UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Uma abordagem com modelo de aprendizado de máquina híbrido para predição de movimentos de preço médio de ativos pelo livro de ofertas

Guilherme Augusto Bileki

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Guilherme Augusto Bileki

Uma abordagem com modelo de aprendizado de máquina híbrido para predição de movimentos de preço médio de ativos pelo livro de ofertas

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Vanderlei Bonato

USP – São Carlos
Maio de 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

B595a

Bileki, Guilherme Augusto
Uma abordagem com modelo de aprendizado de máquina híbrido para predição de movimentos de preço médio de ativos pelo livro de ofertas / Guilherme Augusto Bileki; orientador Vanderlei Bonato. -- São Carlos, 2021.
81 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2021.

1. APRENDIZADO COMPUTACIONAL. 2. MERCADO FINANCEIRO. 3. ANÁLISE DE SÉRIES TEMPORAIS. I. Bonato, Vanderlei, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Guilherme Augusto Bileki

A hybrid machine learning approach for inference of asset mid price movements from order book data

Master dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP, in partial fulfillment of the requirements for the degree of the Master Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Vanderlei Bonato

USP – São Carlos
May 2021

Este trabalho é dedicado àqueles que jamais o lerão.

AGRADECIMENTOS

Agradeço aos meus pais, principalmente à minha mãe Célia Firmino, pelo apoio em todos os momentos.

Gostaria de agradecer em especial ao Prof. Vanderlei Bonato pela sua orientação. Sua experiência e conhecimento foram de fundamental importância no decorrer deste trabalho.

À minha querida e amada Aline que ao meu lado dividiu minhas tristezas e multiplicou minhas alegrias.

Ao Prof. Eduardo do Valle Simões pelo apoio.

Aos meus colegas que ajudaram em diversas dúvidas durante o projeto: Precoce, Danilo, Romeu, Uirá, Zihao e Avraam.

Ao CNPq, pela bolsa de estudos pelo financiamento parcial deste trabalho de pesquisa (bolsa 132847/2017-5).

E a todos aqueles que de forma direta e indireta contribuíram para a execução deste projeto.

"Quero dizer agora o oposto do que eu disse antes
Eu prefiro ser essa metamorfose ambulante
Do que ter aquela velha opinião formada sobre tudo..."
(Raul Seixas)

RESUMO

BILEKI, G. A. **Uma abordagem com modelo de aprendizado de máquina híbrido para predição de movimentos de preço médio de ativos pelo livro de ofertas**. 2021. 81 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Neste trabalho é apresentada a aplicação de um modelo híbrido para prever os movimentos do preço-médio de instrumentos da Bolsa de Valores do Brasil (B3) utilizando dados do livro de ofertas e as mensagens relacionadas. Uma Rede Neural Convolucional (CNN) é utilizada para extrair características espaciais do livro de ofertas e um algoritmo baseado em árvores de decisão é utilizado para combinar as características da CNN com os dados do arquivo de mensagens. Diferente da maioria das outras bolsas de valores pelo mundo, o arquivo de mensagens da B3 inclui a qual corretora uma ordem pertence e neste trabalho também é apresentada uma análise de sua importância.Os resultados demonstram que a solução pode ser melhorada em 8% em termos de precisão (5% devido ao classificador baseado em árvore de decisão e mais 3% combinando com as mensagens) em comparação com uma CNN tradicional, onde as etapas de extração e classificação são ambas resolvidas pelo próprio modelo. Além disso, a utilização deste classificador permite a transferência de aprendizado de forma muito mais rápida do que o treinamento de uma CNN tradicional (cerca de 40 segundos).

Palavras-chave: Livro de Ofertas, Aprendizado de Máquina, Previsão de Tendência de Preço, Mercado de Ações Brasileiro.

ABSTRACT

BILEKI, G. A. **A hybrid machine learning approach for inference of asset mid price movements from order book data**. 2021. 81 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

This work presents the application of a hybrid model to predict the mid-price trend of assets in the Brazilian Stock Exchange (B3) using order book data and related messages. A Convolution Neural Network (CNN) is applied to extract spatial features from the order book and a decision tree based algorithm is used to combine the features from the CNN with features from the file of messages. Unlike most other stock exchanges around the world, the file of messages from B3 includes to which broker an order belongs to and in this work an analysis of its importance is presented as well. Our outlines demonstrate the solution can be improved in 8% in terms of accuracy (5% due to decision tree-based classifier and even more 3% combining with messages) compared to a traditional CNN, where extraction and classification steps are both solved by the model itself. In addition, the use of this classifier allows the transfer learning much faster than the training of a traditional CNN (about 40 seconds).

Keywords: Limit Order Book, Machine Learning, Price Trend Forecasting, Brazilian Stock Market.

LISTA DE ILUSTRAÇÕES

Figura 1 – Exemplo de MLP de 2 camadas. 29
Figura 2 – Exemplo de uma Rede Neural Convolucional comum. 30
Figura 3 – Exemplo de uma convolução. 30
Figura 4 – Exemplos de subamostragem. 31
Figura 5 – Exemplo de uma célula LSTM. 32
Figura 6 – Comparação entre os valores *Logloss* (quanto menor, melhor) para o modo de classificação. A porcentagem é a diferença métrica medida em relação aos resultados CatBoost ajustados. 34
Figura 7 – Exemplo da estrutura do *book*. 35
Figura 8 – Exemplo da chegada de ordem de compra com preço inferior ao *best bid*. . . 36
Figura 9 – Exemplo da chegada de ordem de compra com preço igual ao *best bid*. . . . 36
Figura 10 – Exemplo da chegada de ordem de compra com preço superior ao *best bid*. . 36
Figura 11 – Exemplo da chegada de ordem de compra com preço entre *best bid* e *best ask*, mas volume inferior ao *best ask*. 37
Figura 12 – Exemplo da chegada de ordem de compra com preço entre *best bid* e *best askid*, mas volume superior ao *best ask*, “ordem a mercado”. 37
Figura 13 – Exemplo da chegada de ordem de compra com preço limitado de 20.29. . . 37
Figura 14 – Exemplo da chegada de ordem de cancelamento de compra. 38
Figura 15 – Exemplo de estrutura do *order book*. 38
Figura 16 – Exemplo de estrutura do *message book*. 38
Figura 17 – Exemplo básico do funcionamento da biblioteca de SHAP. 40
Figura 18 – Tipos de ajuste do modelo baseado nas curvas de loss. 41
Figura 19 – Esquema do fluxo de trabalho, desde a aquisição até o teste do modelo. . . . 50
Figura 20 – Um exemplo de classificação de 1000 pontos de um dia do conjunto de dados FI-2010. 53
Figura 21 – Comparação entre FI-2010 and B3-2020 mostrando resultados de classificação, considerando o mesmos parâmetros α e k. 54
Figura 22 – Cenário em que k é variado, mantendo o mesmo α. 54
Figura 23 – Cenário em que α é variado, mantendo o mesmo k. 55
Figura 24 – Fluxo de treinamento para o modelo híbrido. 56
Figura 25 – Exemplo da divisão da base de dados FI-2010 para os experimentos com os 3 pares de modelos. 60

Figura 26 – Diagonal principal das matrizes de confusão dos pares de modelos para a base de dados FI-2010. 61
Figura 27 – Exemplo da divisão da base de dados B3-2020 para os experimentos com os 3 pares de modelos. 62
Figura 28 – Diagonal principal das matrizes de confusão do par CNN-MLP utilizando 12,5%, 25% e 50% e 100% dos filtros, respectivamente. 65
Figura 29 – O valor médio de SHAP para cada variável usado pelo classificador para cada uma das classes. 67
Figura 30 – O valor SHAP dos dados de entrada do classificador para cada uma das classes. 68
Figura 31 – O valor médio de SHAP para cada variável usada pelo classificador para cada uma das classes. 70
Figura 32 – O valor SHAP dos dados de entrada do classificador para cada uma das classes. 71
Figura 33 – Métricas obtidas para os modelos com variações de quantidade de dias, horizontes de previsão, filtros, classificadores e versões de extrator. 81

LISTA DE TABELAS

Tabela 1 – Descrição dos campos do arquivo *order book*. 39
Tabela 2 – Descrição dos campos do arquivo de mensagens. 39
Tabela 3 – Relação das principais características dos trabalhos relacionados. 48
Tabela 4 – Variáveis contidas no arquivo de mensagens associadas com a base de dados B3-2020. 51
Tabela 5 – Normalização de variáveis contidas no arquivo de mensagens associadas com a base de dados B3-2020. 53
Tabela 6 – Métricas obtidas pelas 2 versões de cada um dos 3 pares de modelos analisados no conjunto de teste da base de dados FI-2010. 60
Tabela 7 – Tempos de inferência para cada par de modelos testados. 62
Tabela 8 – Comparação da CNN-MLP-v1 variando o número de filtros utilizando o conjunto de treinamento BMF. 64
Tabela 9 – Comparação da CNN-MLP-v2 variando o número de filtros utilizando o conjunto de treinamento BMF. 64
Tabela 10 – Métricas para os modelos CNN-MLP-v1 e CNN-CatBoost-v1 com 25% dos filtros considerando as opções BMF-MSG0 e BMF-MSG1. 66
Tabela 11 – Métricas para os modelos CNN-MLP-v2 e CNN-CatBoost-v2 com 12,5% dos filtros considerando as opções BMF-MSG0 e BMF-MSG1. 66
Tabela 12 – Métricas para os modelos CNN-MLP-v2 e CNN-CatBoost-v2 com 25% dos filtros considerando as opções BMF-MSG0 e BMF-MSG1. 66
Tabela 13 – Comparação da CNN-MLP-v1 variando o número de filtros utilizando o conjunto de treinamento VISTA. 69
Tabela 14 – Comparação da CNN-MLP-v2 variando o número de filtros utilizando o conjunto de treinamento VISTA. 69
Tabela 15 – Métricas para os modelos CNN-MLP-v1 e CNN-CatBoost-v1 com 25% dos filtros considerando as opções BMF-MSG0 e BMF-MSG1. 70
Tabela 16 – Testes variados com intercâmbio de bases e modelos. 72

LISTA DE ABREVIATURAS E SIGLAS

AD	Árvores de Decisão
B3	Bolsa de Valores do Brasil
BMF	Bolsa de Mercadorias e Futuros
CNN	*Convolutional Neural Network*
FIR	*Finite Impulse Response*
FTP	*File Transfer Protocol*
GRU	*Gated Recurrent Unit*
HFT	*High-Frequency Trading*
LightGBM	*Light Gradient Boosting Machine*
LOB	*Limit Order Book*
LSE	*London Stock Exchange*
LSTM	*Long Short Term Memory*
MBP	*Market By Price*
ML	*Machine Learning*
MLP	*Multi-Layer Perceptron*
NAs	Neurônios Artificiais
OHLC	*Open-High-Low-Close*
RNAs	Redes Neurais Artificiais
RNN	*Recurrent Neural Network*
SVM	*Support Vector Machine*
TABL	*Temporal Attention-Augmented Bilinear*
XGBoost	*eXtreme Gradient Boosting*

SUMÁRIO

1 INTRODUÇÃO . . . 23
1.1 Justificativa . . . 25
1.2 Objetivos . . . 25
1.2.1 Objetivo Geral . . . *25*
1.2.2 Objetivos Específicos . . . *25*
1.3 Estrutura da Dissertação . . . 26

2 REFERENCIAL TEÓRICO . . . 27
2.1 Considerações Iniciais . . . 27
2.2 Algoritmos de Machine Learning . . . 27
2.2.1 Redes Neurais Artificiais . . . *27*
2.2.1.1 Tipos de Redes Neurais Artificiais . . . *28*
2.2.1.1.1 *Multi-Layer Perceptron* . . . 28
2.2.1.1.2 *Convolutional Neural Network* . . . 29
2.2.1.1.3 *Long Short Term Memory* . . . 31
2.2.1.1.4 Algoritmos de Aprendizagem . . . 32
2.2.2 Algoritmo Baseado em Árvores de Decisão . . . *33*
2.2.2.1 CatBoost . . . *33*
2.3 Formato de Dados de Oferta e Negociação do Mercado Financeiro 34
2.3.1 Limit Order Book . . . *34*
2.3.1.1 LOBSTER . . . *38*
2.4 Ferramentas . . . 39
2.4.1 Plataforma TensorFlow . . . *39*
2.4.2 Biblioteca SHAP . . . *39*
2.4.3 Métricas de Desempenho . . . *40*
2.5 Considerações Finais . . . 42

3 TRABALHOS RELACIONADOS . . . 43
3.1 Considerações Iniciais . . . 43
3.2 Revisão Literária . . . 43
3.3 Análise Crítica . . . 44
3.4 Considerações Finais . . . 48

4 MATERIAIS E MÉTODOS . . . 49

4.1 Considerações Iniciais 49
4.2 Fluxo do Trabalho 49
4.3 Bases de Dados 50
4.3.1 Pré-processamento do LOB *51*
4.3.2 Rotulação *52*
4.3.3 Pré-processamento das Mensagens *53*
4.3.4 Separação de Conjuntos de Dados *53*
4.4 Extrator de recursos com CNN 55
4.4.1 Modificação do modelo de Rede Neural para Híbrido *56*
4.5 Experimentos 56
4.6 Considerações Finais 58

5 RESULTADOS 59
5.1 Considerações Iniciais 59
5.2 Resultados para a Base de Dados FI-2010 59
5.3 Resultados para as Bases de Dados da B3 62
5.3.1 Resultados para um instrumento *63*
5.3.2 Resultados para Cinco Instrumentos *64*
5.3.2.1 Resultados para o Conjunto BMF *64*
5.3.2.1.1 Resultados para o Par CNN-CatBoost no Conjunto BMF 65
5.3.2.2 Resultados para o Conjunto VISTA *69*
5.3.2.2.1 Resultados para o Par CNN-CatBoost no Conjunto VISTA 69
5.3.3 Resultados com Intercâmbio de Bases de Dados *72*
5.4 Considerações Finais 73

6 CONCLUSÕES 75
6.1 Trabalhos Futuros 76

REFERÊNCIAS 77

APÊNDICE A TESTES COMPLETOS DE EXTRATORES 81

CAPÍTULO

1

INTRODUÇÃO

A modernização do mercado financeiro mudou drasticamente a forma como negociamos, tornando possível que a uma negociação seja realizada em um piscar de olhos (KAYA; SCHILDBACH; AG, 2016). Como resultado, tornou-se comum o uso de *High-Frequency Trading* (HFT), que é caracterizado pelo uso de algoritmos de alta velocidade para envio e cancelamento de ordens com o intuito de obter pequenos lucros. O HFT é operado sem intervenção humana e por isso é altamente dependente de tecnologias computacionais para fornecer latência muito baixa e alta velocidade de processamento, uma vez que a análise de dados de mercado e as execuções de pedidos são da ordem de nano a microssegundos (ALDRIDGE, 2009). Para mitigar problemas relacionados à comunicação, esses sistemas geralmente estão em *co-location*[1] com o servidor da bolsa de valores. As ordens enviadas pelos investidores são gerenciadas eletronicamente como um fluxo de mensagens pelo *Limit Order Book* (LOB), que é adotado pela maioria dos mercados financeiros no mundo (GOULD *et al.*, 2013). O LOB é "o último nível microscópico de descrição" (BOUCHAUD *et al.*, 2002) de um instrumento financeiro, e portanto o dado mais rico de informação.

Os algoritmos de HFT são compostos por estratégias que utilizam modelagens matemáticas que extraem informações do LOB para tomada de decisão, e portanto, retorno e risco devem ser considerados e a função destes algoritmos é minimizar os riscos e maximizar o retorno. Para lidar com dados financeiros, séries temporais normalmente são utilizadas em problemas de regressão, em que se tenta prever um valor específico, por exemplo, o preço de um determinado ativo. Entretanto, o LOB pode ser modelado por várias séries temporais, uma para cada nível de preço, fazendo com que a complexidade do problema se multiplique. Uma abordagem largamente empregada por autores que utilizam o LOB é a de tratar o problema como um problema de classificação ao invés de regressão, em que se tenta prever um dentre um número definido de possibilidades, chamados de classes. Dentre os autores que usam esta abordagem, destacam-se o

[1] Serviço em que o sistema de estratégia é colocado fisicamente nas próprias instalações da bolsa de valores, a fim de diminuir a latência na obtenção dos dados.

uso de algoritmos de *Machine Learning* (ML), em sua maioria Redes Neurais Artificiais (RNAs) (SIRIGNANO, 2016; TSANTEKIDIS *et al.*, 2017a; BOROVYKH; BOHTE; OOSTERLEE, 2017; DOERING; FAIRBANK; MARKOSE, 2017; DIXON, 2017). Estes trabalhos, em geral, não focam na estratégia de retorno financeiro, apenas em modelos para auxiliar na tomada de decisão, prevendo o comportamento do preço médio de um ativo em um espaço de segundos ou minutos a frente. Além disso, alguns destes autores se baseiam em uma mesma base de dados de *benchmark*, FI-2010, disponibilizada por Ntakaris *et al.* (2018) como forma de mostrar a evolução nas métricas de desempenho. Os autores Kercheval e Zhang (2015), Mäkinen *et al.* (2018), Zhang, Zohren e Roberts (2019) utilizam, além da FI-2010, outras bases de dados com maior volume e sugerem que o LOB tem características universais, pois conseguem utilizar um modelo treinado com dados de um grupo de ativos em outro.

Esta pesquisa utiliza os dados históricos de negociações diárias disponibilizados pela Bolsa de Valores do Brasil (B3) [2], que também contém informações sobre a qual corretora a ordem pertence. Esta informação de corretora não é comum na maioria dos mercados financeiros pelo mundo, pois para algumas bolsas a anonimização da informação da corretora incentiva à participação no mercado, melhorando a liquidez e aumentando a competitividade (POSKITT *et al.*, 2011). Explorar o uso dessa informação e outras obtidas no fluxo de mensagens é uma limitação para os modelos propostos por estes autores, pois estão fortemente atrelados ao tipo de entrada que utilizam, que na maioria dos trabalhos é a transformação da série temporal de todos os níveis do LOB em uma matriz bidimensional (pode-se considerar como uma imagem). Entretanto, este forte atrelamento limita o uso de outras informações, que se estivessem juntas nesta matriz poderiam perder a sua importância para o modelo, que foi inicialmente concebido para lidar com grandes variações de informações que representam os preços e volumes. Considerando a suposição de que os modelos propostos conseguem extrair informações universais do LOB, modificar a estrutura do modelo pode prejudicar esta extração. Além disso, segundo Cao *et al.* (2013), Luo *et al.* (2018), os modelos de RNAs apresentam vantagens no aprendizado de recursos em vários níveis, mas nem sempre são ótimas para classificação e que um modelo híbrido de CNN com *Random Forest* (algoritmo baseado em árvores de decisão) pode produzir melhores resultados.

Neste contexto, este trabalho apresenta um modelo híbrido para prever o movimento do preço médio de ativos utilizando, não só o LOB tradicional, mas também informações adicionais extraídas de um fluxo de mensagens, que contém o *log* de eventos que resultaram no estado atual do LOB. Na mesma linha de Tsantekidis *et al.* (2017a), Mäkinen *et al.* (2018), Zhang, Zohren e Roberts (2019), este trabalho utiliza três classes para prever se após um determinado tempo o preço médio irá subir, descer ou permanecer como está. O modelo híbrido proposto integra RNAs para extrair vetores de características do LOB com um algoritmo baseado em árvores de decisão para obter a classificação final. Esta estrutura é capaz de extrair informações universais

[2] ftp://ftp.bmf.com.br/

do LOB e combiná-la com as informações adicionais associadas às mensagens. Este é o primeiro trabalho que explora estas duas informações de maneira integrada por meio de modelos híbridos de ML.

1.1 Justificativa

Face a constante evolução da tecnologia e o grande aumento do volume de dados disponíveis, há uma demanda por técnicas que sejam mais eficientes e mais rápidas para lidar com o aumento da complexidade dos problemas. Dentre as diversas vertentes de exploração, muitos trabalhos focam na construção de modelos aplicados a classificação de comportamento do preço médio, por meio da filosofia de ML onde parâmetros da função de predição são obtidos a partir de algoritmos de aprendizagem.

1.2 Objetivos

1.2.1 Objetivo Geral

Este projeto tem como propósito desenvolver um modelo híbrido para predição do movimento do preço médio de uma ação utilizando informações não somente do LOB, mas também de eventos que refletem na composição do mesmo.

1.2.2 Objetivos Específicos

Este projeto de mestrado segue as seguintes etapas para alcançar o objetivo proposto:

1. Realizar uma revisão literária com relação ao uso de modelos de ML aplicados ao LOB e outros temas relacionados a esta dissertação;

2. Implementar modelos baseados nos trabalhos relacionados para definir base de comparação entre os dados de FI-2010 e da B3;

3. Verificar a performance dos modelos nos dados brasileiros para verificar se há características universais;

4. Propor melhorias de performance nos modelos base;

5. Propor e validar um modelo de ML híbrido;

6. Avaliar o impacto de informações adicionais ao LOB.

1.3 Estrutura da Dissertação

Esta dissertação possui a seguinte estrutura, além desta seção introdutória:

O Capítulo 2 apresenta os trabalhos relacionados, expondo a metodologia adotada para a busca e seleção dos estudos primários, uma análise crítica sobre cada trabalho e uma tabela comparativa entre os mesmos.

O Capítulo 3 apresenta um resumo geral do referencial teórico, com a descrição dos algoritmos de ML utilizados, a descrição das estruturas de dados e as ferramentas utilizadas para a implementação deste trabalho.

O Capítulo 4 apresenta a metodologia utilizada para geração dos experimentos, descrevendo os processos necessários para a execução dos mesmos.

O Capítulo 5 apresenta os resultados dos experimentos propostos, assim como uma análise e discussão dos mesmos.

Por fim, o Capítulo 6 apresenta as conclusões deste estudo e sugestões de trabalhos futuros.

CAPÍTULO

2

REFERENCIAL TEÓRICO

2.1 Considerações Iniciais

Neste capítulo serão apresentados os referenciais teóricos relacionados a este projeto de mestrado. Serão expostos ao longo das seções os conceitos relacionados aos algoritmos de *Machine Learning*, aos dados e as ferramentas utilizadas. Na Seção 2.2 são apresentados cada um dos algoritmos que compõem o modelo híbrido proposto. Na Seção 2.3 são apresentadas as estruturas de dados, suas propriedades e exemplos. A Seção 2.4 apresenta as ferramentas utilizadas para este projeto, descrevendo seu funcionamento e exemplos.

2.2 Algoritmos de Machine Learning

2.2.1 Redes Neurais Artificiais

As RNAs são algoritmos baseados no funcionamento de sistemas neurais biológicos e, em particular, o do cérebro humano (PERUŠ, 2001). Sua arquitetura é de um sistema paralelo distribuído, que conta com unidades de processamento simples, chamados Neurônios Artificiais (NAs), que calculam funções matemáticas. Os NAs são dispostos espacialmente por meio de camadas e interligados uns aos outros formando conexões ponderadas, que é onde boa parte do conhecimento do modelo está contido.

Com esta estrutura as RNAs podem ser treinadas por meio de exemplos para aprender a identificar padrões. Este treinamento visa determinar a intensidade das conexões entre os NAs por meio de algoritmos de aprendizado, que têm como finalidade ajustar os pesos para melhor generalização da informação exposta por meio dos exemplos. O aprendizado por meio de exemplos faz com que as RNAs sejam uma ferramenta útil para problemas de predição, pois ao contrário de modelos analíticos, são modelos que se adaptam aos dados, sem necessidade, a priori, de suposições sobre o problema estudado. Um das vantagens das RNAs é ter todo o fluxo

de extração de características e de classificação ou de regressão na própria estrutura da rede.

O problema deste modelo baseado unicamente em exemplos é que não há regras evidentes no sistema, o que pode levar ao fracasso do modelo. Entretanto, para muitos problemas do mundo real as próprias regras são desconhecidas. Para uma RNA desempenhar algum papel é necessário que esta passe por um processo de aprendizagem, que pode ser de forma supervisionada ou não-supervisionada. Na forma supervisionada, a RNA é alimentada com sinais de entrada e uma resposta esperada, e um algoritmo de aprendizagem ajusta os pesos por meio de minimização de erro entre a saída obtida e a esperada. Na forma não-supervisionada, a RNA é alimentada apenas com sinais de entrada e o algoritmo de aprendizagem deve agrupá-los baseado nos padrões capturados.

Para a modelagem de séries temporais, objetivo deste projeto, utiliza-se o aprendizado supervisionado treinado para classificação, em que os sinais de entrada são as variáveis da série temporal e a saída esperada é obtida de passos futuros da própria série. Ao longo desta seção serão apresentados os tipos de RNAs mais relevantes para a modelagem de séries temporais, além de funções de ativação, funções de custo, algoritmos de aprendizagem e tipos de normalizações de dados.

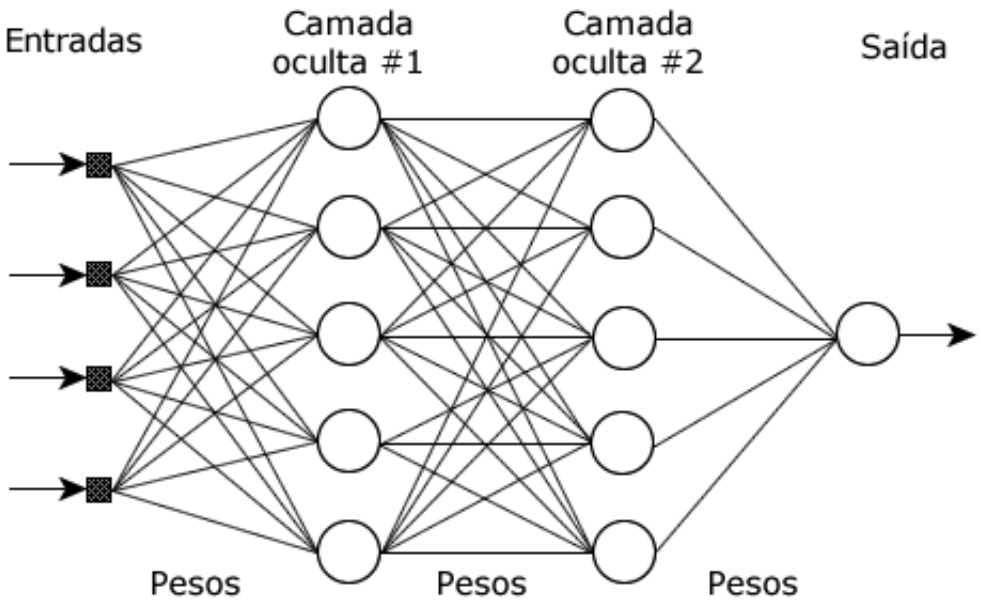
2.2.1.1 Tipos de Redes Neurais Artificiais

2.2.1.1.1 *Multi-Layer Perceptron*

A *Multi-Layer Perceptron* (MLP) (MCCLELLAND *et al.*, 1986) é um tipo de RNA composta por uma camada de entrada, uma de saída e por uma ou mais camadas intermediárias, também conhecidas como escondidas ou ocultas. O tipo de arquitetura da MLP é chamada de *FeedFoward*, em que os dados seguem um fluxo unidirecional, da entrada até a saída, por meio de uma estrutura em que a conexão é completa entre camadas adjacentes (a saída de um dado NA alimenta todos os NAs da camada seguinte). Um exemplo de arquitetura com quatro camadas é exibido na Figura 1.

O treinamento de uma MLP segue as seguintes etapas: inicialmente, apresenta-se os dados de treinamento à camada de entrada; os dados são então ponderados e somados em cada NA da primeira camada oculta; esta soma é transformada pela função de ativação de cada NA, gerando um sinal de saída do NA para a próxima camada; este processo é repetido até a camada de saída. Um algoritmo de aprendizagem é utilizado para calcular os pesos das conexões entre os NAs que minimizem o erro global da RNA, que é a diferença entre a saída obtida e a esperada.

Figura 1 – Exemplo de MLP de 2 camadas.



Fonte: Almeida (2014).

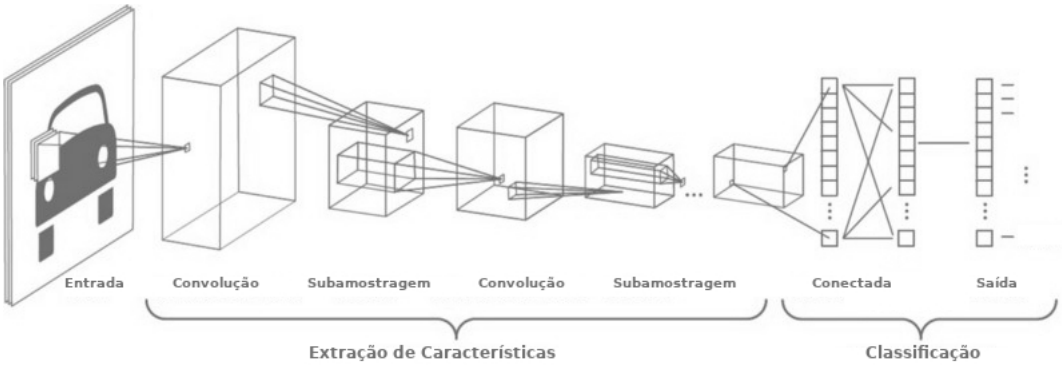
2.2.1.1.2 *Convolutional Neural Network*

A *Convolutional Neural Network* (CNN) (LECUN *et al.*, 1989) é um tipo de RNA baseada em MLP projetada para reconhecer padrões em matrizes (normalmente bidimensionais, como imagens), que foi inspirada no funcionamento da base do córtex visual de animais (HUBEL; WIESEL, 1962). A CNN tem como característica reconhecer padrões com alto grau de invariância à translação, dimensionamento, enviesamento e outras formas de distorção, para isso, sua estrutura inclui as seguintes formas de restrição: extração de características, mapeamento de características e subamostragem (HAYKIN *et al.*, 2009).

- A extração de características é realizada por cada NA, que recebe uma região da camada anterior, possibilitando a localização de padrões com características locais, não importando sua posição dentro de uma imagem.
- O mapeamento de características é realizado pela camada convolutiva, que faz com que os NAs compartilhem seus pesos, tendo como função lidar com variações de distorção, rotação e translação.
- A subamostragem é um processo de sumarização do mapa de características, realizado para diminuir a sensibilidade à distorções.

A Figura 2 mostra um exemplo típico de CNN com, composta por: camadas convolucionais, camadas de subamostragem e camadas de classificação. A função de cada uma dessas camadas é descrita a seguir:

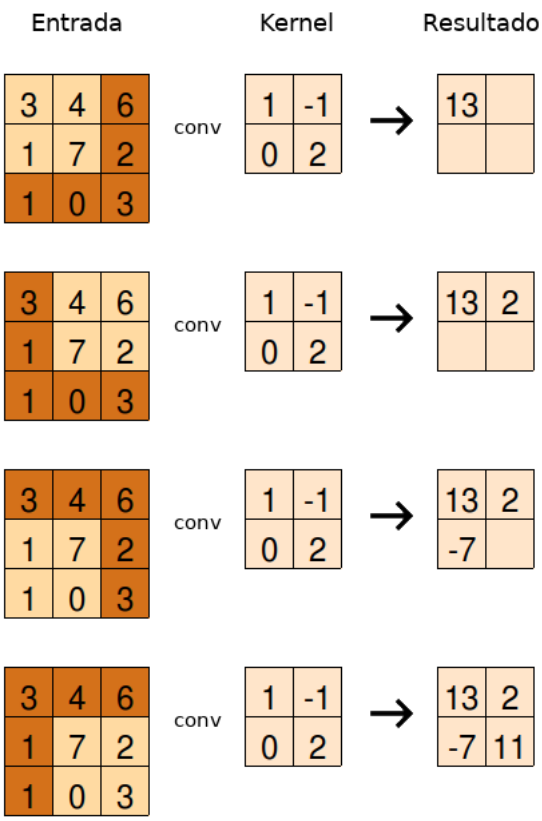
Figura 2 – Exemplo de uma Rede Neural Convolucional comum.



Fonte: MathWorks (2018).

Camadas Convolucionais: são camadas que aplicam a função de convolução entre os dados de entrada (*pixels*) e o *kernel* (pesos). No exemplo da Figura 3, este processo é mostrado em etapas, de cima para baixo, em que uma matriz de entrada (3x3) é convolucionada com um *kernel* (2x2), gerando uma nova matriz final (2x2) com os valores [13,2,-7,11].

Figura 3 – Exemplo de uma convolução.

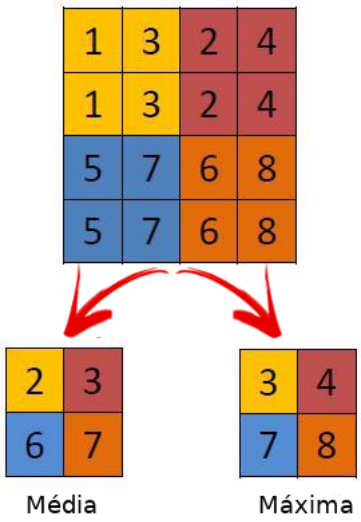


Fonte: Elaborada pelo autor.

Camadas de Subamostragem são camadas que resumem os valores de uma matriz, reduzindo

a complexidade das informações que a RNA deve aprender. Para isso, aplica funções de média ou máximo, como ilustrado na Figura 4.

Figura 4 – Exemplos de subamostragem.



Fonte: Adaptada de Pacheco (2017).

Camadas de Classificação são as camadas finais da CNN, em geral utiliza a mesma estrutura de uma MLP com poucas camadas, que nesse caso não é mais convolução.

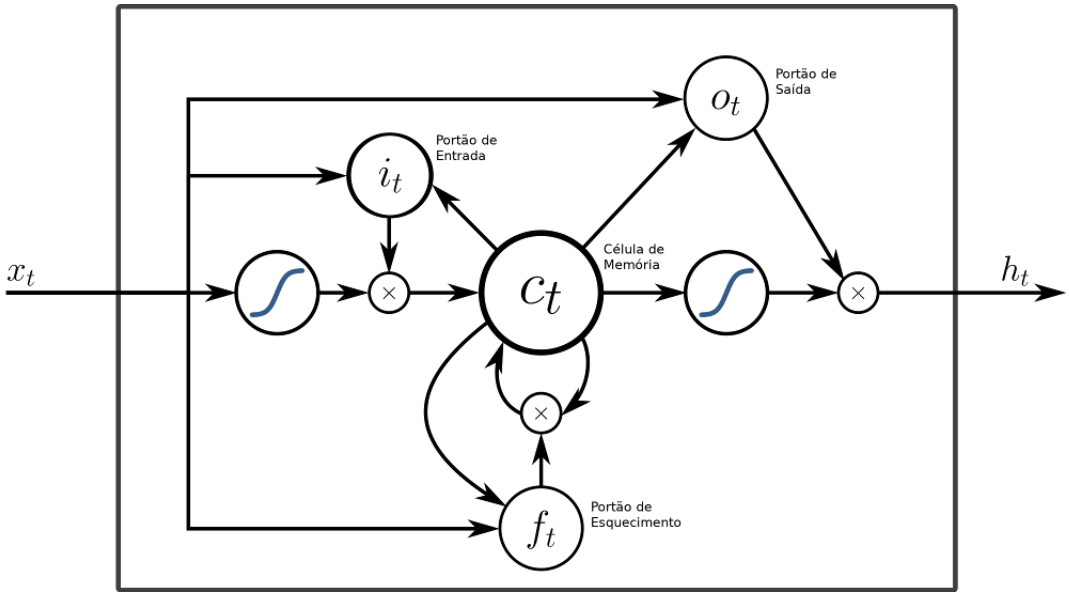
A estrutura e o processo de treinamento da CNN são parecidos com o de uma MLP: as matrizes de entrada (por exemplo, imagens) são apresentadas à primeira camada convolucional tendo-se a convolução entre o *kernel* e a matriz, gerando-se uma nova matriz; esses passarão por uma função de ativação e em seguida pela subamostragem (quando houver); este processo se repete até a chegada na camada de classificação. Um algoritmo de aprendizagem é utilizado para ajustar os pesos contidos nos *kernels* de maneira a minimizarem o erro global da CNN.

2.2.1.1.3 *Long Short Term Memory*

A *Long Short Term Memory* (LSTM) (HOCHREITER; SCHMIDHUBER, 1997) é um tipo de *Recurrent Neural Network* (RNN), que diferente da arquitetura *FeedFoward*, pode estabelecer conexões com camadas anteriores. Para as RNNs a informação não flui em um único sentido, gerando a função de "memória" nos NAs e devido a essa característica são comumente usadas para problemas com séries temporais. A LSTM foi proposta por Hochreiter e Schmidhuber (1997) para mitigar um problema relatado por Bengio, Simard e Frasconi (1994) por meio de experimentos, em que o erro durante treinamentos baseados em gradiente desaparece quando os dados tem dependências de longo termo (longas sequências). A proposta da LSTM considera cada NA como uma célula, em que foram adicionadas unidades especiais, os portões (ou *gates*), que permitem o ajuste de seus pesos para selecionar se uma informação deve ser

"memorizada" ou "esquecida". Um exemplo de célula LSTM está ilustrado na Figura 5, cujas variáveis x_t, i_t, o_t, f_t, c_t e h_t são respectivamente, o dado de entrada, o portão de entrada, o portão de saída, o portão de esquecimento, a célula de memória e o dado de saída. Em termos gerais, c_t é capaz de armazenar por um período de tempo um dado, que com o passar do tempo, caso não colabore com o treinamento da rede, pode ser esquecido e caso colabore, continua sendo armazenado.

Figura 5 – Exemplo de uma célula LSTM.



Fonte: Adaptada de Wikipedia (2018).

O treinamento de uma rede LSTM difere um pouco das anteriores pelo fato de sua recorrência: inicialmente o primeiro dado no tempo é apresentado à camada LSTM, que segue o fluxo entre as células da camada até a saída; em seguida, o próximo dado no tempo é apresentado a rede juntamente com os dados das outras células anteriores, e assim por diante, até o fim da série que se deseja treinar. Um algoritmo de aprendizagem é utilizado para calcular os pesos das variáveis de cada célula LSTM a fim de minimizar o erro global da rede.

2.2.1.1.4 Algoritmos de Aprendizagem

O processo de aprendizagem, ou treinamento de uma RNA, é um problema de minimização não-linear da diferença (erro) entre a saída produzida pela RNA e a saída esperada, que é assumida como verdadeira e obtida do conjunto de treinamento, com isso o treinamento visa controlar o erro pelo ajuste do pesos da RNA. O algoritmo mais utilizado é o *BackPropagation*, que basicamente é um método de gradiente descendente que distribui o erro a partir da saída chegando até a entrada do modelo para então aplicar o ajuste dos pesos conforme o gradiente.

2.2.2 Algoritmo Baseado em Árvores de Decisão

Árvores de Decisão (AD) são estruturas de dados que armazenam informações em elementos chamados "nós". Toda AD tem pelo menos um nó raiz, que é seu ponto de partida, e cada nó pode ter outros nós associados, chamados de "filhos". Um nó que não possui filhos é chamado de "nó folha". Em uma AD, uma decisão é tomada ao seguir um caminho do nó raiz a um nó folha. Um algoritmo baseado em AD tem como objetivo construir, uma AD com nós de decisão e nós folha a partir de técnicas de divisão e poda de nós. Na literatura, há diversos algoritmos que otimizam a criação de ADs, por exemplo *Random Forest* e *Gradient Boosting*. O *Gradient Boosting* tem como objetivo criar diversos modelos fracos, onde cada um tenta minimizar o erro do modelo anterior, por meio de uma função de perda. Para cada modelo fraco é multiplicado um valor chamado de taxa de aprendizagem aos ajustes obtidos, a fim de determinar o impacto de cada árvore no modelo final, em que quanto maior for o valor, maior a contribuição de cada árvore.

2.2.2.1 CatBoost

Nos últimos anos, foram propostas diversas variantes dos algoritmos *Gradient Boosting*, como o *eXtreme Gradient Boosting* (XGBoost), *Light Gradient Boosting Machine* (LightGBM) e o *CatBoost*. Dentre eles, o *CatBoost* tem menos parâmetros e permite treinamento e teste mais rápidos com maior precisão. O *CatBoost* usa o algoritmo *Ordered Boosting*, que melhora a capacidade de generalização (PROKHORENKOVA *et al.*, 2018). Conforme demonstrado em Jhaveri *et al.* (2019), Dorogush, Ershov e Gulin (2018), seus resultados são superiores ao *Random Forest* e outros algoritmos de *Gradient Boosting* quando comparado em treinamentos com bases de dados comuns de competições. A Figura 6 mostra um exemplo de comparação entre os resultados de treinamento do *CatBoost* e outros algoritmos baseados em *Gradient Boosting* sobre diversas bases de dados conhecidas para *benchmarks*.

Figura 6 – Comparação entre os valores *Logloss* (quanto menor, melhor) para o modo de classificação. A porcentagem é a diferença métrica medida em relação aos resultados CatBoost ajustados.

	CatBoost	LightGBM	XGBoost	H2O
	Padrão	Padrão	Padrão	Padrão
Adult	**0.27298**	0.28716 +5.20%	0.28009 +2.61%	0.27607 +1.14%
Amazon	**0.13811**	0.16716 +21.04%	0.16536 +19.74%	0.16950 +22.73%
Click prediction	**0.39112**	0.39749 +1.63%	0.39764 +1.67%	0.39785 +1.73%
KDD appetency	**0.07138**	0.07482 +4.82%	0.07466 +4.60%	0.07355 +3.05%
KDD churn	**0.23193**	0.23565 +1.61%	0.23369 +0.76%	0.23287 +0.41%
KDD internet	**0.22021**	0.23627 +7.30%	0.23468 +6.58%	0.24023 +9.10%
KDD upselling	**0.16674**	0.17107 +2.60%	0.16873 +1.20%	0.16981 +1.85%
KDD 98	**0.19479**	0.19837 +1.84%	0.19795 +1.63%	0.19607 +0.66%
Kick prediction	**0.28491**	0.29877 +4.87%	0.29816 +4.66%	0.29635 +4.02%

Fonte: Adaptada de Yandex (2020).

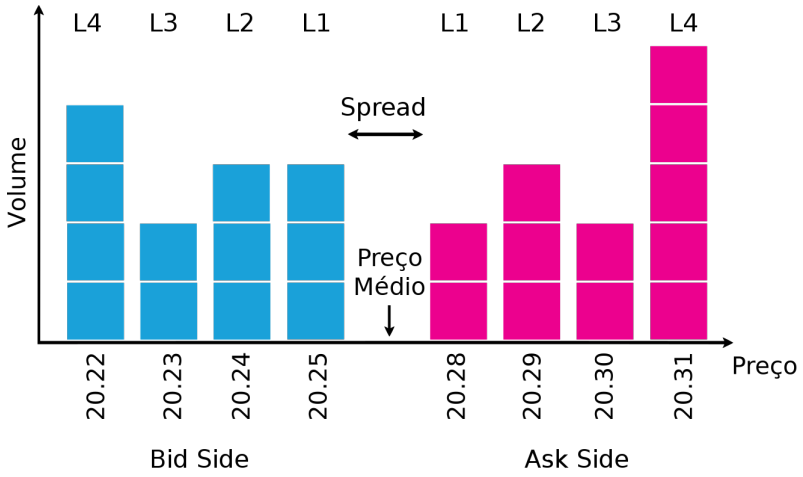
2.3 Formato de Dados de Oferta e Negociação do Mercado Financeiro

2.3.1 Limit Order Book

A maioria dos mercados financeiros utilizam sistemas com LOBs para tratar suas negociações. Nesses sistemas são ofertadas ordens de compra e de venda em tempo real, que podem ser do tipo “ordem a mercado” (OM) ou “ordem limitada” (OL). A OM não tem preço definido e é executada imediatamente sempre que houver ofertas disponíveis. O LOB é formado pelas OLs e é dividido em lado da compra (*bid side*) e lado da venda (*ask side*). Neste estudo o LOB adotado

agrega as ofertas por níveis de preço de maneira que não há discriminação de prioridade por tempo, também conhecido como *Market By Price* (MBP). O grupo com o melhor preço de venda ocupa a posição de nível 1 (L1) do lado da venda (*best ask*) e o grupo com o melhor preço de compra ocupa o nível 1 (L1) do lado da compra (*best bid*). A diferença entre ambos é chamada de *spread* e a média entre os dois é chamada de preço médio. A quantidade de ofertas agregadas em um determinado preço é chamada de volume. Esta estrutura básica do LOB é ilustrada em Figura 7.

Figura 7 – Exemplo da estrutura do *book*.



Fonte: Elaborada pelo autor.

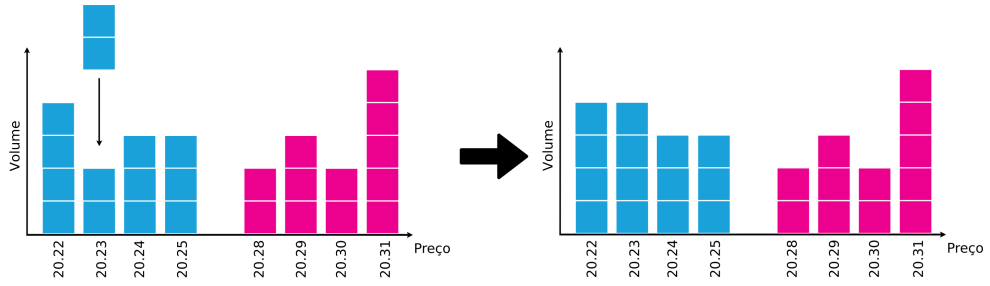
A seguir, apresenta-se um exemplo de interação entre uma OL de compra e um LOB pré-existente que contém inclusive prioridade por ordem de chegada. Esse exemplo é baseado em (CASTRO, 2017) em que considera a ordem sendo submetida ao mercado financeiro:

- Caso o preço limite seja menor ou igual ao *best bid*, a ordem não altera nem o *best bid* nem o *best ask* (Figura 8) e caso já exista aquele valor, a ordem é agrupada ao *best bid* e sua prioridade é menor por ter chegado depois (Figura 9).
- Caso o preço seja maior que o *best bid*, mas menor que o *best ask*, a ordem aumenta o preço do *best bid* e sua ordem de prioridade será maior para aquele nível, enquanto o volume for menor ou igual que o inserido inicialmente (Figura 10).
- Caso o preço seja maior ou igual ao *best ask*, então a ordem é executada imediatamente caso haja ordens disponíveis no *ask side*. Quando isso ocorre, a ordem é combinada com uma ordem do *ask side* (Figura 11). Caso o volume não seja totalmente satisfeito é combinada com a próxima ordem, que atenda as restrições de preço até ser completamente executada (Figura 12), ou consumir todas as ofertas de *ask side* até o limite do preço estipulado (Figura 13). Ao fim deste processo são atualizados os valores de *best bid* e *best ask*.

- Caso uma ordem seja cancelada, ela é removida da fila de preços (Figura 14).

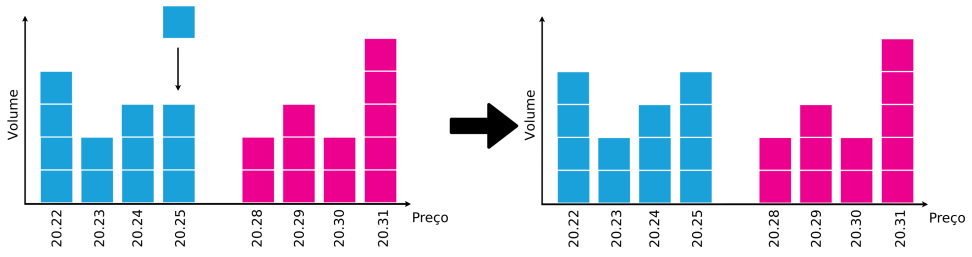
Todo esse processo pode ser armazenado em formato de *log* e disponibilizado aos usuários do mercado por meio de arquivos de mensagens.

Figura 8 – Exemplo da chegada de ordem de compra com preço inferior ao *best bid*.



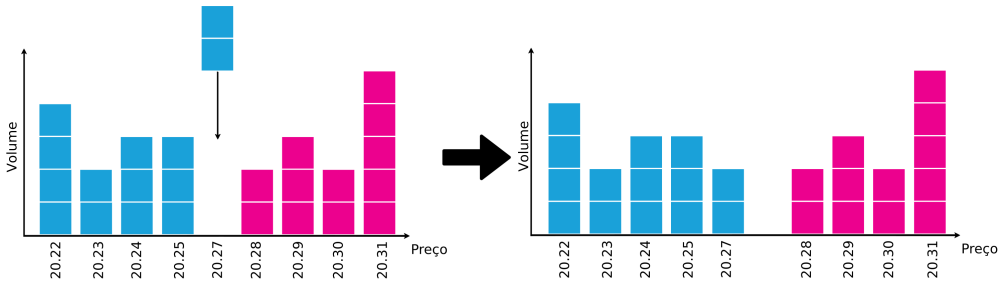
Fonte: Elaborada pelo autor.

Figura 9 – Exemplo da chegada de ordem de compra com preço igual ao *best bid*.



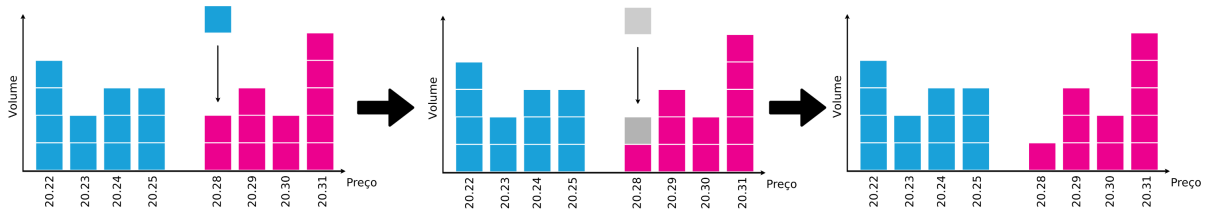
Fonte: Elaborada pelo autor.

Figura 10 – Exemplo da chegada de ordem de compra com preço superior ao *best bid*.



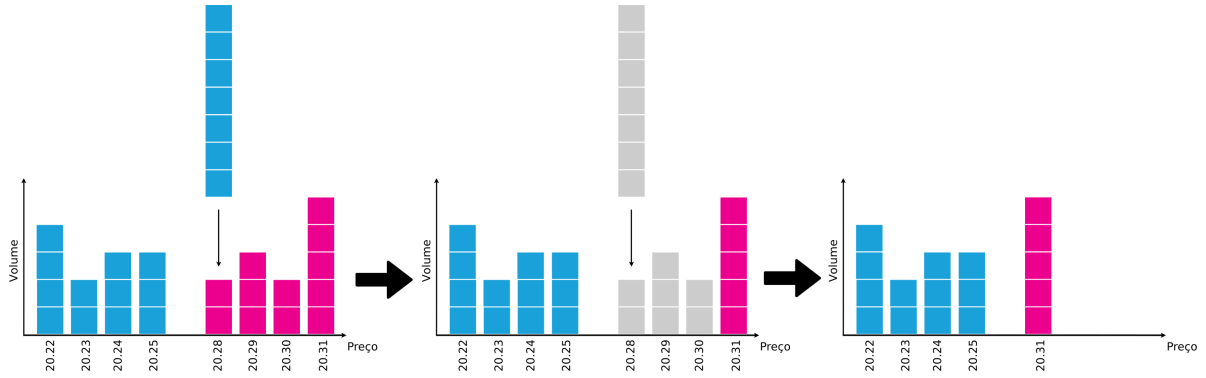
Fonte: Elaborada pelo autor.

Figura 11 – Exemplo da chegada de ordem de compra com preço entre *best bid* e *best ask*, mas volume inferior ao *best ask*.



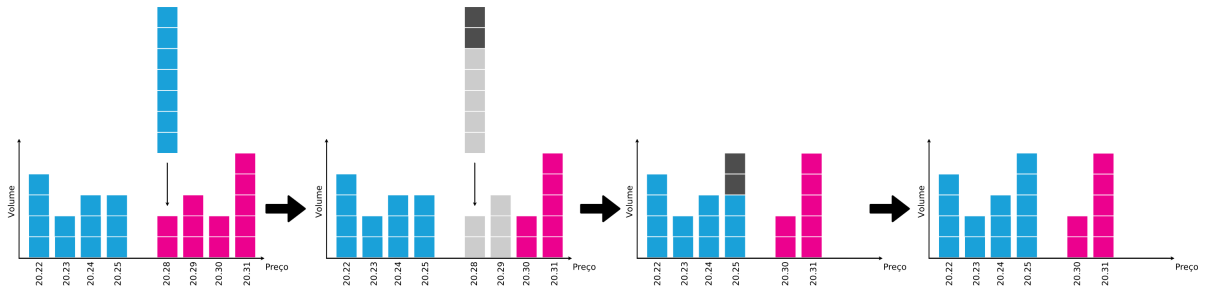
Fonte: Elaborada pelo autor.

Figura 12 – Exemplo da chegada de ordem de compra com preço entre *best bid* e *best askid*, mas volume superior ao *best ask*, "ordem a mercado".



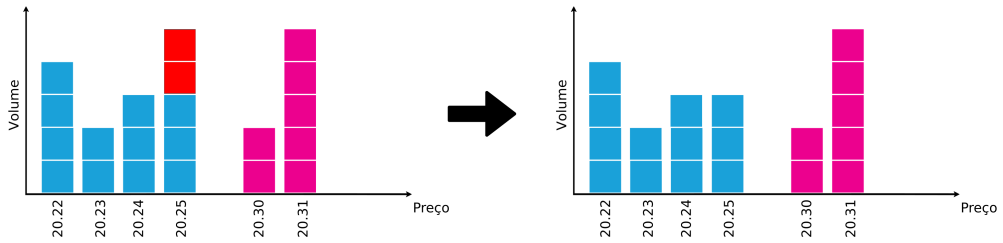
Fonte: Elaborada pelo autor.

Figura 13 – Exemplo da chegada de ordem de compra com preço limitado de 20.29.



Fonte: Elaborada pelo autor.

Figura 14 – Exemplo da chegada de ordem de cancelamento de compra.



Fonte: Elaborada pelo autor.

2.3.1.1 LOBSTER

O LOBSTER[1] é uma ferramenta *online* que provê para os usuários dados reconstruídos do LOB para todo o universo de ações negociadas da NASDAQ. Essa reconstrução é feita a partir de *logs* de eventos, em que um sistema de combinação de ordens estrutura os dados reconstruídos em dois arquivos:: o *order book* e o *message book*, conforme visto nas Figura 15 e Figura 16 (HUANG; POLAK, 2011).

Figura 15 – Exemplo de estrutura do *order book*.

Ask Price 1	Ask Size 1	Bid Price 1	Bid Size 1	Ask Price 2	Ask Size 2	Bid Price 2	Bid Size 2	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1186600	9484	118500	8800	118700	22700	118400	14930	...
1186600	9384	118500	8800	118700	22700	118400	14930	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Fonte: Adaptada de LOBSTER (2020).

Figura 16 – Exemplo de estrutura do *message book*.

Time (sec)	Event Type	Order ID	Size	Price	Direction
⋮	⋮	⋮	⋮	⋮	⋮
34713.685155243	1	206833312	100	118600	-1
34714.133632201	3	206833312	100	118600	-1
⋮	⋮	⋮	⋮	⋮	⋮

Fonte: Adaptada de LOBSTER (2020).

O arquivo *order book* do LOBSTER contem os seguinte campos:

[1] https://lobsterdata.com/

Tabela 1 – Descrição dos campos do arquivo *order book*.

Campo	Descrição
Ask Price 1	preço de venda de nível 1 (melhor preço de venda)
Ask Size 1	volume de venda de nível 1 (melhor volume de venda)
Bid Price 1	preço de compra de nível 1 (melhor preço de compra)
Bid Size 1	volume de compra de nível 1 (melhor volume de compra)
Ask Price 2	preço de venda de nível 2 (segundo melhor preço de venda)
Ask Size 2	volume de venda de nível 2 (segundo melhor volume de venda)

Enquanto que o arquivo mensagens contem esses outros campos:

Tabela 2 – Descrição dos campos do arquivo de mensagens.

Campo	Descrição
Time	Segundos após a meia-noite com precisão decimal de pelo menos milissegundos e até nanossegundos.
Event Type	Tipo de evento (inclusão, cancelamento, exclusão...).
Order ID	Número exclusivo de referência do pedido.
Size	Volume do pedido.
Price	Preço em dólar x10000.
Direction	Direção (compra ou venda).

2.4 Ferramentas

2.4.1 Plataforma TensorFlow

O *TensorFlow* é uma biblioteca de software de código aberto para computação numérica usando grafos de fluxo de dados[2], desenvolvido pela Google. Pode ser utilizado em diferentes linguagens e dispositivos, com um alto nível de abstração que torna capaz a implementação e utilização de uma rede neural complexa com poucas linhas de código.

A arquitetura do *TensorFlow* é baseada em grafos de fluxo de dados - arquitetura que explora o paralelismo em sua forma natural. Os nós são estruturas de computação matemática e os dados que trafegam pelo grafo e são computados nos nós são chamados de tensores, que são tipos de dados multi-dimensionais e podem representar constantes ou variáveis.

2.4.2 Biblioteca SHAP

Modelos complexos de predição, como *Random Forest*, *Gradient Boosting Trees* e RNA, são tidos como algoritmos caixa preta, pois sua complexidade é alta e de difícil análise estrutural devido suas grandes quantidades de parâmetros. A biblioteca SHAP (**SH**apley **A**dditive

[2] https://www.tensorflow.org/

ex**P**lanations)[3] (LUNDBERG; LEE, 2017) tem como objetivo explicar a relação entre as variáveis de entrada e o resultado do modelo. Essa técnica é baseada na teoria dos jogos e usa o valor de Shapley (ROTH, 1988) para definir o impacto de uma variável na saída do modelo. O valor de Shapley é calculado comparando o que o modelo prevê com e sem a variável analisada em todas as ordens possíveis.

A Figura 17 mostra um exemplo básico do funcionamento da biblioteca de SHAP. O lado esquerdo contém valores das variáveis Idade, Sexo, Peso Corporal (PC) e índice de Massa Corpórea (IMC) aplicados na entrada de um modelo caixa preta resultando em um valor de saída. O lado direito explica as variáveis por meio de barras, em que o valor é o valor calculado de SHAP para aquela variável e a cor azul representa um impacto negativo para o valor da saída, enquanto a cor rosa representa um impacto positivo.

Figura 17 – Exemplo básico do funcionamento da biblioteca de SHAP.

Fonte: Adaptada de Lundberg e Lee (2020).

2.4.3 Métricas de Desempenho

A avaliação de desempenho é um passo fundamental para a melhoria de sistemas computacionais, sejam processadores, linguagens de programação, sistemas operacionais, arquiteturas de rede ou sistema de banco de dados (BUKH, 1992). Das diferentes métricas para a avaliação de desempenho de sistemas computacionais, as mais utilizadas ao trabalhar com RNAs são:

***Loss*:** indica o aprendizado do modelo, em que é calculado o erro do modelo, quanto mais perto de 0, menor o erro geral do modelo.

Acurácia: indica o desempenho geral do modelo, em que é calculada a relação entre a quantidade de classificações corretas e a quantidade total de classificações realizadas pelo modelo.

Precisão: indica o quão preciso é o modelo, em que é calculada a relação entre as classificações corretas e as classificações positivas para uma determinada classe.

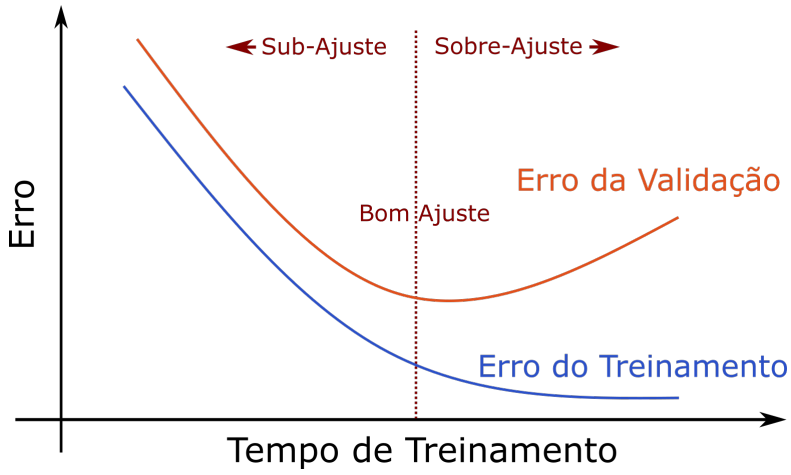
[3] https://github.com/slundberg/shap

***Recall*:** indica o quão sensível é o modelo, em que é calculada a relação das classificações que deveriam ser positivas, de quantas estão corretas.

***F1-Score*:** é a média harmônica da precisão e *recall*.

Estas métricas podem ser utilizadas em todos os conjuntos de dados durante o treinamento de um modelo. Quando utilizadas nos conjuntos de treinamento e validação é possível avaliar o ajuste do modelo, que indica se ele está ou não generalizando os dados apresentados durante o processo de treinamento. Esta avaliação permite decidir se é viável continuar ou não o treinamento e pode indicar sugestões de alterações para que ele consiga generalizar o problema. Como mostra a Figura 18 é possível identificar 3 comportamentos com a análise gráfica da função de perda (*loss*) dos dados de treinamento e validação.

Figura 18 – Tipos de ajuste do modelo baseado nas curvas de loss.



Fonte: Elaborada pelo autor.

Sub-ajuste: indica que o modelo não tem capacidade suficiente para generalizar o problema baseado nos dados, isto normalmente sugere que o modelo é muito simples ou que é necessário um conjunto de dados com maior representatividade do problema, tornando o modelo pouco eficiente.

Bom Ajuste: indica que o modelo é capaz de aprender o problema, isto normalmente sugere que os o modelo tem complexidade suficiente para a representatividade dos dados utilizados e que pode generalizar para dados desconhecidos.

Sobre-ajuste: indica que o modelo é demasiado robusto para o problema e se ajusta aos dados de treinamento, isto normalmente sugere que o modelo é muito complexo ou que os dados são muito simples para representar o problema, tornando o modelo eficiente apenas para os dados que aprendeu, não conseguindo generalizar para dados desconhecidos.

2.5 Considerações Finais

Neste capítulo foram apresentados os referenciais teóricos relacionados a este projeto de mestrado. Nas seções apresentadas foram descritos os conceitos relacionados aos algoritmos de *Machine Learning*, a estrutura dos dados utilizados no mercado financeiro baseado em LOB e as ferramentas utilizadas para construção, treinamento e avaliação de modelos.

CAPÍTULO

3

TRABALHOS RELACIONADOS

3.1 Considerações Iniciais

Neste capítulo serão apresentados os trabalhos relacionados a este projeto de mestrado. Serão expostos o modo de seleção dos trabalhos e uma análise crítica dos trabalhos relacionados e uma tabela comparativa entre eles. Na Seção 3.2 são apresentados os critérios de inclusão e exclusão dos trabalhos relacionados. Na Seção 3.3 são apresentadas as análises de cada trabalho relacionado e como colaboram com este projeto de mestrado, além de uma tabela comparativa entre eles.

3.2 Revisão Literária

Para este projeto, optou-se por uma revisão narrativa, a qual não utiliza critérios explícitos e sistemáticos para a revisão da literatura, flexibilizando assim a escolha das *strings* de busca utilizadas pelos autores. O levantamento bibliográfico orientou-se por meio das seguintes bases de dados: *IEEE Xplore*[1] e *Scopus*[2]. A *string* de busca foi definida para englobar os dois assuntos mais importantes deste projeto "*limit order book*" e "*neural network*" de 2015 a 2020.

A partir da leitura dos trabalhos relacionados, foram definidos critérios de seleção para a escolha dos mais relevantes. A seguir são apresentados os critérios para a seleção dos trabalhos mais relevantes.

Seleção de Estudos Primários

Para avaliação dos estudos primários retornados pela *string* de busca, critérios de inclusão e exclusão de trabalhos foram definidos. O processo de seleção de estudos adotado nesta revisão

[1] http://ieeexplore.ieee.org/
[2] https://www.scopus.com

da literatura consistiu na definição de três fases de avaliação, que são as seguintes: leitura de títulos e resumos, leitura da introdução e conclusão e leitura completa dos trabalhos. A seguir são descritos os critérios adotados.

A. Critérios de Inclusão

CI 1. Estudos primários relacionados aos tópicos de pesquisa;

CI 2. Estudos primários publicados em periódicos e conferências e;

CI 3. Estudos primários que possuam as palavras reservadas da *string* de busca ao menos no título, resumo ou palavras chave.

B. Critérios de Exclusão

CE 1. Resultados que não correspondem a publicações em conferências e periódicos e;

CE 2. Estudos primários que não sejam relacionados as áreas de Engenharia/Ciência da Computação.

3.3 Análise Crítica

Kercheval e Zhang (2015) utiliza *Support Vector Machine* (SVM) em um conjunto de dados da NASDAQ de um dia completo de cinco ações. O objetivo da SVM é classificar os dados de entrada entre 3 classes: subida, descida ou estacionária. Este conjunto de dados está estruturado no formato do *LOBSTER*, contendo o LOB e as mensagens. Com isso, o autor propõe a divisão dos dados em 3 subconjuntos: o primeiro com apenas os dados do LOB; o segundo extrai relações entre os valores de preços e volumes do LOB, sem levar o tempo em consideração; e o terceiro extrai relações temporais entre valores de preços e volumes do LOB. O treinamento da SVM utiliza apenas 2000 pontos e como métricas de desempenho também foram adotadas: precisão, *recall* e *F1-score*. O autor não compara seu modelo a outros trabalhos, entretanto demonstra ter alcançado 80% de *F1-score*.

Sirignano (2016) analisa um grande volume de dados, composto por 489 ações da NASDAQ entre 2014 e 2015, obtidas pelo LOBSTER. O autor propõe uma nova arquitetura de RNA, a *Spartial Neural Network*, que é utilizada para modelar a distribuição conjunta do estado do LOB para prever o movimento do preço na próxima mudança ou o movimento no próximo 1 segundo. Como métricas, são utilizadas: distribuição marginal e precisão. O autor compara a *Spartial Neural Network* à Regressão Logística, *Naive Empirical Model* e uma MLP, superando todas elas, atingindo uma precisão de quase 71%.

Persio e Honchar (2017) compara RNN multi-camadas, LSTM e *Gated Recurrent Unit* (GRU) utilizando dados de velas (*Open-High-Low-Close* (OHLC)[3]) diários da ação *GOOGL* de 2012 a 2016. O objetivo dos modelos é prever o movimento do preço da ação nos próximos, 1, 5, 10 ou mais dias, utilizando para LSTM e GRU camadas de *Dropout* em alguns experimentos. Os resultados obtidos demonstram que a LSTM e a GRU tem resultados muito semelhantes, por volta de 67% de acurácia, mas com a adição do *Dropout*, a LSTM consegue superar em cerca de 1,5% o resultado da GRU. Outro ponto de destaque, é o tempo de treinamento de cada modelo, em que a GRU é cerca de 5x mais lenta que a RNN e cerca de 10% mais lenta que a LSTM.

Doering, Fairbank e Markose (2017) utiliza um conjunto de dados da *London Stock Exchange* (LSE) de 2007 a 2008 de uma ação, com dados do LOB e de mensagens para prever a tendência da volatilidade e do preço da próxima ordem do LOB. Este conjunto de dados é separado em 3 subconjuntos: o primeiro apenas contendo dados do LOB; o segundo com dados apenas da mensagem contendo negócios ocorridos recentemente, entrada de ordens de compra e venda e exclusões de ordens transmitidas; e o terceiro que é uma composição dos dois anteriores. Com isso o autor treina 6 modelos utilizando a mesma CNN baseada na *GoogleNet* e na *CaffeNet*, um em cada subconjunto de dados e para cada uma das duas previsões possíveis. Os resultados obtidos demonstram que o terceiro subconjunto tem os melhores resultados para as duas previsões, em relação aos outros subconjuntos. Entretanto, a previsão da tendência de volatilidade tem uma acurácia maior do que a previsão da tendência do preço, chegando a cerca de 67%.

Tsantekidis *et al.* (2017a), Tsantekidis *et al.* (2017b) propõe modelar o problema de previsão do movimento do preço médio como um problema de classificação de 3 classes - subida, descida e estacionária - introduzindo a utilização do conjunto de dados público FI-2010 (NTAKARIS *et al.*, 2018). FI-2010 contempla 10 dias consecutivos de dados sobre 5 ações da bolsa de valores finlandesa. Tsantekidis *et al.* (2017a) usa um CNN de 4 camadas convolucionais, enquanto Tsantekidis *et al.* (2017b) usa um LSTM de 1 camada, ambos usando uma MLP como camada de classificação. Os dois trabalhos utilizam 3 horizontes de previsão[4] do FI-2010 (10, 20, 50) e propõem duas configurações de treinamento: a primeira utilizando validação cruzada dos 10 dias de dados, em que o conjunto de teste é o último dia do conjunto e o de treinamento, os dias anteriores; a segunda separando o conjunto de treinamento como os primeiros 7 dias e o de teste como os 3 últimos dias. Os resultados dos dois estudos foram comparados com um modelo SVM e uma MLP de 1 camada, usando métricas de desempenho: precisão, *recall*, *F1-score* e *Cohen's kappa*. Ambos os modelos superaram o SVM e a MLP, entretanto a LSTM com melhores resultados do que o CNN, melhorando 5% em todas as métricas, cujo maior valor de *F1-score* foi cerca de 76%.

[3] OHLC é um tipo de dado financeiro, normalmente chamado de vela, que agrega informações de preço de um certo ativo em períodos de tempo definidos, como: 1 segundo, 1 minuto, 15 minutos, etc.

[4] O horizonte de previsão é um valor da janela utilizada para gerar a classificação dos dados de entrada. Mais detalhes na Subseção 4.3.1.

Tran *et al.* (2018) propõe uma nova arquitetura de camada de RNA, denominada *Temporal Attention-Augmented Bilinear* (TABL), com o intuito de superar o estado da arte em relação a performance com menor necessidade de cálculos. Utilizando o FI-2010, o autor utiliza 3 modelos diferentes baseados na TABL, em que se diferem na quantidade de camadas utilizadas. Os resultado dos 3 modelos são comparados aos trabalhos de Tsantekidis *et al.* (2017a), Tsantekidis *et al.* (2017b), em que o maior dos 3 supera todos os outros para todos os horizontes de previsão e para as duas configurações de treinamento utilizados por Tsantekidis *et al.* (2017a), Tsantekidis *et al.* (2017b), chegando a cerca de 79% de *F1-score*.

Sirignano e Cont (2019) utiliza uma LSTM e adiciona 500 ações em relação ao seu trabalho anterior (SIRIGNANO, 2016), propondo com base em seus testes que há características universais no LOB. O autor faz experimentos com treinamentos em curtos e longos períodos em seu modelo, além de que adicionar as mensagens. Como resultado o autor conclui que seu modelo tem: universalidade, demonstrada pelo melhor resultado quando treinado em todas as ações em relação ao treinamento em uma específica; estacionaridade, demonstrada pela estabilidade durante o tempo de 1 ano de treinamento; evidencia de "longa memória" na formação do preço, demonstrada pela adição das mensagens, que melhora o desempenho da previsão; e generalização, demonstrada por performar bem, mesmo em ações não utilizados no treinamento.

Mäkinen *et al.* (2019) propõe um novo modelo baseado nos trabalhos de Tsantekidis *et al.* (2017a), Tsantekidis *et al.* (2017b) unindo CNN com LSTM para extração de recursos e MLP como classificador, além de uma nova camada de atenção à CNN. O autor utiliza apenas as classes de subida e descida, pois tenta prever os saltos dos preços do LOB e utiliza os dados de 5 ações da NASDAQ do período de 1 ano. Para o treinamento, o autor utiliza 7 subconjuntos de dados, separando para o conjunto de teste os 10 últimos dias, e para treinamento os dias anteriores, semelhante a configuração 1 de Tsantekidis *et al.* (2017a). Devido a diferença de classes e de dados, o autor não compara seu modelo a outros trabalhos, entretanto mostra os resultados para cada ação, obtendo na maior média de resultado, cerca de 78% de *F1-score* para uma delas.

Zhang, Zohren e Roberts (2019) propõe um novo modelo também baseado nos trabalhos de Tsantekidis *et al.* (2017a), Tsantekidis *et al.* (2017b) utilizando uma CNN para extração de recursos e uma LSTM como classificador. O autor utiliza os dados de FI-2010 para comparar seu trabalho aos anteriores, além de um conjunto de dados da LSE de 5 ações de janeiro a dezembro de 2017. Como melhorias, o autor propõe uma CNN que atua como um filtro *Finite Impulse Response* (FIR), além da adição de um módulo *Inception* Szegedy *et al.* (2016) para extrair comportamentos dinâmicos de várias escalas de tempo. Para o treinamento, o autor utiliza as 2 configurações propostas por Tsantekidis *et al.* (2017a) para o FI-2010 e a configuração 2 para os dados da LSE. Para verificar a evidência de Sirignano e Cont (2019), o autor utiliza outras 5 ações da LSE para fazer uma transferência de aprendizado do modelo e verificar sua performance nas novas ações. Os resultados obtidos para a base de dados FI-2010 atingiram

cerca de 84% de *F1-score* para um dos horizontes de previsão, enquanto que para os dados da LSE atingiu cerca de 70% e para as novas ações, cerca de 68%, o que sugere que este modelo também pode extrair características universais de formação de preço. O autor também introduz uma análise do resultado do modelo utilizando o método LIME, que tenta explicar o impacto dos dados de entrada na classificação. Além disso, ele faz uma simulação de como o seu modelo se comportaria no mercado financeiro utilizando uma estratégia simples.

Ntakaris *et al.* (2019) propõe uma análise de performance entre extração de recursos automatizada ou feita de forma manual. Para isso, o autor utiliza diversos modelos de MLP, CNN e LSTM com diversos tipos de dados extraídos do LOB, como recursos econométricos e quantitativos, além dos propostos por Kercheval e Zhang (2015). Todos os modelos são testados em 2 ações de 10 dias consecutivos da bolsa de valores dos EUA e 5 ações da bolsa de valores finlandesa, utilizando os dois protocolos: o primeiro com a extração manual de recursos e o segundo com a extração automatizada. Os resultados mostram que recursos manuais fazem os modelos terem desempenho muito maior do que com os recursos automatizados, evidenciado por alguns casos em que o recurso manual faz com que o modelo tenha um desempenho 2x melhor.

Passalis *et al.* (2020) analisa o impacto da normalização dos dados no resultado final do modelo, utilizando o FI-2010 e o modelo proposto por Tsantekidis *et al.* (2017a). O foco do trabalho é testar uma normalização adaptativa, ao invés da normalização *z-score* comumente utilizada nos trabalhos relacionados ao LOB. Os resultados demonstram que a posposta melhora em cerca de 10% todas as métricas em relação a utilização da normalização *z-score*.

Tsantekidis *et al.* (2020) une CNN a LSTM dos seus trabalhos anteriores (TSANTEKIDIS *et al.*, 2017a; TSANTEKIDIS *et al.*, 2017b), utilizando um novo tipo de entrada em que são adicionados alguns recursos em relação aos dados anteriores. Apesar de os resultados da proposta superarem a dos modelos base (TSANTEKIDIS *et al.*, 2017a; TSANTEKIDIS *et al.*, 2017b), a diferença não é muito significativa.

A Tabela 3 resume os trabalhos que podem ser comparados utilizando o *F1-score* (por ser a métrica de maior relevância para comparação), incluindo o nome dos modelos, o tipo de conjunto de dados e a precisão alcançada. Como pode ser visto, os estudos com o LOB tem avançado o estado da arte nos últimos anos, evidenciando que estimativas baseadas em dados LOB tendem a ser mais precisas. Isso sugere que as informações de compra e venda e o fluxo de negociação presente na LOB são relevantes para este tipo de problema.

Tabela 3 – Relação das principais características dos trabalhos relacionados.

Trabalho	Modelo	Dados	F1-Score
(KERCHEVAL; ZHANG, 2015)	SVM	LOB	80%
(SIRIGNANO, 2016)	Sparse MLP	LOB	71%
(PERSIO; HONCHAR, 2017)	RNN/LSTM/GRU	OHLC	67%
(DOERING; FAIRBANK; MARKOSE, 2017)	CNN	LOB	67%
(TSANTEKIDIS *et al.*, 2017a)	CNN	LOB	71%
(TSANTEKIDIS *et al.*, 2017b)	LSTM	LOB	76%
(TRAN *et al.*, 2018)	TABL	LOB	79%
(MÄKINEN *et al.*, 2018)	CNN+LSTM+A	LOB	78%
(ZHANG; ZOHREN; ROBERTS, 2019)	CNN+LSTM	LOB	84%

3.4 Considerações Finais

Neste capítulo foram apresentados os trabalhos relacionados a este projeto de mestrado. Nas seções apresentadas foram descritas a metodologia para a definição da *string* de busca e dos critérios de seleção de trabalhos, além das principais características encontradas, resultando em uma análise crítica e uma tabela comparativa com as principais características dos mesmos.

CAPÍTULO

4

MATERIAIS E MÉTODOS

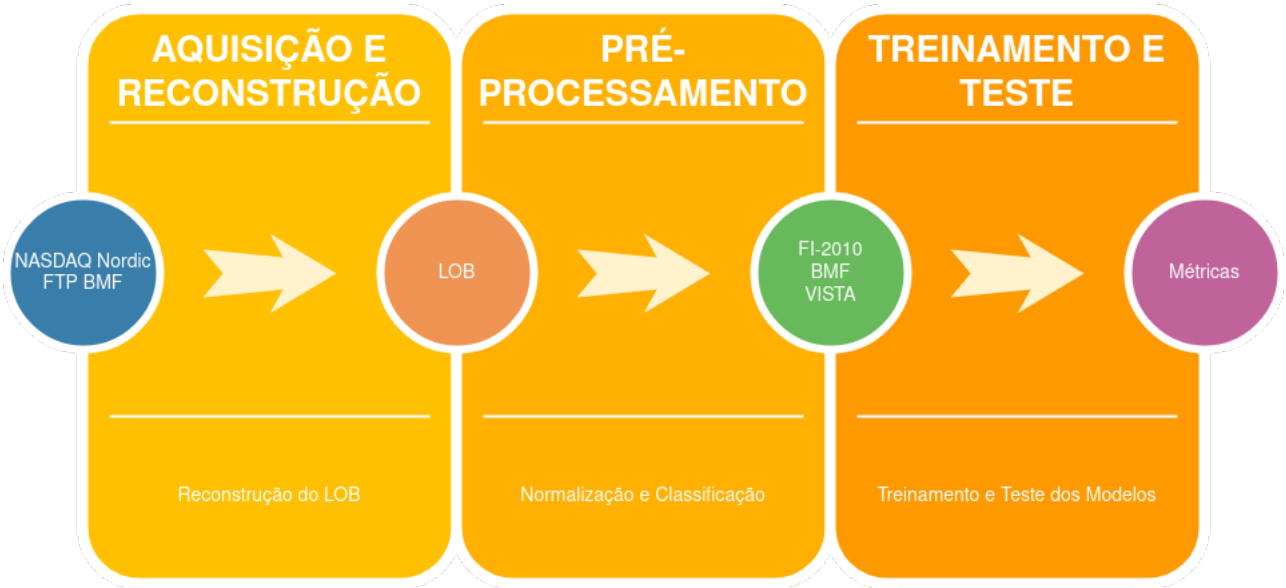
4.1 Considerações Iniciais

Neste capítulo serão apresentados os materiais utilizados e a metodologia proposta para este projeto de mestrado. Serão expostos os dados e sua manipulação, assim como os modelos e sua manipulação, por fim a proposta de experimentos para este trabalho. Na Seção 4.2 é apresentada uma visão geral do fluxo de desenvolvimento deste trabalho. Na Seção 4.3 são apresentadas as bases de dados utilizadas, assim como seu processamento, rotulação e divisão. Na Seção 4.4 são apresentados os extratores de recursos utilizados para compor o modelo híbrido, assim como as modificações necessárias para isso. Na Seção 4.5 são apresentadas as perguntas que geraram os experimentos deste trabalho.

4.2 Fluxo do Trabalho

A Figura 19 exemplifica visualmente o fluxo para desenvolvimento deste trabalho, em que primeiramente são obtidos dados de 2 fontes, *NASDAQ Nordic* e o o *File Transfer Protocol* (FTP) da BMF, ambos com a necessidade de serem reconstruídos como LOB. A seguir o LOB é pré-processado e classificado para gerar as bases de dados FI-2010, BMF e VISTA, que serão as entradas dos modelos. Por fim, os modelos são gerados, treinados e testados com as bases de dados. Este processo será detalhado nas seções a seguir.

Figura 19 – Esquema do fluxo de trabalho, desde a aquisição até o teste do modelo.



Fonte: Elaborada pelo autor.

4.3 Bases de Dados

Neste trabalho foram utilizados 2 bases de dados: FI-2010 e B3-2020. O primeiro é o mesmo FI-2010 usado em alguns trabalhos relacionados (MÄKINEN *et al.*, 2018; ZHANG; ZOHREN; ROBERTS, 2019; TSANTEKIDIS *et al.*, 2017a; TSANTEKIDIS *et al.*, 2017b), o outro da bolsa de valores brasileira (B3).

A base de dados FI-2010 foi gerada a partir da coleta de dados de 5 ações da bolsa de valores *Nasdaq Nordic* no período de 1º de junho de 2010 a 14 de junho de 2010, totalizando 10 dias úteis consecutivos (NTAKARIS *et al.*, 2018). A partir da coleta foi reconstruído um LOB com 10 níveis de preço, totalizando 40 colunas, além das características propostas por Kercheval e Zhang (2015) e concatenadas ao LOB, gerando 139 colunas, para cada ação a cada dia. Estes dados foram normalizados utilizando as funções *z-score*, *min-max* e precisão decimal, e assim classificados baseados em 5 horizontes de previsão diferentes ($k = 1, 2, 3, 5$, e 10), finalizando 144 colunas. As diferentes ações foram concatenadas por dia em cada um dos conjuntos normalizados e cada um destes conjuntos foi separado em 9 subconjuntos de treinamento e teste, em que o último dia de cada sequência é definido como teste e os dias anteriores de treinamento. Vale ressaltar que esses eventos FI-2010 estão publicamente disponíveis de forma sumarizada, contendo apenas o último evento (1 registro de 40 elementos) de cada sequência de 10 eventos.

As bases de dados B3-2020 foi gerada a partir dos dados históricos da B3 utilizando um sistema que foi expandido do simulador de Castro (2017), para reconstruir além do LOB, também as mensagens. Este sistema reconstrói o LOB no mesmo formato do arquivo de *order book* do LOBSTER, enquanto o arquivo de mensagens tem uma estrutura um pouco diferente da

estrutura do LOBSTER devido as diferenças dos dados do mercado brasileiro com a NASDAQ. Esta estrutura tem as seguintes informações descritas na Tabela 4.

B3-2020 utiliza dados da Bolsa de Mercadorias e Futuros (BMF) a partir da coleta dos dados do FTP[1] da BMF de Futuro de Taxa Média de Depósitos Interfinanceiros de Um Dia (DI1F20, DI1F21, DI1F23, DI1F25 e DI1F27) de 1º de novembro de 2017 a 20 de dezembro de 2017 e de 5 ações de alta liquidez da B3 (ABEV3, ITUB4, MGLU3, PETR4, VALE3) no período de 1º de agosto de 2019 até 30 de novembro de 2019. Estes intervalos foram escolhidos por serem períodos consecutivos de dados completos da B3[2] e estar fora de período eleitoral.

Tabela 4 – Variáveis contidas no arquivo de mensagens associadas com a base de dados B3-2020.

Variável	Descrição	Valor
timestamp	É o tempo da ocorrência do evento.	float: maior ou igual a 0.
direction	É o indicador de sentido da ordem, em que lado do livro de ofertas aquele evento vai alterar.	integer: 1 - compra / 2 - venda.
event_id	É o valor do código do evento da ordem, correspondendo à ação que aquela ordem realizará.	integer: 1 - Inserção / 2 - Modificação / 3 - Cancelamento - Solicitado pelo participante / 4 - Negociação / 11 - Expiração.
price	Preço da oferta ajustado no padrão LOBSTER, ou seja, multiplicado por 10000.	integer: valor maior que 0.
volume	Quantidade total do volume de ativos financeiros da oferta.	integer: valor maior ou igual a 0.
broker	Código de identificação da corretora.	integer.

4.3.1 Pré-processamento do LOB

Para a utilização dos dados do LOB, o pré-processamento descrito por Ntakaris *et al.* (2018) foi utilizado. Nele, cada amostra X apresentada ao modelo é uma janela com 100 eventos consecutivos do LOB, onde cada evento x_t possui 40 elementos que representam 10 níveis do LOB, cada um com: preço de *ask* e volume de *ask*, preço de *bid* e volume de *bid*. Sendo, $X = [x_1, x_2, ..., x_t, ..., x_{100}]^T \in \mathbb{R}^{100\times 40}$, onde $x_t = [p_a^{(i)}, v_a^{(i)}, p_b^{(i)}, v_b^{(i)}]_{n=10}^{i=1}$. p_a e p_b representam preço do lado da compra e do lado da venda e v_a e v_b representam volume do lado da compra e do lado da venda, para cada respectivo nível. Para manter a comparação com a base de dados FI-2010, foi selecionada a função de normalização *z-score*. Na função *z-score*, conforme Equação 4.1, em que x é a coluna (de preço ou volume) a ser normalizada, μ e σ são respectivamente a média e o

[1] http://ftp.bmf.com.br/
[2] Alguns dias de dados da B3 não são disponibilizados ou são disponibilizados de forma incompleta.

desvio padrão dos valores da coluna. μ e σ são calculados sobre o dia anterior para reduzir a diferença entre variação e distribuição dos valores de cada ação por dia.

$$z = \frac{x - \mu}{\sigma} \tag{4.1}$$

4.3.2 Rotulação

Para a rotulação dos dados do LOB, ao invés de utilizar a descrita por Ntakaris *et al.* (2018), foi utilizada a de Zhang, Zohren e Roberts (2019) que propõe uma modificação para suavizar a classificação do movimento. Nela, a rotulação de um evento considera a média dos k preços-médios antes e após um instante t, em comparação com um limiar α que define se o movimento é para cima, para baixo ou estacionário. O cálculo do preço médio p no momento t é calculado de acordo com a Equação 4.2, que é a média entre $p_a^{(1)}$ $p_b^{(1)}$ do nível 1, onde 1 representa os melhores preços para o lado de *bid* e *ask*.

$$p_t = \frac{p_a^{(1)}(t) + p_b^{(1)}(t)}{2} \tag{4.2}$$

Utilizando o horizonte de previsão k é calculada a média dos k eventos anteriores m_- e a média dos k eventos próximos m_+, conforme mostram as Equação 4.3 e Equação 4.4.

$$m_-(t) = \frac{1}{k}\sum_{i=0}^{k} p_{t-i} \tag{4.3}$$

$$m_+(t) = \frac{1}{k}\sum_{i=1}^{k} p_{t+i} \tag{4.4}$$

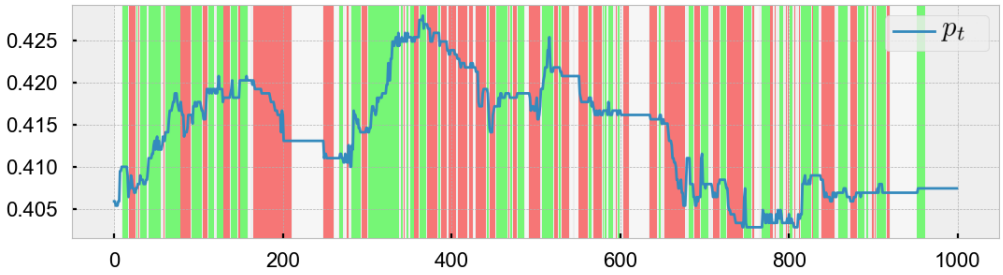
Assim, é obtida a relação l_t entre as médias m_- e m_+ pela Equação 4.5.

$$l_t = \frac{m_+(t) - m_-(t)}{m_-(t)} \tag{4.5}$$

Essa relação é comparada com um limiar α para dar a classificação do movimento do preço médio. Se $l_t > \alpha$, o movimento será classificado como subida $(+1)$, se $l_t < -\alpha$ como descida (-1); caso contrário, será considerado estacionário (0). Os valores de k e de α são definidos empiricamente e estão diretamente relacionados ao número de negócios possíveis e suas margens esperadas.

A Figura 20 ilustra a classificação de um período de alguns minutos com normalização *z-score* utilizando μ e σ do dia anterior, $k = 100$ e $\alpha = 0.0002$. Os rótulos classificados como descida (-1) estão em vermelho, como subida $(+1)$ em verde e os sem cor correspondem a estacionária (0).

Figura 20 – Um exemplo de classificação de 1000 pontos de um dia do conjunto de dados FI-2010.



Fonte: Elaborada pelo autor.

4.3.3 Pré-processamento das Mensagens

Para a utilização dos dados das mensagens, foi utilizado o pré-processamento das variáveis numéricas, enquanto as variáveis categóricas foram mantidas, como mostrado na Tabela 5.

Tabela 5 – Normalização de variáveis contidas no arquivo de mensagens associadas com a base de dados B3-2020.

Variável	Modificação
timestamp	Modificado para ser a diferença entre o atual timestamp e o anterior.
direction	Mantido.
event_id	Mantido.
price	Normalizado com a função de normalização escolhida para o experimento.
volume	Normalizado com a função de normalização escolhida para o experimento.
broker	Mantido.

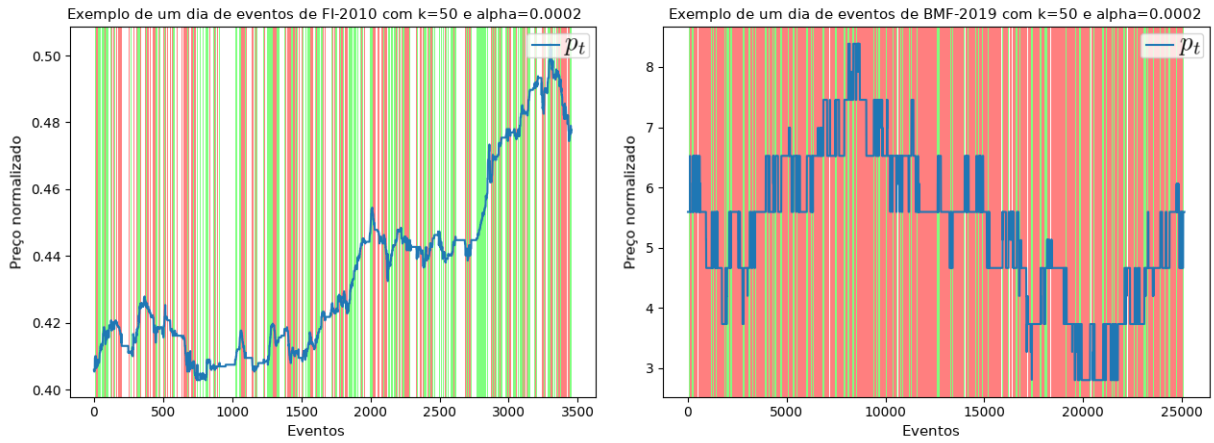
4.3.4 Separação de Conjuntos de Dados

Para seguir o mesmo fluxo de geração de FI-2010, descrito por Ntakaris *et al.* (2018), foram extraídos conjuntos de 10 dias das bases de dados B3-2020. Pela diferença entre a quantidade de eventos de instrumentos a vista e futuros, a fim de equiparar as bases, foi-se extraído um novo conjunto com 29 dias dados de B3-2020. Para a definição de parâmetros k e α, os três conjuntos foram normalizados com a função *z-score* e foram testados os k e α sugeridos por Ntakaris *et al.* (2018). A Figura 21 mostra a diferença entre FI-2010 e B3-2020 ao adotar os mesmos parâmetros $k = 50$ e $\alpha = 0.0002$ e observa-se a diferença entre as curvas de preço médio entre os ativos de um dia de cada bolsa, em que em B3-2020 há uma variação maior no eixo "Preço normalizado". A Figura 22 ilustra os cenários em B3-2020 da variação de k, mantendo o mesmo α e observa-se que aumentando o k a classificação fica menos sensível a variação de preço médio, evidenciado pela quantidade maior de classificações do tipo estacionária (em

branco). E a Figura 23 ilustra os cenários em B3-2020 da variação de α, mantendo o mesmo k e observa-se que aumentando o α a classe estacionária aumenta, o que é esperado.

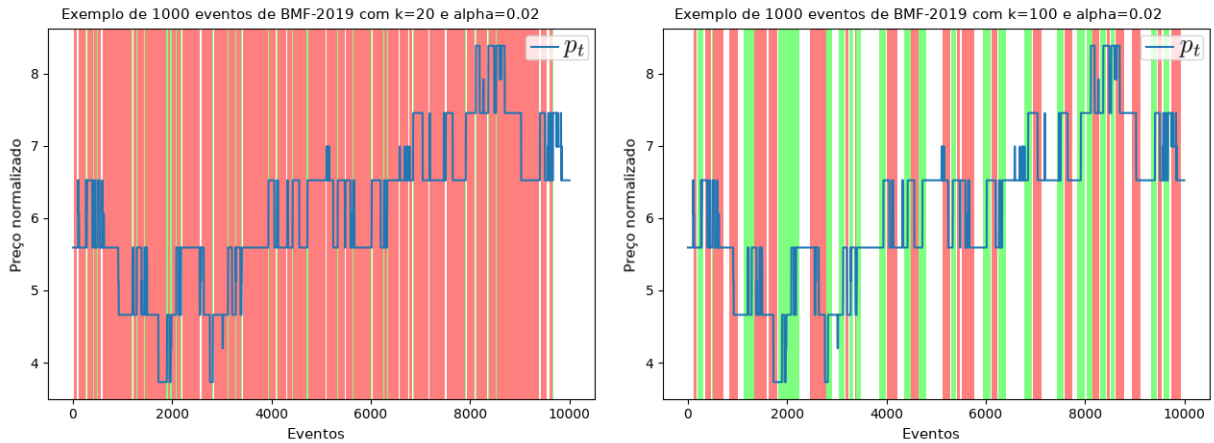
Para a distribuição ideal de classes (NTAKARIS *et al.*, 2018), em que a classe estacionária é representada por 80% a 90% de classificações, foram necessários ajustes para cada base selecionada (mostradas no Capítulo 5). A definição de parâmetros fixos para todos os instrumentos do conjunto gera maiores variações na distribuição de classes diárias, segundo Passalis *et al.* (2020), uma normalização adaptativa para cada instrumento a cada dia pode impactar o resultado positivamente, entretanto para manter uma comparação adequada entre os modelos e as bases de dados, foram definidos valores fixos para k e $alpha$ para todos os ativos de um conjunto de dados, assim como em FI-2010.

Figura 21 – Comparação entre FI-2010 and B3-2020 mostrando resultados de classificação, considerando o mesmos parâmetros α e k.



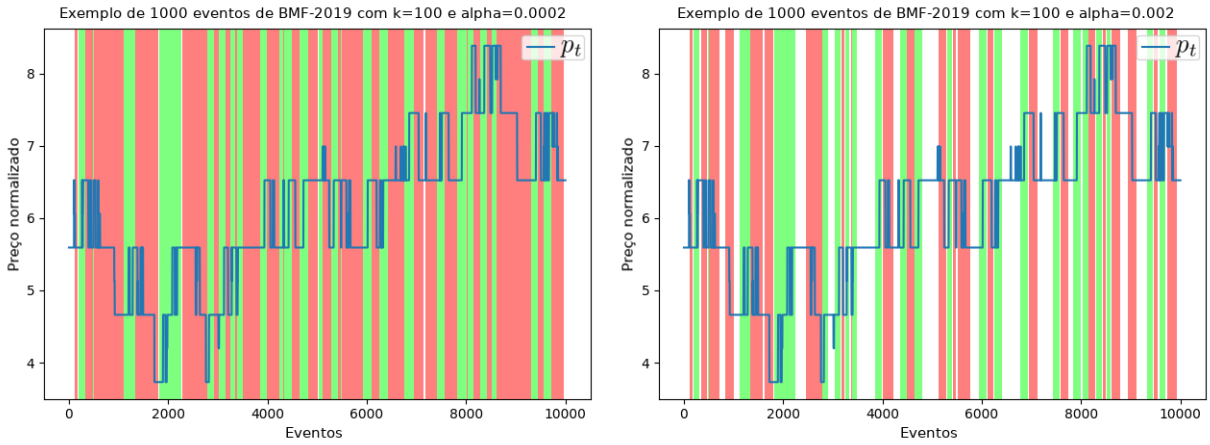
Fonte: Elaborada pelo autor.

Figura 22 – Cenário em que k é variado, mantendo o mesmo α.



Fonte: Elaborada pelo autor.

Figura 23 – Cenário em que α é variado, mantendo o mesmo k.



Fonte: Elaborada pelo autor.

4.4 Extrator de recursos com CNN

Neste trabalho é adotada a configuração do extrator de recursos de Zhang, Zohren e Roberts (2019), que obteve o melhor resultado em relação aos trabalhos relacionados com o FI-2010. Nele foram apresentadas duas versões de configuração de convoluções, ambas unidas a um mesmo módulo *Inception* (SZEGEDY *et al.*, 2016).

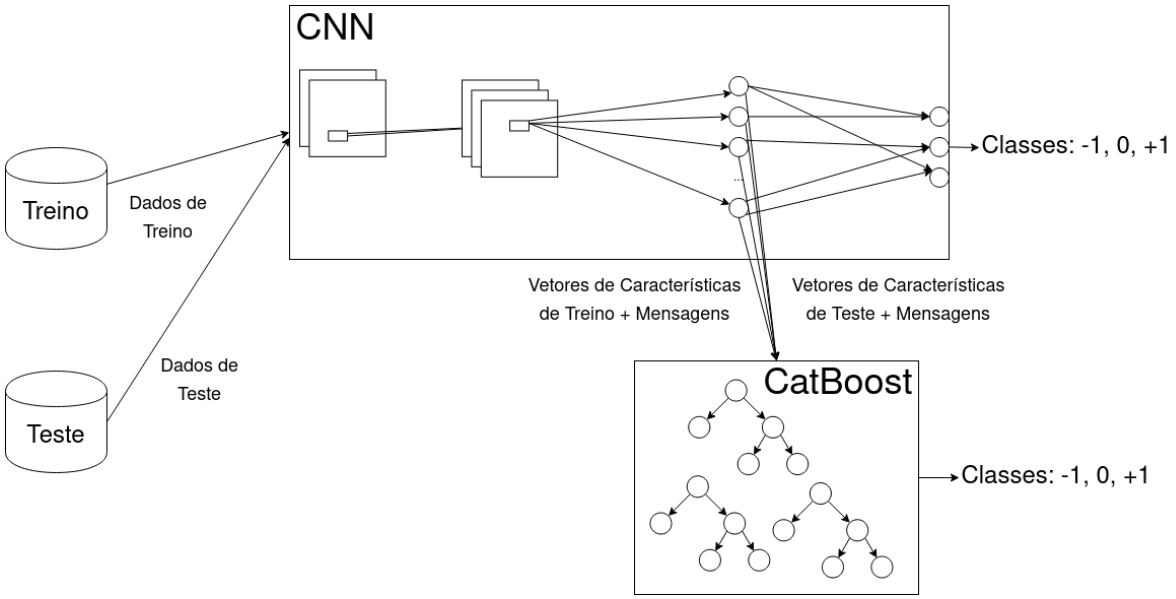
- Versão 1: Composta por um *kernel* de (4x40), para extrair recursos universais apenas olhando para a composição atual do LOB, seguidos de *kernels* de (1x1) e (4x1) para extrair propriedades não lineares dos dados.

- Versão 2: Composto por um conjunto de dois blocos com *kernels* (1x2) para extrair a relação entre preço e volume do LOB para cada nível utilizando *stride* de (1x2) para evitar sobreposição de informações, seguidos de dois *kernels* de (4x1) para extrair propriedades não lineares dos dados, atuando como um filtro de sinal popular de suavisação (FIR).

O módulo *Inception* contém três camadas convolucionais com *kernels* em paralelo de (1x1), (3x1) e (5x1) que recebem as mesmas entradas e concatenam suas saídas.

4.4.1 Modificação do modelo de Rede Neural para Híbrido

Figura 24 – Fluxo de treinamento para o modelo híbrido.



Fonte: Elaborada pelo autor.

O modelo híbrido usa o fluxo de treinamento mostrado na Figura 24, em que parte-se da divisão do conjunto de dados entre conjuntos de treinamento e teste (baseado na proposta de Tsantekidis *et al.* (2017a), para comparação dos experimentos). Para treinar a CNN, são utilizados os conjuntos de treinamento e teste, sendo que o conjunto de treinamento é utilizado para atualizar os pesos da CNN em cada iteração, enquanto o conjunto de teste é utilizado no final de cada época de treinamento para avaliar seu desempenho por ser um conjunto de dados que não foi apresentado ao modelo. Ao final do treinamento, o conjunto de teste é apresentado para cálculo das métricas do modelo, tal procedimento é utilizado pela maioria das abordagens tradicionais, entretanto normalmente o conjunto utilizado nesta etapa é nomeado de conjunto de validação.

Com o modelo de CNN treinado, a camada de classificação é removida, gerando um novo modelo, denominado extrator, cuja saída é um vetor de características. Com o extrator são calculados os vetores de características dos conjuntos de treinamento e teste para realizar o treinamento do *CatBoost*. Os vetores de características extraídas podem ser concatenados com outras informações, por exemplo os dados de mensagens.

4.5 Experimentos

Para estabelecer a metodologia dos experimentos com base nos objetivos deste trabalho, foram-se feitas as seguintes perguntas:

- Pergunta 1: Os modelos propostos na literatura funcionam com os dados brasileiros?
- Pergunta 2: A LSTM como classificador faz uma grande diferença?
- Pergunta 3: Um modelo híbrido pode ter um desempenho melhor?
- Pergunta 4: As informações das mensagens tem impacto sobre a classificação do modelo?
- Pergunta 5: O treinamento baseado em menos dias é melhor do que um baseado em mais dias?
- Pergunta 6: Em um ambiente real de produção, o tempo de treinamento e predição utilizando uma LSTM é viável?

A Pergunta 1 se baseia nas sugestões de Zhang, Zohren e Roberts (2019), Mäkinen *et al.* (2018), Sirignano (2016), de que os modelos de RNAs podem extrair características universais do LOB. Para abordar esta pergunta, foi testado o modelo com melhor desempenho em FI-2010, proposto por Zhang, Zohren e Roberts (2019) na base de dados B3-2020.

A Pergunta 2 se baseia nas sugestões de Zhang, Zohren e Roberts (2019), de que a LSTM é importante para extrair as relações temporais entre os dados. Para abordar esta pergunta, foi testada a substituição da camada LSTM por uma camada MLP.

A Pergunta 3 se baseia nas sugestões de Cao *et al.* (2013), Luo *et al.* (2018), de que a um classificador externo a RNA pode gerar um desempenho melhor do modelo. Para abordar esta pergunta, foi testada a substituição da camada de classificação do modelo pelo *CatBoost* (detalhada na Subseção 4.4.1).

A Pergunta 4 se baseia nas sugestões de Poskitt *et al.* (2011), de que a anonimização das informações de corretora podem melhorar a liquidez do mercado. Para abordar esta pergunta, foi testada a adição das informações de mensagens, que contém a corretora, às informações extraídas do LOB pelo extrator.

A Pergunta 5 se baseia nas sugestões de Sirignano (2016), de que uma um volume maior de dados de treinamento supera um volume menor. Para abordar esta pergunta, foram testadas divisões de treinamento e teste com mais e menos dias com a base de dados B3-2020.

A Pergunta 6 se baseia nos resultados de Zhang, Zohren e Roberts (2019), de que é possível obter lucro com uma estratégia simples utilizando seu modelo. Para abordar esta pergunta, foram testados os tempos de treinamento e predição dos modelos analizados para identificar a sua viabilidade em um ambiente real.

Para os experimentos foram propostos três pares de modelos de inferência, onde cada par tem uma CNN versão 1 e uma CNN versão 2 da Seção 4.4. O primeiro par vem diretamente do trabalho de Zhang, Zohren e Roberts (2019) para ser tomado como referência de comparação e aborda a pergunta 1. O segundo par substitui a LSTM por MLP para abordar a pergunta 2.

O terceiro par, que é o modelo híbrido capaz de incorporar informações de corretora, substitui a LSTM por *CatBoost*, abordando as perguntas 3 e 4. Para a pergunta 5, os modelos foram treinados em 2 subconjuntos da base de dados B3-2020 com 10 e 32 dias. Para a pergunta 6, foram armazenados todos os tempos de previsão de todos os experimentos.

Para comparar os resultados deste trabalho com os trabalhos relacionados, foi utilizada a configuração 2 proposta por Tsantekidis *et al.* (2017a) para todas as bases de dados, em que 70% dos dias do conjunto de dados são utilizados para treinamento e 30% dos dias para teste.

O treinamento foi realizado em um *Intel Core* i7-8750H, *NVIDIA GeForce* GTX 1070 e 32GB de RAM, utilizando a plataforma *Tensorflow* em *Python* para construção e teste dos modelos. Todos os modelos foram treinados utilizando o mesmo tamanho de *batch* e parâmetros de otimização. O classificador *CatBoost* do terceiro par foi treinado por 1000 iterações utilizando os parâmetros originais fornecidos pela biblioteca[3].

4.6 Considerações Finais

Neste capítulo foram apresentados os materiais e metodologia propostas para este projeto de mestrado. Nas seções apresentadas foram descritos os dados, em que foram detalhados os processos de processamento, rotulação e divisão e para os extratores, foram detalhados os modelos e as manipulações necessárias para compor o modelo híbrido, e por fim, as perguntas que geraram os experimentos.

[3] https://catboost.ai/

CAPÍTULO

5

RESULTADOS

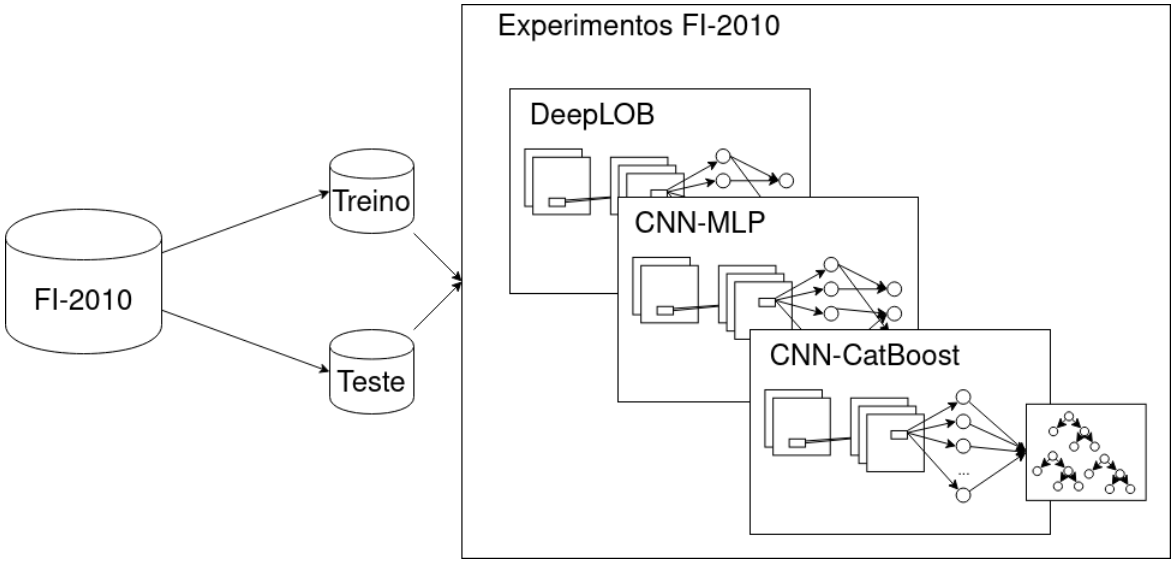
5.1 Considerações Iniciais

Neste capítulo serão apresentados os resultados relacionados a este projeto de mestrado. Serão expostos ao longo das seções os resultados obtidos a partir dos experimentos propostos utilizando as bases de dados finlandesa e brasileira, assim como a análise destes resultados, respondendo as perguntas que geraram os experimentos. Na Seção 5.2 são apresentados os resultados relacionados a base de dados finlandesa de *benchmark* (FI-2010). Na Seção 5.3 são apresentados os resultados para as bases de dados do mercado financeiro brasileiro, tanto com dados (BMF) como a vista (VISTA).

5.2 Resultados para a Base de Dados FI-2010

A Figura 25 exemplifica o processo de divisão dos conjuntos da base de dados FI-2010 para os experimentos com os 3 pares de modelos. Em que são separados da base um conjunto para treinamento e outro para teste. Todos os dados são iguais para cada um dos modelos.

Figura 25 – Exemplo da divisão da base de dados FI-2010 para os experimentos com os 3 pares de modelos.



Fonte: Elaborada pelo autor.

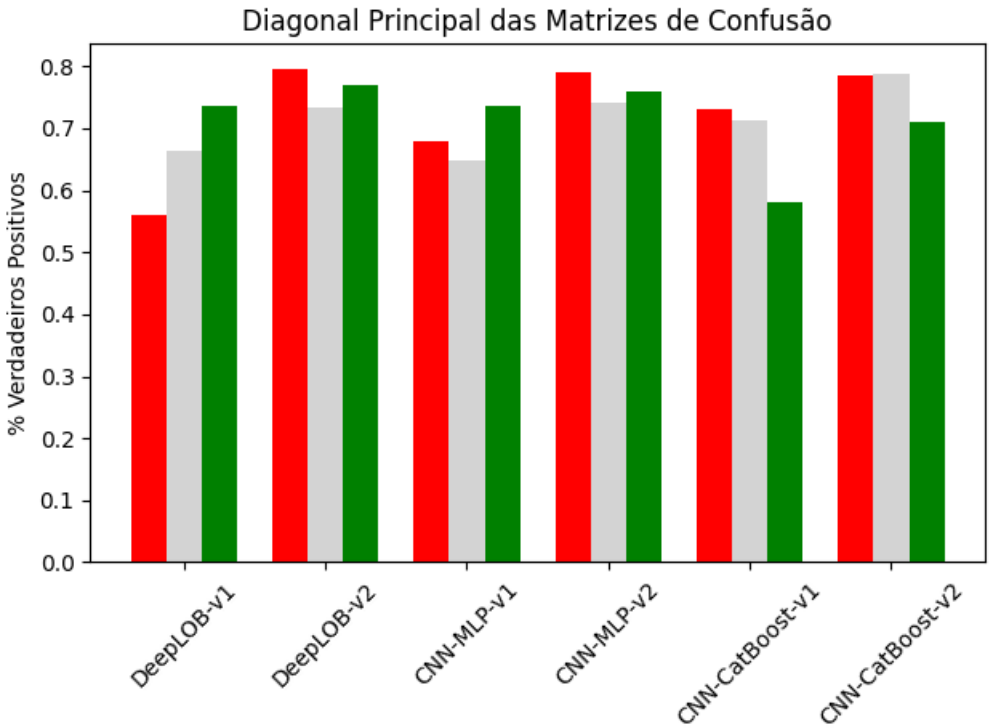
As métricas obtidas para cada um dos seis modelos explorados (3 pares, conforme descrito na Seção 4.5) sobre o conjunto de dados FI-2010 são apresentadas na Tabela 6, utilizando os parâmetros $k = 20$ e $\alpha = 0.00002$, propostos em Ntakaris *et al.* (2018). Analisando a diferença entre os pares DeepLOB e CNN-MLP, pode-se notar que as métricas de desempenho permanecem semelhantes nos modelos v2, enquanto em v1 os resultados foram melhores com MLP. No entanto, comparando v1 com v2, v2 apresenta melhores resultados para todas as métricas. Em relação ao tempo de treinamento do extrator, a diferença entre LSTM e MLP é significativa, alcançando uma redução de 93% para v1 e 84% para v2. Os tempos de treinamento do *CatBoost* são aproximadamente os mesmos para qualquer combinação. A diferença nas métricas entre os pares CNN-MLP e CNN-*CatBoost* tem uma variação inexpressiva. Cabe ressaltar, que neste banco de dados não há dados de mensagens.

Tabela 6 – Métricas obtidas pelas 2 versões de cada um dos 3 pares de modelos analisados no conjunto de teste da base de dados FI-2010.

Modelo	Loss	Acurácia	Precisão	Recall	F1-Score	Tempo
DeepLOB-v1	0.78	0.65	0.62	0.58	0.69	970s
DeepLOB-v2	0.57	0.77	0.76	0.73	0.79	1144s
CNN-MLP-v1	0.74	0.69	0.67	0.63	0.72	62s
CNN-MLP-v2	0.59	0.77	0.76	0.73	0.79	181s
CNN-CatBoost-v1	0.73	0.69	0.66	0.61	0.73	65s
CNN-CatBoost-v2	0.59	0.77	0.75	0.73	0.78	184s

Fonte: Dados da pesquisa.

Figura 26 – Diagonal principal das matrizes de confusão dos pares de modelos para a base de dados FI-2010.



Fonte: Elaborada pelo autor.

A Figura 26 mostra a diagonal principal das matrizes de confusão desses 6 modelos aplicados ao mesmo conjunto de teste, as colunas em vermelho representam as classificações de descida, as em cinza de estacionário e em verde de subida. Os modelos v2, no geral se sobressaem aos modelos v1, com porcentagem de verdadeiros positivos superiores, corroborando a análise da tabela que mostra melhores resultados gerais para v2. Entre DeepLOB e CNN-MLP existe apenas uma pequena variação na distribuição das classificações, contrastando a suposição de Zhang, Zohren e Roberts (2019) de que LSTM é importante para capturar as relações temporais entre os dados. Este estudo sugere que o MLP é suficiente para captar essas relações, mas com a vantagem de reduzir significativamente o tempo de treinamento. Para CNN-MLP e CNN-CatBoost também existe uma pequena variação na distribuição das classificações.

Devido ao teste com todos os pares de modelos utilizando 100% de unidades, foram realizados testes de inferência para cada um deles e os tempos registrados são mostrados na Tabela 7.

Tabela 7 – Tempos de inferência para cada par de modelos testados.

Par	Tempo de Inferência
DeepLOB-v1	2.1ms
DeepLOB-v2	2.4ms
CNN-MLP-v1	2.2ms
CNN-MLP-v2	2.4ms
CNN-CatBoost-v1	2.2ms+1.4ms
CNN-CatBoost-v2	2.4ms+1.4ms

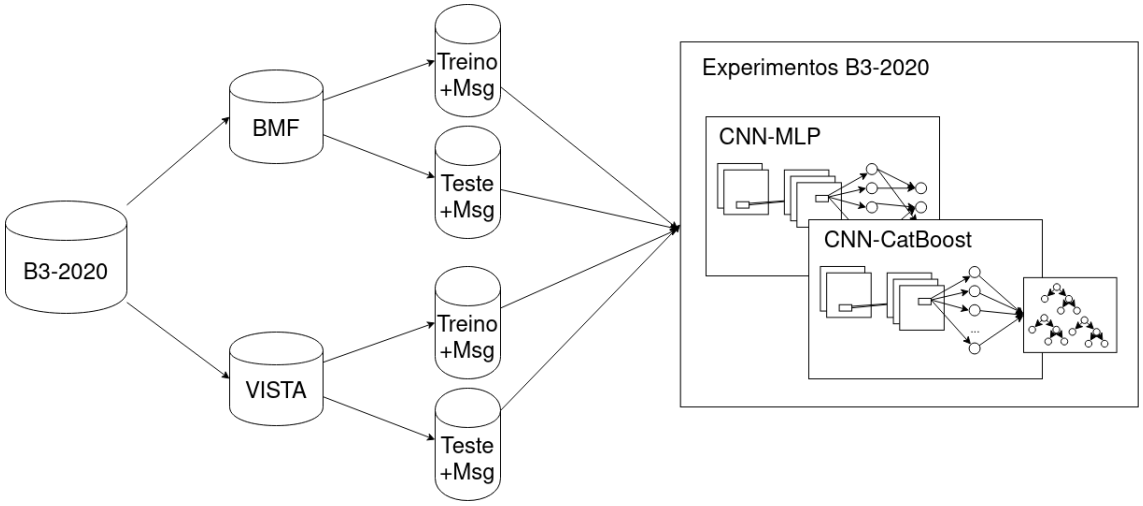
Fonte: Dados da pesquisa.

Da Tabela 7 é notável que em geral qualquer RNA, independente de classificador tem basicamente o mesmo tempo de inferência, variando um pouco pela complexidade do extrator no caso de v1 ser um pouco mais veloz. Já a adição do *CatBoost* aumenta o tempo de classificação em mais de 50%, por se tratar de um modelo a parte da RNA.

5.3 Resultados para as Bases de Dados da B3

A Figura 27 exemplifica o processo de divisão dos conjuntos da base de dados B3-2020 para os experimentos com os 3 pares de modelos. Em que são separados da base um conjunto para treinamento e outro para teste. Todos os dados são iguais para cada um dos modelos.

Figura 27 – Exemplo da divisão da base de dados B3-2020 para os experimentos com os 3 pares de modelos.



Fonte: Elaborada pelo autor.

Por conta do longo tempo de treinamento em FI-2010 (que é composto por 10 dias e sumarizado), este trabalho dividiu a análise dos resultados em 2 etapas: analisar um único instrumento de cada base de dados com variações de pares de modelos e suas unidades, parâmetros k e

α e quantidade de dias de treinamento; analisar os 5 instrumentos concatenados de cada base de dados com variações de unidades para o par CNN-MLP, dado k e α fixados e o maior número de dias de treinamento disponível.

5.3.1 Resultados para um instrumento

Para a definição da melhor configuração do extrator, foram selecionados 1 instrumento de cada base de dados, DI1F20 e ABEV3 de B3-2020. Foram selecionados dois horizontes de previsão ($k = 10, 100$) e para cada um foi selecionado um α que mantivesse uma distribuição média de 80 a 90% de dados com classe estacionária e o restante dividido igualmente para as classes de subida e descida. Além disso, foram selecionados dois subconjuntos de dados para cada um dos instrumentos:

- Conjunto 1: ABEV3 de 13 de agosto de 2019 a 26 de agosto de 2019 (10 dias).
- Conjunto 2: ABEV3 de 02 de novembro de 2019 a 15 de novembro de 2019 (10 dias).
- Conjunto 3: DI1F20 de 3 de novembro de 2017 a 17 de novembro de 2017 (10 dias).
- Conjunto 4: DI1F20 de 3 de novembro de 2017 a 20 de dezembro de 2017 (32 dias).

Utilizando o primeiro par DeepLOB nos conjuntos de dados selecionados os modelos sofrem sobre-ajuste, caracterizando que a complexidade do modelo é maior do que o necessário para aprender os padrões do conjunto de dados, o que indica a necessidade de reduzir a complexidade da RNA ou aumentar a quantidade de dados. Antes de aumentar a quantidade de dados, devido aos resultados em FI-2010 de 10 dias, em que os modelos convergem, foi testada uma abordagem de redução de complexidade da RNA sem alterar sua arquitetura, reduzindo a quantidade de filtros convolucionais por camada, baseado na ideia de *fine-tuning* do modelo (STATHAKIS, 2009). O Apêndice A mostra a tabela com todos os experimentos feitos com as combinações de conjuntos de dados, horizontes de previsão e complexidade do modelo.

Dos resultados obtidos foi observado que para um $k = 10$, a distribuição de classes fica mais próxima do ideal (descida: 5%, estacionária: 90%, subida 5%), tornando viável a utilização da maioria dos modelos. Para um $k = 100$, a variação da distribuição de classes diárias é maior, tornando o problema mais complexo. No caso de DI1F20, o conjunto com 32 dias tem resultados melhores para os modelos menos complexos, mas para ABEV3 nenhum modelo tem um bom resultado. No geral, os melhores resultados se concentram em modelos com 25% ou 50% de filtros, independente do uso de LSTM. Com o aumento de filtros, os modelos demonstram uma tendência maior ao sobre-ajuste.

5.3.2 Resultados para Cinco Instrumentos

A partir dos resultados da análise anterior, foram aplicados aos instrumentos futuros os mesmos parâmetros k e α de DI1F20 e para os instrumentos a vista os mesmos parâmetros de ABEV3. Foram concatenados os 5 instrumentos a vista, dia a dia, formando o novo conjunto VISTA no mesmo período de ABEV3; e o mesmo processo para os instrumentos futuros, formando o novo conjunto BMF. Os conjuntos BMF e VISTA foram testados com todas as variações de filtros utilizando o par CNN-MLP, pois a partir das conclusões da Seção 5.2, o par DeepLOB não foi utilizado.

5.3.2.1 Resultados para o Conjunto BMF

As Tabela 8 e Tabela 9 mostram os resultados das métricas de cada modelo sobre o conjunto BMF com a variação da porcentagem de filtros em relação ao modelo de referência, o tempo médio de treinamento por época e a análise gráfica do treinamento sobre o ajuste do modelo.

Tabela 8 – Comparação da CNN-MLP-v1 variando o número de filtros utilizando o conjunto de treinamento BMF.

CNN versão 1							
% de Filtros	Loss	Acurácia	Precisão	Recall	F1-Score	Tempo	Ajuste
12.5%	0.90	0.58	0.58	0.50	0.52	800s	Sub-ajustado
25.0%	0.86	0.61	0.62	0.55	0.57	759s	Bem ajustado
50.0%	1.10	0.62	0.63	0.58	0.59	870	Sobre-ajustado
100.0%	1.17	0.59	0.60	0.55	0.57	790	Sobre-ajustado

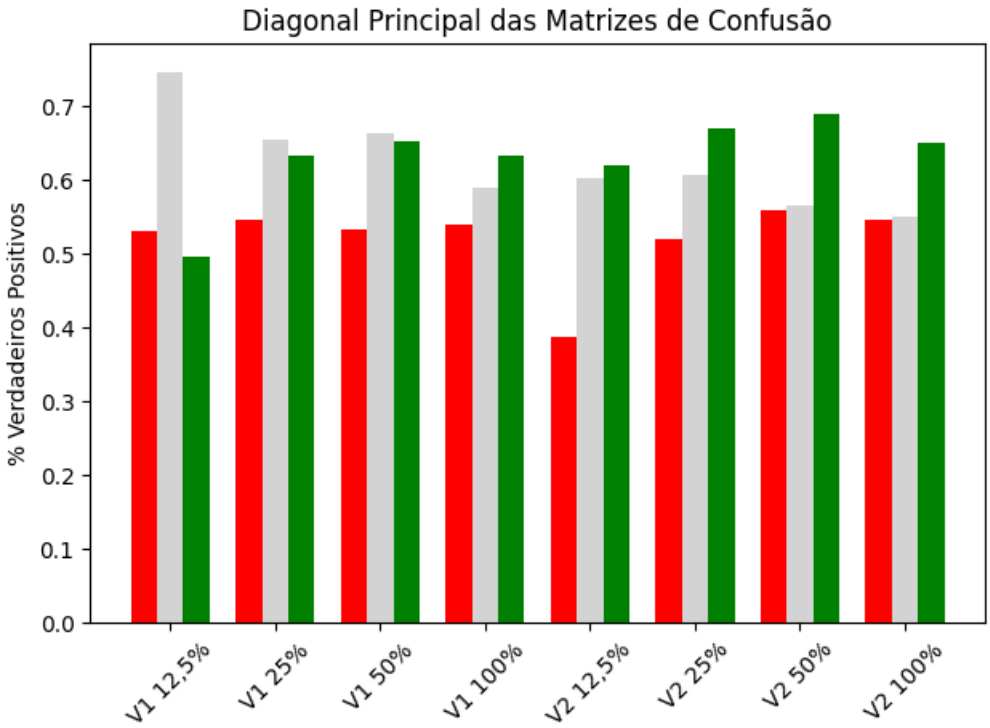
Fonte: Dados da pesquisa.

Tabela 9 – Comparação da CNN-MLP-v2 variando o número de filtros utilizando o conjunto de treinamento BMF.

CNN versão 2							
% de Filtros	Loss	Acurácia	Precisão	Recall	F1-Score	Tempo	Ajuste
12.5%	0.94	0.54	0.55	0.40	0.44	1000s	Bem ajustado
25.0%	0.86	0.61	0.61	0.54	0.56	1360s	Bem ajustado
50.0%	1.01	0.61	0.62	0.55	0.58	1420s	Sobre-ajustado
100.0%	1.20	0.57	0.58	0.54	0.55	1620s	Sobre-ajustado

Fonte: Dados da pesquisa.

Figura 28 – Diagonal principal das matrizes de confusão do par CNN-MLP utilizando 12,5%, 25% e 50% e 100% dos filtros, respectivamente.



Fonte: Elaborada pelo autor.

A Figura 28 mostra a diagonal principal das matrizes de confusão do par CNN-MLP utilizando 12,5%, 25% e 50% e 100% dos filtros, aplicados ao mesmo conjunto de teste, as colunas em vermelho representam as classificações de descida, as em cinza de estacionário e em verde de subida. A partir da Figura 28, nota-se que para v1 no caso de 12,5% há um desbalanceamento de acertos entre a classe estacionário e as outras, isto corrobora com a Tabela 8 que indica um sub-ajuste. Já para mais de 25%, o aumento das unidades gera um resultado inferior de acertos, indicando o sobre-ajuste. Enquanto para v2, o aumento dos filtros parece melhorar o resultado, porém com 50% de unidades há uma queda nos acertos, indicando sobre-ajuste.

5.3.2.1.1 Resultados para o Par CNN-CatBoost no Conjunto BMF

Para analisar o impacto das mensagens, disponível para B3-2020, foram realizados os experimentos sem e com as mensagens, denominados BMF-MSG0 e BMF-MSG1, respectivamente. O primeiro conjunto, sem adicionar nenhum dado das mensagens, foi usado como base de comparação. O segundo conjunto, com o acréscimo de todos os campos das mensagens, foi utilizado para verificar a relevância dos demais dados disponíveis para o resultado dos modelos. Foram analisados apenas os modelos CNN-MLP que não apresentaram sobre-ajuste. As

Tabela 10, Tabela 11 e Tabela 12 incluem as métricas obtidas para os modelos CNN-MLP e CNN-CatBoost com cada uma das opções BMF-MSG0 e BMF-MSG1 para estes modelos.

Para entender a relevância de cada variável para a predição do modelo híbrido, foi utilizada a biblioteca SHAP, que calcula valores de importância para mostrar a influência das variáveis para cada predição particular (LUNDBERG; LEE, 2017), como mostram as Figura 29 e Figura 30sobre o conjunto de teste.

Tabela 10 – Métricas para os modelos CNN-MLP-v1 e CNN-CatBoost-v1 com 25% dos filtros considerando as opções BMF-MSG0 e BMF-MSG1.

CNN versão 1 com 25% dos filtros						
Classificador	Conjunto	Loss	Acurácia	Precisão	Recall	F1-Score
MLP	original	0.856	0.613	0.621	0.550	0.575
CatBoost	BMF-MSG0	0.751	0.672	0.669	0.666	0.667
CatBoost	BMF-MSG1	0.747	0.693	0.690	0.687	0.689

Fonte: Dados da pesquisa.

Tabela 11 – Métricas para os modelos CNN-MLP-v2 e CNN-CatBoost-v2 com 12,5% dos filtros considerando as opções BMF-MSG0 e BMF-MSG1.

CNN versão 2 com 12,5% dos filtros						
Classificador	Conjunto	Loss	Acurácia	Precisão	Recall	F1-Score
MLP	original	0.941	0.545	0.549	0.400	0.440
CatBoost	BMF-MSG0	0.834	0.606	0.605	0.601	0.601
CatBoost	BMF-MSG1	0.831	0.632	0.631	0.626	0.627

Fonte: Dados da pesquisa.

Tabela 12 – Métricas para os modelos CNN-MLP-v2 e CNN-CatBoost-v2 com 25% dos filtros considerando as opções BMF-MSG0 e BMF-MSG1.

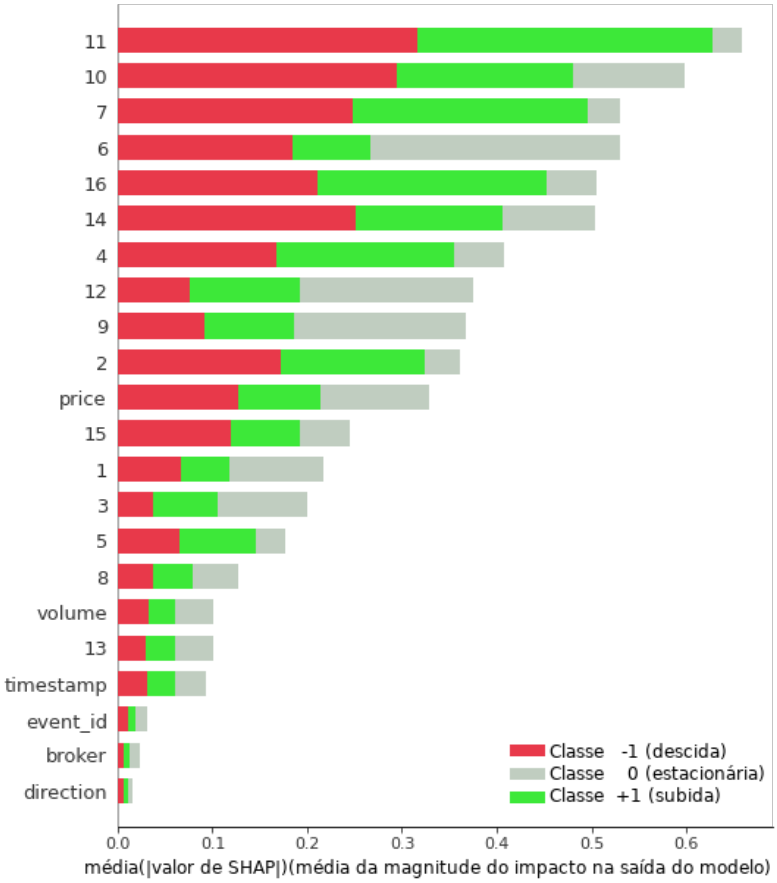
CNN versão 2 com 25% dos filtros						
Classificador	Conjunto	Loss	Acurácia	Precisão	Recall	F1-Score
MLP	original	0.862	0.606	0.613	0.536	0.563
CatBoost	BMF-MSG0	0.741	0.652	0.648	0.644	0.645
CatBoost	BMF-MSG1	0.737	0.673	0.670	0.666	0.667

Fonte: Dados da pesquisa.

As Tabela 10, Tabela 11 e Tabela 12 mostram que a adição do classificador *CatBoost* melhora em média 5% os valores das métricas em comparação aos modelos com classificador MLP. Enquanto que a adição das mensagens (opção BMF-MSG1) gera um aumento extra de 2 a 3%.

Para a análise utilizando a ferramenta SHAP, foi utilizado o CNN-CatBoost-v2 com 25% dos filtros, dada que esta é considerada a melhor configuração para o conjunto BMF-MSG1.

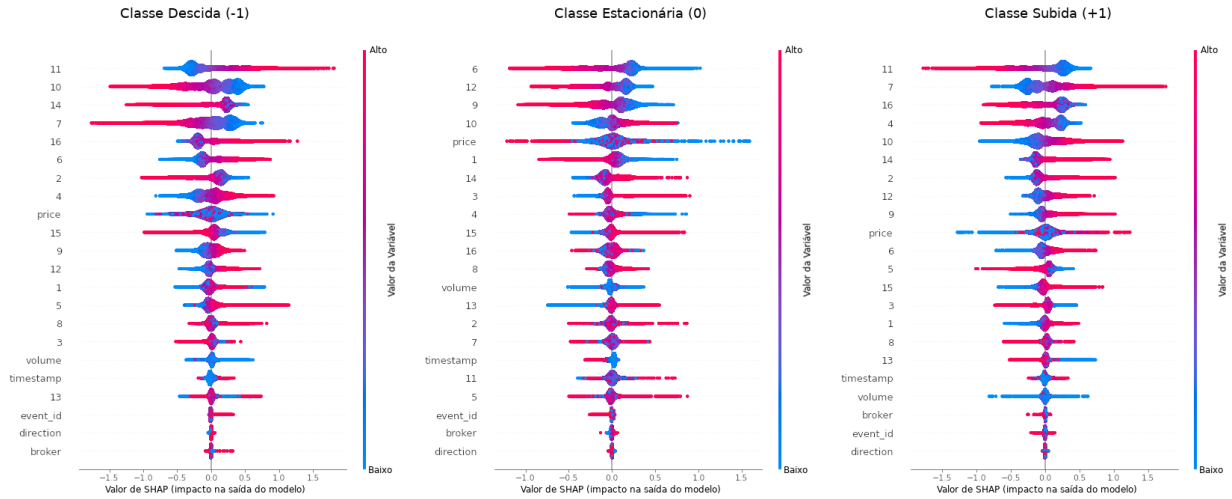
Figura 29 – O valor médio de SHAP para cada variável usado pelo classificador para cada uma das classes.



Fonte: Elaborada pelo autor.

A Figura 29 mostra um resumo geral do impacto de cada variável de entrada na saída do modelo. O gráfico classifica as variáveis em ordem de relevância de classificação e as mais relevantes são aquelas obtidas pela parte do modelo da CNN. As variáveis das mensagens também impactam positivamente o modelo de classificação em 2-3%, conforme visto na Tabela 12.

Figura 30 – O valor SHAP dos dados de entrada do classificador para cada uma das classes.



Fonte: Elaborada pelo autor.

A Figura 30 mostra uma visão mais detalhada da importância de cada variável na classificação final do modelo. Cada variável possui seu próprio gráfico composto por um conjunto de pontos, onde cada ponto representa o quão importante foi aquela variável (valor de SHAP) para o modelo prever tal classe. Como visto, esse impacto na saída do modelo pode ser positivo ou negativo. A cor indica simplesmente o valor da variável da amostra, que não é o valor de SHAP.

É apresentada uma análise considerando as 3 primeiras variáveis mais relevantes de cada uma das classes, que são: (11, 10 e 7) para *descida*, (6, 9 e 12) para *estacionária* e (11, 16 e 7) para *subida*. Na classe *descida*, a variável 11 tem um forte impacto positivo na classificação, as variáveis 10 e 7 se comportam de forma oposta, tendo um forte impacto negativo. Na classe *estacionária*, as variáveis 6, 9 e 12 têm um forte impacto negativo na classificação e um impacto marginal positivo. Na classe *subida*, a variável 11 tem forte impacto negativo, o que colabora com a análise da variável 11 para a classe *descida* onde a influência é na direção oposta. Os valores da variável 16 se comportam de maneira semelhante aos de 11, porém seu impacto negativo é menor para valores mais altos. Finalmente, a variável 7 se comporta ao contrário da 11.

Para as variáveis contidas nas mensagens foi analisado o impacto do preço e do volume, que são comumente informados em dados de mercado. O impacto das da variável corretor, que é menos comum de ser encontrada, também foi analisado. Na classe *descida*, as variáveis de preço e volume têm uma distribuição equilibrada com valores baixos, em que valores baixos têm forte impacto tanto positivo quanto negativo, enquanto a variável corretor tem impacto marginal positivo na classificação final. Para a classe *estacionária*, a variável preço influencia negativamente se for baixa e positivamente se for alta, enquanto a variável volume só impacta o

modelo se for baixa, porém afeta tanto negativamente quanto positivamente. A variável corretor tem influência marginal. Para a classe *subida*, a variável preço tem comportamento oposto ao *estacionária*, enquanto a variável volume se comporta de forma semelhante. A variável corretor tem uma pequena contribuição negativa.

5.3.2.2 Resultados para o Conjunto VISTA

Da mesma maneira que foi feito para o conjunto BMF, as Tabela 13 e Tabela 14 mostram os resultados das métricas de cada modelo sobre o conjunto VISTA com a variação da porcentagem de filtros em relação ao modelo de referência, o tempo médio de treinamento por época e a análise gráfica do treinamento sobre o ajuste do modelo.

Tabela 13 – Comparação da CNN-MLP-v1 variando o número de filtros utilizando o conjunto de treinamento VISTA.

CNN versão 1							
% de Filtros	Loss	Acurácia	Precisão	Recall	F1-Score	Tempo	Ajuste
12.5%	0.19	0.93	0.75	0.65	0.68	650s	Bem ajustado
25.0%	0.19	0.93	0.74	0.65	0.67	670s	Sobre-ajustado
50.0%	0.20	0.93	0.74	0.63	0.66	670s	Sobre-ajustado
100.0%	0.24	0.92	0.74	0.62	0.65	770s	Sobre-ajustado

Fonte: Dados da pesquisa.

Tabela 14 – Comparação da CNN-MLP-v2 variando o número de filtros utilizando o conjunto de treinamento VISTA.

CNN versão 2							
% de Filtros	Loss	Acurácia	Precisão	Recall	F1-Score	Tempo	Ajuste
12.5%	0.19	0.93	0.77	0.63	0.69	1090s	Sobre-ajustado
25.0%	0.19	0.93	0.76	0.66	0.70	1090s	Sobre-ajustado
50.0%	0.19	0.93	0.75	0.65	0.69	1200s	Sobre-ajustado
100.0%	0.22	0.92	0.73	0.65	0.68	1500s	Sobre-ajustado

Fonte: Dados da pesquisa.

5.3.2.2.1 Resultados para o Par CNN-CatBoost no Conjunto VISTA

Foi escolhido o CNN-MLP-v1 com 12,5% de filtros, que demonstrou o melhor comportamento dentre os testados para a análise do impacto das mensagens em comparação ao conjunto BMF. Para isso, foram realizados os experimentos sem e com as mensagens, denominados VISTA-MSG0 e VISTA-MSG1, respectivamente.

A Tabela 15 mostra as métricas obtidas para os modelos CNN-MLP e CNN-CatBoost com cada uma das opções VISTA-MSG0 e VISTA-MSG1 para estes modelos. As Figura 31 e

Figura 32 mostram o impacto de cada variável no conjunto de teste obtido com a ferramenta SHAP.

Tabela 15 – Métricas para os modelos CNN-MLP-v1 e CNN-CatBoost-v1 com 25% dos filtros considerando as opções BMF-MSG0 e BMF-MSG1.

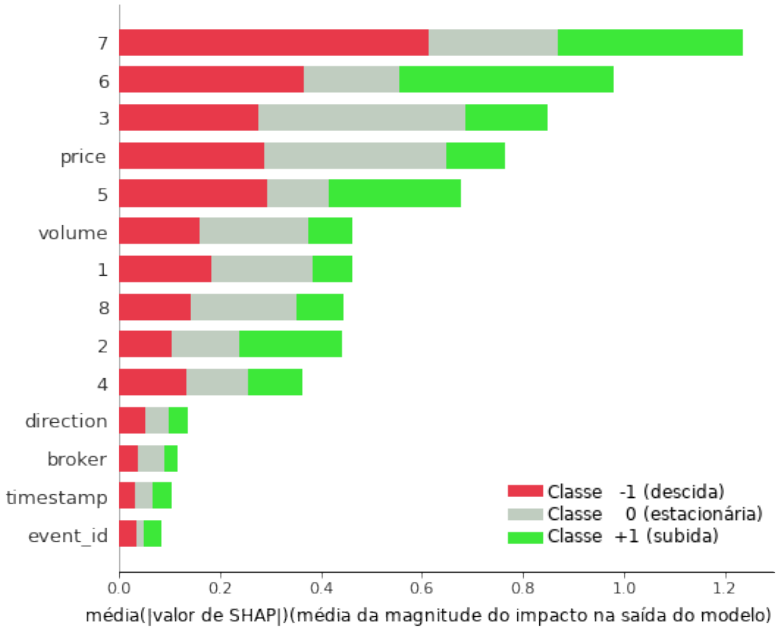
CNN versão 1 com 12,5% dos filtros						
Classificador	Conjunto	Loss	Acurácia	Precisão	Recall	F1-Score
MLP	original	0.193	0.928	0.754	0.635	0.682
CatBoost	BMF-MSG0	0.200	0.929	0.751	0.660	0.699
CatBoost	BMF-MSG1	0.191	0.928	0.744	0.663	0.698

Fonte: Dados da pesquisa.

A Tabela 15 mostra que a adição do classificador *CatBoost* tem um impacto insignificante na performance do modelo, assim como a adição das mensagens (opção VISTA-MSG1), entretanto há um leve aumento do *recall* em detrimento da precisão.

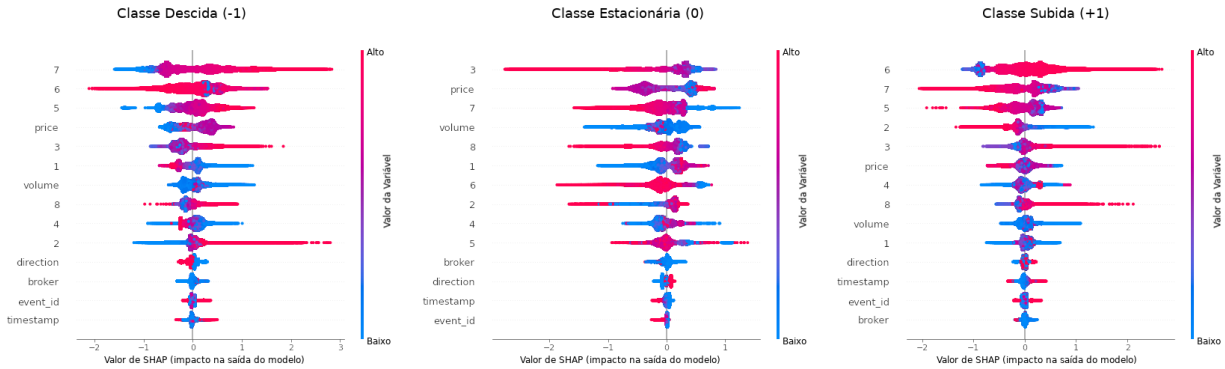
A Figura 31 mostra um resumo geral do impacto de cada variável na saída do modelo. Diferente da Figura 29, nota-se uma relevância maior da classe estacionária (pelas barras cinzas maiores), o que pode ser associado ao número maior de eventos classificados como estacionário. Entretanto, a ordem de relevância geral das variáveis se assemelha ao conjunto BMF, em que aquelas obtidas pela parte do modelo da CNN são as mais relevantes, e dentre as variáveis de mensagens, preço e volume se mantém com forte impacto.

Figura 31 – O valor médio de SHAP para cada variável usada pelo classificador para cada uma das classes.



Fonte: Elaborada pelo autor.

Figura 32 – O valor SHAP dos dados de entrada do classificador para cada uma das classes.



Fonte: Elaborada pelo autor.

Para comparação do comportamento em VISTA, é apresentada uma análise considerando as 3 primeiras variáveis mais relevantes de cada uma das classes, que são: (7, 6 e 5) para *descida*, (3, 7 e 8) para *estacionária* e (6, 7 e 5) para *subida*. Na classe *descida*, as variáveis 7 e 5 tem um forte impacto positivo na classificação e a variável 6 se comporta de forma oposta, tendo um forte impacto negativo. Na classe *estacionária*, as variáveis 3, 7 e 8 têm um forte impacto negativo na classificação e destacando-se a 3. Na classe *subida*, a variável 6 tem forte impacto positivo, o que colabora com a análise da variável 6 para a classe *descida* onde a influência é na direção oposta. Os valores da variável 7 se comportam de maneira oposta aos de 6, porém seu impacto positivo é menor para valores mais baixos. Finalmente, a variável 5 tem um forte impacto negativo, mas para poucos exemplos.

Para as variáveis das mensagens foi analisado o impacto das mesmas do conjunto BMF: preço, volume e corretor. Na classe *descida*, a variável volume, assim como para BMF, têm uma distribuição equilibrada com valores baixos resultando em impacto tanto positivo, como negativo. O preço, tem seus valores médios impactando positivamente, enquanto os valores mais baixos impactam negativamente. A variável corretor tem menor influência se comparada a BMF e esta é dada por valores baixos. Para a classe *estacionária*, a variável preço tem pouca influencia negativa para valores médios, entretanto os mais altos e mais baixos influenciam positivamente, enquanto a variável volume só impacta o modelo se for baixa, porém afeta tanto negativa quanto positivamente. A variável corretor tem influência tanto positiva, quanto negativa para valores baixos. Para a classe *subida*, a variável preço tem pouca influência com os valores medianos, entretanto para valores altos tem um impacto negativo e para valores baixos um impacto positivo. A variável volume tem alto impacto positivo para os valores mais baixos e a variável corretor tem um impacto marginal.

5.3.3 Resultados com Intercâmbio de Bases de Dados

Para validar as sugestões de Zhang, Zohren e Roberts (2019), Mäkinen *et al.* (2018), Sirignano (2016), de que há características universais nos dados, foram feitos testes intercambiando os modelos de extratores e bases de dados, com o treinamento apenas do classificador *CatBoost*. A Tabela 16 mostra os resultados de testes realizados com diferentes extratores e diferentes bases, visando explorar o impacto dos conjuntos de dados, horizonte de previsão e quantidade de dados.

Tabela 16 – Testes variados com intercâmbio de bases e modelos.

Teste	Extrator	Parâmetros do Extrator	Classificador	Parâmetros do Classificador	F1-Score
1	DI1F20 de 10 dias	k=10 com msg	DI1F20 de 32 dias	k=100 com msg	0.37
2	FI-2010 de 10 dias	k=10 sem msg	DI1F20 de 32 dias	k=10 sem msg	0.50
3	FI-2010 de 10 dias	k=10 sem msg	DI1F20 de 32 dias	k=10 com msg	0.53
4	BMF de 10 dias	k=10 com msg	DI1F20 de 32 dias	k=10 com msg	0.73
5	BMF de 32 dias	k=10 com msg	DI1F20 de 10 dias	k=10 com msg	0.75

Fonte: Dados da pesquisa.

Da Tabela 16, nota-se que é possível utilizar um extrator treinado em diferentes tipos de dados e utilizá-lo em outra base (transferência de aprendizado) com um tempo de treinamento muito menor do que o tempo de treinamento de uma RNA a partir do início. Do teste 1, nota-se que um extrator treinado em uma base de dados de 1 instrumento com poucos dias e com horizonte de previsão menor, gera um resultado ruim ao testar numa base de 1 instrumento com mais dias e horizonte de previsão maior. Do teste 2, nota-se que utilizando uma base de dados de outro mercado, mesmo com menos dias, mantendo-se o mesmo horizonte de previsão, pode gerar um resultado melhor do que o teste anterior. Do teste 3, nota-se que apenas adicionando as informações de mensagens, consegue-se uma leve melhora, corroborando com os resultados em BMF e VISTA. Do teste 4, nota-se que um extrator treinado em uma base de dados de 5 instrumentos com poucos dias, gera um resultado muito bom ao testar numa base de 1 instrumento com mais dias, mantendo-se o mesmo horizonte de previsão. Do teste 5, nota-se que um extrator treinado em uma base de dados de 5 instrumentos com mais dias, gera um resultado superior ao testar numa base de 1 instrumento com menos dias, mantendo-se o mesmo horizonte de previsão.

Estes testes mostram a viabilidade de utilizar modelos treinados em diferentes bases de dados para especializar um classificador apenas em um instrumento, além de corroborar com a relevância do uso das informações de mensagens.

5.4 Considerações Finais

Neste capítulo foram apresentados os os resultados relacionados a este projeto de mestrado. Nas seções apresentadas foram descritos os resultados obtidos a partir dos experimentos propostos utilizando as bases de dados finlandesa e brasileira, em que a análise para cada conjunto de resultados respondeu as perguntas que geraram os experimentos.

CAPÍTULO

6

CONCLUSÕES

Este trabalho propôs um modelo de previsão do movimento de preço médio combinando uma rede neural com um algoritmo de árvore de decisão para analisar a melhora do resultado e possibilitar a análise do impacto dos eventos contidos em variáveis do arquivo de mensagens, que não são comumente utilizados em trabalhos relacionados.

A partir dos resultados obtidos, verifica-se que o modelo é capaz de extrair informações relevantes do LOB independentemente do país em que o mercado financeiro está localizado e que um classificador mais simples utilizando MLP ao invés de LSTM pode reduzir significativamente o tempo de treinamento enquanto mantém um desempenho de previsão semelhante.

Foi constatada a importância de uma normalização eficiente, como sugere Passalis *et al.* (2020), para diminuir a variação entre as distribuições de classes diárias dos instrumentos financeiros, principalmente quando os mesmos não tem um relacionamento direto.

A utilização de um modelo híbrido com um classificador baseado em árvore de decisão como o *CatBoost* pode aumentar a precisão dos resultados e as informações das mensagens trouxeram recursos que podem melhorar a precisão do modelo em até 8%. Além disso, a transferência de aprendizado de uma base a outra pode ser feita em um tempo muito menor de treinamento em relação a uma RNA. Contudo, o tempo de inferência pode ser até 50% maior em relação a uma RNA, considerando a implementação em GPU.

A análise utilizando a ferramenta SHAP mostrou a importância do extrator e de como as informações extraídas por ele impactam a saída do modelo. Assim como as informações do arquivo de mensagens que apesar de menor impacto, demonstram que podem melhorar o resultado da classificação. As informações específicas do corretor, com base na relevância da variável, não torna possível confirmar a suposição de Duong *et al.* (2018) de que o não anonimato dos dados do corretor poderia melhorar a inferência, pois os resultados obtidos neste trabalho mostram que tal informação tem um pequeno impacto na classificação final.

A solução apresentada nesta pesquisa atenua o problema de tempo de treinamento e pode ser vista como um modelo moldado à complexidade dos dados. Essas considerações tornaram o modelo implantável em um ambiente de operação real, onde o equilíbrio entre o tempo de treinamento e o desempenho de previsão precisa ser levado em consideração.

6.1 Trabalhos Futuros

Como trabalhos futuros, alguns pontos podem ser explorados, tanto sobre os dados quanto sobre o modelo. Sobre os dados, pode-se analisar o impacto de um volume maior de dados e instrumentos, a fim de gerar um extrator mais robusto e viável para utilização geral. A normalização adaptativa (PASSALIS *et al.*, 2020) pode ser aplicada para melhorar os resultados, ou até ser proposta uma nova abordagem, tanto para normalização quanto rotulação dos dados, a fim de ajustar melhor a distribuição de classes. Sobre o modelo, pode-se repensar na estrutura da RNA para integrar os dados de mensagens diretamente ao modelo. A implementação em *hardware* também pode ajudar a acelerar o tempo de predição, sendo possível analisar uma implementação que dos dois algoritmos (RNA e *CatBoost*) que reduzam o tempo de inferência. Outras propostas de extratores também podem aumentar o desempenho do modelo, por exemplo adicionando camadas de atenção (MÄKINEN *et al.*, 2019; TRAN *et al.*, 2018). E por fim, testar os modelos em um ambiente de estratégia para definir valores viáveis dos parâmetros de normalização e classificação dos dados.

REFERÊNCIAS

ALDRIDGE, I. **High-frequency trading: a practical guide to algorithmic strategies and trading systems**. [S.l.]: John Wiley and Sons, 2009. v. 459. Citado na página 23.

ALMEIDA, T. Uma metodologia de reconhecimento de caracteres manuscritos utilizando redes neurais embarcadas. In: . [S.l.: s.n.], 2014. Citado na página 29.

BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. **IEEE transactions on neural networks**, IEEE, v. 5, n. 2, p. 157–166, 1994. Citado na página 31.

BOROVYKH, A.; BOHTE, S.; OOSTERLEE, C. W. Conditional time series forecasting with convolutional neural networks. **arXiv preprint arXiv:1703.04691**, 2017. Citado na página 24.

BOUCHAUD, J.-P.; MÉZARD, M.; POTTERS, M. *et al.* Statistical properties of stock order books: empirical results and models. **Quantitative finance**, Taylor & Francis, v. 2, n. 4, p. 251–256, 2002. Citado na página 23.

BUKH, P. N. D. **The art of computer systems performance analysis, techniques for experimental design, measurement, simulation and modeling**. [S.l.]: JSTOR, 1992. Citado na página 40.

CAO, G.; WANG, S.; WEI, B.; YIN, Y.; YANG, G. A hybrid CNN-RF method for electron microscopy images segmentation. **Tissue Engineering, J. Biomim Biomater Tissue Eng**, v. 18, p. 2, 2013. Citado nas páginas 24 e 57.

CASTRO, U. C. d. **Derivação de modelos de trading de alta frequência em juros utilizando aprendizado por reforço**. 2017. Citado nas páginas 35 e 50.

DIXON, M. Sequence classification of the limit order book using recurrent neural networks. **Journal of Computational Science**, Elsevier, 2017. Citado na página 24.

DOERING, J.; FAIRBANK, M.; MARKOSE, S. Convolutional neural networks applied to high-frequency market microstructure forecasting. In: IEEE. **Computer Science and Electronic Engineering (CEEC), 2017**. [S.l.], 2017. p. 31–36. Citado nas páginas 24, 45 e 48.

DOROGUSH, A. V.; ERSHOV, V.; GULIN, A. Catboost: gradient boosting with categorical features support. **arXiv preprint arXiv:1810.11363**, 2018. Citado na página 33.

DUONG, H. N.; LAJBCYGIER, P.; LU, J. S. *et al.* The effect of anonymity on price efficiency: Evidence from the removal of broker identities. **Pacific-Basin Finance Journal**, Elsevier, v. 51, p. 95–107, 2018. Citado na página 75.

GOULD, M. D.; PORTER, M. A.; WILLIAMS, S.; MCDONALD, M.; FENN, D. J.; HOWISON, S. D. Limit order books. **Quantitative Finance**, Taylor & Francis, v. 13, n. 11, p. 1709–1742, 2013. Citado na página 23.

HAYKIN, S. S.; HAYKIN, S. S.; HAYKIN, S. S.; HAYKIN, S. S. **Neural networks and learning machines**. [S.l.]: Pearson Upper Saddle River, NJ, USA:, 2009. v. 3. Citado na página 29.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Citado na página 31.

HUANG, R.; POLAK, T. Lobster: Limit order book reconstruction system. **Available at SSRN 1977207**, 2011. Citado na página 38.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. **The Journal of physiology**, Wiley-Blackwell, v. 160, n. 1, p. 106, 1962. Citado na página 29.

Jhaveri, S.; Khedkar, I.; Kantharia, Y.; Jaswal, S. Success prediction using Random Forest, CatBoost, XGBoost and AdaBoost for kickstarter campaigns. In: **2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)**. [S.l.: s.n.], 2019. p. 1170–1173. Citado na página 33.

KAYA, O.; SCHILDBACH, J.; AG, D. B. High-frequency trading. **Reaching the limits, Automated trader magazine**, v. 41, p. 23–27, 2016. Citado na página 23.

KERCHEVAL, A. N.; ZHANG, Y. Modelling high-frequency limit order book dynamics with support vector machines. **Quantitative Finance**, Taylor & Francis, v. 15, n. 8, p. 1315–1329, 2015. Citado nas páginas 24, 44, 47, 48 e 50.

LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; JACKEL, L. D. Backpropagation applied to handwritten zip code recognition. **Neural computation**, MIT Press, v. 1, n. 4, p. 541–551, 1989. Citado na página 29.

LOBSTER. **LOBSTER - output.** 2020. Disponível em: <https://lobsterdata.com/info/DataStructure.php>. Citado na página 38.

LUNDBERG, S. M.; LEE, S.-I. A unified approach to interpreting model predictions. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 30**. Curran Associates, Inc., 2017. p. 4765–4774. Disponível em: <http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf>. Citado nas páginas 40 e 66.

____. **SHAP library**. 2020. Disponível em: <https://github.com/slundberg/shap>. Citado na página 40.

LUO, J.; NING, Z.; ZHANG, S.; FENG, Q.; ZHANG, Y. Bag of deep features for preoperative prediction of sentinel lymph node metastasis in breast cancer. **Physics in Medicine & Biology**, IOP Publishing, v. 63, n. 24, p. 245014, 2018. Citado nas páginas 24 e 57.

MÄKINEN, M.; KANNIAINEN, J.; GABBOUJ, M.; IOSIFIDIS, A. Forecasting of jump arrivals in stock prices: New attention-based network architecture using limit order book data. **arXiv preprint arXiv:1810.10845**, 2018. Citado nas páginas 24, 48, 50, 57 e 72.

MÄKINEN, Y.; KANNIAINEN, J.; GABBOUJ, M.; IOSIFIDIS, A. Forecasting jump arrivals in stock prices: new attention-based network architecture using limit order book data. **Quantitative Finance**, Taylor & Francis, v. 19, n. 12, p. 2033–2050, 2019. Citado nas páginas 46 e 76.

MATHWORKS. **Convolutional Neural Network**. 2018. Disponível em: <https://www.mathworks.com/discovery/convolutional-neural-network.html>. Acesso em: 13/02/2018. Citado na página 30.

MCCLELLAND, J. L.; RUMELHART, D. E.; GROUP, P. R. *et al.* Parallel distributed processing. **Explorations in the Microstructure of Cognition**, MIT Press Cambridge, Ma, v. 2, p. 216–271, 1986. Citado na página 28.

NTAKARIS, A.; MAGRIS, M.; KANNIAINEN, J.; GABBOUJ, M.; IOSIFIDIS, A. Benchmark dataset for mid-price forecasting of limit order book data with machine learning methods. **Journal of Forecasting**, Wiley Online Library, v. 37, n. 8, p. 852–866, 2018. Citado nas páginas 24, 45, 50, 51, 52, 53, 54 e 60.

NTAKARIS, A.; MIRONE, G.; KANNIAINEN, J.; GABBOUJ, M.; IOSIFIDIS, A. Feature engineering for mid-price prediction with deep learning. **Ieee Access**, IEEE, v. 7, p. 82390–82412, 2019. Citado na página 47.

PACHECO, A. **Redes Neurais Convolutivas – CNN**. 2017. Disponível em: <http://www.computacaointeligente.com.br/artigos/redes-neurais-convolutivas-cnn/>. Acesso em: 13/02/2018. Citado na página 31.

PASSALIS, N.; TEFAS, A.; KANNIAINEN, J.; GABBOUJ, M.; IOSIFIDIS, A. Adaptive normalization for forecasting limit order book data using convolutional neural networks. In: IEEE. **ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. [S.l.], 2020. p. 1713–1717. Citado nas páginas 47, 54, 75 e 76.

PERSIO, L. D.; HONCHAR, O. Recurrent neural networks approach to the financial forecast of google assets. **International journal of Mathematics and Computers in simulation**, v. 11, p. 7–13, 2017. Citado nas páginas 45 e 48.

PERUŠ, M. Multi-level synergetic computation in brain. **Nonlinear Phenomena in Complex Systems**, v. 4, n. 2, p. 157–193, 2001. Citado na página 27.

POSKITT, R.; MARSDEN, A.; NGUYEN, N.; SHEN, J. The introduction of broker anonymity on the new zealand exchange. **Pacific Accounting Review**, Emerald Group Publishing Limited, 2011. Citado nas páginas 24 e 57.

PROKHORENKOVA, L.; GUSEV, G.; VOROBEV, A.; DOROGUSH, A. V.; GULIN, A. Catboost: unbiased boosting with categorical features. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2018. p. 6638–6648. Citado na página 33.

ROTH, A. E. **The Shapley value: essays in honor of Lloyd S. Shapley**. [S.l.]: Cambridge University Press, 1988. Citado na página 40.

SIRIGNANO, J. Deep learning for limit order books. 2016. Citado nas páginas 24, 44, 46, 48, 57 e 72.

SIRIGNANO, J.; CONT, R. Universal features of price formation in financial markets: perspectives from deep learning. **Quantitative Finance**, Taylor & Francis, v. 19, n. 9, p. 1449–1459, 2019. Citado na página 46.

STATHAKIS, D. How many hidden layers and nodes? **International Journal of Remote Sensing**, Taylor & Francis, v. 30, n. 8, p. 2133–2147, 2009. Citado na página 63.

SZEGEDY, C.; VANHOUCKE, V.; IOFFE, S.; SHLENS, J.; WOJNA, Z. Rethinking the inception architecture for computer vision. In: **Proceedings of the IEEE conference on computer vision and pattern recognition**. [S.l.: s.n.], 2016. p. 2818–2826. Citado nas páginas 46 e 55.

TRAN, D. T.; IOSIFIDIS, A.; KANNIAINEN, J.; GABBOUJ, M. Temporal attention-augmented bilinear network for financial time-series data analysis. **IEEE transactions on neural networks and learning systems**, IEEE, 2018. Citado nas páginas 46, 48 e 76.

TSANTEKIDIS, A.; PASSALIS, N.; TEFAS, A.; KANNIAINEN, J.; GABBOUJ, M.; IOSIFIDIS, A. Forecasting stock prices from the limit order book using convolutional neural networks. In: IEEE. **Business Informatics (CBI), 2017 IEEE 19th Conference on**. [S.l.], 2017. v. 1, p. 7–12. Citado nas páginas 24, 45, 46, 47, 48, 50, 56 e 58.

____. Using deep learning to detect price change indications in financial markets. In: IEEE. **Signal Processing Conference (EUSIPCO), 2017 25th European**. [S.l.], 2017. p. 2511–2515. Citado nas páginas 45, 46, 47, 48 e 50.

____. Using deep learning for price prediction by exploiting stationary limit order book features. **Applied Soft Computing**, Elsevier, p. 106401, 2020. Citado na página 47.

WIKIPEDIA. **Long short-term memory**. 2018. Disponível em: <https://en.wikipedia.org/wiki/Long_short-term_memory>. Acesso em: 13/02/2018. Citado na página 32.

YANDEX. **CatBoost - open-source gradient boosting library**. 2020. Disponível em: <https://catboost.ai/#benchmark>. Citado na página 34.

ZHANG, Z.; ZOHREN, S.; ROBERTS, S. Deeplob: Deep convolutional neural networks for limit order books. **IEEE Transactions on Signal Processing**, IEEE, 2019. Citado nas páginas 24, 46, 48, 50, 52, 55, 57, 61 e 72.

APÊNDICE

A

TESTES COMPLETOS DE EXTRATORES

Figura 33 – Métricas obtidas para os modelos com variações de quantidade de dias, horizontes de previsão, filtros, classificadores e versões de extrator.

Base de Dados	Dias	Horizonte k	% classe estacionária	Filtros	Com LSTM	Versão	Loss	Acurácia	Precisão	Recall	F1-Score
DIF20	10	10	80%	12,50%	Sim	1	0,390350258	0,853476286	0,85743916	0,795241117	0,818879902
						2	0,630673767	0,805316687	0,805305362	0,805305362	0,805305362
					Não	1	0,381154074	0,857995033	0,870392263	0,825374007	0,844589233
						2	0,380600189	0,855890274	0,866088808	0,785516024	0,81377548
				25%	Sim	1	0,361248491	0,886942625	0,888080478	0,883106172	0,885511041
						2	0,389250129	0,848359108	0,862440884	0,797415555	0,821662188
					Não	1	0,358967891	**0,888857841**	**0,890730977**	**0,884243309**	**0,88728863**
						2	**0,321770885**	0,883261859	0,889411151	0,872792602	0,879771292
				50%	Sim	1	0,345732985	0,863541126	0,86743933	0,851407349	0,857235253
						2	0,334284938	0,886084795	0,889858782	0,879425645	0,884118617
					Não	1	0,351051744	0,883590996	0,885870934	0,877809763	0,881574333
						2	0,342910453	0,863830447	0,8662076	0,857960582	0,861846864
				100%	Sim	2	0,349997728	0,879760623	0,881238163	0,876094162	0,878571808
					Não	2	0,34191505	0,867990017	0,870417774	0,862718403	0,866217613
		100	70%	12,50%	Sim	1	1,018918792	0,525695741	0,114205956	0,109499678	0,110429481
						2	0,984983859	0,525695741	0,239898279	0,225542605	0,228268608
					Não	1	1,016761823	0,530334175	0,513868153	0,455603659	0,470548332
						2	1,029918231	0,525695741	0,52566427	0,52566427	0,52566427
				25%	Sim	1	1,079298355	0,451750636	0,209747091	0,182243451	0,188653827
						2	1,041100491	0,476099759	0,250461578	0,213014677	0,220785543
					Não	1	**0,933391765**	**0,609466314**	**0,608210683**	**0,553333461**	**0,570102513**
						2	1,014582569	0,530753136	0,506972492	0,441230446	0,459944874
				50%	Sim	1	1,090586491	0,498423934	0,364216536	0,304779768	0,319339871
						2	1,040210675	0,512259364	0,368036568	0,324449942	0,334793508
					Não	1	1,508517648	0,527431428	0,528607547	0,501067281	0,510648966
						2	1,34198247	0,485995024	0,486786485	0,457249433	0,467072338
				100%	Sim	2	1,060473078	0,469925195	0,357646734	0,285399377	0,301741123
					Não	2	1,458910942	0,477705747	0,474262089	0,448661417	0,45753324
	32	10	80%	12,50%	Sim	1	0,34476869	0,882727206	0,889364898	0,858110607	0,871473372
						2	0,54436625	0,843560994	0,843537569	0,843537569	0,843537569
					Não	1	0,357659567	0,873541176	0,885697782	0,833247781	0,852937162
						2	0,295845319	0,903278708	0,905225098	0,898182213	0,90152818
				25%	Sim	1	0,373542127	0,871985018	0,88121748	0,848920882	0,861532211
						2	**0,274621769**	0,907219172	0,908250272	0,904745102	0,906353891
					Não	1	0,297935696	**0,916074753**	0,917745352	**0,912763596**	**0,915021658**
						2	0,283784201	0,910682738	**0,912523687**	0,90679884	0,90942955
				50%	Sim	1	0,376698272	0,86124289	0,863696694	0,856366336	0,859538078
						2	0,278133997	0,892954826	0,893981636	0,889017642	0,891004324
					Não	1	0,302056171	0,899756551	0,90084213	0,897111416	0,898815334
						2	0,494653704	0,789440215	0,792901278	0,778394043	0,784336507
				100%	Sim	2	0,265689439	0,908741832	0,910001695	0,90563184	0,907435358
					Não	2	0,287272782	0,898673117	0,900664449	0,891857743	0,895499468
		100	70%	12,50%	Sim	1	0,781871414	0,710300446	0,709124506	0,695564508	0,700069249
						2	0,850138141	0,644948065	0,568446338	0,537076712	0,544042051
					Não	1	0,87225605	0,742301047	0,74407655	0,730813324	0,73566854
						2	0,846996427	0,701737642	0,696002424	0,67590934	0,682358384
				25%	Sim	1	0,850945079	0,680956066	0,625129282	0,610502303	0,613862872
						2	0,793351788	0,704992115	0,706162274	0,690883934	0,696495593
					Não	1	0,823352974	0,746362805	0,747148931	0,735899627	0,739724457
						2	**0,760354616**	**0,73664552**	**0,738016844**	**0,727362454**	**0,731103063**
				50%	Sim	1	0,853216577	0,670176268	0,64237833	0,639869988	0,640408993
						2	0,884915292	0,671983361	0,671944022	0,671944022	0,671944022
					Não	1	0,917396854	0,704360425	0,704962611	0,696171045	0,69929117
						2	0,809479349	0,745074391	0,744715631	0,737798631	0,740132511
				100%	Sim	2	0,999095598	0,640647888	0,637400746	0,624807596	0,628919899
					Não	2	1,101012328	0,666122854	0,66639322	0,66129452	0,663255394
ABEV3	10	10	90%	12,50%	Sim	1	0,530251597	**0,820853651**	**0,832008183**	0,794679463	0,810885429
						2	0,599793451	0,780456066	0,807588518	0,68615067	0,7278983
					Não	1	0,564442185	0,789027274	0,807765841	0,711288452	0,744621277
						2	0,549096934	0,795146406	0,811886191	0,722112179	0,75383997
				25%	Sim	1	0,53037566	0,816013515	0,824223816	0,797227561	0,809194624
						2	0,561225385	0,790283799	0,808246434	0,697535276	0,735975325
					Não	1	0,61797414	0,804852307	0,815066636	0,777772427	0,793593526
						2	0,557059618	0,788799703	0,810827971	0,72238785	0,753757834
				50%	Sim	1	**0,514738433**	0,820228636	0,825458884	**0,806124985**	**0,814792633**
						2	0,551426892	0,789283693	0,797405839	0,767875016	0,781006515
					Não	1	0,551622744	0,800759017	0,811256766	0,772499979	0,788663983
						2	0,597839145	0,756672284	0,767984271	0,735230791	0,750191212
				100%	Sim	2	0,539889231	0,795918882	0,804809668	0,772006392	0,785778046
					Não	2	0,574957288	0,777667344	0,783874333	0,759192288	0,770023644
		100	70%	12,50%	Sim	1	0,545091477	0,800153196	0,724149287	0,700842977	0,706433117
						2	0,597027134	0,82344681	0,823378205	0,823378205	0,823378205
					Não	1	0,543185498	0,816177011	0,767005384	0,735358953	0,743580103
						2	0,452948201	0,857590675	0,799578726	0,769144237	0,777454495
				25%	Sim	1	0,569237342	0,815532684	0,784766555	0,755852518	0,763326526
						2	0,560440914	0,814539015	0,73235029	0,702176273	0,709873259
					Não	1	0,480014908	0,827568948	0,800395548	0,765557706	0,774649858
						2	0,419689188	0,855705917	0,828018963	0,794596136	0,802941263
				50%	Sim	1	0,624743341	0,799544215	0,742207706	0,708676279	0,717025638
						2	0,729749515	0,728352368	0,574524462	0,499842942	0,516657233
					Não	1	0,763890435	0,780667603	0,774973935	0,758903861	0,76410234
						2	0,565693638	0,783523619	0,776235376	0,750519216	0,759137775
				100%	Sim	2	0,768264277	0,667440236	0,662602484	0,640012801	0,647881866
					Não	2	0,595552005	0,781010568	0,778932571	0,76453203	0,769486248

LEGENDA
Bom ajuste
Subajuste
Sobreajuste

Fonte: Elaborada pelo autor.

ICMC USP
SÃO CARLOS