RODRIGO MENEZES SOBRAL ZACARONI

DESENVOLVIMENTO DE ROBÔ DE INVESTIMENTO PARA DAY TRADE BASEADO EM SVM ONE CLASS E RNA

LAVRAS – MG

2021

RODRIGO MENEZES SOBRAL ZACARONI

DESENVOLVIMENTO DE ROBÔ DE INVESTIMENTO PARA DAY TRADE BASEADO EM SVM ONE CLASS E RNA

Dissertação apresenta à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Engenharia de Sistemas e Automação, para obtenção do título de Mestre.

Prof. DSc. Danton Diego Ferreira
Orientador

LAVRAS – MG

2021

Ficha catalográfica elaborada pelo Sistema de Geração de Ficha Catalográfica da Biblioteca Universitária da UFLA, com dados informados pelo(a) próprio(a) autor(a).

Zacaroni, Rodrigo Menezes Sobral.
Desenvolvimento de robô de investimento para day trade baseado em SVM One Class e RNA / Rodrigo Menezes Sobral Zacaroni. – Lavras : UFLA, 2021.
103 p. :

Orientador: Prof. DSc. Danton Diego Ferreira.
Dissertação (Mestrado Profissional)–Universidade Federal de Lavras, 2021.
Bibliografia.

1. Mercado Financeiro. 2. Inteligência Artificial. 3. MetaTrader 5. I. Ferreira, Danton Diego. II. Título.

RODRIGO MENEZES SOBRAL ZACARONI

DESENVOLVIMENTO DE ROBÔ DE INVESTIMENTO PARA DAY TRADE BASEADO EM SVM ONE CLASS E RNA

Dissertação apresenta à Universidade Federal de Lavras, como parte das exigências do Programa de Pós-Graduação em Engenharia de Sistemas e Automação, para obtenção do título de Mestre.

APROVADA em 15 de Dezembro de 2021.

Prof. Dr. Danton Diego Ferreira	UFLA
Prof. Dr. Wilian Soares Lacerda	UFLA
Profa. Dr. Alexandre Pimenta	IFMG

Prof. DSc. Danton Diego Ferreira
Orientador

LAVRAS – MG
2021

AGRADECIMENTOS

Agradeço primeiramente a Deus por me manter forte, perseverante e saudável para enfrentar este grande desafio. Agradeço aos meus pais, José Wander e Marizilda, que são minha fonte de inspiração e sempre me incentivaram a permanecer estudando e me capacitando. Agradeço à minha amada esposa Francyelen e filhos Miguel e Murilo, que são a minha fortaleza e principal fonte de energia para continuar sempre em busca do melhor para lhes dar, é por vocês que tudo faço. Agradeço a todos os quais não citei aqui, mais que de forma direta ou indireta contribuíram para o alcance deste objetivo.

Uma máquina pode fazer o trabalho de cinquenta pessoas comuns. Nenhuma máquina pode fazer o trabalho de uma pessoa extraordinária.
(Elbert Hubbard)

RESUMO

O mercado financeiro é o ambiente de negociação de diversos instrumentos financeiros, incluindo ações, títulos, moedas e derivativos. Este mercado é de vital importância para o bom funcionamento das economias capitalistas. Um segmento importante do mercado financeiro é o mercado de valores mobiliários, que possibilita a especulação sobre contratos futuros. No Brasil o minicontrato futuro de Ibovespa (WIN) é o ativo financeiro mais negociado por pessoas físicas na modalidade de negociação intradiário (daytrade). *Traders* e investidores que realizam operações de compra e venda neste mercado, encontram inúmeros desafios que dificultam a tarefa no momento da tomada de decisão. Desafios estes que podem afetar tanto os investidores mais experientes quanto os iniciantes neste mercado. A maioria dos estudos disponíveis na literatura empregam abordagens estatísticas e econométricas convencionais para tentar prever o preço futuro de determinado ativo financeiro por meio de análise de regressão. Observa-se então, uma carência de pesquisas no campo de desenvolvimento de modelos dedicados a previsão da direção do preço, ou seja, tratar o problema como sendo de classificação. Neste contexto, este trabalho propõe um modelo de inteligência artificial baseado em SVM *One Class* e Redes Neurais Artificiais (RNA), o qual a proposta é prever a direção do preço do contrato futuro do índice Bovespa (WIN) no tempo gráfico de 5 (cinco) minutos. Os principais diferenciais deste trabalho comparado com os disponíveis na literatura, está no uso do SVM *One Class* e o filtro *Savitzky-Golay*. Outro ponto de destaque consiste na validação dos resultados via *backtesting*, utilizando um sistema de negociação automatizado para o mercado financeiro denominado de Expert Advisor (EA) desenvolvido na plataforma gratuita MetaTrader5 (MT5). O *backtesting* permitiu obter métricas em ambiente de simulação com os dados reais do mercado financeiro. Os resultados obtidos do *backtesting* são mais realistas e por isso divergem dos resultados alcançados apenas analisando a assertividade do modelo de IA que apresentou uma taxa média de 60,22% de assertividade das previsões. Esta análise serviu para provar a importância da validação de modelos de IA aplicando sistemas de *backtesting* para analise dos resultados.

Palavras-chave: SVM *One Class*. Redes Neurais Artificiais. Savitzky-Golay. MetaTrader5. *Expert Advisor*.

ABSTRACT

The financial market is the trading environment for various financial instruments, including stocks, bonds, currencies and derivatives. This market is of vital importance for the proper functioning of capitalist economies. An important segment of the financial market is the securities market, which enables speculation on futures contracts. In Brazil, the mini Ibovespa futures contract (WIN) is the financial asset most traded by individuals in the intraday trading modality (daytrade). Traders and investors who carry out purchase and sale operations in this market face numerous challenges that make their task difficult at the time of decision making. These challenges can affect both more experienced investors and beginners in this market. Most studies available in the literature employ conventional statistical and econometric approaches to try to predict the future price of a given financial asset through regression analysis. Therefore, there is a lack of research in the field of developing models dedicated to predicting the direction of price, that is, treating the problem as one of classification. In this context, this work proposes an artificial intelligence model based on SVMOne Class Artificial Neural Networks (ANN), which the proposal is to predict the direction of the price of the Bovespa Index (WIN) futures contract in a 5 (five) minute graph time. The main differential of this work compared to those available in the literature is the use of the SVMOne Class or the Savitzky-Golay filter. Another highlight is the validation of results via backtesting, using an automated trading system for the financial market called Expert Advisor (EA) developed on the free MetaTrader5 (MT5) platform. Backtesting allowed to obtain metrics in a simulation environment with real data from the financial market. The results obtained from the backtesting are more realistic and, therefore, differ from the results achieved only by analyzing the assertiveness of the AI model, which presented an average rate of 60.22% of assertiveness in the predictions. This analysis served to prove the importance of validating AI models by applying backtesting systems to analyze the results.

Keywords: SVM One Class. Artificial Neural Networks. Savitzky-Golay. MetaTrader5. Expert Advisor.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

BB Banda de Bollinger

BBI Número

BBS Banda de Bollinger Superior

DBB Distância entre as Bandas de Bollinger

DBBI Distância da Banda de Bollinger Inferior

DBBS Distância da Banda de Bollinger Superior

DSG Distância do preço suavizado pelo filtro de Savitzky – Golay

DSMA Distância da SMA

EA *Expert Advisor*

EMA *Exponential Moving Average*

HME Hipótese do Mercado Eficiente

IA Inteligência Artificia

IN Índice de Negociabilidade

IND Contrato Futuro de Índice Ibovespa

LWMA *Linearly Weighted Moving Average*

MM Média Móvel

MQL5 *MetaQuotes Language* 5

MT5 MetaTrader 5

OHLC *Open High Low Close*

PCA Análide de Componentes Principais

PIB Produto Interno Bruto

RCP Relação Corpo Pavio

RNA Redes Neurais Artificiais

RP Relação Pavio

RR Relação Range

RV Relação Volume

RVT Relação Volume por Tick

SMA *Simple Moving Average*

SMMA *Smoothed Moving Average*

VT Volume por Tick

WIN Minicontrato Futuro de Ibovespa

SUMÁRIO

1 INTRODUÇÃO

Prever preços de ativos financeiros no mercado de capitais e tomar uma decisão de negociação de compra ou venda é uma tarefa desafiadora para pesquisadores e investidores, devido a uma série de incertezas envolvidas. As variações de preços estão sujeitas aos problemas de séries temperais de alta dimensionalidade e não estacionárias, onde fatores econômicos, políticos e até mesmo emocionais afetam as variações dos preços dos ativos (PATEL et al., 2015; CHANG et al., 2011).

A tomada de decisão de compra ou venda no mercado financeiro é baseada em dois tipos de análise, sendo elas a análise fundamentalista e técnica. A primeira leva em consideração fatores macroeconômicos como PIB (Produto Interno Bruto) e índices econômicos do país, potencial de crescimento das empresas, valor intrínseco dos ativos, desempenho da indústria, economia, fatores políticos e etc. A análise técnica é a tentativa de prever os preços futuros de um ativo, por meio da análise gráfica dos preços e volumes anteriores em busca de padrões de tendência ou reversão, considerando as estatísticas geradas pela movimentação do mercado (HUANG et al., 2015; MURPHY, 1999; EDWARDS; MAGEE; BASSETTI, 2012; PATEL et al., 2015).

Estratégias de negociações automatizadas, ou negociação algorítmica, têm se tornado um tópico de pesquisa popular e amplamente discutido entre pesquisadores e investidores do mercado financeiro. A tomada de decisão em negociações automatizadas é baseada em um modelo preditivo que explora os dados online sem qualquer informação supervisionada, objetivando antecipar o preço futuro de um ativo financeiro, de forma a obter lucro através das previsões do modelo (GIACOMEL, 2016; LEI et al., 2020).

Este cenário é bastante propício para o uso de técnicas de Inteligência Artificial (IA), que têm sido amplamente exploradas por pesquisadores para prever as cotações futuras de ativos do mercado financeiro. Modelos de previsão baseados em Redes Neurais Artificiais (RNA) são os mais utilizados por pesquisadores. Guresen, Kayakutlu e Daim (2011) afirmam que, modelos de previsão de series temporais financeiras baseados em RNA são, na maioria dos casos, melhores preditores quando comparados com os modelos estatísticos. Segundo Giacomel (2016), na maioria dos trabalhos presentes na literatura busca-se apenas a previsão dos preços futuros com foco na comparação entre modelos de RNA ou na acurácia da rede neural. Neste contexto, identifica-se que existe uma lacuna de pesquisa relativa à aplicação prática em ambiente real de negociação utilizando plataformas de *Home Broker* com uso de ferramentas para negociação

automatizada (*algorithmic trading* ou *algotrading*) aplicando técnicas de predição baseadas em IA.

O *Home Broker* é um sistema que possibilita o acesso à bolsa de valores, por exemplo a Bolsa de Valores, Mercadorias e Futuros de São Paulo (BM&FBOVESPA). Por meio dele, é possível conectar usuários ao pregão eletrônico, via rede de internet, o que possibilita realizar operações de compra ou venda de ativos do mercado de capitais (AMORIM; PAULA; ZANE, 2011).

Um exemplo de plataforma de negociação gratuita é o MetaTrader 5[1] (MT5), que segundo o site oficial da plataforma, é um programa para realizar negociações on-line, fazer análise técnica e utilizar sistemas de *trading* automatizados nos mercados financeiros. Além disso o MT5 oferece o MetaEditor, um ambiente de desenvolvimento em linguagem de programação *MetaQuotes Language 5* (MQL5), uma linguagem moderna de alto nível desenvolvida pela *MetaQuotes Software Corp*[2] para desenvolvimento de indicadores técnicos, robôs de negociação (*Expert Advisor*) e para aplicativos auxiliares, possibilitando a automatização de negociação nos mercados financeiros. O *Expert Advisor* (EA) é um sistema que gerencia de forma automática os processos de negociação com base nas regras comerciais estabelecidas nele por intermédio de um algoritmo específico, podendo realizar operações de compra e venda e gerenciar ordens pendentes (METAQUOTES, 2000 - 2021).

A proposta deste trabalho consiste no desenvolvimento de um robô de investimento para a plataforma gratuita MT5, em que o EA deverá utilizar RNA e SVM *One Class* como ferramenta principal para tomada de decisão em negociações automatizadas de compra e venda no prazo máximo de um dia ou um único pregão, modalidade essa conhecida como *day trade*.

1.1 Objetivo

O objetivo geral deste trabalho consiste no desenvolvimento de um EA em linguagem de programação MQL5, para plataforma MetaTrader 5. O EA deverá utilizar SVM *One Class* e RNA para prever a tendência ou direção de movimentação do ativo,“alta” ou “baixa”, e por fim, ser capaz de realizar negociações *day trade* de forma eficiente.

Para alcançar o objetivo geral, têm-se os objetivos específicos a seguir:

- extrair novas *features* dos dados financeiros disponíveis;

[1] https://www.metatrader5.com/pt
[2] https://www.metaquotes.net/ru

- aplicar métodos de seleção de *features*, no pré-processamento dos dados;
- selecionar um modelo com melhor acurácia possível para o problema de classificação de duas classes, ou seja, classificar se a cotação futura do ativo será de alta ou de baixa;
- testar a eficiência do modelo preditor em ambiente real de negociação, por meio de *back-testing* na plataforma MT5.

1.2 Organização do Texto

A organização textual deste trabalho está estruturada conforme descrito a seguir: O Capítulo 2 apresenta os principais conceitos teóricos relacionados ao tema do trabalho; O Capítulo 3 apresenta uma síntese de alguns dos trabalhos atuais relacionados ao tema, além de apresentar as novidades e relevância do trabalho aqui proposto; O Capítulo 4 detalha a metodologia empregada, a hipótese de estratégia e os recursos utilizados na pesquisa; O Capítulo 5 apresenta a análise dos resultados obtidos; Por fim, no Capítulo 6 são apresentadas as conclusões do trabalho e as sugestões de continuidade da pesquisa em trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capítulo é apresentada uma síntese dos principais conceitos abordados neste trabalho sobre o mercado financeiro, indicadores técnicos, pré-processamento de dados, técnicas de IA e a plataforma MT5.

2.1 Séries Temporais Financeiras

Diversos fenômenos do nosso cotidiano e do universo científico são representados por séries temporais, o qual uma sequência de observações estocásticas (y_1,y_2, y_3, ..., y_t) são analisadas ou registradas em determinada periodicidade temporal (t_1, t_2, t_3, ..., t_n). Matematicamente y é uma função de t definida por $y = F(t)$. A Fig. 2.1 apresenta um exemplo gráfico de uma série temporal financeira da cotação do índice Bovespa, o mais importante indicador do desempenho médio das cotações das ações negociadas na Bolsa de Valores de São Paulo.

É conhecido que o comportamento das séries temporais financeiras sofre influência de diversos fenômenos, tais como fatores macroeconômicos, políticos, sociais, climáticos entre outros. Neste contexto, justifica-se a complexidade de uma modelagem analítica a cerca da previsibilidade de series temporais financeiras (WOOLDRIDGE, 2006). No entanto, na teoria de finanças existe a disciplina de Econometria para séries financeiras, que tem como objetivo estimar o comportamento e fazer previsões de séries temporais financeiras, com base nos dados históricos. Hoffmann (2016) define a econometria como sendo a aplicação de métodos matemáticos e estatísticos a problemas de economia, sendo a análise de regressão o método mais importante.

O estudo das séries temporais financeiras é de extrema importância para identificação de características de sazonalidade, periodicidade, tendência e previsibilidade de valores futuros da série, sejam de curto ou longo prazo.

2.2 Ibovespa B3

O B3 (Brasil, Bolsa, Balcão) é o nome da Bolsa de Valores do Brasil e Ibovespa é o principal índice das ações do país. O Ibovespa é um indicador que mede o desempenho das ações negociadas na B3 e reúne as empresas mais importantes do mercado de capitais brasileiro (B3, 2021c).

Figura 2.1 – Exemplo de série temporal

Fonte: do autor

Reavaliado a cada quatro meses, o índice é resultado de uma carteira teórica de ativos que no processo de avaliação pode incluir ou excluir empresas conforme regras de enquadramento. Basicamente o índice é composto pelas ações e units[1] de companhias listadas na B3 que atendem aos seguintes critérios (B3, 2018; B3, 2021c; FGV; ABDI, 2018):

1. Estar entre os ativos que representem 85% em ordem decrescente de Índice de Negociabilidade (*IN*).

$$IN = \frac{\sum_{i=1}^{p} \sqrt[3]{\frac{n_a}{N} \times \left(\frac{v_a}{V}\right)^2}}{P} \tag{2.1}$$

Onde:

[1] Ativos compostos por mais de uma classe de valores mobiliários, como ações ordinárias, ações preferenciais e afins.

- n_a é o número de negócios com o ativo a no mercado a vista (lote-padrão);
- N é o número total de negócios no mercado a vista da B3 (lote-padrão);
- v_a é o volume financeiro gerado pelos negócios com o ativo a no mercado a vista (lote-padrão);
- V é o volume financeiro total do mercado a vista da B3 (lote-padrão);
- P é o número total de pregões no período.

2. Tenham sido negociadas em mais de 95% do total de pregões do período de vigência das três últimas carteiras teóricas;

3. Apresentem participação, em termos de volume, superior a 0,1% do total, considerando o período de vigência das três últimas carteiras teóricas;

4. Não ser classificado como Penny Stock[2];

Não é possível investir diretamente no índice Ibovespa (IBOV), no entanto a B3 disponibiliza alguns produtos ligados ao IBOV no mercado que são altamente negociados, sendo eles, os contratos futuros do IBOV. Com os contratos futuros, é possível especular sobre o valor do IBOV, dando possibilidade aos investidores de negociarem as expectativas futuras do mercado de ações, sem a necessidade de realizar a compra de toda a cesta de ações que compõem o índice. Em termos didáticos, IBOV é o termômetro do mercado e o contrato de índice futuro é a aposta se a temperatura sobe ou cai (B3, 2021b).

2.2.0.1 Contrato Futuro de Ibovespa

O contrato futuro de índice (IND), conhecido também como contrato padrão do futuro de Ibovespa, é um derivativo que permite a negociação da expectativa de preço do IBOV e são negociados com uma data de vencimento predeterminada. A Tabela 2.1 apresenta algumas especificações do IND.

Uma das vantagens de negociar com este ativo, consiste no fato dos contratos não serem negociados pelo seu valor total. Ao negociar o contrato de IND o investidor compra apenas

[2] Conforme o "MANUAL DE DEFINIÇÕES E PROCEDIMENTOS DOS ÍNDICES DA B3" (http://www.b3.com.br/data/files/AF/83/C4/BA/25CB7610F157B776AC094EA8/Conceitos-Procedimentos-nov2018.pdf), Penny Stock é definido como um ativo cujo valor médio ponderado obtido para o período pertinente a vigência da carteira anterior ao rebalanceamento, desconsiderando-se o último dia desse período, é inferior a R$1,00 (um real).

Tabela 2.1 – Características técnicas do IND

IND	
Tamanho do contrato	R$ 1,00 x pontuação
Cotação	Pontos
Lote padrão	5 Contratos
Variação mínima de apregoação	5 pontos
Mês de vencimento do contrato	Meses pares
Dia de vencimento do contrato	Quarta feira mais próxima do dia 15

Fonte: (B3, 2021b)

o direito de realizar o lucro ou prejuízo sobre as oscilações do ativo. Outra vantagem é a alavancagem, o que significa que é possível movimentar quantias maiores do que o investidor tem em conta, bastando apenas dispor de uma fração do total da operação, conhecida como margem de garantia (B3, 2021a).

2.2.0.2 Minicontrato Futuro de Ibovespa

Assim como o IND, o minicontrato futuro de Ibovespa (WIN) é também um derivativo do IBOV. O WIN foi criado em 2011 no intuito de viabilizar o acesso de pessoas físicas e pequenas empresas ao mercado de futuros do IBOV. A diferença entre negociar com IND e WIN é o valor financeiro necessário para negociar com ambos, no caso do WIN o investidor precisará de um valor financeiro 25 vezes menor em comparação com IND. A Tabela 2.2 deixa clara a diferença entre os ativos.

Tabela 2.2 – Características técnicas do WIN

WIN	
Tamanho do contrato	R$ 0,20 x pontuação
Cotação	Pontos
Lote padrão	1 Contratos
Variação mínima de apregoação	5 pontos
Mês de vencimento do contrato	Meses pares
Dia de vencimento do contrato	Quarta feira mais próxima do dia 15

Fonte: (B3, 2021b)

2.3 Grafico de *Candlestick*

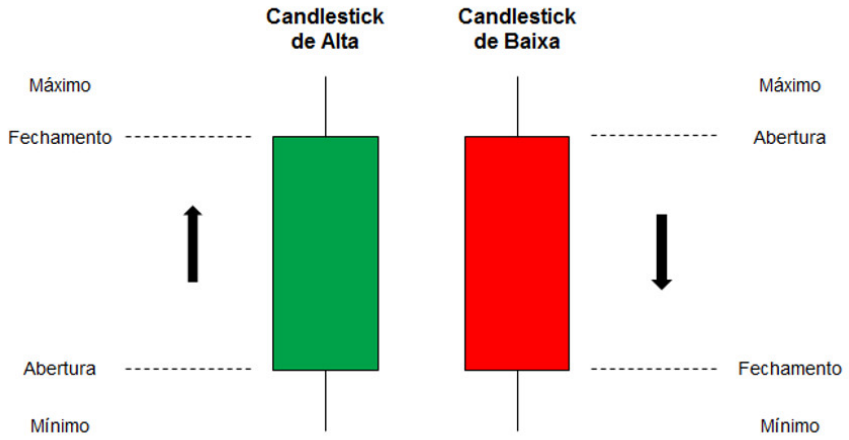
O gráfico de *Candlestick*, popularmente conhecido como gráfico de velas, foi originalmente desenvolvido por Munehisa Honma, um comerciante de arroz de Sakata no Japão, du-

rante o século XVIII. As informações da dinâmica dos preços por unidade de tempo fornecidas por cada *Candle* são as seguintes:

- *Open* - preço de abertura;
- *High* - preço máximo alcançado;
- *Low* - preço mínimo alcançado;
- *Close* - preço de fechamento

O tempo gráfico é a unidade de tempo da representação do histórico de preço de uma vela OHLC (*open, high, low, close*), no MT5 o tempo gráfico mínimo representado pela plataforma é de 1(um) minuto e a máxima representação é mensal. As velas podem ser classificadas como vela de alta ou vela de baixa, a Fig. 2.2 ilustra a representação gráfica de ambas as velas.

Figura 2.2 – Representação gráfica da vela



Fonte: bit.ly/3pMj8lE. Acessado em Fev/2021

Negociações baseadas em padrões gráficos de velas são muito utilizadas por investidores, no entanto, no universo científico a técnica de análise gráfica é menos estudada quando comparada com a análise técnica tradicional (CHEN; CHEN, 2016).

2.4 Análise Técnica

Charles H. Dow, cofundador da *Dow Jones & Company* e fundador do *The Wall Street Journal*, o maior jornal de notícias econômicas dos Estados Unidos, foi quem introduziu a teoria de Dow no final de 1800. A partir daí, a análise técnica foi amplamente utilizada por *traders*,

investidores e especuladores do setor financeiro (PARK; IRWIN, 2011). No livro *"Technical Analysis of Stock Trends"* os autores Edwards, Magee e Bassetti (2012) definem a análise técnica como sendo a ciência de registrar, geralmente em forma gráfica, o histórico real de negociação tais como, variações de preços, volume de transações, médias móveis, osciladores e etc., e em seguida, deduzir dessa história retratada a provável tendência futura.

A análise técnica pode ser subdividida em grafista que é baseada no reconhecimento de padrões gráficos de velas e a computadorizada que faz uso dos indicadores numéricos. Pimenta (2017) afirma que a análise técnica contraria a teoria da Hipótese do Mercado Eficiente (HME) de (FAMA, 1970), que por sua vez acredita na ineficácia da previsão dos preços de ativos financeiros, devido às características randômicas e independentes de um movimento *browniano* de séries temporais financeiras. Ao contrário desta concepção, *traders*, investidores e pesquisadores que utilizam a análise técnica, afirmam identificar padrões de previsibilidade a partir de uma série histórica financeira. Neste contexto, apresenta-se nas subseções seguintes a teoria matemática dos indicadores técnicos utilizados neste trabalho, dos quais as *features* são obtidas para o treinamento do modelo de *Machine learning*.

2.4.1 Médias Móveis

Em séries temporais financeiras a média móvel (MM) é um indicador técnico amplamente utilizado para determinar o valor médio dos preços em um certo período amostral de tempo, além de também determinar a tendência e suavizar a série temporal, eliminando as flutuações aleatórias de curto prazo (KAUFMAN, 2013).

No sistema de negociação MT5 existem quatro tipos diferentes de médias móveis: Simples (também chamada de Aritmética), Exponencial, Suavizada e Ponderada. A Média Móvel pode ser calculada para qualquer conjunto de dados sequenciais, incluindo os preços de abertura e fechamento, máximas e mínimas do preço, o volume de negociação ou quaisquer outros indicadores Metaquotes (2000 - 2021).

2.4.1.1 Média Móvel Simples (SMA)

Como o próprio nome sugere, a SMA (*Simple Moving Average*) é a forma mais simples de calcular uma média móvel, em dados financeiros ela é calculada, por exemplo, através do somatório dos preços de fechamento ao longo de determinado número de amostras (*candlestck*)

e então dividido pelo número de amostras. A equação 2.2 é usada para calcular a média móvel simples (KAUFMAN, 2013).

$$SMA = \frac{1}{n}\sum_{i=1}^{n} p_i, \quad n > 1 \tag{2.2}$$

Onde:

- **n** é o número de amostras;
- **p** é o valor do preço individual das amostras;

Figura 2.3 – Média Móvel Simples

Fonte: do autor.

2.4.1.2 Média Móvel Exponencial

A EMA (*Exponential Moving Average*) é calculada de forma que os valores de preços mais recentes recebem pesos maiores, por isso esse tipo de média móvel também é conhecido como média móvel exponencialmente ponderada. O fato dos pesos atribuídos aos dados

mais recentes serem numericamente maiores, impactam de forma mais significativa no valor da média móvel as variações mais recentes no preço (RAUDYS; LENČIAUSKAS; MALČIUS, 2013).

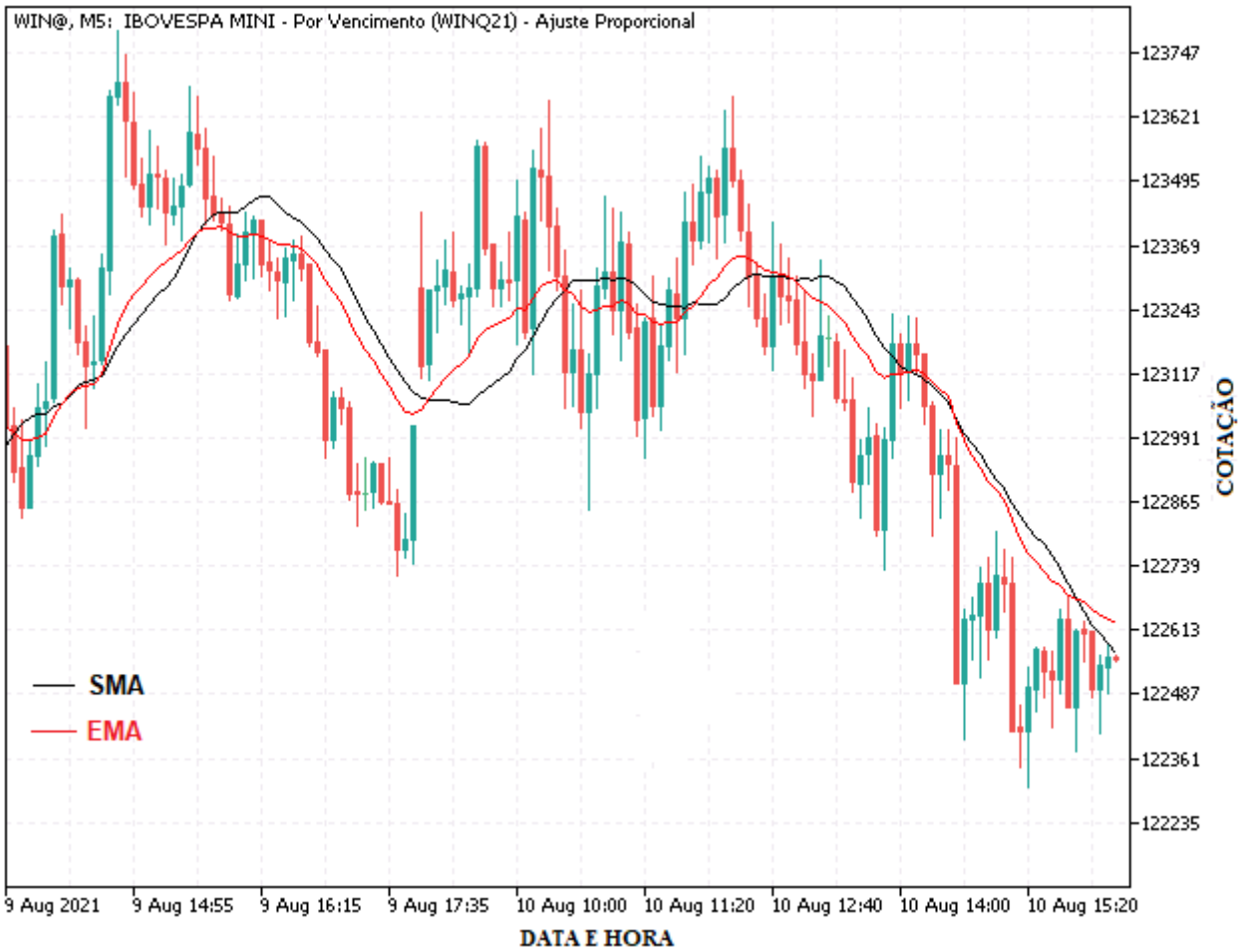
A equação 2.4 é usada para calcular a média móvel exponencial.

$$EMA = \alpha \sum_{i=0}^{n} (1-\alpha)^i p_{i+1} \tag{2.3}$$

onde:

- **n** é o número de amostras;
- α é o peso atribuído as amostras tal que $\alpha = \frac{2}{n+1}$
- p é o valor preço individual das amostras;

Figura 2.4 – Média Móvel Exponencial

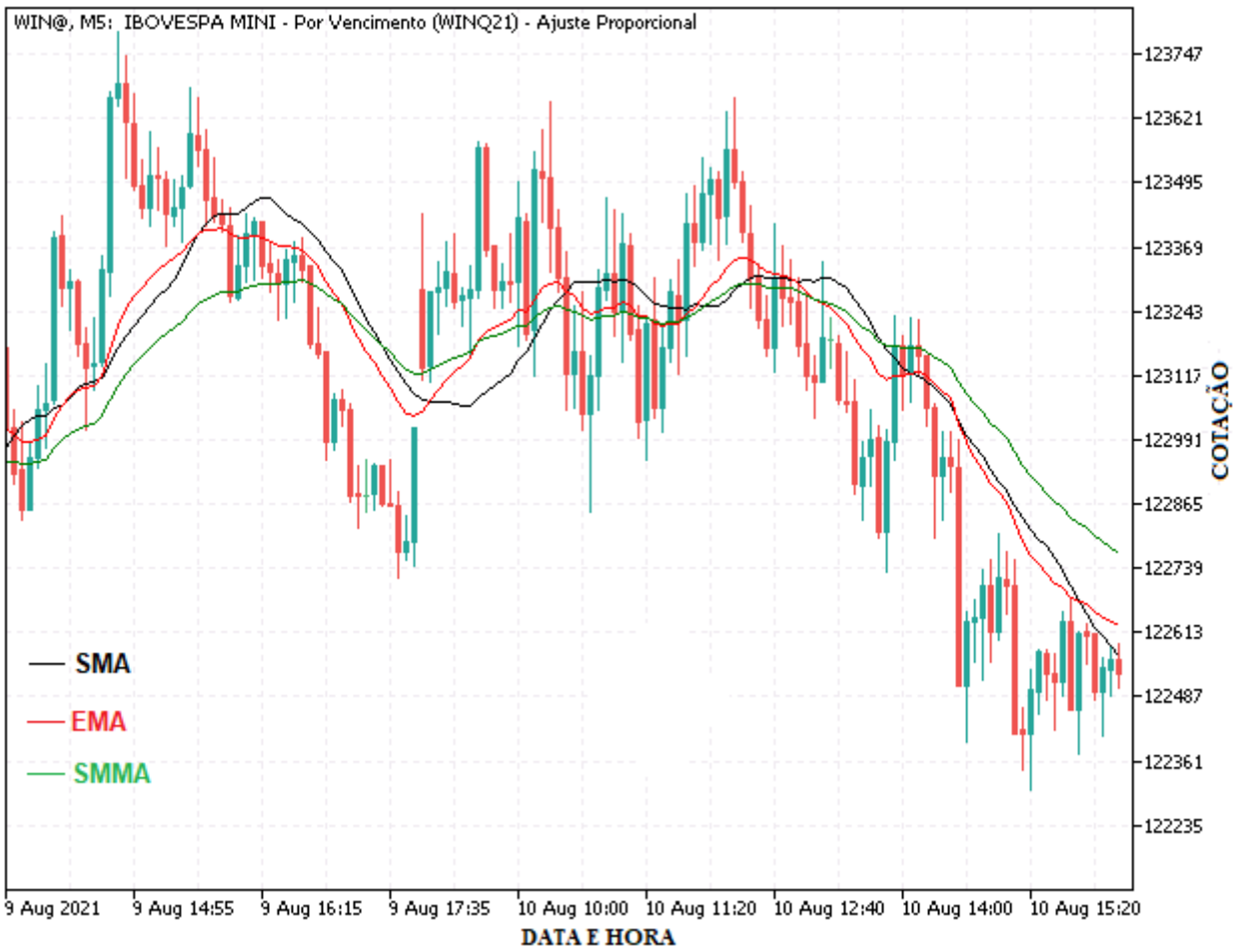


Fonte: do autor.

2.4.1.3 Média Móvel Suavizada

A SMMA (*Smoothed Moving Average*) é uma média móvel exponencial que leva em consideração toda as série de dados disponíveis. Assim, os dados de preço mais antigos na média móvel suavizada nunca são removidos, mas têm apenas um impacto mínimo na média móvel (RAUDYS; LENČIAUSKAS; MALČIUS, 2013).

Figura 2.5 – Média Móvel suavizada



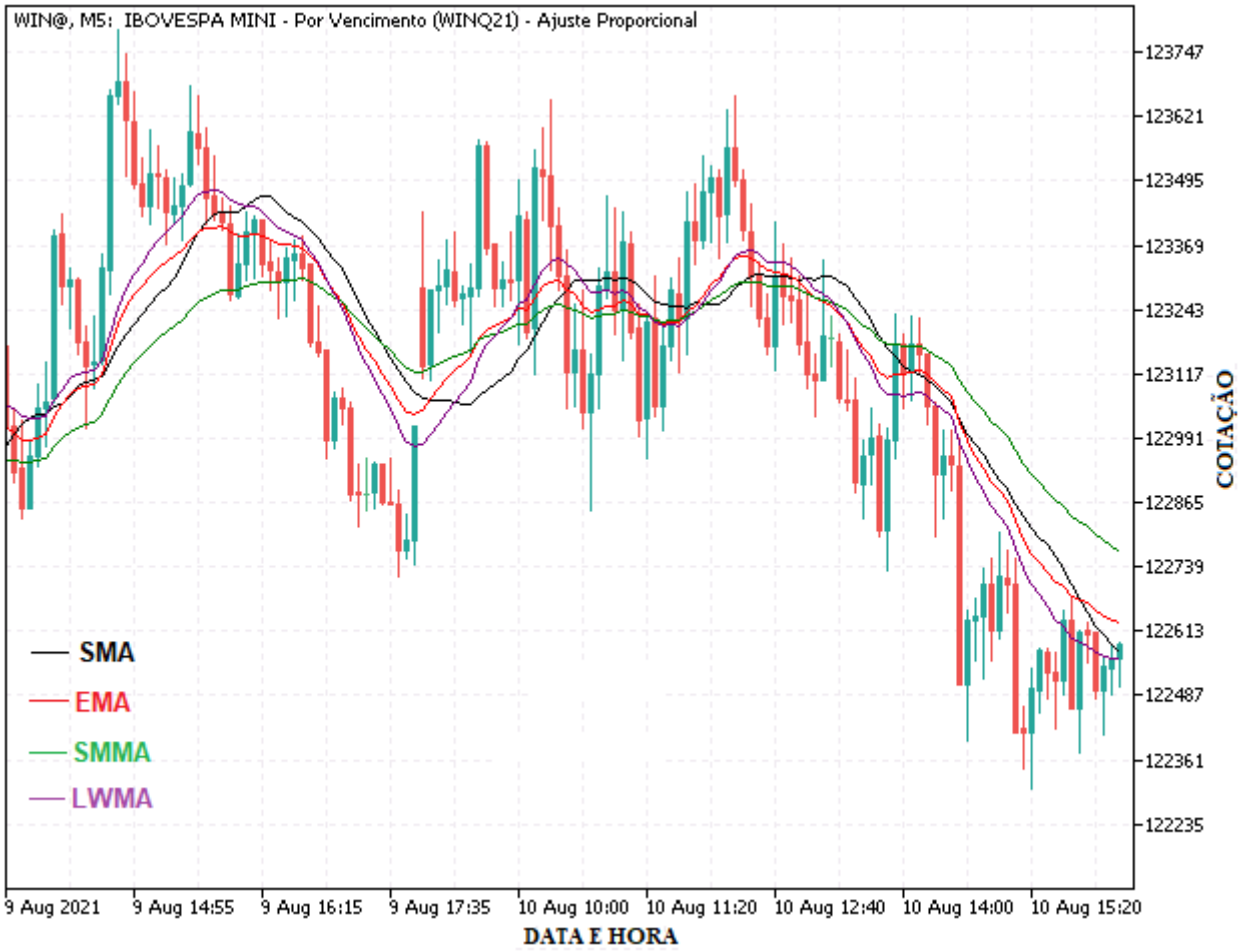
Fonte: do autor.

2.4.1.4 Média Móvel Linearmente Suavizada

A LWMA (*Linearly Weighted Moving Average*) tem o método de cálculo muito similar a EMA, a principal diferença ocorre no decaimento linear da ponderação, ou seja, o preço mais recente tem o peso mais alto e cada preço anterior os pesos decrescem linearmente (RAUDYS; LENČIAUSKAS; MALČIUS, 2013).

$$LWMA = \frac{\sum_{i=0}^{n} p_i w_i}{\sum_{i=0}^{n} w_i} \tag{2.4}$$

Figura 2.6 – Média Móvel Linearmente Suavizada



Fonte: do autor.

2.4.2 Bandas de *Bollinger*

O indicador Bandas de Bollinger foi criado no início da década de 80 pelo analista financeiro John A. Bollinger, naquela época John utilizava um indicador de bandas fixas conhecida como envelopes, conforme ilustrado na Fig.2.7. O envelope basicamente é um canal em torno de uma MM, formado pelas bandas superior e inferior do canal (BOLLINGER, 1992). O posicionamento dessas bandas é determinado por uma coeficiente percentual de distância fixa da MM, conforme as equações abaixo:

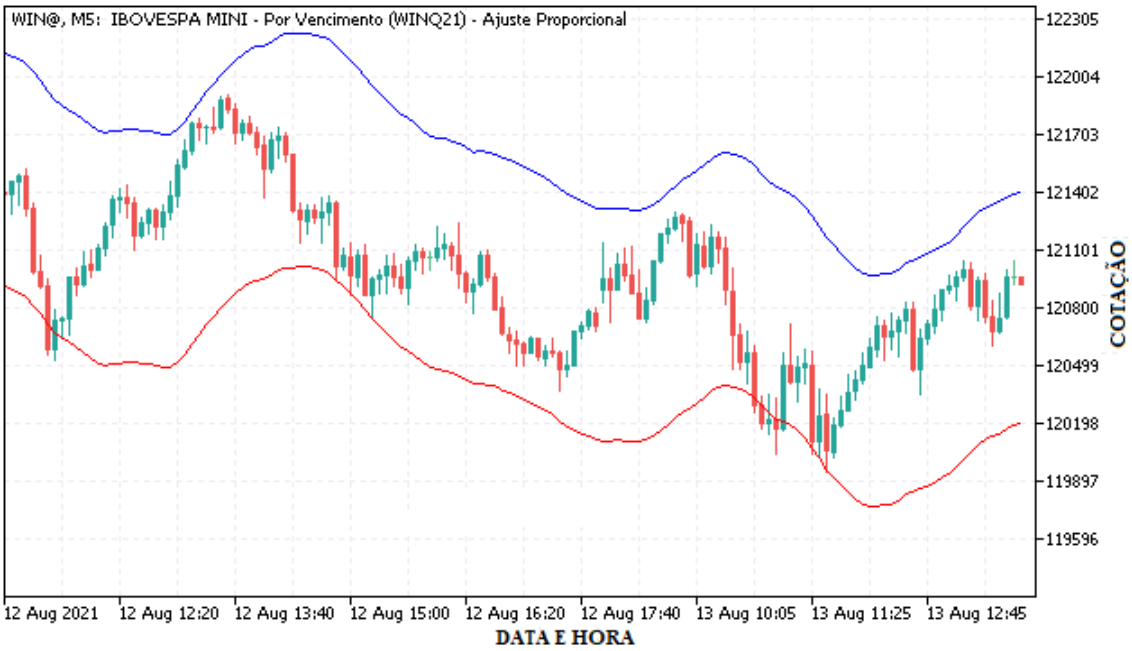
$$BS = SMA_n \times (1 + \alpha) \tag{2.5}$$

$$BI = SMA_n \times (1 - \alpha) \tag{2.6}$$

onde:

- **BS** é a banda superior do envelope;
- **BI** é a banda inferior do envelope;
- **SMA** é a média móvel simples;
- **n** é o período da média móvel;
- α é o coeficiente percentual de distância das bandas;

Figura 2.7 – Indicador Envelopes



Fonte: do autor.

Ao verificar que suas análises dependiam da volatilidade do ativo, John Bollinger introduziu um novo conceito no cálculo dos envelopes, a contribuição dada por ele foi a adição de um elemento de volatilidade aos envelopes (BOLLINGER, 1992; MURPHY, 1999). Agora, para o cálculo das Bandas de Bollinger é considerado o desvio padrão, conforme equações a seguir:

$$BBS = SMA_n + (\sigma_n \times k) \tag{2.7}$$

$$BBI = SMA_n - (\sigma_n \times k) \tag{2.8}$$

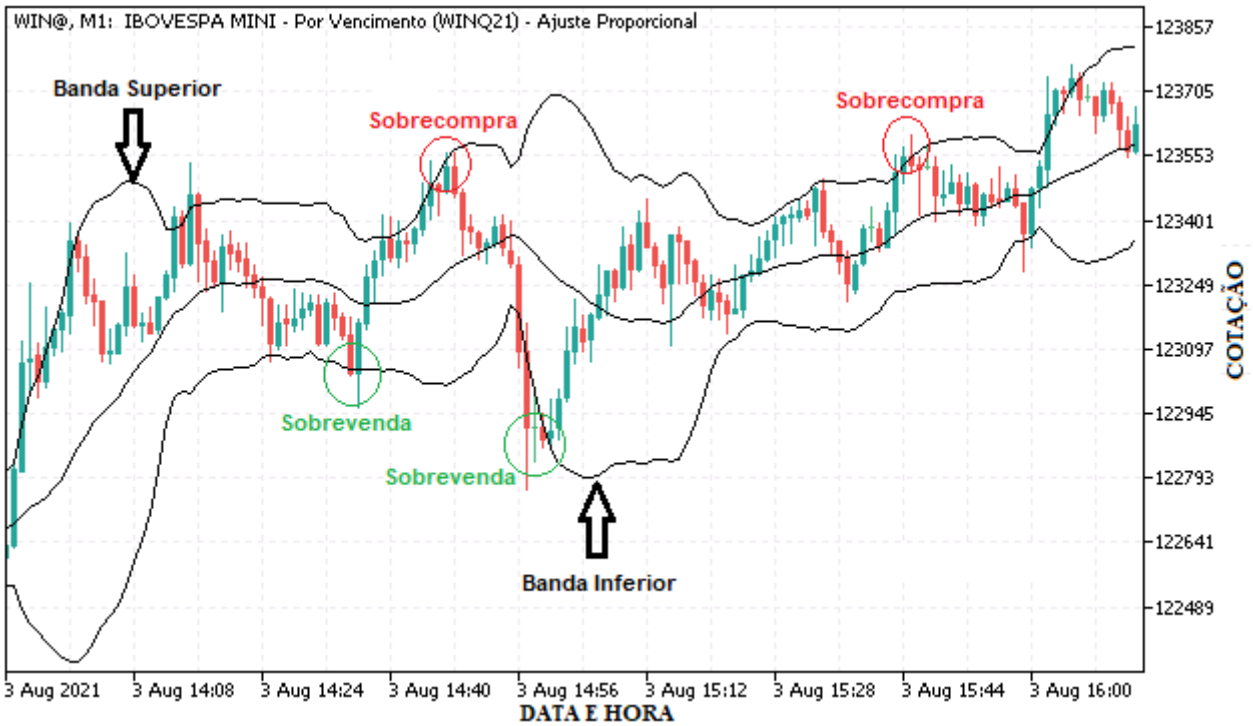
onde:

- **BBS** é a Banda de Bollinger superior;
- **BBI** é a Banda de Bollinger inferior;
- **SMA** é a média móvel simples;
- **n** é o período da média móvel;
- σ é o desvio padrão;
- **k** é uma contante que define a distância das bandas em relação a média móvel;

A literatura define como período padrão n=20 e k=2, no entanto, esses valores podem ser ajustados conforme a necessidade (BOLLINGER, 1992).

Conforme ilustrado na Fig 2.8 as Bandas de Bollinger indicam cenários de mercado sobrecomprado e sobrevendido, e identificam automaticamente os períodos de alta e baixa volatilidade. Quando a volatilidade é alta as bandas se alargam e quando a volatilidade é baixa as bandas se contraem.

Figura 2.8 – Indicador Bandas de Bollinger



Fonte: do autor.

2.4.3 Filtro *Savitzky – Golay*

Em 1964, Abraham Savitzky e Marcel J.E. Golay publicaram o artigo *"Smoothing and Differentiation of Data by Simplified Least Squares Procedures"*, na revista científica *Analytical Chemistry*, o artigo é um dos mais citados segundo os editores da revista, sendo ele o número cinco entre os dez principais artigos já publicados nesse jornal (Savitzky, A.; Golay, 1964; SCHAFER, 2011).

A contribuição do trabalho de Savitzky, A.; Golay (1964) consiste em uma abordagem matemática para suavizar séries temporais com base em uma regressão polinomial. Os filtros de Savitzky-Golay (SG) são filtros digitais que realizam a suavização de séries temporais pela substituição de cada valor da série por um novo valor que é obtido a partir de um ajuste polinomial para uma janela de $2n+1$ pontos vizinhos (incluindo o ponto a ser suavizado), com n sendo igual ou maior que a ordem do polinômio. O ajuste polinomial móvel é numericamente manipulado por mínimos quadrados e pela convolução dos dados de entrada pela janela deslizante $(2n+1)$, determinando o valor suavizado do ponto central do conjunto de dados através de uma regressão polinomial (PRESS; TEUKOLSKY, 1990; SCHAFER, 2011).

A Fig.2.9 ilustra de forma exemplificada a metodologia de suavização de Savitzky-Golay aplicado a uma janela móvel de comprimento $n = 2$.

Similar método de calculo da média móvel ponderada, o sinal suavizado é obtido pela equação 2.9, onde os coeficientes c_J são obtidos pelo ajuste linear não-ponderado de mínimos quadrados (ORFANIDIS, 1995).

$$x_i = \frac{1}{2n+1} \sum_{j=-n}^{n} c_j x_{(i+j)} \tag{2.9}$$

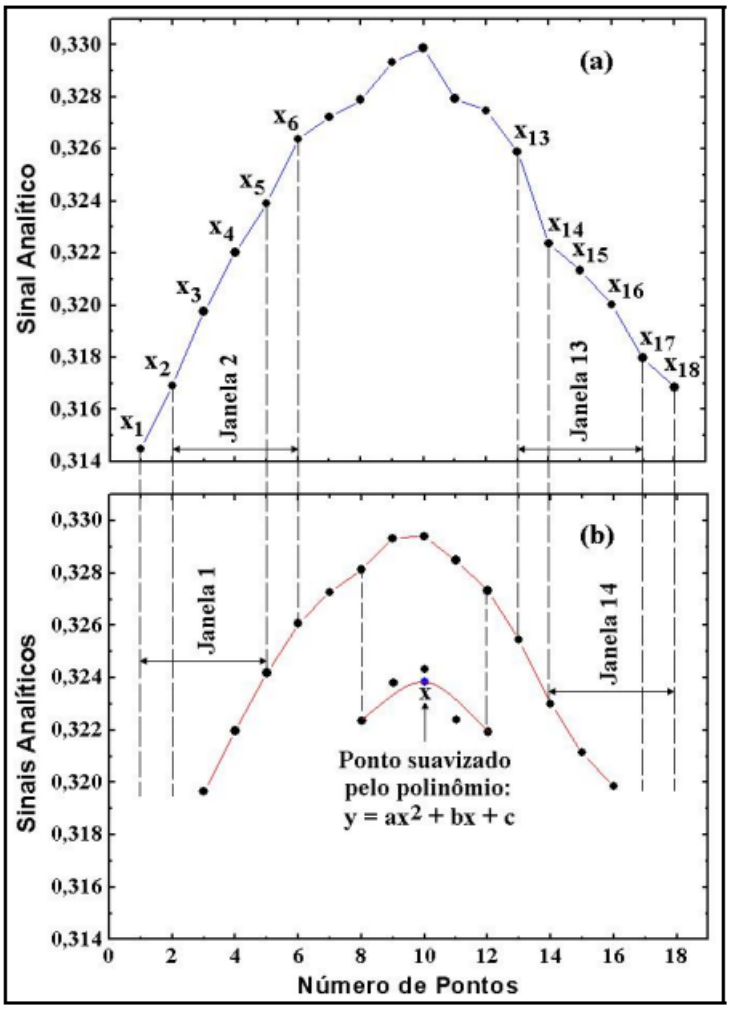
Onde:

- x_i é o resultado da suavização;
- n é o número de pontos a direita e a esquerda do ponto central da janela de suavização;
- c_j é o coeficiênte para a j-ésima suavização;
- $x_{(i+j)}$ são os pontos da janela móvel;

O livro de Orfanidis (1995) *"Introduction to Signal Processing*[3]" disponíıvel gratuitamente apresenta de forma detalhada o procedimento de cálculo dos filtros de Savitzky-Golay.

[3] http://www.ece.rutgers.edu/ orfanidi/intro2sp/

Figura 2.9 – Suavização de sinal pelo método de Savitzky-Golay.(a) Sinal original e (b) Sinal filtrado.



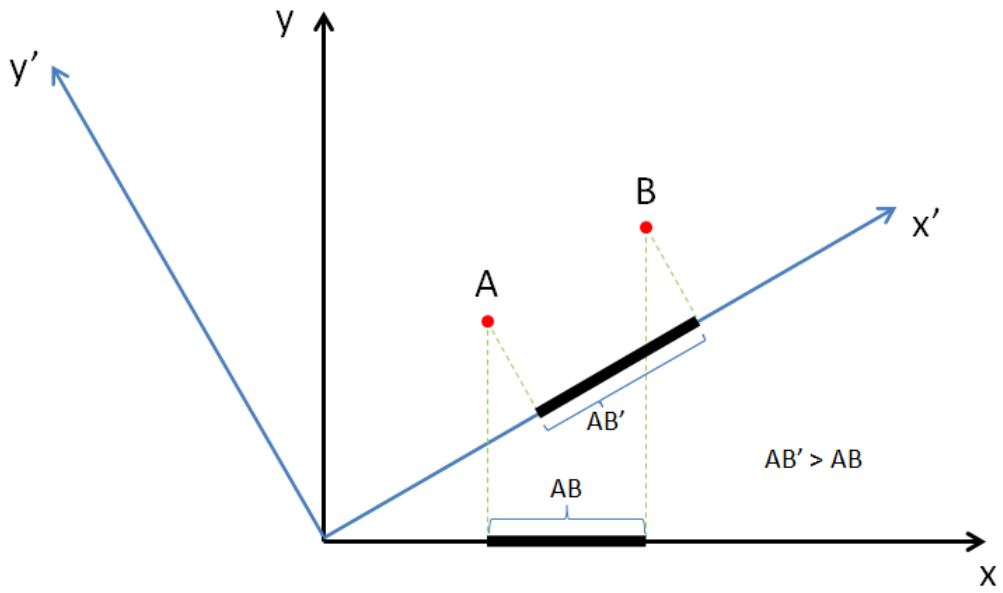
Fonte: (MARTINS, 2005).

2.4.4 Análise de Componentes Principais

A análise de Componentes Principais (PCA) é um método estatístico de análise multivariada, que tem como objetivo principal reduzir a dimensionalidade de um conjunto de dados mantendo o máximo possível da variação dos dados originais. Em outras palavras, o PCA extrai as informações importantes (Principais Componentes), eliminando informações redundantes presente em um grande conjunto de dados multivariados, por meio de uma transformação linear ortogonal. Os principais componentes extraídos dos dados originais conservam a maior variabilidade dos dados e não apresentam correlação entre si. A redução de dimensionalidade é dada pela seleção dos componentes principais que detêm a maior variância dos dados, ou seja, contém a maior parte da informação original dos dados (JOLLIFFE, 2002; JOHNSON RICHARD A., 2007).

Do ponto de vista geométrico, a transformação linear realizada pelo PCA representa a seleção de um novo sistema de coordenadas o qual obtém-se o máximo de variabilidade dos dados (JOHNSON RICHARD A., 2007). A Fig. 2.10 ilustra de forma simplificada o processo de rotação do sistema de coordenadas obtido pela transformação linear. A projeção dos pontos AB no sistema de coordenadas original $x0y$ é menor que a projeção no sistema de coordenadas $x'0y'$, o que representa maior variabilidade dos dados no novo sistema de coordenadas.

Figura 2.10 – Rotação de eixos efetuada pela transformação linear do PCA



Fonte: do autor.

Em síntese, o método matematicamente para obtenção dos componentes principais serão discutidos nas subsecções a seguir. Os livros de Jolliffe (2002) e Johnson Richard A. (2007) apresentam com maior detalhamento e rigor matemático o desenvolvimento da PCA.

2.4.4.1 Preparação dos dados

A primeira etapa para preparação dos dados, consiste em organizá-los em uma matriz $m \times n$, onde n é o número de amostras e m é a quantidade de *features* ou atributos. Quanto maior o valor de m maior é a dimensionalidade dos dados.

$$X = \begin{bmatrix} x_{11}x_{12}\cdots x_{1m} \\ x_{21}x_{22}\cdots x_{2m} \\ \vdots \\ x_{n1}x_{2n}\cdots x_{nm} \end{bmatrix} \tag{2.10}$$

Coso exista diferença de escala entre os atributos, faz-se necessário normalizar ou padronizar os dados a fim de evitar que o algoritmo fique enviesado para as variáveis com maior ordem de grandeza (JOLLIFFE, 2002; JOHNSON RICHARD A., 2007).

A normalização consiste no reescalonamento dos dados para um intervalo comum, por exemplo entre 0 e 1, trazendo assim, todas as *features* para uma escala comum (JOHNSON RICHARD A., 2007). A normalização é calculada pela equação 2.11.

$$X_{normalized} = \frac{x - X_{min}}{X_{max} - X_{min}} \tag{2.11}$$

A padronização redimensiona as *features* de forma a obter média igual a 0 (zero) e desvio padrão igual a 1(um). O cálculo de padronização é feito pela equação 2.12, subtrai-se de cada variável a média e dividindo pelo desvio padrão para centralizar a distribuição dos dados (JOHNSON RICHARD A., 2007).

$$Z = \frac{x - \bar{X}}{\sigma} \tag{2.12}$$

A depender dos dados o qual se pretende aplicar o PCA, normalizar ou padronizar é uma escolha que deve ser melhor avaliada por teste e validação de ambos os métodos (BROWNLEE, 2020).

2.4.4.2 Matriz de Covariância

Em matemática e estatística a covariância é uma métrica que compara dois grupos de dados e avalia o quão relacionados eles são. Entende-se como uma medida de variância entre duas variáveis, no entanto, não avalia a dependência entre as variáveis. A equação 2.13 é usada para o cálculo de covariância entre dados de duas dimensões (JOLLIFFE, 2002; JOHNSON RICHARD A., 2007).

$$cov(X,Y) = \frac{1}{n}\sum_{i=1}^{n}[(X_i - \bar{X})(Y_i - \bar{Y})] \tag{2.13}$$

Onde:

- X é a primeira dimensão dos dados;
- Y é a segunda dimensão dos dados;
- $\bar{X}$ é a média do vetor X;

- $\bar{Y}$ é a média do vetor Y;
- n é o número de amostras dos dados.

Para obter a matriz de covariância é realizado o cálculo de covariância entre cada par de atributos (dimensão ou *feature*) do conjunto de dados. A equação 2.14 apresenta a matriz de covariância obtida para um conjunto de dados tridimensional (XYZ) (JOLLIFFE, 2002; JOHNSON RICHARD A., 2007).

$$cov(X,Y,Z) = \begin{bmatrix} cov(X,Y) & cov(X,Y) & cov(X,Z) \\ cov(Y,X) & cov(Y,Y) & cov(Y,Z) \\ cov(Z,X) & cov(Z,Y) & cov(Z,Z) \end{bmatrix} \tag{2.14}$$

A diagonal principal da matriz contém covariância entre um elemento e ele mesmo, que é a medida de variância. Além disso a matriz é simétrica, de modo que é sempre possível encontrar um conjunto de autovetores ortonormais (JOLLIFFE, 2002; JOHNSON RICHARD A., 2007).

2.4.4.3 Autovetores e Autovalores

De posse da matriz de covariância é possível obter os respectivos autovalores e autovetores. Os autovetores da matriz de covariância determinam as direções do novo espaço de recursos e os autovalores classificam em magnitude a variabilidade dos dados ao longo dos novos eixos. Classificando os autovetores em ordem decrescente de seus autovalores, obtém-se os componentes principais em ordem de significância. Uma propriedade importante dos autovetores é a ortogonalidade entre eles, essa propriedade permite expressar os dados em um novo sistema de coordenadas (JOLLIFFE, 2002; JOHNSON RICHARD A., 2007).

Considere uma matriz $A_{n\times m}$ e a seguinte equação vetorial:

$$Ax = \lambda x \tag{2.15}$$

Onde:

- **A** é uma matriz $n \times m$;
- λ é o auto valor característico da matriz A;
- x é o autovetor não nulo associado ao autovalor λ.

Manipulando a equação 2.15 obtemos:

$$(A - \lambda I)x = 0 \tag{2.16}$$

Onde I é a matriz identidade.

Os autovalores λ podem ser calculados pela solução para equação 2.16, obtida pelo cálculo do determinante característico, o qual obtém-se a equação característica da matriz A, conforme a equação abaixo:

$$det(A - \lambda I) = 0 \tag{2.17}$$

Obtidos os autovalores λ, os respectivos autovetores associados são os vetores não nulos obtidos da solução da equação 2.15.

Resumidamente, a PCA consiste em uma mudança de base o qual os autovetores da matriz de covariância representa o novo sistema de coordenadas dos dados.

2.4.4.4 Componentes Principais

A somatória dos autovalores obtidos de uma matriz de correlação representa a variância total dos dados. Neste sentido, defini-se a proporção da variância original (P_v) dos dados o qual pretende-se manter na seleção dos componentes principais. A literatura adota na maioria dos casos o percentual mínimo de 80% da variação total para seleção dos componentes principais (JOLLIFFE, 2002; JOHNSON RICHARD A., 2007).

A proporção da variação de cada componente principal é calculada por:

$$\frac{\lambda_k}{\sum_{i=1}^{n} \lambda_i} \tag{2.18}$$

Onde:

- **k** é o número da componente principal selecionada (k < n);
- **n** é o número total de componentes (n = dimensão dos dados originais);

Para manter a proporção mínima da variação original dos dados (P_v) , faz-se necessário escolher as componentes tais que:

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \geq P_v \tag{2.19}$$

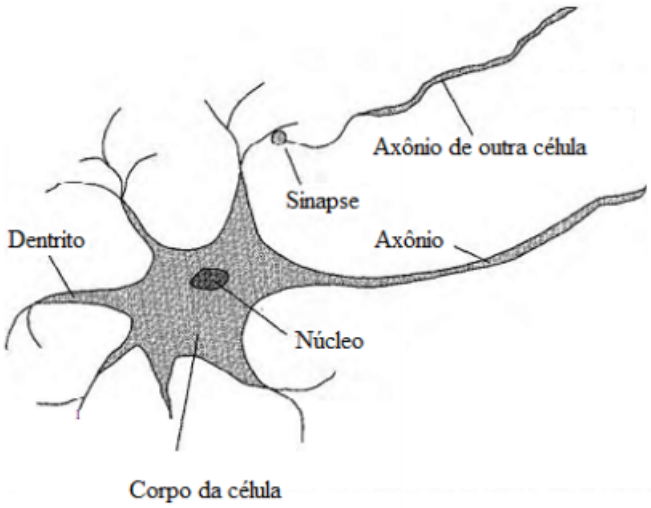
2.5 Redes Neurais Artificiais

A inteligência biológica proveniente do cérebro humano é resultado da perfeita e complexa estrutura neurológica, o qual apresenta como unidade básica os neurônios conectados em redes. A ciência observando a rede neural biológica, propôs copiar artificialmente a estrutura e funcionamento do cérebro humano por meio de mecanismos conhecidos como Redes Neurais Artificiais (RNA). As RNA's podem ser consideradas também como uma metodologia para resolver problemas característicos da inteligência artificial.

2.5.1 Fundamentos

RNA's são modelos matemáticos de algoritmos de *Machine Learning* inspirados em uma estrutura neural biológica (Fig. 2.11). As partes principais que o compõe são: o corpo da célula, os dentritos e o axônio. Os dentritos são responsáveis por receber os impulsos nervosos provenientes de outros neurônios e os transmitem até o corpo celular, este por sua vez é responsável pelo processamento de todas as informações recebidas, resultando em um novo impulso nervoso. Este impulso passa então pelo axônio do neurônio até os dentritos de outros neurônios. A sinapse é o ponto de contato entre a terminação do axônio de um neurônio e o dendrito de outro neurônio. Basicamente este sistema simples, somado à operação em paralelo de bilhões de neurônios, é o responsável pela maioria das funções executadas pelo nosso cérebro (BRAGA; FERREIRA; LUDERMIR, 2007; HAYKIN, 2007).

Figura 2.11 – Neurônio biológico

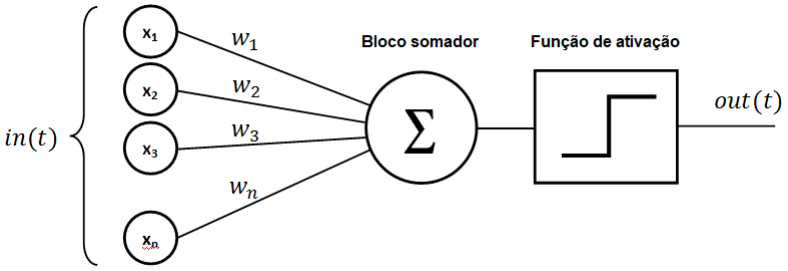


Fonte: Adaptado de (RUSSELL; NORVIG, 2002)

Baseado nas características do neurônio biológico, o neurônio artificial é um modelo matemático bioinspirado que foi introduzido pelo neurofisiologista Warren McCulloch e o matemático Walter Pitts em 1943 no artigo (MCCULLOCH; PITTS, 1990). Mais tarde em 1958 Frank Rosenblatt (ROSENBLATT, 1958) inspirado por McCulloch e Pitss, desenvolveu o mais simples tipo de neurônio das arquiteturas de RNAs, o perceptron. A Fig. 2.12 apresenta o diagrama de um perceptron onde:

- x_j é um sinal de entrada do neurônio;
- w_j é o "peso sináptico", um número real que representa a importância daquele sinal;
- por fim, tem-se um bloco somador e uma função de ativação de saída.

Figura 2.12 – Perceptron de Rosemblatt



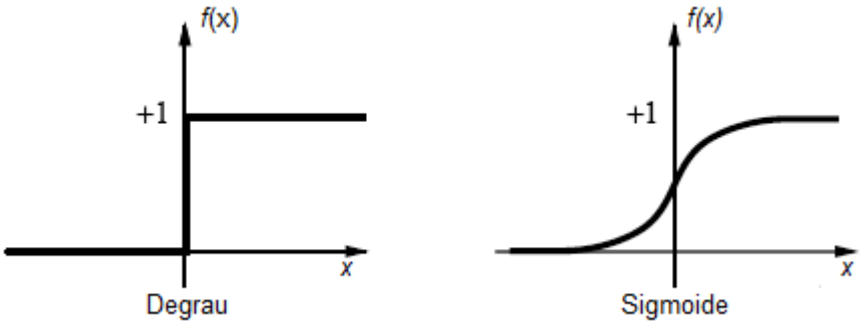
Fonte: Adaptado de bit.ly/3aHO3cw. Acessado em Fev/2021

Os neurônios artificiais são conectados entre diferentes camadas com conexões com ajustes em pesos e polarização para treinar a rede. O padrão de interconexão entre as camadas, a função de ativação dos neurônios e processo de aprendizagem definem o tipo de RNA. Durante o treinamento da RNA, os pesos são ajustados até que a meta e a saída sejam combinadas (BRAGA; FERREIRA; LUDERMIR, 2007; HAYKIN, 2007).

2.5.2 Funções de ativação

As funções de ativação basicamente decidem se um neurônio deve ser ativado ou não. Elas restringem a amplitude de saída de determinado neurônio adicionando não-linearidade ao modelo. A Fig.2.13 apresenta as funções de ativação mais comumente usadas, sendo elas a degrau e a sigmoidal.

Figura 2.13 – Principais funções de ativação



Fonte: Adaptado de bit.ly/3p9wmb8. Acessado em Fev/2021

Além destas, há diversas funções de ativação diferentes que podem ser utilizadas, a depender da aplicação, tipo de rede neural e o problema a ser resolvido. A título de exemplo tem-se as funções: limiar; linear, semi-linear, linear retificada (RELU), linear exponencial (ELU), tangente hiperbólica, entre outras.

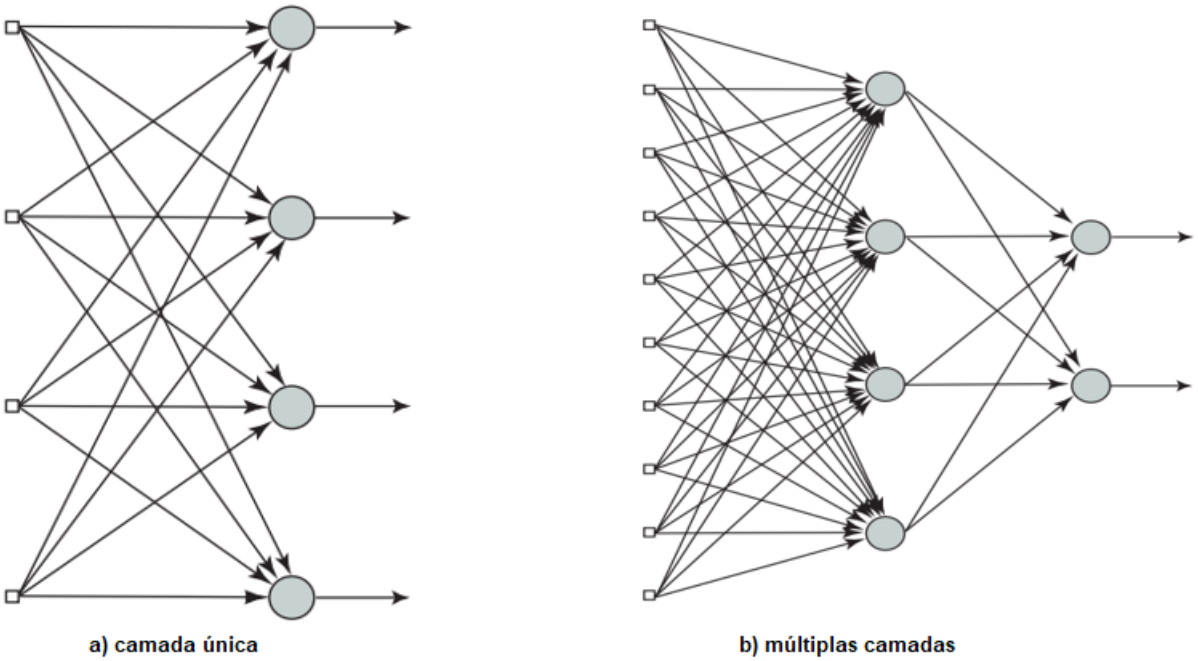
2.5.3 Topologia das RNA's

O potencial e flexibilidade do cálculo baseado em redes neurais vêm da criação de conjuntos de neurônios que estão interligados entre si. Esse paralelismo de elementos com processamento local cria a "inteligência" global da rede. Em termos de arquitetura ou topologia, uma RNA é definida pelo número de nós na camada de entrada, que corresponde ao número de variáveis de entrada da rede, sendo normalmente as variáveis de maior importância para o problema em estudo, o número de camadas escondidas e número de neurônios a serem inseridos nessas camadas, o número de neurônios na camada de saída e o tipo de conexão entre eles (KOTHARI; OH, 1993; BRAGA; FERREIRA; LUDERMIR, 2007; HAYKIN, 2007).

Com relação ao número de camadas a RNA pode ser definida como sendo de:

- camada única (ou simples) - a camada de entrada é diretamente associada a um ou mais neurônios de saída (Fig.2.14-a).
- múltiplas camadas - entre a camada de entrada e a de saída existe uma ou mais camadas escondidas de neurônios (Fig.2.14-b).

Figura 2.14 – Topologia das Redes Neurais



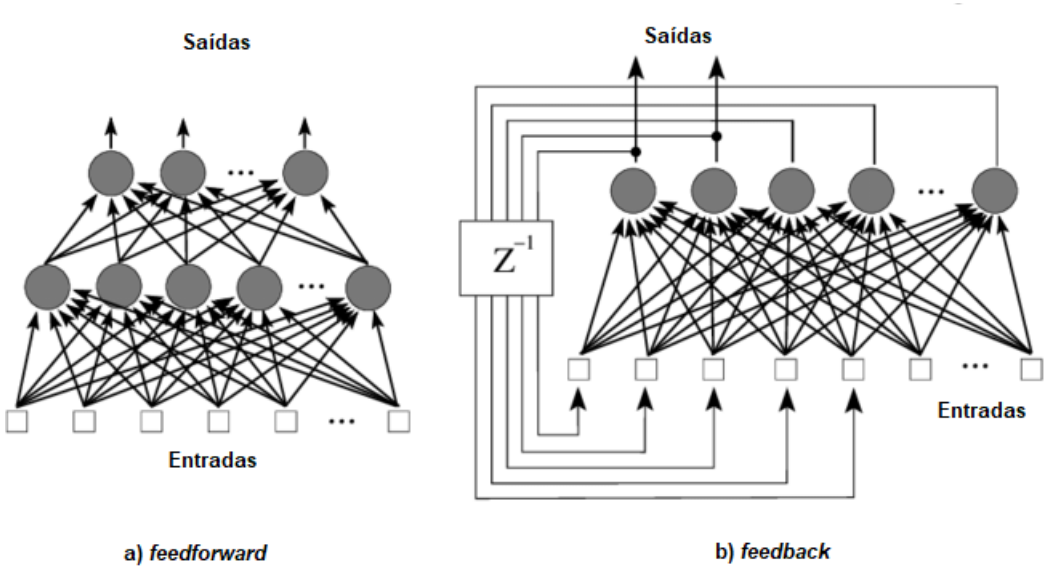
Fonte:Adaptado de bit.ly/3wiZ9yH. Acessado em Mar/2021

Outra categorização fundamental da topologia de uma RNA está relacionado ao tipo de conexão e o método de propagação das informações ao longo da rede. Sendo assim, tem-se as redes de propagação para frente (*feedforward* (Fig. 2.15-a)), o qual o fluxo de dados segue sempre em direção à camada de saída, ou seja, unidirecional. Existem ainda, as redes realimentadas (*recurrent* (Fig. 2.15-b)) que apresentam ao menos um ciclo de realimentação com atraso de tempo ($Z^{-}1$), onde as saídas são realimentadas como sinais de entrada para outros neurônios, geralmente empregadas para o processamento de sistemas variantes no tempo (HAYKIN, 2007; BISHOP et al., 1995) .

Relativo ao tipo de conexão entre as camadas e neurônios de uma rede neural, classifica-se a conectividade como:

- fracamente conectada - quando o neurônio de uma camada não está ligada a todos os neurônios da camada seguinte.
- completamente conectada - quando cada neurônio de uma camada está ligado a todos os neurônios da camada seguinte.

Figura 2.15 – Método de propagação em redes neurais



Fonte: Adaptado de (GOMEZ; MIIKKULAINEN, 2002)

2.5.4 Aprendizado

O aprendizado de uma rede neural está relacionado ao processo de treinamento pelo qual ela é submetida. Neste sentido, os parâmetros livres, pesos sinápticos e *bias*, são modificados de maneira iterativa e otimizada. Algoritmo de aprendizado é o nome dado para as instruções e procedimentos do processo de aprendizagem da RNA, que por sua vês é classificado como (MENDEL; MCLAREN, 1970; BISHOP et al., 1995):

- aprendizado supervisionado - a rede neural recebe os dados de entrada e saídas desejadas, a cada iteração do algoritmo de treinamento é feita uma correção dos parâmetros livres de modo a minimizar o erro entre a saída desejada e a saída fornecida pela RNA.

- aprendizado não supervisionado - somente os dados de entrada estão disponíveis para rede, assim a cada iteração o algoritmo analisa padrões, regularidades e correlações para agrupar os conjuntos de dados em classes ou *clusters*.

Existe uma vasta gama de algoritmos de aprendizagem de redes neurais na literatura, onde cada um apresenta uma vantagem específica. A diferença principal entre estes algoritmos está na formulação do ajuste dos pesos sinápticos dos neurônios. É possível citar cinco regras básicas de aprendizagem, sendo elas (HAYKIN, 2007):

- aprendizado por correção de erros;

- aprendizagem baseada em memória;
- aprendizagem hebbiana;
- aprendizagem competitiva;
- aprendizagem de Boltzmann;

A regra de aprendizado por correção de erros é o principal algoritmo de treinamento utilizado, por esse motivo será dado foco principal na explicação deste algoritmo no subcapítulo que se segue.

2.5.4.1 Aprendizado por correção de erros

Em suma, o processo de aprendizagem por correção de erros, também conhecida como regra delta ou regra de *Widrow-Hoff*, busca minimizar o erro entre resposta gerada pela RNA e a saída desejada conforme equação.2.20:

$$e(t) = d(t) - y(t) \tag{2.20}$$

Onde:

- e(t) é o sinal de erro;
- d(t) é a saída desejada;
- y(t) é a saída real da rede.

O objetivo é minimizar o valor do erro *e(t)*, para isso, a cada iteração do algoritmo de treinamento os pesos devem ser ajustados de forma a obter um erro menor. Tem-se então um problema de otimização da função de custo ou índice de desempenho dada pelo valor instantâneo da energia do erro (equação 2.21) (BISHOP et al., 1995; HAYKIN, 2007).

$$\varepsilon(n) = \frac{1}{2} e_k^2(n) \tag{2.21}$$

De acordo com a regra delta, o ajuste do peso sináptico $\omega_{kj}(n)$ do neurônio k, excitado pela entrada $x_j(n)$ na iteração n, é definido por:

$$\Delta\omega_{kj}(n) = \eta e_k(n) x_j(n) \tag{2.22}$$

Onde:

- $\Delta\omega_{kj}(n)$ é o ajuste sináptico;
- η é uma constante positiva denominada como taxa de aprendizado que controla o tamanho do passo na atualização dos pesos.

De posse do valor de ajuste sináptico $\Delta\omega_{kj}(n)$, o valor atualizado do peso sináptico $\omega_{kj}(n)$ é dado por:

$$\omega_{kj}(n+1) = \omega_{kj}(n) + \Delta\omega_{kj} \tag{2.23}$$

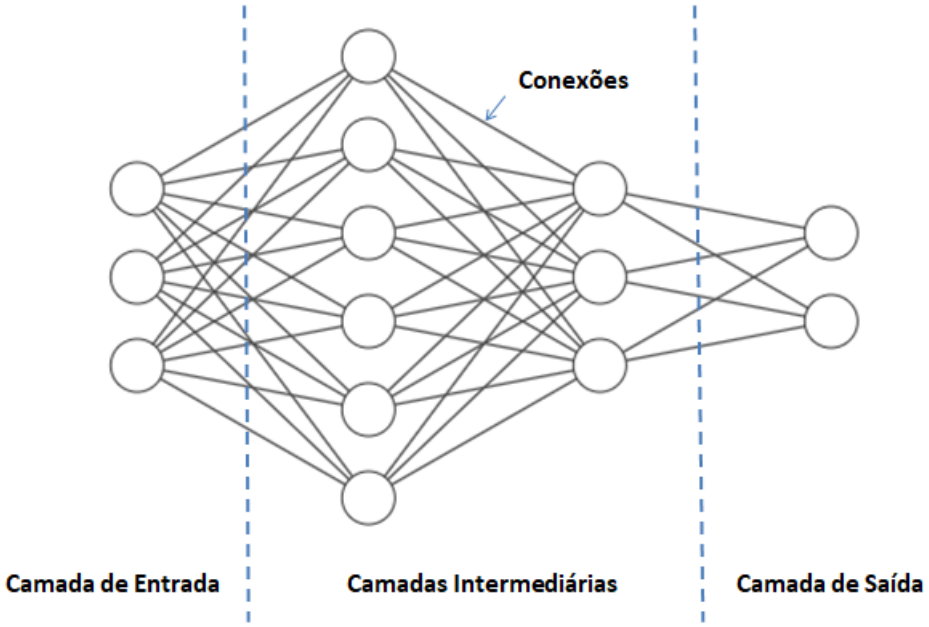
A aplicação de um algoritmo de aprendizagem baseado no critério de correção do erro, requer a definição de um conjunto de parâmetros, tais como: número de iterações, critério de parada, pesos iniciais, taxa de aprendizado (n). A escolha adequada destes parâmetros influência de forma decisiva a capacidade de generalização da rede (BISHOP et al., 1995; HAYKIN, 2007).

2.5.5 Redes *Multilayer Perceptron* – MLP

No final da década de 50, Minsky e Papert publicaram uma obra chamada "Perceptron", nela eles demostraram que o modelo de Rosenblatt, as redes de camada única, não eram capazes de solucionar problemas não linearmente separáveis, o que resultou naquela época no esfriamento das pesquisas e financiamentos na área. Anos mais tarde, especificamente em 1986, Rumelhart, Hinton e William desenvolveram o algoritmo de treinamento *backpropagation*, uma generalização da regra delta. Com isso, conseguiram treinar de forma eficiente redes neurais com camadas intermediárias, resultando no modelo de Redes Neurais Artificiais mais utilizado atualmente, as redes Perceptron Multi-Camadas (MLP). Esta por sua vez, resolve o problema apresentado por Minsky e Papert, pois apresenta um poder computacional maior que as redes de camada única permitindo então a solução de problemas não-linearmente separáveis. (OLAZARAN, 1996).

Uma rede MLP consiste de uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída, com método de propagação *feedforward* e completamente conectada, de forma que cada neurônio da camada anterior fornece sua saída para cada neurônio da camada seguinte, conforme ilustrado na Fig. 2.16.

Figura 2.16 – Exemplo de rede neural MPL



Fonte: do autor

O algoritmo *backpropagation* é o principal algoritmo de treinamento de redes MLP, trata-se de um algoritmo de aprendizado supervisionado cujo treinamento ocorre em duas fases:

- *forward* - dados de entrada são apresentados à camada de entrada da rede e o deslocamento através da rede ocorre camada por camada até a saída. Na saída é feito o cálculo do erro.

- *backward* - nesta fase ocorre o deslocamento no sentido contrário, o qual o sinal de erro é propagado para cada neurônio das camadas anteriores e as mudanças nos pesos são efetuadas de modo que a cada iteração do algoritmo o erro convirja para um valor mínimo possível.

O *backpropagation* apresenta algumas desvantagens, tais como o elevado tempo de treinamento, alto número de iterações e possibilidade de parada em mínimos locais na superfície de erro. Visando minimizar os efeitos destes problemas, foi introduzido um termo chamado *"momentun"* à equação 2.23. Trata-se de um artifício matemático que pondera o efeito das mudanças anteriores dos pesos na direção atual do movimento no espaço de pesos. O *momentun* possibilitou diminuir o tempo de treinamento da RNA e gera maior estabilidade na convergência evitando que a mesma fique presa em mínimos locais. A equação 2.24 apresenta o termo *"momentun"* que fora adicionado na equação 2.23, o qual a constante β (constante de momento) é normalmente ajustada entre 0.5 e 0.9.

$$\omega_{kj}(n+1) = \omega_{kj}(n) + \Delta\omega_{kj} + \beta[\omega_{kj}(n) + \omega_{kj}(n-1)] \tag{2.24}$$

2.6 *Outliers*

Outlier é uma amostra dos dados que se desvia de forma anormal em relação ao restante dos dados (PHAM, 2006). Os termos mais comumente usados na literatura para definir os *outliers* são:

- valor discrepante;
- valor atípico;
- ponto fora da curva;
- anomalia;
- entre outros.

Os *outlier*, as vezes são erros, dados inválidos ou uma anomalia indesejável para a análise dos dados. No entanto, em outras ocasiões, pode revelar percepções sobre casos especiais do conjunto de dados que, de outra forma, não seria possível notar. Pham (2006) cita que detectar essas observações é importante porque elas podem levar a novas descobertas. Didaticamente ele cita o exemplo da descoberta da penicilina pelo médico e bacteriologista escocês Alexander Fleming, em 1928. Fleming ao invés de ignorar um *outlier*, tentou entender o motivo do efeito atípico de um procedimento acidental em seu material de estudo. Atualmente a penicilina, também conhecida como amoxicilina, é o antibiótico mais amplamente utilizado no tratamento de doenças bacterianas.

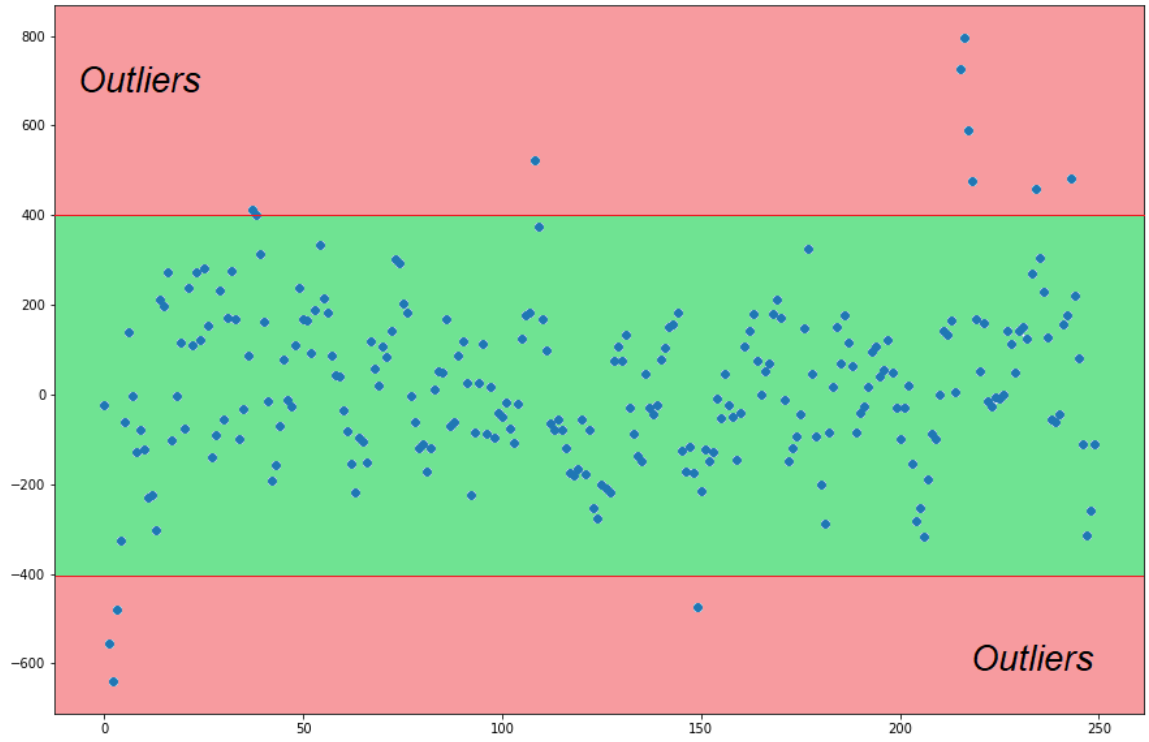
Neste contexto, é de extrema importância identificar e observar os *outliers* na análise dos dados pois:

1. os *outliers* aumentam a variabilidade dos dados, o que pode viesar negativamente todo o resultado da análise. Isso significa que os valores identificados como *outliers* devem ser removidos ou corrigidos;
2. os *outliers* podem revelar percepções sobre casos especiais, carregando informações sobre a área de estudo que dependendo da análise pode ser justamente o padrão que está

sendo procurado. Neste sentido, é essencial entender como ocorrem os *outliers* e se eles podem acontecer novamente como uma parte normal do processo ou área de estudo.

A Fig. 2.17 apresenta um amostra de dados com alguns *outliers* referente a uma *feature* do conjunto de dados utilizado neste trabalho, sendo este a distância em pontos do preço de fechamento do índice Bovespa em relação a uma SMA de 5 (cinco) períodos.

Figura 2.17 – Exemplo de *outliers* observados na *feature* de distância da SMA de 5 (cinco) períodos



Fonte: do autor.

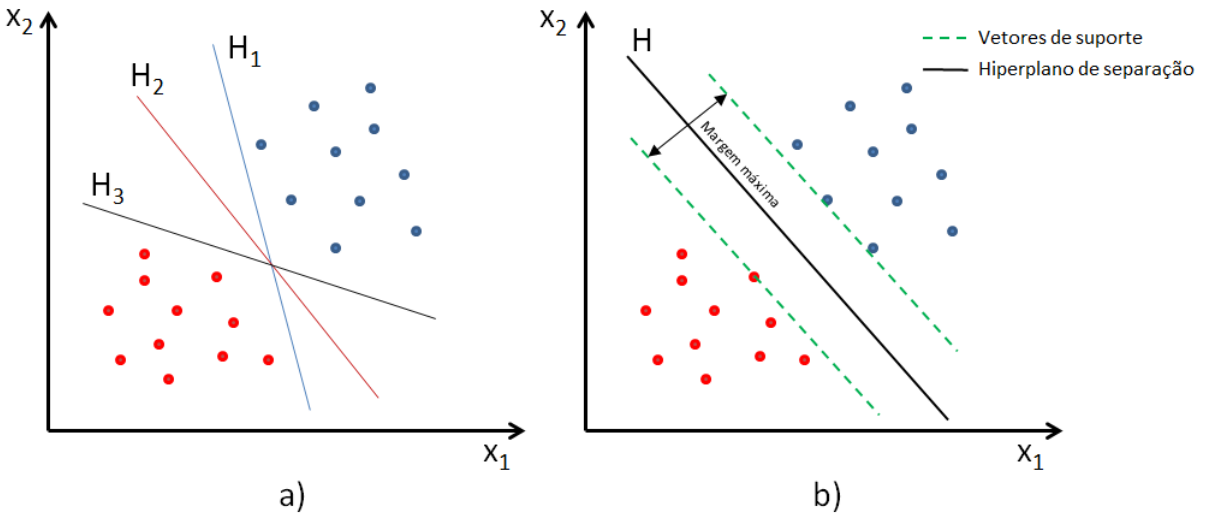
2.7 SVM *One-Class* para detecção de *outliers*

Resumidamente, as máquinas de vetores de suporte (SVM – *Support Vector Machines*) é um algoritmo de aprendizado de máquina proposto por Vapnik e Lerner (1963), para separação linear entre duas classes por meio de um hiperplano de separação. Anos mais tarde, Vapnik (1995) aprimorou o SVM para a separação não-linear entre duas classes por meio da utilização de funções kernel, como uma função de base radial (RBF – Radial Basis Function), método esse conhecido como truque de kernel (STEINWART; CHRISTMANN, 2008).

Existem diversas fronteiras de separação possíveis que são capazes de separar completamente as classes. Basicamente o SVM busca encontrar o melhor hiperplano de separação das classes, a busca é feita de modo a maximizar a distância entre os vetores de suporte e o

hiperplano, conforme ilustrado na Fig. 2.18 (STEINWART; CHRISTMANN, 2008; VAPNIK, 1995).

Figura 2.18 – a) Possíveis hiperplanos de separação para duas classes em espaço bidimensional. b) Melhor hiperplano de separação com margem máxima entre os vetores de suporte.



Fonte: do autor.

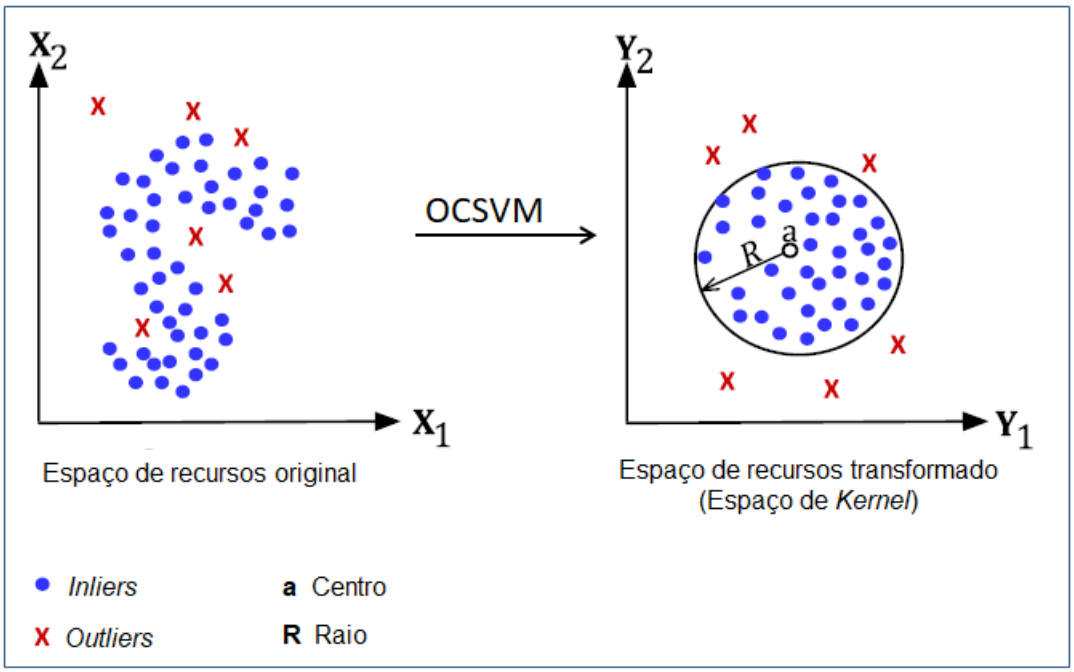
Para um conjunto de dados em um espaço dimensional d, o hiperplano de separação está em um espaço dimensional inferior $(d-1)$. Logo, para um conjunto de dados bidimensional, o hiperplano de separação é uma reta. Para um conjunto de dados tridimensional, o hiperplano será um plano, e assim sucessivamente (STEINWART; CHRISTMANN, 2008; VAPNIK, 1995).

Baseado no SVM, Schölkopf et al. (2001) introduziram o SVM *One-Class* (OCSVM), um método eficaz de detecção de *outliers* e por isso amplamente aplicado. Em suma, o OCSVM define uma fronteira de decisão fechada em torno dos dados, uma hiperesfera gaussiana que engloba todos os *inliers*. Um dado é considerado como *outlier* quando se encontra fora dos limite dessa hiperesfera. A quantidade de dados que devem ser rejeitados pela fronteira de decisão é definida pelos hiperparâmetros *nu* e *gama*, que representam respectivamente, a fração *outliers* presente nos dados e o raio da hiperesfera gaussiana que separa os *inliers* dos *outliers*. A fração de valores discrepantes (*nu*) ajuda a criar limites de decisão mais rígidos para melhorar a detecção de valores discrepantes. Grandes valores de *gama* definem uma hiperesfera menor, que encontra mais valores discrepantes, atuando como um parâmetro de corte para a hiperesfera que governa a fronteira de separação (MISRA; LI; HE, 2019) .

Por meio da função objetivo (equação 2.25), o OCSVM busca minimizar o raio (R), da hiperesfera da Fig. 2.19, no espaço de kernel de dimensão superior (espaço de recurso

transformado) que circunscreve os *inliers* separando ao máximo os dados da origem do espaço do kernel, encontrando uma hiperesfera ideal (ERFANI et al., 2016; MISRA; LI; HE, 2019).

Figura 2.19 – Hiperesfera obtida pela transformação do espaço de recursos com centro 'a' e raio 'R'.



Fonte: adaptado de (ERFANI et al., 2016).

$$\min_{a,R,\xi} \; R^2 + \frac{1}{nv}\sum_{i}^{n}\xi_i \tag{2.25}$$

$$\text{Sujeito a,} \qquad \left\|\Phi(x_i) - a^2\right\| \leq R^2 + \xi_i \;\mid\; \forall i = 1,\ldots,n,\;\; \xi_i \geq 0 \tag{2.26}$$

Onde:

- $\Phi()$ é a função que mapeia os dados no espaço de características para o espaço de kernel $\mathbb{R}^d \rightarrow \mathbb{R}^q \mid d < q$;
- x_i é cada amostra dos dados de treinamento, onde: $X = \{x_i : i = 1,\ldots,n\}$ e $X \in \mathbb{R}^d$;
- a é o centro da hiperesfera;
- R é o raio da hiperesfera;
- n é o número de amostras de treinamento;
- v é o parâmetro predefinido pelo usuário que regula o tamanho da hiperesfera e a fração de dados tidos como *outliers*.

- ξ_i é a variável de folga, associada a cada amostra de dados.

Seja $\alpha = [\alpha_1, ..., \alpha_n]^T$ e $0 \leq \alpha_i \leq \frac{1}{nv}$ e considerando a maximização da equação 2.27.

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i (x_i \cdot x_i) - \sum_{i,t} (x_i \cdot x_t), \ \forall i \geq 1, t \leq n \tag{2.27}$$

As amostras x_i classificadas como *inliers* nos dados de treinamento, satisfazem a inequação 2.28, de forma que o multiplicador de Lagrange correspondente resulta em $\alpha_i = 0$ (ERFANI et al., 2016; SCHÖLKOPF et al., 2001).

$$\|\phi(x_i) - a\|^2 < R^2 + \xi_i \tag{2.28}$$

As amostras x_i classificadas como *outliers* nos dados de treinamento satisfazem a inequação 2.29, de forma que o multiplicador de Lagrange correspondente resulta em $\alpha_i = \frac{1}{nv}$ (ERFANI et al., 2016; SCHÖLKOPF et al., 2001).

$$\|\phi(x_i) - a\|^2 > R^2 + \xi_i \tag{2.29}$$

Por fim, as amostras x_i que definem a fronteira da hiperesfera, satisfazem a equação 2.30, de forma que o multiplicador de Lagrange correspondente resulta em $0 < \alpha_i < \frac{1}{nv}$ (ERFANI et al., 2016; SCHÖLKOPF et al., 2001).

$$\|\phi(x_i) - a\|^2 = R^2 + \xi_i \tag{2.30}$$

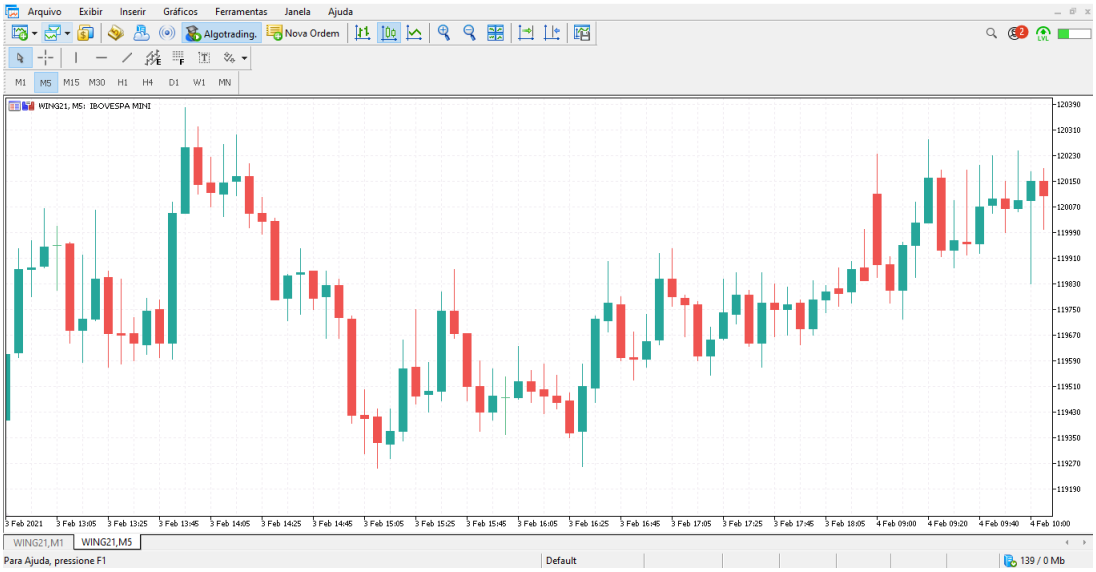
2.8 Plataforma de negociação MetaTrader 5

Tradando-se de um assunto carente de informações na literatura acadêmica, os conceitos, definições e informações descritas nesta secção têm como referências principais o site oficial[4] do MT5, o manual do MQL5 (Metaquotes (2000 - 2021)) e o site da comunidade de *traders* o *MQL5.community*[5] onde é compartilhada uma extensa biblioteca de códigos gratuitos e artigos. Os artigos publicados pela comunidade abrangem diversos tópicos da negociação moderna, tais como: redes neurais, estatísticas e análise, negociação de alta frequência, arbitragem, testes e otimização de estratégias de negociação, uso de robôs para negociações automatizadas e entre outros.

[4] https://www.metatrader5.com/pt
[5] https://www.mql5.com/pt/docs

O MT5 (Fig. 2.20) é uma plataforma institucional multimercado para *trading*, análise técnica, uso de sistemas automáticos de negociação (robôs de negociação) e cópia de transações de outros *traders*. Com a MetaTrader 5 pode-se negociar ao mesmo tempo no mercado de câmbio (Forex), ações e futuros.

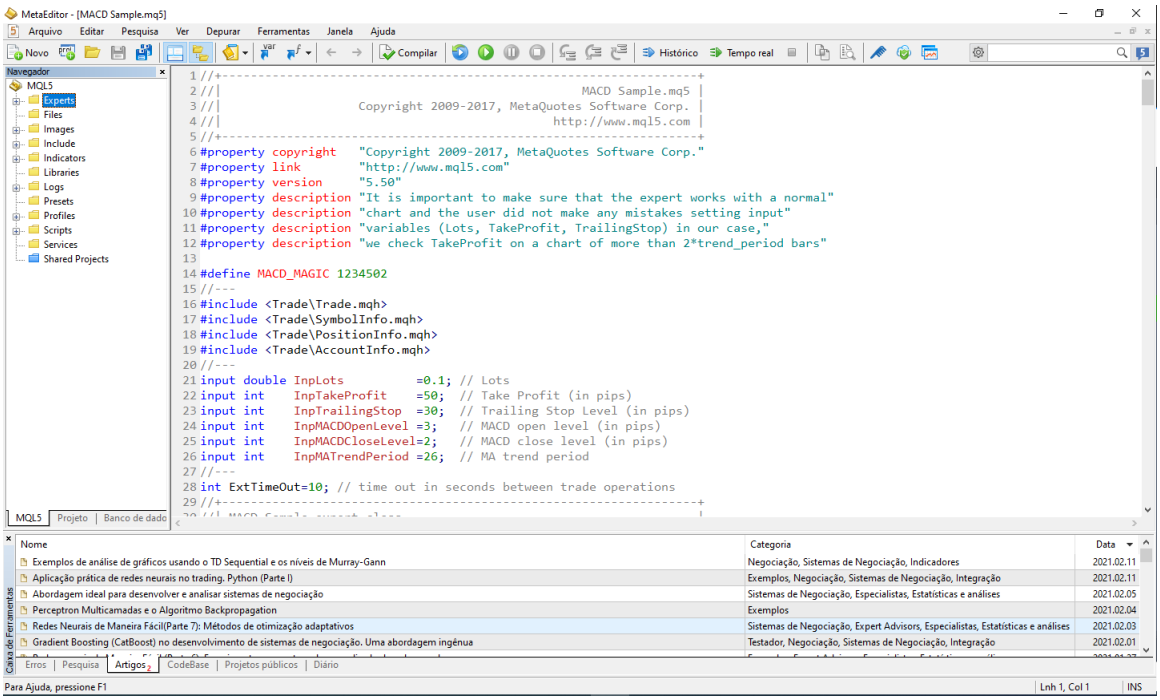
Figura 2.20 – Interface da plataforma MT5



Fonte: do autor.

O ambiente de desenvolvimento MQL5 IDE *(Integrated Development Environment)* detém todos os componentes do *algotrading* que permite criar, depurar, testar, otimizar e executar os robôs de negociação. A documentação Metaquotes (2000 - 2021), abrange a descrição completa de todas as construções da linguagem, que por sua vez inclui uma arquitetura orientada a objetos e uma sintaxe semelhante à lingagem de programação C++. Os algorítimos são criados no MetaEditor, que faz parte do MQL5 IDE e está estritamente ligado à plataforma MT5, nele é possível realizar depuração e compilação em arquivos executáveis. A Fig. 2.21 apresenta a interface do MetaEditor.

Figura 2.21 – Interface do MetaEditor



Fonte: do autor.

Os programas MQL5 têm propriedades e finalidades distintas:

- *Expert Advisor* (EA) - é o sistema de negociação automática que contém as funções manipuladoras de eventos, tais como: inicialização e a desinicialização do programa, movimentação mínima do preço do ativo (*tick*), ativação do temporizador, eventos do gráfico e nos eventos do usuário. O EA realiza *trades* automaticamente na conta de negociação baseando-se nos dados em regras estabelecidas no algoritmo, podendo substituir por completo seres humanos na tarefa de realizar negociações.

- Indicador personalizado - diferente dos EA's, os indicadores não têm acesso às funções de *trading* o que os impossibilita de realizar negociações. No entanto, podem ser utilizados por EA's e assim como os indicadores embutidos, destinam apenas à implementação de funções analíticas podendo usar os valores de outros indicadores em seus cálculos.

- *Script* - é um algoritmo projetado para executar uma sequência de operações uma única vez, tendo acesso a todas as funções analíticas e de negociação. Os *scripts* são executados no manipulador OnStart e ao contrário dos EA's, não lidam com eventos, exceto os de inicialização e desinicialização.

- biblioteca - é um conjunto de funções personalizadas que serve para armazenar e distribuir os blocos de programas usados com frequência.

Um EA desenvolvido em MQL5 executa determinada sequencia de comandos sempre que ocorre um evento de novo *tick*. O *tick* é a mínima movimentação da vela, ou seja, representa a variação mínima da cotação do ativo, significa que houve negociação no ativo e consequentemente movimentação do preço. Para cada novo *tick* as linhas de comando presentes na função (OnTick(void)) são executadas pelo EA.

2.8.1 Tipos de modelagem para backtesting

Considerando que o evento de novo *tick* (mudança de preço) é o principal evento para o EA, é de extrema importância considerar a escolha do tipo de modelagem que irá gerar a sequência de *ticks* para o *backtesting*. Basicamente existem 3 (três) modos de geração de *ticks* que simula a movimentação de cada vela no testador de estratégia, os principais são explicados a seguir:

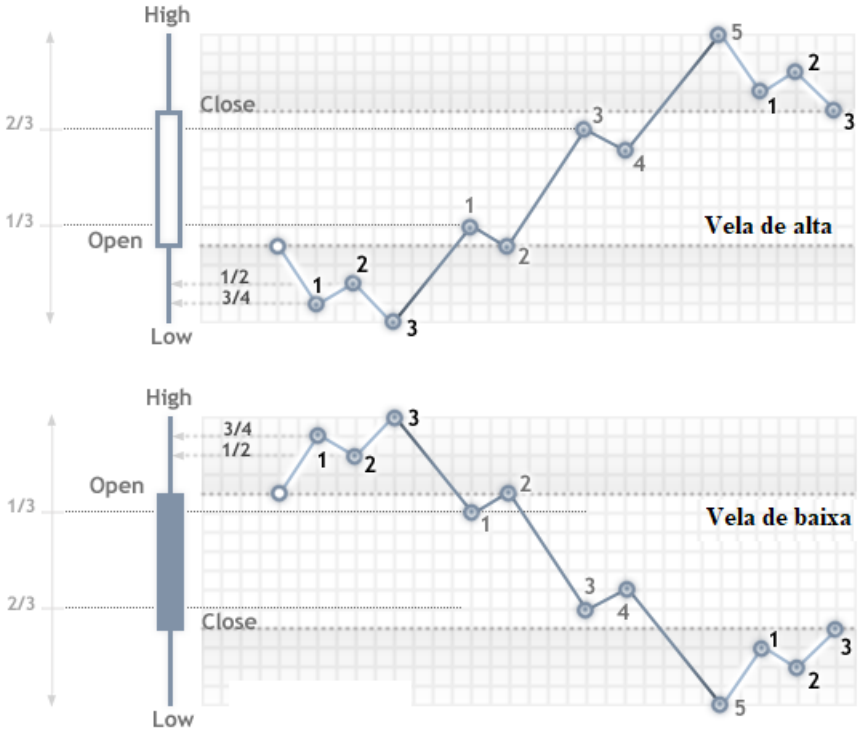
1. Cada *tick* - Nesse modo o MT5 simula a movimentação da vela segundo algumas regras específicas[6]. A Fig. 2.22 ilustra o modo de simulação dos pontos de apoio para a simulação de movimentação uma uma vela de alta e uma vela de baixa.

2. Cada *tick* é baseado em um *tick* real - Como o nome sugere, são usados os *ticks* reais que são provindos da bolsa e dos provedores de liquidez. Esse modo é o que mais se aproxima das condições reais.

3. OHLC por 1 minuto - Nesse modo são gerados apenas os *ticks* OHLC (*Open, High, Low e Close.*) das velas de 1 (um) minuto. Para a vela de 5 (cinco) minutos a movimentação ocorrerá entre os valores OHLC de cada vela de 1 (um) minuto contidas no intervalo de 5 (cinco) minutos, ou seja cada vela de 5 (cinco) minutos terá 20 (vinte) *ticks* de movimentação.

[6] hthttps://www.metatrader5.com/pt/terminal/help/algotrading/tick$_{g}$*eneration*

Figura 2.22 – Pontos de apoio para simulação de movimentação de velas no modo cada *tick*



Fonte: adaptado de (METAQUOTES, 2000 - 2021)

3 ESTADO DA ARTE

Neste capítulo é apresentado uma síntese a respeito do estudo bibliográfico dos artigos mais importantes e atuais relacionados com o tema de pesquisa deste trabalho. Serão apresentadas as principais técnicas de IA aplicadas na previsão de séries temporais financeiras, assim como as metodologias utilizadas no pre-processamento dos dados.

O uso de mecanismos de pré-processamento, pode na maioria das vezes, melhorar a precisão da previsão. Este recurso pode realizar a redução de dimensionalidade, eliminar ruídos ou detectar e corrigir valores discrepantes dos dados de entrada. Em seguida serão apresentados alguns trabalhos da literatura que utilizam algum mecanismo de pré-processamento nos dados de entrada antes da etapa de treinamento. Tsai e Hsiao (2010), Wang e Wang (2015), Zhong e Enke (2017), Yan et al. (2017) e Singh e Srivastava (2017) utilizaram a Análise de Componente Principais (PCA) para resumir o conjunto de dado nos recursos mais relevantes para treinamento. Todos os autores apresentam melhores resultados utilizando o PCA no pré-processamento dos dados quando comparado com situação sem o PCA.

Lu, Lee e Chiu (2009), Kao et al. (2013) usaram a Análise de Componentes Independentes (ICA) para identificar os componentes independentes do conjunto de dados de entrada e então filtram o ruído e *outliers*. Ambos os autores utilizaram o ICA não linear (NICA) em combinação com o método de Regressão por Vetor de Suporte (SVR) para prever os preços das ações. Lu (2010) comparou o desempenho do ICA e a Transformada Wavelet (WT), um método amplamente utilizado na análise de sinais. O ICA e o WT foram utilizados no pré-processamento dos dados de entrada de modo a remover o ruído dos dados antes do treinamento do modelo de rede neural MLP. Os resultados apresentados pelo autor mostram que o modelo de previsão com pré-processamento utilizando ICA tem maior precisão.

Henrique, Sobreiro e Kimura (2019) realizaram um estudo de revisão da literatura dos artigos mais relevantes publicados entre 1991 e 2017, o qual foi feito um levantamento a respeito das principais técnicas de aprendizado de máquina aplicadas à previsão de mercado financeiro. Os resultados apresentados demonstram que os modelos mais comumente usados para predição de séries temporais financeiras, são os que envolvem Máquinas de vetores de Suporte (SVMs) e RNA's.

Outro estudo de revisão mais recente foi realizado por Sezer, Gudelek e Ozbayoglu (2020), o objetivo principal dos autores foi avaliar os trabalhos publicados entre 2005 e 2019 que aplicam técnicas de aprendizado profundo (*Deep Learning* (DL)) na previsão de séries

temporais financeiras. Foi observado a predominância de modelos de Rede Neural Recorrente (RNN) baseados em memória de longo prazo, ou LSTM do inglês *"Long Short-Term Memory"*, como o modelo DL mais popular para previsão de séries temporais financeiras e estudos de regressão. Quando se trata de um problema de classificação, tais como os trabalhos que visam a previsão de direção ou tendência dos preços, observou-se que os modelos *Deep Multi-Layer Perceptron* (DMLP) e Redes Neurais Convolucionais (CNN) foram as principais escolhas das publicações mais recentes.

Na literatura, geralmente o foco principal é dado ao estudo de previsão das séries temporais financeiras considerando principalmente modelos de regressão. No entanto, há também pesquisas desenvolvidas com foco na previsão de tendências ou direção do preço, que usam modelos de classificação para prever se o preço do ativo financeiro irá subir ou descer. Nesta vertente, podemos citar o estudo recente publicado por Bustos e Pomares-Quimbaya (2020), que realiza uma revisão sistemática de artigos focados em prever a direção do preço do mercado de ações. Os autores observaram que a fonte mais popular de informações utilizadas para prever a direção do preço são os indicadores técnicos e/ou o preço de fechamento dos ativos. Observou-se também um aumento recente das análises focadas em mineração de texto, onde notícias ou opiniões de redes sociais geram representações numéricas que podem auxiliar na previsão dos preços das ações. Eles citam ainda, que a aplicação de técnicas de DL recentemente têm se tornado um tema popular de pesquisa, principalmente porque segundo os autores, não requerem a execução prévia do pré-processamento das informações.

Nos trabalhos que usam modelos de classificação, ou seja, previsão de tendências ou direção do preço, as métricas de desempenho são geralmente métricas de precisão. A precisão é calculada em função do percentual de assertividade das previsões. Outra métrica utilizada em alguns artigos é o desempenho como a porcentagem de retorno ou índice de lucro, onde uma técnica de negociação usando o algoritmo de previsão é testada para prever quão lucrativos são as previsões (BUSTOS; POMARES-QUIMBAYA, 2020; SEZER; GUDELEK; OZBAYOGLU, 2020).

Bagheri, Mohammadi Peyhani e Akbari (2014) propôs uma abordagem de previsão da direção do preço de pares de moeda do mercado *Forex* usando redes *neuro-fuzzy* ANFIS com Otimização *Quantum-Behaved Particle Swarm*. Os autores afirmam que o modelo proposto, aplicado em dados reais de Forex, apresenta capacidade de prever a direção do mercado com aproximadamente 69% de assertividade. Weng, Ahmed e Megahed (2017) propuseram modelos

utilizando RNAs, SVMs e árvores de decisão, combinando o uso do Google e a Wikipedia como fontes de dados online somados às séries temporais tradicionais e indicadores técnicos, para previsão da cotação do dia seguinte das ações da AAPL (Apple NASDAQ). Os autores afirmam obter uma precisão de 85% na previsão do movimento das ações da AAPL para o dia seguinte.

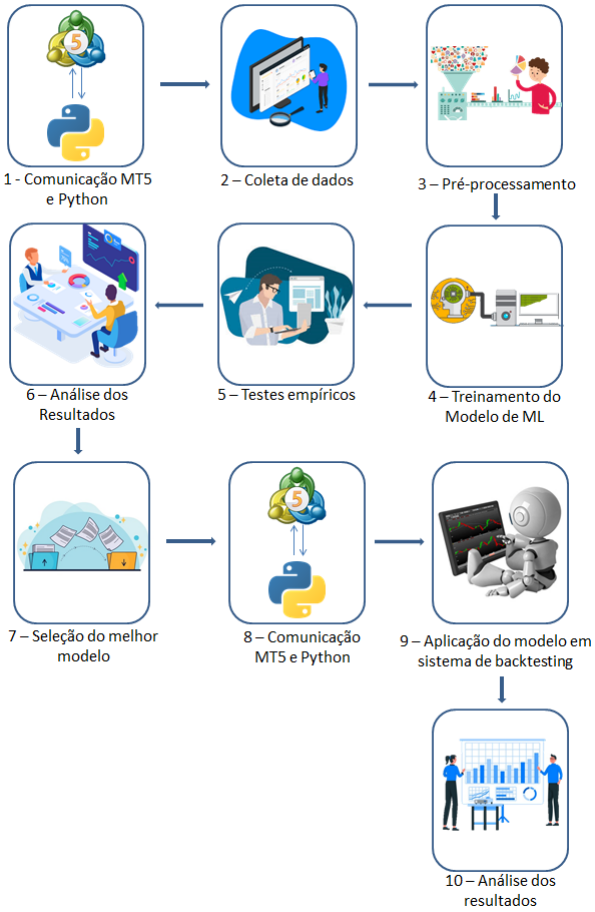
Com certeza, são excelentes os resultados apresentados por Bagheri, Mohammadi Peyhani e Akbari (2014), Weng, Ahmed e Megahed (2017), no entanto não foi desenvolvido pelos autores um sistema de negociação especialista que aplique o modelo em negociações reais, de forma a validar o método em ambiente real de negociação e ao vivo, da mesma forma que os demais trabalhos aqui estudados.

A metodologia empregada neste trabalho, difere-se da maioria dos trabalhos da literatura pelo fato de propor a implementação de um robô de investimento (EA) totalmente autônomo cujas decisões são baseadas em IA. Os trabalhos acadêmicos disponíveis na literatura, abordam técnicas de *machine learning* e *deeep leaning* para previsão de séries temporais financeiras, no entanto, a maioria não aplica essas técnicas em ambiente real de negociação. Reforça-se então a relevância da metodologia proposta, haja vista que, pretende-se avaliar a efetividade e eficiência da aplicação das técnicas de IA, na tomada de decisão em ambiente real de negociação, aplicando métricas de avaliação obtidas de simulação, com dados reais do mercado financeiro.

4 MATERIAIS E MÉTODOS

Este capítulo descreve a metodologia aplicada na proposta deste trabalho, o qual dedica-se a aplicação de técnicas de IA para prever a direção de movimentação do ativo WIN da B3, no tempo gráfico de 5 minutos. Serão usadas métricas baseadas em fator de lucro, rentabilidade, índice de *sharpe*, entre outros, que verdadeiramente evidenciam a eficiência de tais técnicas aliadas a uma estratégia de negociação. A Fig. 4.1 ilustra de forma resumida a metodologia proposta.

Figura 4.1 – Fluxograma simplificado da metodologia



Fonte: do autor.

Para avaliar o modelo adotar-se-á a metodologia experimental desde a seleção de *features* para treinamento do modelo, definição de hiper-parâmetros até a avaliação em ambiente simulado. Utilizar a investigação experimental (ou empírica) nos permite avaliar as relações causais, assumindo que as classes alvo da previsão têm algum tipo de relação explicativa com determinado conjunto de variáveis. De forma simplificada, a investigação empírica permite identificar o conjunto de variáveis que melhor explica determinado fenômeno.

No estudo de previsão de series temporais, as informações extraídas de uma série histórica, são determinantes no desenvolvimento de um modelo robusto e confiável, capaz de descrever as relações entre as variáveis observadas e a função objetivo. Fundamentalmente assume-se que alguns padrões analisados na série histórica se repetirão no futuro.

Nas subseções a seguir, serão apresentados os recursos necessários assim como o detalhamento da metodologia empregada.

4.1 Recursos Necessários

Para a desenvolvimento da pesquisa foi necessário uso de recursos de *hardware* e *softwares*, sendo eles:

- Notebook/Desktop - com sistema operacional *Windows* de 64 bits, memória RAM de 8 GB, processador Intel(R) Core(TM) i7-7500U CPU @ 2.70 GHz 2.90 GHz e SSD de 120 GB, além de acesso a internet e conta cadastrada em corretora de investimentos que possibilite acesso ao mercado financeiro brasileiro, mais especificamente a B3.
- MetaTrader 5 - trata-se de uma plataforma institucional gratuita multimercado para *trading*, análise técnica, uso de sistemas automáticos de negociação (robôs de negociação) e varias outras funcionalidades de *trading*.
- Jupyter Notebook[1] - um aplicativo web gratuito de código aberto que permite criar e compartilhar *scripts* em *Python*, uma linguagem de programação de alto nível, dinâmica, interpretada, modular, multiplataforma e orientada a objetos que permite manipulação, visualização e transformação de dados, simulação numérica, modelagem estatística, aprendizado de máquina e muito mais.

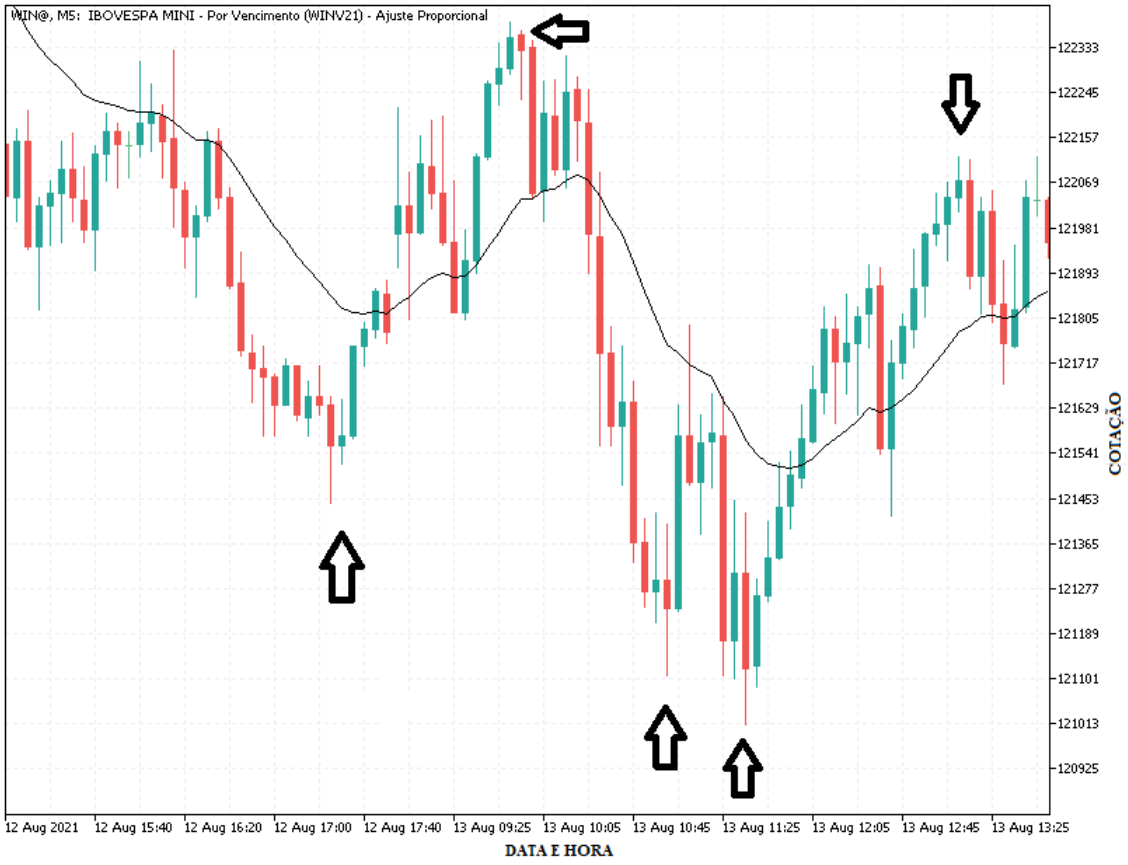
4.2 Hipótese de estratégia

Este trabalho visa criar um modelo preditor de direção para o ativo WIN, no tempo gráfico de 5 (cinco) minutos. Partindo de uma análise gráfica simples da série histórica do WIN, é possível inferir a existência de um padrão de retorno do preço para a média, este não é um padrão exclusivo do ativo WIN, podendo ser observado em qualquer ativo do mercado financeiro. Também é importante frisar que não se trata de uma regra, no entanto observa-se que

[1] https://jupyter.org/

os preços oscilam em torno de uma média e que o afastamento momentâneo do preço em relação a essa média, pode ser considerado como um movimento atípico, que em sua grande maioria, tende a não permanecer afastado da média. Neste contexto, observa-se que a tendência é que ocorra o retorno do preço para a média. A Fig. 4.2 ilustra este padrão observado, onde as setas mostram pontos onde houve um distanciamento excessivo da média móvel (linha contínua).

Figura 4.2 – Exemplo de retorno para a média.

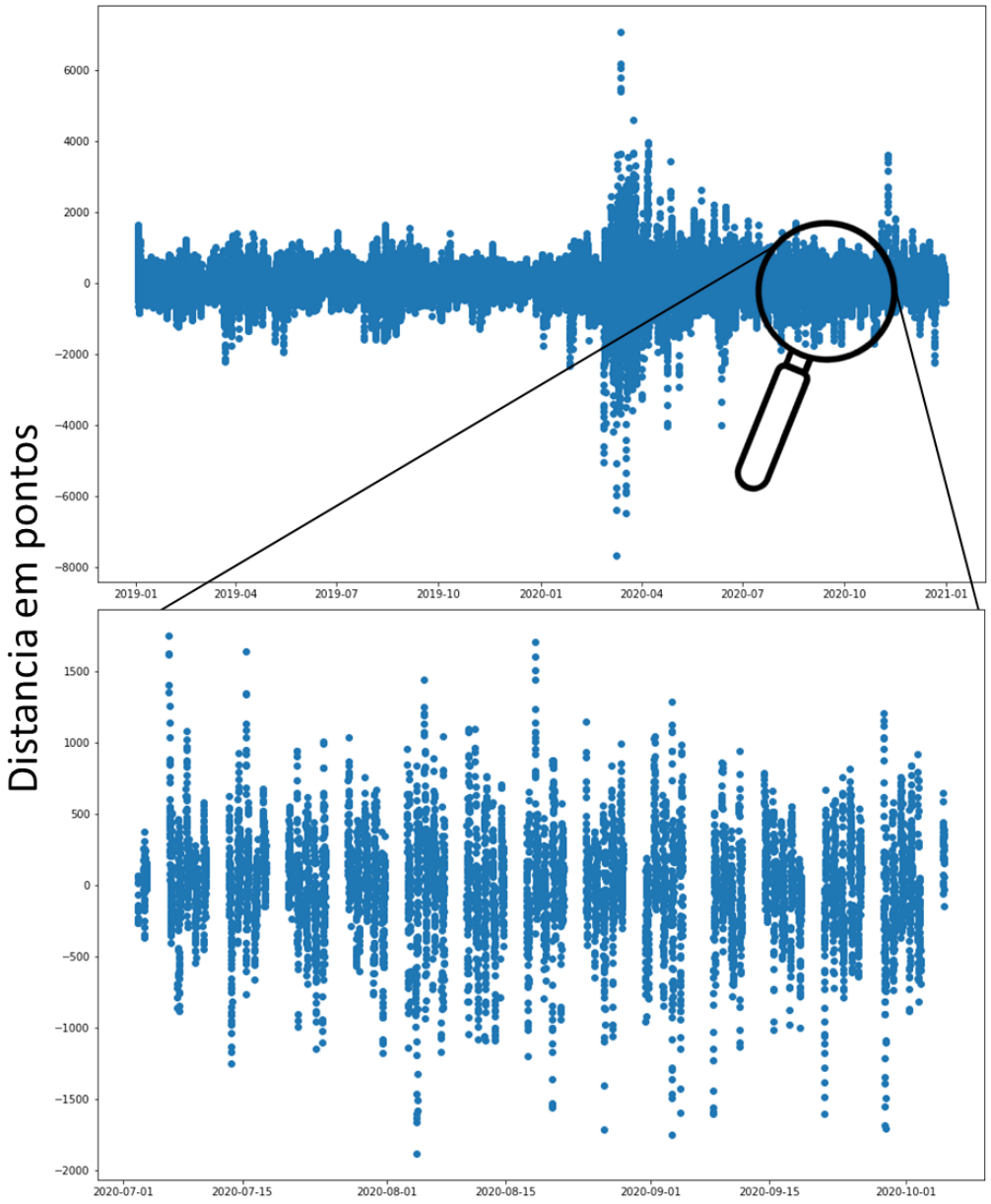


Fonte: do autor.

De posse dos dados históricos de cotação do WIN para o período amostral que compreende os anos de 2019 a 2020, é possível realizar uma análise de dispersão da distância do preço de fechamento em relação a EMA de 21 períodos (Fig. 4.3), e consequentemente podemos analisar o tipo de distribuição dos dados conforme ilustrado na Fig. 4.4.

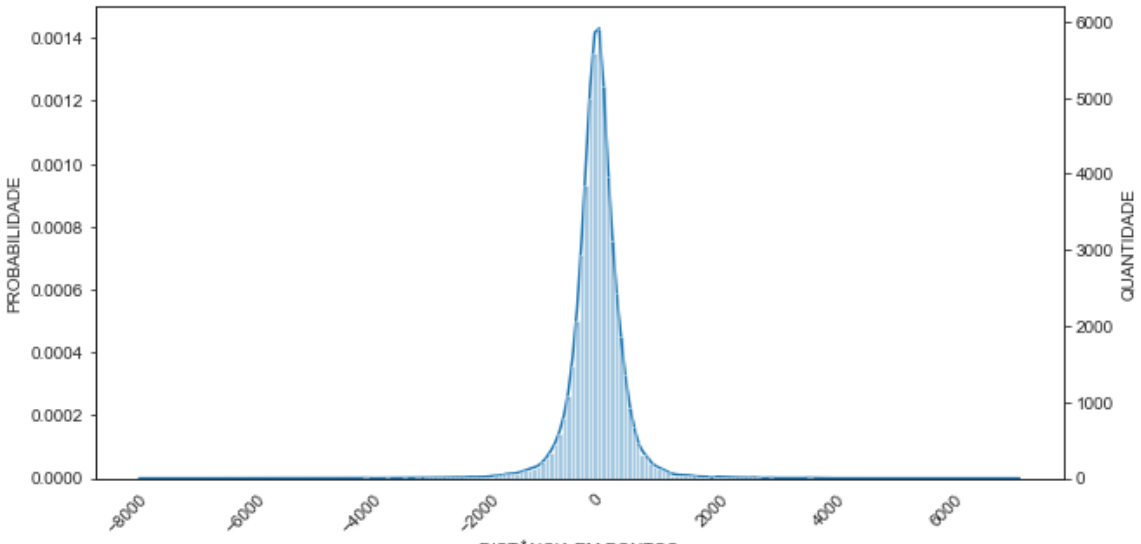
Os vales observados na figura anterior estão relacionados aos dias que não há pregão, por exemplo finais de semana e feriados.

Figura 4.3 – Gráfico de dispersão da distância da EMA 21 períodos



Fonte: do autor.

Figura 4.4 – Gráfico de distribuição da distância da EMA 21 períodos



A partir desta análise, define-se qual será a estratégia exploratória dos dados, o objetivo é extrair dados da série histórica do WIN que apresentem um padrão de distribuição encontrado na análise de dispersão da distância da média. Considerando que os eventos de afastamento da EMA podem ser entendidos como *outliers*, o objetivo será selecionar o melhor conjunto de *features* e hiperparâmetros para o SVM *one class* e RNA, de forma a obter um modelo que seja capaz de identificar os pontos de reversão do distanciamento da EMA. O problema será tratado como sendo de classificação de duas classes, sendo elas: "alta" e "baixa". A classe "alta" representa a previsão que a próxima vela será de alta e a classe baixa indica a previsão que a próxima vela será de baixa.

4.3 Comunicação MT5 e Python

A obtenção dos dados do WIN e análise exploratória dos dados é realizada na linguagem de programação *Python* com uso da interface Jupyter Notebook, que possibilita a prototipagem rápida e compartilhamento *online* de projetos, além de permitir escrever pequenos trechos de código e executá-los sequencialmente, o que torna a codificação mais fácil de depurar e entender. Para isso, é necessário realizar a integração entre os *softwares* MetaTrader 5 e Python. O MetaTrader 5 é o sistema responsável pela conexão com o ambiente de negociação da B3 e nos permite ter acesso aos dados históricos dos ativos disponíveis na bolsa.

O Metaquotes (2000 - 2021) oferece um pacote MetaTrader para Python responsável pela comunicação entre os *softwares*, a partir daí é possível extrair os dados históricos do WIN, diretamente no ambiente de programação Jupyter Notebook, onde será realizada toda a análise exploratória dos dados e treinamento dos modelos de IA. O trecho do algoritmo que realiza essa comunicação é mostrado no Código 4.1.

Código 4.1 – Código que executa a conexão Python e MT5

```
1 # Realiza a conexão com MT5
2 import MetaTrader5 as mt5
3 if not mt5.initialize():
4     print('Erro ao iniciar o MT5')
5     mt5.shutdown()
```

Fonte: do autor.

4.4 Coleta de dados

Dentre as diversas funções do pacote MetaTrader para Python, tem-se a função apresentada a seguir:

```
copy_rates_from_pos(
   symbol,        // nome do símbolo
   timeframe,     // período gráfico
   start_pos,     // número da vela inicial
   count)         // número de velas
```

Esta função permite ao usuário solicitar as informações de um ativo em específico para o terminal do MT5, passando como parâmetros o tempo gráfico (*timeframe*), o número da vela inicial (*start_pos*), onde 0 (zero) representa a vela corrente, 1 (um) a última vela fechada e assim sucessivamente. O último parâmetro (*cont*) define a quantidade de velas que se deseja obter os dados. O trecho de código a seguir (Código 4.2) apresenta a função criada para obtenção dos dados e a extração dos dados de 100 mil velas de 5 minutos. A Fig. 4.5 apresenta um trecho do *dataframe* obtido.

Código 4.2 – Conexão Python e MT5

```
# Função para extrair os dados históricos de um ativo financeiro
    no MT5
# Parâmetros:
# 1 (ativo)     - código do ativo, por exemplo: WIN$;
# 2 (timeframe) - o tempo gráfico, por exemplo: M5 (5 MINUTOS);
# 3 (num_velas) - a quantidades de velas o qual pretende-se
    extrair as informações.

def extrair_mt5(ativo, timeframe, num_velas):
    data = mt5.copy_rates_from_pos(ativo,timeframe,0,num_velas)
    df_data = pd.DataFrame(data)
    df_data['time'] = pd.to_datetime(df_data['time'],unit='s')
    return df_data

#Extraindo os dados do MT5
```

```
14 data = extrair_mt5('WIN$',mt5.TIMEFRAME_M5,100000)
15 data.dropna(inplace=True) # elimina os dados NaN do dataframe
16 data.reset_index(inplace = True, drop = True) # reset no index do
       dataframe
17 data = data[:-1] # elimina a ultima linha pois eh a vela corrente
```

Fonte: do autor.

Figura 4.5 – *DataFrame* com dados obtidos do MT5

	time	open	high	low	close	tick_volume	spread	real_volume
0	2018-01-30 15:15:00	98889.0	98907.0	98819.0	98860.0	3389	1	13429
1	2018-01-30 15:20:00	98860.0	98919.0	98819.0	98860.0	3684	1	14331
2	2018-01-30 15:25:00	98854.0	98872.0	98731.0	98743.0	4115	1	16811
3	2018-01-30 15:30:00	98743.0	98825.0	98737.0	98801.0	2814	1	10611
4	2018-01-30 15:35:00	98807.0	98842.0	98778.0	98819.0	2720	1	10698
...	...	...	...	...	...	...	...	...
99993	2021-10-28 09:25:00	106850.0	106980.0	106680.0	106815.0	25232	0	246254
99994	2021-10-28 09:30:00	106815.0	106840.0	106600.0	106730.0	24306	0	249193
99995	2021-10-28 09:35:00	106725.0	106920.0	106655.0	106805.0	18870	0	225257
99996	2021-10-28 09:40:00	106800.0	106970.0	106620.0	106870.0	18309	0	208614
99997	2021-10-28 09:45:00	106865.0	107015.0	106790.0	106855.0	19433	0	225587

Fonte: do autor.

As informações disponíveis para cada vela são:

1. ***time*** - é a data, hora e minuto de fechamento da vela;
2. ***open*** - é a cotação de abertura da vela;
3. ***high*** - é a cotação máxima da vela;
4. ***low*** - é a cotação mínima da vela;
5. ***close*** - é a cotação de fechamento da vela;
6. ***tick_volume*** - representa a quantidade de negociações de compra e venda ocorreram no período;
7. ***spread*** - é a diferença em *tick* entre as melhores cotações de venda e compra, onde um *tick* representa a mínima movimentação do ativo, por exemplo: o WIN tem um *tick* de 5 (cinco) pontos;

8. ***real_volume*** - é a quantidade de ações, contratos futuros ou unidades de moeda negociados.

4.5 Pré-processamento

O pré-processamento dos dados se subdivide em 5 (cinco) etapas a saber:

1. Extração de recursos para obtenção das *features*;
2. Definição das *labels*, segundo hipótese apresentada na secção 4.2;
3. Normalização dos dados;
4. Seleção de *features*:
    - *SelectKBest*;
    - PCA.

4.5.1 Obtenção das *features*

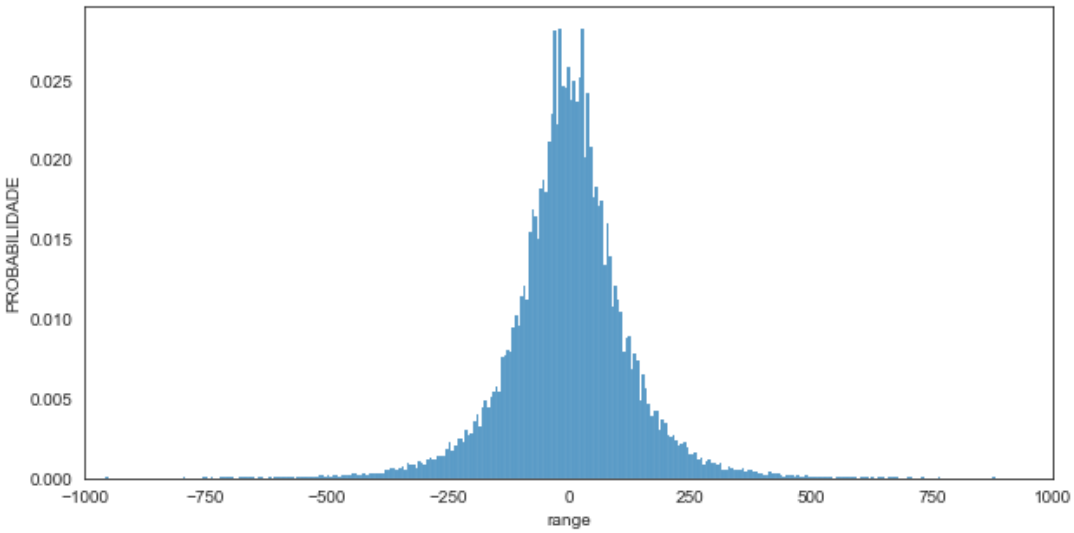
Foi realizada a extração de novos recursos, obtidos da manipulação dos dados históricos disponíveis no *dataframe* (Fig. 4.5), estes por sua vez, constituirão o conjunto de dados para treinamento do modelo. As subceções a seguir apresentam as *features* obtidas para constituição do conjunto de dados.

4.5.1.1 *Range*

O *Range* é basicamente a medida do tamanho do corpo da vela, essa informação foi utilizada para definição da direção da vela seguinte (objeto da previsão), onde valores positivos de range indicam uma vela de alta e valores negativos uma vela de baixa. A equação 4.1 define o calculo do range e a Fig. 4.6 apresenta o gráfico de distribuição do *range*.

$$range_i = close_i - open_i \tag{4.1}$$

Figura 4.6 – Gráfico de distribuição do *range*



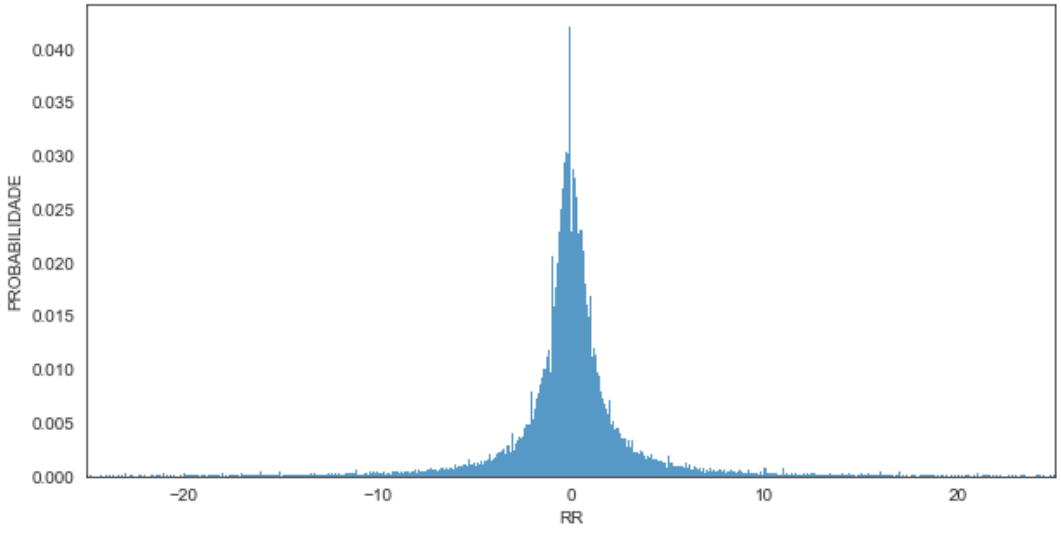
Fonte: do autor.

4.5.1.2 Relação *Range*

A Relação *Range* (RR) mede a relação entre os *ranges* da vela atual e a vela anterior. A equação (4.2) apresenta o cálculo para obter a RR e a Fig. 4.7 apresenta o seu gráfico de distribuição.

$$RR_i = \frac{range_i}{range_{i-1}} \tag{4.2}$$

Figura 4.7 – Gráfico de distribuição da RR



Fonte: do autor.

4.5.1.3 Relação Corpo Pavio

Trata-se de uma *feature* calculada conforme equação (4.3) que expressa a relação entre o tamanho do corpo da vela e sua amplitude total. Acredita-se que essa informação retrata um cenário de força compradora ou vendedora.

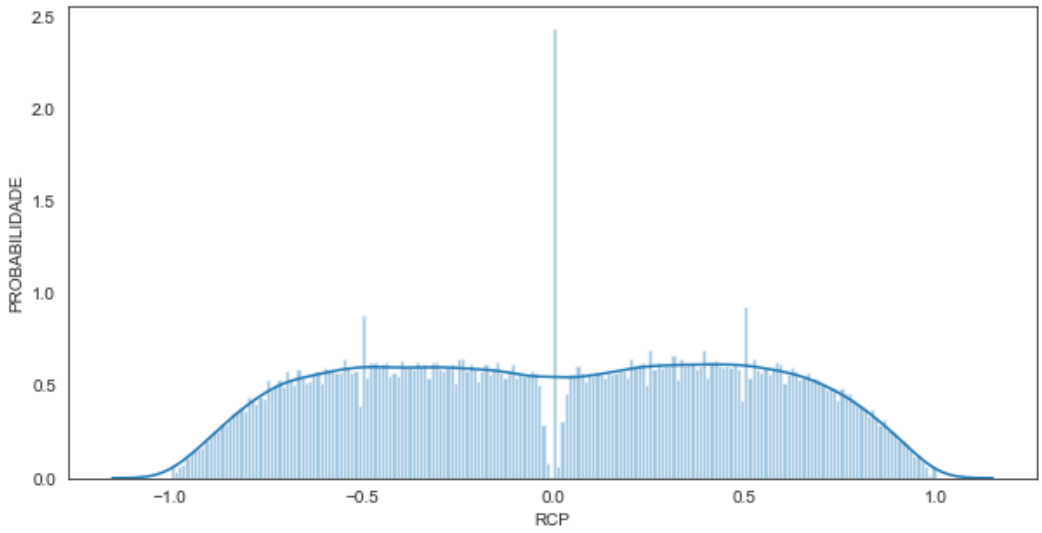
$$RCP_i = \frac{close_i - open_i}{high_i - low_i} \tag{4.3}$$

Onde:

- RCP - é a Relação Corpo Pavio da vela, tal que: $RCP \in \mathbb{R} / RPC \neq 0$;
- i - é o índice da vela, conforme *dataframe* do conjunto de dados (Fig. 4.5).

Valores de $RCP \cong (+1)$ indicam força da ponta compradora e valores $RCP \cong (-1)$ indicam força da ponta vendedora. A Fig. 4.8 demostra a distribuição dos dados da RCP.

Figura 4.8 – Gráfico de distribuição da RCP

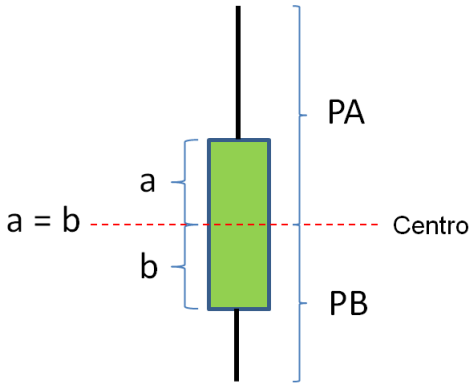


Fonte: do autor.

4.5.1.4 Relação PaviO

Análogo ao RCP, o parâmetro Relação Pavio (RP) mede a relação entre os tamanhos dos pavis de alta e de baixa, medidos do ponto central do corpo da vela até as respectivas extremidades, conforme ilustrado na Fig. 4.9.

Figura 4.9 – Representação para cálculo da RP



Fonte: do autor.

A equação (4.4) é utilizada para o cálculo da RP.

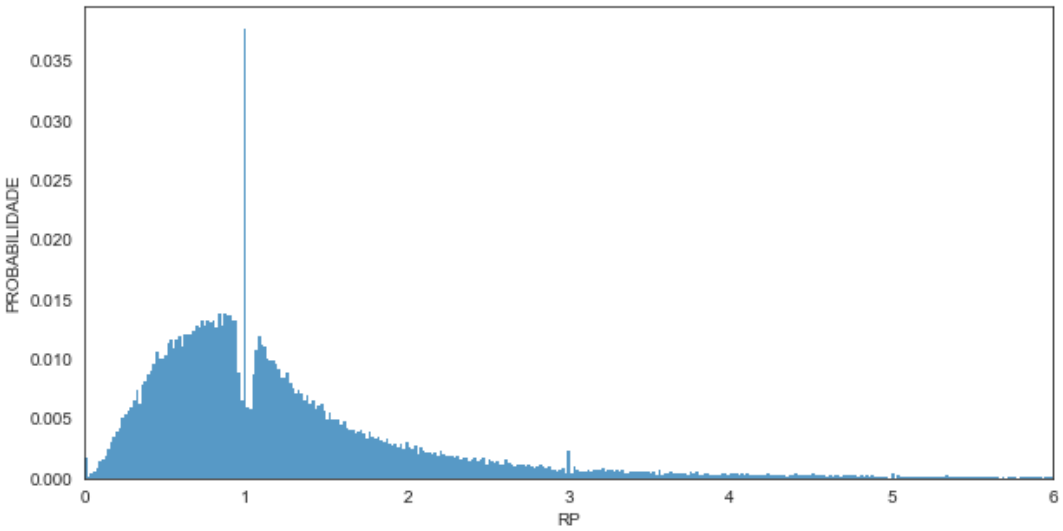
$$RP_i = \frac{PA_i}{PB_i} = \begin{cases} \frac{high_i - \frac{\|close_i - open_i\|}{2}}{\frac{\|close_i - open_i\|}{2} - low_i}, \text{ se } \frac{\|close_i - open_i\|}{2} - low_i > 0 \\ high_i - \frac{\|close_i - open_i\|}{2}, \text{ se } \frac{\|close_i - open_i\|}{2} - low_i = 0 \end{cases} \tag{4.4}$$

Onde:

- RP - é a Relação Pavio da vela, tal que: $RP \in \mathbb{R} / RP \geq 0$;
- PA - é o tamanho do pavio de alta;
- PB - é o tamanho do pavio de baixa;

A Fig. 4.10 apresenta o gráfico de distribuição da RP.

Figura 4.10 – Gráfico de distribuição da RP



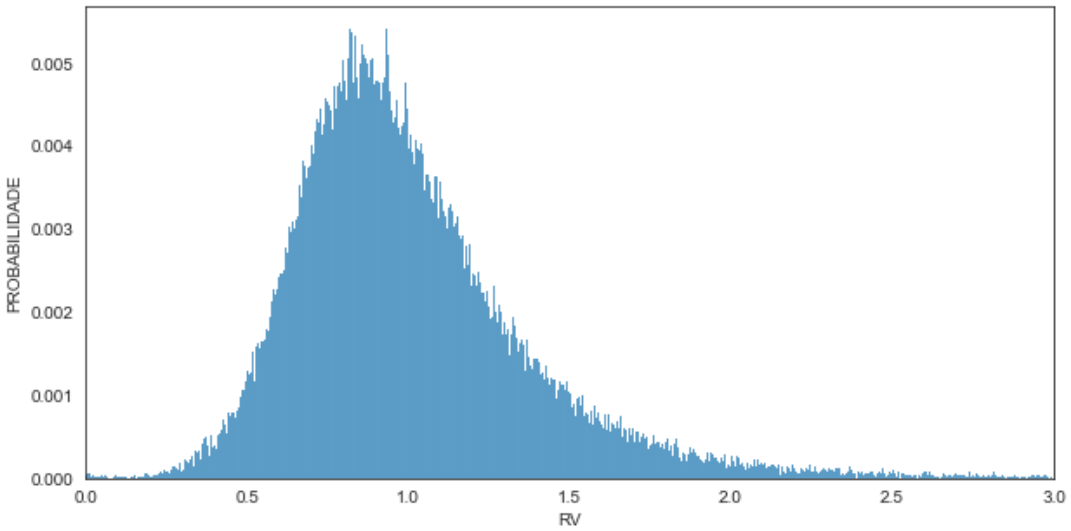
Fonte: do autor.

4.5.1.5 Relação Volume

A Relação Volume (RV) descreve a relação entre a quantidade de contratos negociados na vela atual e a SMA do volume negociado nas últimas 7 (sete) velas ($n = 7$). A equação (4.5) apresenta cálculo para a RV e a Fig. 4.11 apresenta o gráfico de distribuição da RV.

$$RV_i = \frac{real\_volume_i}{\frac{1}{n}\sum_{j=1}^{n} real\_volume\,j} \tag{4.5}$$

Figura 4.11 – Gráfico de distribuição da RV



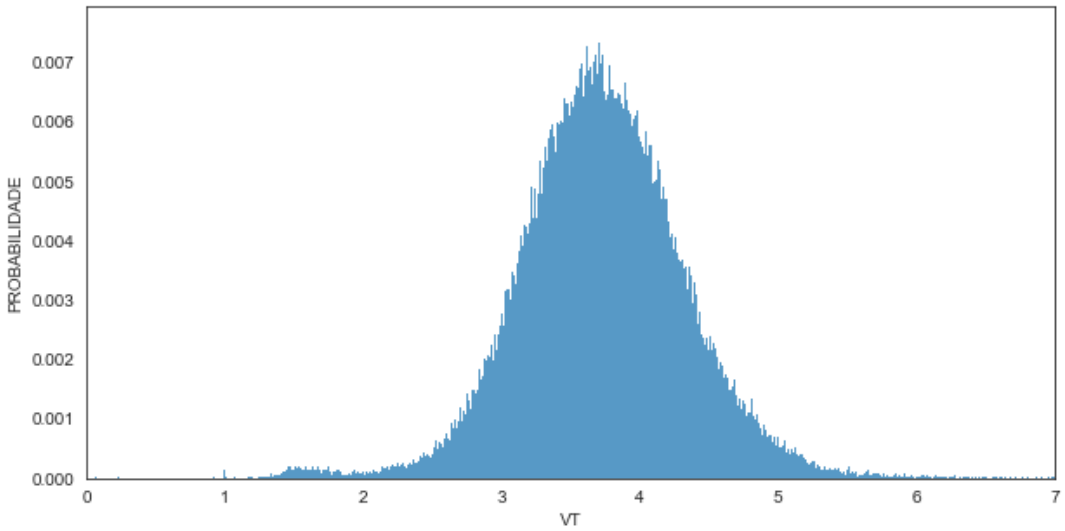
Fonte: do autor.

4.5.1.6 Volume por Tick

O Volume por Tick (VT) representa a média de contratos negociados por cada tick de movimentação da vela, em suma VT mede a quantidade média de contratos que foram negociados em cada faixa de preço de negociação no período de tempo da vela em questão. A equação (4.6) apresenta o cálculo para obter o VT e a Fig. 4.12 apresenta o gráfico de distribuição.

$$VT_i = \frac{real\_volume_i}{tick\_volume_i} \tag{4.6}$$

Figura 4.12 – Gráfico de distribuição da VT



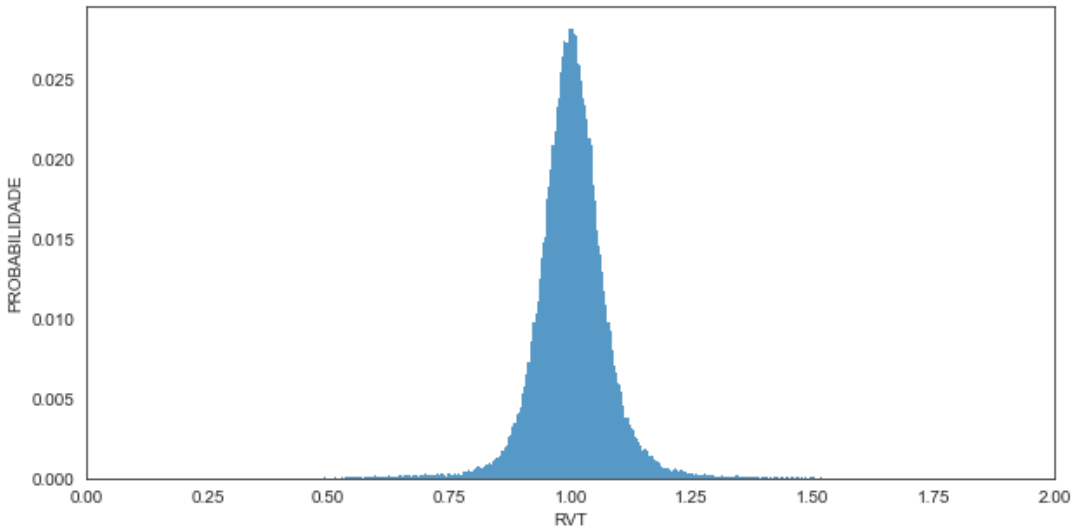
Fonte: do autor.

4.5.1.7 Relação Volume por Tick

Derivado do VT, a Relação Volume por Tick (RVT) representa a relação entre VT da vela atual e VT da vela anterior, valores maiores que 1 (um) indicam aumento de liquidez e valores menores de que 1 indicam diminuição de liquidez. A equação (4.7) apresenta o cálculo de VT e a Fig. 4.13 o gráfico de distribuição.

$$RVT_i = \frac{VT_i}{VT_{i-1}} \tag{4.7}$$

Figura 4.13 – Gráfico de distribuição da RVT



Fonte: do autor.

4.5.1.8 Distância entre as Bandas de Bollinger

Essa *feature* representa a medida de distância entre as bandas superior e inferior de Bollinger. Para isso, faz-se necessário realizar o cálculo do indicador "Bandas de Bollinger"conforme apresentado no Código 4.3.

Código 4.3 – Cálculo das Bandas de Bollinger

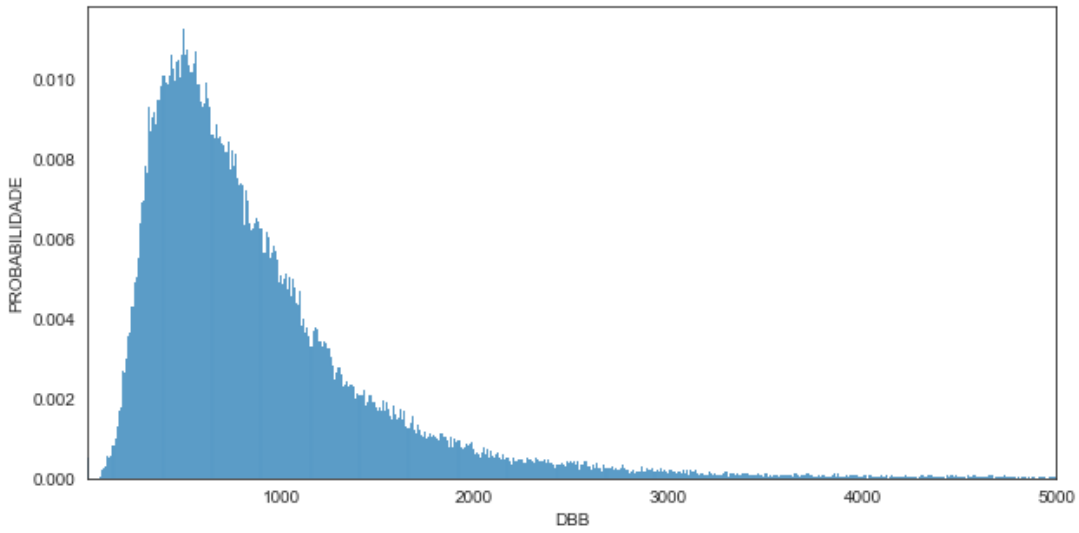
```
# Obtenção de novas features para treinamento do modelo
# Calculando as Bandas de Bollinger

N = 20 # Período da média móvel
k = 2  # Fator de distanciamento das bandas de bollinger
df['Standard Deviation'] = df['close'].rolling(N).std()
df['Middle Band'] = df['close'].rolling(N).mean()
df['Upper Band'] = df['Middle Band'] + df['Standard Deviation'] * k
df['Lower Band'] = df['Middle Band'] - df['Standard Deviation'] * k
```

Fonte: do autor.

A equação (4.8) é utilizada para o cálculo da distância entre as Bandas de Bollinger (DBB) e a Fig. 4.14 mostra o padrão de distribuição dos dados. Essa *feature* explica a volatilidade do ativo, o qual é diretamente proporcional aos valores da DBB.

$$DBB_i = BS_i - BI_i \tag{4.8}$$

Figura 4.14 – Gráfico de distribuição da DBB



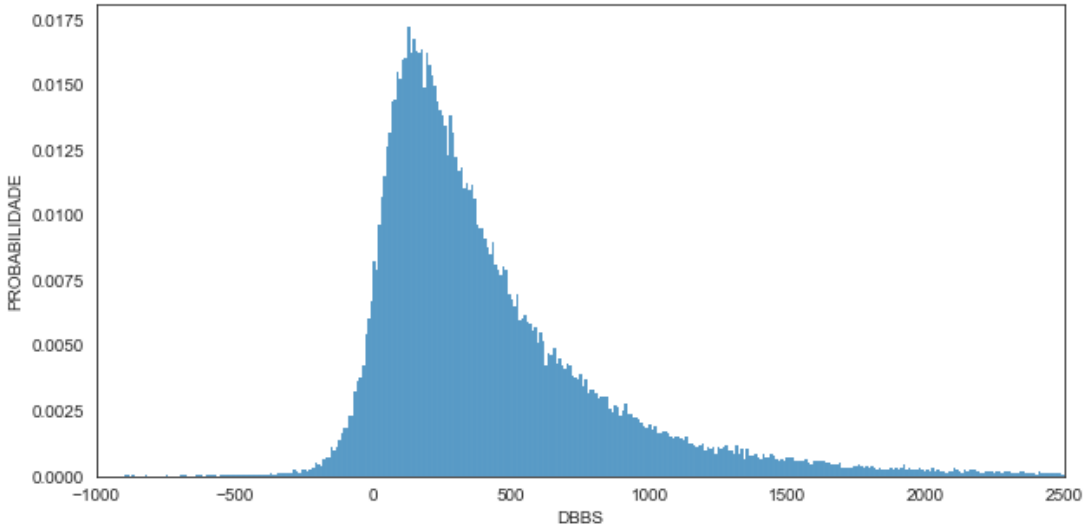
Fonte: do autor.

4.5.1.9 Distância da Banda de Bollinger Superior

A Distância da Banda de Bollinger Superior (DBBS) refere-se a medida de distância em pontos, entre a banda de Bollinger superior e o preço de fechamento da vela. A equação (4.9) é utilizada para o cálculo desta *feature* e a Fig. 4.15 apresenta o gráfico de distribuição da DBBS.

$$DBBS_i = BS_i - close_i \tag{4.9}$$

Figura 4.15 – Gráfico de distribuição da DBBS



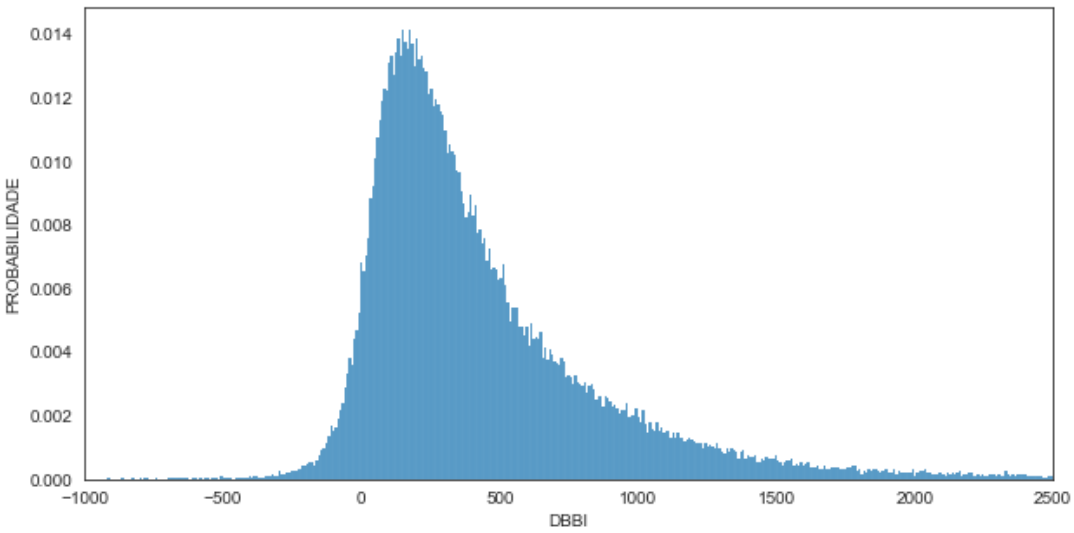
Fonte: do autor.

4.5.1.10 Distância da Banda de Bollinger Inferior

Análogo a DBBS, a Distância da Banda de Bollinger Inferior (DBBI) é a medida de distância entre o preço de fechamento da vela e a banda de Bollinger inferior. A equação (4.10) é utilizada para o cálculo desta *feature* e a Fig. 4.16 apresenta o gráfico de distribuição da DBBI.

$$DBBI_i = close_i - BI_i \tag{4.10}$$

Figura 4.16 – Gráfico de distribuição da DBBI



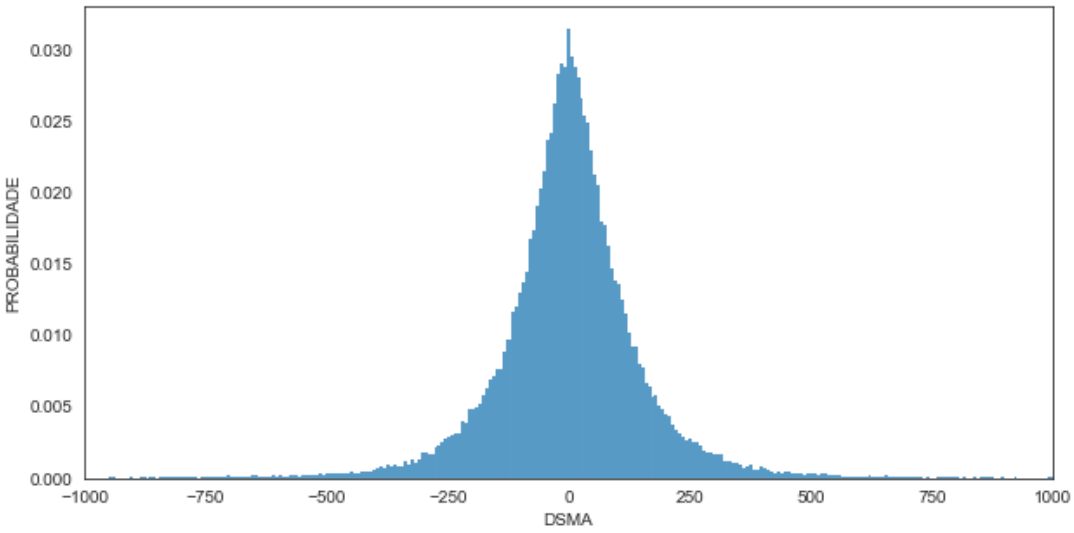
Fonte: do autor.

4.5.1.11 Distância da SMA

A Distância da SMA (DSMA), é a medida de distância do preço de fechamento da vela em relação a uma média móvel simples (SMA) de 5 (cinco) períodos. A equação (4.11) é utilizada para o cálculo desta *feature* e a Fig. 4.17 apresenta o gráfico de distribuição da DSMA.

$$DSMAi = close_i - \frac{1}{n}\sum_{j=1}^{n} real\_volume\, j \tag{4.11}$$

Figura 4.17 – Gráfico de distribuição da DSMA



Fonte: do autor.

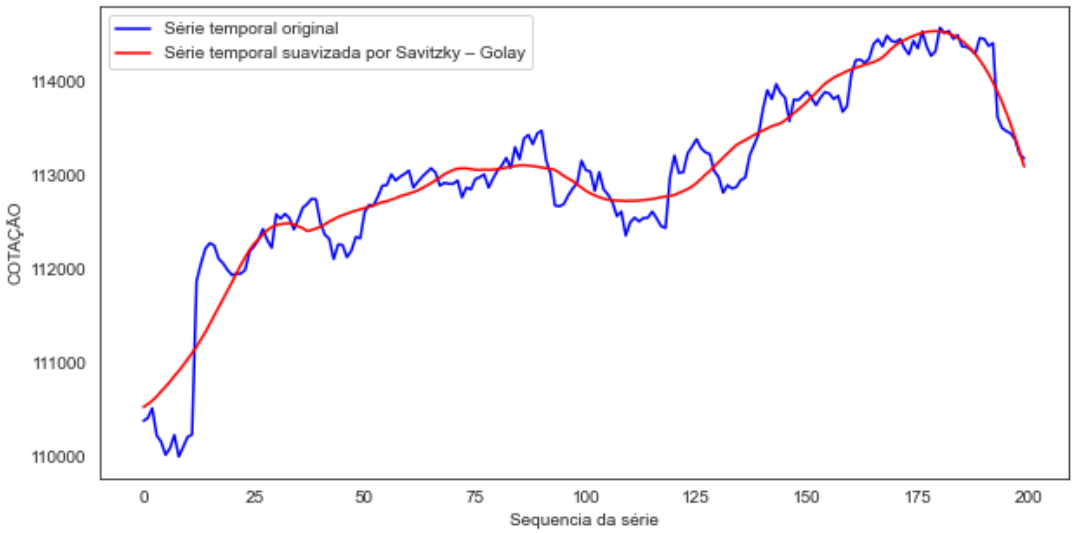
4.5.1.12 Distância do preço suavizado pelo filtro de Savitzky – Golay

Depois de aplicado o filtro de Savitzky - Golay na série histórica original do WIN, obtém-se o preço de fechamento suavizado, conforme ilustrado na Fig. 4.18. De posse da série temporal suavizada obteve-se a distância do preço de fechamento da vela em relação ao preço suavizado, conforme equação (4.12). A Fig. 4.19 apresenta o gráfico de distribuição da DSG.

$$DSGi = close_i - x_i \tag{4.12}$$

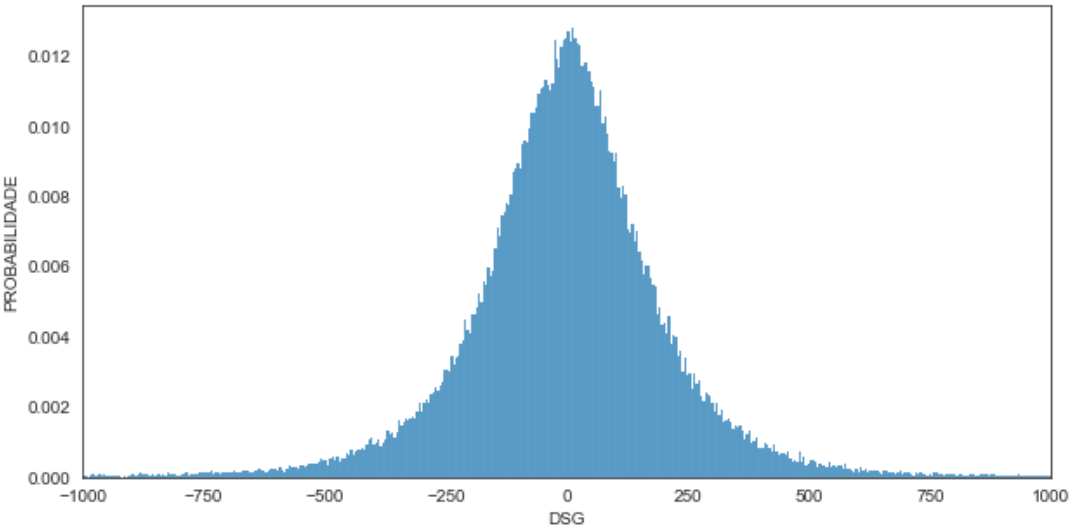
Onde x_i é o resultado da suavização obtida pelo filtro, conforme equação (2.9).

Figura 4.18 – Suavização da série temporal WIN pelo filtro de Savitzky – Golay



Fonte: do autor.

Figura 4.19 – Gráfico de distribuição da DSG



Fonte: do autor.

4.5.2 Definição das *labels*

A definição das *labels* ou rótulos para treinamento do modelo dar-se-á pela identificação dos cenários que, conforme hipótese apresentada, obedeçam aos seguintes requisitos:

- *Label* para velas de alta:

    1. O preço de abertura da vela atual ($open_i$) seja menor que a banda de bollinger inferior (BBI_i);

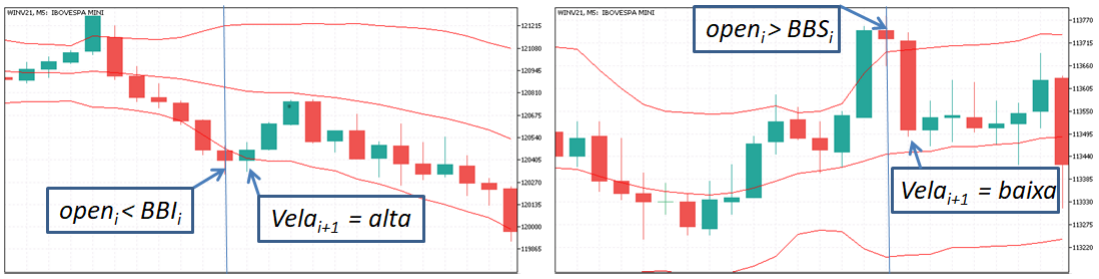
2. A vela seguinte $(i+1)$ seja uma vela de alta;

- *Label* para velas de baixa:

1. O preço de abertura da vela atual $(open_i)$ seja maior que a banda de bollinger superior (BBS_i);
2. A vela seguinte $(i+1)$ seja uma vela de baixa;

A Fig 4.20 apresenta graficamente os critérios enumerados anteriormente.

Figura 4.20 – Definição das *labels*



Fonte: do autor.

As demais situações onde não há o cenário apresentado, define-se o *label* como sendo "inativo", ou seja, estes cenários indicam que não há previsão a ser realizada pelo modelo.

Como o objetivo é prever com base nos dados atuais a direção da vela seguinte, a coluna das *labels* no *dataframe*, deve ser deslocada para cima, ou seja, retardar os dados em $n-1$.

4.5.3 Validação cruzada

A Fig. 4.21 apresenta um resumo conciso das informações do *dataframe*, obtidas pela função "*dataframe.info()*" da biblioteca pandas do *python*. A quantidade de dados sofreu uma redução de 100.000 dados para 99.800 dados, essa redução é causada pela exclusão dos dados "NaN" (dados faltantes) do *dataframe*.

Destacado pelo retângulo verde na Fig. 4.21 estão as *features* que serão utilizadas para treinamento do modelo, a depender ainda do processo de escolha das melhores *features* e/ou aplicação do PCA. As colunas de dados “time” e “direção” são informações complementares utilizadas para validação de resultados.

Figura 4.21 – *Dataframe* contendo as *features* e *label*

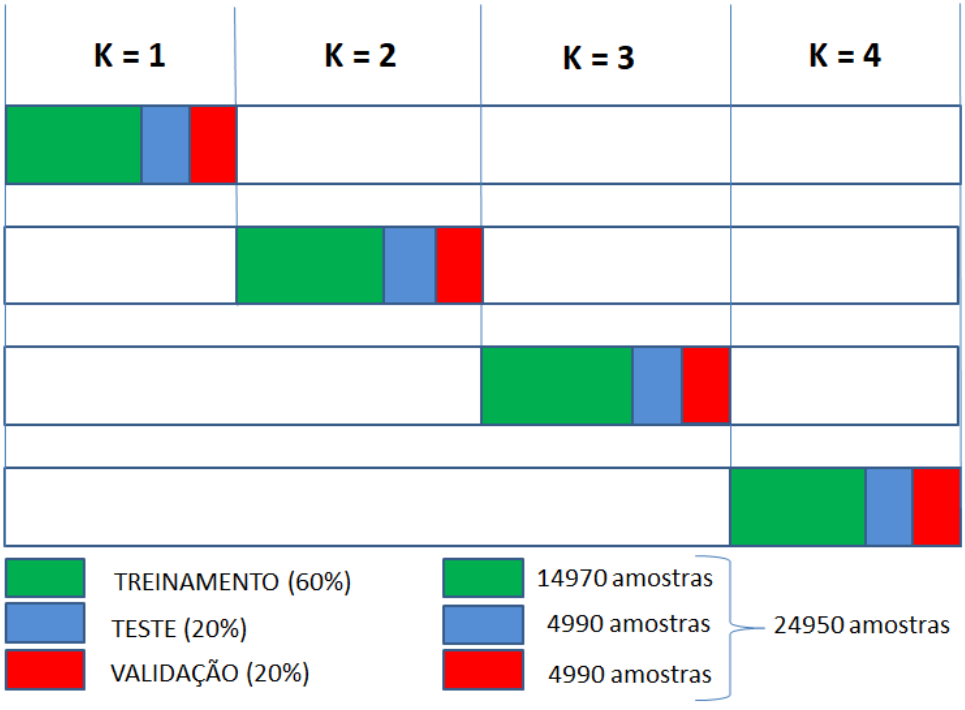
```
Data columns (total 15 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   time     99815 non-null  datetime64[ns]
 1   direcao  99815 non-null  object
 2   range    99815 non-null  float64
 3   RR       99815 non-null  float64
 4   RCP      99815 non-null  float64
 5   RP       99815 non-null  float64
 6   RV       99815 non-null  float64
 7   VT       99815 non-null  float64
 8   RVT      99815 non-null  float64
 9   DBB      99815 non-null  float64
 10  DBBS     99815 non-null  float64
 11  DBBI     99815 non-null  float64
 12  DSMA     99815 non-null  float64
 13  DSG      99815 non-null  float64
 14  Label    99815 non-null  int32
```

Features

Fonte: do autor.

O conjunto de dados será subdividido em 4 (quatro) partes iguais de modo que seja possível realizar a validação cruzada do modelo. A Fig. 4.22 ilustra como será dada a divisão dos dados. É importante destacar que a ordem cronológica dos dados é mantida, ocorre apenas a divisão do banco de dados em períodos iguais.

Figura 4.22 – Divisão dos dados



Fonte: do autor.

Para cada subconjunto de dados tem-se uma média de 225 amostras de treinamento para cada uma das classes "alta" e "baixa", que é obtida conforme descrito na Seção 4.5.2.

4.5.4 Normalização dos dados

De posse do conjunto de dados, optou-se pela normalização dos dados de treinamento utilizando o método *Min-Max Scaler*, de forma que a variação dos dados esteja compreendida entre -1 e 1. Fato que justifica a escolha pela normalização é que este método mantém a distribuição original dos dados, ao contrário da padronização (*Z-score*), que pode alterar a distribuição original dos dados.

4.5.5 Seleção de *features*

No universo do Aprendizado de Máquinas existe uma preocupação por parte dos pesquisadores relacionada a "Maldição da dimensionalidade". Trata-se de um conceito que diz que para uma certa quantidade de amostras, existe um número máximo de *features* a partir do qual o desempenho do classificador irá degradar, ao invés de melhorar.

Neste sentido, para um conjunto grande de dados é desejável selecionar um subconjunto adequado de *features* de modo a reduzir a dimensionalidade. Duas soluções serão abordadas neste trabalho, a primeira é a seleção de características pelo método *SelectKBest* e a outra é o PCA.

4.5.5.1 *SelectKBest*

O método de seleção de features é uma técnica que procura no conjunto de dados aqueles que melhor contribuem para o treinamento do modelo. É um procedimento importante de pré-processamento de dados, que contribui para redução do tempo de treinamento e melhoria da precisão.

A biblioteca *scikit-learn*[2] do *python* oferece a classe "*feature_selection.SelectKBest*"[3]. Seu funcionamento é baseado em testes estatísticos univariados, que avalia a relevância que cada *feature* tem com a variável de saída. O Código 4.4 foi o utilizado para a seleção das melhores *features*.

[2] https://scikit-learn.org/stable/index.html
[3] https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html

Código 4.4 – *SelectKBest*

```
# Seleção das melhores features
# Função SlectKbest
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif, mutual_info_classif

X_train_selectkbest = X[y_train['Label']!=2]
y_train_selectkbest = y_train[y_train['Label']!=2]

features_select = SelectKBest(f_classif, k = 'all')
features_select.fit_transform(X_train_selectkbest,
    y_train_selectkbest.Label.values)
features_scores = features_select.scores_
features_list = X.columns
raw_pairs = zip(features_list, 100*features_scores/sum(
    features_scores))
ordered_pairs = list(reversed(sorted(raw_pairs, key=lambda x: x[1])))
features_best= pd.DataFrame(ordered_pairs, columns = ['FEATURES', '
    Importância 0-100%'])
features_best.set_index('FEATURES', inplace = True)
fig = features_best.plot(kind = 'bar', figsize=(15,7)) # plota as
    melhores features em grafico de barras
melhores_features = features_best[features_best['Importância 0-100%'
    ].cumsum() < soma_kbest].index
```

Fonte: do autor.

4.5.5.2 PCA

O PCA é um método que realiza a transformação dos dados de modo a extrair as informações mais importantes dos dados em forma de componentes principais, o método condensa a informação contida nos dados originais em um subconjunto de dimensionalidade menor com uma perda mínima de informação.

A biblioteca *scikit-learn* do *python* oferece a classe "*decomposition*.PCA"[4], o Código 4.5 foi utilizado para a sua aplicação no conjunto de dados.

Código 4.5 – PCA

```
# Seleção das melhores features
# Função PCA
from sklearn.decomposition import PCA
n_components= X.shape[1]
pca = PCA(n_components= n_components)
pca.fit(X)
X_train_pca = pd.DataFrame(pca.transform(X))
componentes_principais = pd.Series(np.round(pca.
    explained_variance_ratio_,2))
componentes_principais.index += 1
componentes_principais = componentes_principais.add_suffix('  ')
componentes_principais = pd.DataFrame(componentes_principais, columns
     = ['Importância 0-100%'])
X_train_pca.columns = [componentes_principais.index]
principais_componentes = componentes_principais[
    componentes_principais['Importância 0-100%'].cumsum() <= soma_pca
    ].index
X_train_pca = X_train_pca[principais_componentes]
fig = componentes_principais.plot(kind = 'bar', figsize=(15, 7))#
    plota as melhores features em grafico de barras
fig.set(xlabel='COMPONENTES PRINCIPAIS')
```

Fonte: do autor.

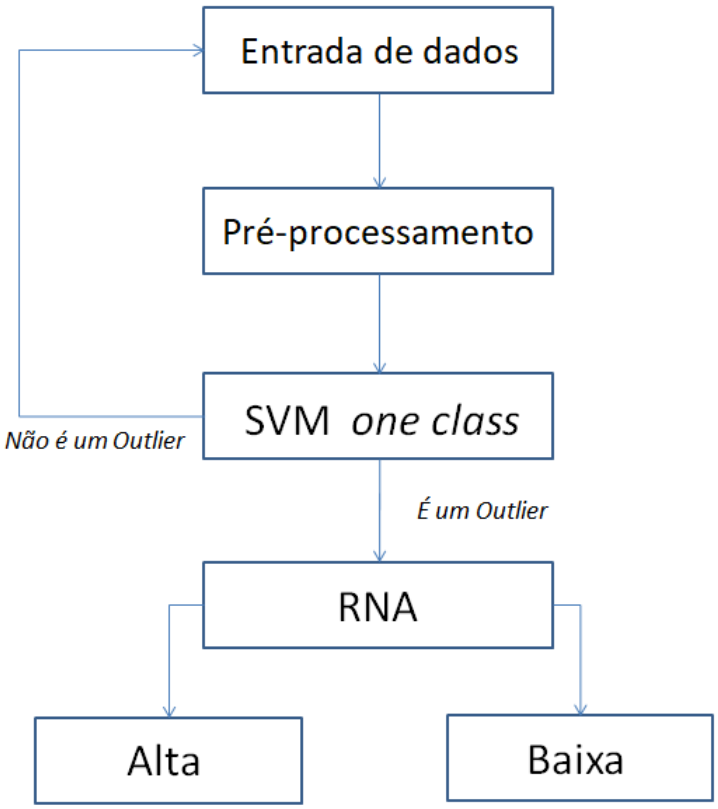
4.6 Modelo classificador

O modelo classificador é constituído por uma etapa de pré-classificação realizada pelo SVM *one class* e uma etapa de classificação realizada pela RNA, conforme exemplificado na Fig. 4.23. O SVM *one class* é responsável pela classificação primária, onde o objetivo é separar a classe rotulada como "inativa", sendo esta a classe predominante dos dados. O SVM é treinado

[4] https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

com todo o conjunto de dados e os *outliers* são classificados conforme parâmetro que define a fração de *outliers* presente nos dados (*nu*). Uma vez identificado um *outlier* pelo SVM, este é submetido a RNA MLP para a classificação entre "alta" ou "baixa".

Figura 4.23 – Fluxograma dos classificadores



Fonte: do autor.

O *scikit-learn* oferece os pacotes para trabalhar com os classificadores SVM *one class*[5] e MLP *Classifier*.[6]

O MLP *Classifier* detém um método chamado "$predict\_proba(X)$" que retorna a probabilidade da previsão para cada uma das classes. A aplicação deste método no modelo retornará uma tupla de valores com as probabilidades para a classe "alta" e "baixa" respectivamente. Neste sentido, é possível definir um limiar para aceitar ou descartar a previsão de determinada classe, os valores retornados pela função acima do limiar pré definido remetem a respectiva classe.

[5] https://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html
[6] https://scikit-learn.org/stable/modules/generated/sklearn.neural$_n etwork.MLPClassifier.html$

4.7 Testes empíricos

O objetivo a ser alcançado com os testes empíricos é definir os melhores parâmetros e hiperparâmetros do modelo, dentre eles:

1. Entrada de dados - serão testados como conjunto de dados de entrada as *features* da última vela, das últimas 2 (duas) velas e das últimas 3 (três) velas, conforme ilustrado na Fig. 4.24;

2. Seleção de *features* - serão testados os métodos *SelectKbest* e PCA;

3. Hiperparâmetros dos classificadores - serão testados diferentes valores para os principais hiper-parâmetros do SVM e MLP conforme detalhado a seguir:

   (a) SMV One Class:

   - *Kernel* = *'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'*;
   - *gamma* = *'scale', 'auto'*
   - *nu* = 0.05 até 0.3 com passo de 0.05

   (b) MLP Classifier:

   - *hidden_layer_sizes* = 1 a 3 camadas ocultas com a quantidade de neurônios por camada variando de $[\frac{N^{\circ}features}{2}]$ até $[N^{\circ}_{features} \times 2]$ com passo igual a $[\frac{N^{\circ}features}{2}]$;
   - activation = *'identity', 'logistic', 'tanh', 'relu'*;
   - solver = *'lbfgs', 'sgd', 'adam'*;
   - alpha = 0.00001, 0.0001 e 0.001;
   - *learning_rate* = *''constant'*, *'invscaling'*, *'adaptive'*
   - *max_iter* = 100 até 1000 com passo de 100.

4. Limiar de saída - serão testados diferentes valores de limiar para o resultado da previsão da RNA, variando de 0,5 até 0,99.

Figura 4.24 – Entrada de dados

range	RR	RCP	RP	RV	VT	RVT	DBB	DBBS	DBBI	DSMA	DSG
50.0	2.000000	0.526316	2.800000	1.198318	3.092816	0.933204	440.282206	200.641103	239.641103	50.0	-2.833722
15.0	0.300000	0.272727	0.692308	0.678425	3.719237	1.202540	408.451505	162.475752	245.975752	43.0	1.935811
-115.0	-7.666667	-0.605263	2.304348	0.795880	3.915068	1.052654	346.560544	231.030272	115.530272	-57.0	-124.858527
175.0	-1.521739	0.555556	0.938462	0.903661	7.531959	1.923839	322.194776	-8.652612	330.847388	127.0	86.764997
70.0	0.400000	0.368421	0.357143	0.880599	7.281226	0.966711	352.630715	-55.434642	408.065358	144.0	136.788118

***Features* da ultima vela;**

***Features* das ultimas 2(duas) velas;**

***Features* das ultimas 3(três) velas.**

Fonte: do autor.

Para cada teste realizado considera-se utilizar os mesmos parâmetros e hiperparâmetros para o treinamento de cada um dos 4 (quatro) subconjunto dos dados. Nesta etapa os modelos são avaliados apenas nos dados de teste, a fim de obter o modelo que melhor generaliza. Em etapa posterior a definição do melhor modelo, este será submetido aos dados de validação.

4.8 Seleção do melhor modelo

A métrica de avaliação será a assertividade das previsões conforme equ. (4.13) e a seleção do melhor modelo será baseada na maior assertividade média seguida do menor desvio padrão obtido para os 4 (quatro) conjuntos de dados de teste.

$$assertividade = \frac{PC}{P} \tag{4.13}$$

Onde:

- **PC** é o número de previsões corretas;
- **P** é o número total de previsões;

Por fim, com modelo estabelecido, submete-se aos dados de validação para observar a capacidade de generalização do modelo.

4.9 Comunicação MT5 e Python para Backtesting

Considerando que ambos os softwares, MT5 (Robô EA) e *python* (Modelo de IA), serão executados no mesmo computador, a comunicação entre eles acontecerá por um sistema de escrita e leitura de arquivos em uma pasta local do computador. O sistema de comunicação segue o seguinte fluxo:

1. Etapa 1 (EA MT5) - O robô de investimento (EA) escreve em um arquivo ".csv" os dados da janela deslizante das ultimas 100 velas do WIN. Os dados têm a mesma estrutura do *dataframe* apresentado na Fig.4.5 da secção 4.4. O arquivo é atualizado a cada nova vela;

2. Etapa 2 (Script em *Python*) - Leitura do arquivo com os dados escritos pelo EA e pré-processamento conforme descrito nas secções anteriores para obtenção das features;

3. Etapa 3 - (Script em *Python*) - As melhores features são submetidas ao modelo de IA para realizar a previsão;

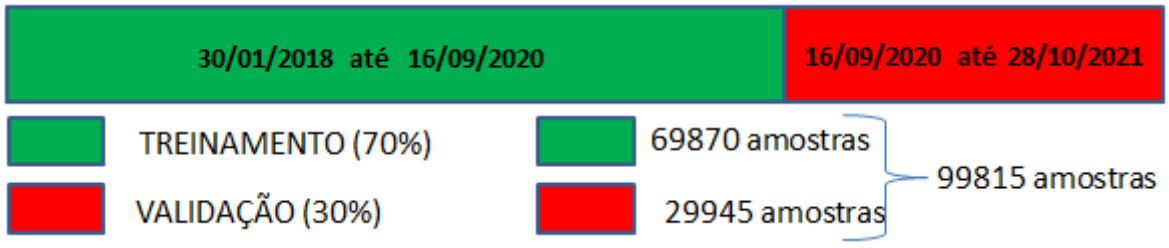
4. Etapa 4 (Script em *Python*) - Se a previsão retornada pelo modelo for da classe "alta" ou "baixa", um arquivo ".txt" é criado e escrito com o resultado da previsão;

5. Etapa 5 (EA MT5) - Se existe o arquivo ".txt" este é lido e excluído pelo EA. Em seguida o EA executada a ordem de compra ou venda, a depender da classe da previsão.

6. Etapa 6 (EA MT5) - Se uma nova vela acaba de fechar, ide a "Etapa 1".

Os sistema de backtesting com o Robô (EA) será utilizado para validação final do modelo. Nesta etapa o modelo será treinado com 70% do conjunto total dos dados e validado com 30% dos dados, conforme demonstrado na Fig. 4.25 a seguir.

Figura 4.25 – Divisão dos dados para validação com backtesting no EA



Fonte: do autor.

5 RESULTADOS

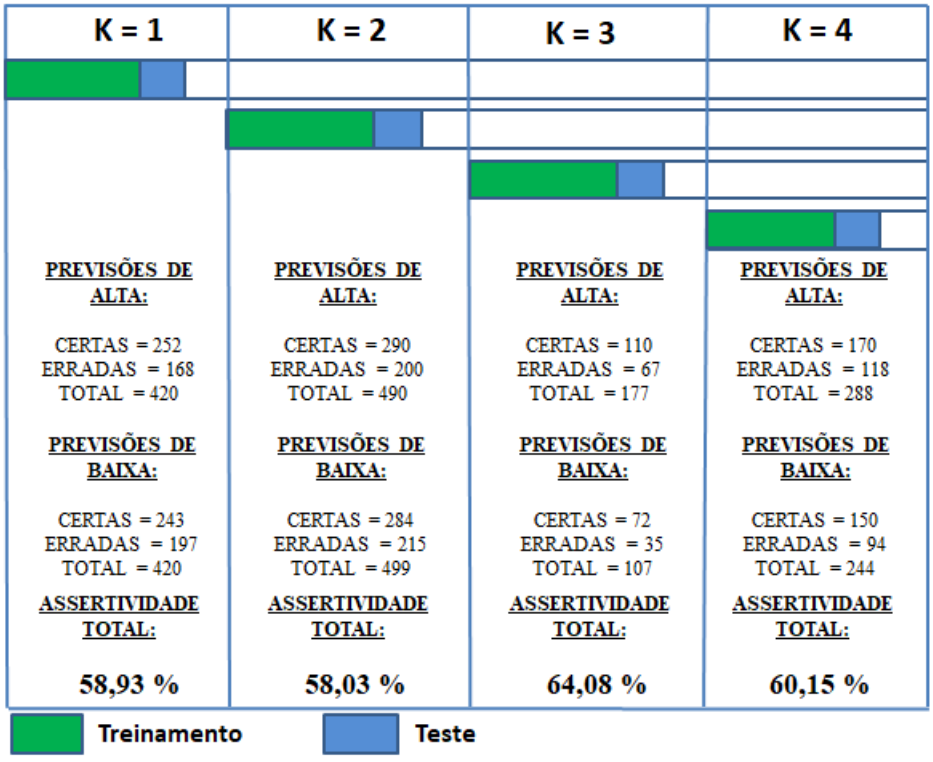
Neste capítulo serão apresentados os principais resultados obtidos da metodologia descrita no Capítulo 4. Vale ressaltar que os experimentos utilizaram os dados da série temporal financeira do WIN, no tempo gráfico de 5(cinco) minutos.

Os resultados são apresentados em duas seções, a primeira (Seção 5.0.1) descreve o modelo que melhor performou, o qual será submetido ao *backtesting* com o robô de investimento e os resultados serão apresentados na segunda seção (Seção 5.0.2).

5.0.1 Melhor modelo

De posse dos resultados da bateria de testes realizados, conforme descrito na Seção 4.7, a seleção do melhor modelo deu-se conforme metodologia detalhada na Seção 4.8. A Fig. 5.1 apresenta as assertividades obtidas e a Tabela 5.1 os respectivos parâmetros e hiper-parâmetros para o modelo em questão.

Figura 5.1 – Assertividade do melhor modelo para os dados de teste



Fonte: do autor.

A assertividade média obtida para o melhor modelo nos dados de teste foi de 60,30 %, com um desvio padrão de 2,31 %.

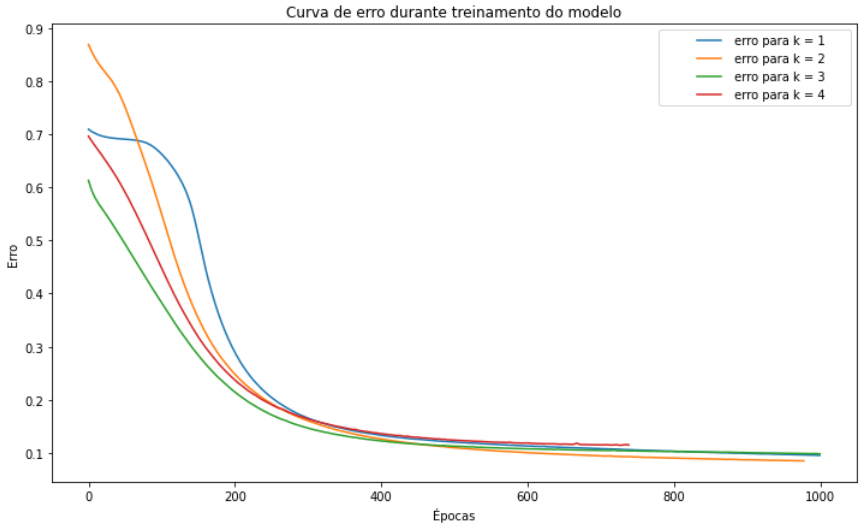
Tabela 5.1 – Parâmetros e hiper-parâmetros do melhor modelo

Parâmetro / Hiper-parâmetros	**Melhor parâmetro**
Entrada de dados	*fatures* da ultima vela
Seleção de *features*	*SelectKBest*
Kernel (SVM *One Class*)	“rbf”
gamma (SVM *One Class*)	“auto”
nu (SVM *One Class*)	0.15
hidden_layer_sizes (MLP *Classifier*)	(6, 3)
activation (MLP *Classifier*)	“tanh”
solver (MLP *Classifier*)	“adam”
alpha (MLP *Classifier*)	0.0001
learnig_rate (MLP *Classifier*)	“adaptative”
max_inter (MLP *Classifier*)	1000
Limiar de saída	0.60

Fonte: do autor.

A seguir, é apresentado na Fig. 5.2 os gráficos das curvas de erro obtidas para o treinamento do modelo.

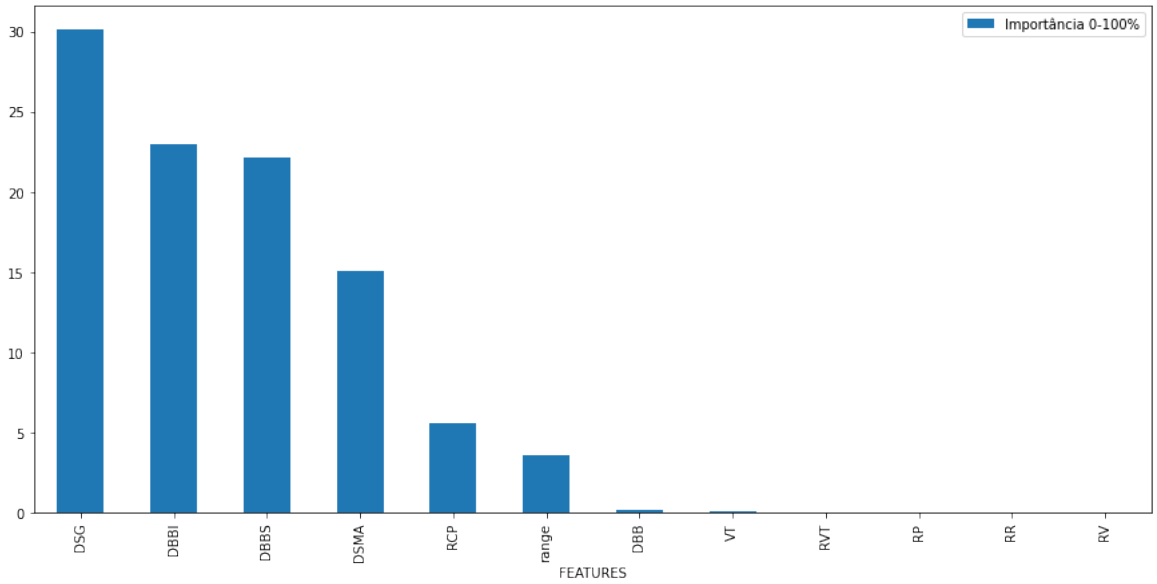
Figura 5.2 – Curva de erro obtido no treinamento do modelo para cada subconjunto k.



Fonte: do autor.

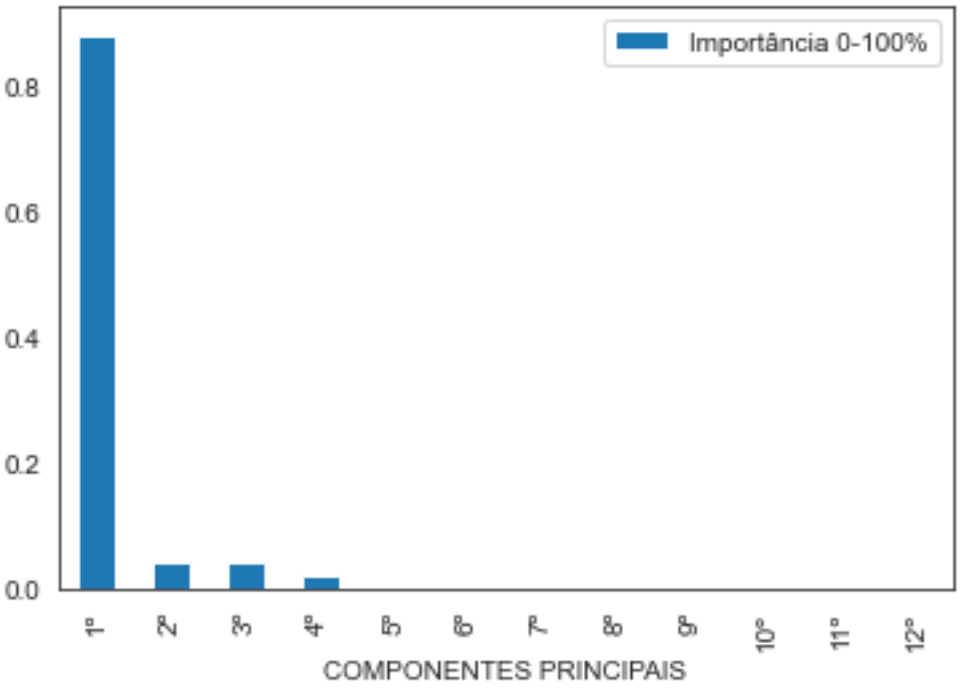
A respeito do método de seleção de *features*, a Fig. 5.3 apresenta o gráfico da pontuação das *features* retornada pelo método SelectKBest e a Fig. 5.4 apresenta o gráfico da pontuação das componentes principais obtida pela PCA. Para ambos os métodos foram utilizados os hiper-parâmetros *default* das classes.

Figura 5.3 – Melhores *features*



Fonte: do autor.

Figura 5.4 – Principais componentes (PCA)

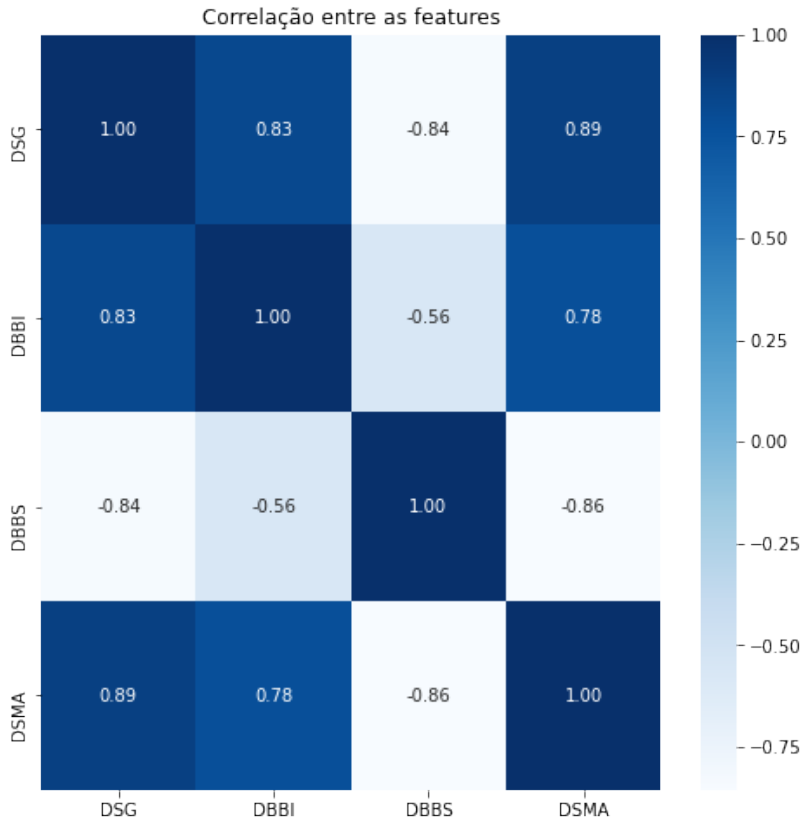


Fonte: do autor.

O SelectKBest foi o método que apresentou os melhores resultados para o modelo, considerou-se a seleção das *features* que somam 90% da importância total, sendo elas: DSG; DBBI; DBBS e DSMA. Realizando uma análise de correlação entre as *features* selecionadas anteriormente, conforme apresentado na Fig. 5.5, observa-se que a *feature* DBBS apresenta em

módulo, as menores correlações com as demais *features*. Observando isso, optou-se por realizar alguns testes empíricos retirando 1 (uma) *feature* por vez de modo a ter sempre as outras 3 (três) *features* como entrada para o modelo. Com estes testes foi possível observar que a exclusão da *feature* DBBS resultou no modelo com melhor assertividade. Neste contexto, presume-se que a melhora da performance obtida com a exclusão da *feature* DBBS está relacionada a baixa correlação desta com as demais.

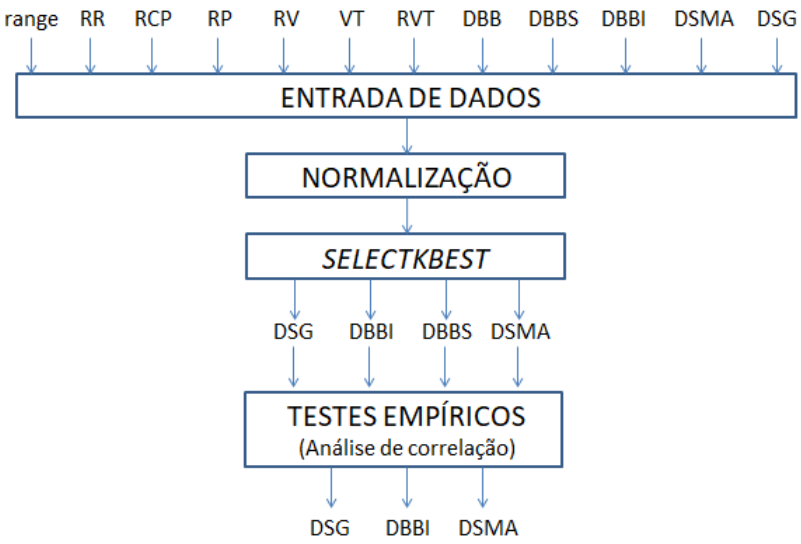
Figura 5.5 – Análise de correlação entre as *features* selecionadas pelo *SelectKBest*



Fonte: do autor.

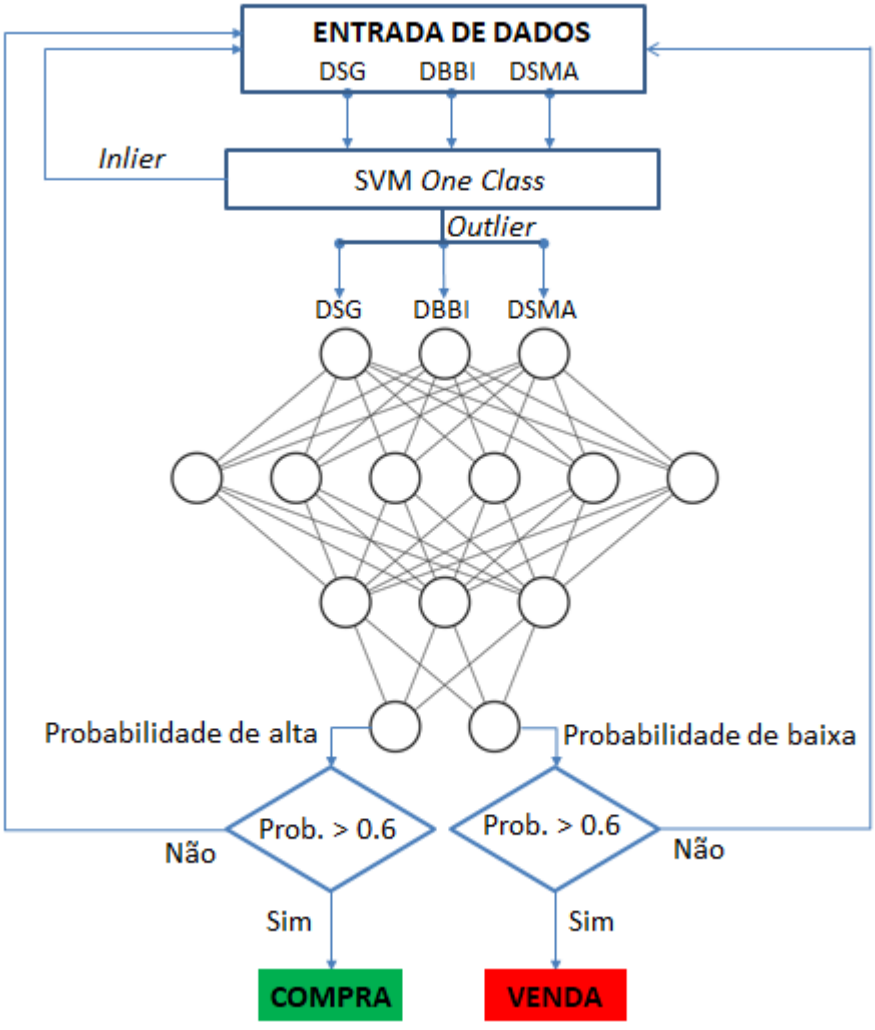
A Fig. 5.6 ilustra de forma exemplificada o procedimento de seleção das principais *features*. Portanto, as *features* DSG, DBBI e DSMA foram utilizadas como entradas no modelo. Em seguida, a Fig. 5.7 apresenta a estrutura do modelo.

Figura 5.6 – Melhores *features* obtidas pelo *SelectKBest*



Fonte: do autor.

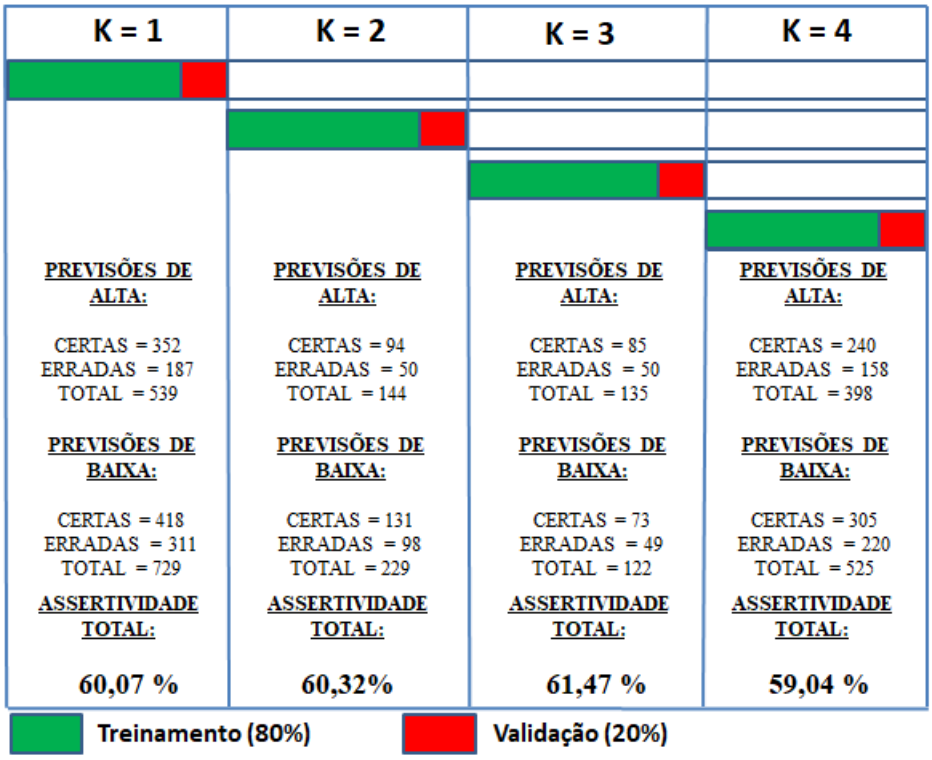
Figura 5.7 – Modelo final



Fonte: do autor.

A Fig. 5.8 a seguir, apresenta o procedimento adotado para validação dos resultados, o qual optou-se por unir os dados de treinamento e teste (Fig. 4.22), de modo a estabelecer o conjunto de dados de treinamento do modelo e submetê-lo aos dados de validação.

Figura 5.8 – Assertividade do modelo para os dados de validação

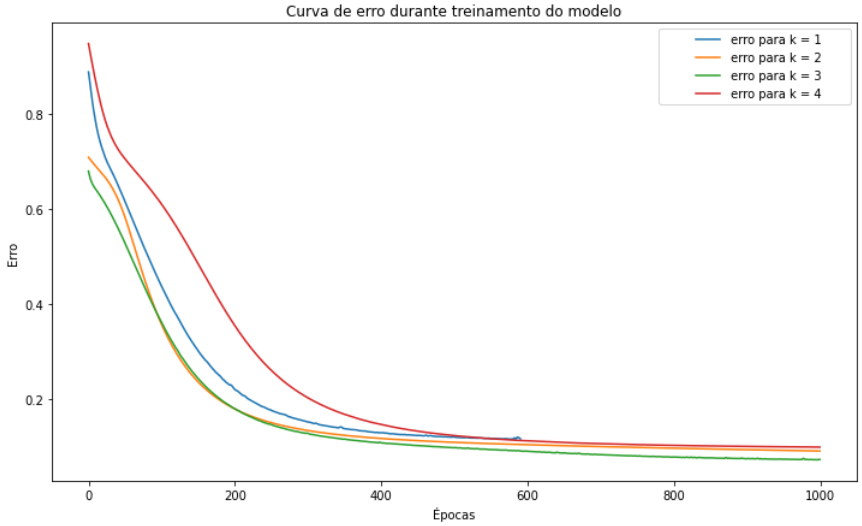


Fonte: do autor.

A assertividade média das previsões obtidas para os dados de validação foi de 60,22 %, com um desvio padrão de 0,86 %.

A Fig. 5.9 apresenta os gráficos das curvas de erro obtidas para o treinamento do modelo considerando a união dos dados de treinamento e teste, conforme ilustrado na Fig. 5.8.

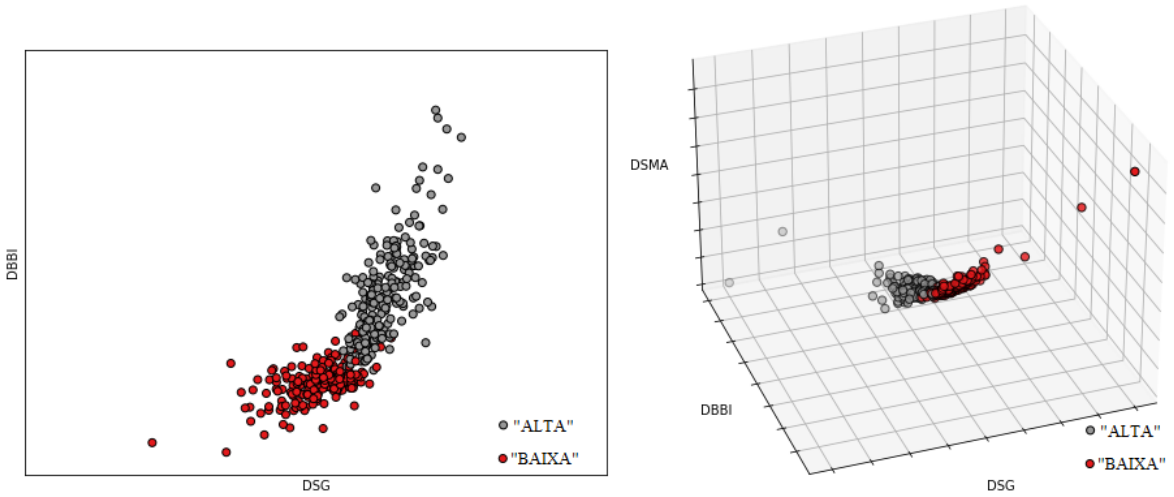
Figura 5.9 – Curva de erro obtido no treinamento do modelo para cada subconjunto k utilizando a união dos dados de treinamento e teste.



Fonte: do autor.

Observa-se na Fig. 5.10 que apesar da separação entre as classes ser nítida, não é linear, o que dificulta muito a classificação. Assim, as 3 três *features* anteriormente selecionadas foram ideais para obter os melhores resultados para o modelo. No entanto, sempre há espaço para melhorias, e como sugestão para tal, deve-se avaliar a inclusão de novas *features*, como as utilizadas por Santos (2020) que utiliza indicadores que medem o índice de força compradora ou vendedora dos movimentos. Além disso, é importante identificar períodos de alta volatilidade dos preços, que geralmente apresentam uma característica de aleatoriedade nos movimentos, o que pode resultar em previsões equivocadas do modelo. Neste contexto, sugere-se avaliar o aplicação de métodos que identifiquem tais cenários de modo a evitar que previsões sejam realizadas em momentos de alta volatilidade. Como foi feito por Conti et al. (2019) que utilizou o expoente de Lyapunov para identificar a presença de caos, ou seja, para identificar cenários de alta volatilidade e consequentemente evitar realizar previsões em momentos inoportunos.

Figura 5.10 – Gráfico de separação das classes



Fonte: do autor.

5.0.2 *Backtesting* com EA

Com o modelo definido, o próximo passo é a validação via *backtesting* na plataforma MT5, utilizando o robô (EA) para simular as negociações de compra e venda conforme procedimento descrito na Seção 4.9.

A princípio é necessário a definição dos parâmetros que se pretende utilizar na execução do *backtesting*. Conforme visto na Seção 2.8.1, a modelagem OHLC por 1 (um) minuto para gerar os *ticks* de movimentação das velas foi usada. Esta escolha justifica-se pelo fato de ser uma modelagem que demanda menor capacidade de processamento e tempo de execução. Outro fato que justifica essa escolha é que a simulação ocorrerá no tempo gráfico de 5 (cinco) minutos, neste caso têm-se 20 (vinte) *ticks* de movimentação por vela, o que garante uma simulação de movimentação aceitável. Outro parâmetro importante a ser definido é o tipo de ajuste da série histórica do WIN, será usado o "WIN$N" que é o método sem nenhum ajuste, ou seja, representa a série histórica nos preços que efetivamente ocorreram no passado, sem ajustes dos *gaps* entre os vencimentos dos contratos. Por fim, determina-se o período em que será executado o backtesting, este por sua vez foi definido conforme Fig. 4.25.

Quando se realiza uma negociação de compra ou venda no mercado financeiro, é de extrema importância definir uma estratégia de gerenciamento de risco, o qual define-se o máximo prejuízo aceitável ("*Stop Loss*") e o lucro ("*Take Profit*") desejado na operação. No robô, estes parâmetros serão definidos em função da amplitude média das velas obtidas dos dados de treinamento, a amplitude é medida em pontos conforme equação (5.1).

$$amplitude_i = high_i - low_i \tag{5.1}$$

Onde:

- *high* é o valor máximo da vela;
- *low* é o valor mínimo da vela;
- *i* é o índice da vela.

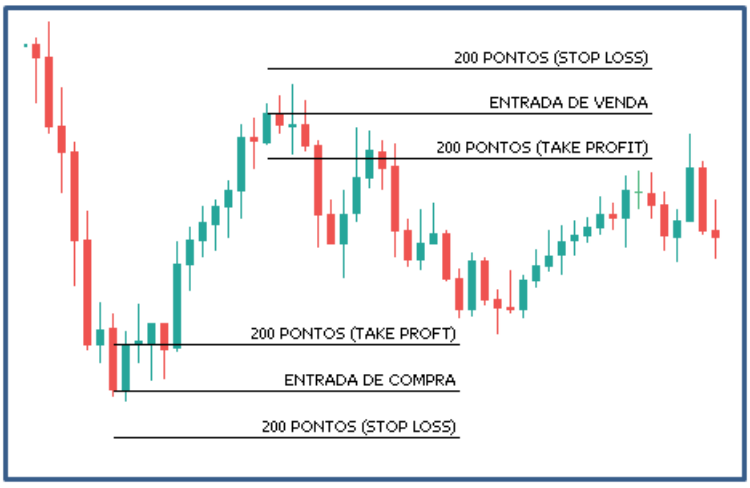
A amplitude média de cada vela, mensurada nos dados de treinamento foi de 196,31 pontos, considerando este valor optou-se por definir 3 (três) *setups* diferentes para testes com o robô, definindo os valores de "*Stop Loss*" (SL) e "*Take Profit*" (TP) de no mínimo 200 (duzentos) pontos. A tabela 5.2 apresenta os *setups* utilizados para teste com o robô e a Fig. 5.11 ilustra os níveis representados no gráfico para o *setup* 1. O volume negociado para todos os *backtestings* será de 1 (um) contrato de WIN que é o volume mínimo para abertura de uma posição de compra ou venda. Com 1 (um) contrato negociado cada *tick* de movimentação (5 pontos) equivale a R$ 1,00 de volume financeiro, neste caso, 200 (duzentos) pontos com 1 (um) contrato negociado equivale a um valor financeiro de R$ 40,00, seja de prejuízo ou lucro.

Tabela 5.2 – *Setups* para *backtestings*

Setup	TP (pontos)	SL (pontos)	Financeiro (R$)
1	200	*200*	40,00 x 40,00
2	400	*200*	80,00 x 40,00
3	200	*400*	40,00 x 80,00

Fonte: do autor.

Figura 5.11 – Níveis de *Stop Loss* e *Take Profit*



Fonte: do autor.

Quando finalizado um *backtesting* no MT5, é emitido um relatório[1], este por sua vez apresenta diversos indicadores que auxiliam na análise da eficiência da estratégia testada. No entanto, serão abordados na análise dos resultados os principais indicadores do relatório, sendo eles: Lucro líquido; Fator de lucro e Negociações com Lucro; os demais indicadores não citados podem ser consultados na página destacada na nota de rodapé. Obteve-se dos *backtestings* realizados, os 3 (três) relatórios para os *setups* da tabela 5.2, cujos resultados são discutidos nas subseções a seguir.

5.0.3 Resultado do *Backtesting* do *setup* 1

A Fig. 5.12 apresenta o resultado retornado para o *backtesting* do *setup* 1.

Figura 5.12 – Resultado para *backtesting* do *setup* 1

Resultados

Qualidade do histórico:	**95%**				
Barras:	**29993**	Ticks:	**599508**	Ativos:	**1**
Lucro Líquido Total:	**623.00**	Rebaixamento Absoluto do Saldo :	**952.00**	Rebaixamento Absoluto do Capital Líquido:	**969.00**
Lucro Bruto:	**56 825.00**	Rebaixamento Máximo do Saldo :	**1 661.00 (24.83%)**	Rebaixamento Máximo do Capital Líquido:	**1 703.00 (25.41%)**
Perda Bruta:	**-56 202.00**	Rebaixamento Relativo do Saldo :	**24.83% (1 661.00)**	Rebaixamento Relativo do Capital Líquido:	**25.41% (1 703.00)**
Fator de Lucro:	**1.01**	Retorno Esperado (Payoff):	**0.22**	Nível de Margem:	
Fator de Recuperação:	**0.37**	Índice de Sharpe:	**0.01**	Z-Pontuação:	**0.55 (41.77%)**
AHPR:	**1.0001 (0.01%)**	Correlação LR :	**0.72**	Resultado OnTester:	**0**
GHPR:	**1.0000 (0.00%)**	Erro Padrão LR :	**362.62**		
Total de Negociações:	**2827**	Posições Vendidas (% e ganhos):	**1497 (51.77%)**	Posições Compradas (% de ganhos):	**1330 (50.00%)**
Ofertas Total:	**5654**	Negociações com Lucro (% of total):	**1440 (50.94%)**	Negociações com Perda (% of total):	**1387 (49.06%)**
		Maior lucro da negociação:	**40.00**	Maior perda na Negociação:	**-41.00**
		Média lucro da negociação:	**39.46**	Média perda na Negociação:	**-40.52**
		Máximo ganhos consecutivos ($):	**14 (557.00)**	Máximo perdas consecutivas ($):	**10 (-407.00)**
		Máxima lucro consecutivo (contagem):	**557.00 (14)**	Máxima perda consecutiva (contagem):	**-407.00 (10)**
		Média ganhos consecutivos:	**2**	Média perdas consecutivas:	**2**

Fonte: do autor.

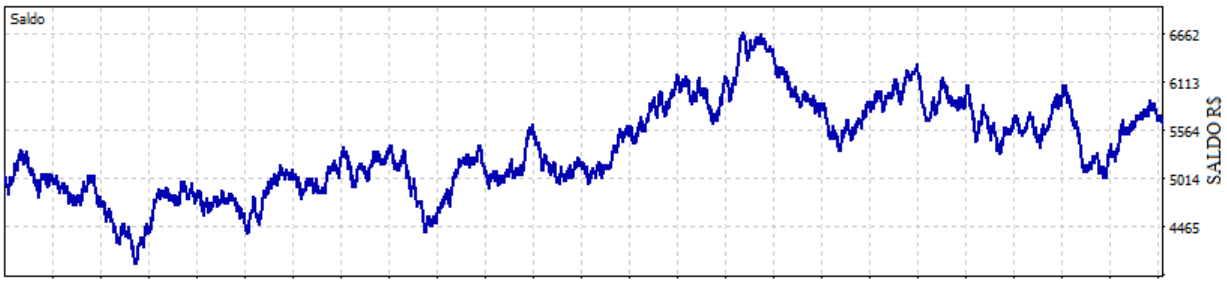
Com base nos resultados obtidos observa-se que o volume financeiro negociado no período analisado foi de R$ 113.027,00 (Lucro Bruto + Perda Bruta). No entanto, o lucro líquido foi extremamente pequeno (Lucro Líquido Total: R$ 623,00) em comparação com o volume financeiro negociado. Este valor de lucro líquido provavelmente não cobre os custos de taxas e emolumentos que são cobradas pelos serviços prestados pela B3 e eventuais taxas das corretoras em um cenário real, considerando o volume negociado no período.

Outro aspecto a ser analisado é o percentual de negociações com lucro que representa 50,94% de assertividade nos trades realizados. Neste caso, constata-se a relevância da análise da acurácia do modelo em ambiente que simule minimamente o mercado financeiro como ele realmente se movimenta.

[1] https://www.metatrader5.com/pt/terminal/help/algotrading/testing_report

Considerando que o saldo financeiro no início do *backtesting* é de R$ 5.000,00, a seguir a Fig.5.13 apresenta o gráfico de evolução de capital para o *setup* 1.

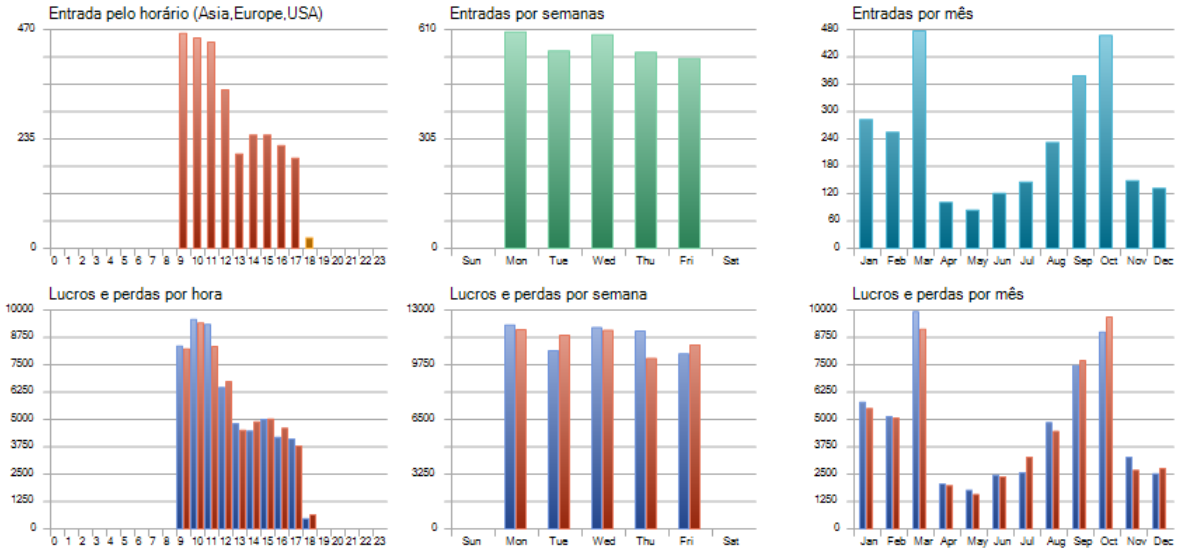
Figura 5.13 – Evolução financeira para o *setup* 1.



Fonte: do autor.

A Fig.5.14 apresenta algumas estatísticas relativas aos resultados obtidos considerando horários, dias da semana e meses do ano. Para os gráficos com barras duplas, a barra azul representa lucro e a vermelha representa prejuízo.

Figura 5.14 – Estatísticas do *backtesting*.



Fonte: do autor.

5.0.4 Resultado do *Backtesting* do *setup* 2

A Fig. 5.15 apresenta o resultado retornado para o *backtesting* do *setup* 2.

Figura 5.15 – Resultado para *backtesting* do *setup* 2

Resultados

Qualidade do histórico:	**95%**				
Barras:	**29993**	Ticks:	**599508**	Ativos:	**1**
Lucro Líquido Total:	**2 485.00**	Rebaixamento Absoluto do Saldo :	**543.00**	Rebaixamento Absoluto do Capital Líquido:	**544.00**
Lucro Bruto:	**62 383.00**	Rebaixamento Máximo do Saldo :	**1 875.00 (22.38%)**	Rebaixamento Máximo do Capital Líquido:	**1 875.00 (22.38%)**
Perda Bruta:	**-59 898.00**	Rebaixamento Relativo do Saldo :	**22.38% (1 875.00)**	Rebaixamento Relativo do Capital Líquido:	**22.38% (1 875.00)**
Fator de Lucro:	**1.04**	Retorno Esperado (Payoff):	**1.10**	Nível de Margem:	
Fator de Recuperação:	**1.33**	Índice de Sharpe:	**0.03**	Z-Pontuação:	**0.23 (18.19%)**
AHPR:	**1.0002 (0.02%)**	Correlação LR :	**0.72**	Resultado OnTester:	**0**
GHPR:	**1.0002 (0.02%)**	Erro Padrão LR :	**534.69**		
Total de Negociações:	**2264**	Posições Vendidas (% e ganhos):	**1155 (36.10%)**	Posições Compradas (% de ganhos):	**1109 (33.18%)**
Ofertas Total:	**4528**	Negociações com Lucro (% of total):	**785 (34.67%)**	Negociações com Perda (% of total):	**1479 (65.33%)**
		Maior lucro da negociação:	**80.00**	Maior perda na Negociação:	**-41.00**
		Média lucro da negociação:	**79.47**	Média perda na Negociação:	**-40.50**
		Máximo ganhos consecutivos ($):	**7 (556.00)**	Máximo perdas consecutivas ($):	**16 (-645.00)**
		Máxima lucro consecutivo (contagem):	**556.00 (7)**	Máxima perda consecutiva (contagem):	**-645.00 (16)**
		Média ganhos consecutivos:	**2**	Média perdas consecutivas:	**3**

Fonte: do autor.

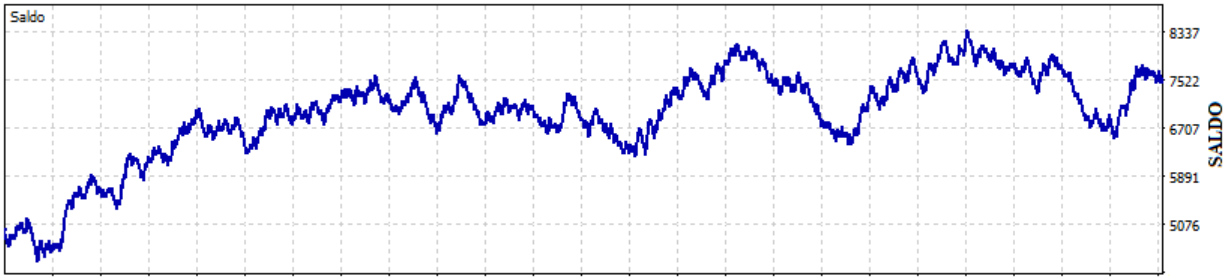
Com base nos resultados obtidos observa-se que o volume financeiro negociado no período analisado foi de R$ 122.281,00 (Lucro Bruto + Perda Bruta). O lucro líquido obtido foi R$ 2.485,00, valor aproximadamente 4 (quatro) vezes maior que o resultado obtido com o *setup* 1 (um). Apesar de obter um lucro líquido maior que o *setup* anterior, quando analisamos o Fator de Lucro, que representa quantas vezes a soma dos lucros excedeu a soma das perdas brutas, observa-se um valor baixo (1,04). Pelo fato do uso de sistemas automatizados para *trading* (EAs) ser um tema pouco abordado na literatura, não há referências de valores que determine o bom desempenho de uma estratégia de *trade* com base no valor do Fator de Lucro. No entanto, existem fóruns de discussões sobre estratégias automatizadas[2], blogs de traders[3] e sites[4] que citam valores maiores ou iguais a 1,75 como sendo um indicativo de um sistema de bom desempenho e lucrativo. Neste contexto, apesar da estratégia utilizando o *setup* 2 (dois) ser lucrativa, ainda assim pode-se considerar que não é uma boa estratégia de *trading* pois apresenta um baixo fator de lucro.

Observa-se também que o percentual de negociações com lucro representa 34,67% de assertividade nos *trades* realizados. Trata-se de um baixo percentual de acertos, mesmo considerando que o a relação risco *x* retorno é de 1 *x* 2.

Considerando que o saldo financeiro no início do *backtesting* é de R$ 5.000,00, a seguir a Fig.5.16 apresenta o gráfico de evolução de capital para o *setup* 2.

[2] https://www.mql5.com/en/forum/188993
[3] https://analyzingalpha.com/profit-factor
[4] https://www.investopedia.com/articles/fundamental-analysis/10/strategy-performance-reports.asp

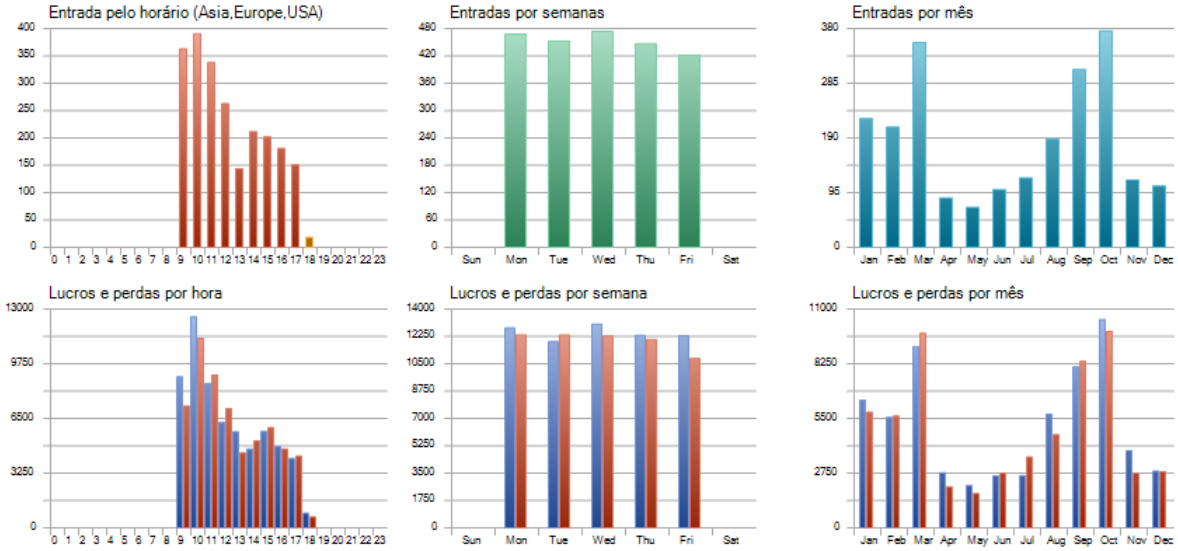
Figura 5.16 – Evolução financeira para o *setup* 2.



Fonte: do autor.

A Fig.5.17 apresenta algumas estatísticas relativas aos resultados obtidos considerando horários, dias da semana e meses do ano. Para os gráficos com barras duplas, a barra azul representa lucro e a vermelha representa prejuízo.

Figura 5.17 – Estatísticas do *backtesting*.



Fonte: do autor.

5.0.5 Resultado do *Backtesting* do *setup* 3

A Fig. 5.18 apresenta o resultado retornado para o *backtesting* do *setup* 3.

Figura 5.18 – Resultado para *backtesting* do *setup* 3

Resultados

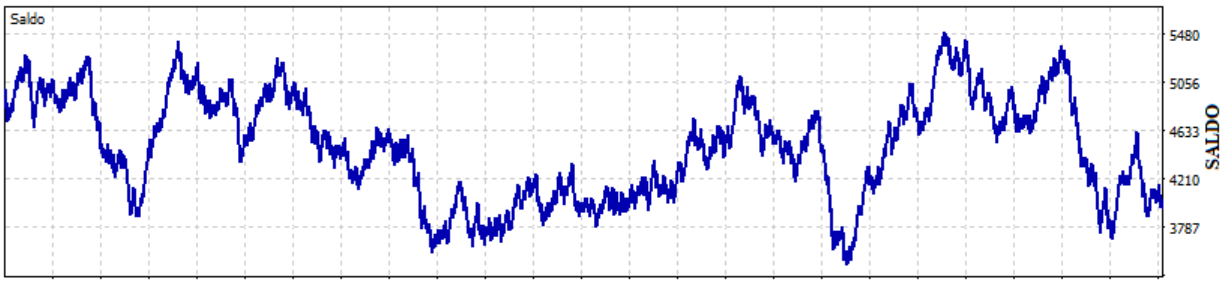
Qualidade do histórico:	**95%**				
Barras:	**29993**	Ticks:	**599508**	Ativos:	**1**
Lucro Líquido Total:	**-1 044.00**	Rebaixamento Absoluto do Saldo :	**1 535.00**	Rebaixamento Absoluto do Capital Líquido:	**1 547.00**
Lucro Bruto:	**55 863.00**	Rebaixamento Máximo do Saldo :	**1 949.00 (36.00%)**	Rebaixamento Máximo do Capital Líquido:	**1 986.00 (36.51%)**
Perda Bruta:	**-56 907.00**	Rebaixamento Relativo do Saldo :	**36.00% (1 949.00)**	Rebaixamento Relativo do Capital Líquido:	**36.51% (1 986.00)**
Fator de Lucro:	**0.98**	Retorno Esperado (Payoff):	**-0.49**	Nível de Margem:	
Fator de Recuperação:	**-0.53**	Índice de Sharpe:	**-0.00**	Z-Pontuação:	**0.76 (55.27%)**
AHPR:	**1.0000 (-0.00%)**	Correlação LR :	**-0.13**	Resultado OnTester:	**0**
GHPR:	**0.9999 (-0.01%)**	Erro Padrão LR :	**449.37**		
Total de Negociações:	**2122**	Posições Vendidas (% e ganhos):	**1084 (67.99%)**	Posições Compradas (% de ganhos):	**1038 (65.32%)**
Ofertas Total:	**4244**	Negociações com Lucro (% of total):	**1415 (66.68%)**	Negociações com Perda (% of total):	**707 (33.32%)**
		Maior lucro da negociação:	**40.00**	Maior perda na Negociação:	**-81.00**
		Média lucro da negociação:	**39.48**	Média perda na Negociação:	**-80.49**
		Máximo ganhos consecutivos ($):	**16 (633.00)**	Máximo perdas consecutivas ($):	**6 (-483.00)**
		Máxima lucro consecutivo (contagem):	**633.00 (16)**	Máxima perda consecutiva (contagem):	**-483.00 (6)**
		Média ganhos consecutivos:	**3**	Média perdas consecutivas:	**1**

Fonte: do autor.

Com base nos resultados obtidos, fazendo uma análise objetiva de desempenho observando apenas o Lucro Líquido Total, observa-se um prejuízo de R$1.044,00 no período, neste caso é possível considerar de imediato que a estratégia utilizando o *setup* 3 (três) não tem eficácia. Apesar do *setup* apresentar uma assertividade de 66,68% nos *trades* realizados, como a relação risco *x* retorno é de 2 *x* 1, este percentual de acerto não garante a eficiência da estratégia.

Considerando que o saldo financeiro no início do *backtesting* é de R$ 5.000,00 a seguir a Fig.5.19 apresenta o gráfico de evolução de capital para o *setup* 3.

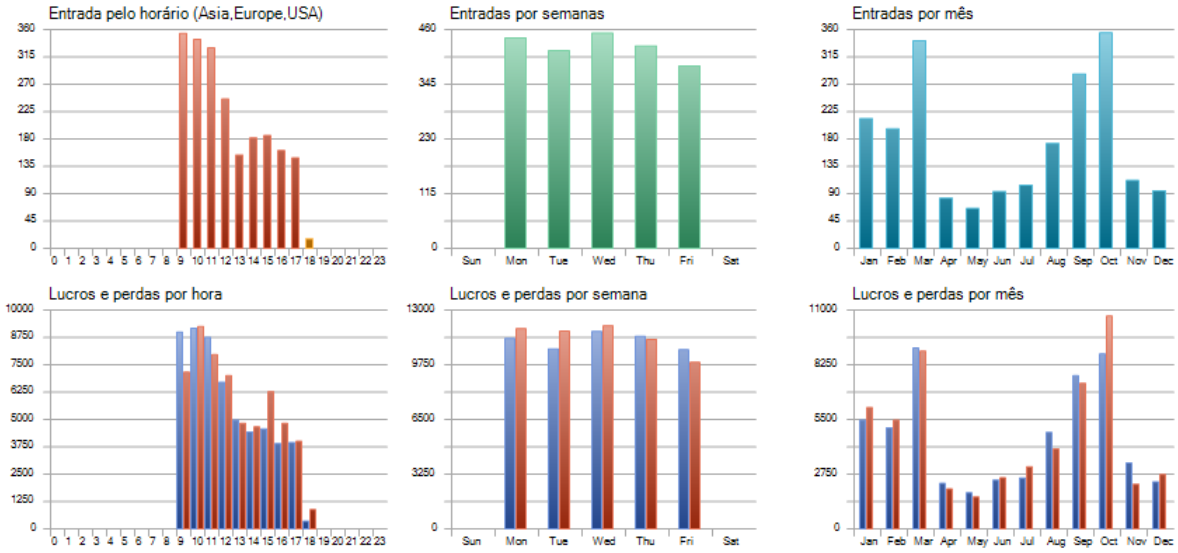
Figura 5.19 – Evolução financeira para o *setup* 3.



Fonte: do autor.

A Fig.5.20 apresenta algumas estatísticas relativas aos resultados obtidos considerando horários, dias da semana e meses do ano. Para os gráficos com barras duplas, a barra azul representa lucro e a vermelha representa prejuízo.

Figura 5.20 – Estatísticas do *backtesting*.



Fonte: do autor.

6 CONCLUSÃO

Este trabalho teve como objetivo desenvolver um sistema de *trading* capaz de prever a direção do próximo *candle* de 5 minutos no ativo WIN da B3. Trata-se de um modelo que busca identificar padrões de reversão no gráfico utilizando para isso técnicas de inteligência artificial, sendo elas o SVM *One Class* e RNA. Além disso, utilizou-se também o filtro Savitzky-Golay para suavizar a série temporal financeira. A principal contribuição do trabalho consiste na avaliação do modelo de IA em um sistema de *backtesting* na plataforma de *trading* MT5, o que permitiu avaliar a eficácia do sistema proposto utilizando a simulação das movimentações da série temporal como ela realmente acontece.

A princípio, ao analisar os resultados de validação do modelo, quando submetido ao banco de dados de validação sem o sistema de *backtesting* pelo MT5, observou-se resultados que apontam para um modelo eficaz na tarefa de previsão da direção do próximo *candle*. No entanto ao avaliar o modelo em um sistema de *backtesting* com o MT5, os melhores resultados, apesar de apontarem para uma estratégia lucrativa, quando analisam-se as métricas retornadas pelo *backtesting* concluí-se que o sistema proposto não é eficiente.

Neste contexto, pode-se concluir que é de extrema importância validar os resultados de uma estratégia de *trading* por meio de ferramentas que simulam a movimentação do mercado financeiro como ela acontece. Além disso, conclui-se que uma boa assertividade nas previsões no modelo de IA não garante que esta seja uma boa estratégia de *trading*.

Em se tratando de estratégias de negociação utilizando modelos de IA, mais importante do que apresentar resultados metodológicos positivos é apresentar resultados realistas. Sendo assim, como sugestões para novos trabalhos propõe-se que sejam analisados trabalhos da literatura que apontam para modelos eficazes na tarefa de previsão de séries temporais financeiras e não aplicaram em suas metodologias uma análise de eficiência com sistemas de *backtesting* para validação dos resultados. Estes por sua vez, devem ser avaliados por um sistema de *backtesting* de modo a obter as métricas que realmente importam para determinar a eficiência da estratégia. Além disso, sugere-se também que sejam realizados testes com novas *features*, explorando principalmente aquelas que são mais utilizadas nos trabalhos científicos e que não foram abordados neste trabalho, como por exemplo as *features* obtidas de indicadores técnicos que remetem ao índice de força dos movimentos do mercado e níveis de sobrecompra e sobrevenda. Outro aspecto importante a ser explorado é a investigação do modelo em tempos gráficos diferentes ao adotado neste trabalho.

REFERÊNCIAS

AMORIM, A. J.; PAULA, A.; ZANE, D. S. Home Broker e BM&F Bovespa: Um Estudo De Caso. **REA - Revista Eletrônica de Administração**, v. 10, n. 1, p. 1–24, 2011. ISSN 1679-9127.

B3. **MANUAL DE DEFINIÇÕESES E PROCEDIMENTOS DOS ÍNDICES DA B3**. 2018. 13 p. Disponível em: <http://www.b3.com.br/data/files/AF/83/C4/BA/25CB7610F157B776AC094EA8/Conceitos-Procedimentos-nov2018.pdf>.

B3. **Características e regras B3**. 2021. Disponível em: <http://www.b3.com.br/pt_br/produtos-e-servicos/negociacao/renda-variavel/mercado-de-acoes/caracteristicas-e-regras.htm>.

B3. **Futuro de Ibovespa**. 2021. Disponível em: <http://www.b3.com.br/pt_br/produtos-e-servicos/negociacao/renda-variavel/futuro-de-ibovespa.htm>.

B3. **Ibovespa B3**. 2021. Disponível em: <http://www.b3.com.br/pt_br/market-data-e-indices/indices/indices-amplos/ibovespa.htm>.

BAGHERI, A.; Mohammadi Peyhani, H.; AKBARI, M. Financial forecasting using ANFIS networks with Quantum-behaved Particle Swarm Optimization. **Expert Systems with Applications**, Elsevier Ltd, v. 41, n. 14, p. 6235–6250, 2014. ISSN 09574174. Disponível em: <http://dx.doi.org/10.1016/j.eswa.2014.04.003>.

BISHOP, C. M. et al. **Neural networks for pattern recognition**. [S.l.]: Oxford university press, 1995.

BOLLINGER, J. Using Bollinger Bands. **Stocks Comodities**, v. 10, n. 2, p. 47–51, 1992.

BRAGA, A. de P.; FERREIRA, A. C. P. de L.; LUDERMIR, T. B. **Redes neurais artificiais: teoria e aplicações**. [S.l.]: LTC editora Rio de Janeiro, Brazil:, 2007.

BROWNLEE, J. **Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python**. [S.l.]: Machine Learning Mastery, 2020.

BUSTOS, O.; POMARES-QUIMBAYA, A. Stock market movement forecast: A Systematic review. **Expert Systems with Applications**, Elsevier Ltd, v. 156, 2020. ISSN 09574174.

CHANG, P. C. et al. A dynamic threshold decision system for stock trading signal detection. **Applied Soft Computing Journal**, Elsevier B.V., v. 11, n. 5, p. 3998–4010, 2011. ISSN 15684946. Disponível em: <http://dx.doi.org/10.1016/j.asoc.2011.02.029>.

CHEN, T. L.; CHEN, F. Y. An intelligent pattern recognition model for supporting investment decisions in stock market. **Information Sciences**, Elsevier Inc., v. 346-347, p. 261–274, 2016. ISSN 00200255. Disponível em: <http://dx.doi.org/10.1016/j.ins.2016.01.079>.

CONTI, J. P. J. et al. **Redes neurais recorrentes e expoente de Lyapunov aplicados a séries temporais financeiras**. Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, 2019.

EDWARDS, R. D.; MAGEE, J.; BASSETTI, W. H. **Technical Analysis of Stock Trends**. [S.l.: s.n.], 2012. 1–566 p. ISBN 9781439898192.

ERFANI, S. M. et al. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. **Pattern Recognition**, Elsevier, v. 58, p. 121–134, 2016. ISSN 00313203. Disponível em: <http://dx.doi.org/10.1016/j.patcog.2016.03.028>.

FAMA, E. F. Session Topic: Stock Market Price Behavior Session Chairman: Burton G. Malkiel Efficient Capital Markets: A Review Of Theory And Empirical Work. **The Journal of Finance**, v. 25, n. 2, p. 383–417, 1970. ISSN 1769-6917.

FGV; ABDI. **Metodologia e Critérios para Composição do Índice**. 2018. 116 p. Disponível em: <http://inteligencia.abdi.com.br/wp-content/uploads/2017/07/2018-09-17_ABDI_relatorio_17-1_metodologia-e-criterios-para-composicao-do-indice_WEB-2.pdf>.

GIACOMEL, F. d. S. Um Método Algorítmico para Operações na Bolsa de Valores Baseado em Ensembles de Redes Neurais para Modelar e Prever os Movimentos dos Mercados de Ações. p. 92, 2016.

GOMEZ, F.; MIIKKULAINEN, R. Robust non-linear control through neuroevolution. 11 2002.

GURESEN, E.; KAYAKUTLU, G.; DAIM, T. U. Using artificial neural network models in stock market index prediction. **Expert Systems with Applications**, Elsevier Ltd, v. 38, n. 8, p. 10389–10397, 2011. ISSN 09574174. Disponível em: <http://dx.doi.org/10.1016/j.eswa.2011.02.068>.

HAYKIN, S. **Redes neurais: princípios e prática**. [S.l.]: Bookman Editora, 2007.

HENRIQUE, B. M.; SOBREIRO, V. A.; KIMURA, H. Literature review: Machine learning techniques applied to financial market prediction. **Expert Systems with Applications**, Elsevier Ltd, v. 124, p. 226–251, 2019. ISSN 09574174. Disponível em: <https://doi.org/10.1016/j.eswa.2019.01.012>.

HOFFMANN, R. **Uma Introdução à Econometria**. [S.l.: s.n.], 2016. ISBN 9788592105709.

HUANG, Q. et al. Biclustering learning of trading rules. **IEEE Transactions on Cybernetics**, v. 45, n. 10, p. 2287–2298, 2015. ISSN 21682267.

JOHNSON RICHARD A., W. D. W. **Applied Multivariate Statistics Analysis**. 6ª ed.. ed. [S.l.]: Duxbury, 2007. 776 p. ISBN 0-13-187715-1.

JOLLIFFE, I. **Principal Component Analysis**. 2nd ed.. ed. [S.l.]: Libray of Congress Cataloging, 2002. 1630–1631 p. ISSN 10780998. ISBN 0-0378-95442-2.

KAO, L. J. et al. Integration of nonlinear independent component analysis and support vector regression for stock price forecasting. **Neurocomputing**, Elsevier, v. 99, p. 534–542, 2013. ISSN 09252312. Disponível em: <http://dx.doi.org/10.1016/j.neucom.2012.06.037>.

KAUFMAN, P. J. **Trading Systems and Methods**. [S.l.]: John Wiley & Sons, 2013. v. 591.

KOTHARI, S. C.; OH, H. Neural Networks for Pattern Recognition. **Advances in Computers**, v. 37, n. C, p. 119–166, 1993. ISSN 00652458.

LEI, K. et al. Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading. **Expert Systems with Applications**, v. 140, p. 1–14, 2020. ISSN 09574174.

LU, C. J. Integrating independent component analysis-based denoising scheme with neural network for stock price prediction. **Expert Systems with Applications**, Elsevier Ltd, v. 37, n. 10, p. 7056–7064, 2010. ISSN 09574174. Disponível em: <http://dx.doi.org/10.1016/j.eswa.2010.03.012>.

LU, C.-J.; LEE, T.-S.; CHIU, C.-C. Financial time series forecasting using independent component analysis and support vector regression. **Decision Support Systems**, v. 47, n. 2, p. 115–125, 2009. ISSN 0167-9236. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167923609000323>.

MARTINS, V. L. Tratamento de dados em multiescala utlizando transformada wavelet para localização de ontos de inflexão em curvas de análise por redissolução potenciométrica. Universidade Federal de Pernambuco, 2005.

MCCULLOCH, W. S.; PITTS, W. A logical calculus nervous activity. **Bulletin of Mathematical Biology**, v. 52, n. l, p. 99–115, 1990. ISSN 00074985.

MENDEL, J. M.; MCLAREN, R. W. Reinforcement-learning control and pattern recognition systems. **Mathematics in Science and Engineering**, Academic Press, Inc., v. 66, n. C, p. 287–318, 1970. ISSN 00765392. Disponível em: <http://dx.doi.org/10.1016/S0076-5392(08)60497-X>.

METAQUOTES, S. C. **Linguagem MQL5 REFERENTE ao terminal do cliente MetaTrader 5**. 2000 – 2021. 6006 p. Disponível em: <https://www.mql5.com/files/pdf/mql5{_}portuguese.>

MISRA, S.; LI, H.; HE, J. **Machine Learning for Subsurface Characterization**. [S.l.]: Elsevier Inc., 2019. v. 148. 442 p. ISBN 9780128177365.

MURPHY, J. J. **John J Murphy - Technical Analysis Of The Financial Markets.pdf**. 1999. 33–4 p. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/20599625>.

OLAZARAN, M. A sociological study of the official history of the perceptrons controversy. **Social Studies of Science**, Sage Publications London, v. 26, n. 3, p. 611–659, 1996.

ORFANIDIS, S. J. **Introduction to signal processing**. [S.l.]: Prentice-Hall, Inc., 1995.

PARK, C.-H.; IRWIN, S. H. The Profitability of Technical Analysis: A Review. **SSRN Electronic Journal**, 2011. ISSN 1556-5068.

PATEL, J. et al. Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. **Expert Systems with Applications**, Elsevier Ltd, v. 42, n. 1, p. 259–268, 2015. ISSN 09574174. Disponível em: <http://dx.doi.org/10.1016/j.eswa.2014.07.040>.

PHAM, H. **Springer Handbook of Engineering Statistics**. 1. ed. [S.l.: s.n.], 2006. 1120 p. ISBN 978-1-84628-288-1.

PIMENTA, A. Métodos Automatizados para Investimento no Mercado Dedicatória. p. 122, 2017. Disponível em: <http://hdl.handle.net/1843/BUOS-ATELWJ>.

PRESS, W. H.; TEUKOLSKY, S. A. Savitzky-Golay Smoothing Filters. **Computers in Physics**, v. 4, n. 6, p. 669, 1990. ISSN 08941866.

RAUDYS, A.; LENČIAUSKAS, V.; MALČIUS, E. Moving averages for financial data smoothing. In: SKERSYS, T.; BUTLERIS, R.; BUTKIENE, R. (Ed.). **Information and Software Technologies**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 34–45. ISBN 978-3-642-41947-8.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, p. 386–408, 1958. ISSN 0033295X.

RUSSELL, S.; NORVIG, P. Artificial intelligence: a modern approach. 2002.

SANTOS, G. C. Algoritmos De Machine Learning Para Previsão De Ações Da B3. p. 94, 2020.

Savitzky, A.; Golay, M. J. E. Smoothing and Differentiation. **Anal. Chem**, v. 36, n. 8, p. 1627–1639, 1964.

SCHAFER, R. W. a Savitzky-Golay Filter? n. July, p. 111–117, 2011.

SCHÖLKOPF, B. et al. Estimating the support of a high-dimensional distribution. **Neural Computation**, v. 13, n. 7, p. 1443–1471, 2001. ISSN 08997667.

SEZER, O. B.; GUDELEK, M. U.; OZBAYOGLU, A. M. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. **Applied Soft Computing Journal**, Elsevier B.V., v. 90, p. 106181, 2020. ISSN 15684946. Disponível em: <https://doi.org/10.1016/j.asoc.2020.106181>.

SINGH, R.; SRIVASTAVA, S. Stock prediction using deep learning. **Multimedia Tools and Applications**, Multimedia Tools and Applications, v. 76, n. 18, p. 18569–18584, 2017. ISSN 15737721.

STEINWART, I.; CHRISTMANN, A. **Support Vector Machines**. New York: Springer Science+Business Media, 2008. 610 p. ISBN 9780387772417. Disponível em: <http://pzs.dstu.dp.ua/DataMining/svm/bibl/Support_Vector.pdf>.

TSAI, C.-F.; HSIAO, Y.-C. Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches. **Decision Support Systems**, v. 50, n. 1, p. 258–269, 2010. ISSN 0167-9236. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167923610001521>.

VAPNIK, V. N. **The Nature of Statistical Learning Theory**. Second edi. [S.l.]: Springer-Verlag New York, 1995. 324 p. ISBN 1441931600,9781441931603.

VAPNIK, V. N.; LERNER, A. Y. Recognition of Patterns with help of Generalized Portraits. **Avtomat. i Telemekh**, v. 8, n. 2, p. 33–165, 1963. Disponível em: <http://www.mathnet.ru/links/6069de6e6122ceb5c41ceadc956149a2/at11885.pdf>.

WANG, J.; WANG, J. Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks. **Neurocomputing**, Elsevier, v. 156, p. 68–78, 2015.

WENG, B.; AHMED, M. A.; MEGAHED, F. M. Stock market one-day ahead movement prediction using disparate data sources. **Expert Systems with Applications**, v. 79, p. 153–163, 2017. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417417301331>.

WOOLDRIDGE, J. M. **Introdução à econometria: uma abordagem moderna**. [S.l.]: Pioneira Thomson Learning, 2006.

YAN, D. et al. Bayesian regularisation neural network based on artificial intelligence optimisation. **International Journal of Production Research**, Taylor & Francis, v. 55, n. 8, p. 2266–2287, 2017. ISSN 1366588X. Disponível em: <http://dx.doi.org/10.1080/00207543.2016.1237785>.

ZHONG, X.; ENKE, D. Forecasting daily stock market return using dimensionality reduction. **Expert Systems with Applications**, v. 67, p. 126–139, 2017. ISSN 0957-4174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0957417416305115>.