ALGORITMOS DE MACHINE LEARNING PARA PREVISÃO DE AÇÕES DA B3

Gustavo Carvalho Santos

Universidade Federal de Uberlândia
Faculdade de Engenharia Elétrica
Pós-Graduação em Engenharia Elétrica

Uberlândia
2020

Gustavo Carvalho Santos

ALGORITMOS DE MACHINE LEARNING PARA PREVISÃO DE AÇÕES DA B3

Dissertação de mestrado apresentada ao Programa de Pós-graduação da Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia como parte dos requisitos para a obtenção do título de Mestre em Ciências.

Área de concentração: Processamento Digital de Sinais.

Orientador: Antônio Cláudio Paschoarelli Veiga
Coorientador: Flávio Luiz de Moraes Barboza

Uberlândia
2020

Ficha Catalográfica Online do Sistema de Bibliotecas da UFU
com dados informados pelo(a) próprio(a) autor(a).

S237 Santos, Gustavo Carvalho, 1996-
2020 Algoritmos de Machine Learning para Previsão de Ações da B3 [recurso eletrônico] / Gustavo Carvalho Santos. - 2020.

Orientador: Antônio Cláudio Paschoarelli Veiga.
Coorientador: Flávio Luiz de Moraes Barboza.
Dissertação (Mestrado) - Universidade Federal de Uberlândia, Pós-graduação em Engenharia Elétrica.
Modo de acesso: Internet.
Disponível em: http://doi.org/10.14393/ufu.di.2020.640
Inclui bibliografia.
Inclui ilustrações.

1. Engenharia elétrica. I. Veiga, Antônio Cláudio Paschoarelli,1963-, (Orient.). II. Barboza, Flávio Luiz de Moraes,1980-, (Coorient.). III. Universidade Federal de Uberlândia. Pós-graduação em Engenharia Elétrica. IV. Título.

CDU: 621.3

Bibliotecários responsáveis pela estrutura de acordo com o AACR2:

Gizele Cristine Nunes do Couto - CRB6/2091

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
Coordenação do Programa de Pós-Graduação em Engenharia Elétrica
Av. João Naves de Ávila, 2121, Bloco 3N - Bairro Santa Mônica, Uberlândia-MG, CEP 38400-902
Telefone: (34) 3239-4707 - www.posgrad.feelt.ufu.br - copel@ufu.br

ATA DE DEFESA - PÓS-GRADUAÇÃO

Programa de Pós-Graduação em:	Engenharia Elétrica				
Defesa de:	Dissertação de Mestrado Acadêmico, 749, PPGEELT				
Data:	Nove de setembro de dois mil e vinte	Hora de início:	09:00	Hora de encerramento:	11:35
Matrícula do Discente:	11822EEL010				
Nome do Discente:	Gustavo Carvalho Santos				
Título do Trabalho:	Algoritmos de Machine Learning para Previsão de Ações da B3				
Área de concentração:	Processamento da informação				
Linha de pesquisa:	Processamento digital de sinais				
Projeto de Pesquisa de vinculação:					

Reuniu-se por meio de videoconferência, a Banca Examinadora, designada pelo Colegiado do Programa de Pós-graduação em Engenharia Elétrica, assim composta: Professores Doutores: Flávio Luiz de Moraes Barboza - FAGEN/UFU, coorientador do candidato; Milena Bueno Pereira Carneiro - FEELT/UFU; Geraldo Nunes Silva - IBILCE/UNESP; e Antônio Cláudio Paschoarelli Veiga - FEELT/UFU, orientador do candidato.

Iniciando os trabalhos o(a) presidente da mesa, Dr(a). Antônio Cláudio Paschoarelli Veiga, apresentou a Comissão Examinadora e o candidato(a), agradeceu a presença do público, e concedeu ao Discente a palavra para a exposição do seu trabalho. A duração da apresentação do Discente e o tempo de arguição e resposta foram conforme as normas do Programa.

A seguir o senhor(a) presidente concedeu a palavra, pela ordem sucessivamente, aos(às) examinadores(as), que passaram a arguir o(a) candidato(a). Ultimada a arguição, que se desenvolveu dentro dos termos regimentais, a Banca, em sessão secreta, atribuiu o resultado final, considerando o(a) candidato(a):

Aprovado.

Esta defesa faz parte dos requisitos necessários à obtenção do título de Mestre.

O competente diploma será expedido após cumprimento dos demais requisitos, conforme as normas do Programa, a legislação pertinente e a regulamentação interna da UFU.

Nada mais havendo a tratar foram encerrados os trabalhos. Foi lavrada a presente ata que após lida e achada conforme foi assinada pela Banca Examinadora.

Documento assinado eletronicamente por **Flávio Luiz de Moraes Barboza**, **Professor(a) do Magistério Superior**, em 09/09/2020, às 11:35, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Antonio Claudio Paschoarelli Veiga**, **Professor(a) do Magistério Superior**, em 09/09/2020, às 11:35, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Milena Bueno Pereira Carneiro**, **Professor(a) do Magistério Superior**, em 09/09/2020, às 11:36, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **GERALDO NUNES SILVA**, **Usuário Externo**, em 09/09/2020, às 16:18, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://www.sei.ufu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2204087** e o código CRC **D0D629CB**.

Referência: Processo nº 23117.047931/2020-11 SEI nº 2204087

Este trabalho é dedicado a minha família e amigos.

Agradecimentos

Aos meus pais, **Edilson** e **Vivian** e meu irmão **Guilherme**, pelo amor, conselhos e incentivos que sempre me ofereceram.

Aos meus avós, em especial ao meu avô **Cecílio** que estará eternamente em meu coração.

Ao meu coorientador **Prof. Dr. Flávio Barboza**, pelo apoio, paciência, confiança e ensinamentos.

Ao meu orientador **Prof. Dr. Paschoarelli**, pelo auxílio, atenção e confiança.

A minha namorada **Bianca**, pelo amor, companheirismo e incentivo.

A todos meus amigos, que sempre estiveram comigo.

A todos os professores e professoras que um dia me ensinaram e contribuíram para minha formação.

Ao Programa de Pós-Graduação em Engenharia Elétrica, por possibilitar a realização deste trabalho.

À CAPES pelo apoio financeiro a esta pesquisa.

"Frequentemente a imaginação nos transporta a mundos que nunca existiram, mas sem ela não vamos a parte alguma."

(Carl Sagan)

Resumo

A predição de movimento nos preços do mercado de ações tem sido uma área importante na pesquisa de algoritmos de *machine learning* nos últimos anos, devido a seu conteúdo complexo e dinâmico. Além disso, a volatilidade típica do mercado de ações torna a tarefa da previsão mais difícil. Dessa forma, este trabalho tem como objetivo propor uma metodologia que preveja a tendência das ações de três empresas negociadas na bolsa de valores brasileira. Por conseguinte, deseja-se facilitar o reconhecimento de movimentações no mercado, além de auxiliar a tomada de decisão para investidores. Ademais, este trabalho propõe comparar as predições dos papeis usando três algoritmos de aprendizado de máquina: Random Forest, Redes Neurais Artificiais e Support Vector Machines. Além disso, é proposto o uso de uma função (kernel-SVM) – utilizada para reconhecimento de imagens – para contornar o problema das previsões. Indicadores técnicos (tais como o Índice de Força Relativa, o oscilador estocástico, entre outros), são usados como entradas para treinar os modelos. Por fim, os resultados são avaliados por meio de indicadores de performance como acurácia, precisão, *recall* e especificidade. A eficiência dos modelos depende da técnica de separação das amostras de treino e teste empregadas, variando entre 18% e 76% de acurácia para técnica de separação temporal, contra 71% e 97% da técnica de separação aleatória.

Palavras-chave: Inteligência Artificial. Aprendizado de Máquina. Ibovespa. Classificação. Mercado de Ações.

Abstract

The prediction of movement in stock market prices has been an interesting area in the research of machine learning algorithms in recent years, due to its complex and dynamic content. In addition, typical stock market volatility makes forecasting more difficult. Thus, this work aims to propose a methodology that predicts the trend of the shares of three Brazilian companies traded on the Brazilian stock exchange. Therefore, we want to facilitate the recognition of movements in the market, in addition to assisting decision-making for investors. Furthermore, this work proposes to compare the predictions of the assets using three powerful machine learning algorithms, known as Random Forest, Artificial Neural Networks and Support Vector Machines. In addition, we propose to use a function (kernel-SVM) – used for image recognition – to solve the prediction problem. Technical indicators (such as the Relative Strength Index, the stochastic oscillator, among others) are used as inputs to train the models. Finally, we evaluate perfomance indicators such as accuracy, precision, recall and specificity. The efficiency of the models depends on the technique of separating the training and test samples used, varying between 18% and 76% of accuracy for the temporal separation technique against 71% and 97% of the random separation technique.

Keywords: Artificial Intelligence. Machine Learning. Ibovespa. Classification. Stock Market.

Lista de ilustrações

Figura 1 – Divisão em uma variável preditora contínua X_i, usando o ponto de divisão c. 31
Figura 2 – Divisão em uma variável preditora categórica X_i, utilizando $S \subset S_i$. . . 32
Figura 3 – Hiperplano SVM. 34
Figura 4 – Perceptron. 37
Figura 5 – Arquitetura de uma RNA. 38
Figura 6 – Gráfico de uma função de ativação ReLU. 38
Figura 7 – Metodologia proposta. 43
Figura 8 – Séries temporais dos dados de fechamento. 44
Figura 9 – Distribuição aleatória das classes. 45
Figura 10 – Distribuição temporal das classes. 46
Figura 11 – Série temporal do ativo VALE3 suavizada exponencialmente. 48
Figura 12 – Exemplo de uma curva *ROC*. 52
Figura 13 – Validação cruzada para 5 folds. 52
Figura 14 – Acurácia Média usando a técnica de embaralhamento. 57
Figura 15 – Acurácia Média usando a ordem temporal. 59
Figura 16 – Curvas ROC VALE3 utilizando embaralhamento. 60
Figura 17 – Curvas ROC PETR4 utilizando embaralhamento. 61
Figura 18 – Curvas ROC ITUB4 utilizando embaralhamento. 62
Figura 19 – Curvas ROC VALE3 utilizando a divisão temporal. 63
Figura 20 – Curvas ROC PETR4 utilizando a divisão temporal. 64
Figura 21 – Curvas ROC ITUB4 utilizando a divisão temporal. 65
Figura 22 – Taxa de Acertos média para horizonte de 30 dias utilizando embaralhamento. 69
Figura 23 – Taxa de Acertos média para horizonte de 60 dias utilizando embaralhamento. 69
Figura 24 – Taxa de Acertos média para horizonte de 90 dias utilizando embaralhamento. 69

Figura 25 – Taxa de Acertos média para horizonte de 30 dias utilizando e divisão temporal. 70
Figura 26 – Taxa de Acertos média para horizonte de 60 dias utilizando divisão temporal. 70
Figura 27 – Taxa de Acertos média para horizonte de 90 dias utilizando divisão temporal. 71

Lista de tabelas

Tabela 1 – Resultados para dados *Samsung*. 25
Tabela 2 – Resultados para dados *Apple*. 25
Tabela 3 – Resultados para dados GE. 25
Tabela 4 – Dados de treinamento obtidos aleatoriamente. 44
Tabela 5 – Dados de validação obtidos aleatoriamente. 45
Tabela 6 – Dados de treinamento obtidos usando a ordem temporal das séries históricas. 46
Tabela 7 – Dados de validação obtidos usando a ordem temporal das séries históricas. 47
Tabela 8 – Previsões 30 dias utilizando divisão aleatória. 56
Tabela 9 – Previsões 60 dias utilizando divisão aleatória. 56
Tabela 10 – Previsões 90 dias utilizando divisão aleatória. 57
Tabela 11 – Previsões 30 dias utilizando divisão temporal. 58
Tabela 12 – Previsões 60 dias utilizando divisão temporal. 58
Tabela 13 – Previsões 90 dias utilizando divisão temporal. 59
Tabela 14 – Validação cruzada para ações VALE3 utilizando embaralhamento. . . . 66
Tabela 15 – Validação cruzada para ações PETR4 utilizando embaralhamento. . . . 66
Tabela 16 – Validação cruzada para ações ITUB4 utilizando embaralhamento. . . . 67
Tabela 17 – Validação cruzada para ações VALE3 utilizando divisão temporal. . . . 67
Tabela 18 – Validação cruzada para ações PETR4 utilizando divisão temporal. . . . 68
Tabela 19 – Validação cruzada para ações ITUB4 utilizando divisão temporal. . . . 68

Sumário

1 INTRODUÇÃO 19
1.1 Introdução à Pesquisa 19
1.2 Objetivos e Desafios da Pesquisa 19
1.3 Hipóteses 20
1.4 Estrutura deste Trabalho 20

2 LITERATURA RELACIONADA 23
2.1 Introdução 23
2.2 Hipótese do Mercado Eficiente 23
2.3 Algoritmos de Machine Learning 24
2.4 Considerações Finais deste Capítulo 26

3 ASPECTOS TEÓRICOS 29
3.1 Random Forest 29
3.1.1 Introdução às árvores de Classificação e Regressão 31
3.2 Support Vector Machine 33
3.2.1 SVM-RBF 35
3.2.2 SVM-GHI 36
3.3 Redes Neurais Artificiais 36
3.4 Ferramentas Utilizadas 39
3.4.1 *Pandas* 39
3.4.2 *NumPy* 39
3.4.3 TA-Lib 39
3.4.4 Scikit-Learn 40
3.4.5 *Keras* 40
3.5 Considerações Finais 40

4 METODOLOGIA DESTE TRABALHO 43

4.1 Introdução . . . 43
4.2 Base de Dados . . . 44
4.3 Pré-processamento dos dados . . . 47
4.4 Extração dos Atributos . . . 48
4.5 Método para a Avaliação das Predições . . . 50
4.5.1 Curva Característica de Operação do Receptor . . . 51
4.5.2 Validação Cruzada utilizando $K - folds$ . . . 52
4.6 Considerações Finais deste Capítulo . . . 53

5 ANÁLISE DOS RESULTADOS . . . 55
5.1 Resultados Obtidos Utilizando a Técnica de Separação Aleatória 55
5.2 Resultados Obtidos Utilizando a Técnica de Separação Temporal 57
5.3 Curvas ROC . . . 59
5.4 Validação Cruzada . . . 65
5.5 Comparação dos Modelos Desenvolvidos . . . 68
5.6 Considerações Finais deste Capítulo . . . 71

6 CONCLUSÃO . . . 73
6.1 Introdução . . . 73
6.2 Principais Contribuições . . . 73
6.3 Trabalhos Futuros . . . 73

REFERÊNCIAS . . . 75

APÊNDICES 79

APÊNDICE A – CÓDIGOS FONTE . . . 81
A.1 Pré-Processamento dos Dados . . . 81
A.2 Divisão Treinamento e Teste . . . 83
A.2.1 Divisão Aleatória . . . 84
A.2.2 Divisão Temporal . . . 84
A.3 Random Forest . . . 85
A.4 SVM-GHI . . . 85
A.5 SVM-RBF . . . 86
A.6 RNA . . . 87
A.7 Cálculo das Curvas ROC . . . 89
A.8 Validação Cruzada utlizando $K - folds$ . . . 90

Capítulo 1

Introdução

1.1 Introdução à Pesquisa

Conforme Khaidem, Saha e Dey (2016), entrever os movimentos dos preços do mercado de ações é um desafio complexo em virtude das várias indeterminações envolvidas e as diversas variáveis que controlam o valor de mercado em um determinado dia, como condições econômicas, sentimentos dos investidores em relação a uma empresa em particular e eventos políticos. Dessa forma, os mercados de títulos são suscetíveis a mudanças rápidas, causando flutuações aleatórias no preço das ações.

As séries do mercado de valores são geralmente dinâmicas, caóticas (envolvem diversas variáveis) e ruidosas e, desse modo, a direção dos preços no mercado de ações é classificada como um processo aleatório, com flutuações mais acentuadas em janelas de tempo curto. No entanto, segundo Khaidem, Saha e Dey (2016), alguns ativos geralmente tendem a desenvolver tendências lineares em longo prazo. Assim, devido à natureza caótica e altamente volátil do comportamento dos papéis, os investimentos neles possuem alto risco. Para minimizar esse fato, o conhecimento do movimento dos preços no futuro pode vir a ser de grande importância.

1.2 Objetivos e Desafios da Pesquisa

Este trabalho objetiva prever a direção do preço das ações de três empresas brasileiras presentes na B3, como uma alternativa aos preços de fechamento, uma vez que pode ajudar a minimizar os riscos de investimento das operações. Para isso, este estudo utiliza três algoritmos de *machine learning* - *random Forest* (RF), redes neurais artificiais (RNA) e *support vector machines* (SVM). Além disso, são definidas as variáveis que mais influenciam nas tendências dos preços das participações diárias das seguintes empresas brasileiras: Vale (VALE3), Petrobras (PETR4), Itaú Unibanco (ITUB4).

Por fim, são estipulados parâmetros estatísticos para comparar a eficiência dos algoritmos propostos. Dessa forma, é possível analisar por meio de medidas (acurácia, precisão,

recall, especificidade, dentre outras) os melhores modelos desenvolvidos.

1.3 Hipóteses

A hipótese deste trabalho reside na comparação de métodos e algoritmos de *machine learning* aplicados à previsão da tendência dos preços de ativos presentes na bolsa de ações brasileira. Busca-se reproduzir, nos ativos citados, a alta performance obtida em trabalhos como o de Khaidem, Saha e Dey (2016) (84.5% e 94.5% de acurácia) e Basak et al. (2019) (55% a 95% de acurácia), além de outros citados no capítulo 2. Assim sendo, são apresentadas duas técnicas de manipulação dos dados de treinamento e validação dos modelos (divisão aleatória e divisão temporal), além de quatro algoritmos de aprendizado de máquina. A pesquisa também procura entender a eficiência dos modelos para diferentes períodos de previsão: 30, 60 e 90 dias úteis. Portanto, de forma geral, são investigadas as possíveis vantagens e desvantagens de cada modelo de aprendizado de máquina proposto.

1.4 Estrutura deste Trabalho

Essa dissertação está dividida em 6 capítulos, da seguinte maneira:

O capítulo 1 institui o tema alvo da pesquisa, apresenta quais os objetivos pretendem ser alcançados com o trabalho, que envolve a pesquisa de algoritmos de *machine learning* aplicados à previsão do mercado de ações brasileiro. Apresenta, também, os desafios, que incluem a comparação de diferentes técnicas de aprendizado e a avaliação de parâmetros estatísticos dos resultados. Por conseguinte, orienta sobre a forma com que este trabalho escrito foi estruturado para facilitar a compreensão do todo.

O capítulo 2 traz o panorama geral da análise técnica e fundamentalista, além de descrever em detalhes os trabalhos presentes na literatura relacionados ao tema da pesquisa, enfatizando os estudos que empregam os mesmos algoritmos e técnicas aqui apresentados. Também há a discussão dos resultados obtidos pelos principais referenciais que guiaram o desenvolvimento deste estudo.

O capítulo 3 especifica os algoritmos utilizados: RF, SVM e RNA. Aborda suas formulações matemáticas, descrevendo o funcionamento, vantagens e os aspectos particulares de cada modelo de aprendizado de máquina. Ademais, o capítulo enumera as ferramentas de software que permitiram o desenvolvimento dos modelos de inteligência artificial.

No capítulo 4 é apresentada a metodologia empregada para elaborar os algoritmos de previsão de ações, descrevendo as etapas de pré-processamento, técnicas de divisão das amostras de treinamento e teste e a extração dos atributos das séries históricas de preços. Além disso, são descritos os parâmetros estatísticos que serão usados para avaliar os resultados descritos no capítulo 5.

O capítulo 5 expõe em detalhes todos os resultados atingidos, tratando da eficiência de cada modelo de predição proposto. Os dados são apresentados de forma estruturada em tabelas e gráficos, permitindo a comparação por diferentes perspectivas das particularidades de cada algoritmo e horizonte de previsão utilizados.

O capítulo 6 apresenta a conclusão, as contribuições realizadas por esta pesquisa e proposições de possíveis trabalhos futuros relacionados.

Os apêndices contêm os códigos dos algoritmos de previsão desenvolvidos para cada modelo e horizonte de previsão propostos.

Capítulo 2

Literatura Relacionada

2.1 Introdução

Prever o mercado de ações é uma tarefa complexa e que pode envolver muitos riscos. No entanto, há na literatura diversas abordagens e algoritmos que se propõem a entrever as tendências dos mercados. À vista disso, esse capítulo pretende tratar sobre os conceitos teóricos do mercado financeiro e os principais métodos utilizados para contornar tal obstáculo.

2.2 Hipótese do Mercado Eficiente

A utilização de algoritmos para prever tendências futuras nos preços do mercado de ações contradiz uma regra básica em finanças denominada Hipótese do Mercado Eficiente (HME) (MALKIEL; FAMA, 1970). A teoria afirma que os preços atuais das ações refletem totalmente todas as informações relevantes. Assim, se alguém obtiver uma vantagem aferindo dados históricos de ações, todo o mercado se perceberá desse benefício e, como efeito, o preço do papel será corrigido. No entanto, de acordo com Malkiel (2003), no início do século 21, diversos economistas financeiros e estatísticos começaram a acreditar que os preços das ações são pelo menos parcialmente previsíveis. Uma nova geração de economistas enfatizou os elementos psicológicos e comportamentais da determinação do preço das ações e passaram a afirmar que os preços futuros das ações podem ser previsíveis com base nos padrões de preços das ações passadas, bem como em certas métricas de avaliações fundamentalistas.

Em contraste à HME, a Hipótese do Mercado Adaptativo (HMA) afirma que os preços das ações são previsíveis e, dessa forma, lucros podem ser obtidos através de previsões (LO, 2017). Em vista disso, a análise técnica e a análise fundamentalista são as duas principais escolas de pensamento que os profissionais de finanças utilizam ao tomar decisões comerciais e prever os preços das ações (KRANTZ, 2016; ROCKEFELLER, 2019).

De acordo com Beyaz et al. (2018), a análise técnica utiliza os preços históricos das ações de uma empresa e as informações do volume de negociação para decidir qual será o preço das ações e para tomar uma decisão de negociação. Sob outra perspectiva, a análise fundamentalista calcula o preço esperado das ações com base em um estudo detalhado dos fatores financeiros subjacentes (lucratividade, eficiência operacional, experiência gerencial, dentre outros) relacionados à empresa, seus produtos, setor e economia geral.

2.3 Algoritmos de Machine Learning

Segundo Khaidem, Saha e Dey (2016), diferentes algoritmos têm sido utilizados para prever o sentido das ações, como SVM (*support vector machine*), redes neurais artificiais, regressão linear, KNN (*K-Nearest Neighbors*) e o classificador Naive Bayes. Isto posto, a literatura revela que o SVM tem sido usado na maioria das vezes nesse tipo de pesquisa.

No trabalho de Lin, Guo e Hu (2013) uma abordagem baseada em SVM é proposta para previsão de tendências do mercado de ações. A análise apresentada consiste em duas partes: seleção de características e modelo de previsão. Na parte de seleção de variáveis, um filtro SVM baseado em correlação é aplicado para classificar e selecionar um bom subconjunto de índices financeiros. Na parte do modelo de previsão, uma função chamada "SVM quasi-linear"é aplicada para prever a direção do movimento do mercado de ações em termos de séries de dados históricos, usando o subconjunto selecionado de índices financeiros como entradas ponderadas. Os resultados experimentais demonstram que o método proposto de previsão pelo autor produz um melhor desempenho de generalização (55 a 70% de acurácia) em relação a outros métodos em termos da taxa de acerto.

De acordo com Powers (2011), a acurácia mede a relação de acertos entre todas as amostras dos dados. Já a precisão, demonstra a capacidade do algoritmo de acertar apenas amostras positivas. O *recall* (ou sensitividade) é a proporção dos casos positivos que são corretamente classificados como positivos. Do mesmo modo, a especificidade é a proporção dos casos negativos que são corretamente classificados como tais. Os detalhes matemáticos desses indicadores serão discutidos mais profundamente no capítulo 4.

Han e Chen (2007) também empregam o SVM junto à análise de demonstrativos financeiros para previsão de tendencias de ativos financeiros. A função de base radial gaussiana (RBF) é escolhida como a função do *kernel* SVM. Os resultados experimentais mostram, segundo o autor, que o método não apenas melhora a taxa de acertos, mas também atende às expectativas dos diferentes acionistas.

Alkhatib et al. (2013) aplica o algoritmo KNN e a abordagem de regressão não linear para prever os preços das ações de uma amostra de seis grandes empresas listadas na bolsa de valores da Jordânia. De acordo com os resultados, o modelo KNN é robusto e produz pequenas taxas de erro; consequentemente, os resultados foram, de acordo com a pesquisa, razoáveis.

Já o uso de algoritmos de *ensemble learning* permaneceram pouco explorados no problema de previsão do mercado, segundo Khaidem, Saha e Dey (2016). No entanto, como demonstrado a seguir, recentemente diversos pesquisadores têm se dedicado ao assunto, desenvolvendo trabalhos utilizando tais modelos. Floresta aleatória (*random forest*) é uma infinidade de árvores de decisão cuja saída é dada pela classificação majoritária do conjunto das árvores individuais (KHAIDEM; SAHA; DEY, 2016).

No trabalho de Khaidem, Saha e Dey (2016), o algoritmo RF foi utilizado para prever as tendencias futuras para um, dois e três meses das ações AAPL, GE (que estão listadas na NASDAQ) e a Samsung Electronics Co. Ltd. (que é negociada na Bolsa de Valores da Coreia). Os resultados obtidos variam entre 84.5 e 94.5% de acurácia, sujeitando-se à janela de tempo prevista. As tabelas 1, 2 e 3 ilustram esses resultados:

Tabela 1 – Resultados para dados *Samsung*.

Período de Previsão	Acurácia%	Precisão	Recall	Especificidade
1 Mês	86.8396	0.881818	0.870736	0.865702
2 Meses	90.6433	0.910321	0.92599	0.880899
3 Meses	93.9664	0.942004	0.950355	0.926174

Fonte: Khaidem, Saha e Dey (2016)

Tabela 2 – Resultados para dados *Apple*.

Período de Previsão	Acurácia%	Precisão	Recall	Especificidade
1 Mês	88.264	0.89263	0.90724	0.84848
2 Meses	93.065	0.94154	0.93858	0.91973
3 Meses	94.533	0.94548	0.9612	0.92341

Fonte: Khaidem, Saha e Dey (2016)

Tabela 3 – Resultados para dados GE.

Período de Previsão	Acurácia%	Precisão	Recall	Especificidade
1 Mês	84.717	0.85531	0.87637	0.80968
2 Meses	90.831	0.91338	0.93099	0.87659
3 Meses	92.543	0.93128	0.94557	0.89516

Fonte: Khaidem, Saha e Dey (2016)

Analisando os resultados, é possível verificar que em todos os ativos previstos, a acurácia se manteve muito alta, além disso, existe uma aparente tendência (carece de mais evidências) de maiores acertos para janelas de tempo mais longas.

Como forma de validar o algoritmo utilizado, Khaidem, Saha e Dey (2016) testam a acurácia de seu modelo usando os dados das ações da empresa 3M, com o intuito de

comparar as predições com o trabalho de Dai e Zhang (2013). Os resultados ficaram entre 82 e 98% de acurácia (para janelas de previsão entre um e noventa dias).

Basak et al. (2019) utilizam RF e o algoritmo *XGBoost* (*gradient boosted decision trees*) para antever a direção das ações das empresas *Apple* e *Facebook*. Para o primeiro algoritmo, as acuracidades obtidas foram de 65% até 95% - de acordo com a janela escolhida. O algoritmo *XGBoost* também demonstrou grande eficácia para classificar as direções do mercado, obtendo acurácias entre 55% e 95%.

Segundo Barboza, Kimura e Altman (2017), o algoritmo RF também possui uma excelente capacidade de prever falências e possíveis inadimplências para gerenciamento de risco de crédito, produzindo melhores taxas de acurácia e erro se comparado a outros modelos de *machine learning*, tais como SVM E RNA. Dessa forma, fica evidenciado que esse algoritmo é capaz de contornar problemas de variadas áreas do conhecimento.

No que tange a predição do mercado usando Redes Neurais, Laboissiere, Fernandes e Lage (2015) elaboraram um modelo para prever os preços máximos e mínimos de alguns ativos do setor energético, sendo eles: Companhia Energética do Rio Grande do Norte (Cosern) - CSRN3 -, Companhia Energética de Brasilia (CEB) - CEBR3 - e CPFL Energia S.A. - CPFE3. Todavia, diferentemente do modelo citado anteriormente que emprega *random forest*, esse último trata a previsão como um problema de regressão. Assim, os resultados são mensurados usando métricas como: MAE (*Mean Absolute Error*), MAPE (*Mean Absolute Percentage Error*) e RMSE (*Root Mean Square Error*).

Nelson, Pereira e Oliveira (2017) construíram um classificador binário que usa as redes LSTM (*Long short-term memory*) para antever o mercado. Cinco ativos, presentes na bolsa de valores brasileira, foram utilizados para a realização das previsões, sendo eles: BOVA11, BBDC4, CIEL3, ITUB4 em PETR4. A janela de previsão utilizada foi de quinze minutos, sendo a acurácia máxima atingida de 55.9%.

Hoseinzade e Haratizadeh (2019) utilizaram redes convolucionais para contornar o problema das previsões. A estrutura sugerida foi aplicada para prever a direção do movimento do dia seguinte para os índices SP500, NASDAQ, DJI, NYSE e RUSSELL com base em vários conjuntos de variáveis iniciais. As avaliações mostram uma melhoria significativa no desempenho da previsão em comparação com outros algoritmos utilizados atualmente.

2.4 Considerações Finais deste Capítulo

Neste capítulo foram abordadas as principais técnicas utilizadas na previsão de ativos financeiros. Na seção 2.2 foram discutidas as técnicas tradicionais (não computacionais) descritas na literatura, com enfoque na análise fundamentalista e técnica - as quais demonstraram ser muito diferentes entre si.

Por outro lado, a seção 2.3 apresentou as principais técnicas e algoritmos de *machine*

learning usadas atualmente para prever os preços dos papéis. Destacaram-se as técnicas utilizando classificação e os modelos: RF, SVM e redes neurais artificiais - utilizados neste trabalho.

No próximo capítulo serão abordados os aspectos teóricos de cada algoritmo usado na pesquisa, incluindo suas formulações matemáticas, estatísticas e a explicação do software e linguagem computacional utilizada.

Capítulo 3

Aspectos Teóricos

3.1 Random Forest

Random Forest (Floresta Aleatória) é um método de aprendizado conjunto para classificação e regressão que opera construindo várias árvores de decisão no momento do treinamento e produzindo a classe, que é o modo das saídas geradas por árvores individuais. O algoritmo foi desenvolvido por Leo Breiman e Adele Cutler (BREIMAN, 2001). O termo veio de florestas de decisão aleatória que foram propostas pela primeira vez por Ho (1995). O método combina a ideia de "ensacamento"de Breiman e a seleção aleatória de recursos, introduzidos independentemente por (HO, 1995) e (AMIT; GEMAN, 1997) para construir uma coleção de árvores de decisão com variação controlada.

Segundo Cutler, Cutler e Stevens (2012), do ponto de vista computacional, os algoritmos RF são atraentes, pois:

- ❏ podem operar com regressão e classificação multiclasse;
- ❏ são relativamente rápidos para treinar e testar;
- ❏ dependem apenas de um ou dois parâmetros de ajuste;
- ❏ possuem uma estimativa embutida do erro de generalização;
- ❏ podem ser usados para problemas de alta dimensionalidade;
- ❏ podem ser facilmente implementados em paralelo.

Por outro lado, estatisticamente eles são interessantes porque podem prover:

- ❏ a importância das variáveis;
- ❏ pesagem diferencial das classes;
- ❏ imputação de valor ausente;

- ❏ visualização das árvores de decisão;
- ❏ detecção fora da curva (*out of bag error*);
- ❏ aprendizado supervisionado e não supervisionado.

De acordo com Cutler, Cutler e Stevens (2012):

> *Random Forest* é um método de aprendizado em conjunto baseado em árvores de decisão (cada árvore depende de uma coleção de variáveis aleatórias). Dessa forma, para um vetor de dimensão p randômico $X = (X_1, ..., X_p)^T$, representando os valores reais de entrada, e uma variável aleatória Y, representando o valor real da resposta, assume-se uma distribuição conjunta desconhecida $P_{XY}(X, Y)$. Nesse sentido, o objetivo é encontrar uma função $f(X)$ para prever Y (CUTLER; CUTLER; STEVENS, 2012, p. 158, tradução nossa).

A função preditiva é determinada por uma função de perda $L(Y, f(X))$ e definida para minimizar o valor esperado do erro:

$$E_{XY}(L(Y, f(X))) \tag{1}$$

em que os subscritos denotam o valor esperado em relação à distribuição conjunta de X e Y.

Nessa perspectiva, $L(Y, f(X))$ é uma medida de quão próximo $f(X)$ está de Y. Assim, valores de $f(X)$ que estão muito distantes de Y são penalizados.

Por conseguinte, é usado o erro quadrático para problemas de regressão:

$$L(Y, f(X)) = (Y - f(X))^2 \tag{2}$$

já para classificação é o usado o erro *"zero-one"*:

$$L(Y, f(X)) = I(Y \neq f(X)) = \begin{cases} 0, & \text{se } Y = f(X) \\ 1, & \text{se } Y \neq f(X) \end{cases} \tag{3}$$

Portanto, minimizando $E_{XY}(L(Y, f(X)))$ (assumindo um erro quadrático) obtém-se a função de regressão:

$$f(x) = E(Y|X = x) \tag{4}$$

No caso da classificação, se os valores possíveis de Y são denotados por $\mathscr{Y}$, minimizando o erro obtém-se a regra de Bayes:

$$f(x) = \arg\max_{y \in \mathscr{Y}} P(Y = y \,|X = x) \tag{5}$$

Segundo Cutler, Cutler e Stevens (2012, p. 159, tradução nossa) "algoritmos *ensembles* produzem f em termos de uma coleção de bases de aprendizado $h_1, ..., h_j(x)$. Essas bases

são combinadas para gerar $f(x)$ (preditor por conjunto)". Em classificação - método usado neste trabalho - $f(x)$ é a classe prevista com mais frequência:

$$f(x) = \arg\max_{y \in \mathscr{Y}} \sum_{j=1}^{J} I(y = h_j(x)) \tag{6}$$

No algoritmo RF, a j-ésima base de aprendizado é uma árvore representada por $h_j(X, \Theta_j)$, em que Θ_j é uma coleção de variáveis aleatórias e essas são independentes para: $j = 1, ..., J$ (CUTLER; CUTLER; STEVENS, 2012).

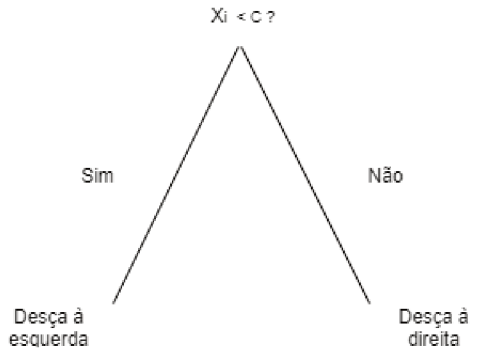
3.1.1 Introdução às árvores de Classificação e Regressão

Segundo Cutler, Cutler e Stevens (2012):

> As árvores usadas nas florestas aleatórias são baseadas nas árvores de particionamento recursivo binário. Essas árvores particionam o espaço do preditor usando uma sequência de partições binárias (divisões) em variáveis individuais. O nó "raiz"da árvore compreende todo o espaço do preditor. Os nós que não são divididos são chamados de "nós terminais"e formam a partição final desse espaço. Dessa maneira, cada nó não terminal se divide em dois nós descendentes, um à esquerda e outro à direita, de acordo com o valor de uma das variáveis preditoras. Para uma variável preditora contínua, uma divisão é determinada por um ponto de divisão; pontos para os quais o preditor é menor que o ponto de divisão, a decisão é feita para a esquerda, o restante para a direita (CUTLER; CUTLER; STEVENS, 2012, p. 159, tradução nossa).

A Figura 1 ilustra esse processo.

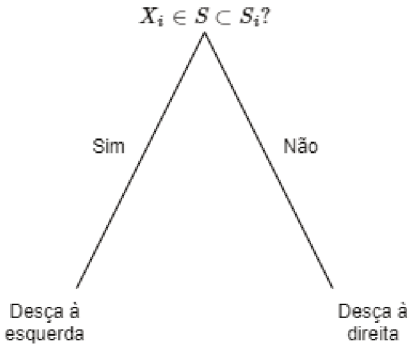
Figura 1 – Divisão em uma variável preditora contínua X_i, usando o ponto de divisão c.



Fonte: Adaptado de Cutler, Cutler e Stevens (2012).

Para o caso desse trabalho, a variável preditora categórica X_i utilizada obtém valores de um conjunto finito de categorias S_i=$\{s_{i,1}..., s_{i,m}\}$. A partição envia um subconjunto dessas categorias $S \subset S_i$ para a esquerda e as demais categorias para a direita. A Figura 2 mostra esse procedimento.

Figura 2 – Divisão em uma variável preditora categórica X_i, utilizando $S \subset S_i$.



Fonte: Adaptado de Cutler, Cutler e Stevens (2012).

De acordo com Cutler, Cutler e Stevens (2012, p. 160, tradução nossa), "a divisão específica que uma árvore usa para particionar um nó em seus dois descendentes deve ser escolhida considerando todas as partições possíveis em cada variável preditora e escolhendo o "melhor"de acordo com algum critério". No contexto da classificação - utilizada neste trabalho - em que existem K classes denotadas $1, ..., K$, um critério de divisão típico é o índice de Gini:

$$Q = \sum_{k \neq k'}^{K} \hat{p}_k \hat{p}_{k'}, \tag{7}$$

em que $\hat{p}_k$ é a proporção de observações da classe k no nó:

$$\hat{p}_k = \frac{1}{n} \sum_{i=1}^{n} I(y_i = k). \tag{8}$$

Na classificação, o critério de partição gera uma medida de pureza do nó e uma possível divisão cria dois nós descendentes. Dessa maneira, denotando o critério de repartição para esses 2 nós como Q_E e Q_D, e seus tamanhos de amostras por n_E e n_D, a separação é escolhida para minimizar a equação : $Q_{separação} = n_E Q_E + n_D Q_D$.

Para uma variável preditora categórica, Q_E, Q_D e $Q_{separação}$ são calculados para todas as maneiras possíveis de escolher um subconjunto de categorias para ir para cada nó descendente.

Para Cutler, Cutler e Stevens (2012):

> Depois que uma divisão é selecionada, os dados são particionados nos dois nós e cada um deles é tratado da mesma maneira que o nó original. O procedimento continua recursivamente até que um critério de parada seja atendido. Por exemplo, o procedimento pode parar quando todos os nós não divididos contiverem menos do que algum número fixo de casos. Quando o critério de parada é atendido, os nós não divididos são chamados de “terminal de nós ". Um valor previsto é obtido para todas as observações nos nós terminais calculando a média da resposta para

> problemas de regressão ou calculando as mais frequentes classes para problemas de classificação. Para prever em um novo ponto, o conjunto de valores preditores são usados para passar o ponto pela árvore até que ele caia em um nó terminal e a previsão para o nó terminal é usada como a predição para o novo ponto (CUTLER; CUTLER; STEVENS, 2012, p. 161, tradução nossa).

O algoritmo RF utiliza árvores $h_j(X, \Theta_j)$ como bases de aprendizados. Ainda de acordo com Cutler, Cutler e Stevens (2012), para um conjunto de dados de treino $\mathscr{D} = \{(x_1, y_1), ..., (x_N, y_N)\}$, onde $x_i = (x_{i,1}, ..., x_{i,p})^T$ denota os preditores e y_i corresponde às saídas e a uma realização particular θ_j de Θ_j, dessa forma, a árvore ajustada é denotada por $\hat{h}_j(x, \theta_j, \mathscr{D})$ (BREIMAN, 2001). Na prática, o componente aleatório j não é considerado explicitamente, mas é usado implicitamente para injetar aleatoriedade de duas maneiras.

Uma delas, é o ensacamento, em que cada árvore é ajustada a uma amostra de *bootstrap* independente dos dados originais. Segundo Maddala e Weller (2003) essa é uma técnica de reamostragem que propõe reduzir desvios, provendo um desvio padrão mais confiável.

A randomização envolvida na amostragem de *bootstrap* fornece uma parte de j. Depois, ao dividir um nó, a melhor divisão é encontrada em um subconjunto selecionado aleatoriamente de m variáveis preditoras em vez de em todos os preditores p, independentemente em cada nó. A randomização usada para amostrar os preditores fornece a parte restante de j. Por fim, as árvores resultantes são combinadas por votação não ponderada se a resposta for uma média categórica, no caso da classificação (CUTLER; CUTLER; STEVENS, 2012).

3.2 Support Vector Machine

O algoritmo Máquinas de vetores de suporte (SVM) foi desenvolvido por Cortes e Vapnik (1995). Vários estudos relatam que essa técnica, geralmente, é capaz de fornecer maior precisão de classificação do que os outros algoritmos existentes (HSU; LIN, 2002; JOACHIMS, 1998). Na última década, o SVM surgiu como um importante método de aprendizado para resolver problemas de classificação e regressão em vários campos, principalmente na biologia computacional, finanças e categorização de textos. Isso se deve em parte a mecanismos internos, para garantir uma boa generalização que leva a previsões precisas, o uso de funções (*kernels*) para modelar distribuições não lineares e a capacidade de treinar relativamente rápido em grandes conjuntos de dados, usando novas técnicas de otimização matemática (APOSTOLIDIS-AFENTOULIS, 2015).

Esse trabalho propõe a utilização de dois *kernels* diferentes para realizar as previsões. O primeiro, RBF (*radial Basis function*), é comumente usado em algoritmos de *machine learning*. Já o segundo, chamado de GHI (*generalized histogram intersection*), foi desen-

volvido por Boughorbel, Tarel e Boujemaa (2005) visando à aplicação em reconhecimento de imagens. No entanto, este trabalho demonstra que o último também se mostrou eficaz no reconhecimento de tendências em ações.

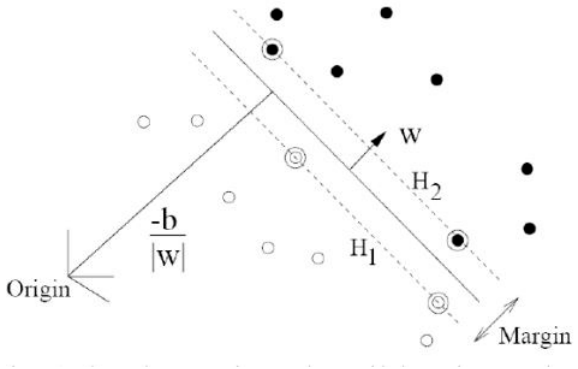
Para entender o funcionamento do SVM, é preciso definir como opera o caso linear. Dessa forma, segundo Apostolidis-Afentoulis (2015), na configuração de classificação binária, seja $((x_1, y_1)...(x_n, y_n))$ o conjunto de dados de treinamento em que x_i são os vetores de características que representam as observações e y_i $(-1, +1)$ são as classes. Nesse sentido, o aprendizado de vetores de suporte tem como objetivo encontrar um hiperplano que separe as classes usando uma margem, definida como a maior distância entre as instâncias positivas e negativas mais próximas no hiperplano.

Assim, suponha para o conjunto de dados as seguintes restrições:

$$f(X) = \begin{cases} +1, & w * x_i + b \geq +1 \\ -1, & w * x_i + b \leq -1 \end{cases} \tag{9}$$

Em que w é normal ao hiperplano, e $|b|/||w||$ é a distância perpendicular entre o hiperplano e a origem. Já $||w||$ representa a distância euclideana de w, assim como mostra a Figura 3:

Figura 3 – Hiperplano SVM.



Fonte: Apostolidis-Afentoulis (2015).

Segundo Apostolidis-Afentoulis (2015), a equação 9 pode ser escrita como:

$$y_i(w * x_i + b) \geq 1 \ \forall_i \tag{10}$$

A margem ρ pode ser calculada como a distância entre H_1 e H_2:

$$\rho = \frac{|1 - b|}{||w||} - \frac{|-1 - b|}{||w||} = \frac{2}{||w||} \tag{11}$$

Portanto, a margem máxima que separa o hiperplano pode ser obtida resolvendo o seguinte problema de otimização:

$$\min_{w \in H} \tau(w) = \frac{1}{2}||w||^2, \text{ sujeito a } y_i(w * x_i + b) \geq 1 \ \forall_i \tag{12}$$

Mudando para a formulação lagrangiana desse problema temos:

$$\min_{w,b} L(w, b, a) = \frac{1}{2}||w||^2 \ \sum_{i=1}^{l} a_i y_i (x_i w + b) + \sum_{i=1}^{l} a_i \tag{13}$$

com os multiplicadores de Lagrange $a_i \geq 0$ para cada restrição em 12. O objetivo é, então, minimizar 13 em relação a w e b simultaneamente, exigindo que os derivados de $L(w, b, a)$ em relação a todos os a desapareçam.

3.2.1 SVM-RBF

Em geral, o *kernel* RBF é uma primeira escolha razoável. Ele mapeia não linearmente as amostras para um espaço dimensional maior, portanto, diferentemente do *kernel* linear, pode lidar com o caso em que a relação entre rótulos e atributos da classe não é linear. Além disso, o SVM linear é um caso especial do RBF (KEERTHI; LIN, 2003), pois a função linear com um parâmetro de penalidade tem o mesmo desempenho que a função RBF com alguns parâmetros (C, γ) (APOSTOLIDIS-AFENTOULIS, 2015).

Segundo Gunn (1998), um produto interno no espaço de características tem um kernel equivalente no espaço de entrada,

$$K(x, x') = \langle \phi(x), \phi(x') \rangle \tag{14}$$

desde que certas condições sejam mantidas. Se K é uma função definida positiva simétrica, que satisfaz as condições de Mercer,

$$K(x, x') = \sum_{m}^{\infty} a_m \phi_m(x) \phi_m(x), \quad a_m \geq 0 \tag{15}$$

$$\iint K(x, x') g(x) g(x') \, dx \, dx' > 0, \quad g \in L_2 \tag{16}$$

então o *kernel* representa um produto interno legítimo no espaço de recursos. Funções válidas que atendem às condições de Mercer agora são fornecidas, as quais, a menos que declaradas, são válidas para todos os reais x e x'.

Para o *kernel* (função) RBF temos a seguinte expressão:

$$K(x, x') = exp(-\frac{\|x - x'\|^2}{2\sigma^2}). \tag{17}$$

em que σ é um parâmetro ajustável. Sendo assim, segundo Vert, Tsuda e Schölkopf (2004), quanto menor o parâmetro σ, mais gaussianos ficam em torno dos vetores de

suporte e, portanto, mais complexo o limite de decisão pode ser. Por outro lado, σ maior corresponde a um limite de decisão mais suave.

3.2.2 SVM-GHI

Proposta por Boughorbel, Tarel e Boujemaa (2005), a função *generalized histogram intersection* (GHI), se baseia em outra função, destinada ao reconhecimento de imagens, denominada *histogram intersection* (HI) (SWAIN; BALLARD, 1991).

No entanto, a função GHI se aplica a uma variedade muito maior de contextos (BOUGHORBEL; TAREL; BOUJEMAA, 2005). Além disso, as imagens não precisam mais ter o mesmo tamanho como proposto por Swain e Ballard (1991).

Para a função GHI temos sua definição matemática como:

$$K(x, x') = min\{|x_i^{\beta}|, |x'^{\beta}_i|\} \tag{18}$$

em que β é um hiper-parâmetro que deve ser maior que zero.

3.3 Redes Neurais Artificiais

Conforme Bishop (2006), o termo "rede neural"tem suas origens nas tentativas de encontrar representações matemáticas do processamento de informações em sistemas biológicos. Uma das principais características das RNA (redes neurais artificiais) é a capacidade de aprender não apenas por meio de exemplos, mas também de generalizar a partir das informações aprendidas.

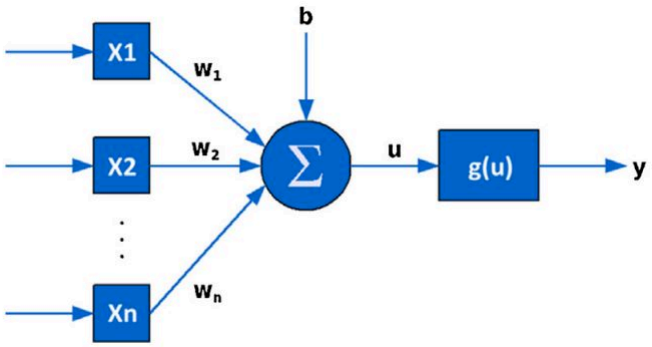
De acordo com Haykin (2007), o uso de redes neurais oferece várias vantagens, dentre elas:

- Não-linearidade: Um neurônio pode ser linear ou não-linear. Dessa forma, diferentes tipos de sinais de entrada podem ser utilizados;
- Adaptabilidade: As redes neurais têm uma capacidade inata de adaptar seus pesos sinápticos a modificações do meio ambiente;
- Resposta a evidências: Uma rede neural pode ser projetada para fornecer informação não somente sobre qual padrão particular selecionar, mas também sobre a confiança ou crença na decisão tomada;

O neurônio artificial da Figura 4 consiste em um modelo matemático com n terminais de entrada (x_1,x_2,..,x_{n-1} e x_n, representando os dendritos) e um único terminal de saída (y, representando o axônio). O comportamento é simulado por meio de pesos sinápticos (w_1, w_2,.., w_{n-1} e w_n), cujos valores podem ser positivos ou negativos. Já o viés do neurônio é representado por um valor limiar b. A função de ativação g é responsável por

processar as informações recebidas (entradas ponderadas recebidas e o valor limite do neurônio) e fornecer uma saída para o neurônio, (LABOISSIERE; FERNANDES; LAGE, 2015).

Figura 4 – Perceptron.



Fonte: Laboissiere, Fernandes e Lage (2015).

De acordo com Laboissiere, Fernandes e Lage (2015), o neurônio artificial opera da seguinte maneira:

1. Os sinais são submetidos às entradas;
2. Cada sinal é multiplicado pelo seu respectivo peso sináptico;
3. É realizada uma soma entre os sinais de entrada ponderados e o neurônio;
4. O valor limite é calculado;
5. Esta informação é processada pela função de ativação do neurônio;
6. Um sinal de saída é produzido.

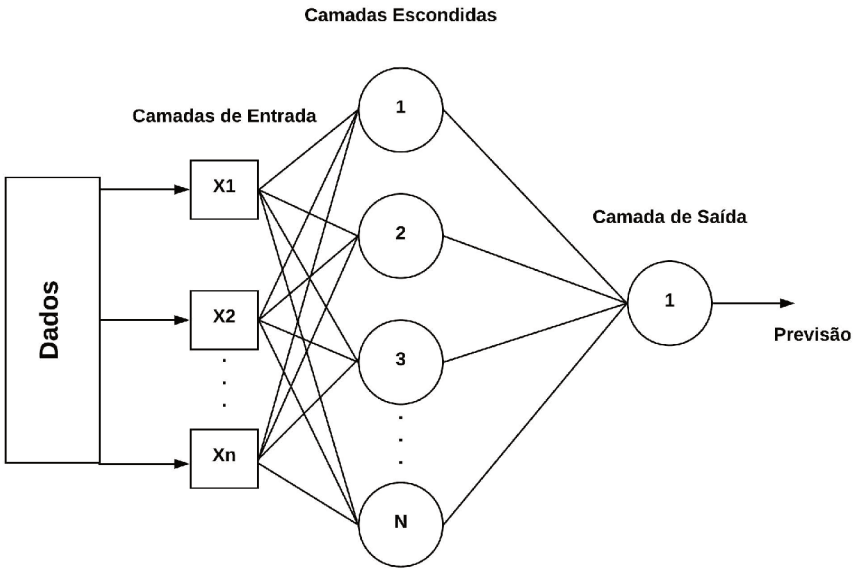
A Figura 5 ilustra o processo em que os dados são processados por meio das camadas de uma rede neural profunda:

Neste trabalho, foi desenvolvida uma RNA com 3 camadas escondidas. A primeira e segunda possuem 12 neurônios, já a terceira possui 3. A função de ativação empregada nas células foi a *Rectified Linear Unit* (ReLU). Matematicamente, essa função é definida como:

$$Y = max(0, x) \tag{19}$$

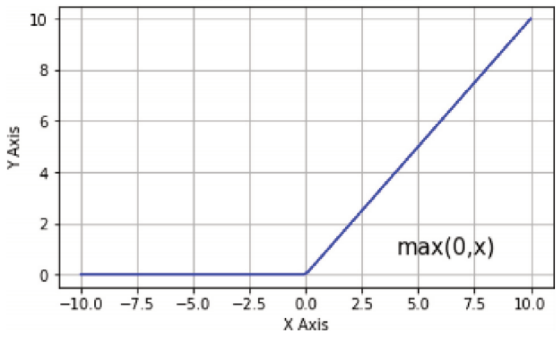
Em que Y é o sinal de resposta do neurônio e x o valor de sua entrada. A Figura 6 ilustra o funcionamento matemático desta função de ativação:

Figura 5 – Arquitetura de uma RNA.



Fonte: Adaptado de Laboissiere, Fernandes e Lage (2015).

Figura 6 – Gráfico de uma função de ativação ReLU.



Fonte: Hammad e Wang (2019).

Conhecida também como função de rampa, como mostrado na Figura 6, essa função de ativação faz com que a rede convirja muito mais rápido. Além disso, é computacionalmente muito eficiente porque é implementada usando limiares simples (HAMMAD; WANG, 2019).

3.4 Ferramentas Utilizadas

Os algoritmos citados nesse trabalho foram implementados por meio da linguagem *Python*. Essa é uma linguagem de programação interpretada, orientada a objetos e de alto nível, com semântica dinâmica. A sintaxe simples e fácil de aprender do *Python* enfatiza a legibilidade e, portanto, reduz o custo de manutenção do programa. Além disso, ela suporta módulos e pacotes, o que incentiva a modularidade do programa e a reutilização de código. O interpretador e a extensa biblioteca padrão estão disponíveis na forma de código-fonte ou binário sem custo para todas as principais plataformas e podem ser distribuídos livremente (ROSSUM, 1998).

3.4.1 *Pandas*

As análises e manipulações das séries temporais das ações pesquisadas foram possíveis por meio do uso da biblioteca *Pandas*. Segundo Wes McKinney (2010), ela é um pacote *Python* que fornece estruturas de dados rápidas, flexíveis e expressivas, projetadas para facilitar o trabalho com dados estruturados (tabulares, multidimensionais, potencialmente heterogêneos) e de séries temporais. Além disso, ela tem o propósito mais amplo de se tornar a ferramenta de análise e manipulação de dados de código aberto mais poderosa e flexível disponível em qualquer linguagem.

3.4.2 *NumPy*

Algumas operações matemáticas, utilizando vetores e matrizes, foram realizadas por meio da biblioteca *NumPy*. De acordo com Lemenkova (2019) ela usa a sintaxe e a semântica do *Python*, operando com matrizes e empregando operações lógicas. Além disso, permite diversas abordagens orientadas a objetos e operações com tabelas usando vetores.

3.4.3 TA-Lib

Os indicadores técnicos utilizados neste trabalho foram obtidos por meio da biblioteca TA-Lib. Ela é amplamente utilizada por desenvolvedores de softwares de negociação que exigem a análise técnica de dados do mercado financeiro (TicTacTec LLC, 2012). Este pacote possui várias vantagens, entre elas:

- ❏ Inclui aproximadamente 200 indicadores, como: OBV, MACD, RSI,K% e bandas de Bollinger;
- ❏ Reconhecimento de velas;
- ❏ API de código aberto para linguagem $C/C++$, *Java*, *Perl*, *Python* e *.NET* 100% gerenciável.

3.4.4 Scikit-Learn

Os algoritmos *Random Forest* e *Support Vector Machine*, empregados neste trabalho, foram implementados utilizando a biblioteca *Scikit-Learn*. Ela fornece muitos algoritmos de aprendizado não supervisionado e supervisionado, facilitando o desenvolvimento de trabalhos na área de machine learning (PEDREGOSA et al., 2011).

As funcionalidades fornecidas pela *Scikit-Learn* incluem:

- ❏ Pré-processamento, incluindo normalização dos dados.
- ❏ Algoritmos de regressão (RF, SVM, KNN, regressão logística, dentre outros)
- ❏ Classificação (RF, SVM,KNN, dentre outros)
- ❏ *Clustering* (aprendizado não supervisionado)
- ❏ Seleção de modelo (Curvas de aprendizado, validação cruzada e etc.)

3.4.5 *Keras*

A rede neural desenvolvida neste trabalho foi implementada por meio da biblioteca *Keras*. Essa é uma API de redes neurais de alto nível, escrita em *Python* e capaz de executar sobre *TensorFlow*, *CNTK* ou *Theano*. Foi desenvolvida com foco em permitir experimentação rápida. Ela também é capaz de passar do esboço para o resultado com o menor atraso possível, sendo essencial em pesquisas na área (Keras SIG, 2020).

A biblioteca *Keras*:

- ❏ Permite prototipagem fácil e rápida (por meio da facilidade de uso, modularidade e extensibilidade).
- ❏ Suporta redes convolucionais e redes recorrentes, bem como combinações das duas.
- ❏ Funciona perfeitamente na CPU e GPU.

3.5 Considerações Finais

Este capitulo destinou-se a abordar os aspectos teóricos e matemáticos dos algoritmos empregados nesta pesquisa. Além disso, tratou-se sobre as ferramentas computacionais utilizadas para a elaboração dos modelos propostos.

Na seção 3.1, foram abordados os aspectos teóricos do algoritmo *random forest*, detalhando seus aspectos matemáticos e modo de funcionamento.

A seção 3.2 destinou-se a detalhar a operação do algoritmo SVM, introduzindo os conceitos do caso linear e tratando dos dois *kernels* (RBF e GHI) utilizados no trabalho.

Na seção 3.3, tratou-se dos conceitos teóricos das redes neurais artificiais, informando seus aspectos matemáticos, vantagens e desvantagens.

A seção 3.4 destinou-se a abordar as ferramentas computacionais empregados neste trabalho, dissertando sobre a linguagem *python* e as bibliotecas utilizadas para implementar os algoritmos citados no capítulo.

No próximo capítulo, será discutido a metodologia empregada na pesquisa, apresentando a base de dados utilizada, etapas de processamento e os métodos para avaliação dos algoritmos.

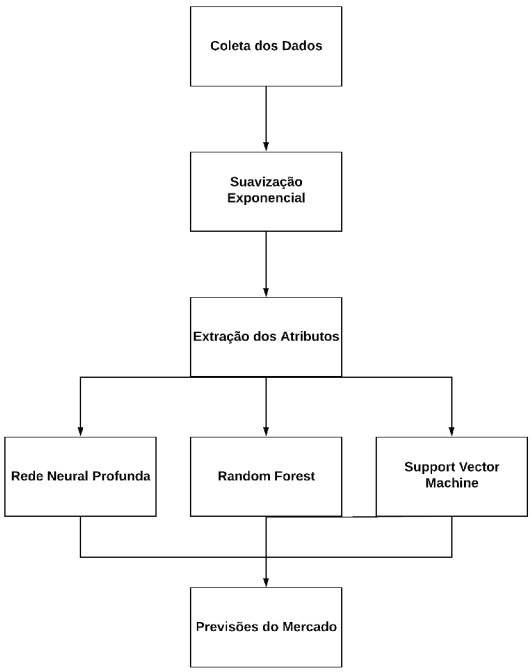
Capítulo 4

Metodologia deste Trabalho

4.1 Introdução

Os dados das séries temporais dos ativos pesquisados são coletados, suavizados e os indicadores técnicos extraídos. Indicadores técnicos são parâmetros que fornecem informações sobre o comportamento esperado dos preços dos papeis no futuro. Essas medidas são então usadas para treinar os três algoritmos propostos. Os detalhes de cada etapa serão discutidos a seguir.

Figura 7 – Metodologia proposta.

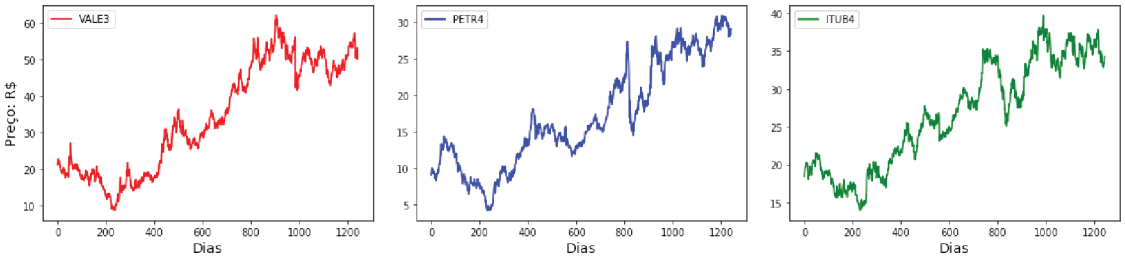


Fonte: O autor.

4.2 Base de Dados

A base de dados usada neste trabalho compreende um período de 5 anos (2015-02-11 até 2020-02-10). Dessa maneira, foram obtidos dados diários de fechamento, abertura, alta, baixa e volume dos seguintes ativos presentes na bolsa de valores brasileira: VALE3, PETR4, ITUB4. A Figura 8 ilustra as séries temporais dos dados de fechamento das ações citadas respectivamente:

Figura 8 – Séries temporais dos dados de fechamento.



Fonte: O autor.

Primeiramente, os dados de cada papel foram divididos aleatoriamente em treinamento e teste, sendo a validação equivalente a 20% da série temporal prevista. As Tabelas 4 e 5 ilustram como as classes "+1" (Movimento de alta) e "0" (Movimento de baixa) estão distribuídas para o *dataset* de cada ativo.

A classe a ser prevista no N-ésimo dia é calculada da seguinte forma:

$$Classe_N = Sinal(Fechamento_{N+d} - Fechamento_N) \tag{20}$$

Em que d indica o número de dias após os quais a previsão será feita. Quando o valor da classe é 1, isso denota que há uma mudança positiva no preço após d dias da previsão. Já o valor 0 indica que há uma mudança negativa, também após d dias da data em que foi feita a estimativa.

Tabela 4 – Dados de treinamento obtidos aleatoriamente.

Ação	Período (dias)	Classe 1	Classe 0
ITUB4	30	506	406
ITUB4	60	543	369
ITUB4	90	571	341
PETR4	30	520	392
PETR4	60	537	375
PETR4	90	600	312
VALE3	30	510	402
VALE3	60	575	337
VALE3	90	554	358

Fonte: O autor.

Analisando a distribuição dos números de cada classe, é possível observar que há uma maior predominância da classe positiva em relação à negativa. Outro ponto, é que à medida que o período de previsão aumenta, o desbalanceamento das classes também aumenta.

Tabela 5 – Dados de validação obtidos aleatoriamente.

Ação	Período (dias)	Classe 1	Classe 0
ITUB4	30	125	103
ITUB4	60	130	98
ITUB4	90	134	94
PETR4	30	134	95
PETR4	60	128	101
PETR4	90	135	94
VALE3	30	133	96
VALE3	60	143	86
VALE3	90	143	86

Fonte: O autor.

A Figura 9 exemplifica o processo de divisão dos dados de previsão para os períodos de 30, 60 e 90 dias. Deste modo, é possível verificar que a separação ocorre de maneira aleatória, contendo as datas de maneira não cronológica.

Figura 9 – Distribuição aleatória das classes.

Data	30 dias	60 dias	90 dias
2016-04-11	0.0	0.0	1.0
2016-12-05	1.0	1.0	0.0
2018-09-20	0.0	0.0	0.0
2015-04-07	1.0	0.0	0.0
2018-11-13	0.0	0.0	0.0
2018-04-11	1.0	1.0	1.0
2015-09-30	0.0	0.0	0.0
2017-03-16	0.0	0.0	0.0

Fonte: O autor.

Em paralelo ao processo citado anteriormente, os dados dos ativos também foram divididos na mesma proporção de treino e teste mencionada. No entanto, essa divisão ocorreu usando a ordem temporal das séries históricas dos papéis pesquisados.

A Figura 10 demonstra como os dados foram divididos utilizando esse processo. É possível observar que a separação obedece a uma ordem cronológica, assim, a utilização do algoritmo se aproxima melhor do uso no dia a dia, onde os dados de treinamento só podem usar dados passados para prever o futuro.

Figura 10 – Distribuição temporal das classes.

Data	30 dias	60 dias	90 dias
2018-10-23	0.0	0.0	0.0
2018-10-24	0.0	0.0	0.0
2018-10-25	0.0	0.0	0.0
2018-10-26	0.0	0.0	0.0
2018-10-29	0.0	0.0	0.0
2018-10-30	0.0	0.0	0.0
2018-10-31	0.0	0.0	0.0
2018-11-01	0.0	0.0	0.0

Fonte: O autor.

As Tabelas 6 e 7 ilustram como as classes “+1” (Movimento de alta) e “0” (Movimento de baixa) foram distribuídas para o dataset de cada ativo, usando a última técnica.

Tabela 6 – Dados de treinamento obtidos usando a ordem temporal das séries históricas.

Ação	Período (dias)	Classe 1	Classe 0
ITUB4	30	511	401
ITUB4	60	547	365
ITUB4	90	600	312
PETR4	30	516	396
PETR4	60	532	380
PETR4	90	548	364
VALE3	30	539	373
VALE3	60	615	297
VALE3	90	605	307

Fonte: O autor.

Assim como no primeiro caso, também é possível observar que há uma maior predominância da classe positiva em relação à negativa. Além disso, também é possível afirmar que o desbalanceamento das classes aumenta ao longo do crescimento das janelas de previsão.

Tabela 7 – Dados de validação obtidos usando a ordem temporal das séries históricas.

Ação	Período (dias)	Classe 1	Classe 0
ITUB4	30	120	108
ITUB4	60	126	102
ITUB4	90	123	105
PETR4	30	138	91
PETR4	60	133	96
PETR4	90	187	42
VALE3	30	125	104
VALE3	60	126	103
VALE3	90	137	92

Fonte: O autor.

4.3 Pré-processamento dos dados

Previamente, as séries temporais dos dados de abertura, fechamento, máxima e mínima são suavizadas exponencialmente. A suavização exponencial aplica mais peso às observações recentes e pesos exponencialmente decrescentes às observações anteriores (KHAIDEM; SAHA; DEY, 2016). A fórmula de uma série suavizada exponencialmente pode ser calculada recursivamente como:

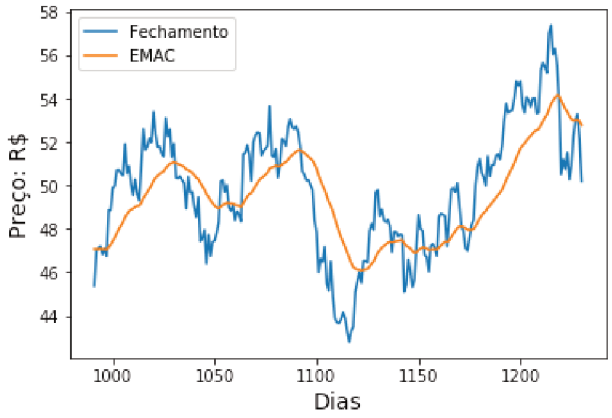
$$S_0 = Y_0 \tag{21}$$

$$S_t = \alpha * Y_t + (1 - \alpha) * S_{t-1} \tag{22}$$

Em que α é o fator de suavização e $0 < \alpha < 1$. Valores maiores de α reduzem o nível de suavização. Quando $\alpha = 1$, a variável suavizada torna-se igual à observação real. O valor suavizado S_t pode ser calculado assim que duas observações estiverem disponíveis.

A suavização remove variações aleatórias ou ruído dos dados históricos, permitindo que o modelo identifique facilmente a tendência de preços de longo prazo no comportamento do preço das ações - ela pode ser interpretada como o uso de filtro passa-baixa digital, que filtra as oscilações de alta frequência desnecessárias. Dessa forma, os indicadores técnicos são então calculados a partir dos dados suavizados da série temporal (KHAIDEM; SAHA; DEY, 2016). A Figura 11 exemplifica a suavização exponencial para os dados de fechamento do ativo VALE3.

Figura 11 – Série temporal do ativo VALE3 suavizada exponencialmente.



Fonte: O autor.

4.4 Extração dos Atributos

Os três algoritmos de *machine learning* referidos empregam atributos específicos para realização do treinamento e previsão dos papéis. Para isso, são calculados alguns indicadores técnicos. Segundo Khaidem, Saha e Dey (2016), esses são parâmetros importantes calculados a partir de dados de séries temporais que visam prever a direção do mercado financeiro. Nesse sentido, são ferramentas amplamente utilizadas pelos investidores para verificar sinais de baixa ou alta. Os indicadores técnicos usados neste trabalho e seus significados, de acordo com Murphy (1999), estão listados abaixo:

- Índice de força relativa (*Relative Strength Index* - RSI)

  A fórmula para calcular o RSI é:

$$RSI = 100 - \frac{100}{1 + RS} \tag{23}$$

$$RS = \frac{\text{Ganho médio nos últimos 14 dias}}{\text{Perda média nos últimos 14 dias}} \tag{24}$$

  Para encontrar o ganho médio, é feita a soma do total de pontos ganhos durante os últimos 14 dias e divido esse total por 14. Já para encontrar a perda média, é feito o mesmo processo, porém, somando as perdas do últimos 14 dias. A força relativa (RS) é então determinada dividindo a média superior pela média inferior. Esse valor é então inserido na fórmula para obter o RSI. Essa ferramenta funciona melhor quando suas flutuações atingem os extremos superior e inferior. Portanto, se o usuário estiver negociando em uma base de prazo muito curto e desejar que as

oscilações do oscilador sejam mais pronunciadas, o período - 14 dias - poderá ser reduzido.

❑ Oscilador Estocástico (*Stochastic Oscillator* - %K)

$$\%K = 100 * \frac{C - L14}{H14 - L14} \tag{25}$$

Onde,
C = Preço atual de fechamento;
$L14$= Menor preço nos últimos 14 dias;
$H14$= Maior preço nos últimos 14 dias.

O oscilador estocástico é baseado na observação de que à medida que os preços aumentam, os preços de fechamento tendem a estar mais próximos do limite superior da faixa de preços. Por outro lado, nas tendências de baixa, o preço de fechamento tende a estar próximo do limite inferior do intervalo.

❑ Williams %R

$$\%R = 100 * \frac{H14 - C}{H14 - L14} \tag{26}$$

Onde,
C = Preço atual de fechamento;
$L14$= Menor preço nos últimos 14 dias;
$H14$= Maior preço nos últimos 14 dias.

O oscilador Larry Williams $\%R$ é baseado em um conceito semelhante de medir o último fechamento em relação à sua faixa de preço em um determinado número de dias. O fechamento de hoje é subtraído do preço mais alto do intervalo por um determinado número de dias e essa diferença é dividida pelo intervalo total no mesmo período.

❑ Convergência e divergência da média móvel (*Moving Average Convergence Divergence* - MACD)

$$MACD = EMA_{12}(C) - EMA_{26}(C) \tag{27}$$

$$\text{Linha de Sinal} = EMA_9(MACD) \tag{28}$$

Onde,
C = Série temporal dos dados de fechamento;
EMA_n= Média exponencial de n dias;
$H14$= Maior preço nos últimos 14 dias.

Quando o MACD fica abaixo da linha de sinal, indica um sinal de venda. Já quando ultrapassa, indica um sinal de compra.

❏ Taxa de mudança dos preços (*Price Rate of Change* - PROC)

$$PROC(t) = \frac{C(t) - C(t-n)}{C(t-n)} \tag{29}$$

Onde,
$PROC(t)$ = Taxa de mudança do preço no tempo t;
$C(t)$ = Preço de fechamento no tempo t.

Dessa maneira, é medida a mudança de preço com a relação a n dias atrás.

❏ Indicador Saldo de Volume (*On-Balance Volume* - OBV)

$$OBV(t) = \begin{cases} OBV(t-1) + Vol(t), & \text{se } C(t) > C(t-1) \\ OBV(t-1) - Vol(t), & \text{se } C(t) < C(t-1) \\ OBV(t-1), & \text{se } C(t) = C(t-1) \end{cases} \tag{30}$$

Onde,
$OBV(t)$ = Indicador saldo de volume no tempo t;
$Vol(t)$ = Volume negociado no tempo t.
$C(t)$ = Preço de fechamento no tempo t

O OBV deve seguir na mesma direção que a tendência de preço. Logo, se os preços mostrarem uma tendência de alta, a linha OBV deve fazer o mesmo. Por outro lado, se os preços tendem para baixo, o mesmo deve acontecer com a linha OBV.

4.5 Método para a Avaliação das Predições

De acordo com os resultados das previsões obtidas pelos modelos, é possível decidir sobre a compra ou venda das ações das empresas pesquisadas. Dessa modo, é necessário obter medidas estatísticas que provem a confiabilidade do algoritmo. Os parâmetros usados para avaliar a robustez de um classificador binário são acurácia, precisão, *recall*

(também conhecido como sensibilidade) e especificidade. As fórmulas para calcular esses parâmetros são apresentadas abaixo:

$$Acurácia = \frac{t_p + t_n}{t_p + t_n + f_p + f_n} \tag{31}$$

$$Precisão = \frac{t_p}{t_p + f_p} \tag{32}$$

$$Recall = \frac{t_p}{t_p + f_n} \tag{33}$$

$$Especificidade = \frac{t_n}{t_n + f_p} \tag{34}$$

onde,

t_p = Número de previsões de alta classificadas corretamente

t_n = Número de previsões de baixa classificadas corretamente

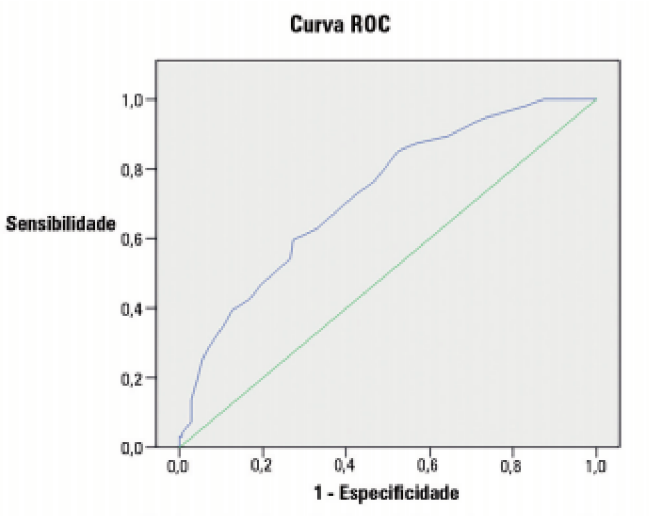
f_p = Número de previsões de alta classificadas incorretamente

f_n = Número de previsões de baixa classificadas incorretamente

4.5.1 Curva Característica de Operação do Receptor

Outra forma para avaliar o desempenho de um classificador binário é usando o parâmetro ROC (Característica de Operação do Receptor). Esse é um método gráfico em que uma curva é desenhada plotando a taxa de verdadeiros positivos contra a taxa de falsos positivos em vários valores limites. A curva ROC mostra a relação entre sensibilidade e especificidade. Quando essa se aproxima da borda esquerda e da borda superior do gráfico ROC, isso indica que o teste é preciso. Assim, quanto mais próxima ela estiver da borda superior e esquerda, mais preciso será o teste. No entanto, se ela estiver próxima da diagonal de 45 graus do espaço ROC, isso significa que o teste não é preciso (KHAIDEM; SAHA; DEY, 2016). A Figura 12 exemplifica uma curva ROC:

Figura 12 – Exemplo de uma curva *ROC*.



Fonte: Hamshary et al. (2017).

4.5.2 Validação Cruzada utilizando $K - folds$

Na validação cruzada utilizando $k - folds$, o conjunto de dados é dividido em k segmentos. Dessa forma, um classificador é treinado usando $k - 1$ partes da amostra e um valor de erro é calculado testando o classificador no segmento restante - repetindo o procedimento até que todas as amostras sejam usadas como teste. Finalmente, a estimativa k-cv do erro é o valor médio dos erros cometidos em cada iteração (RODRIGUEZ; PEREZ; LOZANO, 2009). A Figura 13 ilustra esse processo:

Figura 13 – Validação cruzada para 5 folds.

Fonte: O autor.

Por conseguinte, através da validação cruzada, é possível observar a média e desvio padrão do erro - ou acurácia - e detectar um possível *overfitting* no processo de treinamento dos algoritmos. O *overfitting* é um problema comum nos algoritmos supervisionados de aprendizado de máquina. Segundo Jabbar e Khan (2015), é o fenômeno detectado quando um algoritmo de aprendizado se ajusta ao conjunto de dados de treinamento tão bem que o ruído e as peculiaridades dos dados de treinamento são memorizados. Logo, o desempenho do modelo cai drasticamente quando é testado em um conjunto de dados desconhecidos.

4.6 Considerações Finais deste Capítulo

Este capítulo descreve os métodos adotados para realização da previsões de três ativos presentes na bolsa de valores brasileira. Na seção 4.2, foram apresentadas as bases de dados e as técnicas usadas para gerar as classes e dividir os conjuntos de treinamento e teste.

Na seção 4.3 e 4.4, foram a abordadas as técnicas de pré-processamento dos dados e extração dos atributos - indicadores técnicos - que servirão de entradas para os modelos de *machine learning*. Por conseguinte, a última seção deste capítulo se destinou a explicar os métodos para avaliação das previsões realizadas.

No próximo capítulo é exibido o panorama final de execução deste trabalho, com todos os resultados obtidos e a comparação da eficiência dos algoritmos propostos.

Capítulo 5

Análise dos Resultados

Este capítulo pretende discutir sobre os resultados e as técnicas utilizadas para avaliar os modelos construídos através dos horizontes de previsão propostos. Assim, os resultados de duas técnicas de separação das amostras de treino e teste são comparados. Para cada técnica proposta, foram efetuadas diversas previsões usando os algoritmos de *machine learning* citados no capítulo 4. A avaliação do desempenho dos modelos é feita através de métodos estatísticos, os quais foram discutidos na seção anterior.

5.1 Resultados Obtidos Utilizando a Técnica de Separação Aleatória

Os algoritmos citados foram treinados duas vezes cada (com e sem a técnica de embaralhamento) para gerarem previsões de 30, 60 e 90 dias. Desse modo, foi possível observar o comportamento das previsões em diferentes janelas de tempo. As Tabelas 8,9 e 10 exibem os resultados obtidos - com embaralhamento - para os períodos pesquisados respectivamente.

Por meio da Tabela 8, para as previsões de 30 dias, é possível observar que para o ativo VALE3, o algoritmo RF demonstrou maior acurácia (96%). No entanto, para as outras ações pesquisadas os dois *kernels* SVM utilizados obtiveram resultados superiores - porém não atingiram o mesmo valor do título anterior. Já as previsões da RNA foram muito inferiores se comparadas às estimativas dos outros modelos.

Na Tabela 9, horizonte de previsão de 60 dias, os modelos RF, SVM-GHI e SVM-RBF obtiveram a maior acurácia (97% no ativo PETR4). No entanto, da mesma maneira citada anteriormente, a RNA demostrou-se menos eficaz que os outros algoritmos utilizados.

Para o período de previsão de 90 dias (Tabela 10), em linhas gerais, os resultados obtidos demonstraram-se melhores em comparação às outras janelas de tempo pesquisadas. O algoritmo SVM-RBF atingiu a maior acurácia (97%) para os ativos PETR4 e ITUB4. Apesar dos melhores resultados alcançados pela RNA nesta janela, estes ainda

Tabela 8 – Previsões 30 dias utilizando divisão aleatória.

	Algoritmo	Acurácia	Precisão	Recall	Especificidade
VALE3	RF	96%	0.96	0.97	0.95
	RNA	71%	0.72	0.81	0.56
	SVM - GHI	93%	0.95	0.92	0.94
	SVM - RBF	91%	0.94	0.90	0.92
PETR4	RF	92%	0.95	0.92	0.93
	RNA	77%	0.85	0.74	0.81
	SVM - GHI	92%	0.94	0.93	0.92
	SVM - RBF	93%	0.97	0.91	0.96
ITUB4	RF	91%	0.91	0.93	0.89
	RNA	78%	0.84	0.74	0.83
	SVM - GHI	92%	0.94	0.90	0.93
	SVM -RBF	92%	0.93	0.9	0.91

Fonte: O autor.

Tabela 9 – Previsões 60 dias utilizando divisão aleatória.

	Algoritmo	Acurácia	Precisão	Recall	Especificidade
VALE3	RF	91%	0.94	0.92	0.91
	RNA	78%	0.83	0.83	0.71
	SVM - GHI	92%	0.95	0.92	0.92
	SVM - RBF	94%	0.96	0.94	0.94
PETR4	RF	97%	0.96	0.98	0.95
	RNA	82%	0.84	0.84	0.80
	SVM - GHI	97%	0.96	0.98	0.95
	SVM - RBF	97%	0.97	0.97	0.96
ITUB4	RF	94%	0.95	0.95	0.93
	RNA	78%	0.79	0.83	0.71
	SVM - GHI	93%	0.95	0.93	0.93
	SVM -RBF	90%	0.94	0.89	0.92

Fonte: O autor.

se mantiveram menores comparados aos outros modelos usados.

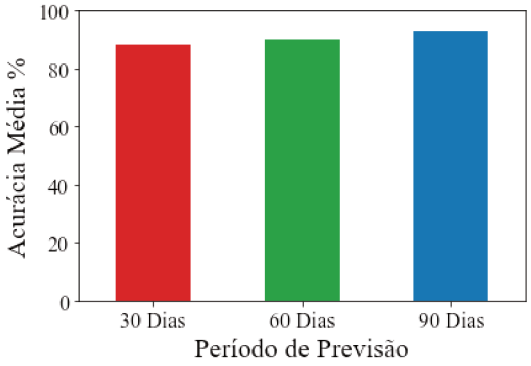
Com o intuito de observar a eficiência dos modelos ao longo das janelas de previsão, foram calculadas as médias das acurácias -utilizando todos os modelos - para cada período previsto. A figura 14 ilustra o resultado desses cálculos. É possível aferir que houve um pequeno aumento da acurácia à medida que se aumenta o numero de dias preditos.

Tabela 10 – Previsões 90 dias utilizando divisão aleatória.

	Algoritmo	Acurácia	Precisão	Recall	Especificidade
VALE3	RF	94%	0.98	0.93	0.97
	RNA	83%	0.86	0.86	0.77
	SVM - GHI	93%	0.97	0.91	0.95
	SVM - RBF	93%	0.98	0.90	0.98
PETR4	RF	96%	0.95	0.99	0.93
	RNA	82%	0.80	0.91	0.68
	SVM - GHI	95%	0.94	0.99	0.90
	SVM - RBF	97%	0.96	0.99	0.94
ITUB4	RF	96%	0.98	0.96	0.98
	RNA	89%	0.91	0.91	0.87
	SVM - GHI	96%	0.98	0.96	0.97
	SVM -RBF	97%	0.99	0.96	0.99

Fonte: O autor.

Figura 14 – Acurácia Média usando a técnica de embaralhamento.



Fonte: O autor.

5.2 Resultados Obtidos Utilizando a Técnica de Separação Temporal

Os mesmos dados utilizados anteriormente foram usados para gerarem previsões dos ativos sem a utilização da técnica de embaralhamento durante o treinamento e teste dos algoritmos. As Tabelas 11, 12 e 13 exibem os resultados obtidos para esses modelos.

Observando os valores obtidos para as previsões de 30 dias (Tabela 11), fica nítida a diferença dos resultados das duas técnicas utilizadas. Nessa janela de tempo, as acurácias dos algoritmos ficaram entre 41% e 67% (contra 71% e 96% da técnica anterior). Além disso, os outros indicadores estatísticos - precisão, *recall* e especificidade - também

Tabela 11 – Previsões 30 dias utilizando divisão temporal.

	Algoritmo	Acurácia	Precisão	Recall	Especificidade
VALE3	RF	52%	0.49	0.89	0.22
	RNA	46%	0.46	1	0.01
	SVM - GHI	51%	0.47	0.72	0.34
	SVM - RBF	47%	0.4	0.62	0.34
PETR4	RF	51%	1	0.19	1
	RNA	67%	0.7	0.8	0.47
	SVM - GHI	41%	1	0.01	1
	SVM - RBF	55%	0.74	0.38	0.79
ITUB4	RF	49%	1	0.03	1
	RNA	57%	0.82	0.23	94
	SVM - GHI	51%	0.59	0.24	0.81
	SVM -RBF	53%	0.73	0.16	0.94

Fonte: O autor.

demonstraram que houve uma queda significativa no desempenho dos modelos.

Tabela 12 – Previsões 60 dias utilizando divisão temporal.

	Algoritmo	Acurácia	Precisão	Recall	Especificidade
VALE3	RF	62%	0.54	0.99	0.31
	RNA	45%	0.45	1	0
	SVM - GHI	60%	0.53	0.92	0.33
	SVM - RBF	51%	0.44	0.62	0.34
PETR4	RF	54%	0.85	0.25	94
	RNA	61%	0.63	0.8	0.35
	SVM - GHI	42%	0	0	1
	SVM - RBF	62%	0.73	0.54	0.71
ITUB4	RF	45%	0	0	1
	RNA	64%	0.8	0.45	86
	SVM - GHI	48%	0.9	0.07	0.99
	SVM -RBF	47%	0.55	0.22	0.77

Fonte: O autor.

Analisando a Tabela 12, os resultados desta se mostraram inferiores em relação aos da Tabela 9. As acurácias dos modelos ficaram entre 42% e 64%. Nesse sentido, também é possível observar que os outros indicadores estatísticos tiveram um resultado muito inferior em relação a mesma janela de previsão da técnica anterior.

De outro modo, para a janela de previsão de 90 dias, Tabela 13, as acurácias obtidas tiveram um aumento satisfatório, em relação a Tabela 10, alcançando até 77% para o caso

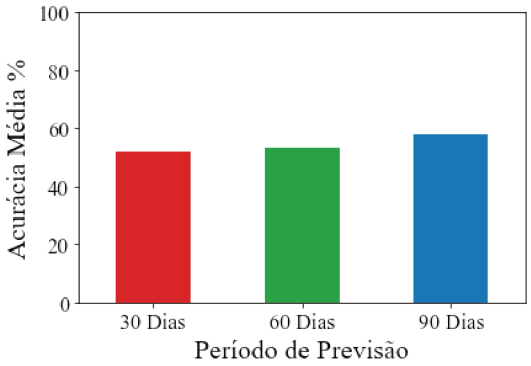
Tabela 13 – Previsões 90 dias utilizando divisão temporal.

	Algoritmo	Acurácia	Precisão	Recall	Especificidade
VALE3	RF	60%	0.5	0.95	0.36
	RNA	52%	0.46	1	0.2
	SVM - GHI	68%	0.56	0.92	0.52
	SVM - RBF	62%	0.51	0.99	0.37
PETR4	RF	76%	0.98	0.72	93
	RNA	18%	0	0	1
	SVM - GHI	57%	0.84	0.58	0.52
	SVM - RBF	77%	0.94	0.78	0.76
ITUB4	RF	57%	0.82	0.23	0.94
	RNA	54%	0	0	1
	SVM - GHI	55%	1	0.02	1
	SVM -RBF	60%	1	0.12	1

Fonte: O autor.

do ativo PETR4. No entanto, a acurácia obtida com o modelo *RNA* foi de apenas 18%. A figura 15 ilustra as acurácias médias alcançadas para cada janela de previsão utilizando a técnica de separação por ordem temporal. É possível afirmar que houve, dessa vez mais evidente, um aumento no desempenho das predições ao passo que se aumenta as janelas antevistas.

Figura 15 – Acurácia Média usando a ordem temporal.



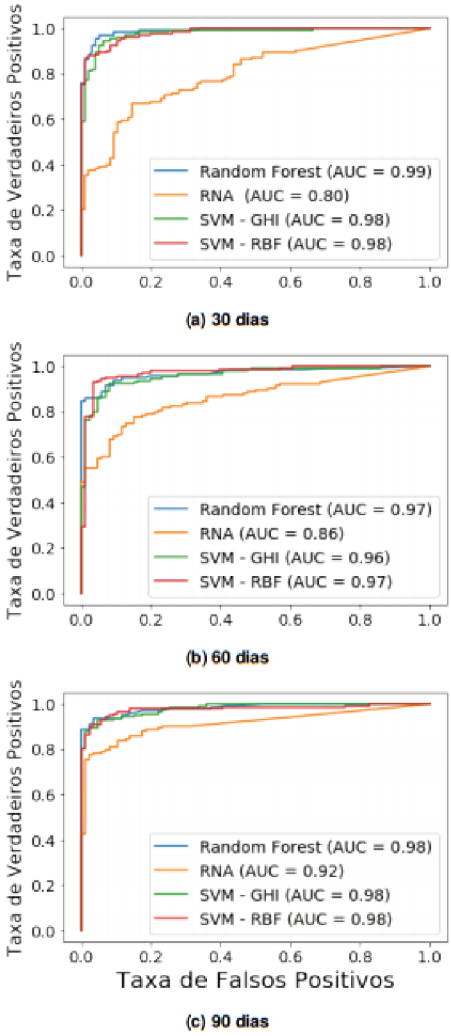
Fonte: O autor.

5.3 Curvas ROC

Para avaliar a qualidade das previsões, foram obtidas as curvas ROC para todos os modelos desenvolvidos. As figuras 16, 17 e 18 ilustram o cálculo das curvas ROC para os

modelos desenvolvidos - utilizando a técnica de embaralhamento - aplicados aos papeis previstos:

Figura 16 – Curvas ROC VALE3 utilizando embaralhamento.

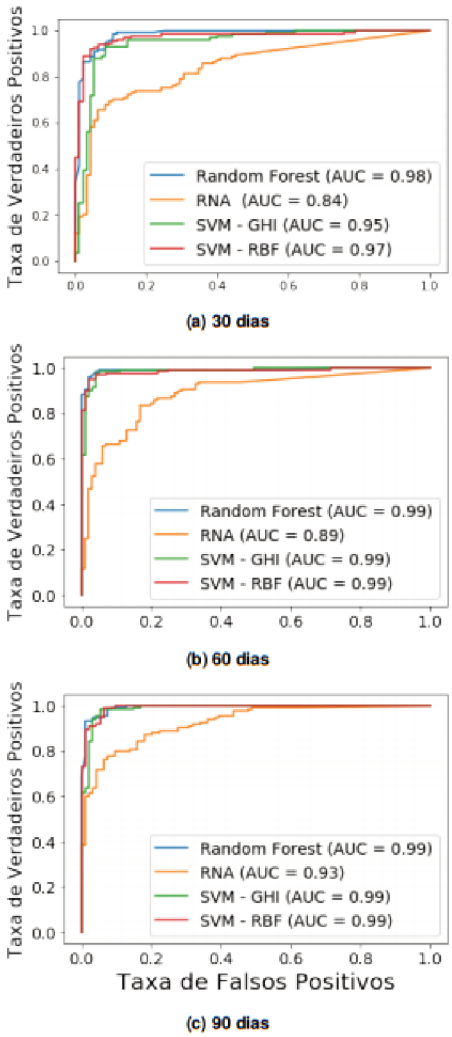


Fonte: O autor.

É possível observar nas três ações pesquisadas que, de forma geral, a medida que se aumentam os dias de previsão, a precisão dos algoritmos aumenta. Isso fica mais claro ao analisar os gráficos do modelo RNA. No entanto, os demais (RF, SVM-GHI, SVM-RBF) também demonstram uma pequena melhora com o aumento dos dias previstos.

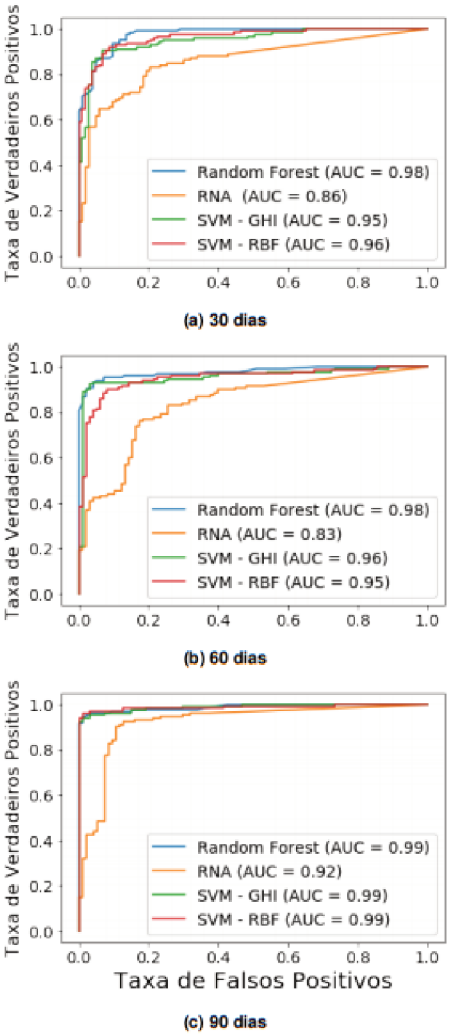
Para as ações ITUB4, foi possível verificar através da curva ROC que para o modelo RNA houve uma piora dos resultados de 60 dias em relação às previsões de 30 dias. No entanto, em um panorama geral, é concebível afirmar que os resultados das predições melhoram com o aumento da janela de tempo a prever.

Figura 17 – Curvas ROC PETR4 utilizando embaralhamento.



Fonte: O autor.

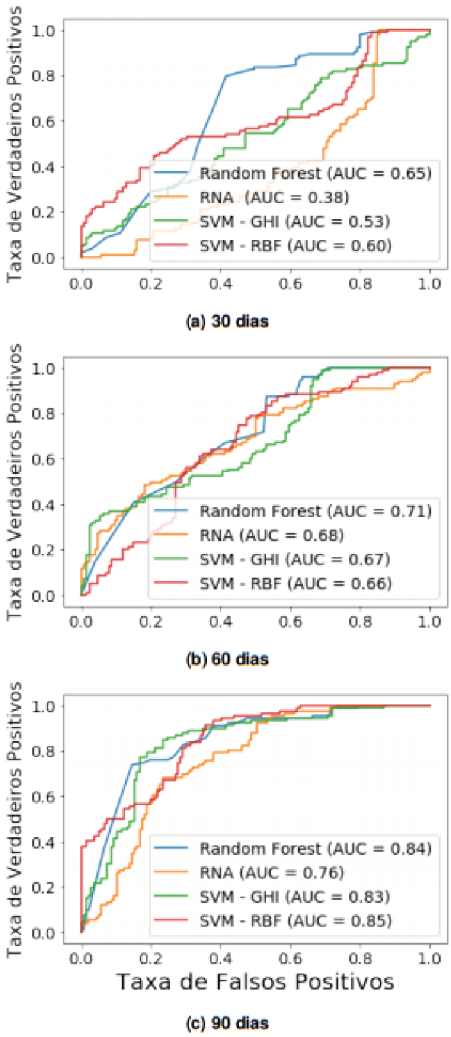
Figura 18 – Curvas ROC ITUB4 utilizando embaralhamento.



Fonte: O autor.

Foram calculadas também as curvas ROC para a técnica de separação dos dados de treinamento e teste utilizando a ordem temporal da séries históricas. As figuras 19, 20 e 21 ilustram esse processo.

Figura 19 – Curvas ROC VALE3 utilizando a divisão temporal.

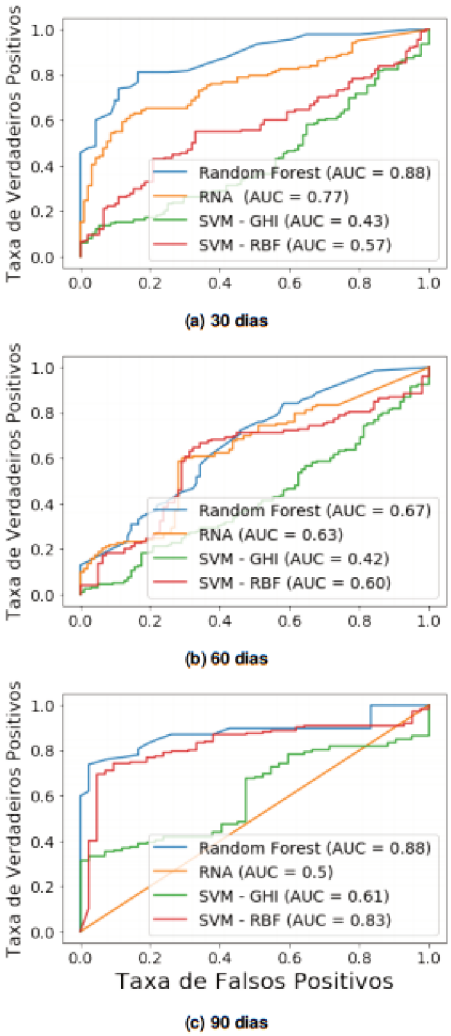


Fonte: O autor.

Averiguando os resultados obtidos para ações VALE3, fica evidente que a performance dos modelos utilizando esta última técnica de separação decaiu bastante em relação à anterior. Para a janela de previsão de 30 dias, o coeficiente Área Abaixo da Curva (do inglês *Area Under Curve* - AUC) ficou entre 0,38 (RNA) e 0.65 (RF), contra 0,80 e 0,99 das previsões para VALE3 empregando a técnica de separação aleatória.

Por meio da análise das curvas ROC dos ativos PETR4 e ITUB4, fica claro que as performances dos modelos caíram bastante. Entretanto, também é possível afirmar que os resultados melhoram muito à medida que se aumenta a janela de previsão.

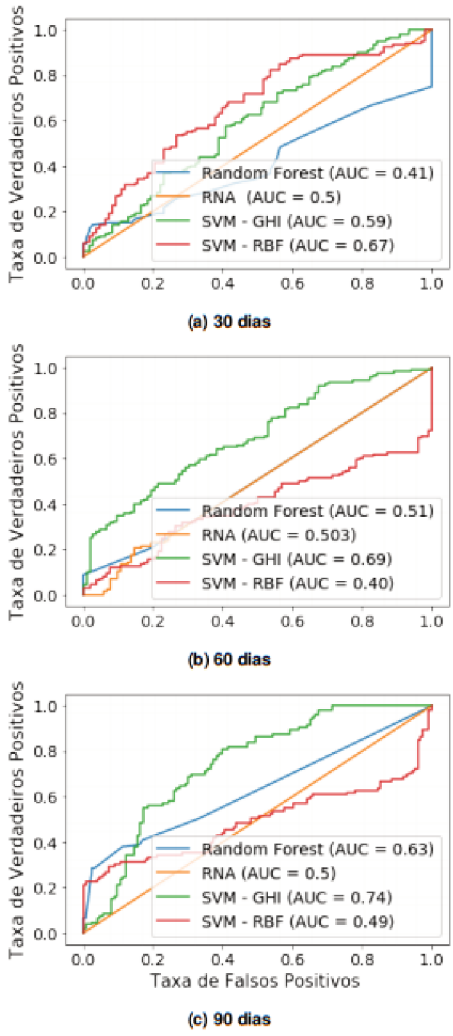
Figura 20 – Curvas ROC PETR4 utilizando a divisão temporal.



Fonte: O autor.

Para essa última técnica de separação das amostras, a melhora dos resultados com o aumento do período de previsão é mais evidente do que no método anterior. Nesse sentido, isso vai de encontro aos resultados relatados na literatura - como em Khaidem, Saha e Dey (2016) e Basak et al. (2019) -, em que também há uma relação entre melhor performance e maior período de predição.

Figura 21 – Curvas ROC ITUB4 utilizando a divisão temporal.



Fonte: O autor.

5.4 Validação Cruzada

Os resultados demonstrados anteriormente avaliaram a performance dos modelos nas amostras de teste, onde dados desconhecidos são usados para prever os preços. No entanto, para atribuir maior confiança às predições é necessário avaliar o desempenho dos algoritmos durante a fase de treinamento. Utilizando a biblioteca $scikit - learn$ é possível realizar a validação $k - cv$ para os algoritmos RF, SVM-RBF e SVM-GHI, os quais obtiveram também os melhores resultados. Foi realizada a validação cruzada utilizando 10 $folds$ nas amostras de treino de cada modelo. Desse modo, cada iteração através dos $folds$ gerou uma taxa de acertos (acurácia) e, por fim, foi obtido a acurácia média e desvio padrão das dez iterações feitas em cada algoritmo proposto.

As Tabelas 14, 15 e 16 ilustram os valores obtidos aplicando o método de validação

citado para os modelos que usam a técnica de separação aleatória:

Tabela 14 – Validação cruzada para ações VALE3 utilizando embaralhamento.

Algoritmo	Período de previsão	Acurácia média	Desvio Padrão
RF	30 dias	0.93	0.04
SVM - RBF	30 dias	0.89	0.05
SVM - GHI	30 dias	0.91	0.05
RF	60 dias	0.93	0.05
SVM - RBF	60 dias	0.92	0.06
SVM - GHI	60 dias	0.91	0.07
RF	90 dias	0.96	0.03
SVM - RBF	90 dias	0.96	0.02
SVM - GHI	90 dias	0.96	0.03

Fonte: O autor.

Através da comparação entre os valores obtidos para a acurácia média e desvio padrão, é possível afirmar que para essa técnica não houve *overfitting* para os modelos que previram os ativos VALE3, PETR4 e ITUB4. Isso fica evidente ao observar a alta taxa de acertos e o baixo desvio, o que significa que não houve uma grande diferença de acertos ao testar o algoritmo em amostras desconhecidas. Além disso, reforçando as métricas anteriores, é possível verificar também que houve uma melhora na performance dos modelos em maiores janelas de previsão.

Tabela 15 – Validação cruzada para ações PETR4 utilizando embaralhamento.

Algoritmo	Período de previsão	Acurácia média	Desvio Padrão
RF	30 dias	0.93	0.05
SVM - RBF	30 dias	0.92	0.07
SVM - GHI	30 dias	0.92	0.06
RF	60 dias	0.95	0.05
SVM - RBF	60 dias	0.94	0.05
SVM - GHI	60 dias	0.94	0.05
RF	90 dias	0.96	0.03
SVM - RBF	90 dias	0.94	0.06
SVM - GHI	90 dias	0.96	0.03

Fonte: O autor.

Tabela 16 – Validação cruzada para ações ITUB4 utilizando embaralhamento.

Algoritmo	Período de previsão	Acurácia média	Desvio Padrão
RF	30 dias	0.92	0.05
SVM - RBF	30 dias	0.90	0.07
SVM - GHI	30 dias	0.90	0.09
RF	60 dias	0.94	0.05
SVM - RBF	60 dias	0.92	0.06
SVM - GHI	60 dias	0.93	0.04
RF	90 dias	0.95	0.04
SVM - RBF	90 dias	0.96	0.04
SVM - GHI	90 dias	0.94	0.05

Fonte: O autor.

Por conseguinte, com o intuito de averiguar a causa da baixa performance dos modelos utilizando a separação temporal, foi realizada a validação cruzada para os modelos desenvolvidos utilizando os mesmos parâmetros da técnica anterior.

As Tabelas 17, 18 e 19 exibem os valores obtidos:

Tabela 17 – Validação cruzada para ações VALE3 utilizando divisão temporal.

Algoritmo	Período de previsão	Acurácia média	Desvio Padrão
RF	30 dias	0.47	0.42
SVM - RBF	30 dias	0.49	0.28
SVM - GHI	30 dias	0.52	0.36
RF	60 dias	0.48	0.31
SVM - RBF	60 dias	0.64	0.27
SVM - GHI	60 dias	0.51	0.34
RF	90 dias	0.72	0.33
SVM - RBF	90 dias	0.74	0.35
SVM - GHI	90 dias	0.65	0.36

Fonte: O autor.

Por meio dos valores de acurácia média e desvio padrão obtidos, é possível afirmar que os modelos desenvolvidos utilizando os 3 modelos citados nas Tabelas 17, 18 e 19, tiveram $overfitting$. Isso fica mais claro ao examinar os valores de desvio gerados, entre 0.27 e 0.42 para o ativo VALE3. Dessa maneira, é possível afirmar que algumas amostras das interações obtiveram alta acurácia, e outras um desempenho muito baixo - caracterizando o problema referido.

Além disso, as Tabelas demonstram uma tendência de maior acurácia média para as janelas de 60 e 90 dias. Por outro lado, apenas a validação cruzada não permite identificar de forma clara o melhor algoritmo a ser utilizado (RF, SVM-RBF ou SVM-GHI). Dessa forma, os outros parâmetros estatísticos devem ser levados em conta.

Tabela 18 – Validação cruzada para ações PETR4 utilizando divisão temporal.

Algoritmo	Período de previsão	Acurácia média	Desvio Padrão
RF	30 dias	0.48	0.34
SVM - RBF	30 dias	0.55	0.39
SVM - GHI	30 dias	0.50	0.37
RF	60 dias	0.72	0.30
SVM - RBF	60 dias	0.74	0.30
SVM - GHI	60 dias	0.70	0.26
RF	90 dias	0.66	0.50
SVM - RBF	90 dias	0.62	0.52
SVM - GHI	90 dias	0.62	0.38

Fonte: O autor.

Tabela 19 – Validação cruzada para ações ITUB4 utilizando divisão temporal.

Algoritmo	Período de previsão	Acurácia média	Desvio Padrão
RF	30 dias	0.51	0.34
SVM - RBF	30 dias	0.59	0.26
SVM - GHI	30 dias	0.62	0.26
RF	60 dias	0.61	0.48
SVM - RBF	60 dias	0.56	0.31
SVM - GHI	60 dias	0.65	0.28
RF	90 dias	0.68	0.40
SVM - RBF	90 dias	0.69	0.24
SVM - GHI	90 dias	0.63	0.38

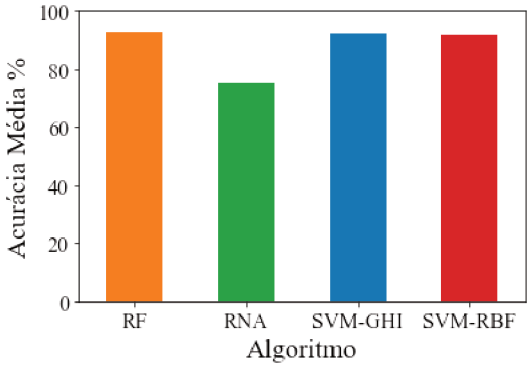
Fonte: O autor.

5.5 Comparação dos Modelos Desenvolvidos

Com o intuito de eleger o algoritmo com a melhor performance nas previsões realizadas, foram calculadas as acurácias médias para cada modelo - RF, RNA, SVM-GHI e SVM-RBF - utilizando os resultados dos três ativos previstos. As figuras 22, 23 e 24 ilustram respectivamente esses cálculos para a técnica de divisão aleatória.

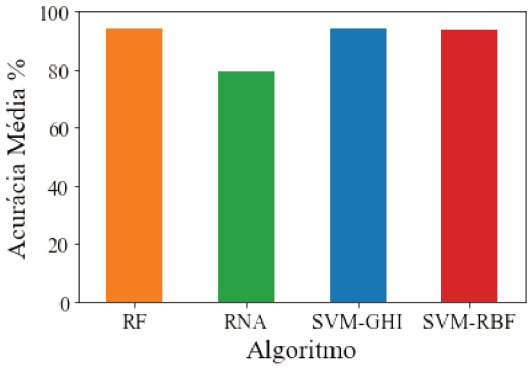
Analisando os resultados é possível notar uma clara inferioridade do modelo RNA, no entanto, sua construção envolve muitas complexidades, como por exemplo a escolha dos hiperparâmetros (número de camadas escondidas, função de ativação, taxa de aprendizado, dentre outros), fato esse que pode afetar muito a eficiência da rede neural. Ademais, os algoritmos RF, SVM-GHI e SVM-RBF obtiveram desempenhos muito altos e semelhantes. Além disso, esses últimos, diferentemente da RNA, são facilmente implementados. Dessa forma, é necessário mais estudos (previsões de outros ativos) que possam identificar as vantagens e desvantagens dessas ferramentas pesquisadas.

Figura 22 – Taxa de Acertos média para horizonte de 30 dias utilizando embaralhamento.



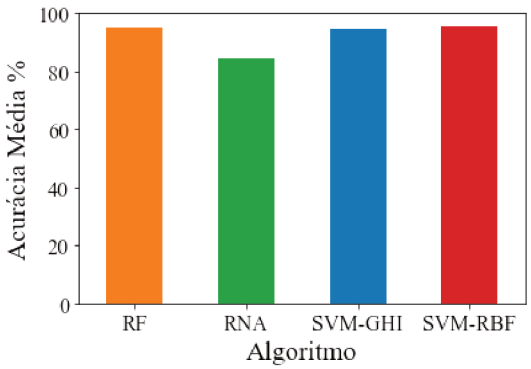
Fonte: O autor.

Figura 23 – Taxa de Acertos média para horizonte de 60 dias utilizando embaralhamento.



Fonte: O autor.

Figura 24 – Taxa de Acertos média para horizonte de 90 dias utilizando embaralhamento.

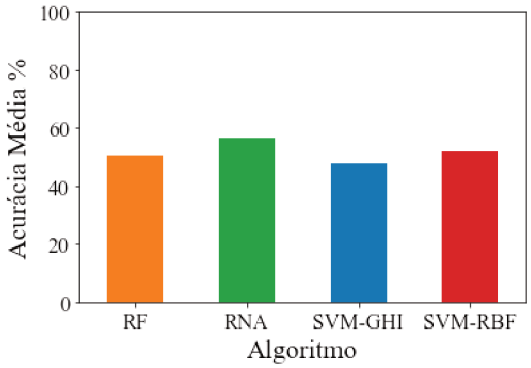


Fonte: O autor.

Diferentemente do primeiro caso, para a técnica de divisão temporal dos dados os resultados foram, como já citado, muito inferiores. No entanto, é importante verificar que o modelo RNA obteve acurácia média superior aos outros nas janelas de previsão de 30 e 60 dias. Para a o período de 90 dias, os demais algoritmos obtiveram resultados parecidos, e a RNA uma acurácia média de 41%. Desse modo, isso demonstra que os quatro modelos podem ser utilizados no dia a dia (a depender do ativo a ser previsto), pois essa técnica de divisão das amostras de treinamento e teste aproxima-se mais de um sistema que operará de fato.

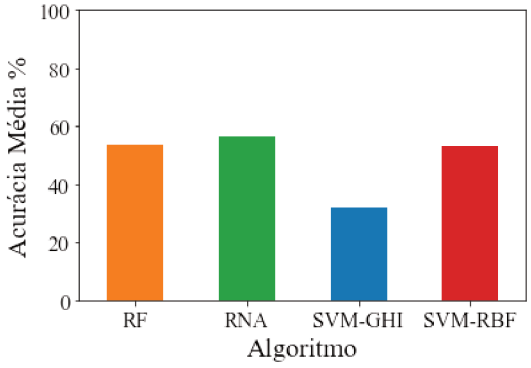
As figuras 25, 26 e 27 ilustram esses resultados alcançados para a técnica citada:

Figura 25 – Taxa de Acertos média para horizonte de 30 dias utilizando e divisão temporal.



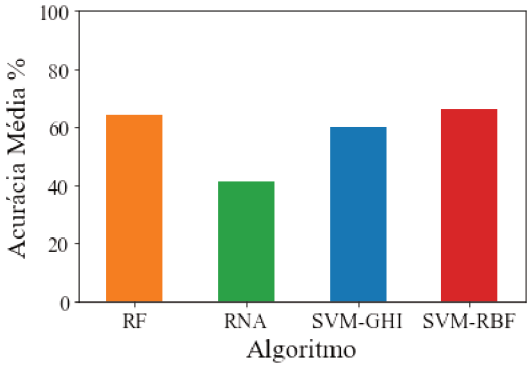
Fonte: O autor.

Figura 26 – Taxa de Acertos média para horizonte de 60 dias utilizando divisão temporal.



Fonte: O autor.

Figura 27 – Taxa de Acertos média para horizonte de 90 dias utilizando divisão temporal.



Fonte: O autor.

5.6 Considerações Finais deste Capítulo

Este capítulo apresentou de forma detalhada os resultados das previsões realizadas, para os horizontes de 30, 60 e 90 dias úteis, utilizando os algoritmos desenvolvidos para as ações VALE3, PETR4, ITUB4, negociadas na bolsa de valores brasileira (Ibovespa).

A seção 5.1 apresentou os índices de acertos obtidos utilizando a técnica de separação dos dados aleatória, demonstrando os parâmetros de acurácia, precisão, *recall* e especificidade. A seção 5.2 apresenta os mesmos dados, porém, utilizando a divisão temporal dos dados.

A seção 5.3, por sua vez, demonstrou o cálculo da curva característica de operação do receptor para os modelos desenvolvidos, enfatizando a diferença dos resultados para diferentes janelas de previsão.

Na seção 5.4 foram apresentados os dados resultantes de validação cruzada realizada nos dados de treinamento. Essa técnica permite identificar *overfitting* nos modelos desenvolvidos.

No próximo capítulo serão apresentadas de maneira estruturada as conclusões e contribuições deste trabalho, e serão indicadas, também, as possibilidades futuras de trabalhos relacionados ao tema de estudo.

Capítulo 6

Conclusão

6.1 Introdução

O desenvolvimento desta pesquisa resultou na exposição de quatro algoritmos de aprendizado de máquina destinados à predição do mercado financeiro. Além disso, diferentes técnicas de manipulação e validação dos dados foram apresentadas e discutidas.

O trabalho procurou discutir as vantagens e desvantagens dos modelos de classificação utilizados. A técnica de divisão temporal dos dados demonstrou ser mais plausível de aplicação em um sistema real, apesar dos baixos resultados em relação à outra técnica empregada.

6.2 Principais Contribuições

A principal contribuição deste trabalho consiste na proposição de um método de classificação de tendência dos ativos VALE3, PETR4 e ITUB4 utilizando diferentes algoritmos.

A pesquisa procurou reproduzir resultados já presentes na literatura, porém aplicados ao mercado brasileiro. Os resultados obtidos mostram que provavelmente as altas taxas de acertos encontradas na bibliografia não são reproduzíveis no dia a dia. Este fato se deve ao método de validação utilizado que separa treinamento e teste de maneira aleatória.

A função SVM-GHI demonstrou ser capaz de contornar o problema de previsões financeiras, reforçando a afirmação de Boughorbel, Tarel e Boujemaa (2005) em relação à sua aplicação em outros contextos, além do reconhecimento de imagens.

6.3 Trabalhos Futuros

Para trabalhos futuros a ideia é testar novos algoritmos de *machine learning*, como as Redes LSTM (muito utilizada para séries temporais), redes convolucionais e modelos híbridos que combinem esses algoritmos citados.

Futuramente os modelos de previsão também podem incluir outras métricas, como preço previsto (regressão), nível de risco e probabilidade de ocorrência. No mais, o desenvolvimento de um sistema de gerenciamento de carteiras utilizando algoritmos genéticos e *deep learning* também será um complemento para esta pesquisa.

Referências

ALKHATIB, K. et al. Stock price prediction using k-nearest neighbor (knn) algorithm. **International Journal of Business, Humanities and Technology**, v. 3, n. 3, p. 32–44, 2013.

AMIT, Y.; GEMAN, D. Shape Quantization and Recognition with Randomized Trees. **Neural Computation**, 1997. ISSN 08997667.

APOSTOLIDIS-AFENTOULIS, V. Svm classification with linear and rbf kernels. 07 2015. Disponível em: <https://www.academia.edu/13811676/SVM_Classification_with_Linear_and_RBF_kernels/>. Acesso em: 13-04-2020.

BARBOZA, F.; KIMURA, H.; ALTMAN, E. Machine learning models and bankruptcy prediction. **Expert Systems with Applications**, 2017. ISSN 09574174.

BASAK, S. et al. Predicting the direction of stock market prices using tree-based classifiers. **The North American Journal of Economics and Finance**, Elsevier, v. 47, p. 552–567, 2019. Disponível em: <https://doi.org/10.1016/j.najef.2018.06.013>.

BEYAZ, E. et al. Comparing technical and fundamental indicators in stock price forecasting. In: IEEE. **2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)**. 2018. p. 1607–1613. Disponível em: <https://doi.org/10.1109/HPCC/SmartCity/DSS.2018.00262>.

BISHOP, C. M. **Pattern Recogniton and Machine Learning**. [S.l.: s.n.], 2006. ISBN 978-0-387-31073-2.

BOUGHORBEL, S.; TAREL, J. P.; BOUJEMAA, N. Generalized histogram intersection kernel for image recognition. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.: s.n.], 2005. ISBN 0780391349. ISSN 15224880.

BREIMAN, L. Random forests. **Machine Learning**, 2001. ISSN 08856125.

CORTES, C.; VAPNIK, V. Support-Vector Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. Machine Learning, 20(3), 273–297. **Machine Learning**, 1995. ISSN 15730565. Disponível em: <https://doi.org/10.1007/BF00994018>.

CUTLER, A.; CUTLER, D. R.; STEVENS, J. R. **Random Forests**. Boston, MA: Springer US, 2012. 157–175 p. ISBN 978-1-4419-9326-7. Disponível em: <https://doi.org/10.1007/978-1-4419-9326-7_5>.

DAI, Y.; ZHANG, Y. **Machine learning in stock price trend forecasting**. [S.l.]: Stanford University Stanford, 2013.

GUNN, S. R. **Support Vector Machines for Classification and Regression**. 1998. Disponível em: <https://doi.org/10.1.1.10.7171>.

HAMMAD, M.; WANG, K. Parallel score fusion of ecg and fingerprint for human authentication based on convolution neural network. **Computers & Security**, Elsevier, v. 81, p. 107–122, 2019. Disponível em: <https://doi.org/10.1016/j.cose.2018.11.003>.

HAMSHARY, A. A. E. E. et al. Prevalência da falência de múltiplos órgãos na unidade de terapia intensiva pediátrica: comparação dos escores pediatric risk of mortality iii e pediatric logistic organ dysfunction para predição de mortalidade. **Revista Brasileira de Terapia Intensiva**, SciELO Brasil, v. 29, n. 2, p. 206–212, 2017. Disponível em: <http://dx.doi.org/10.5935/0103-507x.20170029.>

HAN, S.; CHEN, R.-C. Using svm with financial statement analysis for prediction of stocks. **Communications of the IIMA**, v. 7, n. 4, p. 8, 2007.

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

HO, T. K. Random decision forests. In: **Proceedings of the International Conference on Document Analysis and Recognition, ICDAR**. [S.l.: s.n.], 1995. ISBN 0818671289. ISSN 15205363.

HOSEINZADE, E.; HARATIZADEH, S. Cnnpred: Cnn-based stock market prediction using a diverse set of variables. **Expert Systems with Applications**, Elsevier, v. 129, p. 273–285, 2019. Disponível em: <https://doi.org/10.1016/j.eswa.2019.03.029>.

HSU, C. W.; LIN, C. J. A comparison of methods for multiclass support vector machines. **IEEE Transactions on Neural Networks**, 2002. ISSN 10459227. Disponível em: <https://doi.org/10.1109/72.991427>.

JABBAR, H.; KHAN, R. Z. Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). **Computer Science, Communication and Instrumentation Devices**, p. 163–172, 2015. Disponível em: <https://doi.org/10.3850/978-981-09-5247-1_017>.

JOACHIMS, T. Text categorization with support vector machines: Learning with many relevant features. In: **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)**. [S.l.: s.n.], 1998. ISBN 3540644172. ISSN 16113349.

KEERTHI, S. S.; LIN, C. J. Asymptotic behaviors of support vector machines with gaussian kernel. **Neural Computation**, 2003. ISSN 08997667. Disponível em: <https://doi.org/10.1162/089976603321891855>.

Keras SIG. **About Keras**. 2020. Disponível em: <https://keras.io/about/>. Acesso em: 13-04-2020.

KHAIDEM, L.; SAHA, S.; DEY, S. R. Predicting the direction of stock market prices using random forest. **arXiv preprint arXiv:1605.00003**, 04 2016.

KRANTZ, M. **Fundamental analysis for dummies**. [S.l.]: John Wiley & Sons, 2016. ISBN 978-1-119-26360-9.

LABOISSIERE, L. A.; FERNANDES, R. A.; LAGE, G. G. Maximum and minimum stock price forecasting of Brazilian power distribution companies based on artificial neural networks. **Applied Soft Computing Journal**, 2015. ISSN 15684946.

LEMENKOVA, P. Processing oceanographic data by Python libraries NumPy, SciPy and Pandas. **Aquatic Research**, ScientificWebJournals, v. 2, n. 2, p. 73–91, abr. 2019. Disponível em: <https://10.3153/AR19009>.

LIN, Y.; GUO, H.; HU, J. An svm-based approach for stock market trend prediction. In: IEEE. **The 2013 international joint conference on neural networks (IJCNN)**. 2013. p. 1–7. Disponível em: <https://doi.org/10.1109/IJCNN.2013.6706743>.

LO, A. W. Adaptive markets: Financial evolution at the speed of thought. Princeton University Press, 2017. Disponível em: <https://doi.org/10.1515/9781400887767>.

MADDALA, G. S.; WELLER, L. **Introdução à econometria**. [S.l.]: LTC, 2003.

MALKIEL, B. G. The efficient market hypothesis and its. In: **Journal of Economic Perspectives**. [S.l.: s.n.], 2003. ISSN 08953309.

MALKIEL, B. G.; FAMA, E. F. Efficient capital markets: A review of theory and empirical work*. **The Journal of Finance**, v. 25, n. 2, p. 383–417, 1970. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1970.tb00518.x>.

MURPHY, J. **Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications**. New York Institute of Finance, 1999. (New York Institute of Finance Series). ISBN 9780735200661. Disponível em: <https://books.google.com.br/books?id=5zhXEqdr_IcC>.

NELSON, D. M.; PEREIRA, A. C.; OLIVEIRA, R. A. de. Stock market's price movement prediction with lstm neural networks. In: IEEE. **2017 International joint conference on neural networks (IJCNN)**. 2017. p. 1419–1426. Disponível em: <https://doi.org/10.1109/IJCNN.2017.7966019>.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

POWERS, D. M. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. Bioinfo Publications, 2011.

ROCKEFELLER, B. **Technical analysis for dummies**. [S.l.]: John Wiley & Sons, 2019. ISBN 978-0470888001.

RODRIGUEZ, J. D.; PEREZ, A.; LOZANO, J. A. Sensitivity analysis of k-fold cross validation in prediction error estimation. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, v. 32, n. 3, p. 569–575, 2009. Disponível em: <https://doi.org/10.1109/TPAMI.2009.187>.

ROSSUM, G. van. **What is Python? Executive Summary**. 1998. Disponível em: <https://www.python.org/doc/essays/blurb/>. Acesso em: 13-04-2020.

SWAIN, M. J.; BALLARD, D. H. Color indexing. **International journal of computer vision**, Springer, v. 7, n. 1, p. 11–32, 1991. Disponível em: <https://doi.org/10.1007/BF00130487>.

TicTacTec LLC. **TA-Lib : Technical Analysis Library**. 2012. Disponível em: <https://ta-lib.org/http://www.ta-lib.org/>. Acesso em: 07-04-2020.

VERT, J.-P.; TSUDA, K.; SCHÖLKOPF, B. A primer on kernel methods. **Kernel methods in computational biology**, MIT press Cambridge, MA, v. 47, p. 35–70, 2004.

Wes McKinney. **pandas · PyPI**. 2010. Disponível em: <https://pypi.org/project/pandas/>. Acesso em: 07-04-2020.

Apêndices

APÊNDICE A

Códigos Fonte

A.1 Pré-Processamento dos Dados

Este algoritmo tem como função pré-processar as séries temporais, calculando as médias exponenciais, indicadores técnicos e as classes para treinamento dos modelos.

Requer bibliotecas:

- ❏ *Numpy*
- ❏ *Pandas*
- ❏ $Ta - Lib$

```
#PRÉ-PROCESSAMENTO DOS DADOS

import numpy as np
import pandas as pd
import talib as ta
from talib import MA_Type


Dias=  30 # HORIZONTE DE PREVISÃO 30 DIAS
Dias2= 60 # HORIZONTE DE PREVISÃO 60 DIAS
Dias3= 90 # HORIZONTE DE PREVISÃO 90 DIAS


test_size= 0.2
df = pd.read_csv ('VALE3.SA.csv') # Lê o arquivo VALE3, PETR4 ou ITUB4.

# Média móvel exponencial do Close, Open, High e Low

```

```
19 df['EMAC'] = ta.EMA(df['Close'], timeperiod=SM)
20
21 df['EMAO'] = ta.EMA(df['Open'], timeperiod=SM)
22 df['EMAH'] = ta.EMA(df['High'], timeperiod=SM)
23
24 df['EMAL'] = ta.EMA(df['Low'], timeperiod=SM)
25
26 df['RSI'] = ta.RSI(df['EMAC'], timeperiod=14) # Relative Strength Index
27
28 df['WILLR'] = ta.WILLR(df['EMAH'], df['EMAL'],
29 df['EMAC'], timeperiod=14) # Williams %R
30
31
32 df['macd'], df['macdsignal'], df['macdhist'] =
33 ta.MACD(df['EMAC'], fastperiod=14, slowperiod=24,
34 signalperiod=14) #Moving Average Convergence Divergence
35
36
37 df['OBV'] = ta.OBV(df['EMAC'], df['Volume']) # On Balance Volume
38
39
40 df['ROC'] = ta.ROC(df['EMAC'], timeperiod=14) #Price Rate of Change
41
42
43 df['fastk'], df['fastd'] =
44 ta.STOCHRSI(df['EMAC'], timeperiod=14,
45 fastk_period=5, fastd_period=3, fastd_matype=0) #Stochastic Oscillator
46
47
48 df.fillna(method="ffill", inplace= True)
49 df.fillna(method="bfill",inplace= True)
50
51 df= df.reset_index()
52 x_data=df
53 x_data.fillna(method="ffill", inplace= True)
54 x_data.fillna(method="bfill",inplace= True)
55
56
57 # Cálculo das Classes:
```

```
58
59 Y_T1['True']= X_Data['Close'].shift(-Dias)
60 Y_T2['True']= X_Data['Close'].shift(-Dias2)
61 Y_T3['True']= X_Data['Close'].shift(-Dias3)
62
63 Y_T1.fillna(method="ffill", inplace= True)
64 Y_T1.fillna(method="bfill", inplace= True)
65
66 Y_T2.fillna(method="ffill", inplace= True)
67 Y_T2.fillna(method="bfill", inplace= True)
68
69 Y_T3.fillna(method="ffill", inplace= True)
70 Y_T3.fillna(method="bfill", inplace= True)
71
72 X_Data['Y_T1']=Y_T1
73 X_Data['Y_T2']= Y_T2
74 X_Data['Y_T3']= Y_T3
75 X_Data.index= X_Data['Date']
76
77 X_Data['-Dias']= X_Data['Y_T1'].shift(Dias)
78 X_Data['-2Dias']= X_Data['Y_T1'].shift(2*Dias)
79 X_Data['-3Dias']= X_Data['Y_T1'].shift(3*Dias)
80
81 X_Data.fillna(method="ffill", inplace= True)
82 X_Data.fillna(method="bfill",inplace= True)
83 Y_T= pd.concat([Y_T1,Y_T2,Y_T3],axis=1)
```

A.2 Divisão Treinamento e Teste

Os dois algoritmos a seguir fazem a divisão dos dados de treinamento e teste, dividindo-os aleatoriamente ou de forma de cronológica respectivamente.

Requer bibliotecas:

- ❏ *Numpy*
- ❏ *Pandas*
- ❏ *Scikit − learn*

A.2.1 Divisão Aleatória

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import QuantileTransformer

X_train,X_test,y_train,y_test=
train_test_split(X_Data.head(X_Data.shape[0]-Dias3),
Y_T.head(X_Data.shape[0]-Dias3),
test_size= test_size, random_state=101) # Divisão Aleatória


X_train = X_train.drop(['Date',],axis=1)
X_test = X_test.drop(['Date',],axis=1)


scaler=QuantileTransformer()
scaler.fit(X_train)
X_train = pd.DataFrame(data=scaler.transform(X_train),
columns = X_train.columns,
index=X_train.index)

X_test = pd.DataFrame(data=scaler.transform(X_test),
columns = X_test.columns,
index=X_test.index)
```

A.2.2 Divisão Temporal

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import QuantileTransformer

X_train,X_test,y_train,y_test=
train_test_split(X_Data.head(X_Data.shape[0]-Dias3),
Y_T.head(X_Data.shape[0]-Dias3),
test_size= test_size, shuffle= False) #Divisão temporal dos dados

X_train = X_train.drop(['Date',],axis=1)
X_test = X_test.drop(['Date',],axis=1)


scaler=QuantileTransformer()
```

```
scaler.fit(X_train)

X_train = pd.DataFrame(data=scaler.transform(X_train),
columns = X_train.columns,
index=X_train.index)

X_test = pd.DataFrame(data=scaler.transform(X_test),
columns = X_test.columns,
index=X_test.index)
```

A.3 Random Forest

O algoritmo a seguir demonstra o desenvolvimento dos três modelos *RandomForest* para as janelas de previsão utilizadas.

Requer bibliotecas:

- ❏ *Numpy*
- ❏ *Pandas*
- ❏ *Scikit − learn*

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier()
rf2= RandomForestClassifier()
rf3= RandomForestClassifier()

rf.fit(X_train[features], y_train['Y_T1'])
rf2.fit(X_train[features],y_train['Y_T2'])
rf3.fit(X_train[features],y_train['Y_T3'])

predictionsRF= rf.predict(X_test[features])
predictionsRF2= rf2.predict(X_test[features])
predictionsRF3=rf3.predict(X_test[features])
```

A.4 SVM-GHI

Os algoritmos que utilizam o modelo SVM-GHI são apresentados a seguir.

Requer bibliotecas:

- *Numpy*
- *Pandas*
- *Scikit – learn*

```
from sklearn.svm import SVC
from pykernels.regular import GeneralizedHistogramIntersection

X= X_train[features].values
Y= y_train['Y_T1'].values

Y2= y_train['Y_T2'].values

Y3= y_train['Y_T3'].values

for clf13, name in [(SVC(kernel= GeneralizedHistogramIntersection(),C=10,probability=Tru
    clf13.fit(X, Y)

for clf13_2, name in [(SVC(kernel= GeneralizedHistogramIntersection(),C=10,probability=T
    clf13_2.fit(X, Y2)

for clf13_3, name in [(SVC(kernel= GeneralizedHistogramIntersection(),C=10,probability=T
    clf13_3.fit(X, Y3)
```

A.5 SVM-RBF

Algoritmo desenvolvido para realizar as previsões utilizando SVM-RBF.
Requer bibliotecas:

- *Numpy*
- *Pandas*
- *Scikit – learn*

```
from sklearn.svm import SVC

X= X_train[features].values
Y= y_train['Y_T1'].values

Y2= y_train['Y_T2'].values
```

```
7
8  Y3= y_train['Y_T3'].values
9
10
11 svr_rbf = SVC(kernel='rbf', C=100, probability=True)
12 svr_rbf2 = SVC(kernel='rbf', C=100, probability=True)
13 svr_rbf3 = SVC(kernel='rbf', C=100, probability=True)
14
15 svr_rbf.fit(X,Y)
16 svr_rbf2.fit(X,Y2)
17 svr_rbf3.fit(X,Y3)
```

A.6 RNA

O código a seguir foi utilizado para a aplicação da rede neural proposta. Requer bibliotecas:

- ❏ *Keras*
- ❏ *Tensorfloow*

```
1  import keras
2  from keras.models import Sequential
3  from keras.layers import Dense
4  from keras.layers import Dropout
5  import tensorflow as tf
6
7
8  model= Sequential()
9  model.add(Dense(units=12 ,input_dim=X_train[features].shape[1],
10 activation= 'relu'))
11
12 model.add(Dense(units=12 ,activation= 'relu'))
13
14
15 model.add(Dense(units=3,activation= 'relu'))
16
17 model.add(Dense(units=1,activation='relu'))
18
19
```

```
20
21 model2= Sequential()
22 model2.add(Dense(units=12 ,input_dim=X_train[features].shape[1],
23 activation= 'relu'))
24
25 model2.add(Dense(units=12 ,activation= 'relu'))
26
27
28 model2.add(Dense(units=3,activation= 'relu'))
29
30 model2.add(Dense(units=1,activation='relu'))
31
32
33 model3= Sequential()
34 model3.add(Dense(units=12 ,input_dim=X_train[features].shape[1],
35 activation= 'relu'))
36
37 model3.add(Dense(units=12 ,activation= 'relu'))
38
39
40 model3.add(Dense(units=3,activation= 'relu'))
41
42 model3.add(Dense(units=1,activation='relu'))
43
44
45 model.compile(optimizer = 'Adam',
46 loss='categorical_hinge', metrics =['accuracy'])
47
48
49 model2.compile(optimizer = 'Adam',
50 loss='categorical_hinge', metrics =['accuracy'])
51
52
53 model3.compile(optimizer = 'Adam',
54 loss='categorical_hinge', metrics =['accuracy'])
55
56
57 model.fit(X_train[features],y_train['Y_T1'],
58 epochs= 500,batch_size= 64,verbose=0)
```

```
59
60 model2.fit(X_train[features],y_train['Y_T2'],
61 epochs= 500,batch_size= 64,verbose=0)
62
63 model3.fit(X_train[features],y_train['Y_T3']
64 ,epochs= 500,batch_size= 64,verbose=0)
65
66 predictions= model.predict_proba(X_test[features])
67 predictions2= model2.predict_proba(X_test[features])
68 predictions3= model3.predict_proba(X_test[features])
69
70 predictions = pd.DataFrame(predictions)
71 predictions.columns= ['PRED']
72
73 predictions2 = pd.DataFrame(predictions2)
74 predictions2.columns= ['PRED']
75
76 predictions3 = pd.DataFrame(predictions3)
77 predictions3.columns= ['PRED']
78
79
80 predictions['PRED']  = np.where(predictions['PRED']>0.5, 1, 0)
81 predictions2['PRED']  = np.where(predictions2['PRED']>0.5, 1, 0)
82 predictions3['PRED']  = np.where(predictions3['PRED']>0.5, 1, 0)
```

A.7 Cálculo das Curvas ROC

Requer bibliotecas:

- ❏ *Numpy*
- ❏ *Pandas*
- ❏ *Scikit − learn*

```
1 from sklearn.metrics import plot_roc_curve
2 import matplotlib.pyplot as plt
3
4 ax = plt.gca()
5 plt.rcParams.update({'font.size': 14})
6 svm_rbf_disp = plot_roc_curve(rf, X_test[features],
```

```
y_test['Y_T1'], ax=ax,name= 'Random Forest')

rna_disp      = plt.plot(fpr,tpr)
svm2_rbf_disp = plot_roc_curve(clf13, XX, y_test['Y_T1'],
ax=ax, name='SVM - GHI')

svm3_rbf_disp = plot_roc_curve(svr_rbf, XX, y_test['Y_T1'],
ax=ax,name='SVM - RBF')

plt.ylabel('Taxa de Verdadeiros Positivos',fontsize=16)
plt.xlabel('',fontsize=20)
ax.axes.get_xaxis().set_ticks([])

plt.show()
```

A.8 Validação Cruzada utlizando $K - folds$

Requer bibliotecas:

- $Numpy$
- $Pandas$
- $Scikit - learn$

```
from sklearn.model_selection import cross_val_score

scores = cross_val_score(rf, X_train[features].values,
y_train['Y_T3'], cv=10)
print("Accuracy rf: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))

scores = cross_val_score(svr_rbf, X_train[features].values,
y_train['Y_T3'], cv=10)
print("Accuracy RBF: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))


scores = cross_val_score(clf13, X_train[features].values,
y_train['Y_T3'], cv=10)
print("Accuracy GHI: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
```