado norte americano.

2.1.5 Técnicas de análise de investimento

De acordo com Krieger (2012) o investimento em ações, exige certas tomadas de decisões no momento de venda e compra dos ativos, onde é preciso ter uma análise das expectativas nos rendimentos a serem recebidos a curto ou longo prazo, além da possível valorização que possa ocorrer nas ações.

Em outras palavras, a tarefa principal do investidor é avaliar o retorno esperado do seu capital investido em ativos, tendo em vista o risco assumido. Estas análises tendem projetar o comportamento futuro das ações, formulando previsões em relação às variações de seus preços no mercado. Para isso, existem dois tipos distintos de análise, a técnica e a fundamentalista. (NETO,2003)

2.1.5.1 Análise fundamentalista

Segundo Neto (2003), a análise fundamentalista é baseada no desempenho econômico e financeiro de uma companhia, processando avaliações e comparações, através de análises das variáveis internas e externas da empresa, as quais exerceram influências sobre o seu desempenho.

Krieger (2012) complementa que a análise fundamentalista se baseia nos fundamentos da empresa, através de principais demonstrativos financeiros como: fluxo de caixa, balanço patrimonial, demonstrativo de resultados, entre outros. Este tipo de análise tem como objetivo tentar antecipar o comportamento no futuro de uma determinada companhia no mercado

De acordo com Lagioia (2007) a análise fundamentalista, é o estudo dos fatores que afetam as situações de oferta e demanda de um mercado, com o objetivo de determinar o valor intrínseco (preço justo) de um ativo. Geralmente, este tipo de análise está atrelada a movimentos no longo prazo e define qual o ativo deve ser comprado ou vendido, porém não prediz um tempo certo para entrar ou sair do mercado.

2.1.5.2 Análise técnica

Lagioia (2007) aponta que a análise técnica é uma abordagem que utiliza gráficos como a principal ferramenta na determinação do melhor momento para comprar e vender ativos. Ela se utiliza de informações passadas sobre os preços para definir as decisões de investimento. A análise técnica baseia-se na ideia de que o fator psicológico predomina no mercado e que os preços das ações não se movem aleatoriamente, e sim em padrões repetitivos e identificáveis.

A análise técnica tem a base que as variações nos preços guardam uma relação entre si, descrevendo tendências no mercado. Desta forma, pelas movimentações passadas é possível explicar suas futuras evoluções, não havendo preocupação com as causas que determinam as oscilações nos preços das ações. Esta análise tem como objetivo a predição de quando os preços irão se mover, e quando é a hora certa para entrar ou sair do mercado (NETO, 2003).

Segundo Roque (2009), os investidores que utilizam a análise técnica, fazem uso de gráficos para detectar possíveis tendências, tais tendências são baseadas, geralmente, na oferta e demanda, e normalmente apresentam um padrão cíclico. Este tipo de análise, provê diversos indicadores técnicos, o quais podem ser usados como regras de aplicações.

Com este fato, se faz interessante a exploração destes indicadores técnicos na modelagem e escolha dos parâmetros de entradas de uma Rede Neural Artificial. De acordo com Roque (2009), existem alguns indicadores que se destacam em relação à RNAs, são eles:

- Médias Móveis: Consistem em médias extraídas de um corpo de dados sequenciais numa janela de tempo, com a finalidade de informar sobre um início ou fim de uma tendência de preços. Por exemplo, numa média móvel de 15 períodos, o resultado de tal média, seria o preço médio do fechamento dos últimos 15 períodos. Existem três tipos e média móvel, a simples (média aritmética simples entre os períodos), a ponderada (são estabelecidos pesos diferentes aos períodos) e a exponencial.
- Volume: É uma medida que expressa o valor financeiro negociado num dia de pregão.
- Tendências: Pode ser verificado através de gráficos, as tendências representam a direção dos preços em tais gráficos, o qual tem um padrão notório de ziguezague.
- Momento: Mede a diferença entre os preços de fechamento em um determinado intervalo de tempo, representando a velocidade de evolução dos mesmos em tendências.

Pelo fato da análise técnica acreditar que os preços das ações apresentam uma tendência do mercado e que essas tendências se originam através de padrões cíclicos e identificáveis, com base na oferta e na demanda, faz-se interessante o uso de Redes Neurais

Artificias (RNAs) para realizar a predição dos valores de ações no mercado, visto que através do histórico das ações, uma RNA pode identificar tais padrões e tendências, a fim de predizer quais os próximos movimentos do mercado.

2.2 REDES NEURAIS ARTIFICIAIS

De acordo com Bookshear (2013), a Inteligência Artificial é o ramo da ciência da computação, onde cientistas têm como objetivo o desenvolvimento de máquinas que interajam com os seus ambientes da maneira mais semelhante o possível dos seres humanos, e que desempenhem suas funções de forma autônoma e inteligente, sem a necessidade de intervenção humana. A concretização deste objetivo exige que a máquina entenda e consiga tirar conclusões mediante alguma forma de raciocínio artificial.

Bookshear (2013) reforça que tanto a percepção quanto o raciocínio são atividades corriqueiras que, embora sejam simples para mente humana, são de grande complexidade para as máquinas. Além disto, diversos problemas ainda são de certa dificuldade e desafiam a capacidade de resolução através de computadores que utilizam abordagens algorítmicas tradicionais.

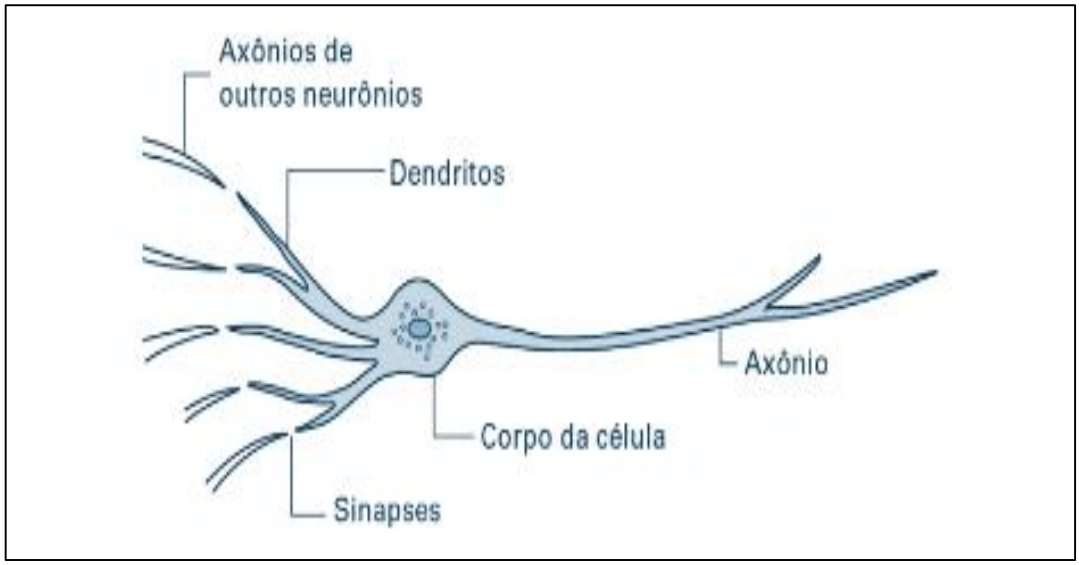
Estes fatores motivam muitos pesquisadores ao estudo de soluções para tais problemas. Muitos destes estudos envolvem abordagens influenciadas por fenômenos naturais, e especificamente, pelo sistema nervoso dos seres vivos, onde pode-se destacar as redes de neurônios biológicos como inspiradora para o conceito de Redes Neurais Artificiais (RNA).

“Redes Neurais Artificiais são sistemas paralelos distribuídos compostos por unidades de processamentos simples (neurônios artificiais) que calculam determinadas funções matemáticas (normalmente não-lineares). Tais unidades são dispostas em uma ou mais camadas interligadas por um grande número de conexões, geralmente unidimensionais. Na maioria dos modelos essas conexões estão associadas a pesos, os quais armazenam o conhecimento adquirido pelo modelo e servem para ponderar a entrada recebida por cada neurônio de entrada” (BRAGA; CARVALHO; LUDERMIR, 2011).

Segundo Braga, Carvalho e Ludermir (2011), as RNAs tentam reproduzir as funções das redes biológicas, buscando implementar seu comportamento funcional e sua dinâmica. E para melhor entendimento do conceito de Redes Neurais Artificiais, torna-se importante entender o funcionamento dos neurônios no cérebro humano, que é formado por aproximadamente 10 bilhões de neurônios.

Os neurônios biológicos são divididos em 3 seções: o corpo celular (ou soma), os dendritos e o axônio (Figura 1).

Figura 1 – Neurônio em um sistema biológico vivo



Fonte – Bookshear (2013, p. 445)

Os dendritos tem como função receber as informações, ou impulsos nervosos, oriundas de outros neurônios e conduzi-las até o corpo celular. O corpo celular é responsável pelo processamento das informações e gerar novos impulsos que serão transmitidos a outros neurônios, passando através do axônio até dendritos dos neurônios seguintes. (BRAGA; CARVALHO; LUDERMIR, 2011).

O ponto de contato entre a terminação axônica de um neurônio e o dendrito de outro é chamado de sinapse. É através das sinapses que os neurônios se juntam e formam as redes neurais biológicas, ou seja, as sinapses funcionam como válvulas, e são capazes de controlar a transmissão de impulsos, isto é, o fluxo da informação entre os neurônios e a rede neural.

2.2.1 Neurônios Artificiais

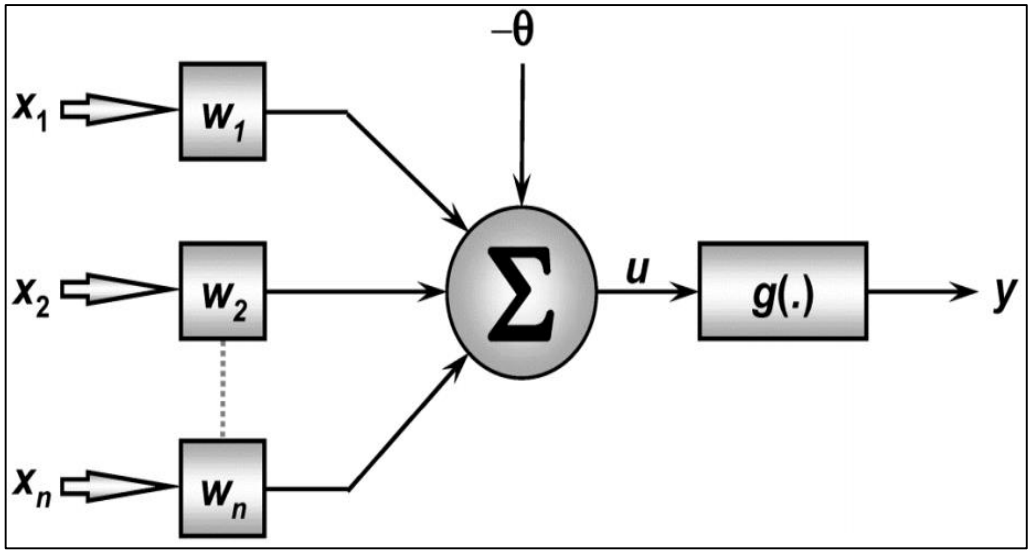
Assim como uma rede neural biológica, as RNAs são compostas por diversos neurônios artificiais, porém, diferentemente das redes biológicas, ou neurônios artificiais possuem menos conexões e as RNAs são relativamente menores, em relação ao número de neurônios. De acordo com Braga, Carvalho e Ludermir (2011), o modelo de neurônio artificial (Figura 2) foi proposto por McCulloch e Pitts, e é uma simplificação do que se sabia na época a respeito de neurônio biológico. Sua descrição matemática resultou em n terminais de entrada

(dendritos) que recebem os valores x_1, x_2, ..., x_n (que representam as ativações dos neurônios anteriores) e apenas um terminal de saída y (representando o axônio).

Braga, Carvalho e Ludermir (2011) ainda reforça que para representar o comportamento das sinapses, os terminais de entrada do neurônio tem pesos acoplados w_1, w_2, ..., w_n, cujos valores podem ser positivos ou negativos. Os pesos determinam em que grau o neurônio deve considerar sinais de disparo que ocorrem naquela conexão.

Um neurônio biológico dispara quando a soma dos impulsos que ele recebe, passam o limiar de excitação (ou *threshold*). Num neurônio artificial, tal comportamento é representado por um mecanismo que faz a soma dos produtos $x_i w_i$ e decide se o neurônio deve disparar ou não (saída igual a 1 ou 0), comparando a soma obtida ao limiar de excitação.

Figura 2 - Representação de um neurônio artificial



Fonte - Silva, Spatti e Flauzino (2010, p.34)

Neste modelo de neurônio artificial, a ativação do neurônio é obtida através da aplicação de uma "função de ativação", que ativa ou não a saída, dependendo do valor da soma ponderada das entradas.

Em outras palavras, a função de ativação é responsável por gerar a saída y do neurônio a partir dos valores dos vetores de peso (w_1, w_2, ..., w_n) e de entrada (x_1, x_2, ..., x_n). O seu principal objetivo, é limitar a saída do neurônio dentro de um intervalor de valores razoáveis a serem assumidos pela imagem da função. Segundo Silva, Spatti e Flauzino (2010), o resultado do neurônio artificial pode ser defino pelo seguinte:

$$y = g(u) \tag{1}$$

Sendo:

$$u = \sum_{i=1}^{n} w_i \cdot x_i - \theta \tag{2}$$

Braga, Carvalho e Ludermir (2011) reforçam, que a função de ativação g(u), ainda pode ser do tipo degrau, sigmoidal, linear ou gaussiana, cada uma com suas características.

2.2.2 Arquiteturas de Redes Neurais Artificiais

Independentemente da função de ativação escolhida, neurônios individuais possuem uma capacidade computacional limitada, no entanto, um conjunto de neurônios artificias conectados em forma de rede (neural) é capaz de resolver problemas de complexidade elevada (BRAGA; CARVALHO; LUDERMIR, 2011). A arquitetura de uma RNA define a disposição dos neurônios em relação aos outros. As principais arquiteturas de RNAs, segundo Silva, Spatti e Flauzino (2010, p. 46-50) podem ser divididas em:

- Arquitetura *feedforward* de camada simples: Possui uma única camada de entrada de sinais e uma única camada de neurônios, que é também a camada de saída. Como exemplos dessa arquitetura destacam-se as redes *Adaline* e *Perceptron*.
- Arquitetura *feedforward* de camadas múltiplas: Além das camadas de entrada e saída, existe também, uma ou mais camadas ocultas (ou escondidas) de neurônios. Esta arquitetura inclui as redes *Multilayer Perceptron* (MLP) e as redes de base radial.
- Arquitetura recorrente ou realimentada: Nesta arquitetura, o sinal dos neurônios de saída, são utilizados como entrada para outros neurônios. Como exemplo, pode-se citar redes de Hopfield e redes auto regressivas com entradas exógenas (NARX).
- Arquitetura em estrutura reticulada: o processo de ajuste dos pesos está diretamente relacionado com a localização (espacial) dos neurônios. Por exemplo, tem-se a rede Kohonen.

2.2.3 Métodos de treinamento

Segundo Braga, Carvalho e Ludermir (2011), uma das características mais importantes das RNAs é a sua capacidade de aprender por meio de exemplos. Diferente da Inteligência Artificial (IA) simbólica, onde o conhecimento é obtido através de regras explicitas, mas RNAs, o conhecimento é obtido através do ajuste das intensidades das conexões entre os neurônios. A etapa de aprendizado de uma RNA consiste em um processo iterativo de ajuste de parâmetros da rede (os pesos das conexões), que guardam ao final do processo, o conhecimento que a rede adquiriu do ambiente externo. O objetivo final do processo de treinamento é a generalização de soluções a serem produzidas pelas saídas, cujas respostas representam o sistema físico em que estão mapeando (SILVA; SPATTI; FLAUZINO, 2010).

Em outras palavras, "Aprendizado é o processo pelo qual os parâmetros livres de uma rede neural são ajustados por meio de uma forma continuada de estimulo pelo ambiente externo, sendo o tipo especifico de aprendizado definido pela maneira particular como ocorrem os ajustes dos parâmetros livres". (BRAGA; CARVALHO; LUDERMIR, 2011).

De forma genérica, o valor do vetor de pesos $w(t + 1)$ no instante $t + 1$ pode ser escrito da seguinte forma:

$$\mathrm{w}(\mathrm{t}+1) = \mathrm{w}(\mathrm{t}) + \Delta w(t) \tag{3}$$

Onde $w(t)$ e $w(t + 1)$, representam os valores dos pesos nos instantes t e $t + 1$, respectivamente, e $\Delta w(t)$ é o ajuste aplicado aos pesos. Os algoritmo de treinamento, se diferenciam basicamente na forma como é calculado $\Delta w(t)$.

Segundo Braga, Carvalho e Ludermir (2011), existem diversos algoritmos de treinamento de redes neurais, os quais podem ser agrupados em dois paradigmas principais: aprendizado supervisionado e aprendizado não supervisionado.

2.2.3.1 Aprendizado supervisionado

O aprendizado supervisionado, segundo Braga, Carvalho e Ludermir (2011), implica, necessariamente, na existência de um supervisor, o qual é responsável por estimular as entradas da rede por meio de padrões de entrada e observar a saída calculada pela mesma, comparando-a com a saída desejada. Como a resposta da rede é a função do valores do vetor de pesos, estes são ajustados de modo que venha aproximar a saída da rede com a saída desejada.

Para cada padrão de entrada, a rede rem sua saída corrente comparada com a saída desejada pelo supervisor, que fornece informações sobre a direção dos ajustes dos pesos. A minimização da diferença é incremental, já que pequenos ajustes são feitos nos pesos a cada etapa de treinamento, de tal forma que estes convirjam para um solução, caso tal solução exista. (BRAGA; CARVALHO; LUDERMIR, 2011).

Braga, Carvalho e Ludermir (2011) ainda reforçam que o aprendizado supervisionado pode ser implementado de duas formas diferentes. Sendo um treinamento off-line, onde os dados do conjunto de treinamento não mudam, e uma vez encontrada a solução para a rede, esta, deverá permanecer fixa. E outra um treinamento on-line, onde o conjunto de dados muda continuamente, e a rede deve estar em continuo processo de adaptação.

2.2.3.2 Aprendizado não supervisionado

Como o nome sugere, no aprendizado não supervisionado, não existe a figura do supervisor para acompanha o processo de treinamento. Neste esquema, ao contrário do modelo supervisionado, onde a rede possui os dados de entrada e saída, a rede possui apenas os padrões de entrada. Durante o processo de aprendizado, os padrões de entrada são apresentados continuamente à rede, e a existência de regularidades nesses dados faz com que o aprendizado seja possível. Ou seja, regularidade e redundância são características essenciais para que possa haver aprendizado supervisionado (BRAGA; CARVALHO; LUDERMIR, 2011).

2.2.4 Redes Multilayer Perceptron (MLP)

Segundo Silva, Spatti e Flauzino (2010) as redes *Multilayer Perceptron (MLP)* pertencem à arquitetura *feedfoward* de múltiplas camadas, ou seja, ela é caracterizada por possuir pelo menos uma camada de neurônios oculta (intermediária) situada entre as camadas dos neurônios de entrada e saída. Assim sendo, as redes MLP possuem no mínimo duas camadas de neurônios, que estarão distribuídos entre as camadas intermediarias e a camada de saída. As redes MLP apresentam um poder computacional maior do que aquele apresentado pelas redes de camada única (rede *Perceptron* ou *Adaline*, por exemplo), se caracterizando principalmente pelo fato de poder lidar com conjuntos de dados que não sejam linearmente separáveis (BRAGA; CARVALHO; LUDERMIR, 2011).

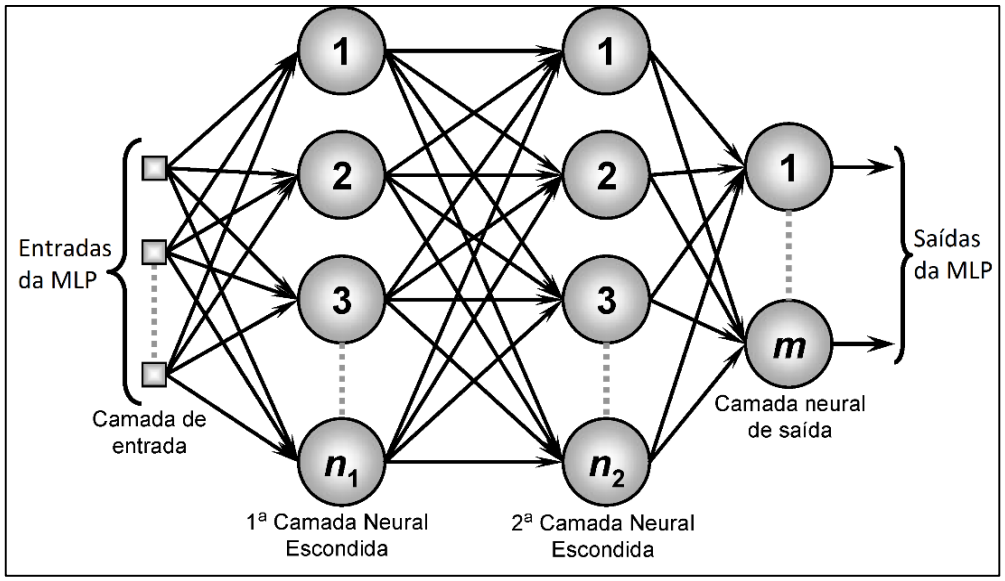
Braga, Carvalho e Ludermir (2011) ainda reforçam que, teoricamente, redes com duas camadas intermediárias podem implementar qualquer função, seja ela linearmente

separável ou não. E com apenas uma camada intermediária é possível implementar qualquer função contínua, desse modo, a maioria dos problemas práticos dificilmente necessita de mais de uma camada oculta, sendo necessária apenas no caso de haverem descontinuidades na função a ser aproximada.

A arquitetura de uma rede MLP pode ser acompanhada através da Figura 3, nesta arquitetura, os sinais são apresentados à rede em sua camada de entrada. As camadas intermediarias tem por objetivo obter as informações referentes ao comportamento da rede e codificar tais informações em forma de pesos sinápticos (SILVA; SPATTI; FLAUZINO, 2010).

De acordo com Braga, Carvalho e Ludermir (2011), o que define o processamento realizado por cada neurônio de uma determinada entrada, é a combinação dos processamentos realizados pelos neurônios da camada anterior. Já os neurônios da camada de saída recebem os estímulos vindos dos neurônios da última camada oculta, gerando um padrão de resposta, resultando na saída da rede. Dessa forma, todos os neurônios constituintes na rede MLP estarão distributivamente relacionados com o comportamento de entrada e saída do sistema.

Figura 3 - Rede MLP com duas camadas intermediárias



Fonte - Silva, Sapatti e Flauzino (2010, p. 92)

Ainda de acordo com Braga, Carvalho e Ludermir (2011) é muito importante definir a quantidade de neurônios em cada uma das camadas da rede, isso pelo fato de tal decisão implicar no desempenho à na capacidade de generalização da rede. A rede pode se tornar de maior complexidade conforme for maior o número de neurônios da rede. Porém, uma rede com mais neurônios pode resultar em um maior abrangência em termos de soluções possíveis. Braga,

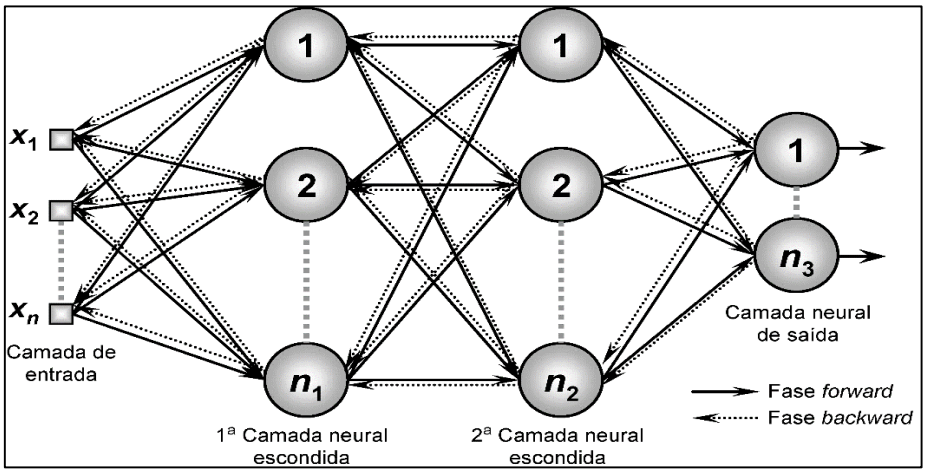
Carvalho e Ludermir (2011) ainda reforçam, que não há uma regra geral para definição do número de neurônios de uma rede, e que tal número irá variar de acordo com o problema e a pesquisa realizada.

De acordo com Haykin (2001), redes MLP têm sido aplicadas com sucesso na resolução de diversos problemas, através de seu treinamento de forma supervisionada com o algoritmo de retropropagação de erro, também conhecido pelo termo *backpropagation*.

2.2.4.1 Método de treinamento de uma MLP

Segundo Braga, Carvalho e Ludermir (2011) o algoritmo de treinamento mais comumente utilizado em uma rede MLP é o *backpropagation*, ou algoritmo de retropropagação, que é um método supervisionado de treinamento que utiliza pares de entrada e saída para ajustar os pesos da rede por meio de um mecanismo de correção de erros. Silva, Spatti e Flauzino (2010) reforça que o treinamento ocorre através de aplicações sucessivas compostas por duas fases, uma delas é a fase de propagação adiante (ou *foward*) e a outra é a fase de propagação reversiva (ou *backward*). Tais fases do treinamento são ilustradas pela Figura 4.

Figura 4 - Fases de treinamento de uma rede MLP



Fonte - Silva, Spatti e Flauzino (210, p.94)

Na fase de propagação adiante, de acordo com Braga, Carvalho e Ludermir (2011), o vetor de entrada *X* é apresentado às entradas da rede e os sinais dessa amostra são propagados por cada camada, até que se chegue na última camada, produzindo a saída da rede. A seguir, é calculado o erro da rede comparando as saídas geradas, com as saídas esperadas. Ou seja, essa fase do treinamento tem como característica obter os erros da rede após a propagação do sinal

por todas as camadas, através da consideração dos valores atuais dos pesos sinápticos e limiares dos neurônios.

Segundo Silva, Spatti e Flauzino (2010), a partir do valor destes erros, é aplicado a fase de propagação reversiva, que executa alterações nos pesos sinápticos e limiares de todos os neurônios da rede. De tal forma, a aplicação sucessiva destas fases de treinamento faz com que os pesos e limiares se ajustem em cada iteração, de modo que a soma dos erros produzidos pelas respostas da rede em função às desejadas, venha a diminuir.

Como aponta Silva, Spatti e Flauzino (2010, p.99), a medição da evolução do desempenho global deste algoritmo de treinamento pode ser efetuada por meio da avaliação do Erro Quadrático Médio (*MSE – Mean Squared Error*), que assumindo um conjunto de treinamento composto por *p* amostras, pode ser definido por:

$$E_M = \frac{1}{p}\sum_{k=1}^{p} E(k) \tag{4}$$

Onde $E(k)$ é o erro quadrático definido por:

$$E(k) = \frac{1}{2}\sum_{j=1}^{n} (d_j(k) - Y_j(k))^2 \tag{5}$$

Sendo $Y_j(k)$ o valor produzido pelo j-ésimo neurônio de saída da rede considerando a k-ésima amostra de treinamento e $d_j(k)$ seu valor desejado.

De acordo com Braga, Carvalho e Ludermir (2011) o ajuste dos pesos é baseado na regra delta, que foi proposta para o treinamento das tedes do tipo Adaline. A generalização deste algoritmo para utilização nas redes de camadas múltiplas é conhecida por regra delta generalizada.

O ajuste a ser aplicado em um determinado peso w_{ik} que conecta a entrada k ao neurônio i da camada escondida, pode ser definido por:

$$\Delta w_{ik}(n) = -\eta \cdot \frac{\partial E(n)}{\partial w_{ji}(n)} \tag{6}$$

No qual n é o número da iteração, η é uma constante de proporcionalidade correspondente à taxa de aprendizado e $\frac{\partial E\,(n)}{\partial w_{ji}\,(n)}$ a derivação da função de erro global em relação ao peso sináptico w_{ik} na iteração n, que determina a direção de busca no espaço de pesos. (BRAGA; CARVALHO; LUDERMIR, 2011).

2.3 LÓGICA PARACONSISTENTE

De acordo com Da Silva Filho (1999), a lógica foi iniciada na Grécia antiga, com Aristóteles (384-322 a.C) e seu grupo de filósofos, edificando-se assim as bases da lógica clássica. Estes primeiros estudos estabelecem que uma proposição ou é verdadeira, ou é falsa sem outros valores lógicos. Através da lógica clássica, o universo é considerado totalmente isento de situações contraditórias e indefinidas. Portanto, ao se aplicar a lógica clássica em qualquer situação do mundo real expressa por uma proposição A, temos como resultado o valor de A, ou não A, nada além disto.

Através da lógica clássica, pode-se dar o exemplo da determinação de que uma bola é azul, ou não azul, ou seja, a bola não pode ser azul e não azul ao mesmo tempo, ou ainda, quase azul. Cada sentença é qualificada como sendo verdadeira ou falsa, atribuindo-se a esta, os valores 1 e 0, respectivamente.

Da Silva Filho (1999) ainda reforça que a lógica clássica, desde que foi primeiramente formulada obteve poucas modificações, tendo mudanças revolucionárias apenas no século passado. Motivados pelo aparecimento de situações que não se enquadravam nas rígidas regras da lógica clássica, criaram-se estudos paralelos que culminaram com a instituição das logicas alternativas da clássica. Com isso, foram surgindo novas Lógicas chamadas Lógicas Não Clássicas, cujo objetivo era de se estudar o tratamento de situações como as indefinições, as inconsistências e os paradoxos que apareciam no mundo real, mas que não podem ser tratadas pela Lógica Clássica, pelo menos diretamente (DA SILVA FILHO, 1999).

Em 1910 o russo Nikolai A. Vasil'év (1880-1940) e o polonês Jan Lukasiewicz (1878-1956), publicaram trabalhos que tratavam da possibilidade de uma lógica que não eliminasse as contradições e inconsistências do mundo real. Em 1948, o lógico polonês Stanislaw Jàskowski formalizou com base na Lógica discursiva um cálculo proposicional paraconsistente denominado Cálculo Proposicional Discursivo. E de forma independente, na mesma época, o lógico brasileiro Newton C. A. da Costa, desenvolveu vários Sistemas Paraconsistentes contendo todos os níveis lógicos usuais: cálculo proposicional, cálculo de

predicados, cálculo de predicados com igualdade, cálculo de descrições e linguagem de ordem superior (na forma de teoria dos conjuntos). A partir dos resultados de Da Costa a Lógica Paraconsistente vem sendo um campo de pesquisa muito progressivo e promissor tanto numa perspectiva puramente teórica como em aplicações em áreas de Inteligência Artificial e Sistema de Computação (DA SILVA FILHO, 1999).

Segundo Da Silva Filho (1999), na lógica clássica, através do princípio da não contradição, não há a admissão de contradição do tipo de uma proposição e sua negação serem verdadeiras simultaneamente. Para isso, a Lógica Paraconsistente tem por objetivo, obter teorias, nas quais seja possível trabalhar com inconsistências e contradições.

2.3.1 Lógica Clássica e Não Clássicas

De acordo com Martins (2012) as lógicas podem ser classificadas em: lógica clássica e lógica não clássica.

A lógica clássica é fundamentada em 3 princípios básicos, são eles:

- Princípio de Identidade: Uma proposição p será sempre idêntica a si mesma.
- Lei da Não Contradição: Uma proposição p não pode ser verdadeira e falsa ao mesmo tempo
- Lei do Terceiro Excluído: Uma proposição p deve ser ou verdadeira ou falsa, nada além disto.

Estes três princípios constituem leis ontológicas e normas lógicas. Quando um destes princípios não é seguido, ou seja, é corrompido, tem-se uma lógica não clássica.

Como menciona Martins (2012), “o fato é que os princípios binários da lógica clássica não admitem algumas das situações que acontecem com frequência na vida real, como as situações contraditórias, as de indefinições, as de ambiguidades, aquelas vagas ou de pouca clareza”. Assim umas das razões mais importantes ao se considerar lógicas não clássicas, é o de se obter teorias nas quais inconsistências são permitidas sem que haja trivialização.

2.3.2 Imprecisão e Inconsistência

Martins (2012) aponta que se observarmos diversas situações do cotidiano, em quase todos os conceitos com os quais se pode deparar, há certo grau de imprecisão. Um exemplo que pode ser destacado, é que mesmo nas expectativas climáticas, por mais precisas

que possam parecer, há certo grau de imprecisão. Muitas situações semelhantes faz acreditar que a imprecisão é essencial e não se deixa eliminar mesmo com métodos científicos estritos.

Martins (2012) ainda reforça que no mundo real, seres humanos (ou agentes de forma geral) têm conhecimentos inconsistentes de forma sútil. Pois, é possível acreditar em determinada proposição p e também em sua negação. Isto pode acontecer, por exemplo, se uma agente tem as proposições p e não q como verdadeira, sem perceber que relativamente p e q são equivalentes.

Como a lógica clássica tem se mostrado incapaz de lidar com as condições de inconsistência e imprecisão, naturalmente existe o interesse por lógicas alternativas (não clássicas) para tal propósito. (MARTINS, 2012)

2.3.3 Lógica Paraconsistente Anotada

De acordo com Bispo e Cazarini (2006) sempre haverá a averiguação por parte da paraconsistência, se existe algum nível de inconsistência, divergência ou contradição em diferentes análises. Em outras palavras, a lógica paraconsistente nasceu da necessidade de se encontrar meios para tratar de forma não trivial, situações que se mostram contraditórias.

Assim sendo, Da Silva Filho (1999) afirma que a lógica paraconsistente trabalha com evidências e admite uma contradição de modo não trivial, onde as anotações são representadas por graus de certeza (graus de crença) e graus de incerteza (graus de descrença) à proposição, dando-lhe conotações de valoração, onde tal proposição pode possuir de fato, certo percentual de verdade e de falsidade simultaneamente.

3. TRABALHOS CORRELATOS

Na realização deste trabalho foi efetuada uma revisão bibliográfica com intuito de identificar trabalhos correlatos para previsão de valores da bolsa de valores. Além disto, a revisão buscou também as características utilizadas nas predições, tais como, variáveis utilizadas nos modelos, arquitetura das Redes Neurais utilizadas, tempo de antecedência e índices para análise de qualidade da previsão.

Neta etapa do trabalho, foram selecionados quatro trabalhos correlatos, que serão elucidados a seguir.

3.1 COMPARAÇÃO ENTRE TRABALHOS CORRELATOS

Na Tabela 1, são listados os trabalhos correlatos encontrados com data de publicação posterior à 2010.

Tabela 1 – Trabalhos Correlatos

Referência	Técnica utilizada	Métricas de erro
Krieger (2012)	TLFN focada (*focused time lagged feed-forward network*)	MAPE e MSE
De Oliveira, Nobre, Zárate (2013)	MLP	MAPE, RMSE, coeficiente THEIL e POCID
Guresen, Kayakutlu e Daim (2011)	MLP, DAN2 e GARCH-MLP	MSE e MAD %
Wang, Wang, Zhang e Guo (2011)	WDBP e MLP	MAE, RMSE, MAPE

3.1.1 Uso de Redes Neurais Artificiais para Predição da Bolsa de Valores

Este trabalho foi proposto por Krieger (2012), com intuito de realizar a predição da ação BVMF3 da Bolsa de Valores de São Paulo (BM&FBOVESPA), com o objetivo de avaliar se o uso de redes neurais possui capacidade de predizer o futuro fechamento da ação escolhida.

A arquitetura da rede neural escolhida para a execução deste projeto foi a TLFN focada (*focused time lagged feed-forward network*). Esta arquitetura é semelhante à *Multilayer Perceptron*, diferenciando-se somente na adição de atraso de tempo na camada de entrada, termo denominado de *momentum*. Esse tempo foi necessário, segundo o autor para que os dados

entrem de forma sequencial na rede. O algoritmo de aprendizagem para esta rede foi o *backpropagation with momentum*, que é a mais indicada para este tipo de arquitetura.

Como parâmetros de entrada, foram usados os seguintes indicadores técnicos: índice de negociabilidade, ROC (*Rate of Change*), RSI (*Relative Strenght Index*) e Percentual R.

Para testar a eficiência da topologia e das entradas de dados escolhidas, foram executados vários treinos com objetivo de encontrar uma rede neural que alcance alto grau de precisão. Para obter bons resultados é necessário vários ajustes na rede. Estes ajustes se baseiam em alterar a quantidade de neurônios ocultos, o termo *momentum* e a taxa de aprendizado, executar o treinamento e testar os resultados obtidos. A principal métrica de qualidade utilizada fora o MAPE e MSE.

Segundo o autor, os resultados obtidos na predição da ação da BVMF3 foram satisfatórios, pois, principalmente no primeiro mês, ficaram próximos do fechamento real. Depois do primeiro mês, a diferença foi aumentando, porém os índices de acerto nas oscilações continuaram altos.

3.1.2 Applying Artificial Neural Networks to prediction of stock price and improvement of the directional prediction index – Case study of PETR4, Petrobras, Brazil

O intuito do trabalho desenvolvido por De Oliveira, Nobre, Zárate (2013), é prever o preço da ação PETR4 da Petrobras.

Os autores estudaram diversas variáveis para modelagem do problema, porém nem todas foram utilizadas no treinamento da RNA, dentre elas pode-se citar algumas referente à uma visão técnica, tais como, o preço de abertura e fechamento da ação, o preço máximo e mínimo, o volume de compras, índice estocástico e médias móveis, além também de variáveis de visão fundamentalista utilizadas pelos autores, como o preço do barril de óleo (visto que o trabalho pretende prever ações da Petrobras), taxa Selic, índice de venda automobilística, dentre outros.

Foi escolhida a rede *MultiLayer Perceptron*, pelo fato dela ser uma rede aconselhável para aprendizagem de dados não lineares. A arquitetura é formada por 3 camadas de neurônios, uma para os neurônios de entrada, outra para a saída e uma camada de neurônios oculta. Para o treinamento da rede, foi utilizado o método supervisionado de retropropagação, ou *backpropagation*. No treinamento, os autores utilizaram, como metodologia de parada do algoritmo, 10 mil épocas de treinamento ou o erro mínimo de 0.0001. Para avaliação dos

resultados, foram utilizadas as métricas MAPE, RMSE, coeficiente THEIL e POCID (*Percentage of Correct Directional Prediction,* ou Percentual de Acerto na Direção de Predição).

De Oliveira, Nobre, Zárate (2013) definiram os seguintes passos para construir um modelo de RNA capaz de prever índices da bolsa, primeiro há de se entender o domínio do problema e a identificação das variáveis chave, para a partir daí, num segundo passo, realizar a coleta de dados históricos para treinamento da rede e posteriormente realizar o processamento das entradas e por fim ter os resultados da RNA.

Considerando o caso de estudo, o modelo que obteve melhor desempenho, mostrou um índice de acerto de direção (*POCID)* de 93,62% e *MAPE* de 5.45%. Por este fato, os autores apontaram que o estudo de RNAs pode apresentar resultados satisfatórios em relação a predição de índices da bolsa de valores.

3.1.3 Using artificial neural network models in stock market index prediction

Proposto por Guresen, Kayakutlu e Daim (2011), este trabalho teve o objetivo de avaliar a efetividade das RNAs na predição da bolsa de valores, comparando a performance de modelos já conhecidos e encontrados na literatura, tais como, a rede clássica MLP, a rede ARCH de Engle (1982), um modelo generalizado da rede ARCH, conhecido por GARCH, de Bollerslev (1986), o modelo exponencial ARCH ou EARCH de Nelson (1991), a *dynamic artificial network, DAN2,* entre outros trabalhos, que são citados por Guresen, Kayakutlu e Daim (2011).

Para realizar os testes das Redes Neurais Artificiais, foram usados dados das taxas de câmbio diárias da NASDAQ de Nova Iorque, entre o período de outubro de 2008 a junho de 2009. Para realizar a análise do erro obtido pelas redes foram utilizadas duas métricas em ambos os métodos, o MSE (*Mean Square Error*) e o MAD % (*Mean Absolute Deviate in Percentage*).

Segundo os autores, o modelo clássico da RNA do tipo MLP teve resultado melhor que os modelos DAN2 e o GARCH-MLP, onde este último obteve pior resultado dentre os modelos analisados.

3.1.4 Forecasting stock indices with back propagation neural network

Wang, Wang, Zhang e Guo (2011) propuseram um trabalho a fim de prever o valor do índice composto de Shanghai da bolsa de valores chinesa, através da arquitetura de RNA de retropropagação *Wavelet De-noising-based* (WDBP).

Os dados utilizados para treinamento e validação da rede, foram os valores dos fechamentos mensais dos preços do índice composto de Shanghai entre janeiro de 1993 e dezembro de 2009.

Além disto os autores realizam uma comparação entre o modelo WDBP, com o modelo clássico do algoritmo de retropropagação (*Backpropagation*). Basicamente, o algoritmo WDBP primeiro decompõem os dados originais em camadas, através da transformada *wavelet* e depois normaliza-os utilizando o modelo de retropropagação utilizando a frequência mais baixa de cada camada para a predição.

Neste trabalho, os autores utilizaram como métricas de erro, o MAE (*Mean Absolute Error*), RMSE (*Root Mean Square Error*) e o MAPE (*Mean Absolute Percentage Error*).

Segundo os autores, o modelo de previsão WDBP, pode ser bastante promissor em respeito a predição de índices e valores do mercado de ações, isto pelo fato de ser feito um pré-processamento dos dados antes da normalização dos mesmos. Onde tal processamento se dá através da transformada de *wavelet*. Além disto, comparado as métricas de erro propostas pelos autores, entre o modelo WDBP e o modelo clássico do *backpropagation*, os resultados obtidos pelo WDBP mostram-se mais satisfatórios, ou seja, com uma taxa de erro menor.

4. DESENVOLVIMENTO

Este capítulo tem por objetivo o detalhamento dos dados disponíveis, sua forma de obtenção e os tratamentos realizados antes de realizar os testes e treinamentos das Redes Neurais. Também serão abordados conceitos referentes à validação cruzada empregada no conjunto de dados. Além disto serão elucidados os recursos utilizados, incluindo o *software* utilizado nas simulações, as funções de ativação e algoritmos de treinamento. Também serão abordados os modelos de RNAs utilizados, bem como os parâmetros de entrada alterados durante os treinamentos. Por fim, são mostrados os índices para análise de qualidade da previsão.

4.1 DADOS DISPONÍVEIS

Os dados utilizados para treinamento, validação e testes das RNAs foram retirados do site oficial do índice S&P 500 [4] contendo informações dos valores do índice desde 09/09/2007 até 21/10/2016. A Figura 5 mostra a disposição dos dados da forma como foram obtidos e tais informações englobam amostras com os seguintes dados referentes ao índice: data, valor de abertura, máximo, mínimo, fechamento, variação, volume e índice de interesse de abertura.

Figura 5 - Amostra de dados disponíveis

	A	B	C	D	E	F	G	H	I
1	Date	Open	High	Low	Last	Change	Settle	Volume	Open Interest
2	2016-10-21	2136	2138,5	2123,25	2136	2,25	2134,75	1404835	2944425
3	2016-10-20	2137,5	2144,5	2126,75	2136,75	1	2137	1602595	2944305
4	2016-10-19	2131,5	2142,5	2126,5	2138	6	2138	930618	2928096
5	2016-10-18	2122,5	2139,75	2120,25	2132	9	2132	1129535	2925829
6	2016-10-17	2126,25	2130	2116,75	2121,75	4	2123	1246192	2920332
7	2016-10-14	2126,25	2143,25	2122,25	2126,25	0,75	2127	1834488	2928712
8	2016-10-13	2127,75	2132,25	2107,75	2126	5,25	2126,25	2175906	2929169
9	2016-10-12	2133	2140,75	2126,25	2128	3	2131,5	1465871	2927494
10	2016-10-11	2159,5	2160,75	2121,75	2133,5	24,5	2134,5	2231047	2942901

Fonte: Acervo do autor, 2017

Para os treinamentos, os dados disponíveis foram divididos em 4 grupos, o primeiro contém dados de 6 meses entre as datas de 20/10/2015 até 19/04/2016, o segundo possui as informações referentes a 1 ano, entre as datas de 20/04/2015 até 19/04/2016, o terceiro grupo

[4] Site oficial do índice S&P 500: http://www.standardandpoors.com

tem os dados de 3 anos, entre 22/04/2013 até 19/04/2016 e por fim, o último grupo contém os dados de 5 anos, entre 20/04/2011 até 19/04/2016.

Cada um dos modelos de RNAs apresentados foram treinados e validados para cada um destes grupos de dados, ou seja, um modelo de RNA, foi treinado e validado 4 vezes, uma vez para cada grupo de dados. Isto foi feito para avaliar se o tamanho do conjunto de dados presente no treinamento da RNA pode gerar uma diferença grande nos resultados da rede.

Além disto, os dados referentes às datas entre 20/04/2016 até 21/10/2016 foram utilizados para testar as RNAs que obtiveram os resultados com menor erro, ou seja, as melhores RNAs de cada modelo e conjunto de testes, foram testadas com este conjunto de dados.

4.2 TRATAMENTO DOS DADOS

Os dados foram obtidos no formato de uma planilha, e para utilização dos mesmos para as RNAs é necessário um pré-processamentos dos dados, organizando-os de forma a poderem ser utilizados no modelos neurais artificiais, que serão apresentados na seção 4.5.

Tal pré-processamento consiste em obter os dados da planilha base adquirida e dispô-los de acordo com as entradas e saídas de cada configuração de RNA. Após estes dados estarem organizados, foi necessário realizar a normalização dos dados, para que a partir daí sim, o dados estejam prontos para serem treinados e testados nas Redes Neurais modeladas.

4.2.1 Normalização dos dados

Para a utilização dos dados nos treinamentos e testes das RNAs, faz-se necessário a realização de um pré-processamento dos dados, denominado de normalização. De acordo com Silva, Spatti e Flauzino (2010), este processo objetiva escalonar as amostras de dados para uma faixa de valores dinâmica das funções de ativação das camadas escondidas, a fim de evitar o ruído dos dados externos à rede.

Segundo Roque (2009), a normalização ou pré-processamento dos dados de entrada, tem por objetivo diminuir a influência causada por valores que se destacam excessivamente em relação aos outros, ou seja, diminuir a distância entre os valores de variáveis muito espaçadas, se propondo a normalizar os dados do menor ao maior valor.

Os dados foram normalizados para dois intervalos diferentes, um intervalo de -1 a 1 e outro de 0 a 1, isto pelo fato das funções de ativação descritas na seção 4.4.1 utilizarem tais intervalos em seu escopo.

Na conversão dos dados para o intervalo de -1 a 1, foi utilizado um método de normalização representado pela seguinte equação:

$$y = \frac{2 \cdot (\mathrm{x} - \mathrm{x_{min}})}{\mathrm{x_{max}} - \mathrm{x_{min}}} - 1 \tag{7}$$

Onde, y é o valor normalizado, x é o valor original e $\mathrm{x_{min}}$ e x_{max} são os valores mínimo e máximo encontrado na amostra de dados que se está normalizando (DEMUTH; BEALE, 2002).

Para normalizar os valores proporcionalmente para o intervalo de 0 a 1, tem-se a utilização da expressão representada a seguir:

$$y = \frac{\mathrm{x} - \mathrm{x_{min}}}{\mathrm{x_{max}} - \mathrm{x_{min}}} \tag{8}$$

Onde da mesma forma como na expressão anterior, y vem a ser o valor normalizado, x, o valor original e $\mathrm{x_{min}}$ e x_{max} os valores mínimo e máximo (ZHANG; YOU, 2015).

Todos os dados, tanto de entrada como de saída, que foram utilizados nas RNAs empregadas neste estudo, passaram pelo processo de normalização, levando em consideração os valores máximos e mínimos dentro do conjunto de dados. Vale ressaltar que os dados foram normalizados de forma individual para cada intervalo utilizado, ou seja, a normalização para cada intervalo, se deu a partir dos dados originais.

De acordo bom Baghirli (2015), após os conjuntos de dados, já normalizados, serem processados pela RNA, a saída encontrada deverá passar pelo processo de desnormalização dos dados, para que somente assim, se tenha o valor exato da saída da rede. Para realizar a desnormalização dos dados que foram normalizados no intervalo de -1 a 1, pode ser utilizada a seguinte regra matemática:

$$x = \frac{(y + 1) \cdot (\mathrm{x_{max}} - \mathrm{x_{min}}) + \mathrm{x_{min}}}{2} \tag{9}$$

Sendo, x o dado desnormalizado, y o dado normalizado e x_{max} e x_{min} os valores máximo e mínimo do conjunto de dados antes da normalização (DEMUTH; BEALE, 2002).

Para desnormalizar os dados que foram normalizados no intervalo de 0 a 1, faz-se uso da seguinte expressão:

$$x = y \cdot (x_{max} - x_{min}) + x_{min} \quad (10)$$

Onde, de forma semelhante, x é o dado desnormalizado, y o dado normalizado e x_{max} e x_{min} os valores máximo e mínimo.

4.3 VALIDAÇÃO CRUZADA

De acordo com Silva, Spatti e Flauzino (2010), a validação cruzada (*cross-validation)* tem como objetivo avaliar a aptidão de uma determinada topologia de Rede Neural quando aplicada em um conjunto de dados diferente daquele utilizado no treinamento da RNA. Em outras palavras, a validação cruzada é um método a fim de estimar o erro da previsão de um modelo, utilizando parte do conjunto de dados para validação/teste do modelo.

Existem diferentes métodos de validação cruzada, dentre as opções disponíveis o método utilizado foi o de amostragem aleatória (*random subsampling cross-validation*). Neste método, segundo Silva, Spatti e Flauzino (2010), o conjunto total dos dados é aleatoriamente dividido em duas partes, uma destas partes é o subconjunto de treinamento e outra é o subconjunto de validação/teste. O subconjunto de treinamento, como o próprio nome propõe, foi utilizado para realizar o treinamento da topologia da RNA, enquanto o subconjunto de validação/teste, foi usado somente para testar e validar a topologia.

Assim, 70% do conjunto de dados utilizado para o treinamento das RNAs é aleatoriamente selecionado para o subconjunto de treinamento, já os 30% restantes são utilizados no subconjunto de teste e validação. No trabalho, a partição dos dois conjuntos foi repetida 30 vezes para cada modelo de RNA. Com esta estratégia, em cada execução, a possibilidade de contemplação de amostras diferentes tanto no subconjunto de treinamento como no de teste e validação (SILVA; SPATTI; FLAUZINO, 2010).

4.4 RECURSOS UTILIZADOS

A implementação, construção e testes das RNAs deste trabalho, foram feitas através da *toolbox* de Redes Neurais Artificiais disponível no *software* Matlab. Em virtude disto, no decorrer do trabalho, empregou-se a nomenclatura utilizada no *software* para as funções de ativação e algoritmos de treinamento.

Durante a modelagem das RNAs foram utilizadas diversas configurações, variando em cada modelo, além do número de neurônios da camada oculta e os parâmetros de entrada, a função de ativação dos neurônios da camada oculta e o algoritmo de treinamento.

Sobre as funções de ativação, foi usada a função linear (purelin) para a camada de saída, e na camada oculta foram utilizadas as funções, sigmoide tangente hiperbólica (tansig) e sigmoide logística (logsig). Os algoritmos de treinamento foram variados entre o Levenberg-Marquardt (TrainLM) e Regularização Bayesiana (TrainBR).

4.4.1 Funções de Ativação

Segundo Braga, Carvalho e Ludermir (2011), uma vez definida a estrutura de uma RNA, como sendo de múltiplas camadas e não de estrutura simples, é necessário determinar as funções de ativação de cada camada e o número de neurônios em cada uma delas. Nas camadas intermediárias da RNA, deve-se obrigatoriamente utilizar funções de ativação não lineares, para que a composição das funções nas camadas sucessivas seja capaz de resolver problemas de maior ordem no espaço de entrada. De acordo com Braga, Carvalho e Ludermir (2011), a utilização de funções lineares nas camadas intermediárias resultaria numa RNA linear, ou de única camada, já que funções lineares sucessivas podem ser descritas com uma única transformação linear.

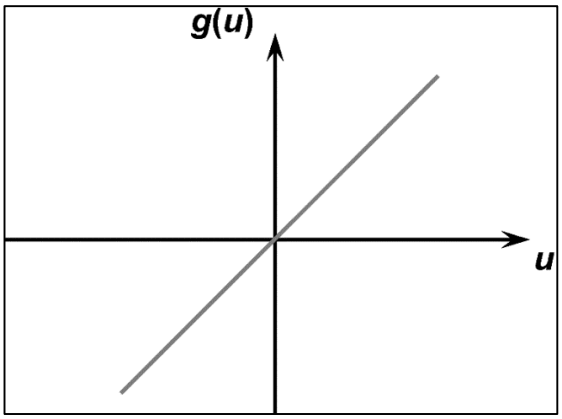
Assim, neste caso, as RNAs treinadas irão possuir na camada de saída sempre uma função de ativação linear, e as camadas intermediárias, variarão sua função de ativação entre a sigmoide tangente hiperbólica (tansig) e sigmoide logística (logsig).

A função de ativação linear (purelin) utilizada nas camadas de saída das RNAs, segundo Silva, Spatti e Flauzino (2010), produzem resultados de saída idênticos aos valores de potencial de ativação e pode ser definida pela expressão:

$$g(u) = u \tag{11}$$

Onde a representação gráfica desta função pode ser vista através da figura 6.

Figura 6 - Função de ativação linear



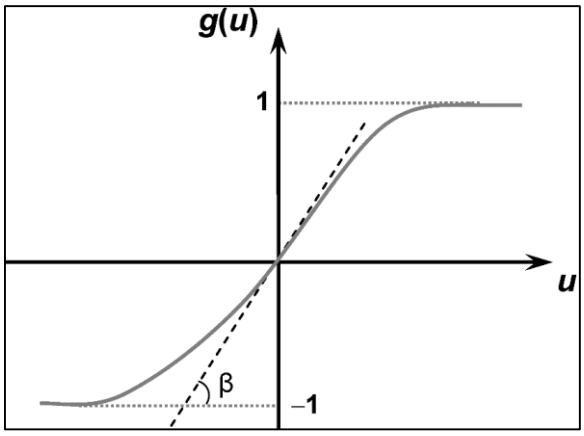
Fonte: Silva, Spatti e Flauzino (2010, p.42).

A função sigmoide tangente hiperbólica (tansig), possui resultados de saída entre os valores -1 e 1, e é definida por:

$$g(u) = \frac{1 - e^{-\beta \cdot u}}{1 + e^{-\beta \cdot u}} \tag{12}$$

Onde, de acordo com Silva, Spatti e Flauzino (2010), β está associado ao nível de inclinação da função em relação ao seu ponto de inflexão. A representação gráfica da função pode ser acompanhada na figura 7.

Figura 7 - Função de ativação sigmoide tangente hiperbólica



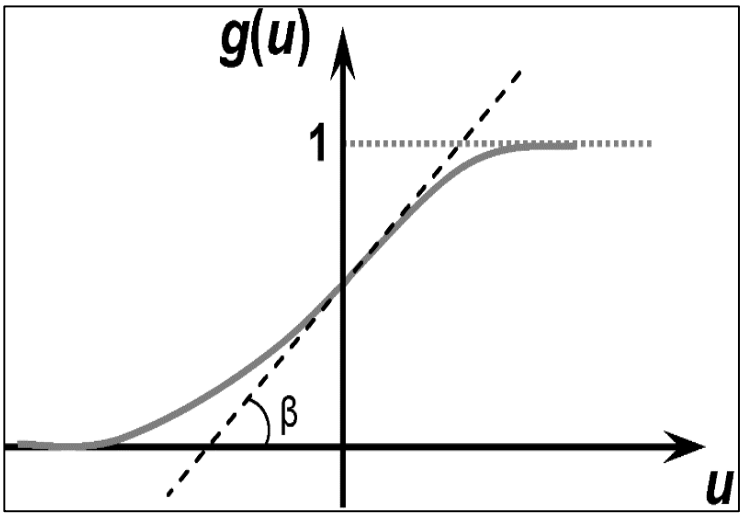
Fonte: Silva, Spatti e Flauzino (2010, p.40).

Já função sigmoide logística (logsig) possui resultado variando entre os valores 0 e 1, e pode ser definida pela seguinte expressão:

$$g(u) = \frac{1}{1 + e^{-\beta \cdot u}} \tag{13}$$

Sendo, do mesmo modo, β uma constante real associada ao nível de inclinação da função em relação ao seu ponto de inflexão. A visualização gráfica desta função pode ser vista na figura 8.

Figura 8 - Função de ativação logística



Fonte: Adaptado de Silva, Spatti e Flauzino (2010, p.39).

4.4.2 Algoritmos de treinamento

Segundo Silva, Spatti e Flauzino (2010), existem diversas variações do algoritmo backpropagation, tais variações foram propostas com o intuito de tornar mais eficiente o processo de convergência dos pesos da RNA.

Uma variação do algoritmo *backpropagation* é o algoritmo de Levenberg-Marquardt (TrainLM), que de acordo com Silva, Spatti e Flauzino (2010), tem o objetivo de reduzir o tempo de convergência e diminuir o esforço computacional requerido pelo algoritmo *backpropagation*. Além disso, normalmente o TrainLM, dentro dos algoritmos disponíveis no Matlab, é considerado o método de treinamento mais rápido para redes *feedforward*. Assim como aponta Silva, Spatti e Flauzino (2010), o TrainLM é um método gradiente de segunda

ordem, baseado no método dos mínimos quadrados para modelos não lineares. E de acordo com Beale, Hagan e Demuth (2014), em muitos casos, o TrainLM é capaz de obter menores erros quadráticos médios, em comparação com os outros algoritmos, porém conforme aumenta o volume da rede e o número de pesos que a rede possui, tal algoritmo tem suas vantagens anuladas, pois nestes casos podem necessitar muita memória e tempo de processamento.

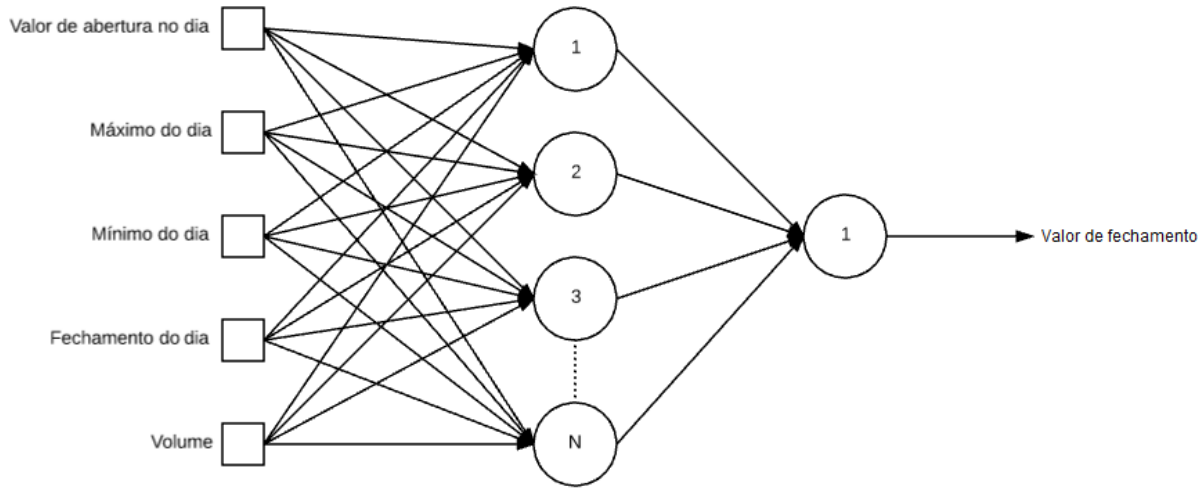
Outra variação do *backpropagation* é o algoritmo de Regularização Bayesiana (TrainBR). De acordo com Beale, Hagan e Demuth (2014), este método atualiza os valores dos pesos de acordo com a otimização de Levenberg-Marquardt, e logo após, através da aplicação do teorema de Bayes, determina uma combinação correta que produza uma rede com uma boa capacidade de generalização, fazendo uma estimação dos parâmetros de distribuição. Através de informações estatísticas bayesianas, a regularização bayesiana automaticamente seleciona os parâmetros de regularização e integra as propriedades de convergência do algoritmo de *backpropagation* tradicional.

4.5 MODELOS DE RNAS

Neste trabalho foram modeladas RNAs com diferentes parâmetros de entradas, a fim de identificar qual combinação de entrada possuirá melhor desempenho em relação as predições. Todos os modelos criados terão além da camada de entrada, uma camada com *n* neurônios ocultos e um neurônio na camada de saída. Além disto, todos os modelos foram treinados e validados da mesma forma, variando da mesma maneira o seu algoritmo de treinamento, função de ativação e quantidade de neurônios da camada oculta.

Um dos modelos implementados pode ser observado na Figura 9. Neste modelo, tem-se 5 parâmetros de entradas, e tais dados estão colocados assim como obtidos através do site oficial do S&P 500, todos os parâmetros de entradas são informações do dia atual. No neurônio de saída tem-se o valor de fechamento do dia posterior ao dia da predição.

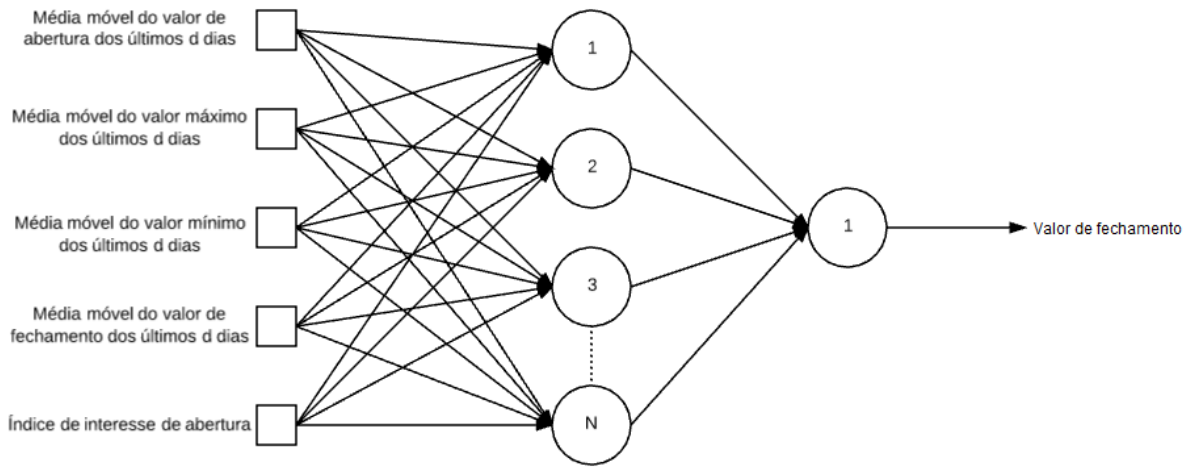
Figura 9 - Modelo 01 de predição do índice da bolsa de valores



Fonte: Acervo do Autor, 2017

O segundo modelo pode ser observado na Figura 10, e assim como o modelo anterior e todos os outros modelos, a saída da rede é o valor de fechamento do índice no dia posterior ao dia da predição. Neste outro modelo, os dados de entrada são as médias móveis de *d* dias anteriores ao dia da predição. Para os valores em médias móveis, foram utilizados os valores de abertura, máximo, mínimo e fechamento do índice. Além disto, como parâmetro de entrada também foi contemplado o índice de interesse de abertura do dia, onde neste parâmetro não é utilizado a média móvel.

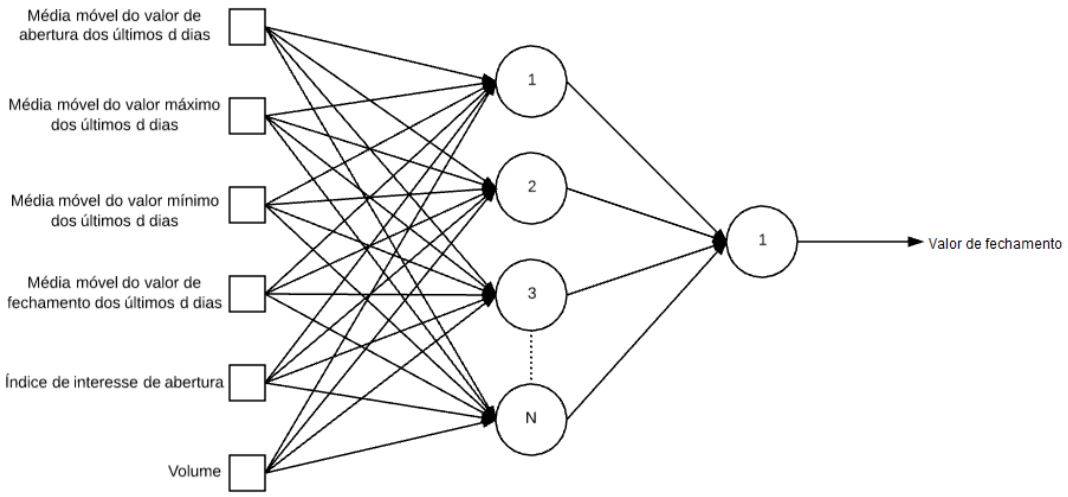
Figura 10 - Modelo 02 de predição do índice da bolsa de valores



Fonte: Acervo do autor, 2017

Da mesma forma, o terceiro modelo criado, utilizou os mesmos parâmetros do modelo anterior, porém além destes, foi adicionado também o volume de compra do dia, que da mesma forma como o índice de interesse de abertura, não é utilizado com médias móveis. A modelagem desta rede pode ser acompanhada através da Figura 11.

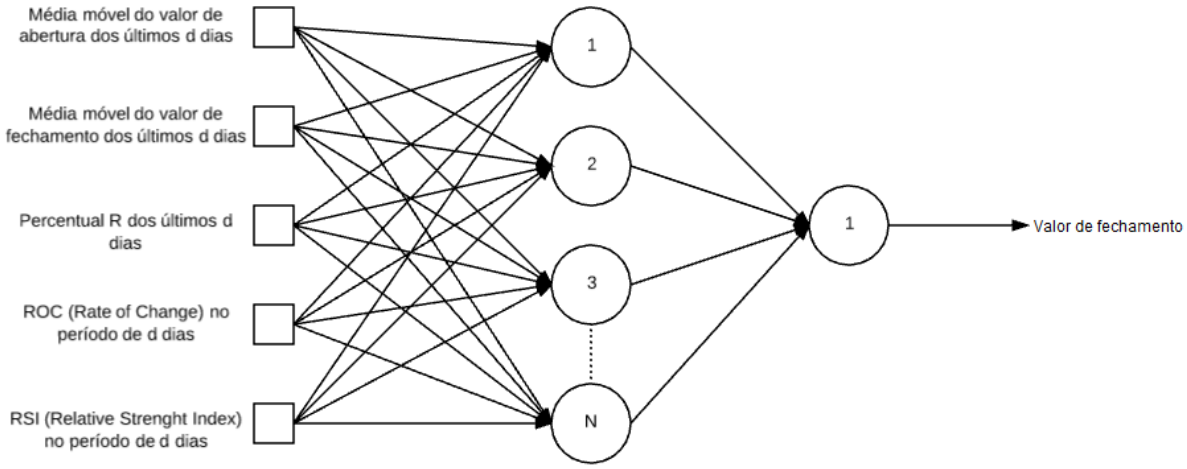
Figura 11 - Modelo 03 de predição do índice da bolsa de valores



Fonte: Acervo do autor, 2017

O último modelo apresentado, está representado pela Figura 12. E assim como as duas primeiras redes, este também terá 5 parâmetros na camada de entrada.

Figura 12 - Modelo 04 de predição do índice da bolsa de valores



Fonte: Acervo do autor, 2017

Como visto na Figura 12, esta rede, além de conter a média móvel dos últimos *d* dias do valor de abertura e fechamento do índice, conterá também outros 3 indicadores técnicos como entrada. Estes indicadores técnicos foram observados no trabalho de Krieger (2012) e são definidos pelo que segue:

- Percentual R: Composto pela seguinte expressão:

$$\%R = 100\ \frac{max(t) - preço(n)}{max(t) - min(t)} \tag{14}$$

Onde segundo Krieger (2012), *max(t)* é o maior valor e *min(t)* o menor valor da ação em um determinado período e *preço(n)* o valor do fechamento da ação no dia.

- ROC (*Rate of Change)*: Este índice pode ser calculado através da seguinte equação:

$$ROC = \frac{[p(t) - BA(d)]}{[BA(d)]} \tag{15}$$

De acordo com Krieger (2012), *p(t)* é o valor do fechamento da ação no dia, *BA* é a média móvel do valor da cotação dos últimos *d* dias.

- RSI (*Relative Strenght Index)*: Este índice é composto pela expressão a seguir:

$$RSI = 100 - \frac{100}{1 + RS(d)} \tag{16}$$

Neste caso, *RS(d)* é a média dos valores que fecharam em alta ou baixa em um período d dias, obtido através da divisão entre a média dos valores que fecharam em alta pela média dos valores que fecharam em baixa, acompanhando o período de *d* dias.

4.6 ÍNDICES PARA ANÁLISE DE QUALIDADE DA PREVISÃO

Após realizar a modelagem e os testes nas RNAs propostas é necessário realizar a medição da qualidade da predição obtida. Para isto, foram empregados dois índices de avaliação,

um deles é o MAPE (*Mean Absolute Percentage Error,* ou Erro Médio Percentual Absoluto) e outro o RMSE (*Root Mean Square Error*, ou Raiz do Erro Quadrático Médio).

O MAPE pode ser calculado através da seguinte expressão:

$$MAPE = \frac{1}{n} \cdot \sum_{k=1}^{n} \frac{|t_k - o_k|}{(t_k + o_k)/2} \cdot 100\% \quad (17)$$

Por sua vez, o RMSE é definido por:

$$RMSE = \sqrt{\frac{\sum_{k=1}^{n}(t_k - o_k)^2}{n}} \quad (18)$$

Onde, de acordo com Zafari, Kianmehr e Abdolahzadeh (2013), para ambos os casos, t_k é a saída esperada na amostra k (valor alvo ou observado), o_k é a saída da rede (valor previsto) para a amostra k e n é o número total de amostras.

Além disto, para ambas as métricas, quanto menor for o valor, melhor o desempenho da rede.

4.7 ESCOLHA DA MELHOR CONFIGURAÇÃO DE RNA

Durante a realização dos treinamentos, várias configurações de RNAs foram modeladas, variando o algoritmo de treinamento, quantidade de neurônios da camada oculta e função de ativação. Cada configuração de RNA foi treinada 30 vezes e com 4 conjuntos de dados diferentes, e dentre estes treinamentos, foi escolhida a rede com menor RMSE, como sendo a melhor rede daquela configuração.

Feito isso para cada configuração determinada, para definir a melhor RNA, foram comparadas as diferentes redes entre si, objetivando identificar as 8 melhores RNAs, ou seja, a rede com melhor MAPE e a rede com melhor RMSE para cada um dos conjuntos de dados apresentados no tópico 4.1.

Após definidas as melhores configurações, foi realizada um comparação entre elas com o intuito de verificar o tipo, configuração e dados de entrada mais aderentes ao problema da presente pesquisa. E a partir deste resultado, ou seja, a partir da eleição da melhor configuração de RNA, foi realizada a comparação com o modelo para consistente.

4.8 COMPARAÇÃO COM O MODELO PARACONSISTENTE

Após definidas a melhor configuração de RNA, foi realizada uma comparação com o modelo de predição da bolsa de valores baseado na Lógica Paraconsistente, proposto por Martins (2012).

Ao contrário da RNA, que prevê o valor de fechamento do índice no dia posterior a previsão, o modelo paraconsistente visa a predição da direção do índice, indicando apenas a subida ou descida do valor do índice no fechamento do dia posterior à predição. Para realizar a comparação entre os dois modelos, foi necessário uma adaptação dos dados retornados pela RNA para que se tenha uma predição com a mesma saída que o modelo paraconsistente. Para isso, uma tabela foi montada com as saídas da RNA indicando a subida ou descida do índice.

Para montar esta tabela, foi utilizado o valor indicado pelo modelo neural e comparado com o valor de fechamento do dia anterior, caso o valor previsto for maior que o valor do dia anterior, significa que a RNA estaria indicando uma subida no valor, caso contrário, há a indicação de uma descida no valor do índice.

Para se ter uma comparação entre as predições, foram usadas as mesmas amostras de testes, e como índice de avaliação para ambos os casos, foi calculado para cada modelo, o percentual de acerto em relação à indicação de subida ou descida do valor do índice.

5. RESULTADOS

Este capítulo apresenta inicialmente uma descrição dos treinamentos das RNAs bem como os resultados obtidos pelas mesmas. Além disto, busca-se também definir qual das Redes Neurais, apresentou melhor resultado. Posteriormente é descrito a estratégia utilizada para a predição utilizando-se da lógica paraconsistente bem como os resultados obtidos por tal método. Por fim, são analisadas e comparadas as melhores RNAs com o método paraconsistente, visando identificar as vantagens e desvantagens de cada um dos métodos.

5.1 TREINAMENTO E RESULTADOS DAS RNAS

As subseções que seguem descrevem o processo de treinamento e os resultados obtidos com a utilização das RNAs *Multilayer Percpetron.* O algoritmo elaborado para o treinamento e teste das redes MLP é descrito no Apêndice A.

5.1.1 Treinamento

Existem diversos parâmetros iniciais a serem configurados ao realizar o treinamento de uma rede MLP, como os algoritmos de treinamento, que para este trabalho foram usados alternadamente os métodos TrainBR e TrainLM, as funções de ativações, sendo alternadas entre a LOGSIG e TANSIG, e a quantidade de neurônios da camada oculta, que foi variado entre 3 e 15 neurônios. Ao se utilizar a *toolbox* de RNA do software Matlab, os parâmetros iniciais de treinamento não foram modificados, permanecendo os valores padrão do Matlab.

Em ambos os algoritmos, não foi utilizado como condição de parada o tempo máximo de treinamento, porém foram utilizados outros critérios de parada, que são os seguintes:

- Máximo de épocas de treinamento em 1.000 iterações;
- O desempenho/performance da rede atingir a meta de desempenho (valor padrão: 0)
- O desempenho do gradiente ser atingido (valor padrão: 1^{-07})
- O desempenho da validação aumentou mais que o máximo de verificações da validação desde a última vez que diminuiu (quando utilizada a validação)

Segundo Beale, Hagane e Demuth (2016), no algoritmo TrainBR, por padrão a validação é desabilitada, assim sendo, o treinamento continua até que a melhor combinação de erros e pesos seja encontrada. De modo oposto, o algoritmo TrainLM, utiliza os vetores de validação para finalizar o treinamento da rede mais rápido, caso seu desempenho nos vetores de validação permanece o mesmo e não a há melhora durante algumas épocas consecutivas.

Para avaliar o desempenho das RNAs na fase de treinamento é realizado o cálculo do MSE, descrito no tópico 2.2.4.1, que é usado como função de desempenho neural padrão do Matlab. Vale ressaltar que, quanto mais próximo de zero for o valor do MSE, melhor é o desempenho da rede, visto que esse índice é referente ao Erro Quadrático Médio (*Mean Square Error*) da rede.

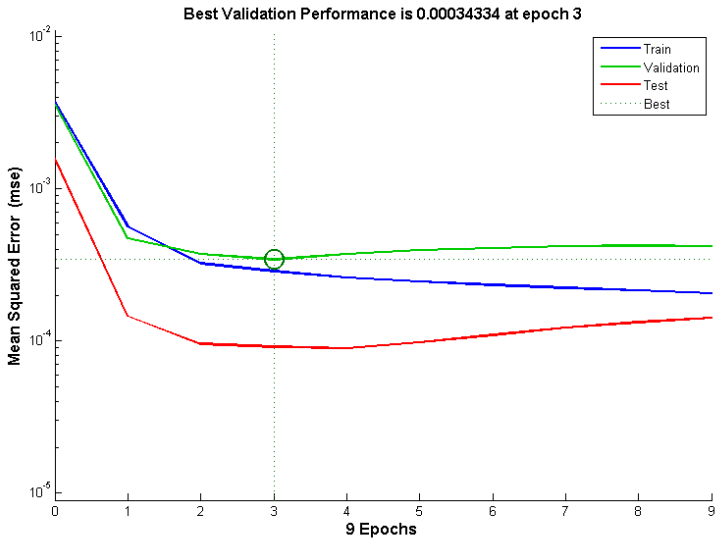
As subseções que seguem, abordam as configurações da RNA MLP que obtiveram o melhor desempenho nos treinamentos para previsão do índice S&P 500, subdividindo os resultados pelo grupo de dados de treinamento utilizado.

5.1.1.1 Dados de 6 meses

Neste conjunto de treinamento, foram repassados para as redes dados de 6 meses da bolsa de valores, entre as datas de 20/10/2015 até 19/04/2016. Para este conjunto de treinamento, as duas melhores redes, ou seja, a rede com melhor RMSE e com melhor MAPE foram referentes ao quarto modelo de RNA apresentado no tópico 4.5, utilizando 50 dias para os valores das médias móveis.

A rede com menor RMSE, utilizou o algoritmo de treinamento TrainLM, com função de ativação LOGSIG e 11 neurônios na camada oculta. O RMSE obtido por esta rede foi de 0.00956 e o seu MAPE foi de 0.8787. No Gráfico 1 é apresentado a evolução do erro desta rede, pode ser observar que com apenas 9 épocas de treinamento houve a convergência do algoritmo.

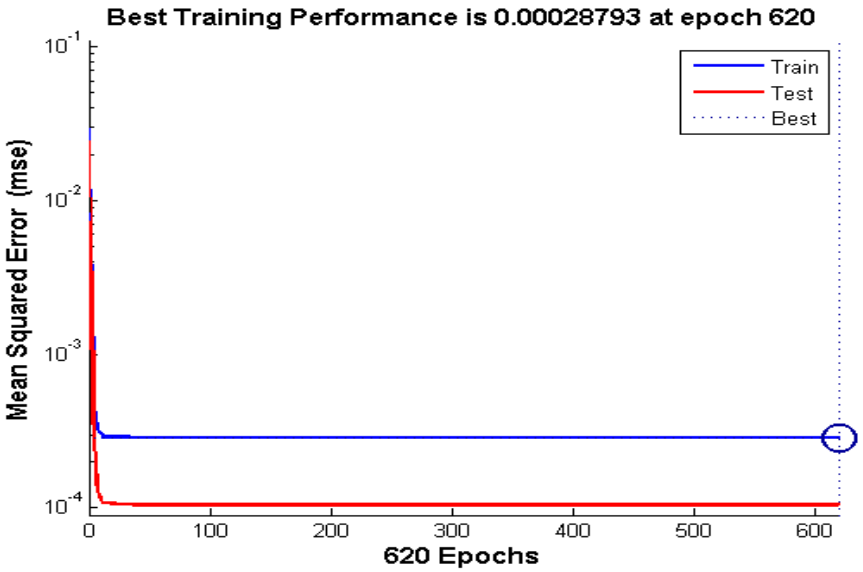
Gráfico 1 – Evolução do erro da MLP com menor RMSE para 6 meses de dados.



Fonte: Acervo do autor, 2017.

Já a rede com menor MAPE utilizou o algoritmo de treinamento TrainBR, função de ativação LOGSIG e 13 neurônios na camada oculta. Seu RMSE foi de 0.01022 e seu MAPE de 0.8594. Assim como no caso anterior, o Gráfico 2 apresenta a evolução do Erro Quadrático Médio desta rede, onde foi necessário um total de 620 épocas para se ter a convergência.

Gráfico 2 - Evolução do erro da MLP com menor MAPE para 6 meses de dados.



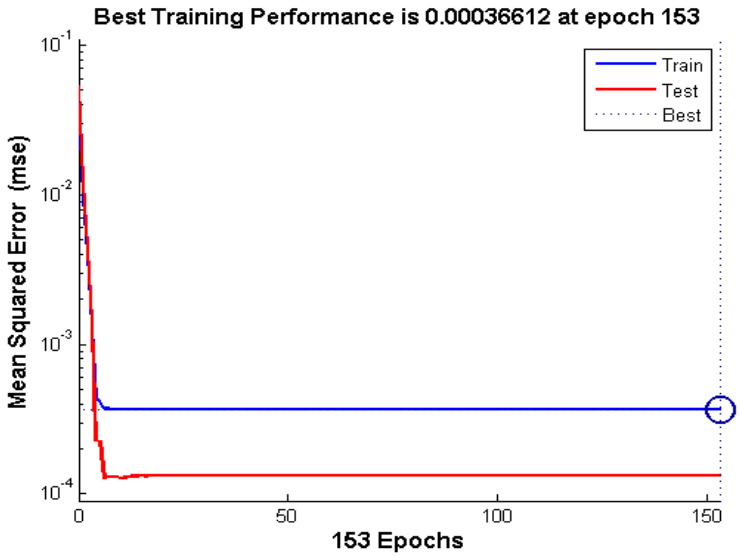
Fonte: Acervo do autor, 2017.

Como acompanhado nos dois gráficos, a rede com menor RMSE, que utilizou o algoritmo de treinamento TrainLM teve uma convergência muito mais rápida neste caso do que a outra rede com menor MAPE que utilizou o algoritmo de treinamento TrainBR. Porém, mesmo assim, ambas as redes apresentaram resultados satisfatórios na predição dos índices S&P 500.

5.1.1.2 Dados de 1 ano

Para realizar este teste foram utilizados dados entre as datas de 20/04/2015 até 19/04/2016, totalizando um ano de informações sobre a bolsa de valores. Os mesmos modelos foram treinados novamente com este conjunto de dados. Nesta série de treinamentos, houve a coincidência de que a rede com menor RMSE também obteve o menor MAPE. Esta RNA apresentou um RMSE de 0.01151 e MAPE de 1.2011, o Gráfico 3 mostra a evolução do MSE no treinamento, precisando de 153 épocas para atingir a convergência.

Gráfico 3 - Evolução do erro da MLP com menor RMSE e MAPE para 1 ano de dados.



Fonte: Acervo do autor, 2017.

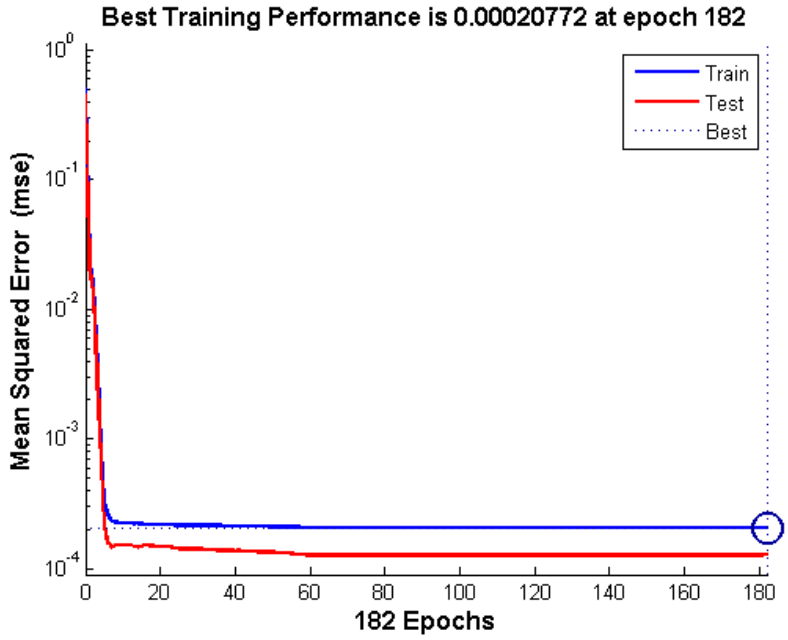
Esta rede com melhor desempenho para este conjunto de dados, seguiu o modelo 1 apresentado no tópico 4.5 e possuiu 5 neurônios na camada oculta, algoritmo de treinamento TrainBR e função de ativação LOGSIG. Em comparação com os modelos

anteriores, já se percebe que as redes que foram treinadas com o algoritmo TrainBR, tendem a levar mais épocas de treinamento para convergirem.

5.1.1.3 Dados de 3 anos

Para este terceiro teste, foram utilizados dados entre as datas 22/04/2013 e 19/04/2016 totalizando 3 anos de informações sobre o índice. As duas melhores redes, com menor RMSE e a com menor MAPE seguiram o quarto modelo apresentado no tópico 4.5 porém, diferentemente do resultado do primeiro teste, neste caso, foram usados 20 dias para as médias móveis. Ambas as redes possuíram 11 neurônios na camada oculta e função de ativação LOGSIG, porém a rede com menor RMSE, utilizou o algoritmo de treinamento TrainBR, enquanto a com menor MAPE utilizou o TrainLM. O Gráfico 4 mostra a evolução do Erro Médio Quadrático para a rede com o menor RMSE, apresentando 182 épocas de treinamento.

Gráfico 4 - Evolução do erro da MLP com menor RMSE para 3 anos de dados.

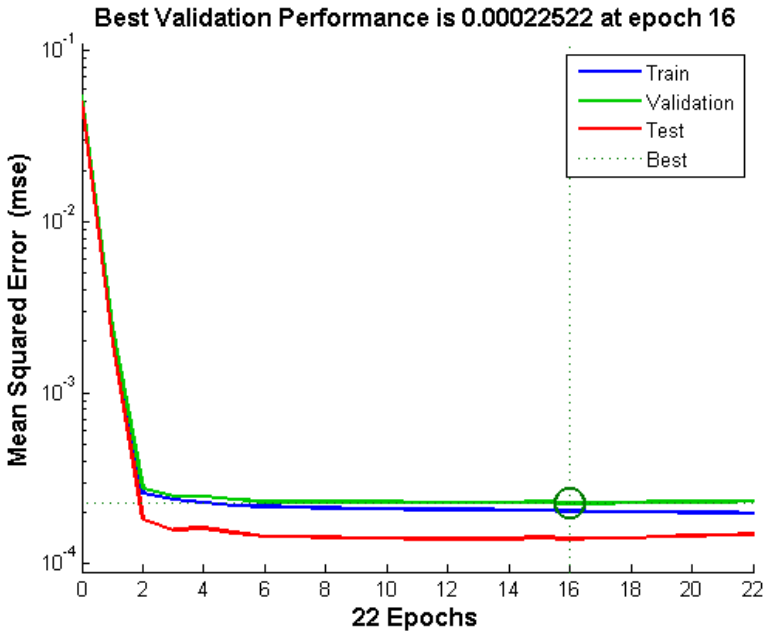


Fonte: Acervo do autor, 2017

Esta rede apresentou o RMSE de 0. 011271 e MAPE de 1.1433, enquanto a rede que obteve melhor MAPE, apresentou um RMSE de 0.011856 e MAPE de 1.1427,

totalizando 22 épocas de treinamento para atingir a convergência. O Gráfico 5 demonstra a evolução do erro para a segunda rede.

Gráfico 5 - Evolução do erro da MLP com menor MAPE para 3 anos de dados.



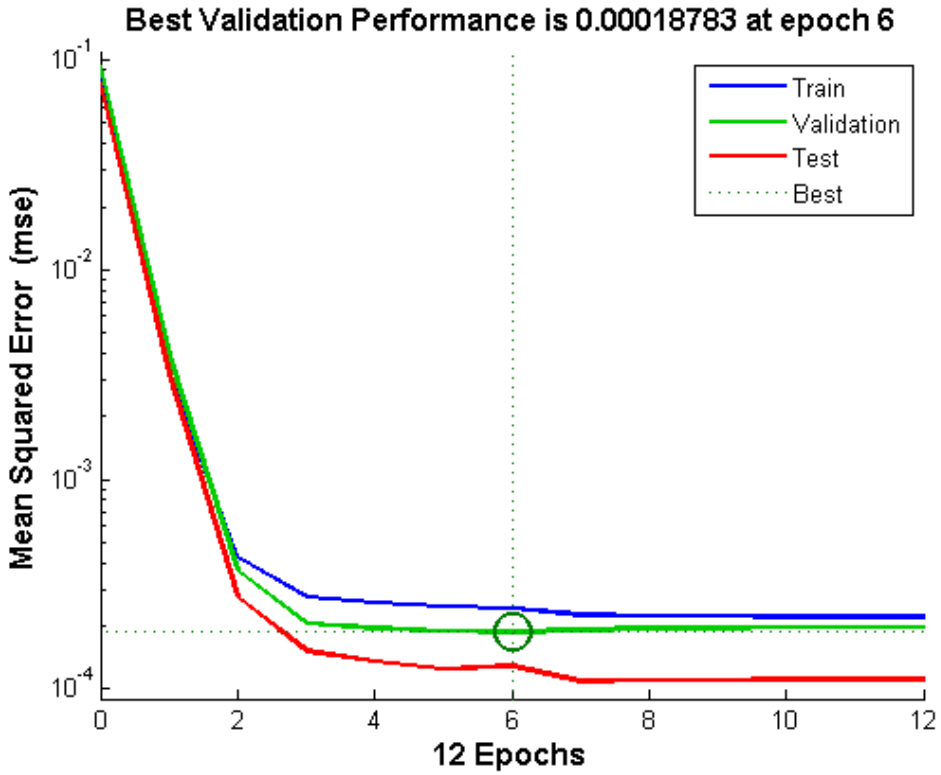
Fonte: Acervo do autor, 2017.

Em comparação ainda entre os algoritmos de treinamento, se torna perceptível que o algoritmo TrainLM possui uma curva de convergência muito menor do que a do algoritmo TrainBR.

5.1.1.4 Dados de 5 anos

Por fim, este treinamento levou em considerações dados de 5 anos entre 20/04/2011 e 19/04/2016. Neste caso, a rede com menor RMSE seguiu o modelo 4 apresentado no tópico 4.5 e utilizou 50 dias uteis para as médias móveis. O algoritmo de treinamento foi o TrainLM com função de ativação LOGSIG e 11 neurônios na camada oculta. O RMSE resultante desta rede foi de 0.011324 e seu MAPE de 2.2694. A evolução do erro para esta RNA pode ser acompanhado pelo Gráfico 6, onde nota-se que para adquirir a convergência, foi necessária 12 épocas de treinamento.

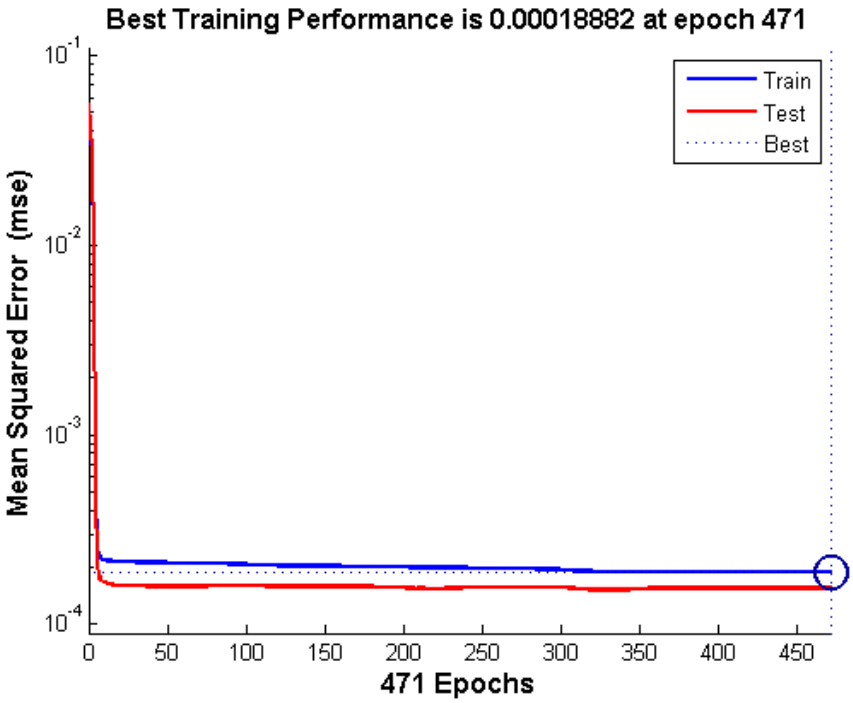
Gráfico 6 - Evolução do erro da MLP com menor RMSE para 5 anos de dados.



Fonte: Acervo do autor, 2017.

Assim como a rede anterior, a RNA com menor MAPE, também utilizou o quarto modelo de RNA, porém neste caso foram usados 5 dias para as médias móveis. O algoritmo de treinamento neste caso foi o TrainBR, com função de ativação LOGSIG e 9 neurônios na camada oculta. O RMSE total desta rede foi de 0.012453 enquanto o MAPE foi de 2.1746. No Gráfico 7, pode ser observado a evolução do Erro Médio Quadrático, é notável também que foram necessárias 471 épocas de treinamento para se atingir a convergência. Além disto como foi notado, em todos os casos, as redes que usam o algoritmo de treinamento TrainBR, necessitaram de muito mais épocas de treinamento para atingir a convergência, em relação aos treinamentos que utilizaram o algoritmo TrainLM.

Gráfico 7 - Evolução do erro da MLP com menor MAPE para 5 anos de dados.



Fonte: Acervo do autor 2017.

5.1.2 Resultados

Após obtidas as sete melhores redes apresentadas nas subseções anteriores, foram realizados testes nestas redes com dados nunca apresentados para as RNAs, estes dados compreendem 6 meses de informações sobre o índice entre as datas de 20/04/2016 a 20/10/2016. Em síntese, a Tabela 2 mostra as configurações das sete melhores RNAs apresentadas anteriormente, bem como seu RMSE e MAPE.

Tabela 2 – Sete melhores combinações de RNAs

Rede	Mod.	Méd. Móv.	Alg. Trein.	Fun. Ativ.	Nr. Neur.	Épocas	RMSE	MAPE
1	04	50 dias	TrainLM	LOGSIG	11	9	0.00956	0.8787
2	04	50 dias	TrainBR	LOGSIG	13	620	0.01022	0.8594
3	01	N/A	TrainBR	LOGSIG	5	153	0.01151	1.2011
4	04	20 dias	TrainBR	LOGSIG	11	182	0.01127	1.1433
5	04	20 dias	TrainLM	LOGSIG	11	22	0.01186	1.1427
6	04	50 dias	TrainLM	LOGSIG	11	12	0.01132	2.2694
7	04	5 dias	TrainBR	LOGSIG	9	471	0.01245	2.1746

Na Tabela 2 estão mostrados em síntese os mesmos modelos apresentados nas subseções anteriores, sendo os dois primeiros referentes ao conjunto de dados de 6 meses, o terceiro referente a 1 ano, o quarto e quinto referentes a 3 anos e os dois últimos referentes a 5 anos. É interessante observar que, conforme aumentou o número de amostras no conjunto de treinamento, o MAPE e o RMSE tenderam a aumentar gradativamente, o que pode significar que um maior conjunto de dados pode trazer um erro maior, porém neste caso o aumento é pouco e quase insignificativo.

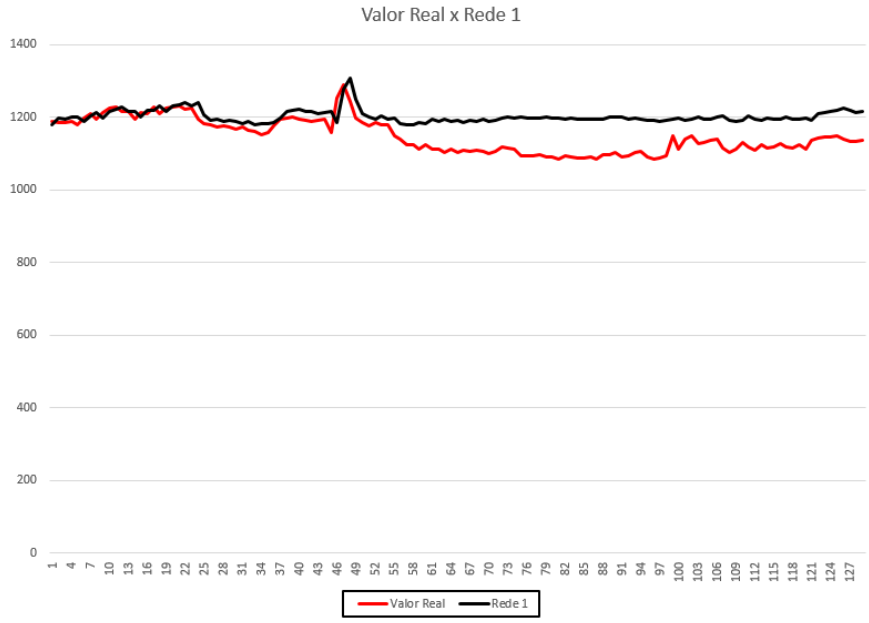
Estas 7 Redes Neurais foram aplicadas a dados nunca visto por elas antes, e com o intuito de verificar a acurácia destas redes e selecionar a melhor dentre elas, decidiu-se calcular o RMSE e o MAPE de cada uma delas, referente a predição delas em relação aos valores reais. Além disso, realizou-se também uma comparação para cada rede, em relação a direção da predição, ou seja, através do valor previsto pela rede e do último valor conhecido, projeta-se um sentido para um índice, sendo este sentido um sinal de que o índice irá subir ou cair.

Para realizar a predição da direção do índice, foram utilizadas duas estratégias, a primeira delas é comparando o valor previsto pela rede com o valor real de fechamento do dia anterior, caso o valor previsto seja maior, significa que a rede está prevendo uma alta do índice, caso contrário, uma queda. A outra estratégia que observou-se promissora, foi ao invés de comparar o valor previsto com o último valor real, compara-lo com o último valor previsto pela própria rede. Isto pelo fato de que, mesmo que a rede possa estar errando o valor do índice, ela irá estar se aproximando da variação do mesmo. A precisão da previsão da direção do índice vai ser calculada através do percentual de acertos em que a rede de fato previu a direção do índice.

Com o objetivo de comparação, antes de apresentar os RMSEs, MAPEs e percentuais de acerto, é apresentado os gráficos de cada uma das sete redes em relação ao valor real do índice, onde em cada gráfico, o eixo x representa o n-ésimo dia de predição e o eixo y o valor do índice no dia.

A seguir estão apresentados os gráficos para cada uma das redes, o Gráfico 8 mostra a comparação entre os valores reais e os valores obtidos através da predição utilizando a RNA 1, mostrada na Tabela 2.

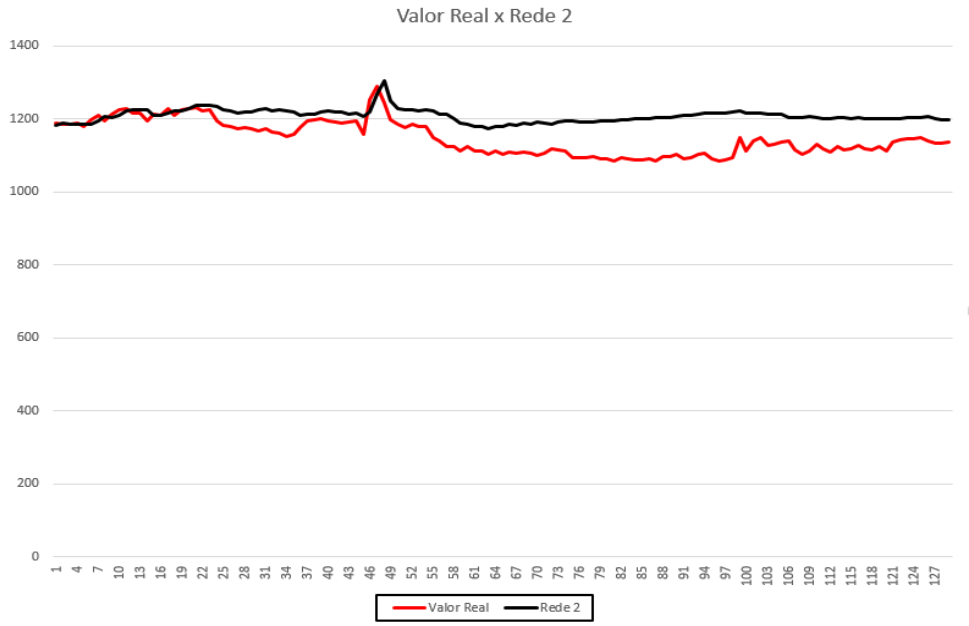
Gráfico 8 – Valores reais e previstos pela Rede 1 (ver tabela 2)



Fonte: Acervo do autor.

A seguir, o Gráfico 9, mostra a relação do valor real com o valor previsto pela RNA 2, pode-se observar nestes dois primeiros testes que até aproximadamente o 50º dia as previsões estavam acompanhando pelo menos as direções do índice real, porém após este dia, ambas perderam precisão e não acompanharam muito bem o índice.

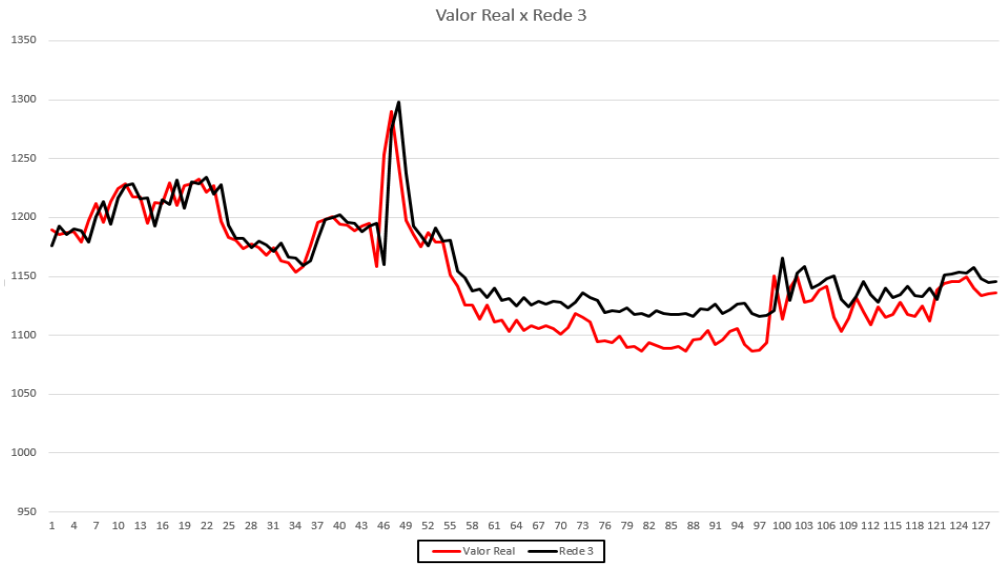
Gráfico 9 - Valores reais e previstos pela Rede 2 (ver tabela 2)



Fonte: Acervo do autor, 2017

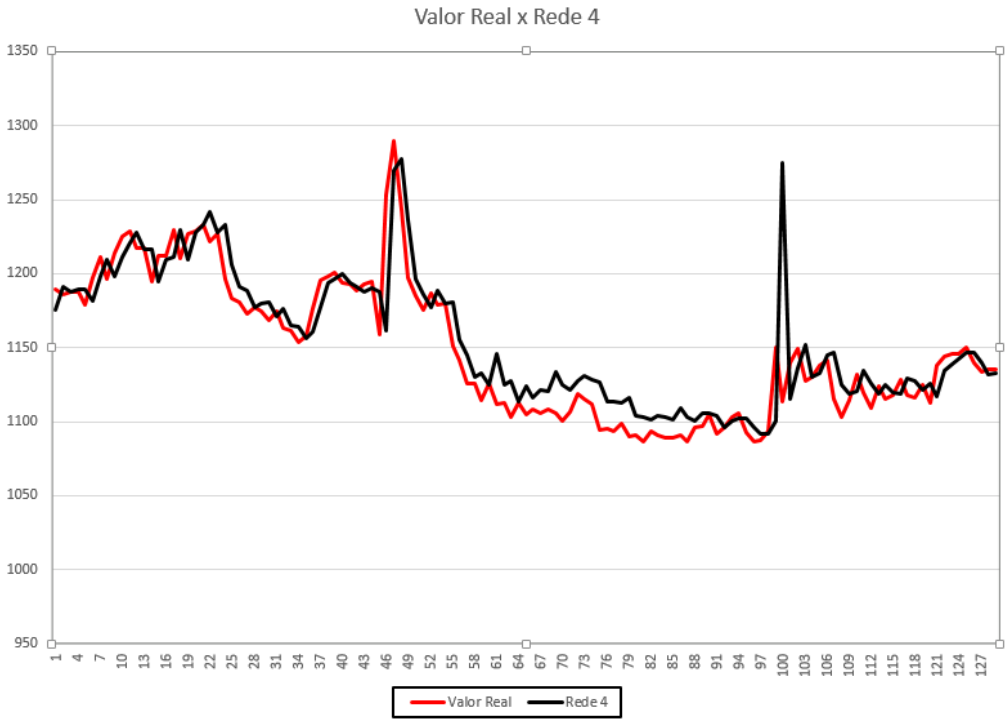
O Gráfico 10, mostra a comparação com os resultados da terceira RNA, nota-se que aqui, houve uma melhora na predição, pois o valor previsto varia de forma bem parecida com o valor real, o que é um ponto positivo, pois assim torna-se mais fácil prever a direção do índice. Já o Gráfico 11, mostra os resultados para a quarta RNA apresentada, onde também nota-se bastante semelhança na variação dos valores entre o valor real e o previsto.

Gráfico 10 - Valores reais e previstos pela Rede 3 (ver tabela 2)



Fonte: Acervo do autor, 2017.

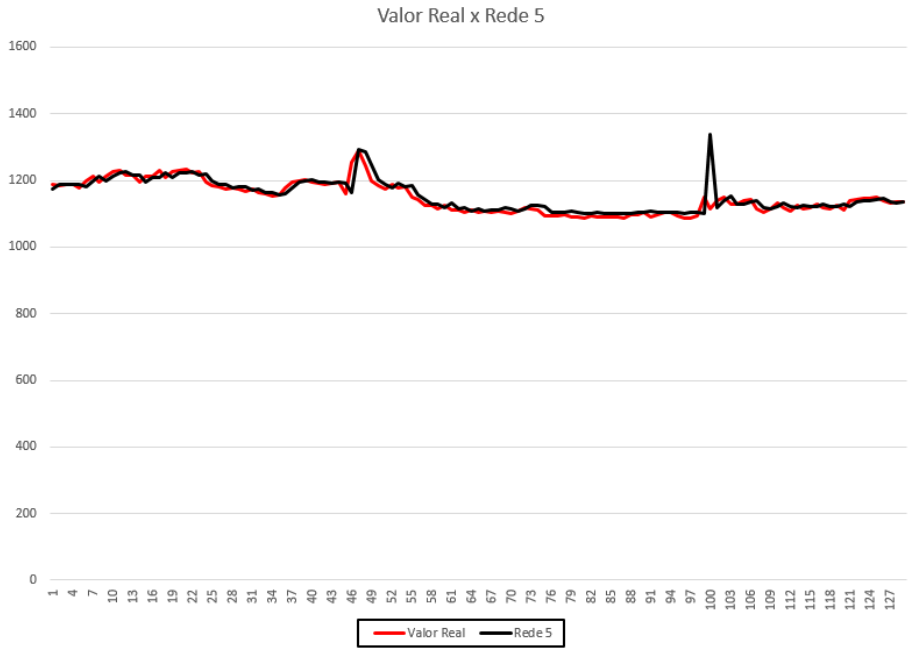
Gráfico 11 - Valores reais e previstos pela Rede 4 (ver tabela 2)



Fonte: Acervo do autor, 2017.

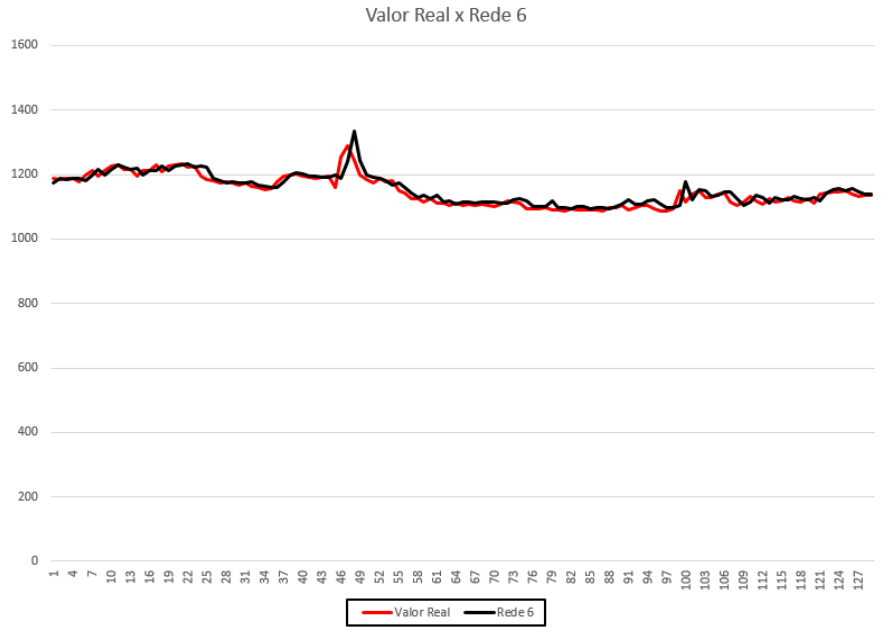
O Gráfico 12, mostra os resultados para a RNA 5, neste caso, pode-se observar certa semelhança com o modelo anterior nas proximidades do dia 100, onde há um pico na predição, porém o mesmo não aconteceu com o valor real, porém observa-se que mesmo com este pico, poucos dias depois as redes voltaram para bem próximo do valor real.

Gráfico 12 - Valores reais e previstos pela Rede 5 (ver tabela 2)



Fonte: Acervo do autor, 2017.

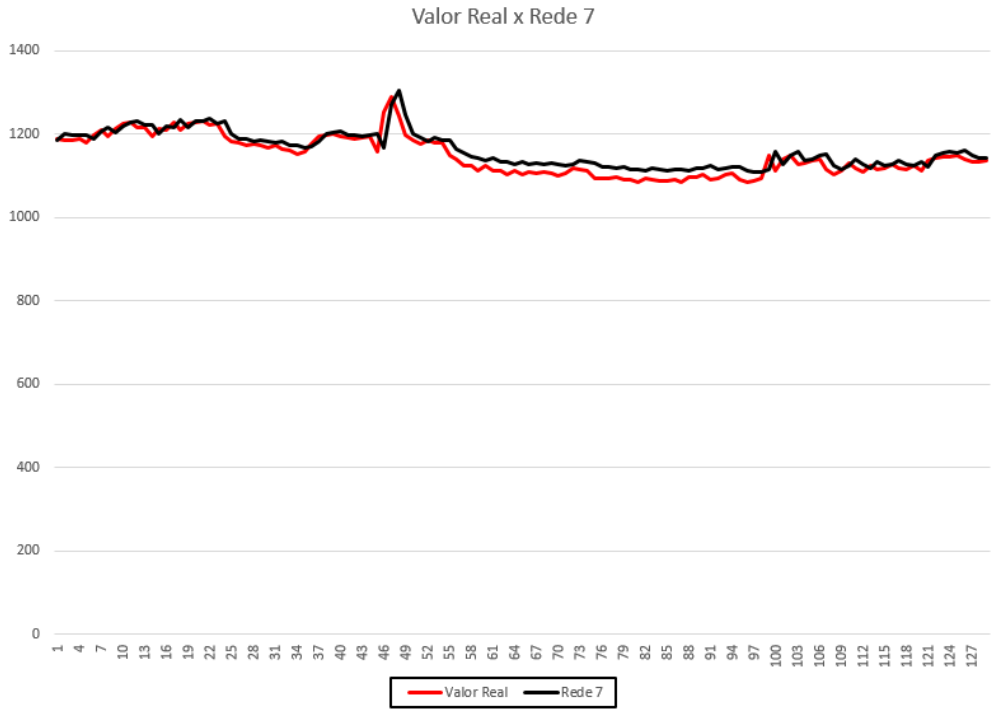
Gráfico 13 - Valores reais e previstos pela Rede 6 (ver tabela 2)



Fonte: Acervo do autor, 2017.

O Gráfico 13 apresentou os resultados através da utilização da sexta RNA e por fim, o Gráfico 14 apresenta os valores obtidos através da sétima rede. Nota-se que estas duas últimas redes apresentam comportamento bastante semelhante, seguindo de forma muito parecida os valores reais do índice.

Gráfico 14 - Valores reais e previstos pela Rede 7 (ver tabela 2)



Fonte: Acervo do autor, 2017.

Como mencionado anteriormente, além de medir a precisão desta Redes Neurais com os valores do RMSE e do MAPE, foi medido o percentual de acerto em relação a predição da direção do índice, ou seja, através dos valores previstos do índice buscou-se analisar se o acerto em relação a alta ou baixa do índice. As duas estratégias para a predição da direção tiveram sua precisão medida através do percentual de acerto da mesma. Em relação à segunda estratégia, pôde ser observado nos gráficos anteriores que de fato, os valores previstos pela rede tendem a seguir o movimento do próprio valor real, o que significa que, ao tentar predizer a direção do índice, torna-se interessante a utilização da variação em relação aos valores previstos pela própria rede, e não em relação aos valores reais.

A Tabela 3 mostra o RMSE, MAPE e o percentual de acerto para a primeira estratégia de predição de direção (comparando o previsto com o valor real) e também da segunda estratégia (comprando o previsto com o último valor previsto).

Tabela 3 – Resultados da aplicação das melhores RNAs

Rede	RMSE	MAPE	Perc. Acerto Estratégia 1	Perc. Acerto Estratégia 2
1	0.003683	5.30356	52.7131%	38.7596%
2	0.004231	5.87288	53.4883%	57.3643%
3	0.000385	1.64757	50.3875%	38.7596%
4	0.000417	1.35756	52.7131%	44.1860%
5	0.000535	1.26970	51.1162%	44.1860%
6	**0.000299**	**1.25339**	**54.2663%**	**43.4108%**
7	0.000400	1.71788	52.7131%	42.6356%

Como visto na Tabela 3, ambas as redes selecionadas apresentaram um baixo RMSE e MAPE, porém as duas primeiras obtiveram valores um pouco superiores aos das outras redes, o que pode explicar o fato de após o 50^{o} dia elas não conseguirem seguir o padrão do valor real. As demais redes apresentaram um RMSE e MAPE bem parecidos, o percentual de acerto entre elas também foi semelhante.

Ao observar a tabela, nota-se que em relação a predição da direção do índice, torna-se mais interessante utilizar a primeira estratégia, pois ela manteve um percentual de acertos constante em todas as redes, embora a rede 2, teve o maior percentual de acerto com a segunda estratégia o MAPE dela se apresenta muito alto, o que torna duvidoso o uso desta RNA.

Em suma, a melhor RNA a ser escolhida é a sexta, pois obteve o menor RMSE e menor MAPE nos testes, além de ter tido o melhor percentual de acertos da direção do índice para a estratégia 1. Esta RNA, utilizou o quarto modelo apresentado na seção 4.5, com 50 dias para as médias móveis, algoritmo de treinamento TrainLM, função de ativação LOGSIG e 11 neurônios na camada oculta.

Além disto, como observou-se, o modelo 4 foi o que teve melhor desempenho, talvez isso se dê pelo fato da utilização dos indicadores técnicos, como Percentual R, ROC e RSI, que neste caso irão dar foco para a variação dos preços do índice, e não apenas ao valor do índice.

5.2 ESTRATÉGIA PARACONSISTENTE

As sequentes subseções descrevem sucintamente a estratégia paraconsistente para predição da direção do índice da bolsa de valores, assim como descrito por Martins (2012) e também demonstram os resultados obtidos através da aplicação no mesmo conjunto de dados

utilizado para testes das RNAs, compreendendo as datas de 20/04/2016 a 20/10/2016. Para facilitação da obtenção dos resultados desta estratégia foi criado um algoritmo em MATLAB, cujo elaboração está descrita no Apêndice B.

5.2.1 Estratégia e treinamento

Para o treinamento da técnica utilizando a lógica paraconsistente foram utilizados dados entre as datas 09/09/1997 e 19/04/2016, pois acredita-se de acordo com Martins (2012) que um maior número de amostras pode trazer maior precisão para o modelo.

A estratégia paraconsistente trabalha com médias móveis para realizar as predições. São utilizadas combinações de médias móveis dos valores de fechamento de 1, 2, 5, 50, 150 e 200 dias. Essas médias móveis indicam um sinal de subida ou descida do índice, ou seja, se o valor da média móvel for maior que o fechamento do dia anterior, então o sinal será de compra (alta), pois o valor do índice terá um aumento, caso contrário o sinal será de venda (baixa).

Para realizar a estratégia então, são armazenados o sentido do índice para 5 modelos. Os três primeiros, comparando o valor da média móvel de 50, 150 e 200 dias com o valor do fechamento do dia anterior e os dois últimos comparando o valor da média móvel de 150 e 200 dias com o valor da média móvel dos últimos 5 dias. Estes sinais são armazenados durante o período de treinamento e além disto é calculado o erro para cada um dos 5 modelos, sendo este, o acerto ou não da direção do índice.

Estes dados mencionados são todos coletados durante o período de treinamento, ao entrar no período de testes, o modelo calcula a viabilidade de cada uma das séries temporais a partir do histórico de acerto dos últimos 5 dias, e de forma agregada, caso pelo menos metade dos modelos indiquem uma compra (alta) o modelo em si irá indicar uma compra, caso contrário indicará uma venda (baixa).

O modelo também emite um sinal *lambda* que se refere ao nível de viabilidade daquela predição, ou seja, o valor de *lambda* é o grau de viabilidade do sinal de compra ou venda do modelo. A estratégia paraconsistente circunda conceitos em torno do valor de *lambda*, onde o modelo irá discriminar a predição como viável, inviável ou inconclusiva. Para realizar tal procedimento, há uma variável dita pelo modelador, que é o nível de exigência, onde para tal modelo de predições decidiu-se realizar testes com 60, 70, 80, 90 e 100% de nível de exigência.

Para a formação do sinal *lambda* foram divididos os 5 grupos de médias móveis descritos acima em 4 grupos, denominados respectivamente, A, B, C e D. Para cada um dos

grupos foram calculadas duas variáveis, denominadas no trabalho de Martins (2012) de *mu_1* e *mu_2*. No grupo A, para *mu_1* foi calculado a média de acertos dos últimos 5 dias do sinal de subida ou descida das médias móveis de 200 dias comparando-a com o valor de fechamento do dia anterior, onde *mu_2* é complementação de *mu_1*, sendo que seu valor é formado por *1 – mu_1*. Por exemplo, caso o valor de *mu_1* tenha sido 0.3, o valor de *mu_2* será *1 – 0.3 = 0.7*.

Os Grupos B e C, tiveram suas variáveis *mu_1* e *mu_2* calculadas da mesma forma que para o grupo A, porém ao invés de utilizar as médias móveis de 200 dias comparando com o valor de fechamento do dia anterior, os Grupos B e C utilizaram respectivamente as médias móveis de 150 e 200 dias, comparando-as com a média móvel do valor de fechamento dos 5 dias anteriores.

O Grupo D, apresenta a ideia de maximização da lógica paraconsistente, denotada pelo operador OR. Ou seja, o coeficiente *mu_1* do Grupo D foi a maximização da média de acertos dos grupos de médias móveis de 50 e 150 dias, comparadas com o valor de fechamento do dia anterior. Em outras palavras, o *mu_1* do Grupo D, foi a maior média de acerto entre os grupos de médias móveis de 50 e 150 dias. E de forma semelhante aos outros grupos, o *mu_2* tem seu valor complementar ao de *mu_1*, sendo formado por *1 – mu_1*.

Formado os valores de *mu_1* e *mu_2* para os 4 grupos, foi formado um grupo chamado de resultante, onde este grupo também apresenta as variáveis *mu_1* e *mu_2*. Neste grupo há a ideia de minimização da lógica paraconsistente, denotada pelo operador AND. Para este caso o valor de *mu_1* é o menor valor de *mu_1* para os 4 grupos, A, B, C e D. Já o valor de *mu_2* é formado pelo menor valor de *mu_2* dos grupos A, B, C e D.

A partir dos valores de *mu_1* e *mu_2* do grupo resultante calcula-se então valor do sinal *lambda* através da diferença entre *mu_1* e *mu_2 (mu_1 – mu_2)*, que resultará num valor entre -1 e 1, ou seja, $-1 \leq lambda \leq 1$.

O nível de exigência trabalha em conjunto com o valor de *lambda*, por exemplo, caso o nível de exigência seja de 60, a classificação da predição se daria da seguinte forma: caso o valor de *lambda* for maior ou igual que 0.6 (60 dividido por 100) então a predição é viável, porém caso o valor de *lambda* for inferior a -0.6 então a predição é inviável. Agora, se o valor de *lambda* permanecer entre -0.6 e 0.6 então a predição torna-se inconclusiva, ou seja, o modelo não pode agir sobre o mercado, pois não é possível formar uma conclusão.

A partir dessa classificação denotada pelo valor de *lambda* é que o modelo irá definir a direção do índice. Para realizar esta predição o modelo seguirá as seguintes regras:

- Caso o modelo preveja compra (alta) e o valor de *lambda* indique a predição como viável, então o modelo indicará uma compra (alta) do índice.
- Caso o modelo preveja compra (alta) e o valor de *lambda* indique a predição como inviável, então o modelo indicará uma venda (baixa) do índice.
- Caso o modelo preveja venda (baixa) e o valor de *lambda* indique a predição como viável, então o modelo indicará uma venda (baixa) do índice.
- Caso o modelo preveja venda (baixa) e o valor de *lambda* indique a predição como inviável, então o modelo indicará uma compra (alta) do índice.
- Caso o valor de *lambda* indique a predição como inconclusiva, então o modelo não indicará nem compra (alta) nem venda (baixa) do índice.

5.2.2 Resultados

A precisão do modelo paraconsistente foi medida através do percentual de acertos em relação à direção do índice, ou seja, quando o modelo indicou uma compra (alta) e de fato o valor do índice subiu, ou quando o modelo indicou uma venda (baixa) e de fato o valor do índice desceu.

Porém há um certo problema quando o modelo indica uma predição inconclusiva, pois neste caso o modelo não indica nem compra (alta) nem venda (baixa), por isso, os percentuais de acertos foram divididos em dois grupos, um deles levando em consideração os dias inconclusivos como erro do modelo e outro, desprezando os dias inconclusivos.

Além disto foram realizados testes com os seguintes níveis de exigência: 60%, 70%, 80%, 90% e 100%.

A Tabela 4 demonstra os resultados obtidos pela estratégia baseada na lógica paraconsistente, apresentando os percentuais de acerto levando em consideração os dias inconclusivos e os percentuais desprezando tais dias.

Tabela 4 – Resultados para a estratégia paraconsistente.

Nível Exigência	**Perc. Acerto Com Dias Inconclusivos**	**Perc. Acerto Sem Dias Inconclusivos**
60%	35.6589%	51.1111%
70%	12.4031%	48.4849%
80%	12.4031%	50%
90%	0%	0%
100%	0%	0%

Como pôde ser observado, conforme o nível de exigência aumenta para o percentual de acerto com dias inconclusivos, a precisão tende a diminuir, porém desprezando os dias inconclusivos, a precisão manteve-se estável. Outro ponto a ser observado é que com 90% e 100% de precisão, o percentual de acerto foi nulo isto por que em ambos os casos, houve apenas 1 dia em que o modelo não foi inconclusivo, e neste mesmo dia, o modelo errou a predição, o que talvez configura-se não muito interessante a utilização de um nível de exigência muito alto.

O melhor dos modelos, como apresentado na Tabela 4, foi o modelo com um nível de exigência de 60%, este modelo acerto 35.6589% dos casos, contando com os dias inconclusivos e obteve um percentual de acerto de 51.1111% ao se desprezar os dias inconclusivos.

5.3 COMPARAÇÃO DOS RESULTADOS

Ao compararmos os resultados obtidos pela melhor RNA, com o melhor modelo baseado na lógica paraconsistente, temos que nos ater ao detalhe de que as RNAs predizem o valor para o qual o índice caminhará, porém a estratégia paraconsistente prevê a direção do índice, ou seja, se o valor do mesmo aumentará ou diminuirá no dia posterior.

Por isso, para realizar a comparação dos dois modelos, foi acrescentado nos modelos baseados em Redes Neurais Artificiais o sinal de subida e descida do índice assim como explicado no tópico 5.1.2. Deste modo é possível comparar os percentuais de acerto em relação a predição da direção do índice, tanto para o melhor modelo neural quanto para o melhor modelo paraconsistente.

Como visto anteriormente a melhor RNA, referente ao quarto modelo do tópico 4.5 deste trabalho, utilizando 50 dias para as médias móveis, algoritmo de treinamento TrainLM, função de ativação LOGSIG e 11 neurônios na camada oculta, obteve 54.2663% de acerto, quando a predição utiliza o valor real do índice para realizar a comparação de subida e decida, e obteve um percentual de 43.4108% quando a comparação foi realizada através da comparação utilizando o próprio valor previsto do índice.

Já em relação ao modelo paraconsistente, tem-se que com nível de exigência de 60%, considerando os dias em que o modelo se apresentou inconclusivo, houve um percentual de acerto de 35.6589%, já ao desprezar os dias inconclusivos, houve um acerto de 51.1111%

Em comparação entre os dois métodos, vê-se que a melhor RNA teve desempenho melhor que os dois modelos paraconsistentes, já o melhor modelo paraconsistente, se saiu

melhor que a RNA contando a predição através da comparação com o valor previsto. Porém a desvantagem deste modelo paraconsistente é que o mesmo não irá prever para todos os dias, pois em alguns casos ele poderá se demonstrar inconclusivo e não apresentar predição, enquanto a RNA irá realizar a predição todos os dias, tentando manter o percentual de acerto estável.

Porém podemos analisar os resultados da lógica paraconsistente com um problema nos dados utilizados, pois como os dados de treinamento são desde 1997 significa que o período de treinamento passou pela fase da crise de 2008 ocorrida nos Estado Unidos, o que pode ter afetado o resultado do modelo, desviando os padrões do índice.

Olhando para os resultados obtidos por este trabalho, vê-se certa vantagem na utilização de Redes Neurais Artificiais para a predição de índices da bolsa de valores, isto pelo fato, dela ter apresentado um percentual de acerto maior que o apresentado pela lógica paraconsistente, além de se manter mais estável, conseguir realizar a predição em todos os dias, não tendo o estado de inconclusiva, como na lógica paraconsistente.

6. CONCLUSÕES

Dentre as diversas técnicas de Inteligência Artificial existem as Redes Neurais Artificiais, que têm como principal característica a capacidade de generalização, aplicando padrões e tendências entre variáveis de uma aplicação. Tal aspecto é muito significativo quando se diz respeito à previsão de índices da bolsa de valores, principalmente pelo fato da não necessidade de compreensão das relações entre as variáveis impostas no problema. A partir disto, este trabalho se propôs a modelar RNAs para realizar a predição do valor de fechamento do índice S&P 500 da bolsa de valores de Nova Iorque com um dia de antecedência. Além disto, este trabalho compara o melhor modelo de RNA encontrado com um modelo de predição baseado na lógica paraconsistente.

Com intuito de atingir os objetivos do trabalho, primeiramente foi realizada uma pesquisa bibliográfica na literatura a fim de identificar nos trabalhos correlatos os modelos que mais têm sido usados para este tipo de predição, bem como os que obtiveram maior destaque. Após realizar tal coleta de dados, foram modelados 4 modelos de RNA, diferenciando cada um deles pelos conjuntos de dados de entradas, com o objetivo de identificar qual o melhor conjunto de entrada para a predição de índices da bolsa.

A topologia de Rede Neural utilizada neste trabalho foi a *Multilayer Perceptron*, onde foram criadas configurações variando o algoritmo de treinamento entre o TrainLM e TrainBR, a função de ativação dos neurônios da camada oculta entre LOGSIG e TANSIG e a quantidade de neurônios presente na camada oculta. Cada configuração de rede foi treinada 30 vezes para tentar diminuir os efeitos causados pela aleatoriedade dos valores iniciais. Além disto, cada modelo foi treinado com 4 conjuntos de dados diferente, o primeiro conjunto apresentando dados de 6 meses da bolsa de valores, o segundo apresentando dados de 1 ano, o terceiro com dados de 3 anos e por fim, o último contendo 5 anos de dados.

Para avaliar o desempenho das redes modeladas, foram utilizadas as métricas de erro RMSE e MAPE. Além disto, as RNAs também foram testadas na predição da direção do índice, em relação a subida ou queda do valor. Para isto utilizou-se duas estratégias, uma delas comparando o valor previsto com o último valor real conhecido, caso o previsto seja maior, então significa que índice irá subir, caso contrário, irá cair. A segunda estratégia foi comparar o valor previsto, com o último valor previsto no dia anterior, isto pelo fato de que os valores preditos pelas RNAs acompanham de forma bastante semelhante a variação do valor real do índice, assim como foi possível verificar de forma gráfica no decorrer do trabalho.

A melhor configuração de RNA foi modelada com 11 neurônios na camada oculta, função de ativação LOGSIG e algoritmo de treinamento TrainLM. O conjunto de dados de entrada que obteve este resultado foi composto pelas médias móveis de 50 dias do valor de abertura e fechamento do índice, além dos indicadores de análise técnica, Percentual R, ROC e RSI, ambos utilizando o cálculo com 50 dias de informações do índice.

Ao comparar os resultados obtidos pelas Redes Neurais Artificiais com os resultados da lógica paraconsistente, percebe-se certa vantagem em relação as RNAs, isso devido ao fato de que a lógica paraconsistente apresenta dias em que a predição é inconclusiva, ou seja, o modelo não afirma uma predição, enquanto nas Redes Neurais, há sempre uma predição, e como visto no capítulo de resultados, tal predição mantém um percentual de erro estável.

Apesar dos resultados obtidos pelas RNAs, destaca-se que a generalização dos resultados apresentados neste trabalho, realizando um estudo de caso específico para o índice S&P 500, não pode ser utilizada diretamente para a predição de outros índices, pois deve ser levado em consideração as características específicas de cada índice, sendo os modelos construídos neste trabalho, específicos para o índice S&P 500, onde para aplicação em outros índices da bolsa, deve-se realizar novos estudos em cima do índice a ser previsto, objetivando encontrar a melhor configuração para o novo caso.

Como este trabalho propôs-se a criar uma comparação entre os modelos de Redes Neurais com o modelo baseado na lógica paraconsistente, propõem-se como trabalhos futuros a realização de comparações entre outros modelos de predição de índices da bolsa de valores, como por exemplo, modelos estatísticos ou outros modelos que trabalhem com séries temporais.

Outra possibilidade na continuação das pesquisas circundando este tema, seria a mescla das duas técnicas apresentadas neste trabalho, ou seja, encontrar e utilizar os pontos fortes de cada uma das técnicas com intuito de formalizar um método de previsão que faça uso de características do modelos baseados em Redes Neurais Artificiais e dos baseados na lógica paraconsistente.

REFERÊNCIAS

BAGHIRLI, O. **Comparison of Lavenberg-Marquardt, Scaled Conjugate Gradient And Bayesian Regularization Backpropagation Algorithms for Multistep Ahead Wind Speed Forecasting Using Multilayer Perceptron Feedforward Neural Network.** Dissertação (Mestrado em Ciência). Department of Earth Sciences, Uppsala University, 2015.

BEALE, M. H.; HAGAN, M. T.; DEMUTH, H. B. **Neural Network Toolbox TM User's Guide.** 8.2 ed. Natick: Math Works, 2014.

BEALE, M. H.; HAGAN, M. T.; DEMUTH, H. B. **Neural Network Toolbox TM Reference.** 9 ed. Natick: Math Works, 2016.

BISPO, C. A. F.; CAZARINI, E. W. **Avaliação qualitativa paraconsistente do processo de implantação de um sistema de gestão ambiental.** Gestão e Produção. v. 13, n. 1, p. 117-127, 2006.

BRAGA, A. P; LUDERMI, T. B; LEON FERREIRA CARVALHO, A. C. P. **Redes Neurais Artificiais - Teoria e aplicações.** LTC – Livros Técnicos e Científicos Editora S.A., 2011.

BROOKSHEAR, J. G. **Ciência da Computação: Uma Visão Abrangente.** 11. ed. Porto Alegre: Bookmann, 2013.

BM&FBOVESPA. **Programa de Qualificação Operacional: Apresentação da Certificação Profissional do PQO**, 2016.

CAVALCANTE, F. **Mercado de Capitais.** Editora Campus Ltda, 2002.

DA SILVA FILHO, J. I.; **Métodos de aplicações da Lógica Paraconsistente anotada de anotação com dois valores LPA2v com construção de algoritmo e implementação de circuitos eletrônicos.** 1999. 226 f. Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais. 1999.

DE OLIVEIRA, F.A.; NOBRE, C.N.; ZÁRATE, L.E. **Applying Artificial Neural Networks to prediction of stock price and improvement of the directional prediction index – Case study of PETR4, Petrobras, Brazil.** Expert System with Applications, 40 (2013) 7596-7606

DEMUTH, H.; BEALE, M. **Neural Network Toolbox TM User's Guide**. 4. ed. Natick: Math Works, 2002.

GURESEN, E.; KAYAKUTLU, G.; DAIM, T.U. **Using artificial neural network models in stock market index prediction.** Expert System with Applications, 38 (2011) 10389-10397

KIRSTEN, H. A. **Comparação entre os modelos holt-winters e redes neurais para previsão de śeries temporais financeiras.** 2009. 87 f. Dissertação (Pós-Graduação em Engenharia de Produção e Sistemas) - Pontifícia Universidade Católica do Paraná, Curitiba. 2009.

HAYKIN, S. **Redes neurais: princípios e práticas.** 2. ed. Porto Alegre: Bookmann, 2001.

KRIEGER EDUARDO, Paulo. **Uso de Redes Neurais Artificiais para Predição da Bolsa de Valores.** 2012. 91 folhas. Trabalho Técnico-científico de Conclusão de Curso (Graduação em Ciência da Computação) – Centro de Ciências Tecnológicas da Terra e do Mar, Universidade do Vale do Itajaí, Itajaí, 2012.

LAGIOIA, U. C. T. **Fundamentos do Mercado de Capitais**. Editora Atlas, 2007.

MARTINS, G. B. **Aplicação da Lógica Paraconsistente anotada evidencial et em mercados financeiros.** 2012. 62 f. Dissertação (Mestrado em Economia) - Departamento de Economia, Universidade Federal de Santa Catarina, Florianópolis 2012.

RÊGO, R. H. T; MUSSA, A. **Anomalias do mercado acionário: A verificação do efeito feriado no IBOVESPA e IBX-100 no período de 2002 a 2007**. Congresso USP de Controladoria e Contabilidade, n. 8, 2008. São Paulo.

ROQUE, R. C. **Estudo sobre a empregabilidade da previsão do índice BOVESPA usando Redes Neurais Artificiais.** 2009. 101 f. Escola Politécnica, Departamento de Eletrônica e de Computação, Universidade do Rio de Janeiro, Rio de Janeiro. 2009.

SILVA, I. N. DA; SPATTI, D. H.; FLAUZINO, R. A. **Redes Neurais Artificiais para engenharia e ciências aplicadas.** São Paulo: Artliber, 2010.

SIEGEL, J. J. **Investindo em ações no longo Prazo: O guia indispensável do investidor no mercado financeiro.** 5. Ed. Editora Bookman, 2015.

TSAIH, R. HSU, Y. LAI, C.C. **Forecasting S&P 500 stock index futures with a hybrid AI system.** Decision Support Systems 23 1998 161–174 (1998) 163

VARELA, D. A. **Lógica paraconsistente: Lógicas da inconsistência formal e dialeteísmo.** Fundamento, v. 1, n. 1, p. 186-201, set-dez 2010.

WANG, J-Z, WANG, J-J, ZHANG, Z-G, GUO, S-P. **Forecasting stock indices with back propagation neural network.** Expert System with Applications, 38 (2011) 14346–14355

YUDONG, Z. LENAN, W. **Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network.** Expert Systems with Applications 36 (2009) 8849–8854

ZAFARI, A.; KIANMEHR, M. H.; ABDOLAHZADEH, R. **Modeling the effect of extrusion parameters on density of biomass pellet using artificial neural network**. International Journal Of Recycling of Organic Waste in Agriculture, v. 2, n. 1, p. 9, 2013.

ZHANG, T.; YOU, X. **Improvement of the Training and Normalization Method of Artificial Neural Network in the Prediction of Indoor Environment.** Procedia Engineering, v. 121, p. 1245–1251, 2015.

APÊNDICE A – ALGORITMO PARA MODELAGEM DAS REDES MLP

```
fileId = fopen('arquivo_log.txt','wt');
c = clock;
data = datestr(datenum(c(1),c(2),c(3),c(4),c(5),c(6)));
texto_log = strcat('[',data,'] Inicio do processamento.');
fprintf(fileId,'%s\n',texto_log);
fprintf('%s\n',texto_log);

caminhoPlanilha = 'C:\Users\Avell\Documents\Estudo\IFC
BCC\TCC\RNA\Planilhas\planilha_dados_rna_normalizados.xlsx';

%valores abertura
menorValorNorm = 1087.5;
maiorValorNorm = 2186;

num_inicial = '50';
num_final = '75';
algoritmos = {'trainbr'; 'trainlm'};
funcoes = {'tansig'; 'logsig'};
lista_modelos = {strcat('B',num_inicial,':F',num_final);
                 strcat('G',num_inicial,':K',num_final);
                 strcat('L',num_inicial,':P',num_final);
                 strcat('Q',num_inicial,':U',num_final);
                 strcat('V',num_inicial,':AA',num_final);
                 strcat('AB',num_inicial,':AG',num_final);
                 strcat('AH',num_inicial,':AM',num_final);
                 strcat('AN',num_inicial,':AR',num_final);
                 strcat('AS',num_inicial,':AW',num_final);
                 strcat('AX',num_inicial,':BB',num_final);
                 };
%mapeamento_targets = strcat('BC',num_inicial,':BC',num_final);%abertura
mapeamento_targets = strcat('BD',num_inicial,':BD',num_final);%fechamento
% 1 = img 1 / 2 = img 2 (5) / 3 = img 2 (20) / 4 = img 2 (50)
% 5 = img 3 (5) / 6 = img 3 (20) / 7 = img 3 (50)
% 8 = img 4 (5) / 9 = img 4 (20) / 10 = img 4 (50)

arquivo_menor_rmse_local = '';
menor_rmse_local = 0;
arquivo_menor_rmse_global = '';
menor_rmse_global = 999;
menor_mape = 9999;
arquivo_menor_mape = '';

vetorResultado = {'Modelo','Algoritmo','Funcao','Nr Neuronios',
'Treinamento NR','Nr. Epocas','RMSE Treinamento','RMSE Teste','MAPE
Treinamento','MAPE Teste','Arquivo';0,0,0,0,0,0,0,0,0,0,0};
resultadoLocal = {0,0,0,0,0,0,0,0,0,0,0};
cont_vet_rusult = 2;

for modelo = 1:10
   for algor = 1:2
        for funcao = 1:2
            c = clock;
            data = datestr(datenum(c(1),c(2),c(3),c(4),c(5),c(6)));
            texto_log = strcat('   [',data,'] Iniciando modelo neural:
',int2str(modelo),', algoritmo treinamento: ',algoritmos{algor},', funcao
ativacao: ',funcoes{funcao}, '.');
```

```
            fprintf(fileId,'%s\n',texto_log);
            fprintf('%s\n',texto_log);
            %pega dados
            if funcao == 1
                %tansig (-1 a 1)
                [~, ~, raw] = xlsread(caminhoPlanilha,
'dados_norm_menos_um_a_um', lista_modelos{modelo});
                [~, ~, rawTarget] = xlsread(caminhoPlanilha,
'dados_norm_menos_um_a_um', mapeamento_targets);
            else
                %logsig (0 a 1)
                [~, ~, raw] = xlsread(caminhoPlanilha,
'dados_norm_zero_a_um', lista_modelos{modelo});
                [~, ~, rawTarget] = xlsread(caminhoPlanilha,
'dados_norm_zero_a_um', mapeamento_targets);
            end

            entradas = reshape([raw{:}],size(raw));
            targets = reshape([rawTarget{:}],size(rawTarget));
            tamTargets = size(targets);
            clearvars raw;
            clearvars rawTargets;

            entradas = entradas';
            targets = targets';


            %loop neuronios
            for neuros = 3:15
                tlog = strcat('
[m_',int2str(modelo),'__at_',int2str(algor),'__fa_',int2str(funcao),']');
                c = clock;
                data = datestr(datenum(c(1),c(2),c(3),c(4),c(5),c(6)));
                texto_log = strcat('      [',data,']',tlog,' Executando
treinamentos para: ',int2str(neuros),' neuronios na camada oculta.');
                fprintf(fileId,'%s\n',texto_log);
                fprintf('%s\n',texto_log);

                for i = 1:30
                    c = clock;
                    data =
datestr(datenum(c(1),c(2),c(3),c(4),c(5),c(6)));
                    texto_log = strcat('         [',data,'] ',tlog,'
Executando treinamento numero: ',int2str(i));
                    fprintf(fileId,'%s\n',texto_log);
                    fprintf('%s\n',texto_log);

                    % executa o treinamento da rede neural

                    %modela a RNA
                    neuroniosCamadaOculta = neuros;
                    algoTreinamento = char(algoritmos(algor));
                    net =
feedforwardnet(neuroniosCamadaOculta,algoTreinamento);
                    net = configure(net,entradas,targets);

                    %selciona as funcoes de ativacao
                    net.layers{1}.transferFcn = char(funcoes(funcao));
                    net.layers{2}.transferFcn = 'purelin';
                    net.trainParam.epochs = 1000;
```

```
                    % seleciona os dados para treinamento, validacao e
testes
                    net.divideFcn = 'dividerand';
                    net.divideMode = 'sample';
                    net.divideParam.trainRatio = 70/100;
                    net.divideParam.valRatio = 15/100;
                    net.divideParam.testRatio = 15/100;

                    % treina a rna
                    [net,tr] = train(net,entradas,targets);

                    % faz os testes da rna
                    saidas = net(entradas);
                    erro = gsubtract(targets,saidas);
                    trainTargets = targets .* tr.trainMask{1};
                    valTargets = targets .* tr.valMask{1};
                    testTargets = targets .* tr.testMask{1};
                    trainPerformance = perform(net, trainTargets,
saidas);
                    valPerformance = perform(net, valTargets, saidas);
                    testPerformance = perform(net, testTargets, saidas);
                    epochs = length(tr.epoch)-1;

                    % calcula erro
                    RMSE_treinamento = sqrt(trainPerformance);
                    RMSE_teste = sqrt(testPerformance);

                    % calcula mape
                    %MAPE
                    qtdTestes = 0;
                    qtdTreinamento = 0;
                    epTestes = [0,0];
                    epTreinamento = [0,0];
                    for x = 1:tamTargets(1) % total de amostras
                        if (~isnan(testTargets(x)))
                            if(targets(x)~=0)
                                qtdTestes = qtdTestes+1;
                                epTestes(qtdTestes) =
abs((erro(x)/targets(x))*100); %erro percentual para ser usado no MAPE
                            end
                        end;
                        if (~isnan(trainTargets(x)))
                            if(targets(x)~=0)
                                qtdTreinamento = qtdTreinamento + 1;
                                epTreinamento(x) =
abs((erro(x)/targets(x))*100); %erro percentual para ser usado no MAPE
                            end
                        end;
                    end;

                    mapeTeste = sum(epTestes)/qtdTestes; % mape para
conjunte de testes
                    mapeTreinamento = sum(epTreinamento)/qtdTreinamento;
% mape para conjunte de testes

                    c = clock;
                    data =
datestr(datenum(c(1),c(2),c(3),c(4),c(5),c(6)));
```

```
                    texto_log = strcat('             [',data,'] Fim do
treinamento em questao, rmse_treinamento = ',
num2str(RMSE_treinamento),', rmse_teste = ',num2str(RMSE_teste),',
mape_teste = ',num2str(mapeTeste),', mape_treinamento =
',num2str(mapeTreinamento));
                    fprintf(fileId,'%s\n',texto_log);
                    fprintf('%s\n',texto_log);


                    %faz comparacoes
                    if (i == 1)
                        arquivo_menor_rmse_local =
strcat('modelo_',int2str(modelo),'__',algoritmos{algor},'__',funcoes{func
ao},'__neuronios_',int2str(neuros),'__treinamento_',int2str(i),'__rmse_',
num2str(RMSE_teste), '__nr_epocas_',num2str(epochs));
                        arquivo_menor_rmse_local =
strrep(arquivo_menor_rmse_local,'.','_');
                        arquivo_menor_rmse_local =
strcat(arquivo_menor_rmse_local,'.mat');

                        resultadoLocal{1} = modelo;
                        resultadoLocal{2} = algoritmos{algor};
                        resultadoLocal{3} = funcoes{funcao};
                        resultadoLocal{4} = neuros;
                        resultadoLocal{5} = i;
                        resultadoLocal{6} = epochs;
                        resultadoLocal{7} = RMSE_treinamento;
                        resultadoLocal{8} = RMSE_teste;
                        resultadoLocal{9} = mapeTreinamento;
                        resultadoLocal{10} = mapeTeste;
                        resultadoLocal{11} = arquivo_menor_rmse_local;

                        menor_rmse_local = RMSE_teste;
                        save(arquivo_menor_rmse_local);
                    end;
                    if (RMSE_teste < menor_rmse_local)
                        delete(arquivo_menor_rmse_local);
                        arquivo_menor_rmse_local =
strcat('modelo_',int2str(modelo),'__',algoritmos{algor},'__',funcoes{func
ao},'__neuronios_',int2str(neuros),'__treinamento_',int2str(i),'__rmse_',
num2str(RMSE_teste), '__nr_epocas_',num2str(epochs));
                        arquivo_menor_rmse_local =
strrep(arquivo_menor_rmse_local,'.','_');
                        arquivo_menor_rmse_local =
strcat(arquivo_menor_rmse_local,'.mat');


                        resultadoLocal{1} = modelo;
                        resultadoLocal{2} = algoritmos{algor};
                        resultadoLocal{3} = funcoes{funcao};
                        resultadoLocal{4} = neuros;
                        resultadoLocal{5} = i;
                        resultadoLocal{6} = epochs;
                        resultadoLocal{7} = RMSE_treinamento;
                        resultadoLocal{8} = RMSE_teste;
                        resultadoLocal{9} = mapeTreinamento;
                        resultadoLocal{10} = mapeTeste;
                        resultadoLocal{11} = arquivo_menor_rmse_local;

                        menor_rmse_local = RMSE_teste;
```

```
                        save(arquivo_menor_rmse_local);
                    end;

                    clear net; clear tr;
                end

                vetorResultado{cont_vet_rusult,1} = resultadoLocal{1};
                vetorResultado{cont_vet_rusult,2} = resultadoLocal{2};
                vetorResultado{cont_vet_rusult,3} = resultadoLocal{3};
                vetorResultado{cont_vet_rusult,4} = resultadoLocal{4};
                vetorResultado{cont_vet_rusult,5} = resultadoLocal{5};
                vetorResultado{cont_vet_rusult,6} = resultadoLocal{6};
                vetorResultado{cont_vet_rusult,7} = resultadoLocal{7};
                vetorResultado{cont_vet_rusult,8} = resultadoLocal{8};
                vetorResultado{cont_vet_rusult,9} = resultadoLocal{9};
                vetorResultado{cont_vet_rusult,10} = resultadoLocal{10};
                vetorResultado{cont_vet_rusult,11} = resultadoLocal{11};
                cont_vet_rusult = cont_vet_rusult + 1;

                % salva resultados
                if (menor_rmse_local < menor_rmse_global)
                    arquivo_menor_rmse_global = arquivo_menor_rmse_local;
                    menor_rmse_global = menor_rmse_local;
                end
                % salva resultados
                if (resultadoLocal{10} < menor_mape)
                    arquivo_menor_mape = arquivo_menor_rmse_local;
                    menor_mape = resultadoLocal{10};
                end


            end
        end

    end
end

xlswrite('ResultadoTreinamento.xlsx',vetorResultado,'resultados_treinamen
to');

c = clock;
data = datestr(datenum(c(1),c(2),c(3),c(4),c(5),c(6)));
texto_log = strcat('[',data,'] Processamento Terminado.');
fprintf(fileId,'%s\n',texto_log);
fprintf('%s\n',texto_log);

texto_log = strcat('Arquivo com menor RMSE: ',arquivo_menor_rmse_global);
fprintf(fileId,'%s\n',texto_log);
fprintf('%s\n',texto_log);

texto_log = strcat('Arquivo com menor MAPE: ',arquivo_menor_mape);
fprintf(fileId,'%s\n',texto_log);
fprintf('%s\n',texto_log);

fclose(fileId);
```

Fonte: Acervo do autor, 2017

APÊNDICE B – ALGORITMO PARA ESTRATÉGIA PARACONSISTENTE

```
%% obtem dados da planilha original e joga para matriz de dados
%caminhoXLS = input('Digite o caminho da planilha original: ');
caminhoXLS = 'C:\Users\Avell\Documents\Estudo\IFC BCC\TCC\Logica
paraconsistente\Planilhas\planilha_entrada.xlsx';
%inverteSequencia = input('Voce deseja inverter a sequecia da planilha?
(1 - Sim / 0 - Nao): ');
inverteSequencia = 1;
%buyNHold = input('Digite o valor inicial para o investimento: ');
investimento = 100000;

fprintf('Executando, por favor aguarde...\n');

[~,~,raw] = xlsread(caminhoXLS);
tamanho = size(raw);
nlinhas = tamanho(1);
ncolunas = tamanho(2);
matrizDadosRetorno =  {'Data', 'Ultimo Valor';0,0};
% configuracoes das colunas
matrizDadosRetorno{1,3} = 'Media 1';%matriz numerica coluna 2
matrizDadosRetorno{1,4} = 'Media 2';%matriz numerica coluna 3
matrizDadosRetorno{1,5} = 'Media 5';%matriz numerica coluna 4
matrizDadosRetorno{1,6} = 'Media 50';%matriz numerica coluna 5
matrizDadosRetorno{1,7} = 'Media 150';%matriz numerica coluna 6
matrizDadosRetorno{1,8} = 'Media 200';%matriz numerica coluna 7
matrizDadosRetorno{1,9} = 'Up';%matriz numerica coluna 8
matrizDadosRetorno{1,10} = 'Sinal 1-50';%matriz numerica coluna 9
matrizDadosRetorno{1,11} = 'Sinal 1-150';%matriz numerica coluna 10
matrizDadosRetorno{1,12} = 'Sinal 1-200';%matriz numerica coluna 11
matrizDadosRetorno{1,13} = 'Sinal 5-150';%matriz numerica coluna 12
matrizDadosRetorno{1,14} = 'Sinal 2-200';%matriz numerica coluna 13
matrizDadosRetorno{1,15} = 'Acerto Sinal 1-50';%matriz numerica coluna 14
matrizDadosRetorno{1,16} = 'Acerto Sinal 1-150';%matriz numerica coluna
15
matrizDadosRetorno{1,17} = 'Acerto Sinal 1-200';%matriz numerica coluna
16
matrizDadosRetorno{1,18} = 'Acerto Sinal 5-150';%matriz numerica coluna
17
matrizDadosRetorno{1,19} = 'Acerto Sinal 2-200';%matriz numerica coluna
18
matrizDadosRetorno{1,20} = 'C_n 1.-50';%matriz numerica coluna 19
matrizDadosRetorno{1,21} = 'C_n 1.-150';%matriz numerica coluna 20
matrizDadosRetorno{1,22} = 'C_n 1.-200';%matriz numerica coluna 21
matrizDadosRetorno{1,23} = 'C_n 5.-150';%matriz numerica coluna 22
matrizDadosRetorno{1,24} = 'C_n 2.-200';%matriz numerica coluna 23
matrizDadosRetorno{1,25} = 'Buy and Hold';%matriz numerica coluna 24
matrizDadosRetorno{1,26} = '1.-50 mu_1';%matriz numerica coluna 25
matrizDadosRetorno{1,27} = '1.-50 mu_2';%matriz numerica coluna 26
matrizDadosRetorno{1,28} = '1.-150 mu_1';%matriz numerica coluna 27
matrizDadosRetorno{1,29} = '1.-150 mu_2';%matriz numerica coluna 28
matrizDadosRetorno{1,30} = '1.-200 mu_1';%matriz numerica coluna 29
matrizDadosRetorno{1,31} = '1.-200 mu_2';%matriz numerica coluna 30
matrizDadosRetorno{1,32} = '5.-150 mu_1';%matriz numerica coluna 31
matrizDadosRetorno{1,33} = '5.-150 mu_2';%matriz numerica coluna 32
matrizDadosRetorno{1,34} = '2.-200 mu_1';%matriz numerica coluna 33
matrizDadosRetorno{1,35} = '2.-200 mu_2';%matriz numerica coluna 34
matrizDadosRetorno{1,36} = 'Grupo A mu_1';%matriz numerica coluna 35
matrizDadosRetorno{1,37} = 'Grupo A mu_2';%matriz numerica coluna 36
```

```
matrizDadosRetorno{1,38} = 'Grupo B mu_1';%matriz numerica coluna 37
matrizDadosRetorno{1,39} = 'Grupo B mu_2';%matriz numerica coluna 38
matrizDadosRetorno{1,40} = 'Grupo C mu_1';%matriz numerica coluna 39
matrizDadosRetorno{1,41} = 'Grupo C mu_2';%matriz numerica coluna 40
matrizDadosRetorno{1,42} = 'Grupo D mu_1';%matriz numerica coluna 41
matrizDadosRetorno{1,43} = 'Grupo D mu_2';%matriz numerica coluna 42
matrizDadosRetorno{1,44} = 'Resultante mu_1';%matriz numerica coluna 43
matrizDadosRetorno{1,45} = 'Resultante mu_2';%matriz numerica coluna 44
matrizDadosRetorno{1,46} = 'Lambda';%matriz numerica coluna 45
matrizDadosRetorno{1,47} = 'percentual indice';%matriz numerica coluna 46
matrizDadosRetorno{1,48} = 'Indica Compra (Sinais)';% 1-compra;0-venda
%matriz numerica coluna 47
matrizDadosRetorno{1,49} = 'Viavel a 60';%matriz numerica coluna 48
matrizDadosRetorno{1,50} = 'Inviavel a 60';%matriz numerica coluna 49
matrizDadosRetorno{1,51} = 'Inconclusivo 60';%matriz numerica coluna 50
matrizDadosRetorno{1,52} = 'Compra a 60';%matriz numerica coluna 51
matrizDadosRetorno{1,53} = 'Vende a 60';%matriz numerica coluna 52
matrizDadosRetorno{1,54} = 'Acerto a 60';%matriz numerica coluna 53
matrizDadosRetorno{1,55} = 'Investimento a 60';%matriz numerica coluna 54

matrizNumerica = [0,0];
if(inverteSequencia == 1)
    cont = 2;
    for i = nlinhas:-1:2
        if ~isnan(raw{i,1}) & ~isnan(raw{i,1})
            matrizDadosRetorno{i,1} = raw{cont,1};
            matrizDadosRetorno{i,2} = raw{cont,2};
            matrizNumerica(i-1,1) = raw{cont,2};
            cont = cont + 1;
        end;
    end
else
    for i = 2:nlinhas
        if ~isnan(raw{i,1}) & ~isnan(raw{i,1})
            matrizDadosRetorno{i,1} = raw{i,1};
            matrizDadosRetorno{i,2} = raw{i,2};
            matrizNumerica(i-1,1) = raw{i,2};
        end;
    end
end
clearvars raw;
clearvars nlinhas;
clearvars ncolunas;
clearvars tamanho;

nlinhas = size(matrizNumerica, 1);
nAmTrein = 4767;%nlinhas;
while(nAmTrein <= 250 | nAmTrein > nlinhas)
    fprintf('Foram encontradas %d amostras, digite a quantidade desejada
para treinamento (deve ser maior que 250): ',nlinhas - 1);
nAmTrein = input('');
end
fprintf('Continuando execucao...\n');
%% inicia manipulacao colunas da matriz de dados
for i = 1 : nlinhas-1
    if i > 1
       %valor para 1 anterior
       matrizNumerica(i,2) = matrizNumerica(i-1,1);
       % up dia
       matrizNumerica(i,8) = matrizNumerica(i,1)>matrizNumerica(i-1,1);
       %matrizNumerica(i,8) = matrizNumerica(i,1)<matrizNumerica(i+1,1);
```

```
        %converte para matriz resultado
        matrizDadosRetorno{i+1,3} = matrizNumerica(i,2);
        matrizDadosRetorno{i+1,9} = matrizNumerica(i,8);
    end
    if i > 2
        %valor para 2 anterior
        matrizNumerica(i,3) = mean(matrizNumerica(i-2:i-1,1));

        %converte para matriz resultado
        matrizDadosRetorno{i+1,4} = matrizNumerica(i,3);
    end
    if i > 5
        %valor para 5 anterior
        matrizNumerica(i,4) = mean(matrizNumerica(i-5:i-1,1));

        %converte para matriz resultado
        matrizDadosRetorno{i+1,5} = matrizNumerica(i,4);
    end
    if i > 50
        %valor para 50 anterior
        matrizNumerica(i,5) = mean(matrizNumerica(i-50:i-1,1));

        %sinal 1-50
        matrizNumerica(i,9) = matrizNumerica(i,2)>matrizNumerica(i,5);
        %acerto sinal 1-50
        matrizNumerica(i,14) = matrizNumerica(i,8) == matrizNumerica(i,9);
        %c_n 1.-50
        matrizNumerica(i,19) = mean(matrizNumerica(51:i,14));

        %converte para matriz resultado
        matrizDadosRetorno{i+1,6} = matrizNumerica(i,5);
        matrizDadosRetorno{i+1,10} = matrizNumerica(i,9);
        matrizDadosRetorno{i+1,15} = matrizNumerica(i,14);
        matrizDadosRetorno{i+1,20} = matrizNumerica(i,19);
    end
    if i > 150
        %valor para 150 anterior
        matrizNumerica(i,6) = mean(matrizNumerica(i-150:i-1,1));

        %sinal 1-150
        matrizNumerica(i,10) = matrizNumerica(i,2)>matrizNumerica(i,6);
        %acerto sinal 1-150
        matrizNumerica(i,15) = matrizNumerica(i,8) ==
matrizNumerica(i,10);

        %sinal 5-150
        matrizNumerica(i,12) = matrizNumerica(i,4)>matrizNumerica(i,6);
        %acerto sinal 5-150
        matrizNumerica(i,17) = matrizNumerica(i,8) ==
matrizNumerica(i,12);

        %c_n 1.-150
        matrizNumerica(i,20) = mean(matrizNumerica(151:i,15));
        %c_n 5.-150
        matrizNumerica(i,22) = mean(matrizNumerica(151:i,17));

        %converte para matriz resultado
        matrizDadosRetorno{i+1,7} = matrizNumerica(i,6);
```

```
        matrizDadosRetorno{i+1,11} = matrizNumerica(i,10);
        matrizDadosRetorno{i+1,16} = matrizNumerica(i,15);
        matrizDadosRetorno{i+1,13} = matrizNumerica(i,12);
        matrizDadosRetorno{i+1,18} = matrizNumerica(i,17);
        matrizDadosRetorno{i+1,21} = matrizNumerica(i,20);
        matrizDadosRetorno{i+1,23} = matrizNumerica(i,22);
    end
    if i > 200
        %valor para 200 anterior
        matrizNumerica(i,7) = mean(matrizNumerica(i-200:i-1,1));

        %sinal 1-200
        matrizNumerica(i,11) = matrizNumerica(i,2)>matrizNumerica(i,7);
        %acerto sinal 1-200
        matrizNumerica(i,16) = matrizNumerica(i,8) ==
matrizNumerica(i,11);

        %sinal 2-200
        matrizNumerica(i,13) = matrizNumerica(i,3)>matrizNumerica(i,7);
        %acerto sinal 2-200
        matrizNumerica(i,18) = matrizNumerica(i,8) ==
matrizNumerica(i,13);

        %c_n 1.-200
        matrizNumerica(i,21) = mean(matrizNumerica(201:i,16));
        %c_n 2.-200
        matrizNumerica(i,23) = mean(matrizNumerica(201:i,18));

        %converte para matriz resultado
        matrizDadosRetorno{i+1,8} = matrizNumerica(i,7);
        matrizDadosRetorno{i+1,12} = matrizNumerica(i,11);
        matrizDadosRetorno{i+1,17} = matrizNumerica(i,16);
        matrizDadosRetorno{i+1,14} = matrizNumerica(i,13);
        matrizDadosRetorno{i+1,19} = matrizNumerica(i,18);
        matrizDadosRetorno{i+1,22} = matrizNumerica(i,21);
        matrizDadosRetorno{i+1,24} = matrizNumerica(i,23);
    end
    %valores apos o treinamento
    if i == nAmTrein
         %inicia valores buy and hold
         matrizNumerica(i,24) = investimento;
         %inicia valor ivest a 60
         matrizNumerica(i,54) = investimento;

         matrizDadosRetorno{i+1,25} = matrizNumerica(i,24);
         matrizDadosRetorno{i+1,55} = matrizNumerica(i,54);
    elseif i > nAmTrein
         %atualiza buy n hold
         matrizNumerica(i,24) = matrizNumerica(i-1,24) *
(matrizNumerica(i,1)/matrizNumerica(i-1,1));
         %1-50 mu_1 e mu_2
         matrizNumerica(i,25) = (matrizNumerica(i-
1,14)+mean(matrizNumerica(i-5:i-1,14)))/2;
         matrizNumerica(i,26) = 1 - matrizNumerica(i,25);
         %1-150 mu_1 e mu_2
         matrizNumerica(i,27) = (matrizNumerica(i-
1,15)+mean(matrizNumerica(i-5:i-1,15)))/2;
         matrizNumerica(i,28) = 1 - matrizNumerica(i,27);
         %1-200 mu_1 e mu_2
```

```
        matrizNumerica(i,29) = (matrizNumerica(i-
1,16)+mean(matrizNumerica(i-5:i-1,16)))/2;
        matrizNumerica(i,30) = 1 - matrizNumerica(i,29);
        %5-150 mu_1 e mu_2
        matrizNumerica(i,31) = (matrizNumerica(i-
1,17)+mean(matrizNumerica(i-5:i-1,17)))/2;
        matrizNumerica(i,32) = 1 - matrizNumerica(i,31);
        %2-200 mu_1 e mu_2
        matrizNumerica(i,33) = (matrizNumerica(i-
1,18)+mean(matrizNumerica(i-5:i-1,18)))/2;
        matrizNumerica(i,34) = 1 - matrizNumerica(i,33);

        %Grupo A
        matrizNumerica(i,35) = matrizNumerica(i,29);
        matrizNumerica(i,36) = 1 - matrizNumerica(i,35);
        %Grupo B
        matrizNumerica(i,37) = matrizNumerica(i,31);
        matrizNumerica(i,38) = 1 - matrizNumerica(i,37);
        %Grupo C
        matrizNumerica(i,39) = matrizNumerica(i,33);
        matrizNumerica(i,40) = 1 - matrizNumerica(i,39);
        %Grupo D
        matrizNumerica(i,41) =
max([matrizNumerica(i,25),matrizNumerica(i,27)]);
        matrizNumerica(i,42) =
max([matrizNumerica(i,26),matrizNumerica(i,28)]);
        %Resultante
        matrizNumerica(i,43) =
min([matrizNumerica(i,35),matrizNumerica(i,37),matrizNumerica(i,39),matri
zNumerica(i,41)]);
        matrizNumerica(i,44) =
min([matrizNumerica(i,36),matrizNumerica(i,38),matrizNumerica(i,40),matri
zNumerica(i,42)]);
        %lambda
        matrizNumerica(i,45) = matrizNumerica(i,43)-matrizNumerica(i,44);
        %percentual do indice
        matrizNumerica(i,46) = matrizNumerica(i,2)/matrizNumerica(i-1,2)-
1;
        %compra?
        matrizNumerica(i,47) =
mean([matrizNumerica(i,14),matrizNumerica(i,15),matrizNumerica(i,16),matr
izNumerica(i,17),matrizNumerica(i,18)])>=0.5;
        viabilidade = 1;
        %viavel a 60
        matrizNumerica(i,48) = matrizNumerica(i,45) >= viabilidade;
        %inviavel a 60
        matrizNumerica(i,49) = matrizNumerica(i,45) <= (viabilidade*-1);
        %inconclusivo a 60
        matrizNumerica(i,50) = matrizNumerica(i,45) > (viabilidade*-1) &
matrizNumerica(i,45) < viabilidade;
        %a 60% deve comprar: sinais indicam compra e viavel ou sinais
indica venda e inviavel
        matrizNumerica(i,51) = (matrizNumerica(i,47) &
matrizNumerica(i,48)) | (~matrizNumerica(i,47) & matrizNumerica(i,49));
        %a 60% deve vender: sinais indicam venda e viavel ou sinais
indicam compra e inviavel
        matrizNumerica(i,52) = (~matrizNumerica(i,47) &
matrizNumerica(i,48)) | (matrizNumerica(i,47) & matrizNumerica(i,49));
        %acerto a 60 subiu e mandou comprar ou desceu e vender
        auxSubiu = matrizNumerica(i,1)<matrizNumerica(i+1,1);%pega se o
valor referente a amanha vai aumentar em relacao a hoje
```

```
        %a variavel auxSubiu eh diferente da coluna Up pq a coluna up diz
        %se hoje subiu em relacao a ontem ja esta variavel diz se amanha
        %vai subir em relacao a hoje (usado pra comparacao de acerto)
        %matrizNumerica(i,53) = (matrizNumerica(i,8) &
matrizNumerica(i,51))|(~matrizNumerica(i,8) & matrizNumerica(i,52));
        matrizNumerica(i,53) = (auxSubiu &
matrizNumerica(i,51))|(~auxSubiu & matrizNumerica(i,52));
        %investimento a 60
        if(matrizNumerica(i,52))%se mandou vender
            matrizNumerica(i,54) = matrizNumerica(i-1,54)*(1-
matrizNumerica(i,46));
        else%se mandou comprar ou nao faz nada
            matrizNumerica(i,54) = matrizNumerica(i-
1,54)*(1+matrizNumerica(i,46));
        end

        %transcreve para matriz retorno
        matrizDadosRetorno{i+1,25} = matrizNumerica(i,24);
        matrizDadosRetorno{i+1,26} = matrizNumerica(i,25);
        matrizDadosRetorno{i+1,27} = matrizNumerica(i,26);
        matrizDadosRetorno{i+1,28} = matrizNumerica(i,27);
        matrizDadosRetorno{i+1,29} = matrizNumerica(i,28);
        matrizDadosRetorno{i+1,30} = matrizNumerica(i,29);
        matrizDadosRetorno{i+1,31} = matrizNumerica(i,30);
        matrizDadosRetorno{i+1,32} = matrizNumerica(i,31);
        matrizDadosRetorno{i+1,33} = matrizNumerica(i,32);
        matrizDadosRetorno{i+1,34} = matrizNumerica(i,33);
        matrizDadosRetorno{i+1,35} = matrizNumerica(i,34);
        matrizDadosRetorno{i+1,36} = matrizNumerica(i,35);
        matrizDadosRetorno{i+1,37} = matrizNumerica(i,36);
        matrizDadosRetorno{i+1,38} = matrizNumerica(i,37);
        matrizDadosRetorno{i+1,39} = matrizNumerica(i,38);
        matrizDadosRetorno{i+1,40} = matrizNumerica(i,39);
        matrizDadosRetorno{i+1,41} = matrizNumerica(i,40);
        matrizDadosRetorno{i+1,42} = matrizNumerica(i,41);
        matrizDadosRetorno{i+1,43} = matrizNumerica(i,42);
        matrizDadosRetorno{i+1,44} = matrizNumerica(i,43);
        matrizDadosRetorno{i+1,45} = matrizNumerica(i,44);
        matrizDadosRetorno{i+1,46} = matrizNumerica(i,45);
        matrizDadosRetorno{i+1,47} = matrizNumerica(i,46);
        matrizDadosRetorno{i+1,48} = matrizNumerica(i,47);
        matrizDadosRetorno{i+1,49} = matrizNumerica(i,48);
        matrizDadosRetorno{i+1,50} = matrizNumerica(i,49);
        matrizDadosRetorno{i+1,51} = matrizNumerica(i,50);
        matrizDadosRetorno{i+1,52} = matrizNumerica(i,51);
        matrizDadosRetorno{i+1,53} = matrizNumerica(i,52);
        matrizDadosRetorno{i+1,54} = matrizNumerica(i,53);
        matrizDadosRetorno{i+1,55} = matrizNumerica(i,54);

    end
end


%% armazena resultados na planilha de resultado
xlswrite(caminhoXLS,matrizDadosRetorno,'Resultados a 1');
fprintf('Resultados obtidos com sucesso! Dados salvos na aba Resultados
da planilha de entrada\n');
```

Fonte: Acervo do autor, 2017