



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Aplicação de Redes Neurais visando suporte à tomada de decisão no Mercado Financeiro

Jônatas Rocha de Paiva

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Orientador
Prof. Dr. Marcus Vinicius Lamar

Brasília
2022

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Curso de Engenharia da Computação

Coordenador: Prof. Dr. João José Costa Gondim

Banca examinadora composta por:

Prof. Dr. Marcus Vinicius Lamar (Orientador) — CIC/UnB

Prof. Dr. Marcos Fagundes Caetano — CIC/UnB

Prof. Dr. Luís Paulo Faina Garcia — CIC/UnB

CIP — Catalogação Internacional na Publicação

de Paiva, Jônatas Rocha.

Aplicação de Redes Neurais visando suporte à tomada de decisão no Mercado Financeiro / Jônatas Rocha de Paiva. Brasília : UnB, 2022.
88 p. : il. ; 29,5 cm.

Monografia (Graduação) — Universidade de Brasília, Brasília, 2022.

1. redes neurais, 2. mercado financeiro, 3. adaptação

CDU 004

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília-DF — Brasil

Dedicatória

Dedico o fruto dos meus esforços aos meus familiares e amigos que sempre me apoiaram, e também, aos professores e pessoas que passaram em minha vida, mesmo que brevemente, e me forçaram a crescer.

Agradecimentos

Agradeço às pessoas que mais convivo, meus pais e irmão por me apoiarem e me aturarem, agradeço aos meus amigos pelos momentos de lazer, agradeço aos meus professores que tiveram e ainda têm muito a me ensinar, inclusive, agradeço ao meu orientador pelo apoio e extrema paciência em me ouvir durante este trajeto.

Resumo

Nos últimos anos o interesse por Aprendizado de Máquina (do Inglês, *Machine Learning*) (ML) tem crescido muito na área do mercado financeiro, surgindo trabalhos usando Redes Neurais Artificiais (RNA), principalmente, para realizar regressão nos preços ou classificação do sinal de negociação de ativos financeiros. Então, seguindo a mesma linha, este trabalho propõe a aplicação de redes neurais no contexto do mercado financeiro, para realizar a previsão de séries temporais usando metodologias de adaptação, para que a rede se mantenha atualizada e consiga lidar com *concept drifts* reduzindo o erro de previsão. Esta aplicação visa dar suporte à tomada de decisão na negociação de ativos financeiros, e assim, facilitar o processo envolvido na negociação, eximindo o investidor do esforço mental e emocional da análise e da negociação de ativos financeiros, já que o maior interesse dos investidores é conseguir uma forma de se analisar os dados baseando-se em critérios objetivos para se atingir um maior retorno dos seus investimentos. A fim de validar o sistema, primeiro, é realizada uma avaliação estatística utilizando o Raiz-Quadrada do Erro Médio (do Inglês, *Root-Mean-Square Error*) (RMSE) e Erro Médio Absoluto Percentual (do Inglês, *Mean Absolute Percentage Error*) (MAPE) para mensurar o erro de previsão e encontrar parâmetros ótimos, em seguida o sistema é simulado para avaliar o retorno financeiro que o sistema conseguiria atingir. Os resultados demonstram que o sistema consegue, no geral, superar a estratégia *Buy and Hold*, entretanto não consegue superar outras abordagens como o sistema *Convolutional Neural Network - Trading Classifier* (CNN-TC). Ainda há muito espaço para aprimorar o sistema, com a implementação de algoritmos tomadores de decisão mais robustos, a previsão de outros dados que compõem um *candlestick*, o uso de séries temporais com períodos menores e a implementação de adaptação ativa são os principais focos para trabalhos futuros.

Palavras-chave: redes neurais, mercado financeiro, adaptação

Abstract

In the lastest years the interest for Machine Learning have exponentially risen, and so have the interest in the application of this technique in the financial market field, following a lot of reaserch being made with neural networks, those mainly comprise of regression of the price and the classification of trading signals of financial assets. This project goes along this line, proposing an application of neural network in the context of the financial market, to forecast time series using adaptative methods for maintaning the model updated and being able to handle concept drifts, reducing the forecasting error. This application aims at giving support for decision making in financial asset trading, therefore, acting as a facilitator of the process of financial asset trading, freeing the investors of the mental and emotional effort of trading, since the interest of investors mainly lies in achieving a way of analysing data based on objective criteria for optimizing profit. In order to validate the system, first, a statistical evaluation is executed utilizing the Root-Mean-Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) to measure the forecasting error and finding optimal parameters, secondly, a simulation is executed to evaluate the financial results, as to verify the system performance. The results demonstrates that the proposed system can mostly achieve higher profit than the Buy and Hold strategy, however it couldn't beat other approaches like the Convolutional Neural Network - Trading Classifier (CNN-TC). There still a lot of room for improvemments, a better performance could be achieved implementing a more robust decision making algorithm, forecasting time series with lower time frames, forecasting all the data that forms a candlestick and implementing of active adaptation.

Keywords: neural network, financial market, adaptation

Sumário

1	Introdução	1
1.1	Problema	4
1.2	Objetivos	4
1.3	Organização	5
2	Fundamentação Teórica	6
2.1	Mercado Financeiro	6
2.2	Teoria do Mercado Eficiente	8
2.3	Teoria do Passeio Aleatório	8
2.4	Métodos de Análise Financeira	9
2.4.1	Análise Fundamental	9
2.4.2	Análise Sentimental	10
2.4.3	Análise Técnica	11
2.5	Séries Temporais	14
2.6	<i>Concept Drift</i>	16
2.6.1	Adaptação Passiva	17
2.6.2	Adaptação Ativa	18
2.7	Redes Neurais Artificiais	18
2.7.1	Neurônio Artificial	19
2.7.2	Funções de ativação	21
2.7.3	Problema de Otimização	26
2.7.4	<i>Multilayer Perceptron</i>	28
2.7.5	<i>Backpropagation</i>	28
2.7.6	<i>Long Short-Term Memory</i>	30
2.8	<i>K-Fold</i>	32
3	Sistema Proposto	34
3.1	Visão Geral	34
3.1.1	Arquitetura do Sistema	35

3.2	Séries Temporais	37
3.2.1	Coleta dos Dados	37
3.2.2	Janela Deslizante	37
3.2.3	Processamento dos Dados	39
3.3	Treinamento	39
3.4	Avaliação Estatística	39
3.5	Avaliação Financeira	40
4	Resultados Obtidos	45
4.1	Seleção de modelos de RNA	45
4.1.1	Experimentos	45
4.1.2	Experimentos Adaptativos	46
4.2	Avaliação do Modelo	61
4.3	Simulação do Sistema	63
5	Conclusão	69
5.1	Limitações do Sistema Proposto	70
5.2	Trabalhos Futuros	70
	Referências	72

Lista de Figuras

1.1	Taxa de interesse pelo aprendizado de máquina por ano no Google Trends (Fonte: [1]).	2
1.2	Crescimento do índice Bovespa (Fonte: [2]).	2
2.1	Metodologia <i>Box-Jenkins</i> da construção de um modelo (Fonte: [3]).	16
2.2	Modelo de um neurônio artificial (Fonte: [4]).	20
2.3	Função de ativação linear (Fonte: Elaborado pelo autor).	22
2.4	Função de ativação de degrau (Fonte: Elaborado pelo autor).	23
2.5	Função de ativação sigmoide (Fonte: Elaborado pelo autor).	24
2.6	Função de ativação Tangente Hiperbólica (Tanh) (Fonte: Elaborado pelo autor).	25
2.7	Função de ativação ReLU (Fonte: Elaborado pelo autor).	26
2.8	RNN e redes <i>feedward</i> (Fonte: [5]).	31
2.9	Célula da rede LSTM (Fonte: [6]).	31
2.10	<i>K-Fold</i> tradicional (Fonte: Elaborado pelo autor).	33
3.1	Diagrama do sistema proposto (Fonte: Elaborado pelo autor).	35
3.2	Fluxograma do módulo <i>Finance Processor</i> (Fonte: Elaborado pelo autor).	36
3.3	<i>Sliding window</i> implementado no trabalho (Fonte: Elaborado pelo autor).	38
3.4	<i>K-Fold</i> implementado no trabalho (Fonte: Elaborado pelo autor).	40
4.1	Gráficos do RMSE \times tamanho da janela do BTC (MLP).	48
4.2	Gráficos do RMSE \times tamanho da janela da TAEE (MLP).	49
4.3	Gráficos do RMSE \times tamanho da janela da VALE (MLP).	50
4.4	Gráficos do RMSE \times tamanho do janela do ITUB (MLP).	51
4.5	Gráficos do RMSE \times tamanho da janela da BOVA (MLP).	52
4.6	Gráficos do RMSE \times tamanho da janela do PETR (MLP).	53
4.7	Gráficos do RMSE \times tamanho da janela do BTC (LSTM).	55
4.8	Gráficos do RMSE \times tamanho da janela da TAEE (LSTM).	56
4.9	Gráficos do RMSE \times tamanho da janela da VALE (LSTM).	57

4.10	Gráficos do RMSE \times tamanho da janela do ITUB (LSTM).	58
4.11	Gráficos do RMSE \times tamanho da janela da BOVA (LSTM).	59
4.12	Gráficos do RMSE \times tamanho da janela do PETR (LSTM).	60
4.13	Resultados financeiros da simulação da BTC.	64
4.14	Resultados financeiros da simulação da TAEE.	64
4.15	Resultados financeiros da simulação da VALE.	65
4.16	Resultados financeiros da simulação da ITUB.	66
4.17	Resultados financeiros da simulação da BOVA.	66
4.18	Resultados financeiros da simulação da PETR.	67

Lista de Tabelas

3.1	Ativos financeiros usados no trabalho.	37
4.1	RMSE para janela de tamanho 1.	46
4.2	Tamanho ótimo de janela do Bitcoin (BTC).	49
4.3	Tamanho ótimo de janela da Transmissora Aliança de Energia Elétrica S.A. (TAEE).	50
4.4	Tamanho ótimo de janela da Vale S.A. (VALE).	51
4.5	Tamanho ótimo de janela do Itaú Unibanco (ITUB).	52
4.6	Tamanho ótimo de janela da Bovespa (BOVA).	53
4.7	Tamanho ótimo de janela da Petrobrás (PETR).	54
4.8	Tamanho ótimo de janela do Bitcoin (BTC).	55
4.9	Tamanho ótimo de janela da Transmissora Aliança de Energia Elétrica S.A. (TAEE).	56
4.10	Tamanho ótimo de janela da Vale S.A. (VALE).	57
4.11	Tamanho ótimo de janela do Itaú Unibanco (ITUB).	58
4.12	Tamanho ótimo de janela da Bovespa (BOVA).	59
4.13	Tamanho ótimo de janela da Petrobrás (PETR).	60
4.14	Tabela resumo para os melhores resultados.	61
4.15	Resultados k-fold com treinamento não-adaptativo.	62
4.16	Resultados k-fold com treinamento adaptativo.	62
4.17	Resultados da Simulação.	68

Lista de Abreviaturas e Siglas

AdaGrad Algoritmo do Gradiente Adaptativo (do Inglês, *Adaptive Gradient Algorithm*).

Adam Estimativa Adaptativa do Momentum (do Inglês, *Adaptive Momentum estimation*).

ALVINN Veículo Terrestre Autônomo em uma Rede Neural (do Inglês, *Autonomous Land Vehicle In a Neural Network*).

API Interface de Programação de Aplicação (do Inglês, *Application Programming Interface*).

BOVA Bovespa.

BTC Bitcoin.

CNN Redes Neurais Convolucionais (do Inglês, *Convolutional Neural Network*).

CNN-TA *Convolutional Neural Network - Technical Analysis*.

CNN-TC *Convolutional Neural Network - Trading Classifier*.

ES Parada Antecipada (do Inglês, *Early Stopping*).

ITUB Itaú Unibanco.

LSTM Memória de Longo Curto Prazo (do Inglês, *Long Short-Term Memory*).

MAPE Erro Médio Absoluto Percentual (do Inglês, *Mean Absolute Percentage Error*).

ML Aprendizado de Máquina (do Inglês, *Machine Learning*).

MLP Perceptron Multicamadas (do Inglês, *Multilayer Perceptron*).

MSE Erro Médio Quadrático (do Inglês, *Mean Squared Error*).

PETR Petrobrás.

ReLU Unidade Linear Retificada (do Inglês, *Rectified Linear Unit*).

RMSE Raiz-Quadrada do Erro Médio (do Inglês, *Root-Mean-Square Error*).

RMSProp Propagação da Média Quadrática (do Inglês, *Root Mean Squared Propagation*).

RNA Redes Neurais Artificiais.

RNN Rede Neural Recorrente (do Inglês, *Recurrent Neural Network*).

RSI Indicador de Força Relativa (do Inglês, *Relative Strength Index*).

SMA Média Móvel Simples (do Inglês, *Simple Moving Average*).

SRV Regressão de Vetores de Suporte (do Inglês, *Support Regression Vector*).

TAE Transmissora Aliança de Energia Elétrica S.A..

Tanh Tangente Hiperbólica.

VALE Vale S.A..

Capítulo 1

Introdução

Uma operação no mercado financeiro é o ato de comprar e vender ativos visando o lucro, este tipo de ato é comumente chamado de investimento de renda variável. O interesse dos investidores ao expor seu capital ao risco de uma variação negativa é o potencial ainda maior de uma variação positiva. Com o avanço da tecnologia, operar no mercado financeiro se tornou cada vez mais acessível para pequenos investidores, e consequentemente o interesse em investir se tornou cada vez mais popular. Não só o acesso ao ato de investir se tornou mais fácil, mas também, o acesso aos dados. O que torna possível a implementação de sistemas capazes de tirar vantagem desta acessibilidade.

Nos últimos 20 anos o mercado financeiro tem tido crescimento exponencial, e que somado à popularização da inteligência artificial, tem tido como resultado, um gradual aumento de interesse em todo o mercado financeiro global. As pessoas estão cada vez mais investindo tempo e recursos no desenvolvimento de sistemas de decisão inteligentes. O interesse em Aprendizado de Máquina (do Inglês, *Machine Learning*) (ML) é evidenciado pelo aumento na procura na ferramenta de pesquisa do Google, conforme ilustrado na Figura 1.1. Assim como no mercado financeiro global, o Brasil, teve um crescimento significativo no mercado de ações que pode ser observado com base na variação do índice Bovespa. Este índice assimila todas as ações com maior capital investidos do Brasil em um só índice, conforme ilustrado na Figura 1.2.

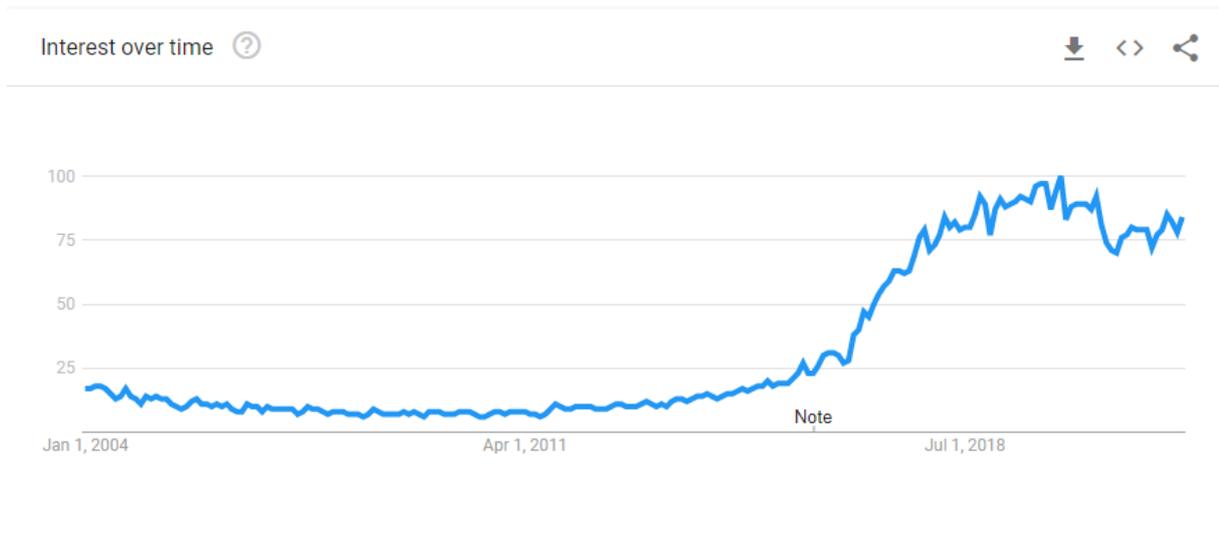


Figura 1.1: Taxa de interesse pelo aprendizado de máquina por ano no Google Trends (Fonte: [1]).

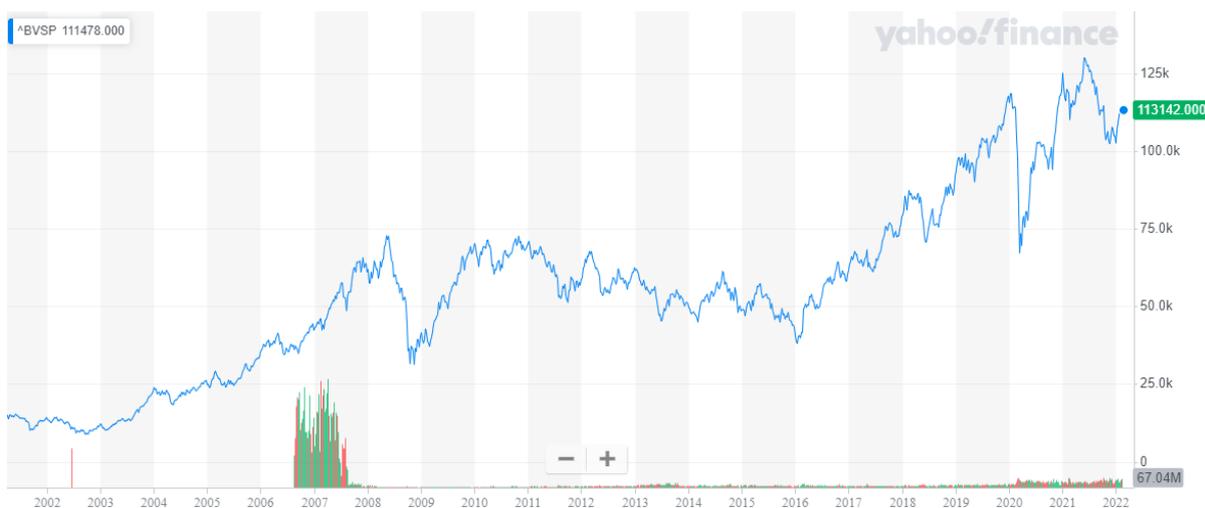


Figura 1.2: Crescimento do índice Bovespa (Fonte: [2]).

Estes crescimentos ao longos dos anos pode ser facilmente explicado pela adoção mundial da internet, e mais recentemente, pode ser atribuído às redes sociais. Novas maneiras de se compartilhar informações e interagir com pessoas surgiram devido ao desenvolvimento das tecnologias digitais e aumento no uso da internet que está presente em grande parte do mundo e em qualquer dispositivo. Sendo assim, seus usuários sejam investidores, companhias, instituições, e em um sentido mais geral, a sociedade, usam as redes sociais para obter e compartilhar informações. Atualmente, as mídias sociais fazem parte do cotidiano, influenciando toda a sociedade e tendo o potencial de mudar o comportamento de um investidor. Essa influência resulta em consequências que terão impacto em vários

níveis do mercado financeiro [7]. Isto posto, algumas variáveis advindas de redes sociais e seus usuários podem ser úteis para prever a performance do mercado.

Os seres humanos são criaturas naturalmente emocionais e sociais, sendo possível observar esta característica no mercado financeiro. De acordo com Lemos [8], a ação do mercado traz embutida a variável psicológica dos investidores, a qual tende a permanecer ao longo do tempo. O comportamento deles, bem como seu grau de aversão ao risco, tende a variar de acordo com a tendência vigente. Ademais, a teoria das finanças comportamentais e análise técnica reforça a existência do fator psicológico no mercado. Ao considerar que os modelos econômicos são falhos para explicar o funcionamento dos mercados, quando não englobam as decisões emocionais e automáticas. Sendo assim, o desejo dos investidores inteligentes é basear suas decisões em critérios objetivos, para elevar o retorno nas operações sem ser afetado por seu próprio fator psicológico.

As possíveis variáveis para analisar o mercado não se resumem a apenas o que tange o aspecto social. Na realidade os principais fatores, são advindos da relação entre o valor que o ativo financeiro oferece e seu valor real. E também, da movimentação gerada primariamente, a partir desta relação que é expressada no valor real do ativo. Esta relação é a base da análise fundamental e a movimentação é base da análise técnica, entretanto, existem duas teorias chamadas de mercado eficiente e passeio aleatório que desafiam estes tipos de análise, a conseguir validar seus efeitos positivos no retorno de um investimento.

Na literatura pode ser encontrado diversos trabalhos correlatos à implementação de Aprendizado de Máquina (do Inglês, *Machine Learning*) (ML) aplicado no mercado financeiro. Alguns trabalhos tentam criar um modelo de previsão de séries temporais usando sistemas *neuro-fuzzy* (lógica nebulosa) [9]. Há trabalhos na literatura que utilizam Regressão de Vetores de Suporte (do Inglês, *Support Regression Vector*) (SRV) para previsão de séries temporais [10]. Jiang [11], compara o desempenho de modelos de previsão usando Perceptron Multicamadas (do Inglês, *Multilayer Perceptron*) (MLP) com modelos de Memória de Longo Curto Prazo (do Inglês, *Long Short-Term Memory*) (LSTM). O trabalho de G. Nascimento [12] usa Redes Neurais Convolucionais (do Inglês, *Convolutional Neural Network*) (CNN) para identificar padrões e gerar uma classificação de compra, venda ou neutro, para automatizar um sistema de negociação de ativos. Sendo assim, é possível notar as variedades de soluções facilitadoras que podem ser concebidas para investimentos no mercado financeiro.

Considerando estas ponderações, cada vez mais se evidencia o desejo de tomar decisões estratégicas e automatizáveis, e por conseguinte, surge o interesse em desenvolver sistemas inteligentes capazes de conseguir identificar o melhor momento de compra e venda para auxiliar o investidor a maximizar seu lucro.

1.1 Problema

O investimento em rendas variáveis requer que o investidor administre seu capital tomando decisões sobre comprar, vender ou se permanecer neutro. Essas decisões devem ser feitas de forma estratégicas e conscientes, levando em conta vários fatores que estão associados ao ativo financeiro que está sendo negociado. Porém, muitas vezes o investidor é afetado psicologicamente por algum acontecimento como notícias de uma empresa, investimentos em prejuízo, ou qualquer situação que leva o investidor à exaustão física e mental. Isto pode afetar negativamente sua capacidade de tomar decisões, e conseqüentemente, levar o investidor à grandes prejuízos. Todas essas questões são vulnerabilidades humanas que ocorrem com a grande maioria dos investidores e adicionam complexidade na gestão de um investimento. Sendo assim, o problema levantado é a possibilidade de aplicação de Redes Neurais Artificiais (RNA) para automatizar decisões de investimento em ativos financeiros.

1.2 Objetivos

O objetivo deste trabalho é a proposição de um sistema baseado em redes neurais que consiga analisar os dados da série temporal de um ativo financeiro. Um modelo de RNA é usado para prever o preço futuro com um passo de um dia. As séries temporais dos ativos financeiros apresentam diferentes padrões de comportamento com o passar do tempo [13]. Sendo assim, o modelo precisa ser adaptado para lidar com estas mudanças. O objetivo é a identificação dos melhores momentos de negociação do ativo. Para que ao final, esta previsão em conjunto de outras ferramentas possa auxiliar o usuário a fazer tomadas de decisões que sejam mais rentáveis. Em seguida, são listados os objetivos específicos deste trabalho:

1. Construir um modelo de rede neural que possa ser aplicado à predição do mercado financeiro;
2. Ser apto à lidar com a mudança nos padrões de comportamento dos dados;
3. Avaliar o desempenho do modelo baseado em métricas estatísticas;
4. Manter um erro de predição baixo com séries temporais tipicamente não-estacionárias;
5. Avaliar o desempenho financeiro do sistema e comparar com estratégias tradicionais.

1.3 Organização

Este trabalho está organizado em 5 capítulos, sendo o presente, o Capítulo 1 de Introdução, onde os principais assuntos que envolvem o trabalho são identificados e enunciados de forma superficial, e o problema a ser resolvido é exposto. O Capítulo 2 de Fundamentação Teórica elucida melhor os temas abordados, trazendo todos os conceitos teóricos necessários para a compreensão do trabalho. No Capítulo 3, o Sistema Proposto é discutido e definido em detalhes, em seguida, o procedimento dos experimentos e seus resultados são explicitados no Capítulo 4. E por fim, no Capítulo 5 é debatido sobre as conclusões deste trabalho, as limitações do sistema e os possíveis trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Este capítulo apresenta conceitos sobre o Mercado Financeiro e os subprodutos que são gerados a partir da moção dos seus processos. Também, apresenta métodos de análise financeira feita pelos investidores para identificar o valor de produtos financeiros. Em seguida é apresentado as teorias sobre séries temporais e como lidar com as mudanças no comportamento dos dados para se obter um modelo mais eficiente. Por último é apresentado a base técnica relacionado à inteligência artificial e redes neurais, como funcionam e os tipos de redes usados no trabalho

2.1 Mercado Financeiro

O setor real e o setor financeiro possuem diferentes características, no primeiro é que se realizam as operações de geração de produtos tangíveis e de serviços não financeiros. Já no segundo são realizadas operações de custódia, intermediação e compensação de ativos considerados não reais. O Mercado Financeiro pode ser definido como um conjunto de instituições e instrumentos financeiros destinados a possibilitar a transferência de recursos dos ofertadores para os tomadores, criando condições de liquidez no mercado [14]. Resumidamente, é o ambiente em que entidades interessadas participam do processo de compra e venda de produtos financeiros. Sendo assim surgem dois fatores que devem ser compreendidos, quem são as entidades e o que são os produtos financeiros negociados.

Os produtos financeiros são ativos intangíveis, como a moeda, os títulos de crédito, as ações que representam cotas-partes das empresas e outros ativos negociáveis em diferentes tipos de mercado. As entidades se referem a todos os participantes que atuam diretamente ou indiretamente, com o Mercado Financeiro. Sendo eles, os emissores que são governos ou empresas que oferecem títulos ou ações a fim de atrair o dinheiro da entidade chamada de investidor, estes podem ser pessoas físicas ou jurídicas [14]. E por último, os intermediários que para compatibilizar os diversos interesses entre emissores e investidores, cumprem a

função de reunir o capital de investidores e emissores. Isto possibilita a alocação eficiente dos recursos financeiros na economia. O papel dos intermediários financeiro é alinhar as necessidades dos investidores com as das companhias abertas [15].

Intermediários, tal como as corretoras de valores, tornam o processo de compra e venda muito mais simples e rápido. Isto reduz consideravelmente a preocupação com a comunicação e com processos burocráticos entre investidores e emissores, e estimulam a economia como um todo. Essas vantagens que a intermediação promove são chamadas de benefícios sociais, e são definidas por Galvão [14] como:

Eficiência operacional O financiamento de forma indireta é processado com mais eficiência do que o financiamento direto. Não é papel do agente que esteja disposto de superavit financeiro, procurar por um agente deficitário, os intermediários automatizam essa relação.

Especialização Considerando os imprevistos, os riscos e as incertezas que ocorrem na economia, torna-se necessário que os intermediários tenham capacidade de previsão e de julgamento, a fim de minimizá-los.

Diluição de riscos O custo das operações financeiras deve cobrir os riscos inerentes às intenções, às possibilidades futuras de liquidações pelos mutuários e os decorrentes de contingências diversas.

Ganhos de eficácia Os intermediários selecionam os empreendimentos financiáveis com base na rentabilidade e retorno, e filtram pelos riscos, resultados privados e benefícios sociais, o que tende a elevar os níveis do resultado financeiro do sistema econômico como um todo.

Descasamento seguro de prazos Os intermediários conseguem operar empréstimos e financiamentos a prazos mais longos com segurança, devido à uma diversificação dos mercados e dos instrumentos de captação de recursos superavitários. Dessa forma, abrangem um maior número de mutuários dispostos de assumir dívidas a prazo e prestações, aumentando os níveis de investimento.

Expansão dos fluxos reais A intermediação financeira estimula os fluxos do setor real da economia, convertendo os excessos de recurso em operações de crédito direcionados ao processo produtivo, e assim, gerando empregos e expandindo o crescimento da economia.

Considerando todas as abstrações dos processos inerentes às operações financeiras e os benefícios que os intermediários promovem. O investidor pode se focar nos produtos

financeiros em si e seu valor atrelado, valor este, que pode ser avaliado e estimado a partir de diversos métodos de análise financeira.

2.2 Teoria do Mercado Eficiente

Na teoria do mercado eficiente se pressupõe que, no mercado existe um grande número de investidores racionais e que maximizam seus lucros competindo uns com os outros. Enquanto tentam prever o valor do mercado futuro de algum ativo, e em que todas as informações relevantes sobre este ativo é inteiramente disponível para todos os participantes [16].

Em um mercado eficiente, a competição entre os participantes inteligentes acarreta uma situação em que, em um ponto específico no tempo, os preços dos ativos já refletem os efeitos da informação. Isto dá uma característica de ajuste instantâneo do preço em relação à novas informações, tornando o comportamento das séries temporais completamente sistemático. Esses efeitos são baseados, tanto em eventos que já ocorreram quanto em eventos em que, o mercado espera que ocorra em algum ponto no futuro. Ou seja, em um mercado eficiente o preço real do ativo financeiro em qualquer ponto no tempo, será uma boa estimativa do seu valor intrínseco [16]. A partir dos estudos sobre o mercado eficiente surge outra teoria, chamada de teoria do passeio aleatório.

2.3 Teoria do Passeio Aleatório

Considerando, um mundo onde o valor intrínseco de um ativo não pode ser exatamente determinado, este cenário provocaria um aumento nas discrepâncias entre o valor real e o valor intrínseco deste ativo. De modo que, sempre haja espaço para desacordos entre os participantes do mercado acerca de qual o valor intrínseco do ativo. Já em um mercado eficiente, as ações dos investidores levariam o valor real do ativo à passear aleatoriamente em torno do seu valor intrínseco. Se essas discrepâncias entre os dois valores fossem sistemáticas ao em vez de aleatórias, os investidores iriam se aproveitar deste conhecimento. Por conseguinte, vários investidores conseguiriam prever a direção em que o valor real do ativo tomaria para se aproximar do valor intrínseco, fazendo com que o comportamento sistemático nas séries temporais se neutralizasse. Sendo assim, apesar das incertezas a respeito dos valores intrínsecos se manter, o preço real dos ativos passaria aleatoriamente ao redor do valor intrínseco [16], invalidando a teoria do mercado eficiente. Estas ponderações são a base para o surgimento da teoria do passeio aleatório.

O mercado onde as mudanças de preço sucessivas nos ativos são independentes, é por definição, um mercado de passeio aleatório. Simplificadamente, a partir da teoria do

passaio aleatório, infere-se que, os preços dos ativos financeiros variam sem memória [16]. Em outras palavras, esta teoria sugere que o histórico passado da série temporal não pode ser usado como ferramenta para prever o seu comportamento futuro.

2.4 Métodos de Análise Financeira

Os métodos de análise financeira são a análise fundamental, a análise sentimental e a análise técnica que são a base para avaliar um ativo e se preparar para negociá-lo. Cada um destes métodos tenta analisar um ativo e identificar seu valor, nenhum deles é completo quando utilizado sozinho, mas quando utilizados em conjunto é possível ter uma melhor perspectiva sobre o ativo. Entretanto, dependendo do tipo de operação de negociação do ativo, é comum depender mais de um só tipo de análise em detrimento dos outros para otimizar a gestão de risco. Tradicionalmente, apenas a análise fundamental e técnica são usadas, pois a primeira faz uso de dados que são disponibilizados pelas próprias empresas via relatórios contábeis e os dados da segunda faz uso do preço e volume do ativo, que estão amplamente disponível na internet. Já para a análise sentimental ser completa é necessário de uma coleta de dados da opinião dos investidores em relação ao ativo.

2.4.1 Análise Fundamental

A análise fundamental se concentra em observar os dados diretamente da própria empresa, o que incluem registros financeiros, relatórios econômicos, ativos da empresa e quota de mercado. Para conduzir uma análise fundamentalista em uma empresa ou setor público, investidores e analistas, geralmente, aferem as métricas do demonstrativo financeiro, balanço patrimonial, declaração de renda e demonstrativo de fluxo de caixa da empresa. Além do mais, os relatórios de ganhos que são lançados durante o comunicado de imprensa de ganhos trimestrais são analisados pelos investidores que tentam averiguar quanto de receitas, despesas e lucros a empresa fez. E quando se faz este tipo de análise, geralmente, procura-se estabelecer parâmetros para a rentabilidade, liquidez, solvência, eficiência e trajetória de crescimento. Sendo que diferentes parâmetros podem ser usados para determinar as condições da empresa, e também, estimar o valor intrínseco do ativo da empresa [17].

Métodos de análise de ações consistem em comparar o demonstrativo financeiro atual de uma empresa com seu demonstrativo financeiro dos anos passados para dar para o investidor uma perspectiva do quanto a empresa está crescendo, estável ou em decaimento. O demonstrativo financeiro de uma empresa pode também, ser comparado à demonstrativos de outras empresas que compõem o mesmo setor de negócios. Como exemplo, pode

ser feita uma comparação da margem de lucro operacional de duas empresas que competem entre si, usando seus demonstrativos financeiros como base. A margem de lucro operacional é uma métrica que mostra o quanto de uma receita sobrou depois de se pagar despesas de uma operação, e quanto sobrou para cobrir os custos não-operacionais [17]. Estas são formas que podem ser usadas para se aferir os fatores fundamentais ao que se refere à um ativo financeiro.

A abordagem da análise fundamental pode ser explicada ao entender o pilar sobre a qual se sustenta. Esse método supõe que em qualquer ponto no tempo um único ativo tem um valor intrínseco, ou um preço que atua como ponto de equilíbrio. Este ponto de equilíbrio depende do potencial de ganho de um ativo, que depende, por sua vez, de fatores fundamentais citados anteriormente. É por meio de estudos desses fatores fundamentais que o analista deveria, em princípio, ser capaz de determinar se o preço atual de um ativo está acima ou abaixo do seu valor intrínseco. Se o preço real tende a mover para o valor intrínseco, então tentar determinar seu valor intrínseco, essencialmente, equivale à fazer uma predição do seu preço futuro [16].

Em relação à análise fundamental, se a negociação de um ativo for como em um mercado eficiente, o valor real(preço) das ações em um ponto específico no tempo, representaria uma boa estimativa para seu valor intrínseco ou fundamental. Portanto, a análise fundamental só acrescentaria algum valor quando o analista possui algum tipo de percepção ou novas informações que não estão implícitas e que ainda não foram completamente integralizadas na formação do preço. Por outro lado, em um cenário onde a teoria do passeio aleatório é válida. O desafio para a análise fundamental é mostrar que os seus procedimentos complexos, que precisam de bastante conhecimento na área de ciências contábeis, são mais lucrativos do que uma simples seleção aleatória de um ativo [16].

Segundo Lemos [8], o problema de se utilizar somente a análise fundamental está no fato de não se levar em conta outros investidores como variáveis. Esse fator é relevante, pois, são as expectativas dos investidores para o futuro que fazem os preços subirem, e não os modelos ou as notícias em si. O fato de o modelo fazer projeções lógicas e coerentes baseadas nas variáveis relevantes não agrega valor para os investidores, pois ou eles não estão cientes das projeções do modelo ou não acreditam nelas. Ademais, a maior parte da atividade gerada nas negociações se deve à resposta a reações emocionais por parte dos investidores. Entretanto, as emoções dos investidores estão totalmente fora dos parâmetros de um modelo, e nem sequer são racionais.

2.4.2 Análise Sentimental

O sentimento do mercado se refere percepção da atitude geral dos investidores em relação ao mercado financeiro. É de consenso que a psicologia dos participantes é refletida nas

atividades e movimento do preço dos ativos negociados no mercado. Em termos gerais, preços subindo indica um sentimento altista, enquanto preços caindo indica sentimento baixista do mercado [18]. Entretanto, este método de análise pode ser feita de forma mais completa do que simplesmente olhar para o preço de um ativo.

O sentimento do mercado nem sempre é baseado nos fundamentos do ativo. Analistas técnicos e *day traders* (investidores que operam à curto prazo) se apoiam nos sentimentos do mercado, pois influenciam os indicadores técnicos que utilizam para aferir e lucrar com movimentos de curto-prazo. Este tipo de movimento é frequentemente causado pela atitude dos investidores em relação ao ativo. A emoção fomenta o mercado financeiro, então o sentimento do mercado nem sempre é correlato ao valor fundamental, mas sim, pertinente aos sentimentos e emoção, enquanto o valor fundamental é sobre a performance da empresa [18].

Existem alguns indicadores de sentimento que são indicadores psicológicos que tentam medir o grau de otimismo ou pessimismo do mercado. Assim como há investidores que lucram tentando encontrar ativos supervalorizados ou subvalorizados baseados no sentimento do mercado, usando estes tipos de indicadores [8]. Entretanto, indicadores são em grande maioria, ferramentas matemáticas calculadas baseadas apenas no preço e volume a partir de alguma premissa, o que faz parte da análise técnica ao vez da análise sentimental. Estes indicadores, não conseguem mensurar o sentimento real dos investidores, como por exemplo, uma pesquisa com as próprias opiniões dos investidores conseguiria.

A atividade das mídias sociais nas quais investidores interagem, segundo os estudos de Piñeiro-Chousa *et al.* [7], influenciam as decisões dos investidores o que leva a uma variação do risco no mercado. Ademais, este estudo mostra como o sentimento nas mensagens tem um efeito importante na relação entre redes sociais e o mercado financeiro. E também, a refere como certas combinações entre diferentes variáveis das redes sociais são responsáveis para este efeito observado no mercado [7]. Portanto, uma melhor alternativa para a realização da análise sentimental é a investigação das redes sociais que pode ser extremamente complexo e custoso.

2.4.3 Análise Técnica

De acordo com Fama [16], a principal suposição de todas as teorias da análise técnica é que a história tende a se repetir, padrões de preço anteriores formados em um ativo tendem a ocorrer novamente no futuro. Aliás, o meio para se prever preços de ativos, e assim, aumentar o potencial de retorno, é desenvolver uma familiaridade com padrões passados do comportamento do preço no intuito de reconhecer situações de provável recorrência. Um estatístico caracterizaria técnicas do tipo, como assumir que variações sucessivas do preço em um ativo são dependentes. Porém, a teoria do passeio aleatório surge com um

desafio, este é um desafio empírico no qual o analista deve demonstrar que suas seleções de ativos são, de fato, melhores que seleções aleatórias.

A análise técnica se diferencia dos outros métodos de análise ao se focar primariamente, no estudo da ação do preço passado e presente para estimar a probabilidade da ocorrência de determinados movimentos futuros. Ao contrário da análise fundamental que tenta identificar o valor intrínseco do ativo. A análise técnica se preocupa com o mercado financeiro como um todo, observando apenas o preço, o volume e os fatores sobre a demanda e oferta que movem o mercado [17]. Edwards *et al.* [19] concorda ao expressar que, o valor real de um ativo é determinado a qualquer instante, inexoravelmente, pela demanda e oferta, que são precisamente refletidos nas transações consumadas nas bolsas de valores. O preço move em tendências e como tal, tendem a continuar até que algo aconteça para mudar o balanço entre oferta e demanda, essas mudanças, geralmente, são detectáveis na própria ação do mercado. Aliás, a existência de características como tendências e sazonalidades são o suficiente para invalidar a teoria do passeio aleatório.

Gráficos de *candlestick* são as principais ferramentas de um analista técnico, um *candlestick* é composto por um preço de abertura, fechamento, máximo e mínimo de um determinado período [20]. Este tipo de gráfico foi desenvolvido em uma multitude de formas e estilos para representar graficamente, praticamente, tudo que acontece no mercado. Seja a ocorrência de uma tendência de baixa ou de alta, o preço varia com uma mudança no volume de negociação, que também pode ser representado no gráfico. Uma das formas tradicionais de se operar um ativo utilizando a análise técnica, é usar esta ferramenta marcando certas regiões como um suporte ou como resistência. Os níveis de suporte são marcados a partir dos menores preços e os níveis de resistência são marcados a partir dos maiores preços atingidos em movimentos anteriores [8]. Baseado nessas regiões de interesse o investidor pode tomar decisões como uma simples regra de “se ou senão”, um movimento acima da resistência sinaliza uma tendência de alta e o movimento abaixo do suporte sinaliza uma tendência de baixa. A título de exemplo, se o preço, de fato, ultrapassar uma resistência o analista opta por operar o ativo em uma posição comprada, senão entra em uma posição vendida. E no caso dos níveis de suporte, se ocorrer a quebra do mesmo, o investidor entra em uma posição vendida, senão entra em uma posição comprada.

Indicadores Técnicos

Os indicadores técnicos são outra importante ferramenta no leque dos investidores, os indicadores são derivados dos dados originais do ativo. Que podem ser calculados a partir de alguma equação, usando uma variável ou combinação de variáveis do preço de abertura, máxima, mínima, fechamento ou volume do ativo a ser analisado. Segundo Lemos [8], os

indicadores utilizados na análise técnica proporcionam uma fonte extremamente útil de informação adicional. Eles ajudam a identificar a impulsão, as tendências, a volatilidade e vários outros aspectos dos ativos para auxiliar na análise técnica, possibilitando gerar sinais de compra e venda.

SMA

A Média Móvel Simples (do Inglês, *Simple Moving Average*) (SMA) é um dos primeiros indicadores a surgir na análise de gráficos, é considerado um indicador atrasado. Com uma simples equação do cálculo da média do preço, de acordo com um tamanho de janela conforme a Equação 2.1,

$$SMA = \frac{\sum_{i=1}^n price}{n} \quad (2.1)$$

sendo *price* o preço e *n* o tamanho da janela contendo os dados. O indicador suaviza os dados para representar a tendência do ativo. Este tipo de indicador não acompanha a direção dos preços, mas define a direção geral da tendência com um certo atraso [8].

RSI

O Indicador de Força Relativa (do Inglês, *Relative Strength Index*) (RSI) é um indicador antecedente criado por Wilder [21]. Este tipo de indicador ao contrário dos atrasados, consegue prever a direção de um movimento. O RSI é um oscilador de impulsão que mede a velocidade e a mudança dos movimentos dos preços ao comparar a média de preços dos períodos de alta com a variação média dos períodos de baixa [8]. Sendo assim, possibilitando que o investidor consiga identificar condições de sobrecompra e sobrevenda. A condição de sobrecompra explicita uma perda na força de compra dos investidores, e por conseguinte, o preço do ativo cai ou se mantém por não haver novos compradores e para a condição de sobrevenda o oposto é verdadeiro,

$$RS = \frac{AverageGain}{AverageLoss} \quad (2.2)$$

$$RSI = 100 - \frac{100}{1 + RS} \quad (2.3)$$

Conforme Wilder [21], primeiramente, é calculado a Equação 2.2 para descobrir a força relativa. O *AverageGain*, que representa a média dos últimos preços que fecharam em um período de alta, é dividido sobre o *AverageLoss*, que representa a média do preços que fecharam em um período de baixa de acordo com um tamanho de janela. Em seguida é usado a Equação 2.3 para calcular o RSI resultando em valores oscilando entre 0 a 100,

com valores próximos de 0 indicando sobrevenda e valores próximos de 100 indicando sobrecompra.

2.5 Séries Temporais

Uma série temporal é uma sequência de dados que são dispostos no tempo. Estes dados podem ser qualquer tipo de observação que possa variar com o tempo. Na literatura existem predominantemente, duas formas de se abordar a predição de séries temporais, os modelos estatísticos tradicionais e métodos inteligentes computacionais. Os modelos estatísticos são baseados na suposição de que há linearidade entre variáveis normalmente distribuídas [22]. Ao contrário dos modelos estatísticos tradicionais, os métodos inteligentes são baseados nos dados e não são paramétricos. Ademais, não requerem nenhum tipo de suposição sobre o modelo e conseguem mapear qualquer função não-linear sem nenhuma pressuposição a respeito das propriedades dos dados [10].

Para se atingir um nível decente de acurácia na previsão de séries temporais é necessário primeiro, saber propriamente analisá-las. A análise de séries temporais faz parte de um ramo da estatística que lida com dependências estruturais entre as observações dos dados e os parâmetros relacionados. E para ser efetivamente realizada Palit e Popovic [3] sugerem que há três importantes passos que devem ser seguidos. O primeiro é a definição e classificação da série temporal, o segundo é a construção do modelo e por último a avaliação dos valores futuros.

A definição da série temporal pode ser feita analisando o conjunto de observações e a definindo conforme um ou mais das características seguintes [3]:

- Estacionariedade: esta propriedade está relacionada à média, variância e covariância se mantendo constantes ao longo do tempo.
- Linearidade: uma série temporal linear pode ser descrita com uma função linear dos valores passados e presentes.
- Tendência: é um componente que pode ser manifestado através de aumentos ou decrementos dos valores das observações.
- Sazonalidade: é o componente que denota um padrão de comportamento que se repete periodicamente.

Conforme o comportamento das séries temporais, estas podem ser classificadas como lineares, não-lineares, estacionárias ou não-estacionárias. Séries temporais são estacionárias se as propriedades estatísticas são constantes ao longo do tempo, não apresentando padrões de tendências ou sazonalidades. E para as séries temporais não-estacionárias o

contrário é verdadeiro, sendo a classificação que mais se encaixa no contexto do mercado financeiro. As séries lineares são geradas por uma combinação linear dos valores presentes e passados junto com fatores aleatórios resumidos em um modelo linear, tal como uma equação diferencial ordinária. Já séries temporais não-lineares, são resultados de combinações não-lineares dos valores presentes e passados junto com fatores aleatórios [13]. Estas devem ser analisadas conforme as particularidades de cada caso, dada a característica caótica que estes tipos de séries temporais podem apresentar. Cujas observações podem ser repetidas aleatoriamente diversas vezes, sem que haja alguma periodicidade definida [3].

A construção do modelo pode ser considerada uma das principais tarefas da análise de séries temporais. Um bom modelo que se encaixe na situação desejada, ajuda a entender como prever o futuro, permitindo que seus usuários tomem decisões mais informadas. Estatísticos tentam descrever os comportamentos de séries temporais através de relações matemáticas entre seus componentes. Entretanto, devido à presença de um componente único encontrado nas séries temporais, chamado de correlação serial entre pontos adjacentes no tempo, restringe a aplicabilidade de vários métodos estatísticos convencionais. Estes métodos dependem da suposição de que as observações dos dados são independentes e distribuídas de forma idêntica. Já as abordagens de ML tentam extrair automaticamente as relações que definem os comportamentos das séries temporais sem a priori, fazer suposições estatísticas sobre os dados [13].

A construção de modelos de Box-Jenkins [23] espera que a série temporal seja estacionária, entretanto dado a natureza da sua metodologia, também pode ser usada como base para séries não-estacionárias. A construção de modelos de Box-Jenkins é ilustrado na Figura 2.1, onde na fase de identificação do modelo é necessário estabelecer uma estrutura inicial para o modelo que combine o suficiente com os dados coletados. Em seguida, quando o modelo preliminar for identificado, ou seja, o número necessário de parâmetros do modelo for determinado. Os valores reais devem ser estimados, enquanto se realiza os procedimentos de treinamento do modelo com o conjunto de dados disponível, por meio do ajuste de parâmetros [3]. E na última fase, se define as métricas de avaliação e de mensuração da acurácia dos resultados obtidos dos valores futuros. Usando o modelo treinado nos dados de teste aplica-se métricas como o Raiz-Quadrada do Erro Médio (do Inglês, *Root-Mean-Square Error*) (RMSE), Equação 2.4, e o Erro Médio Absoluto Percentual (do Inglês, *Mean Absolute Percentage Error*) (MAPE), Equação 2.5,

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}} \quad (2.4)$$

$$MAPE = \frac{100\%}{n} \cdot \sum_{i=1}^n \left| \frac{\hat{y}_i - y_i}{y_i} \right| \quad (2.5)$$

onde \hat{y}_i é o valor futuro previsto, y_i é o valor real e n é o tamanho da janela. Estas duas métricas são algumas das mais utilizadas na literatura.

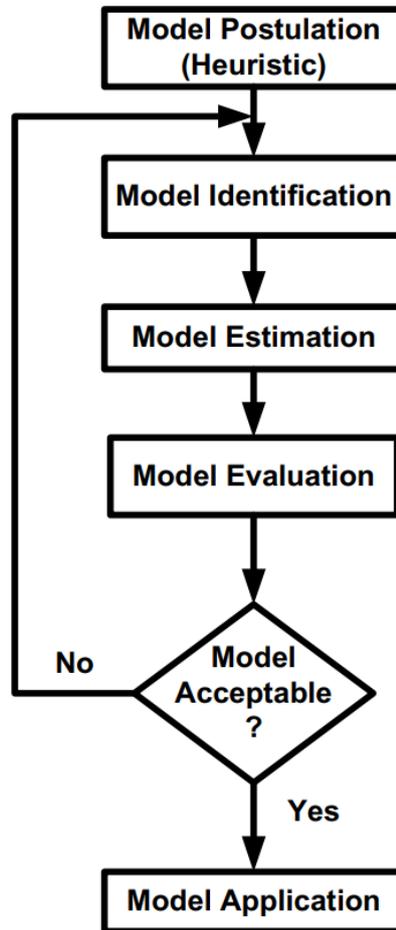


Figura 2.1: Metodologia *Box-Jenkins* da construção de um modelo (Fonte: [3]).

2.6 *Concept Drift*

As séries temporais já são vastamente utilizadas como base para modelos de ML, entretanto é necessário abordar as séries temporais como uma forma de *data stream* (fluxo de dados) constante. Um fluxo de dados é um conjunto de observações que são gerados sequencialmente um por um [24], apresentando um dinamismo inerente ao fluxo de dados. Este dinamismo implica em padrões que evoluem com o passar do tempo e introduz um desafio para algoritmos inteligentes tradicionais. Este desafio se refere à capacidade de manter um modelo acurado mesmo quando ocorre uma mudança no processo de geração

dos dados. Esta mudança é um fenômeno chamado de *concept drift* [13], ou também, pode ser conhecido como *change-point* na literatura da área de estatística.

Os modelos de predição tradicionais frequentemente operam de forma a descartar a continuidade dos dados. Com uma fase de aprendizado usando um conjunto fixo de dados e com a fase de predição sendo executada sem que o modelo seja atualizado apropriadamente, estes modelos acabam sendo ignorantes perante à *concept drifts*. Estes métodos são baseados na suposição de que as séries temporais são estáveis de forma que os dados se comportem conforme uma probabilidade de distribuição constante ao longo do tempo [13]. Entretanto, os preços dos ativos do mercado financeiro podem ser afetados por diversos fatores, como questões macroeconômicas, microeconômicas, políticas ou até psicológicas, estando longe de ser uma série estacionária.

Os desafios dos *data streams* são diversos, tais como o grande volume de dado, *data streams* dinâmicos e a constante mudança no padrão do comportamento dos dados. São características como essas que frequentemente impedem o uso de modelos de decisões estáticos. Com as dinâmicas inerentes, a suposição de estacionariedade não pode ser aplicada à vários setores que precisam lidar com grande fluxo de dados contínuo [13]. Essas características demandam a aplicação de um paradigma diferente e que seja eficiente. Como citado, o desafio é lidar com a evolução de padrões com o tempo, então os dados não podem ser considerados identicamente distribuídos. Um paradigma eficiente que pode ser considerado como um método robusto de mineração de dados é a aplicação de mecanismos para monitorar os dados que são gerados sequencialmente e de formas para detectar mudanças e incorporar estas mudanças no modelo [24]. Estas mudanças são os chamados *concept drifts* que causam uma degradação na acurácia de previsão dos modelos de ML [25].

2.6.1 Adaptação Passiva

Os métodos de adaptação tanto passiva quanto ativa assumem que os dados não são estáveis, tentando lidar com os *concept drifts* com diferentes abordagens. Para a adaptação passiva a essência de todas as abordagens é de efetuar uma adaptação contínua dos modelos de ML. Dessa forma, é possível manter os modelos atualizados, e conseqüentemente, evitar que ocorra uma perda na acurácia da capacidade de previsão do modelo [13].

Apesar da metodologia promover uma melhora significativa no aprendizado de modelos na presença de *concept drifts*, há alguns fatores limitantes. Um desses fatores limitantes é a forma implícita de se lidar com os *concept drifts*, ou seja, neste caso não há como saber se ocorreu uma mudança nos padrões ou não. Ao detectar explicitamente e informar os usuários sobre a ocorrência de *concept drift* aumenta a confiabilidade no sistema e os usuários podem aproveitar o conhecimento de que ocorreu uma mudança no processo de

geração dos dados para tomar alguma decisão [13]. Outro desafio da adaptação passiva, é que as metodologias passivas dependem de retreinamentos sucessivos, podendo ocorrer adaptações excessivas e em decorrência disto, há um desperdício de recurso para um aumento ínfimo da acurácia da predição [26].

2.6.2 Adaptação Ativa

As metodologias de adaptação ativa são aquelas em que se restringem à atualizar o modelo, apenas depois de um *concept drift* ser explicitamente detectado no processo de geração dos dados. Para isso, é necessário que exista um mecanismo de detecção *concept drifts* que normalmente são construídos para se focarem em observar os resíduos do modelo treinado [13]. Os parâmetros do modelo são ajustados conforme os dados, e então são usados para realizar uma predição, esta predição é comparada com o valor real e calculado com algum tipo de erro, estes são os resíduos da predição. Estes resíduos são o que os mecanismos de detecção geralmente observam para tentar verificar alguma mudança no processo de geração dos dados.

Na literatura pode-se também encontrar outras formas de se detectar *concept drifts*, como o monitoramento direto das séries temporais. As abordagens de detecção, em grande maioria, são baseadas na observação dos resíduos. Porém estes tipos de métodos requerem uma análise estatística retrospectiva, exigindo que o mecanismo de detecção seja executado após diversas predições. Entretanto, aplicações de uso real necessitam de uma forma de detectar *concept drifts* em tempo real, já que permite o modelo ser atualizado de forma rápida e eficiente [13].

2.7 Redes Neurais Artificiais

As Redes Neurais Artificiais (RNA) são um tópico da inteligência artificial que apesar de ter sido popularizado apenas nas últimas duas décadas, é uma área que já é estudada desde o século passado no final da Segunda Guerra mundial. Alan Turing já tinha ciência do potencial das máquinas computacionais e escreveu um artigo em 1950 que indagava a capacidade dessas máquinas pensarem. Para conseguir responder esta indagação, Turing criou o jogo da imitação para descobrir se estas máquinas podiam imitar o comportamento humano [27]. Em 1943, McCulloch e Pitts já tentavam descrever um neurônio matematicamente [28]. Seguindo em 1958 Rosenblatt [29] começava estudar sobre os neurônios artificiais dispostos como uma rede, criando o que chamou de *Perceptron* e fazendo grandes contribuições para a área de inteligência artificial. Na universidade de Carnegie Mellon e com as contribuições de Rosenblatt, o pesquisador Pomerleau criou o Veículo Terrestre Autônomo em uma Rede Neural (do Inglês, *Autonomous Land Vehicle*

In a Neural Network) (ALVINN) e testou com sucesso o sistema de navegação concluindo que a rede neural consegue seguir estradas reais sob certas condições [30]. Nota-se que as redes neurais já são antigas e possuem grande potencial na aplicação em diversas áreas.

2.7.1 Neurônio Artificial

O sistema nervoso biológico é uma parte do organismo responsável por transmitir sinais para as outras partes. Com o objetivo de tentar compreender melhor como o cérebro consegue produzir padrões complexos usando as células chamadas de neurônios. Em 1943 Mcculloch e Pitts [28] tentaram descrever os neurônios como um modelo matemático de forma simplificada, este foi o início do surgimento dos neurônios artificiais.

A estrutura básica de um neurônio é representada por um soma(corpo) e um axônio, que com as junções do axônio de um neurônio com o soma de outro, forma a sinapse. Quando há uma excitação na sinapse que, em algum instante, supera o limiar de ativação do neurônio, é gerado um pulso que se propaga à todas as partes do neurônio. De acordo com Mcculloch e Pitts [28], a cada reação de um neurônio há uma afirmação correspondente à uma proposição simples. A fim de construir um modelo matemático que simule este comportamento, Mcculloch e Pitts fizeram as seguintes suposições:

- A atividade dos neurônios é um processo de “tudo ou nada”.
- Um número fixo de sinapses precisam ser excitados durante um período de adição latente para excitar um neurônio em si.
- O único atraso significativo dentro do sistema nervoso é o atraso da sinapse.
- A ocorrência de qualquer sinapse inibitória impede a excitação de um neurônio
- A estrutura da rede não muda com o tempo.

Segundo com Mcculloch e Pitts [28], o estado da rede de neurônios é determinado a partir da caracterização dos estímulos e das atividades dos neurônios constituintes. Ademais, a pesquisa conclui que ao introduzir características para uma rede neural, é possível obter as conexões necessárias para conseguir calcular um estado anterior, a partir das descrições de qualquer estado da rede, entretanto a inclusão de relações disjuntivas impede que um estado anterior seja determinado por completo.

Haykin [4], define um neurônio como uma unidade de processamento de informações que é fundamental para a operação de uma rede neural. E ainda, cita 3 elementos básicos de um modelo neural:

- O conjunto de conexões entre neurônios, chamados de sinapses, podem ser descritos, cada um, como um peso w_{kj} que é aplicado em um sinal x_j , dado uma entrada j conectada a um único neurônio k .
- Uma somatória dos sinais de entrada, ponderados pelos respectivos pesos da sinapse do neurônio, e somado com um *bias* b_k para ajustar o potencial de ativação v_k .
- Uma função de ativação φ que age como um limiar para a amplitude da saída do neurônio, limitando o intervalo da saída, geralmente entre 0 à 1, ou -1 à 1.

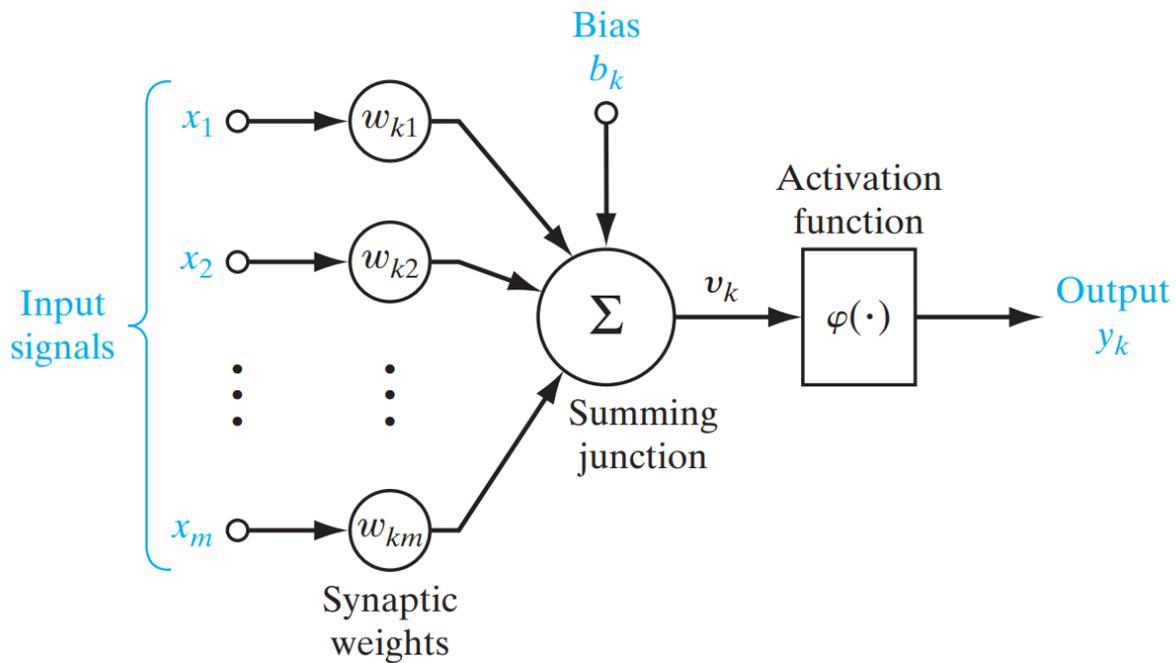


Figura 2.2: Modelo de um neurônio artificial (Fonte: [4]).

Com os elementos básicos citados, é possível construir graficamente o modelo de um neurônio artificial como mostrado na Figura 2.2. Esta é a estrutura básica do neurônio usado em Redes Neurais Artificiais (RNA), tais como as redes Perceptron Multicamadas (do Inglês, *Multilayer Perceptron*) (MLP). E também, baseado nos postulados de Haykin, é possível formular o modelo matemático para o potencial de ativação v_k e para a saída do neurônio y_k ,

$$v_k = \sum_{j=1}^m w_{kj}x_j + b_k \quad (2.6)$$

onde w_{kj} é o peso para cada sinapse j , x_j é a entrada e b_k é o *bias* dado um neurônio k . A saída y_k dado um neurônio k , pode ser escrita como

$$y_k = \varphi(v_k) \quad (2.7)$$

$$y_k = \varphi \left(\sum_{j=1}^m w_{kj} x_j + b_k \right) \quad (2.8)$$

onde φ é uma função de ativação que é aplicada no potencial de ativação v_k .

2.7.2 Funções de ativação

De acordo com McCulloch e Pitts [28], em qualquer instante de tempo um neurônio possui um limiar, no qual uma excitação deve exceder para iniciar um pulso. Este pulso, com exceção do fato e do momento da sua ocorrência, é determinado pelo neurônio e não pela excitação. É esta excitação à que McCulloch e Pitts se referem, que deve ser transcrita em uma forma matemática para conseguir simular um neurônio biológico. A aplicação das funções de ativação também conhecidas por função limiar, é a forma de descrever matematicamente a excitação de um neurônio, no qual o resultado determina a intensidade da saída do neurônio artificial. Mais especificamente, a função de ativação denotado por $\varphi(v)$, define a saída do neurônio em termos do potencial de ativação v . Algumas das funções mais populares são descritas a seguir.

Função de ativação Linear

A função de ativação linear é uma função simples cuja entrada é retornada na saída. Esta função de ativação $\varphi(v_k)$ pode ser descrita matematicamente pela Equação 2.9,

$$\varphi(v_k) = v_k \quad (2.9)$$

onde v_k é o potencial de ativação, ou seja, a saída é exatamente igual à entrada, como mostra a Figura 2.3.

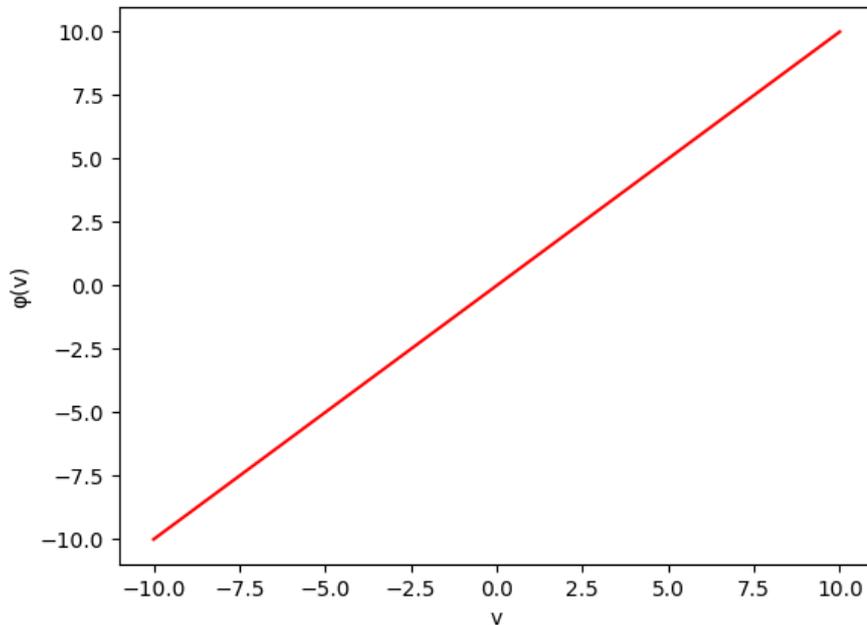


Figura 2.3: Função de ativação linear (Fonte: Elaborado pelo autor).

Função de ativação Degrau (*Threshold*)

A função de ativação degrau também pode ser conhecida como *threshold* ou função de *heaviside* na engenharia. Correspondentemente, a função é descrita matematicamente como na Equação 2.10,

$$\varphi(v_k) = \begin{cases} 1, & \text{se } v_k \geq 0 \\ 0, & \text{se } v_k < 0 \end{cases} \quad (2.10)$$

onde qualquer potencial de ativação v_k maior ou igual a 0 resulta em 1, e uma entrada menor que 0 sempre terá uma saída igual a 0, conforme ilustrado na Figura 2.4.

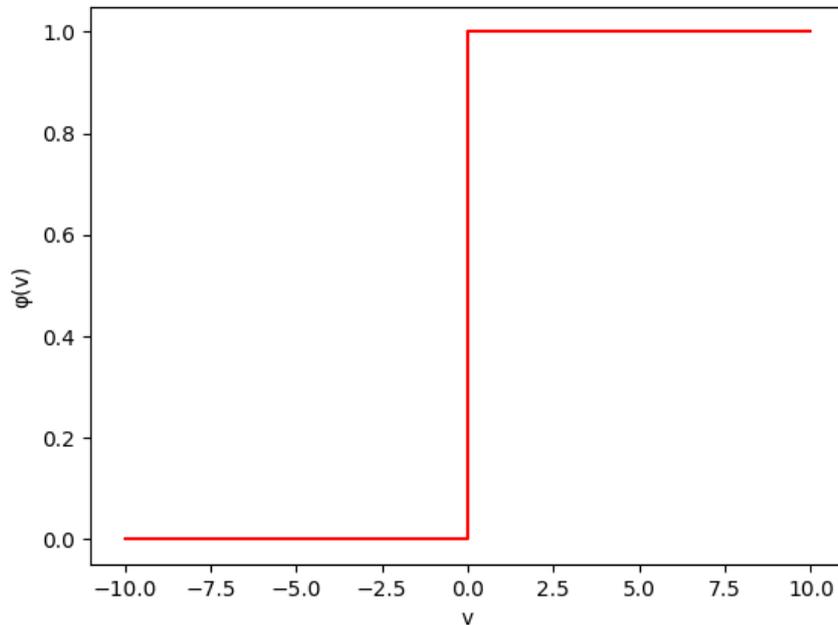


Figura 2.4: Função de ativação de degrau (Fonte: Elaborado pelo autor).

Função de ativação Sigmoide

A função sigmoide tem um gráfico resultante com uma forma de “S”, e é uma das funções mais populares usadas na construção de modelos de ML. Haykin [4] discorre que, este tipo de função de ativação exibe um equilíbrio entre comportamento linear e não-linear, esta função pode ser escrita como na Equação 2.11,

$$\varphi(v_k) = \frac{1}{1 + e^{-av_k}} \quad (2.11)$$

onde v_k é o potencial de ativação e a é a inclinação, que ao variar este parâmetro, obtém-se diferentes inclinações de curva sigmoide. Quanto mais aumentar o valor do parâmetro a maior será a inclinação, ao se aproximar de ∞ a função sigmoide, simplesmente se aproxima de uma função degrau ou função *threshold*. Estes comportamentos são mostrados na Figura 2.5.

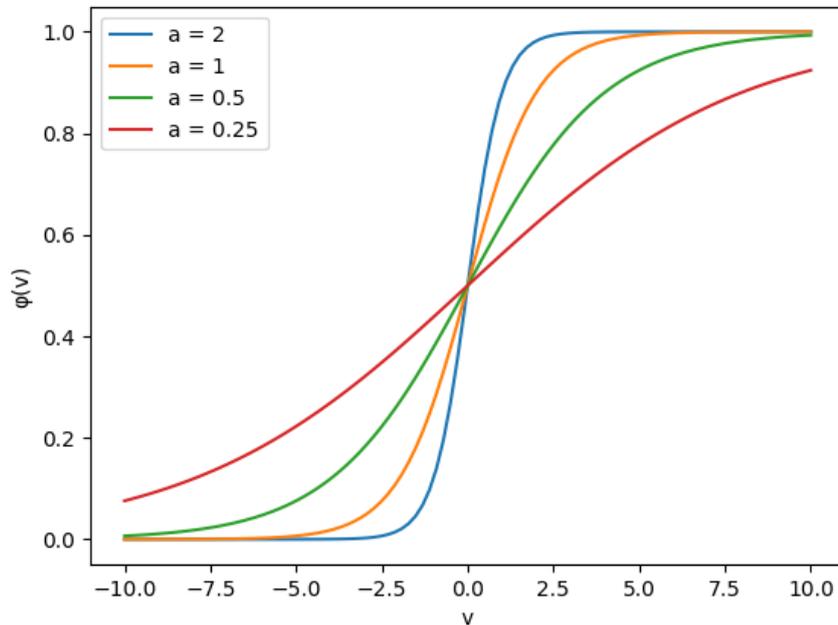


Figura 2.5: Função de ativação sigmoide (Fonte: Elaborado pelo autor).

Função de ativação Tangente Hiperbólica (Tanh)

A função de ativação sigmoide definida na equação Equação 2.11 varia em um intervalo de 0 a +1. Entretanto, por vezes é desejável ter uma função de ativação em que varia em um intervalo de -1 a +1, que no caso, a função de ativação se torna uma função ímpar do potencial de ativação induzido. A função ímpar da função sigmoide é a função Tanh, que pode ser descrita matematicamente como na Equação 2.12,

$$\varphi(v_k) = \frac{1 - e^{av_k}}{1 + e^{av_k}} \quad (2.12)$$

onde v_k é o potencial de ativação e a é a inclinação, assim como na função sigmoide. O comportamento da Tanh pode visualizado na Figura 2.6.

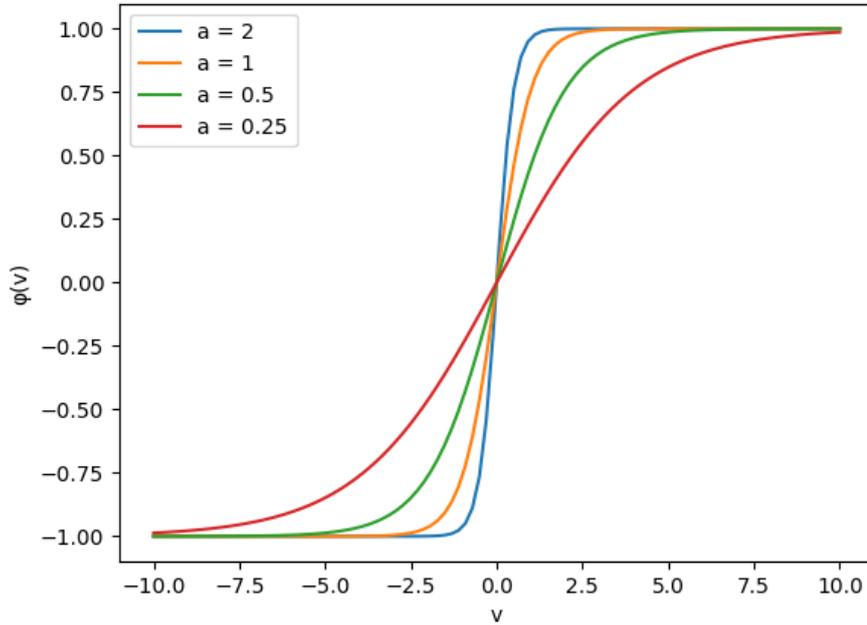


Figura 2.6: Função de ativação Tangente Hiperbólica (Tanh) (Fonte: Elaborado pelo autor).

Função de ativação Unidade Linear Retificada (do Inglês, *Rectified Linear Unit*) (ReLU)

A função de ativação Unidade Linear Retificada (do Inglês, *Rectified Linear Unit*) (ReLU), também conhecida como função rampa [31], se tornou o padrão para muitos tipos de redes neurais. O modelo é mais fácil de ser treinado ao utilizar esta função, e frequentemente, é capaz de atingir melhor performance [32]. Esta função é uma variação da função linear, que pode ser descrita matematicamente pela Equação 2.13,

$$\varphi(v_k) = \begin{cases} v_k, & \text{se } v_k > 0 \\ 0, & \text{se } v_k \leq 0 \end{cases} \quad (2.13)$$

onde v_k é o potencial de ativação fornecido na entrada, caso v_k seja positivo, a saída é exatamente a entrada, caso negativo, a saída é igual à 0, como ilustra na Figura 2.7.

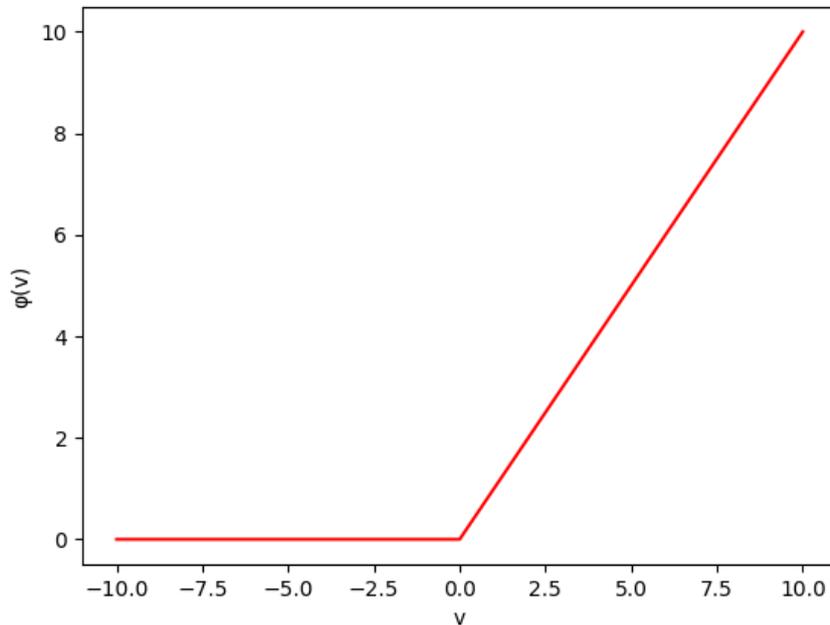


Figura 2.7: Função de ativação ReLU (Fonte: Elaborado pelo autor).

Um dos problemas de se usar funções não-lineares como a sigmoide e Tanh é que elas saturam. Valores que são muito altos são retornados como +1 na saída e valores muito baixos são retornados como -1 na saída da Tanh, ou 0 no caso da sigmoide. Estes tipos de funções são sensíveis, apenas à valores medianos. A saturação e a sensibilidade limitada são características que ocorrem independentemente do potencial de ativação produzir informações úteis. Quando a função está saturada se torna difícil algoritmos baseados no gradiente descendente ajustarem os pesos e melhorar a acurácia do modelo [32].

Algumas das vantagens de se usar a função ReLU é a simplicidade computacional promovida, já que as funções de ativação Tanh e sigmoide requerem o cálculo da exponencial. Outra vantagem que pode ser citada, é que as funções não-lineares aprendem a aproximar do valor 0, mas não o verdadeiro valor 0 que a função ReLU consegue atingir. Este fato, confere uma propriedade chamada de representação esparsa que pode simplificar o modelo e acelerar o seu aprendizado. A função de ativação ReLU, basicamente age como uma função linear, que geralmente são mais fáceis de se otimizar [32].

2.7.3 Problema de Otimização

O problema de otimização é um problema matemático que pode ser encontrado em diversas áreas, este problema é ocasionado quando há o desejo de minimizar ou maximizar alguma função. A tarefa de se maximizar uma função $f(x)$ pode ser atingida minimi-

zando $-f(x)$. A função que se quer minimizar recebe o nome de função do erro, no caso de problemas de otimização no escopo de ML. A função de erro mais comum é o Erro Médio Quadrático (do Inglês, *Mean Squared Error*) (MSE) descrito como na equação Equação 2.14,

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 \quad (2.14)$$

onde n é a quantidade de amostras, y_i é o valor real e \hat{y}_i é a saída da rede. Algoritmos de ML envolvem alguma forma de otimização em vários contextos. Dentre eles, a tarefa mais desafiadora é o do treinamento das redes neurais. Este problema é extremamente importante, dado a complexidade e custo envolvido, para isso foram criadas diversas técnicas de otimização especializadas para Redes Neurais Artificiais (RNA).

Gradiente Descendente

A técnica chamada de gradiente descendente foi proposta inicialmente por Cauchy em 1847 [33]. Motivado por cálculos astronômicos, Cauchy queria solucionar não as equações diferenciais, mas as equações algébricas que representam a órbita dos corpos celestiais. Esta técnica baseia-se em derivar uma função $y = f(x)$, o que resulta em $f'(x)$, representando o grau de inclinação no ponto x , com este resultado é possível descrever como uma pequena mudança na entrada x consegue provocar uma pequena melhoria na saída y . Então, o processo desta técnica se resume em mover x iterativamente em pequenos passos com o sinal oposto da derivada, para reduzir $f(x)$ [6]. O aprendizado no contexto de ML se dá de forma semelhante, conforme a Equação 2.15,

$$w'_{kj} = w_{kj} - \epsilon \nabla E \quad (2.15)$$

onde, w'_{kj} é o novo peso calculado de cada sinapse j para um neurônio k , w_{kj} é o peso que deve ser ajustado e ∇E é o gradiente de uma função de erro, que contém todas as derivadas parciais. O gradiente orienta o modelo a aumentar ou diminuir os pesos para otimizar a função do erro e ϵ é a taxa de aprendizado que determina o quanto os pesos serão ajustados [6]. O aprendizado de um modelo terá um hiperparâmetro chamado de época com um valor definido previamente. A passagem de uma época é equivalente ao processamento de todo o conjunto de dados, realizando o ajuste de cada um dos pesos que compõem a rede, e consequentemente fazendo com que a rede “aprenda”.

2.7.4 *Multilayer Perceptron*

A primeira rede neural descrita na forma de um algoritmo foi proposta por Rosenblatt, que queria buscar formas de se entender a capacidade de organismos pensar, reconhecer, generalizar e recordar. Com isto em mente, Rosenblatt se foca em responder duas questões em sua pesquisa. A primeira é, como a informação é armazenada ou lembrada, e a segunda é, como esta informação influencia o reconhecimento e comportamento. Para Rosenblatt [29], o sistema nervoso central age como uma rede de comutação, onde a retenção de informação é dada por novas conexões entre centros de atividade ao invés de ser propriamente armazenada. Esta teoria combina respostas para as duas questões, sendo a base para a pesquisa da máquina chamada de *perceptron*. A *perceptron* foi construída para ilustrar algumas das propriedades de sistemas inteligentes, excluindo-se os fatores desconhecidos dos organismos biológicos.

A rede *perceptron* é uma rede composta de somente uma camada de neurônios, por este fato, a rede é extremamente limitada, sendo capaz de resolver apenas problemas de classificação de padrões que podem ser separados linearmente [4]. A fim de superar estas limitações a rede neural Perceptron Multicamadas (do Inglês, *Multilayer Perceptron*) (MLP) surgiu em 1986 [34]. Esta rede é caracterizada por ter uma função de ativação não-linear para cada neurônio e podendo possuir uma ou mais camadas de neurônios escondidas da entrada e saída, que é onde ocorre o processamento dos sinais. Além disso, em uma camada, todos os neurônios são totalmente conectados aos neurônios das camadas vizinhas, apresentando um alto grau de conectividade [4]. Estas características denotam uma dificuldade no processo de aprendizado do modelo, pois a adaptação dos pesos deve ser conduzida em um escopo muito maior de possíveis funções.

2.7.5 *Backpropagation*

A retro-propagação de erros ou *backpropagation* é um algoritmo de ajuste de pesos de redes neurais criado, especificamente, para o treinamento supervisionado de redes neurais *feedforward*. O treinamento supervisionado é um tipo de treinamento em que se fornece junto a uma entrada da rede, uma saída que se espera que essa entrada resulte. As redes neurais *feedforward* são um tipo de rede onde a saída de cada neurônio é propagada para os neurônios vizinhos da camada seguinte apenas uma vez. Sendo assim, é necessário que a rede seja *feedforward*, que exista um valor esperado e que exista uma função de erro derivável, para que a aplicação do algoritmo de *backpropagation* seja possível. Isto posto, o algoritmo é executado em duas fases, segundo Haykin [4]. A primeira é a fase em que o fluxo ocorre para a frente, os pesos da rede são fixados e a entrada da rede é propagada do começo, através das camadas escondidas e até a saída da rede, com as

mudanças ocorrendo apenas nos potenciais de ativação. Na segunda fase do algoritmo, um sinal de erro é calculado comparando a saída da rede com o valor esperado. Em seguida é propagado camada a camada, mas desta vez, a partir da saída até a entrada da rede, originando no nome de *backpropagation*. Dessa forma, sucessivos ajustes dos pesos das sinapses da rede são realizados.

O *backpropagation* é necessário, pois no treinamento supervisionado, a saída da rede depende de todos os pesos das sinapses. O erro é calculado apenas no final do processamento e comparado com a saída esperada. No caso de um problema de regressão, uma função como o MSE dado pela equação Equação 2.14, é usada para obter o erro, e este deve ser propagado até a entrada para ser possível ajustar todos os pesos. E como o fator de interesse é o peso das sinapses, deseja-se saber o quanto o erro varia conforme a variação do peso, o gradiente do erro é descrito na Equação 2.8,

$$\nabla E = \frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial v_k} \frac{\partial v_k}{\partial w_{kj}} \quad (2.16)$$

tal que o gradiente do erro ∇E é determinado em função dos pesos sináptico w_{kj} . Sendo possível ser resolvida pela regra da cadeia, onde v_k é o potencial de ativação e y_k é a saída do neurônio com o gradiente da função do erro, é possível aplicar a Equação 2.15 e realizar o ajuste dos pesos.

Estimação Adaptativa do Momentum (do Inglês, *Adaptive Momentum estimation*) (Adam)

A função do erro pode possuir diversos mínimos locais que podem acabar sendo identificados como mínimos globais, e assim, degradar sua performance. Para prevenir isto, é necessário monitorar e ajustar a taxa de aprendizado apropriadamente, e é isto que os algoritmos de otimização atuais fazem automaticamente. Segundo Goodfellow *et al.* [6], a maior dificuldade dos algoritmos de otimização é, notavelmente, o ajuste da taxa de aprendizado, pois tem um impacto significativo no desempenho do modelo. O erro, frequentemente, acaba sendo bastante sensível à alguns tipos de parâmetros e insensível para outros. Dessa forma, é interessante o uso de diferentes taxas de aprendizado para cada parâmetro e adaptar automaticamente estas taxas de aprendizado ao longo treinamento.

Nos últimos anos vários métodos incrementais de adaptação da taxa de aprendizado dos parâmetros dos modelos foram introduzidos. Entre eles o Algoritmo do Gradiente Adaptativo (do Inglês, *Adaptive Gradient Algorithm*) (AdaGrad), Propagação da Média Quadrática (do Inglês, *Root Mean Squared Propagation*) (RMSProp) e Estimação Adaptativa do Momentum (do Inglês, *Adaptive Momentum estimation*) (Adam) [6]. Adam foi apresentado por Kingma e Lei Ba [35] como um dos algoritmos de otimização que imple-

mentam estes tipos de métodos. Adam pode ser usado como um substituto do gradiente descendente estocástico para atualizar os pesos da rede iterativamente. Diferentemente do gradiente descendente estocástico, que mantém apenas uma taxa de aprendizado constante ao longo do processo de otimização para ajustar os pesos da rede. O algoritmo Adam usa diferentes taxas de aprendizado para cada peso da rede, e cada um deles é adaptado separadamente [36].

Para os autores Kingma e Lei Ba [35], Adam pode ser considerado como uma combinação dos algoritmos AdaGrad e RMSProp. O algoritmo de AdaGrad adapta a taxa de aprendizado de todos os parâmetros fazendo o escalonamento inversamente proporcional à raiz quadrada da soma de todos os valores quadráticos históricos do gradiente [37]. Os parâmetros têm uma queda proporcional à intensidade dos valores da derivada parcial, conferindo grande performance em gradientes esparsos. E o algoritmo RMSProp, que também mantém uma taxa de aprendizado para cada parâmetro, adapta esta taxa de aprendizado baseado na média das magnitudes mais recentes dos gradientes, se sobresaindo em problemas com séries não-estacionárias. Adam adapta as taxas de aprendizado baseado na média do primeiro momento como no RMSProp, mas também, usa a média do segundo momento dos gradientes [36]. Como uma combinação dos algoritmos AdaGrad e RMSProp, Adam admite o melhor dos dois mundos.

2.7.6 *Long Short-Term Memory*

A rede Memória de Longo Curto Prazo (do Inglês, *Long Short-Term Memory*) (LSTM) é um tipo de Rede Neural Recorrente (do Inglês, *Recurrent Neural Network*) (RNN) proposto inicialmente por Hochreiter e Schmidhuber em 1997 [38]. Foi desenvolvida para solucionar dificuldades matemáticas fundamentais provenientes da construção de modelos para sequências longas de dados. Uma RNN é uma adaptação das redes *feedforward*, onde a informação é propagada pela rede da entrada até a saída apenas uma vez. Já nas RNN as informações podem também, ser repassadas para o mesmo nó em um auto-ciclo. Ao realizar o cálculo de uma saída o nó de uma RNN leva em consideração a entrada atual e as saídas anteriores, dando uma característica de memória de curto-prazo [5].

O modelo inicial de Hochreiter e Schimidhuber [38] tinha como objetivo resolver dois problemas provenientes do uso do *backpropagation* convencional. O primeiro é o problema do Gradiente Explosivo (do Inglês *Exploding Gradient*), onde grandes gradientes do erro se acumulam e resultam em atualizações significativas dos pesos da rede neural durante o treinamento [39]. O segundo é o problema da Dissipação do Gradiente (do Inglês *Vanishing Gradient*), que ocorre quando os gradientes propagados se tornam muito próximos de 0 e acabam sendo computados como 0, neste caso, o peso não é atualizado. No primeiro

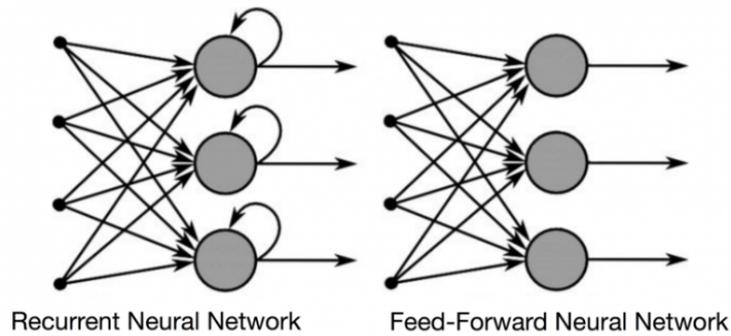


Figura 2.8: RNN e redes *feedward* (Fonte: [5]).

problema, pode acarretar em pesos instáveis e no segundo, acarreta na dificuldade do aprendizado quando as entradas são temporalmente distantes [4].

A LSTM foi desenvolvida para superar estes problemas, sua arquitetura permite um constante fluxo do erro através de unidades conectadas a si mesmas [38]. A arquitetura de LSTM passou por algumas mudanças durante os anos, uma melhora crucial para a arquitetura foi tornar os pesos nos auto-ciclos condicionados ao contexto ao invés de mantê-los fixos. Ao adicionar os componentes chamados de portões, a escala temporal da integração pode ser mudada dinamicamente [6]. Estes componentes adicionais resultam em uma unidade mais complexa chamada como célula de memória.

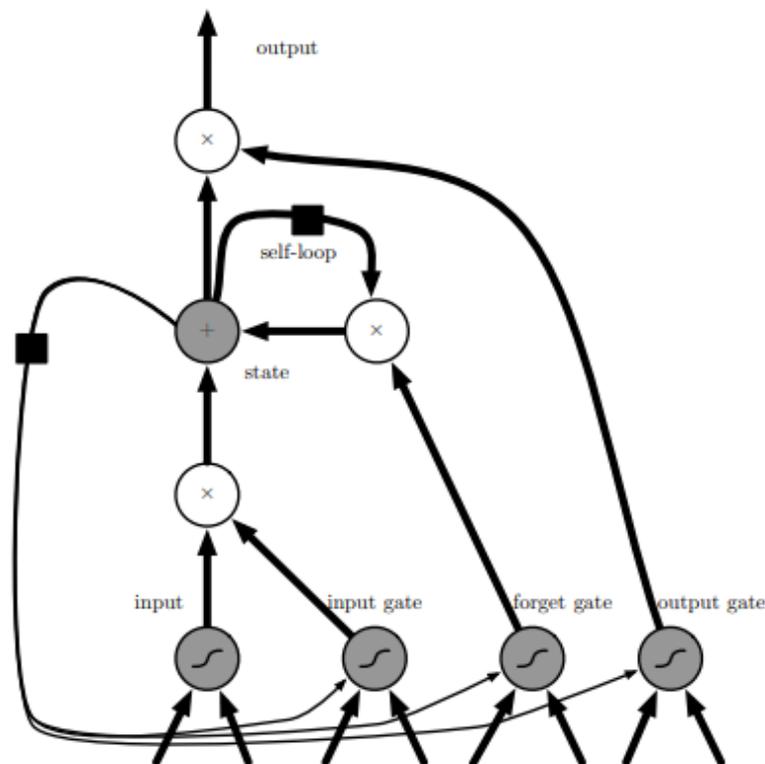


Figura 2.9: Célula da rede LSTM (Fonte: [6]).

Segundo Goodfellow *et al.* [6], as células de memória são conectas recorrentemente entre si, substituindo as unidades escondidas convencionais das RNN. Um portão de entrada é calculado com uma unidade de neurônio artificial, com o valor podendo ser acumulado no estado se o portão permitir. A unidade de estado tem um ciclo em si mesmo, onde o peso é controlado por um portão de esquecimento. A saída da célula pode ser desligada pelo portão de saída. Todas as unidades de portão possuem uma não-linearidade sigmoide, enquanto a unidade de entrada pode ter qualquer não-linearidade. A unidade de estado pode ser usada, também, como uma entrada extra.

2.8 *K-Fold*

O *K-Fold* é um dos métodos estatísticos para se avaliar um modelo de ML, estes métodos estatísticos são chamados de validação cruzada. O procedimento do *K-Fold* possui um parâmetro chamado k que se refere ao número de grupos ou dobras em que os dados são separados. No geral cada dobra possui um conjunto de dados para o treinamento e teste do modelo. A validação cruzada é usada para estimar a capacidade de generalização de um modelo de ML, quando usado para fazer previsões a partir de dados que não foram vistos pelo modelo durante o treinamento [40]. Isto é feito para simular uma situação semelhante ao uso de caso real com um conjunto de dados limitado, e assim realizar uma avaliação mais fidedigna. O processo do *K-fold* é ilustrado na Figura 2.10, este método consiste em dividir aleatoriamente os dados disponíveis em k dobras de conjuntos de teste e treino. A cada valor de k um par de conjunto é extraído, tal que não sejam iguais ao conjunto de outra dobra, e assim, é calculado a média dos erros das dobras. Dessa forma é possível realizar a validação sem que o modelo esteja enviesado.

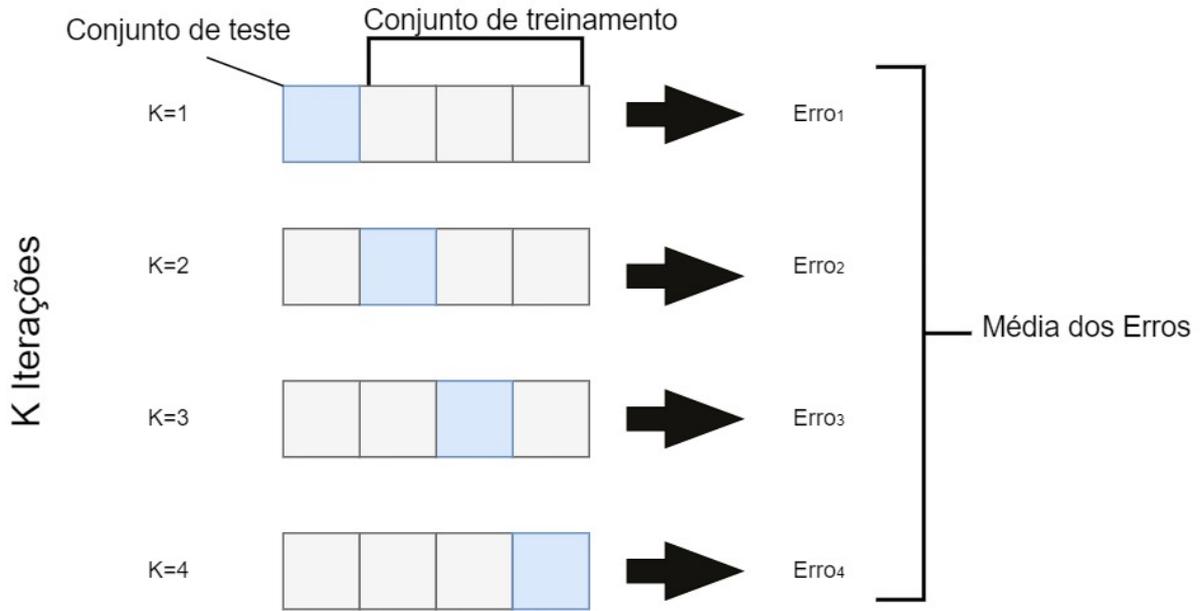


Figura 2.10: *K-Fold* tradicional (Fonte: Elaborado pelo autor).

Este capítulo explicou conceitos básicos do mercado financeiro necessários para a compreensão do objetivo do trabalho proposto, juntamente com as séries temporais e seu comportamento que são geradas a partir dos processos do mercado financeiro e formas de se analisar estes dados. Também, discorreu-se sobre os conceitos envolvidos em uma RNA. No capítulo seguinte serão expostos o sistema proposto e as metodologias usadas no trabalho.

Capítulo 3

Sistema Proposto

Neste capítulo serão explicitados de forma detalhada as metodologias usada no processo de treinamento das redes neurais, todas as questões que envolvem a construção do modelo de rede neural, seus hiperparâmetros e algoritmos usados. Em seguida será introduzido o sistema proposto neste trabalho para a se fazer uma tomada de decisão baseado nas saídas da rede que será a previsão da séries temporal de um ativo financeiro. Com a introdução do sistema, o algoritmo usado para a decisão é demonstrado e explicado. E por fim, são listadas todas as bases de dados que foram utilizadas no trabalho, juntamente à metodologia de avaliação estatística e financeira do trabalho para verificar a viabilidade do sistema proposto.

3.1 Visão Geral

Como discutido nos capítulos anteriores, com o mercado financeiro em ascensão, o avanço da tecnologia na área também está progredindo. Vários sistemas facilitadores surgiram e permitiram ao pequeno investidor ser integrado neste mundo financeiro, e como tal, o interesse nesta área é puramente capitalista, visando o lucro advindo da operação no mercado financeiro. Com o mercado financeiro gerando várias séries temporais a todo momento, e considerando que estas, são suficientes para se ter noção da condição do ativo financeiro. As Redes Neurais Artificiais (RNA) são uma óbvia escolha para construir sistemas inteligentes, dada a capacidade de generalização que possuem. Ademais, a proposta do trabalho parte da ideia de que as séries temporais geradas pelo mercado financeiro e os indicadores que podem ser calculados a partir das mesmas, provêm informação o suficiente a cerca do ativo. O que torna possível identificar fatores técnicos, sentimentais e fundamentais, apenas a partir destes dados.

O trabalho proposto foi desenvolvido em *python* usando o pacote *keras*, dado a grande popularidade, flexibilidade, suporte à ciência de dados e ML que esta linguagem de progra-

mação consegue prover. Além de que, *python* tem a capacidade de facilmente desenvolver APIs Interface de Programação de Aplicação (do Inglês, *Application Programming Interface*) (API), o que é necessário para implementar o sistema no cenário real. A linguagem *MATLAB*, por exemplo, apesar de eficiente nas áreas de ciências de dados, cálculo numérico e ML, não possui a flexibilidade no desenvolvimento de API como *python*.

3.1.1 Arquitetura do Sistema

A arquitetura do sistema pode ser diagramada como mostrado na Figura 3.1.

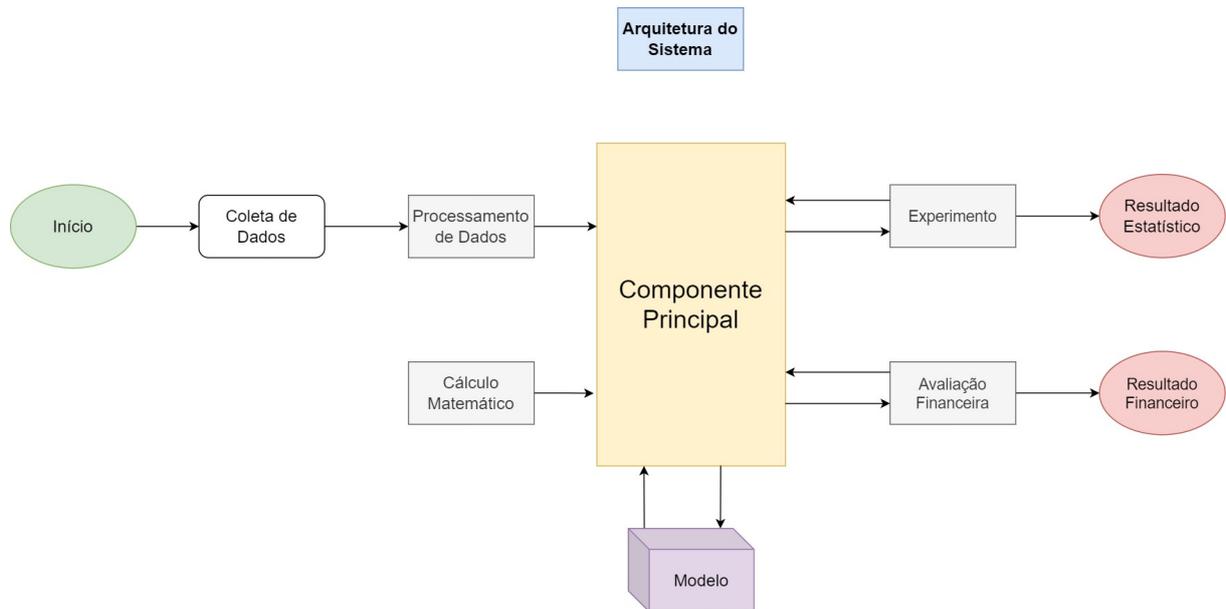


Figura 3.1: Diagrama do sistema proposto (Fonte: Elaborado pelo autor).

A arquitetura do sistema foi desenvolvida e separada em pequenos módulos que são alimentados no componente principal que organiza, decide e os executa. O processo do algoritmo de negociação se inicia com a coleta de dados, estes são adquiridos da plataforma do *yahoo finance*. Estes dados são então processados pelo módulo *Data Processor* que é responsável por fazer um pré-processamento para retornar diferentes conjuntos de dados no formato correto, salvar e ler arquivos, e lidar com os dados brutos no geral. O módulo *Math Handler* é responsável por calcular indicadores técnicos, resíduos estatísticos e o processo de procura do mínimo global. O módulo *Experiment* é usado para realizar os experimentos com a previsão das séries temporais, a avaliação estatística e gerar o modelo. E por último, o módulo *Finance Processor* é responsável por realizar a simulação do modelo, e que portanto, comporta a estratégia de decisão do sistema que pode ser visualizado no fluxograma da Figura 3.2.

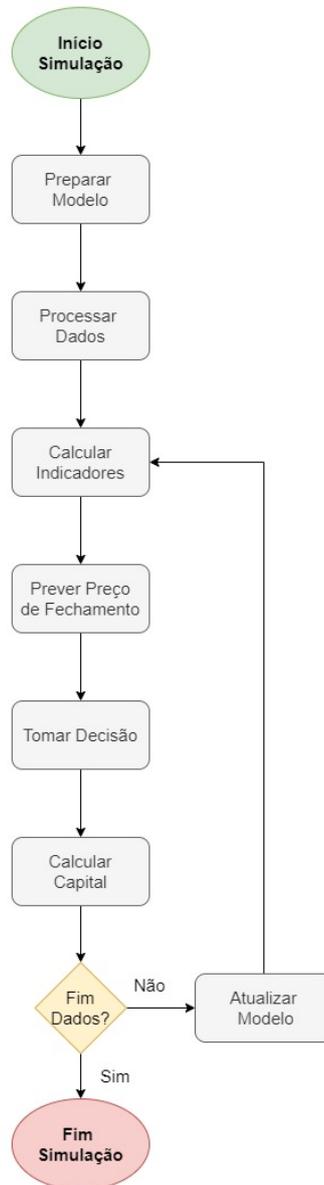


Figura 3.2: Fluxograma do módulo *Finance Processor* (Fonte: Elaborado pelo autor).

A simulação do sistema de avaliação financeira se inicia a partir das etapas anteriores do componente principal. Após a realização dos experimentos o componente principal passa os dados relevantes e o modelo de RNA a ser usado, para o módulo *Finance Processor*. Dentro deste módulo, é executado a preparação do modelo onde o treinamento do modelo para o conjunto de dados de treinamento é realizado. Em seguida é feito o processamento dos dados, obtendo-se o conjunto de dados de simulação no período correto para serem manipulados, e enfim, os indicadores técnicos são calculados. Após estas etapas, há um ciclo de repetição até o final do conjunto de dados para simular um cenário real, em que a cada novo preço de fechamento do dia, o processamento é realizado incluindo esta

nova amostra. Primeiramente, os indicadores técnicos da nova amostra são calculados, em seguida o preditor é executado para se obter o preço de fechamento do próximo dia. Com esta previsão, é executado o sistema de decisão, que retornará um sinal de compra, venda ou neutro. Este sinal é necessário para que o algoritmo de avaliação financeira possa ser executado. Este algoritmo simula uma compra ou venda de um ativo financeiro e calcula o capital total que o sistema atingiria.

3.2 Séries Temporais

3.2.1 Coleta dos Dados

As séries temporais foram obtidas a partir da plataforma do *yahoo finance* [41] usando o pacote *pandas_datereader* [42] que faz uma requisição para uma plataforma de escolha, de acordo com o símbolo e o período desejado. Assim, retorna um *dataframe* com diversos dados sobre os ativos escolhidos, que podem ser visualizados com mais detalhes na Tabela 3.1.

Tabela 3.1: Ativos financeiros usados no trabalho.

Moeda	Símbolo	Descrição	Data Inicial	Data Final
Dólar	BTC-USD	Bitcoin	17/09/2014	15/09/2021
Real	TAE11.SA	Transmissora Aliança de Energia Elétrica S.A.	01/01/2011	01/01/2022
Real	VALE3.SA	Vale S.A.	01/01/2009	01/01/2019
Real	ITUB4.SA	Itaú Unibanco S.A	01/01/2009	01/01/2019
Real	BOVA11.SA	Ishares Ibovespa Fundo de Índice	01/01/2009	01/01/2019
Real	PETR4.SA	Petróleo Brasileiro S.A	01/01/2009	01/01/2019

O *dataframe* obtido é composto pelo preço mínimo, máximo, de abertura e de fechamento, a data e o volume do ativo financeiro. É também, importante explicitar que no caso do mercado financeiro, é possível extrair as séries temporais com um intervalo de 1 minuto. Entretanto o sistema precisaria levar o tempo em consideração, então cada dado é extraído em período diário.

3.2.2 Janela Deslizante

O processamento dos dados é feito usando o método de janela deslizante como base. Este método é uma forma de reestruturar os dados de séries temporais em um vetor de entrada de dimensão definida, permitindo o uso de modelos clássicos não-recursivos de ML [43]. A abordagem de janela deslizante é usada no trabalho para transformar o problema de previsão de séries temporais em um problema similar à classificação de

padrões. O procedimento da janela deslizante é ilustrado na Figura 3.3, onde p representa o preço de fechamento do dia, w representa o tamanho da janela e n é o tamanho do conjunto de dados disponíveis.

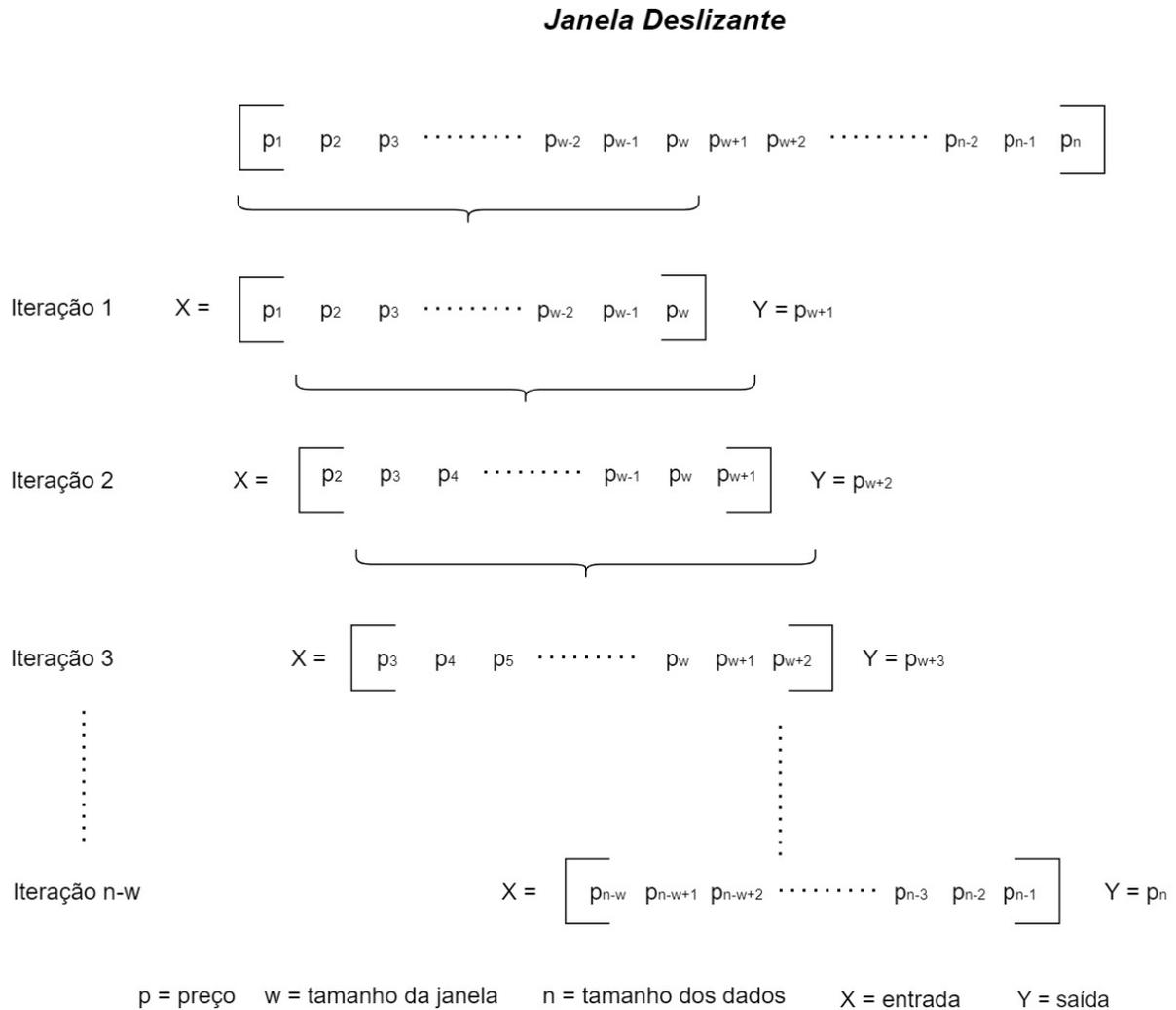


Figura 3.3: *Sliding window* implementado no trabalho (Fonte: Elaborado pelo autor).

Para realizar o aprendizado supervisionado é necessário um vetor de entrada X que representa o sub-conjunto de amostras temporais fornecido para a entrada, e uma saída Y fornecido como valor esperado do previsor. O procedimento se inicia montando um sub-conjunto temporal X , como um vetor de dimensão igual ao número w de preços de fechamento p , indo de p_1 até p_w , e atribuindo p_{w+1} à saída desejada Y . Deste modo, cada iteração desliza a janela com um passo de 1 dia e continua até o fim do conjunto de treino. A última iteração $n - w$ resulta em vetor de entrada X com as amostras de p_{n-w} até p_{n-1} e a correspondente saída $Y = p_n$, o valor predito.

3.2.3 Processamento dos Dados

Após a coleta das séries temporais, é realizada a etapa de processamento dos dados. Esta etapa, retorna vários conjuntos de dados, dependendo do módulo a ser executado, para o treinamento dos modelos, um conjunto do preço de fechamento é gerado. A partir deste, a rede é alimentada usando o método da janela deslizante, com um sub-conjunto para os valores de entrada e valor esperado da saída da rede. Para o *k-fold* e avaliação financeira, é feita a separação dos dados em blocos de um ano, considerando $k = 4$, os 4 últimos anos são utilizados para teste e simulação. Ademais, na avaliação financeira, subconjuntos adquiridos a partir dos blocos de um ano são usados para calcular os indicadores técnicos dependendo do tamanho de janela escolhido do SMA e RSI, conforme as Equações 2.1 e 2.2.

3.3 Treinamento

O treinamento de Redes Neurais Artificiais (RNA) é uma parte essencial do processo para tornar uma rede neural funcional. Aliás, como foi explicado no capítulo anterior é este processo que a faz assimilar um conjunto de dados, generalizá-lo e assim, aprender. Isto posto, a metodologia de treinamento tem grande impacto na performance do modelo, especialmente em cenários onde há *concept drifts* presentes nas séries temporais. Então é necessário um esforço significativo em criar uma metodologia de treinamento eficiente e manter o modelo adaptado. Portanto, foi realizado uma pesquisa com foco em trabalhos correlatos que consigam lidar com *concept drifts*. Como explicado no Capítulo 2, há várias formas de enfrentar este problema, com as soluções sendo classificadas em adaptação passiva e ativa na literatura. Apesar da adaptação ativa ser mais eficiente, ela é substancialmente mais complexa, então optou-se por implementar a abordagem de adaptação passiva, na qual o modelo é constantemente atualizado a cada nova amostra. Mais especificamente, todo novo preço de fechamento do dia é incluído no modelo como valor esperado em um treinamento supervisionado, e a entrada da rede são os preços de fechamento dos últimos w dias, com w sendo o tamanho de janela escolhido.

3.4 Avaliação Estatística

O método de validação usado nos experimentos é o *k-fold*, no entanto, no caso do mercado financeiro, não é possível fazer a divisão aleatória dos dados como da forma tradicional. O preço de um ativo em um ponto no tempo t , depende dos preços passados $t - n$, tal que para os dados fazerem sentido é necessário haver uma continuidade entre o conjunto de treino e o conjunto de teste. Então, o *k-fold* é implementado excluindo os últimos blocos

dos dados completo, e deslocando um bloco a cada iteração de k , como apresentado na Figura 3.4. Além do mais, como os conjuntos de teste são compostos dos últimos blocos de dados continuamente, não é necessário calcular o erro para cada uma das dobras e fazer a média. É suficiente que o cálculo do erro seja feito incluindo os resultados de todas as dobras para calcular um só erro.

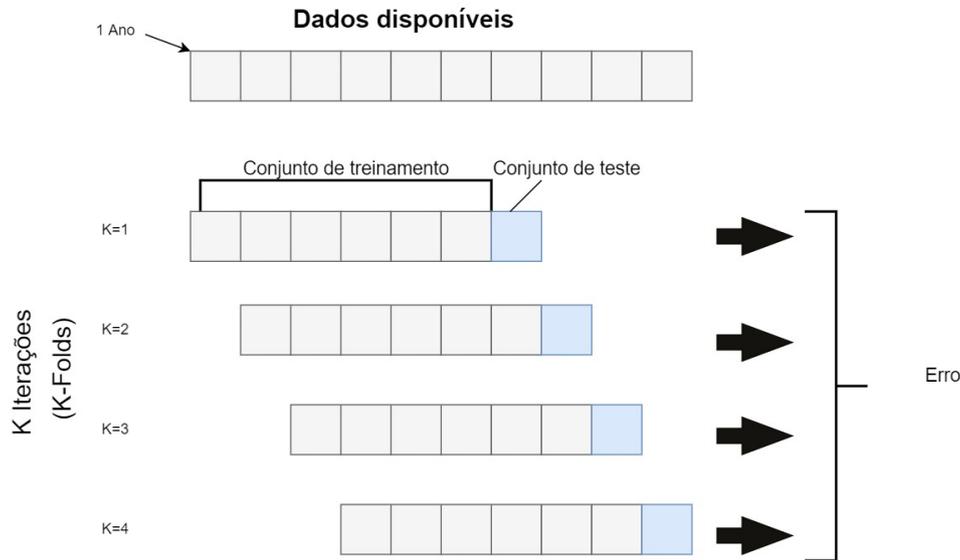


Figura 3.4: K -Fold implementado no trabalho (Fonte: Elaborado pelo autor).

Na avaliação do modelo é usado duas metodologias de k -fold, no primeiro, o modelo é treinado e em seguida no conjunto de teste é feito as previsões usando o valor real de cada dia até o final do conjunto de teste para calcular o erro. No segundo, é feito o k -fold adaptativo onde o mesmo processo será feito, porém após uma previsão ser realizada, em seguida, o modelo é retreinado para introduzir a metodologia de adaptação passiva e fazer a comparação da performance das duas abordagens.

3.5 Avaliação Financeira

A avaliação financeira é a última etapa que ocorre na simulação do sistema. É no módulo *Finance Processor* onde será calculado o potencial de lucro que o sistema proveria. Este módulo contém alguns argumentos que podem ser alterados, como o tamanho da janela do SMA e RSI usado no cálculo dos indicadores, os limites superiores e inferiores do RSI e o mecanismo de “parar perda” (*stop loss*). Porém, os valores padrão usado no trabalho são, respectivamente, de 40 e 14 para os tamanhos de janela, 70 e 30 para os limites e 0.90 para o *stop loss*. Aliás, para calcular o capital total advindo das negociações do ativo, é necessário, primeiramente, saber quando efetivamente realizar a negociação do ativo. Em outras palavras, é preciso sinalizar quando há uma compra, venda ou quando é

preciso se permanecer neutro. Estes sinais são gerados a partir de um algoritmo tomador de decisões que analisa o conjunto de informações provenientes do ativo, dos indicadores e da previsão para conseguir verificar um potencial momento de negociação.

O tomador de decisão é descrito no Algoritmo 1, a estratégia leva em consideração diversas informações, como o preço de fechamento, preço máximo e mínimo atingido no dia. O SMA e RSI calculados a partir dos preços de fechamento, assim como parâmetros da decisão que podem ser alterados, como o valor limite inferior e superior do RSI. O algoritmo primeiramente verifica se é possível comprar ou vender, de acordo com a situação financeira no dia, caso seja possível a compra, é verificado se o RSI está num limite inferior (sobrevenda) e sinaliza a compra, já que o RSI é um indicador antecedente. Caso não esteja, o algoritmo verifica se a previsão é maior que o preço atual e maior que o SMA para incluir a tendência na tomada de decisão. Em seguida, é feita a verificação para o caso da previsão ser menor que o preço máximo do dia, o que indica um poder de força de compra fraco. É também, verificado se o RSI está num limite superior (sobrecompra) para não sinalizar compra precipitadamente, e por fim, caso as verificações sejam falsas, sinaliza a compra. No caso de venda, é necessário que a quantidade do ativo comprado seja maior que zero, o processo de decisão é semelhante ao de compra, porém em magnitudes contrárias.

Algorithm 1 Decision Making Algorithm

```
1: procedure DECISION _MAKING(pricesDataFrame, prediction, sma, rsi, canBuy,
   quantity, lowerLimit, higherLimit)
2:   priceClose  $\leftarrow$  current close price in pricesArray
3:   priceHigh  $\leftarrow$  current high price in pricesArray
4:   priceLow  $\leftarrow$  current low price in pricesArray
5:   if canBuy then
6:     if rsi  $\leq$  lowerLimit then
7:       return “buy”
8:     else if priceClose  $<$  prediction and prediction  $>$  sma then
9:       if prediction  $<$  priceHigh then
10:        return “neutral”
11:       else if rsi  $\geq$  higherLimit then
12:        return “neutral”
13:       else
14:        return “buy”
15:       end if
16:     else
17:       return “neutral”
18:     end if
19:   else if quantity  $>$  0 then
20:     if rsi  $\geq$  higherLimit then
21:       return “sell”
22:     else if priceClose  $>$  prediction and prediction  $<$  sma then
23:       if prediction  $>$  priceLow then
24:        return “neutral”
25:       else if rsi  $\leq$  lowerLimit then
26:        return “neutral”
27:       else
28:        return “sell”
29:       end if
30:     else
31:       return “neutral”
32:     end if
33:   else
34:     return “neutral”
35:   end if
36: end procedure
```

A avaliação financeira é descrita no Algoritmo 2. Resumidamente, é onde se calculará o capital total a partir dos sinais de negociação e onde será executado o mecanismo de *stop loss*. O algoritmo verifica se o último sinal gerado foi uma compra e se é possível comprar, e então calcula a quantidade resultante da compra, o capital disponível para compra que sobrar e o capital total, efetivando a compra. Na venda do ativo, o algoritmo vende toda a quantidade comprada e calcula o capital disponível para compra, e no caso de neutro, o algoritmo apenas atualiza o capital total com o preço do dia. Após a negociação o algoritmo atualiza o valor máximo atingido pelo capital total, como referência para o mecanismo de *stop loss*. E por fim, verifica-se a variação entre o valor máximo e capital total corrente, caso seja menor que um valor definido no módulo, é efetivado a venda do ativo.

Algoritmo 2 Financial Evaluation Algorithm

```

1: procedure FINANCIAL _EVALUATION(priceClose, totalCapital, leftCapital, quantity,
   signalArray, maxValue, stopLoss)
2:   canBuy  $\leftarrow$  leftCapital/priceClose
3:   signal  $\leftarrow$  last signal in/signalhistory
4:   if signal == “buy” and canBuy then
5:     quantity = quantity + leftCapital/priceClose
6:     leftCapital  $\leftarrow$  leftCapital - quantity  $\times$  priceClose
7:     totalCapital  $\leftarrow$  leftCapital + quantity  $\times$  priceClose
8:   else if signal == “sell” and quantity > 0 then
9:     leftCapital  $\leftarrow$  leftCapital + quantity  $\times$  priceClose
10:    quantity  $\leftarrow$  0
11:    totalCapital  $\leftarrow$  leftCapital
12:   else
13:     totalCapital  $\leftarrow$  leftCapital + quantity  $\times$  priceClose
14:   end if
15:   if totalCapital > maxValue then
16:     maxValue  $\leftarrow$  totalCapital
17:   end if
18:   if quantity > 0 and totalCapital/maxValue  $\leq$  stopLoss then
19:     leftCapital  $\leftarrow$  leftCapital + quantity  $\times$  priceClose
20:     quantity  $\leftarrow$  0
21:     totalCapital  $\leftarrow$  leftCapital
22:     maxValue  $\leftarrow$  totalCapital
23:   end if
24: end procedure

```

O capítulo presente focou em explicar a motivação do trabalho a tecnologia e ativos usados, e a ideia geral do sistema proposto, entrando em detalhes como os processos ocorrem dentro do sistema. No capítulo seguinte, o sistema será colocado em prática e os resultados serão expostos.

Capítulo 4

Resultados Obtidos

Neste capítulo são descritos os experimentos realizados. Na Seção 4.1 todo o processo de seleção dos modelos das redes até os resultados obtidos por elas é exposto e detalhado para explicar os motivos das escolhas tomadas. Na Seção 4.2 o modelo escolhido é avaliado usando os parâmetros que forneceram os melhores resultados obtidos da seção anterior. Por fim, na Seção 4.3 é realizada a simulação com os melhores modelos para aferir os ganhos financeiros que o sistema desenvolvido no Capítulo 3 consegue atingir.

4.1 Seleção de modelos de RNA

Nesta seção são realizadas várias rodadas de experimentos de treinamento para mensurar a capacidade dos modelos e seus parâmetros para selecionar os melhores casos. Inicialmente, é utilizado o RMSE como métrica de avaliação estatística para um mesmo ativo. Os resultados serão obtidos por várias sequências de experimentos executados variando, principalmente, o tamanho de janela da entrada da rede, o número de neurônios e épocas. A seguir, é introduzida a metodologia de adaptação e os experimentos são executados novamente, obtendo uma combinação ótima de parâmetros, metodologia e modelo.

4.1.1 Experimentos

Identificação dos Parâmetros

Os primeiros experimentos conduzidos foram uma forma de estabelecer o número de hiperparâmetros do modelo que conseguisse, inicialmente, atingir uma menor acurácia para a coleção de dados de séries temporais. Para isso utilizou-se um modelo de Redes Neurais Artificiais (RNA) simples, baseado no algoritmo MLP. O modelo foi propositalmente, construído com uma estrutura simples, com uma entrada proporcional ao tamanho da janela de variável. Em seguida a camada intermediária variando entre 16, 32, 64, 128,

512 e 1024 neurônios com a função de ativação ReLU. Por fim, a rede possui apenas um único neurônio de saída para a previsão de um passo usando a função de ativação linear. O modelo é treinado com o algoritmo de otimização Adam.

Estes experimentos foram realizados usando os preços diários de fechamento da criptomoeda Bitcoin entre 2014 e 2021. Esta série temporal possui uma alta volatilidade esporádica. Em relação ao treinamento, primeiramente, foi feita a separação dos dados, com os primeiros 80% para o treinamento e 20% para testes. Em seguida é feita uma série de experimentos variando o número de neurônios, o número de épocas e o tamanho da janela que caminhará de um e um dia e representará o número de entradas da rede. Assim, usando o modelo treinado as previsões dos preços do Bitcoin (em dólar) são feitas com os dados de testes e calculado o RMSE. O experimento com melhor resultado é apresentado na Tabela 4.1.

Tabela 4.1: RMSE para janela de tamanho 1.

Épocas/Neurônios	16	32	64	128	512	1024
10	1415	1415	1426	1425	1448	1460
20	1419	1424	1425	1432	1441	1475
30	1422	1426	1429	1435	1463	1481
40	1419	1430	1431	1448	1477	1513
50	1418	1436	1434	1443	1500	1509
100	1425	1423	1453	1463	1475	1489
1000	1425	1429	1430	1452	1448	1471
5000	1431	1432	1445	1436	1427	1439
10000	1423	1425	1433	1442	1445	1426

O conjunto de testes dos dados de séries temporais do Bitcoin, possui como menor valor USD 6.880 e como maior valor USD 63.314, considerando este fato, os resultados do RMSE não foram particularmente surpreendentes. Nota-se que os hiperparâmetros da rede não apresentam uma relação linear, isto é, apenas aumentar o número de neurônios, épocas ou tamanho da janela não necessariamente resulta em uma melhor estimativa. Como exemplo, temos a combinação de uma janela de tamanho de 1 dia na Tabela 4.1, com 10 épocas e 16 neurônios possui o menor RMSE de 1415, resultado este que é muito próximo de 1400 é o menor RMSE que este modelo conseguiu atingir usando 10 épocas com 16 e 32 neurônios.

4.1.2 Experimentos Adaptativos

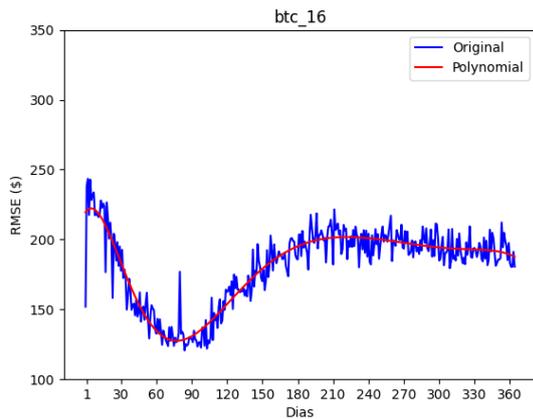
Como ajustar os hiperparâmetros da rede não se mostrou eficaz, foi necessário buscar alternativas que não se limitassem a estes fatores. A alternativa encontrada foi a mudança do método de treinamento aplicado. Considerando a característica particular das séries

temporais do mercado financeiro de evoluírem ao longo do tempo e dada a necessidade do sistema de se moldar em tempo real. Foi aplicada uma metodologia de treinamento adaptativo, que como o nome sugere, adapta a rede conforme novos dados vão sendo gerados, coletados e introduzidos no modelo [13]. Considerando os experimentos passados, nos treinamentos adaptativos realizados foi aplicado o mecanismo de Parada Antecipada (do Inglês, *Early Stopping*) (ES) com uma paciência de 10 amostras, visando melhorar a eficiência do treinamento. O número de épocas do treinamento foi mantido em 100, pois inicialmente observa-se uma queda no erro com 100 épocas, mas em seguida o aprendizado rapidamente se estabiliza após 10 épocas. Tanto as funções de ativação quanto o algoritmo de otimização se mantiveram como nos experimentos anteriores.

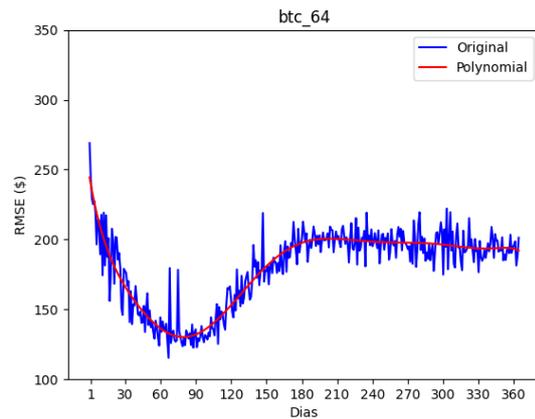
Experimentos Adaptativos com MLP

Neste experimento, o RMSE é calculado pela Equação 2.4, o y_i é o valor real, que no caso, é o valor esperado usado no treinamento supervisionado e \hat{y}_i é o valor futuro advindo da saída da rede. Isto posto, o objetivo do experimento é mensurar o potencial de generalização que a rede pode vir a atingir e com estes resultados produzir uma visualização gráfica para comparar com os resultados dos experimentos anteriores. O gráfico gerado é analisado para determinar o tamanho de janela ótimo, e para isso, é necessário realizar um ajuste de curvas, estabelecendo uma função polinomial [44] que melhor represente o gráfico resultante. Assim, é calculado o mínimo global determinando o tamanho de janela ótimo.

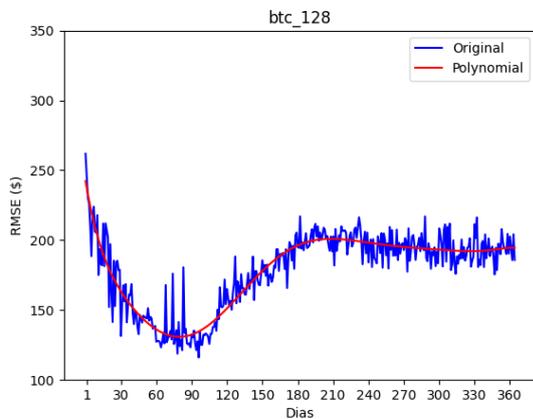
Os gráficos gerados a partir dos resultados dos treinamentos adaptativos do Bitcoin (BTC) são mostrados na Figura 4.1.



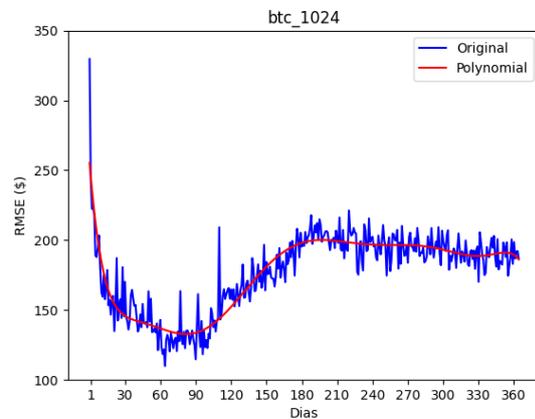
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



(d) 1024 neurônios

Figura 4.1: Gráficos do RMSE \times tamanho da janela do BTC (MLP).

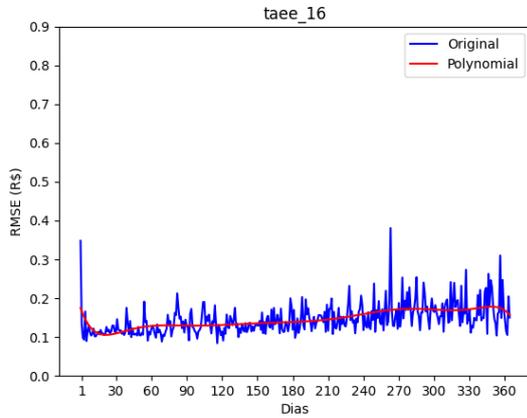
Nestes gráficos, o eixo y representa o RMSE do treinamento em dólares e o eixo x o tamanho da janela em dias. Em azul são todos os valores obtidos da rede e em vermelho são os valores resultantes do cálculo a partir da função polinomial obtida pelo ajuste de curvas. Nota-se que mesmo variando o número de neurônios da rede, a saída das redes têm um comportamento similar. Apresentam uma queda inicial e em seguida um aumento e estabilização nos erros. Conforme o tamanho da janela aumenta, com os menores valores se consolidando na mesma região entre 50 e 100. Ao calcular a derivada para se descobrir o mínimo global para os neurônios 16, 64, 128 e 1024, de acordo com a Tabela 4.2, obtém-se 78, 80, 80, e 82 como tamanho ótimo de janela. O melhor resultado é de RMSE, 16 neurônios com USD 127,56 de erro.

Na Figura 4.2 é mostrado os resultados dos treinamentos adaptativos para o caso do ativo da Transmissora Aliança de Energia Elétrica S.A. (TAEE). O eixo y representa o resultado do RMSE em reais do treinamento e o eixo x o tamanho da janela em dias. Diferentemente do Bitcoin (BTC), é possível notar uma relação direta entre o aumento

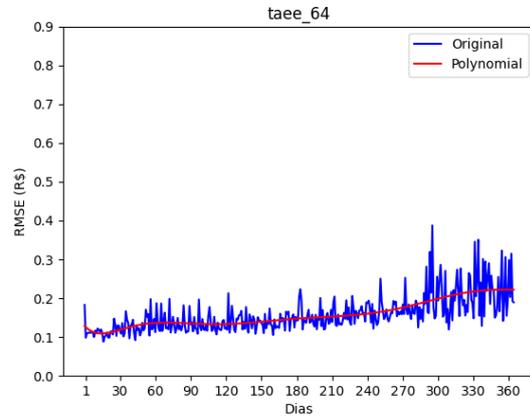
Tabela 4.2: Tamanho ótimo de janela do Bitcoin (BTC).

Neurônios	Janela	RMSE
16	78	127,56
64	80	130,21
128	80	130,82
1024	82	132,78

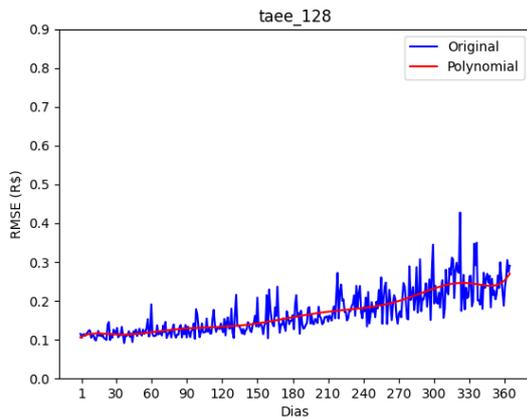
do tamanho da janela com o aumento dos valores de RMSE. Ao tentar determinar o mínimo global para os neurônios 16, 64, 128, 1024, pode ser verificado na Tabela 4.3, que obtém-se 22, 15, 1 e 13 para o tamanho de janela ótimo e com o menor valor de RMSE de R\$ 0,096 com 1024 neurônios.



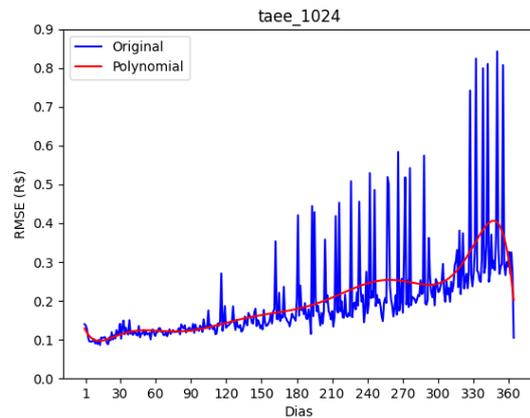
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



(d) 1024 neurônios

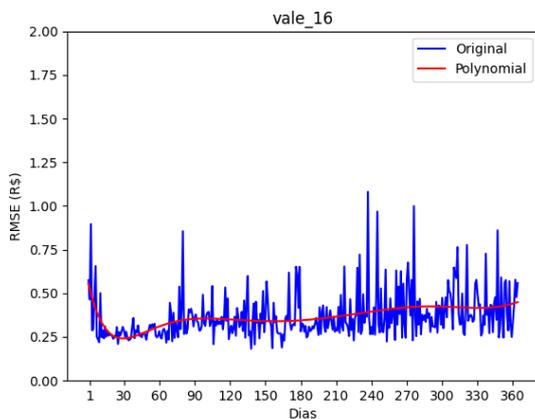
Figura 4.2: Gráficos do RMSE \times tamanho da janela da TAE (MLP).

A Figura 4.3 mostram os resultados dos treinamentos adaptativos para o caso do ativo da VALE, com o eixo y representando o resultado do RMSE em reais do treinamento e o eixo x o tamanho da janela em dias. Neste caso, a curva da função polinomial permanece

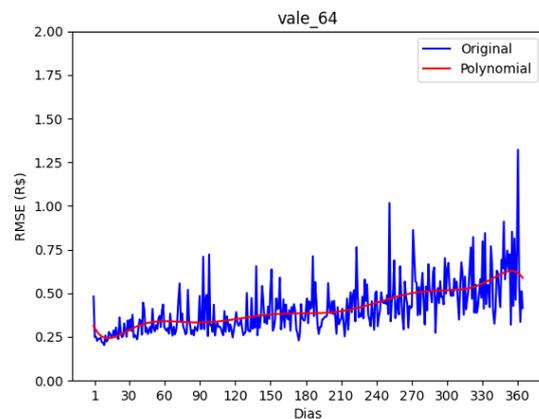
Tabela 4.3: Tamanho ótimo de janela da Transmissora Aliança de Energia Elétrica S.A. (TAEE).

Neurônios	Janela	RMSE
16	22	0,105
64	15	0,109
128	1	0,107
1024	13	0,096

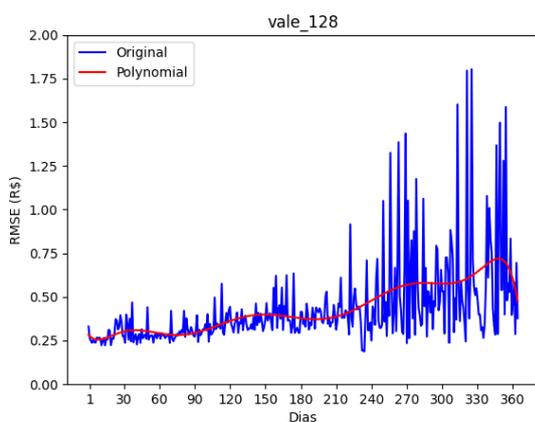
constante com menos neurônios e aumentando a variação do RMSE com o aumento de neurônios e tamanho da janela. O mínimo global também varia de acordo, que a partir do cálculo das derivadas determina-se o número de dias de 30, 13, 9 e 1 para o tamanho ótimo da janela com o menor erro sendo o de 16 neurônios com R\$ 0,241 como mostrado na Tabela 4.4.



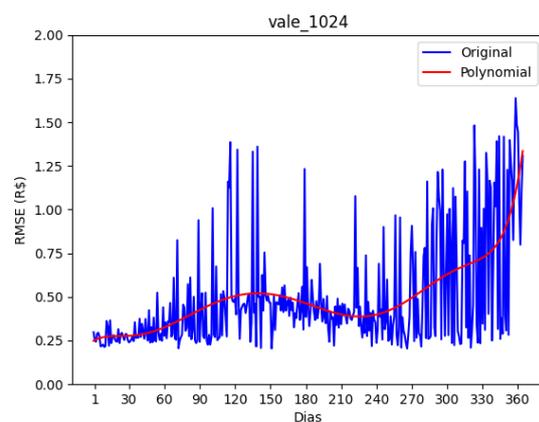
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



(d) 1024 neurônios

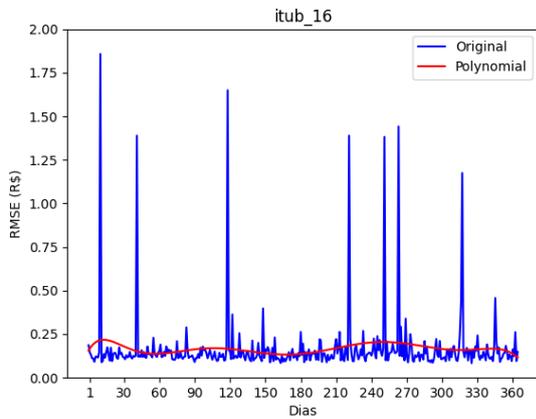
Figura 4.3: Gráficos do RMSE \times tamanho da janela da VALE (MLP).

A Figura 4.4 mostram os resultados dos treinamentos adaptativos para o caso do ativo

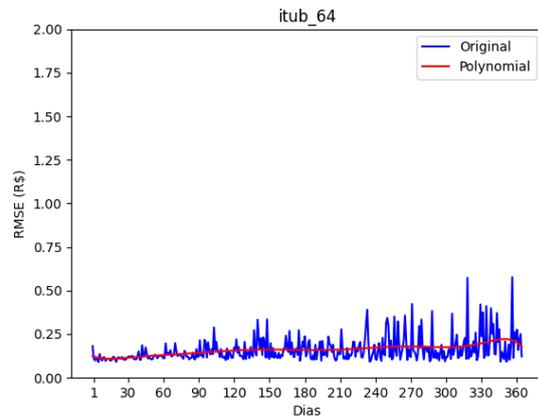
Tabela 4.4: Tamanho ótimo de janela da Vale S.A. (VALE).

Neurônios	Janela	RMSE
16	30	0,241
64	13	0,243
128	9	0,255
1024	1	0,252

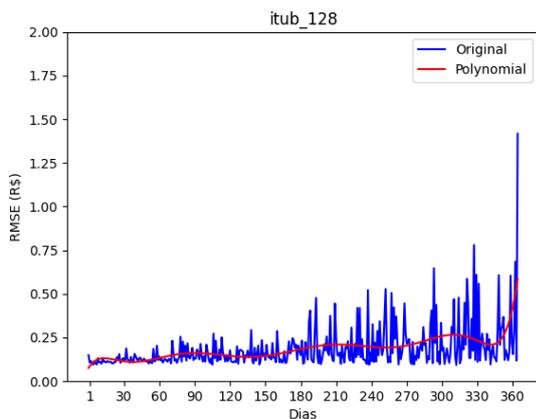
da ITUB, com o eixo y representando o resultado do RMSE em reais do treinamento e o eixo x o tamanho da janela em dias. Os resultados para ITUB com 16 neurônios, explicita erros esporádicos com valores muito altos, o que pode ter prejudicado o ajuste de curvas. Para o mínimo global, determina-se o número de dias de 58, 15, 1 e 1 para o tamanho ótimo da janela com o menor erro sendo o de 128 neurônios com R\$ 0,086 como mostrado na Tabela 4.4.



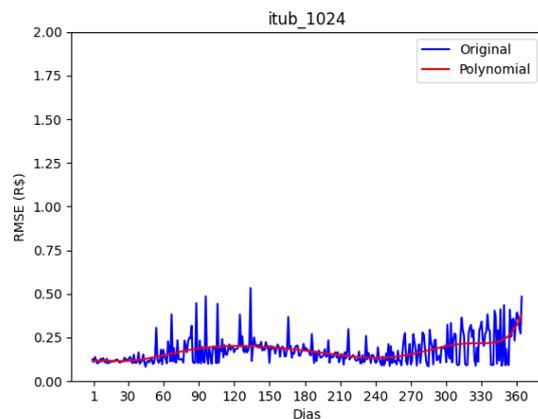
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



(d) 1024 neurônios

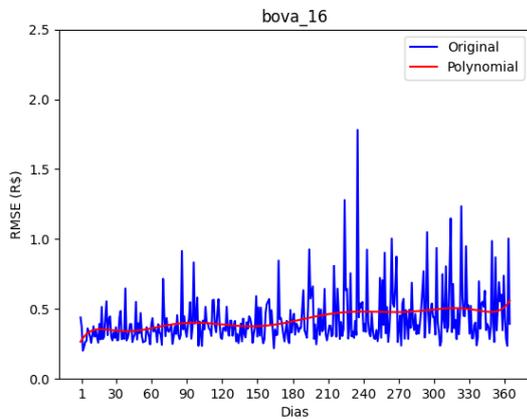
Figura 4.4: Gráficos do RMSE \times tamanho do janela do ITUB (MLP).

Os gráficos gerados a partir dos resultados dos treinamentos adaptativos da Bovespa

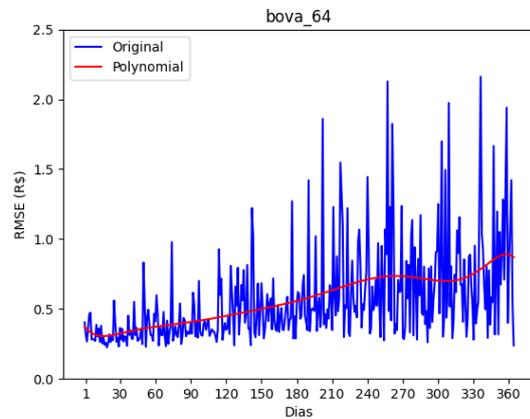
Tabela 4.5: Tamanho ótimo de janela do Itaú Unibanco (ITUB).

Neurônios	Janela	RMSE
16	58	0,136
64	15	0,107
128	1	0,086
1024	1	0,116

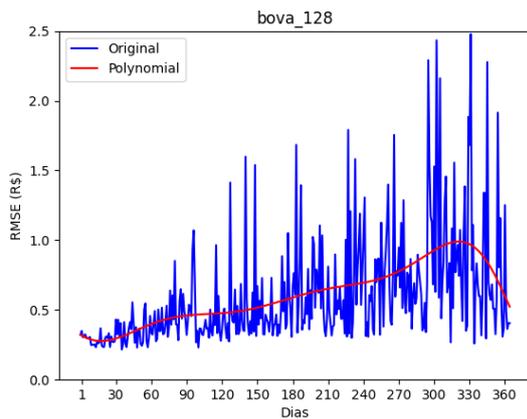
(BOVA) são mostrados na Figura 4.5. Nota-se que o comportamento dos resultados do RMSE em 1024 possui uma intensa variação dos erros quanto maior o tamanho da janela. Os tamanhos de janelas para 16, 64, 128 e 1024 neurônios foram respectivamente, 1, 17, 18 e 1 com o menor erro de R\$ 0,276 para 128 neurônios.



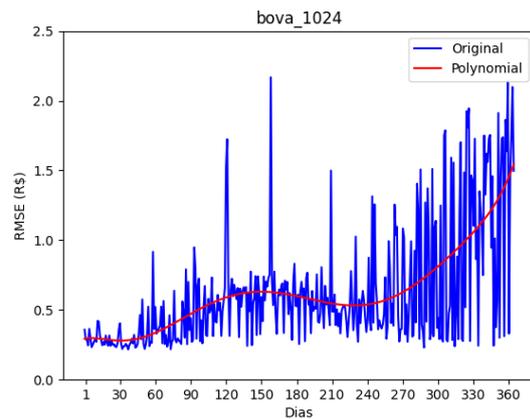
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



(d) 1024 neurônios

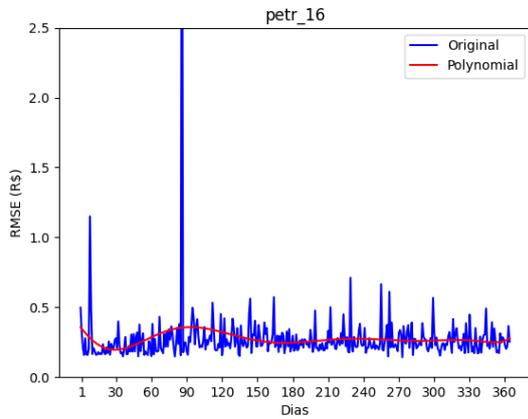
Figura 4.5: Gráficos do RMSE \times tamanho da janela da BOVA (MLP).

Os resultados da PETR se mostram semelhantes ao da BOVA e da VALE, tendo intensa variação do valor do erro no experimento de 1024 neurônios. Os tamanhos de

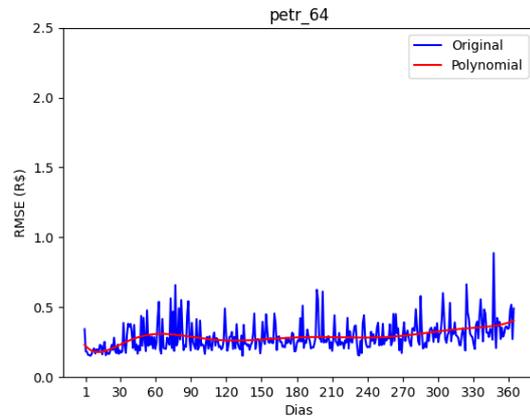
Tabela 4.6: Tamanho ótimo de janela da Bovespa (BOVA).

Neurônios	Janela	RMSE
16	1	0,279
64	17	0,306
128	18	0,276
1024	1	0,294

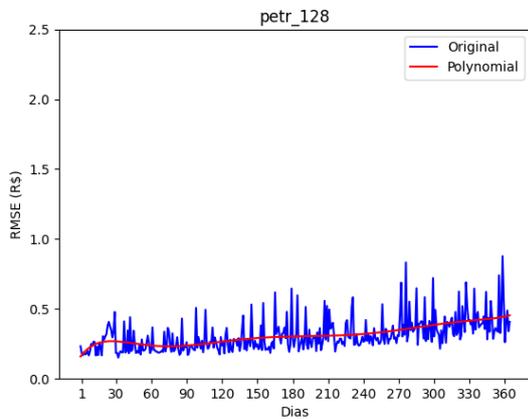
janelas para 16, 64, 128 e 1024 neurônios foram respectivamente, 31, 12, 1 e 14 com o menor erro de R\$ 0,169 para 128 neurônios.



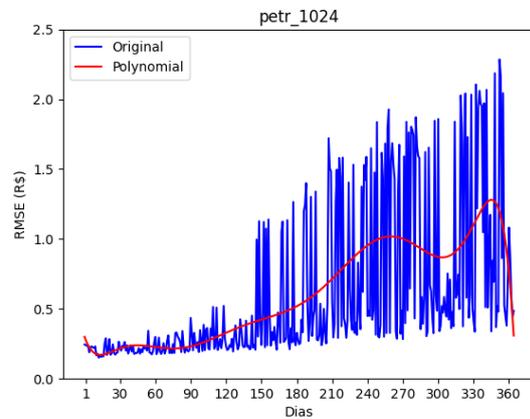
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



(d) 1024 neurônios

Figura 4.6: Gráficos do RMSE \times tamanho da janela do PETR (MLP).

Comparando os gráficos da Figura 4.1 e a Tabela 4.2 com os primeiros experimentos. É possível perceber que há uma melhora significativa nos resultados do RMSE, mostrando que esta metodologia de treinamento adaptativo tem o potencial de se tornar um candidato para ser o modelo escolhido. Ademais, uma característica importante do mercado financeiro é as situações e comportamentos diversos que cada ativo apresenta, que de certa

Tabela 4.7: Tamanho ótimo de janela da Petrobrás (PETR).

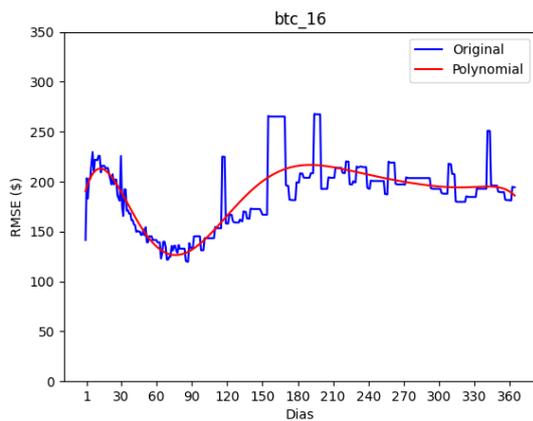
Neurônios	Janela	RMSE
16	31	0,211
64	12	0,179
128	1	0,169
1024	14	0,171

forma, pode ser conferido nos experimentos realizados. Alguns ativos apresentaram comportamentos bastante dissimilares como o BTC, e ativos como a VALE, BOVA e PETR apresentando semelhanças, tais como, o menor erro resultante de um tamanho de janela baixo com 128 neurônios. Entretanto, outros ativos resultaram em menor erro e comportamento de curvas diferentes entre si advindos de diferentes parâmetros. Sugerindo que não há uma forma universal de se construir um modelo de RNA que possa atingir resultados ótimos, para o cenário do mercado financeiro.

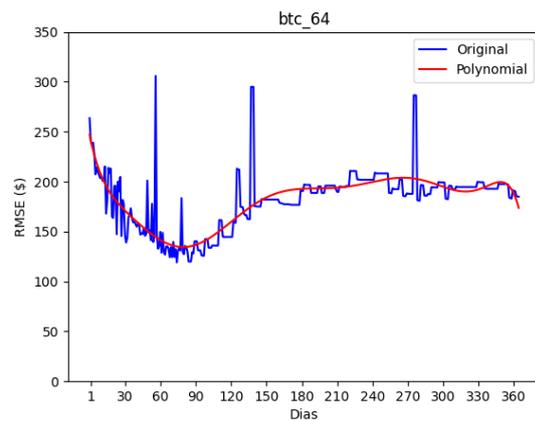
Experimentos Adaptativos com LSTM

A seguir é mostrado os resultados dos experimentos adaptativos usando arquitetura LSTM. Os experimentos foram realizados como no experimento com MLP, mantendo os mesmos hiperparâmetros, algoritmo de otimização, função de ativação, metodologia de treinamento e objetivo do experimento de encontrar a janela ótima. Neste experimento, os treinamentos foram realizados com o tamanho da janela variando entre 1 à 90 incrementando a janela em 1 a cada iteração e de 90 até 366. Os experimentos foram realizados incrementando a janela em 3 a cada iteração. Já que as janelas ótimas se mantiveram com tamanho menor que 90 no experimento anterior, e também, por questão de eficiência de recursos.

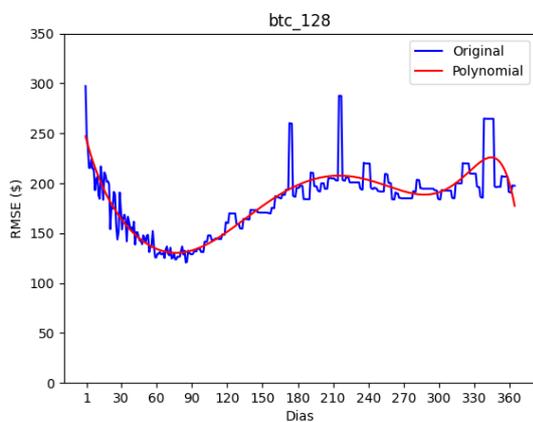
Os gráficos gerados a partir dos resultados dos treinamentos adaptativos do Bitcoin (BTC) são mostrados na Figura 4.7, com o eixo y representando o resultado do RMSE do treinamento em dólares e o eixo x o tamanho da janela em dias. Os experimentos com LSTM resultaram em picos de RMSE, possivelmente pelo problema do gradiente explosivo. Especialmente para o BTC, que teve alguns picos muito altos e tiveram que ser filtrados para o ajuste de curvas. Entretanto, o comportamento foi similar ao do experimento com MLP, com menores valores se consolidando na mesma região entre 50 e 100. Ao calcular a derivada para se descobrir o mínimo global para os neurônios 16, 64, 128 e 1024, que de acordo com a Tabela 4.8, obtém-se 77, 78, 77, e 84 como tamanho ótimo de janela. O melhor resultado foi de RMSE, 16 neurônios com USD 126,43 de erro.



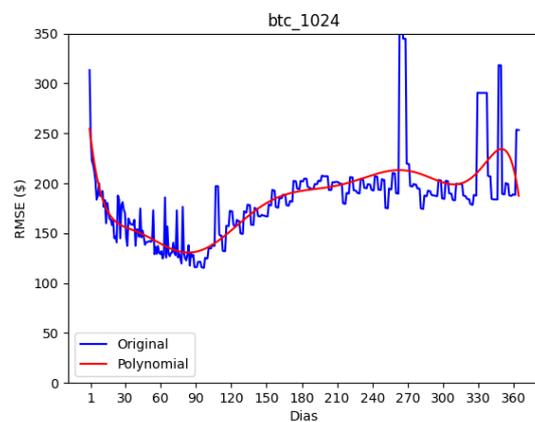
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



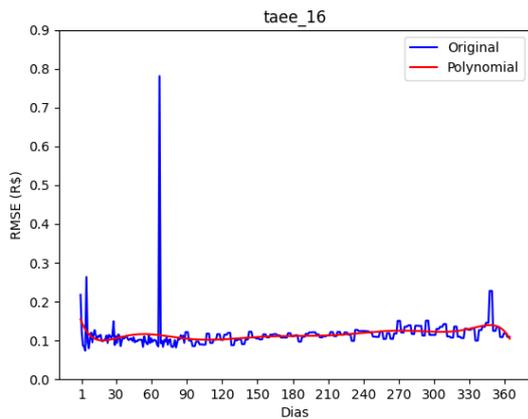
(d) 1024 neurônios

Figura 4.7: Gráficos do RMSE \times tamanho da janela do BTC (LSTM).

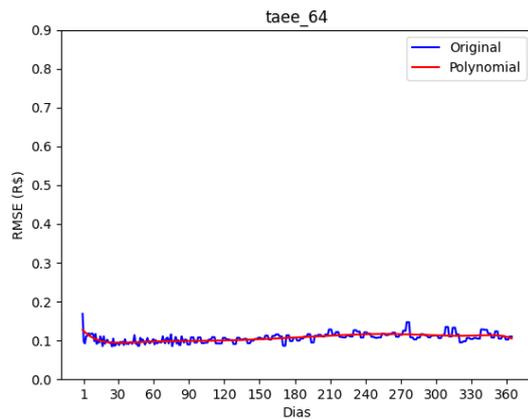
Tabela 4.8: Tamanho ótimo de janela do Bitcoin (BTC).

Neurônios	Janela	RMSE
16	77	126,43
64	78	134,44
128	77	130,28
1024	84	130,59

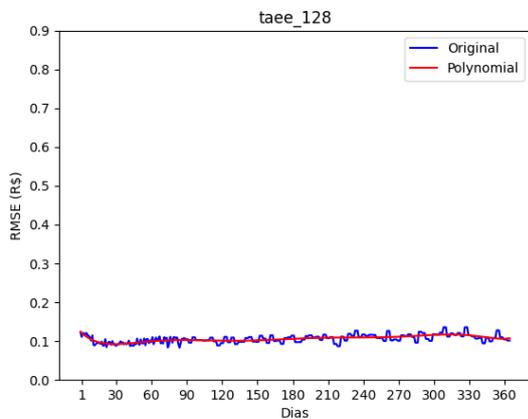
Na Figura 4.8 é mostrado os resultados do experimento com LSTM para o caso do ativo da Transmissora Aliança de Energia Elétrica S.A. (TAEE), com o eixo y representando o resultado do RMSE em reais do treinamento e o eixo x o tamanho da janela em dias. Os resultados para este ativo se mostraram bastante estáveis com exceção do caso de 1024 neurônios que teve um aumento brusco com o aumento do tamanho da janela. Entretanto, foi o que teve o menor resultado com R\$ 0,089 para 37 dias de tamanho de janela.



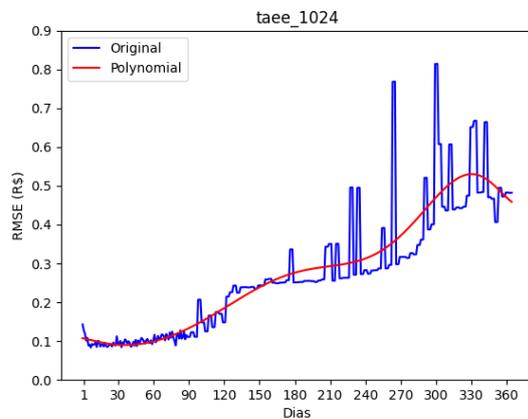
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



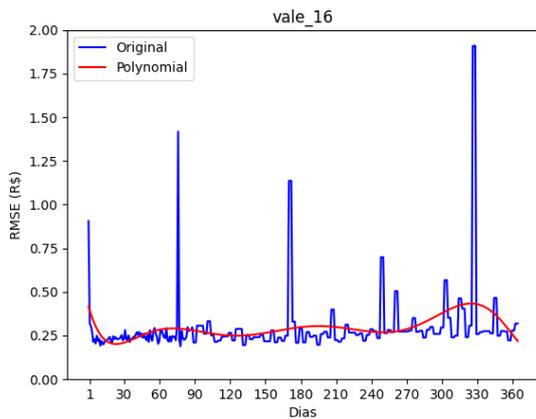
(d) 1024 neurônios

Figura 4.8: Gráficos do RMSE \times tamanho da janela da TAEE (LSTM).

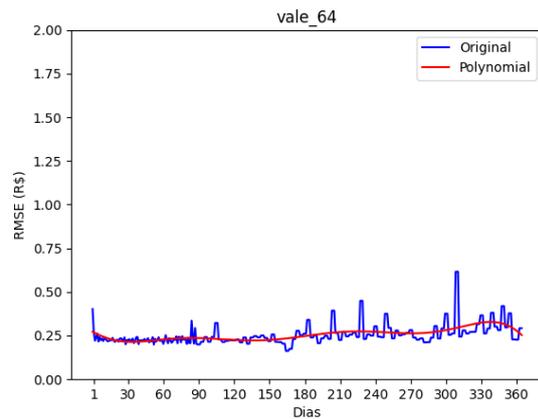
Tabela 4.9: Tamanho ótimo de janela da Transmissora Aliança de Energia Elétrica S.A. (TAEE).

Neurônios	Janela	RMSE
16	19	0,100
64	31	0,094
128	30	0,092
1024	37	0,089

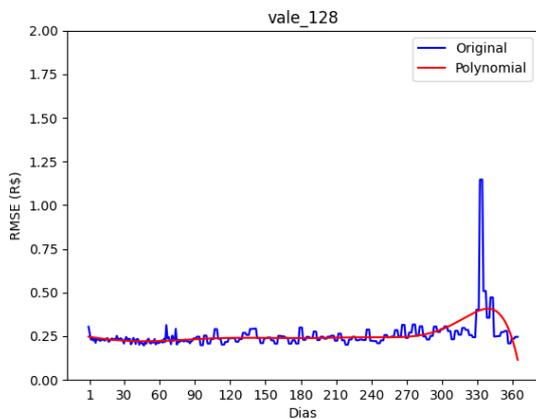
A Figura 4.9 mostram os resultados dos experimentos com LSTM para o caso do ativo da VALE. Os resultados tiveram picos com 16 neurônios, se mantiveram constantes para 64 e 128 neurônios e o RMSE teve aumento significativo a partir de 60 dias para 1024 neurônios. O menor erro R\$ de 0,202 ocorreu com 16 neurônios em 24 dias para o tamanho de janela, como mostrado na Tabela 4.10.



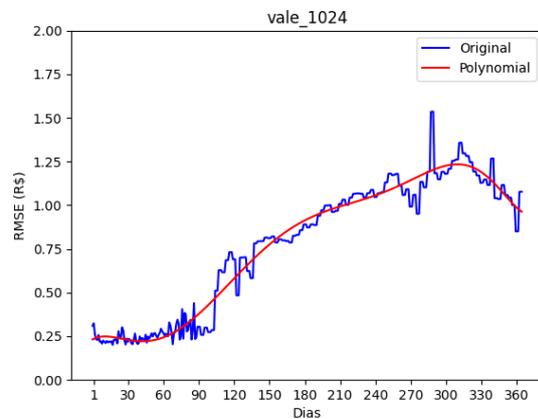
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



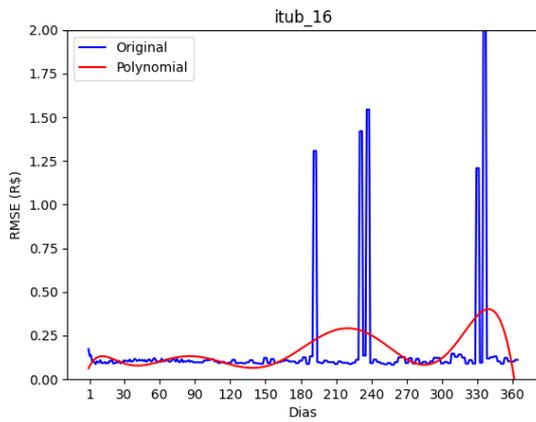
(d) 1024 neurônios

Figura 4.9: Gráficos do RMSE \times tamanho da janela da VALE (LSTM).

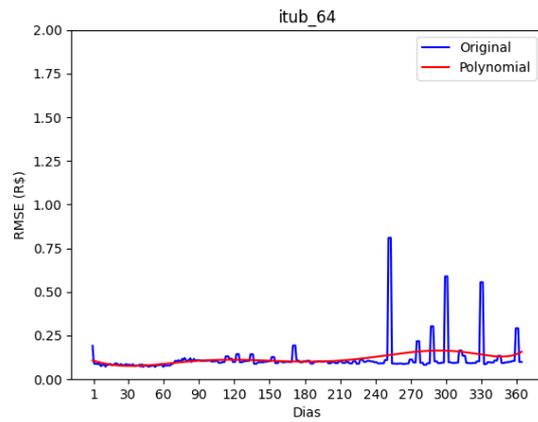
Tabela 4.10: Tamanho ótimo de janela da Vale S.A. (VALE).

Neurônios	Janela	RMSE
16	24	0,202
64	34	0,214
128	49	0,221
1024	1	0,235

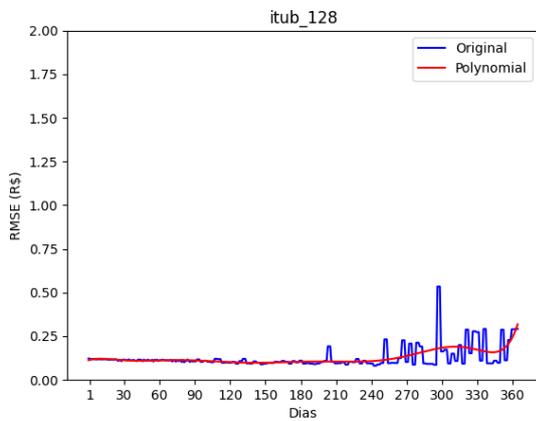
Os experimentos com LSTM para o ativo ITUB mostrados na Figura 4.10, revelam picos para 16 neurônios, e também, ocorrem picos em 64 e 128 neurônios com o aumento do tamanho da janela. E para 1024 neurônios, o RMSE teve aumento significativo com tamanho de janela acima de 120, entretanto apresentou menor resultado com R\$ 0,069 com janela de tamanho de 1 dia.



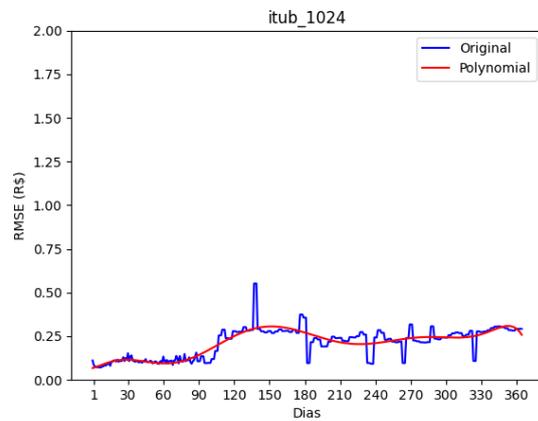
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



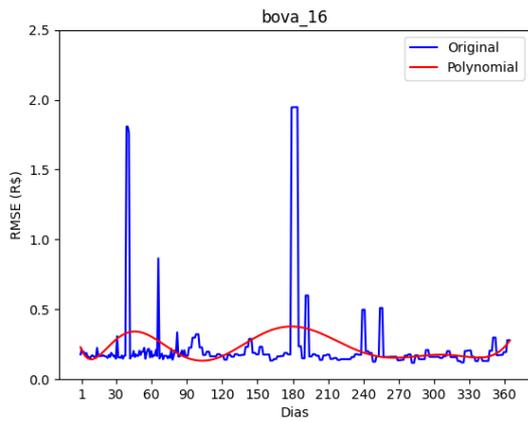
(d) 1024 neurônios

Figura 4.10: Gráficos do RMSE \times tamanho da janela do ITUB (LSTM).

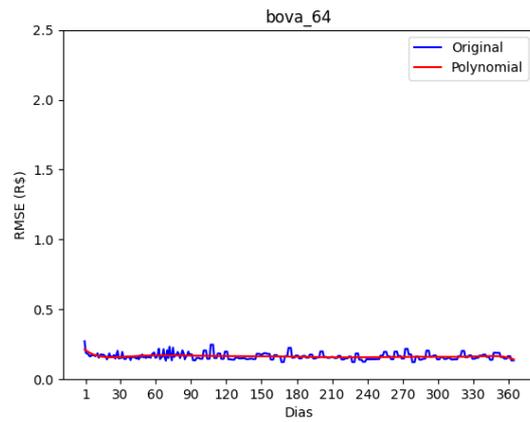
Tabela 4.11: Tamanho ótimo de janela do Itaú Unibanco (ITUB).

Neurônios	Janela	RMSE
16	1	0,076
64	33	0,075
128	142	0,098
1024	7	0,071

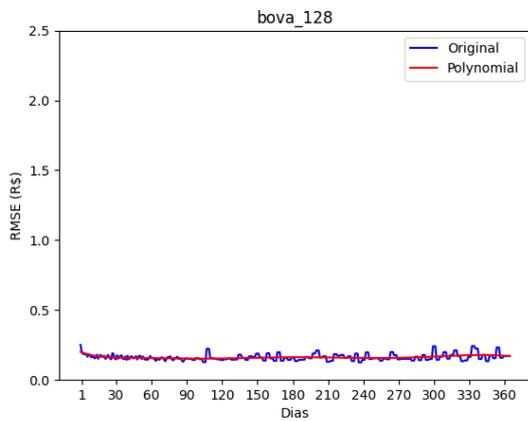
Os gráficos gerados a partir dos resultados dos treinamentos adaptativos da Bovespa (BOVA) são mostrados na Figura 4.11. Nota-se que o comportamento dos resultados do RMSE em 1024 possui uma intensa variação dos erros a partir dos 60 dias para o tamanho da janela. Os tamanhos de janelas para 16, 64, 128 e 1024 neurônios foram respectivamente, 28, 37, 42 e 19 com o menor erro de R\$ 0,229 para 16 neurônios como mostrado na Tabela 4.13.



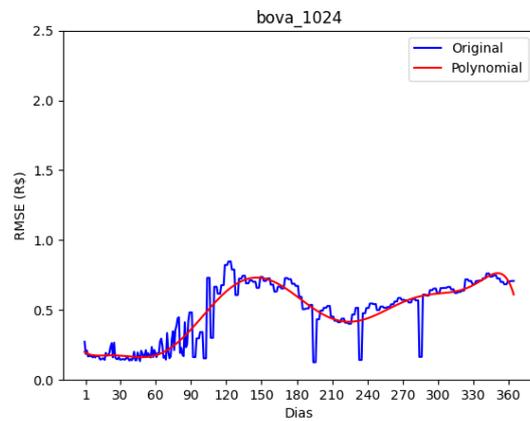
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



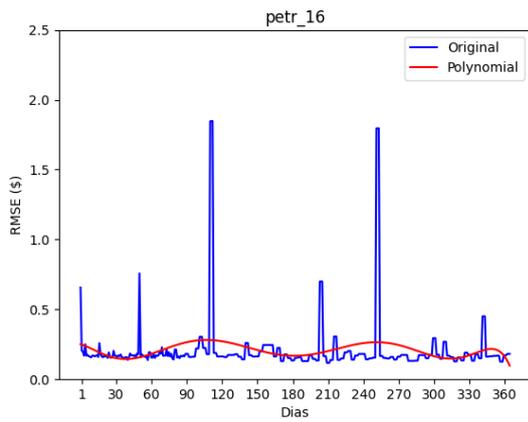
(d) 1024 neurônios

Figura 4.11: Gráficos do RMSE \times tamanho da janela da BOVA (LSTM).

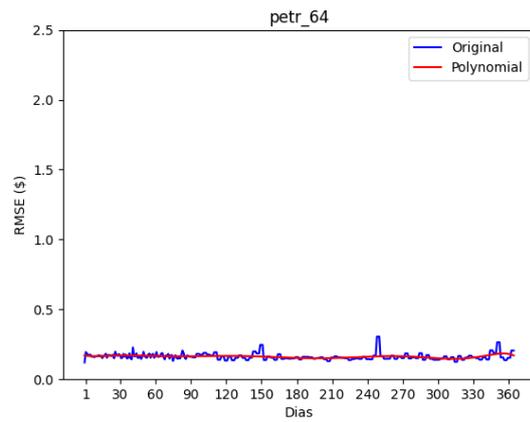
Tabela 4.12: Tamanho ótimo de janela da Bovespa (BOVA).

Neurônios	Janela	RMSE
16	28	0,239
64	37	0,229
128	42	0,231
1024	19	0,244

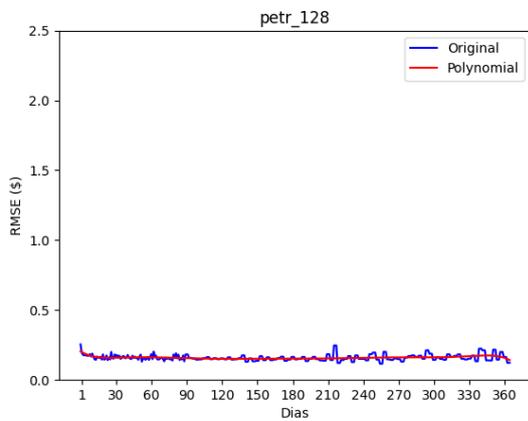
Os resultados da PETR se mostram semelhantes ao da BOVA, tendo intensa variação do valor do erro no experimento de 1024 neurônios a partir de 60 dias. Os tamanhos de janelas para 16, 64, 128 e 1024 neurônios foram respectivamente, 104, 219, 101 e 12 com o menor erro de R\$ 0,132 para 16 neurônios.



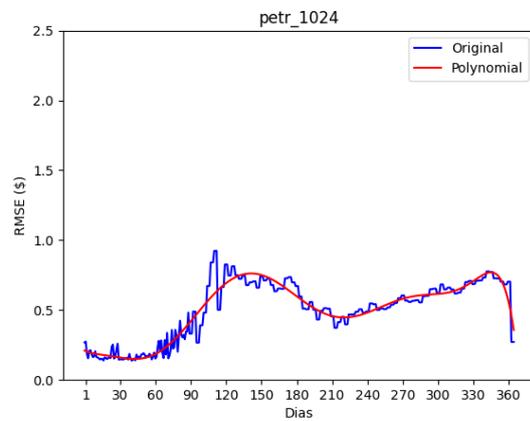
(a) 16 neurônios



(b) 64 neurônios



(c) 128 neurônios



(d) 1024 neurônios

Figura 4.12: Gráficos do RMSE \times tamanho da janela do PETR (LSTM).

Tabela 4.13: Tamanho ótimo de janela da Petrobrás (PETR).

Neurônios	Janela	RMSE
16	39	0,145
64	82	0,164
128	128	0,150
1024	43	0,149

Os experimentos usando o modelo de LSTM tiveram no geral, picos extremamente altos para o BTC, possivelmente pelo problema do gradiente explosivo exacerbado pela alta volatilidade esporádica que o ativo apresenta. Ademais, os outros ativos, também tiveram picos para 16 neurônios, mas alguns dos melhores resultados, mostrando a instabilidade no uso de 16 neurônios.

Os melhores resultados para os experimentos de MLP e LSTM são resumidos na Tabela 4.14. Com relação ao ativo do BTC teve melhoras ínfimas no uso da LSTM. Esta

pequena diferença pode indicar que é a metodologia de treinamento e a forma de se lidar com as características não-estacionárias da série temporal que precisa ser mais bem adaptada para conseguir tratar a volatilidade do ativo. Já, para os outros ativos, o RMSE teve uma melhora mais relevante, chegando a ter uma redução de mais de 20% no erro, apenas trocando o modelo de RNA usado nos experimentos.

Tabela 4.14: Tabela resumo para os melhores resultados.

Ativo	MLP			LSTM		
	Neurônios	Janela	RMSE	Neurônios	Janela	RMSE
BTC	16	78	127,56	16	77	126,43
TAE	1024	13	0,096	1024	37	0,089
VALE	16	30	0,241	16	24	0,202
ITUB	128	1	0,086	1024	7	0,071
BOVA	128	18	0,276	16	37	0,229
PETR	128	1	0,169	16	39	0,145

4.2 Avaliação do Modelo

A avaliação do modelo foi feita baseado nos resíduos estatísticos resultantes dos experimentos anteriores, o *k-fold* foi implementado para os melhores resultados dos experimentos, mantendo os mesmos parâmetros e usando $k = 4$, ou 4 dobras. Considerando a característica dos dados, foi feita uma separação em blocos equivalente à um ano de amostras, pois características de sazonalidade ocorrem com frequência nos ativos do mercado financeiro. Os parâmetros escolhidos para a realização da avaliação do modelo foram aqueles que obtiveram melhor performance nos experimentos. A métrica escolhida para a avaliação foi, novamente, o RMSE. Entretanto, como o RMSE retorna o erro na mesma unidade e mesma magnitude do dado original acaba impossibilitando a comparação com os outros ativos. Isto posto, foi adicionado uma outra métrica, o Erro Médio Absoluto Percentual (do Inglês, *Mean Absolute Percentage Error*) (MAPE), definido pela Equação 2.5, para avaliar todos os ativos em porcentagem, e ter uma melhor noção do erro.

A Tabela 4.15 apresenta os resultados do MAPE para as redes MLP e LSTM utilizando treinamento não-adaptativo.

Tabela 4.15: Resultados k-fold com treinamento não-adaptativo.

Ativo	MLP				LSTM			
	Neurônios	Janela	RMSE	MAPE	Neurônios	Janela	RMSE	MAPE
BTC	16	78	12.643	33,18%	16	77	14.434	53,79%
TAEF	1024	13	1,046	2,88%	1024	37	1,748	4,67%
VALE	16	30	4,231	12,17%	16	24	3,191	8,75%
ITUB	128	1	0,889	2,92%	1024	7	0,848	2,78%
BOVA	128	18	2,563	3,29%	16	37	3,769	5,07%
PETR	128	1	2,284	11,72%	16	39	2,886	16,82%

Através dos resultados para o *k-fold* com treinamento não-adaptativo apresentados na Tabela 4.15. É possível notar um erro muito alto, principalmente para ativos que apresentaram alta volatilidade ou comportamento dissimilares no período usado para o conjunto de testes, como o BTC e a PETR. Além de que 4 dos 6 ativos obtiveram um erro maior para a LSTM do que para a MLP.

Os resultados para o *k-fold* usando a metodologia adaptativa são apresentados na Tabela 4.16

Tabela 4.16: Resultados k-fold com treinamento adaptativo.

Ativo	MLP				LSTM			
	Neurônios	Janela	RMSE	MAPE	Neurônios	Janela	RMSE	MAPE
BTC	16	78	965,43	3,29%	16	77	949,89	2,92%
TAEF	1024	13	0,452	1,14%	1024	37	0,450	1,13%
VALE	16	30	0,871	2,56%	16	24	0,877	2,58%
ITUB	128	1	0,739	2,38%	1024	7	0,504	1,62%
BOVA	128	18	0,941	1,17%	16	37	0,958	1,20%
PETR	128	1	0,715	3,90%	16	39	0,507	2,67%

De acordo com a Tabela 4.16, foram obtidos resultados significativamente melhores, com 4 dos 6 ativos tendo menor erro para a LSTM e os outros dois ativos tiveram resultados semelhantes para a MLP. Ademais, é possível observar uma grande evolução dos ativos, excepcionalmente, para o BTC e PETR que tiveram o MAPE de 53,79% e 16,82% reduzidos para 2,92% e 2,67%, respectivamente, realçando a eficácia da metodologia adaptativa.

4.3 Simulação do Sistema

A simulação do sistema é feita a partir dos resultados da avaliação do modelo, onde foi possível esclarecer a superioridade da abordagem adaptativa e, portanto, foi a única abordagem implementada na simulação. Para realizar a avaliação financeira, a estratégia de negociação descrita no Algoritmo 1 é comparada com uma estratégia simples, popular e eficaz, conhecida como *Buy and Hold*. Nessa estratégia o investidor compra o ativo, em qualquer que seja o preço, e o mantém visando um investimento a longo prazo. Aplicando a metodologia adaptativa para o MLP e LSTM no algoritmo tomador de decisão, é possível verificar se a estratégia do sistema consegue superar a estratégia de *Buy and Hold*. Em seguida será mostrado os gráficos resultantes usando 10.000 unidades da moeda corrente como capital para o início da simulação. A simulação é realizada para cada um dos ativos em relação às 3 estratégias aplicadas ao conjunto de dados de simulação, que compreende os últimos 4 anos das séries temporais representados no eixo x e o capital total na moeda corrente representada no eixo y do gráfico.

O gráfico do resultado das simulações para o BTC é ilustrado na Figura 4.13, onde o eixo y representa o capital total em dólares e o eixo x compreende o período entre 01/01/2018 até 14/09/2021. É possível observar que as estratégias implementadas no trabalho obtiveram resultados excepcionais, a grande diferença entre as estratégias pode ser explicada pela queda que ocorreu no início do período. O sistema atingiu um capital final de USD 105.335,72 para MLP e USD 119.776,19 para LSTM em comparação com a estratégia *Buy and Hold*, que resultou em USD 34.381,80. Nota-se que mesmo usando a mesma estratégia, a aplicação do modelo LSTM resultou em lucro superior que o modelo com MLP.

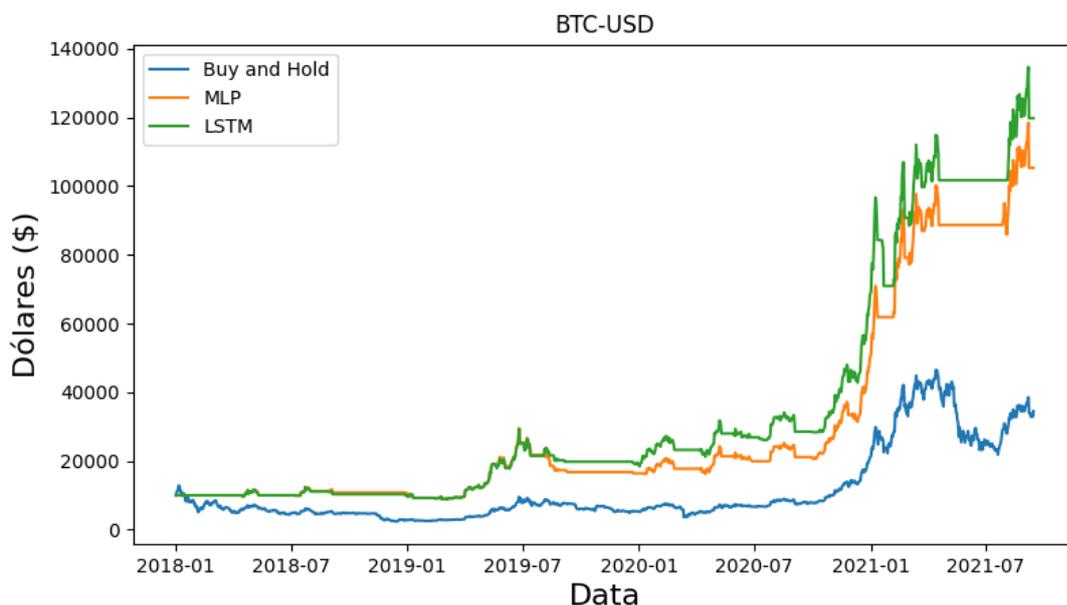


Figura 4.13: Resultados financeiros da simulação da BTC.

O gráfico do resultado das simulações para a TAEE é ilustrado na Figura 4.14, onde o eixo y representa o capital total em reais e o eixo x compreende o período entre 02/01/2018 até 29/12/2021. É possível observar no gráfico que o sistema conseguiu se permanecer neutro de forma preemptiva, o que é uma característica desejável em negociações no mercado financeiro. A estratégia com MLP obteve resultado de R\$ 15.252,36, abaixo da estratégia *Buy and Hold*, que resultou em R\$ 16.877,91, entretanto obteve-se R\$ 17.434,40 com a LSTM, cumprindo o objetivo de superar o *Buy and Hold*.

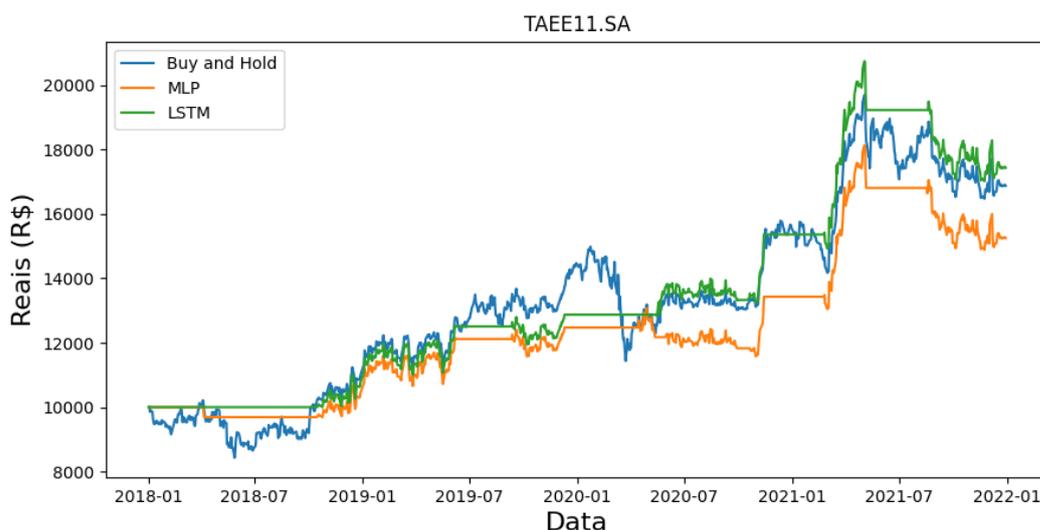


Figura 4.14: Resultados financeiros da simulação da TAEE.

A simulação da VALE foi realizada entre 02/01/2015 a 27/12/2018, conforme a Figura 4.15. A estratégia com MLP obteve resultado de R\$ 29.231,30 e R\$ 29.167,24 para a LSTM, os dois modelos tiveram resultados finais similares. No entanto, é possível observar que a LSTM se manteve superior à MLP durante toda a simulação, entretanto aparentou ter uma falha com o mecanismo de *stop loss* e obteve uma queda no capital total ao final do período. Enquanto a estratégia *Buy and Hold* resultou em R\$ 23.261,28 de capital total, ou seja, há uma boa margem entre as estratégias, mostrando que a performance do sistema proposto é satisfatória.

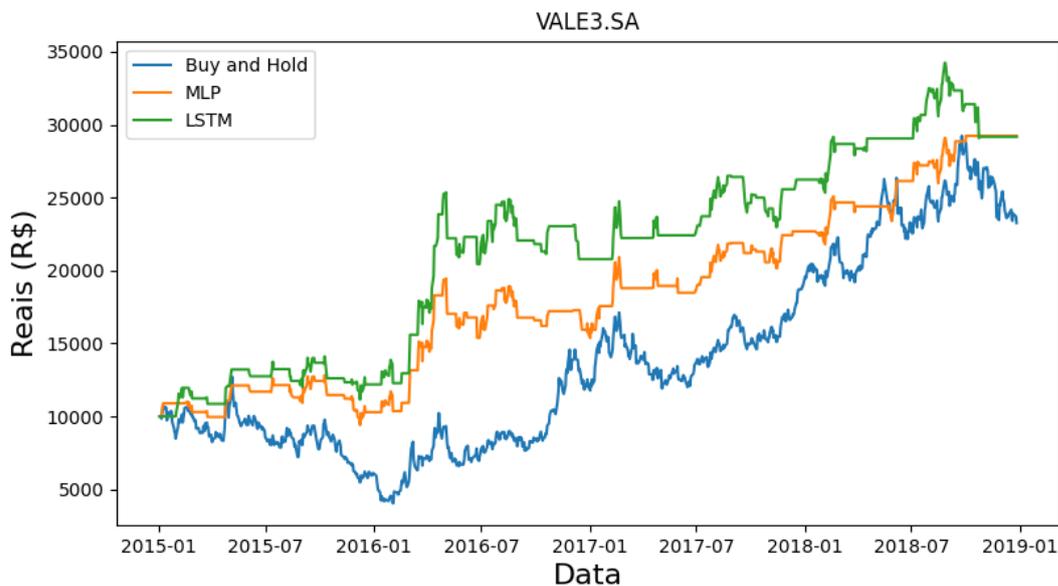


Figura 4.15: Resultados financeiros da simulação da VALE.

A simulação da ITUB foi realizada entre 02/01/2015 a 27/12/2018, conforme a Figura 4.16. As estratégias *Buy and Hold*, MLP e LSTM obtiveram resultado de R\$ 18.579,32, R\$ 18.270,73 e R\$ 15.162,89, respectivamente. Para este ativo, o resultado não foi o esperado, além da estratégia com MLP ter tido melhor desempenho do que com LSTM, nenhuma das duas conseguiu ultrapassar o *Buy and Hold*. No gráfico é possível notar que o sistema não conseguiu lidar com as quedas bruscas que ocorreram obtendo resultando inferior ao *Buy and hold* e, portanto, o sistema não foi satisfatório para este caso.

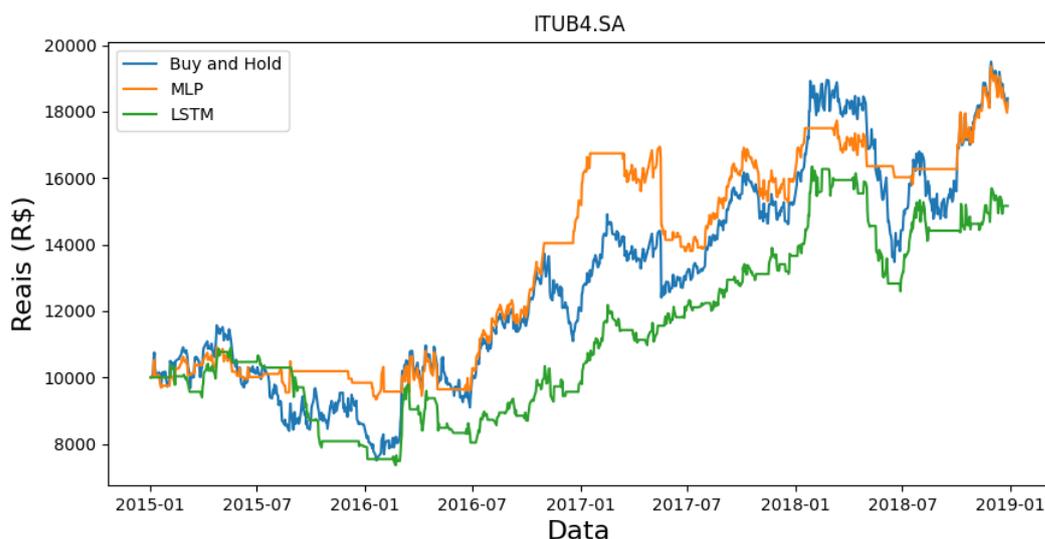


Figura 4.16: Resultados financeiros da simulação da ITUB.

A simulação da BOVA foi realizada entre 02/01/2015 a 27/12/2018, conforme a Figura 4.17. A estratégia com MLP obteve resultado de R\$ 15.399,51 e R\$ 19.429,12 para a LSTM, enquanto a estratégia *Buy and Hold* resultou em R\$ 17.488,36 de capital total, logo a LSTM se mostrou superior com uma margem razoável e, portanto, os resultados são satisfatórios.

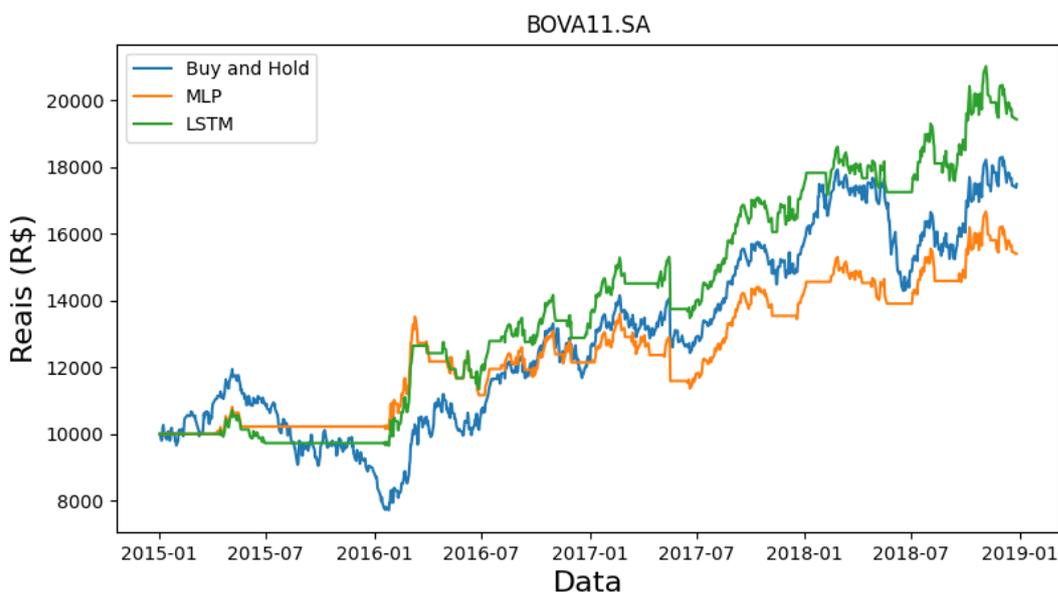


Figura 4.17: Resultados financeiros da simulação da BOVA.

A simulação da PETR foi realizada entre 02/01/2015 a 27/12/2018, conforme a Figura 4.18. A estratégia com MLP obteve resultado de R\$ 24.704,92 e R\$ 25.937,69 para

a LSTM, enquanto a estratégia *Buy and Hold* resultou em R\$ 24.230,77 de capital total, logo a LSTM se mostrou superior com uma margem decente e, portanto, os resultados são satisfatórios.

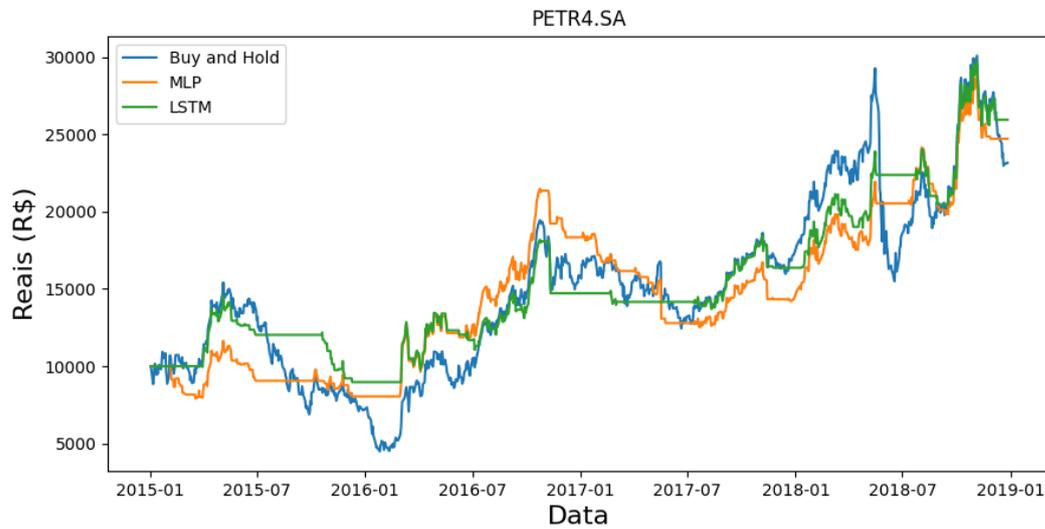


Figura 4.18: Resultados financeiros da simulação da PETR.

Os resultados da simulação são resumidos na Tabela 4.17. Os sistemas *Convolutional Neural Network - Technical Analysis* (CNN-TA) desenvolvido por Sezer e Ozbayoglu [45] e *Convolutional Neural Network - Trading Classifier* (CNN-TC) desenvolvido e usados na dissertação de mestrado de G. Nascimento [12], são adicionadas na tabela para fins de comparação. Estas estratégias abordam o problema como um problema de classificação, onde a RNA usada é uma CNN, que já retorna um sinal de compra, venda ou neutro na saída da rede. Os ativos VALE, ITUB, BOVA e PETR tiveram as mesmas base de dados e períodos usados para possibilitar a comparação.

Tabela 4.17: Resultados da Simulação.

Ativo	Estratégias				
	<i>Buy and Hold</i>	MLP	LSTM	CNN-TA	CNN-TC
BTC	USD 34.481,80	USD 105.335,72	USD 119.776,19	-	-
TAEF	R\$ 16.877,91	R\$ 15.252,36	R\$ 17.434,40	-	-
VALE	R\$ 23.261,28	R\$ 29.231,30	R\$ 29.167,24	R\$ 20.753,56	R\$ 60.315,19
ITUB	R\$ 18.579,32	R\$ 18.270,73	R\$ 15.162,89	R\$ 17.873,84	R\$ 24.337,77
BOVA	R\$ 17.488,36	R\$ 15.399,51	R\$ 19.429,12	R\$ 15.2585,10	R\$ 22.278,80
PETR	R\$ 24.230,77	R\$ 24.704,92	R\$ 25.937,69	R\$ 31.238,00	R\$ 14.928,37

É possível notar que o sistema proposto conseguiu ter resultados satisfatórios para 5 de 6 ativos que superaram a estratégia *Buy and Hold*. A maioria obtiveram melhores resultados usando LSTM, com exceção do caso dos ativo VALE e ITUB que tiveram dificuldades em reconhecer as quedas bruscas. A estratégia do sistema proposto, também conseguiu superar a estratégia CNN-TA para 3 de 4 ativos, mas só conseguiu superar a estratégia CNN-TC para o caso da PETR. No geral, a abordagem de G. Nascimento [12] se mostrou superior ao do sistema proposto, principalmente, para o ativo VALE que obteve um resultado significativamente maior que todas as outras estratégias.

Neste capítulo foram apresentados os resultados obtidos de cada etapa do estudo da previsão de séries temporais como suporte à decisão de negociação de ativos financeiros e as explicações das escolhas que foram tomadas, para desenvolver o sistema final. Com o sistema desenvolvido foi realizada uma última etapa para simular a capacidade de retorno do sistema, e assim, é comparado com outros trabalhos correlatos para aferir o quão satisfatório são os resultados. No capítulo seguinte será discutido as conclusões do trabalho, as limitações do sistema e o caminho possível a ser seguido para conseguir aperfeiçoar os resultados.

Capítulo 5

Conclusão

O trabalho proposto teve como objetivo a criação de um modelo de Redes Neurais Artificiais (RNA), implementada como um problema de previsão de séries temporais e analisar se a previsão do preço de fechamento de um ativo financeiro é capaz de dar suporte à tomada de decisão dentro do contexto do mercado financeiro. Para alcançar o objetivo, foi utilizada uma metodologia de treinamento adaptativa a fim de manter o modelo atualizado para tratar ocorrências de *concept drifts* e minimizar o erro da previsão. Com o propósito de verificar se a RNA consegue agregar valor à uma tomada de decisão, foi desenvolvido um sistema para realizar uma simulação de uma negociação no mercado financeiro e a partir desta, avaliar a performance financeira do sistema.

Os resultados obtidos ao aferir as diversas rodadas de treinamento e testes das redes usadas no trabalho, mostraram que a metodologia adaptativa teve impacto significativo no erro das previsões. Aliás, nos experimentos adaptativos, onde os potenciais máximos do desempenho das redes foram expostos, a arquitetura LSTM foi capaz de reduzir o erro de previsão em relação à arquitetura MLP. Porém, os ativos que apresentaram mudanças bruscas no comportamento, não tiveram melhoras significativas. Já na avaliação do modelo obteve-se resultados variados, os erros dos ativos TAEE, VALE e BOVA se mantiveram, aproximadamente, os mesmos para ambas as arquiteturas. Logo, é possível perceber que a escolha da arquitetura não é expressiva na mesma intensidade que a metodologia de adaptação implementada.

A simulação do sistema foi comparada com outros trabalhos similares encontrados na literatura. As estratégias chamadas de CNN-TA e CNN-TC, que ao abordarem a negociação de ativos financeiros como um problema de classificação, conquistaram resultados considerados satisfatórios em relação ao ganho de capital. A comparação também foi efetivada usando uma estratégia base, chamada de *Buy and Hold* que é de fato, simples e eficiente, demonstrando obter um lucro razoável em todas as simulações. O sistema desenvolvido alcançou o objetivo de superar a estratégia *Buy and Hold*, com exceção do

ativo ITUB, onde o sistema não conseguiu lidar com as quedas bruscas que ocorreram. O sistema também superou a estratégia CNN-TA com exceção do ativo PETR. Entretanto, a estratégia CNN-TC obteve ganhos significativamente maiores para os ativos VALE, ITUB e BOVA. Isto posto, o uso de uma RNA direcionada a um problema de previsão de séries temporais, mesmo implementando abordagens bastantes simples, mostrou ter aplicabilidade na área. A previsão agrega valor à uma decisão de negociação no mercado financeiro.

5.1 Limitações do Sistema Proposto

As limitações do sistema compreendem o uso da adaptação passiva, o sistema apenas atualiza o modelo continuamente, ao invés de detectar ativamente os *concept drifts*. Além do potencial de reduzir o erro de previsão, também, pode ser extremamente vantajoso para o usuário perceber quando uma mudança de comportamento do ativo acontece e quando se mantém. Outra limitação, é o período das séries temporais usadas, é possível obter dados com intervalos de 1 minuto, que além de apresentar mais dados para alimentar a rede, poderia reduzir significativamente o erro de previsão. Já que a variação entre dois dados diários tem o potencial de ser consideravelmente maior do que a variação entre minutos, entretanto o sistema teria que realizar todo o processamento em um tempo menor que 1 minuto. Ademais, o sistema não faz o escalonamento dos dados usando, por exemplo, a normalização *min-max*, pois é possível que uma previsão seja maior que o valor máximo do conjunto de dados. Entretanto outras formas de escalonamento podem ser analisadas e possivelmente aplicadas.

No decorrer do desenvolvimento da simulação, observou-se que a previsão é tão boa quanto a forma de se usar as informações disponíveis, com o sistema prevendo somente o preço de fechamento, acaba limitando o procedimento para tomar uma decisão. Ao incluir a previsão do preço mínimo e máximo, por exemplo, possibilita novas maneiras de se usar estas informações. Ademais, o sistema deste trabalho possui um tomador de decisão bastante simplificado, com isto em mente, é possível desenvolver um algoritmo de decisão que considera uma maior abrangência de informações e técnicas de negociação.

5.2 Trabalhos Futuros

As ideias que podem ser implementadas em trabalhos futuros foram no geral, citadas nas limitações do sistema, sendo a implementação de detecções de *concepts drifts*, e conseqüentemente, a implementação de metodologias de adaptação ativa. O uso das séries temporais com períodos menores, a previsão de todos os dados que compõem o *candlestick*

de um ativo e a construção de um algoritmo tomador de decisão mais robusto são possíveis questões à serem tratadas. Outra possível ideia, é testar o uso da arquitetura CNN para classificação, cuja saída da rede seria a direção de um movimento que seria usada na entrada de uma LSTM para prever a próxima série temporal. Estes são os principais focos que têm o potencial de alcançar um melhor resultado e serão explorados em um trabalho futuro.

Referências

- [1] Google. Google trends. <https://trends.google.com.br/trends/explore?date=all&q=machine%20learning>. [Online; acessado 05-03-2022]. x, 2
- [2] yahoo/finance. Gráficos, dados e notícias da ibovespa. <https://br.financas.yahoo.com/quote/%5EBVSP/chart?p=%5EBVSP>. [Online; acessado 05-03-2022]. x, 2
- [3] Dobrivoje K. Palit, Ajoy; Popovic. *Computational Intelligence in Time Series Forecasting*. Springer, 2005. x, 14, 15, 16
- [4] Simon Haykin. *Neural Networks and Learning Machines Third Edition*. Prentice Hall PTR, 2009. x, 19, 20, 23, 28, 31
- [5] Niklas Donges. A guide to rnn: Understanding recurrent neural networks and lstm networks. <https://builtin.com/data-science/recurrent-neural-networks-and-lstm>, 2021. [Online; acessado 20-03-2022]. x, 30, 31
- [6] Ian *et al.* Goodfellow. *Deep learning-(Adaptive Computation and Machine Learning series)*. Massachusetts Institute of Technology, 2016. x, 27, 29, 31, 32
- [7] Juan *et al.* Piñeiro-Chousa. Influence of social media over the stock market. *Psychology & Marketing*, 2016. 3, 11
- [8] Flávio Lemos. *Análise Técnica dos Mercados Financeiros*. Editora Saraiva, 2016. 3, 10, 11, 12, 13
- [9] Cláudio P. Faustino. Previsão de séries temporais via modelos baseados em regras. Technical report, Universidade Federal de Itajubá, 2011. Disponível em https://repositorio.unifei.edu.br/xmlui/handle/123456789/1379?locale-attribute=pt_BR. 3
- [10] Chi-Jie *et al.* Lu. Financial time series forecasting using independent component analysis and support vector regression. *Elsevier*, 2009. 3, 14
- [11] Xiangxi Jiang. Bitcoin price prediction based on deep learning methods. *Journal of Mathematical Finance*, 2020. 3
- [12] Davi G. Nascimento. Redes neurais convolucionais aplicas à negociação de ativos no mercado financeiro. Technical report, Centro Universitário FEI, 2021. Disponível em <https://repositorio.fei.edu.br/handle/FEI/3212>. 3, 67, 68

- [13] Rodolfo C. Cavalcante. An adaptive learning system for time series forecasting in the presence of concept drift. Technical report, Universidade Federal do Pernambuco, 2017. Disponível em <https://repositorio.ufpe.br/handle/123456789/25349>. 4, 15, 17, 18, 47
- [14] Alexandre Galvão. *Mercado financeiro: uma abordagem prática dos principais produtos e serviços*. Elsevier, 2006. 6, 7
- [15] Comissão de Valores Mobiliários. Entendendo o mercado de valores mobiliários. https://www.investidor.gov.br/menu/primeiros_passos/Entendendo_mercado_valores.html. [Online; acessado 18-02-2022]. 7
- [16] Eugene F. Fama. Random walks in stock market prices. *Financial Analysts Journal*, 1965-1974. 8, 9, 10, 11
- [17] James Chen. Stock analysis. <https://www.investopedia.com/terms/s/stock-analysis.asp>, 2021. [Online; acessado 26-02-2022]. 9, 10, 12
- [18] Tim Smith. Market sentiment. <https://www.investopedia.com/terms/m/marketsentiment.asp>, 2021. [Online; acessado 27-02-2022]. 11
- [19] John; W.H.C.Bassetti D. Edwards, Robert; Magee. *Technical Analysis of Stock Trends, 8th edition*. CRC Press, 2001. 12
- [20] Steve Nison. *Japanese Candlestick Charting Techniques*. New York Institute of Finance, 1979. 12
- [21] J. Welles Wilder. *New Concepts in Technical Trading Systems*. Greensboro, N.C. : Trend Research, 1978. 13
- [22] Jian-Zhou *et al.* Wang. Forecasting stock indices with back propagation neural network. *Elsevier*, 2011. 14
- [23] Gwilym *et al.* E.P. Box, George; M. Jenkins. *Time Series Analysis, Forecasting and Control, Fifth Edition*. Wiley, 2016. 15
- [24] João Gama. A survey on learning from data streams: current and future trends. *Springer*, 2012. 16, 17
- [25] João *et al.* Gama. Learning with drift detection. *Springer*, 2004. 17
- [26] Indre *et al.* Zliobaite. Towards cost-sensitive adaptation: When is it worth updating your predictive model? *Science Direct*, 2015. 18
- [27] A. M. Turing. Computing machinery intelligence. *Psychological Review*, 1950. 18
- [28] W. McCulloch, W.S.; Pitts. A logical calculus of the ideas immanent in nervous activity. *Springer*, 1943. 18, 19, 21
- [29] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 1958. 18, 28

- [30] Dean Pomerleau. Alvin: An autonomous land vehicle in a neural network. In D.S. Touretzky, editor, *Proceedings of (NeurIPS) Neural Information Processing Systems*, pages 305 – 313. Morgan Kaufmann, December 1989. 19
- [31] Eric W. Weisstein. Ramp function. <https://mathworld.wolfram.com/RampFunction.html>. [Online; acessado 01-05-2022]. 25
- [32] Jason Brownlee. A gentle introduction to the rectified linear unit (relu). <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>, 2020. [Online; acessado 01-05-2022]. 25, 26
- [33] A. Cauchy. Méthode générale pour la résolution des systèmes d'équations simultanées. In Claude Lemaréchal, editor, *Cauchy and the Gradient Method*, pages 251 – 254. Documenta Mathematica, 1847. 27
- [34] James L. Rumelhart, David E.; McClelland. *A General Framework for Parallel Distributed Processing*. MIT Press, 1986. 28
- [35] Jimmy Lei Kingma, Diederik P. e Ba. Adam: A method for stochastic optimization. In *ICLR 2015*, pages 1 – 15. ICLR, 2015. 29, 30
- [36] Jason Brownlee. Gentle introduction to the adam optimization algorithm for deep learning. <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/#:~:text=Adam%20is%20a%20replacement%20optimization,sparse%20gradients%20on%20noisy%20problems.>, 2021. [Online; acessado 13-03-2022]. 30
- [37] Yoram Duchi, John; Hazan Elad e Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *JMLR Volume 12*, pages 2121 – 2159. Journal of Machine Learning Reasearch, 2011. 30
- [38] Jürgen Hochreiter, Sepp; Schmidhuber. Long short-term memory. In *NEUC Volume 9, Issue 8*, page 1735 – 1780. Massachusetts Institute of Technology, 1997. 30, 31
- [39] Jason Brownlee. A gentle introduction to exploding gradients in neural networks. <https://machinelearningmastery.com/exploding-gradients-in-neural-networks/>, 2019. [Online; acessado 20-03-2022]. 30
- [40] Jason Brownlee. A gentle introduction to k-fold cross-validation. <https://machinelearningmastery.com/k-fold-cross-validation/>, 2018. [Online; acessado 31-03-2022]. 32
- [41] Yahoo finance. <https://finance.yahoo.com/>. [Online; acessado 01-05-2022]. 37
- [42] pandas-datareader. <https://pandas-datareader.readthedocs.io/en/latest/>. [Online; acessado 01-05-2022]. 37

- [43] Jason Brownlee. Time series forecasting as supervised learning. <https://machinelearningmastery.com/time-series-forecasting-supervised-learning/>, 2020. [Online; acessado 03-05-2022]. 37
- [44] Abhigyan Singh. Understanding polynomial regression!!! <https://medium.com/analytics-vidhya/understanding-polynomial-regression-5ac25b970e18>. [Online; acessado 01-05-2022]. 47
- [45] Ahmet Murat Sezer, Omer Berat; Ozbayoglu. Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. In *Applied Soft Computing*, pages 525–538. Elsevier, 2018. 67