UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE TECNOLOGIA

ARTHUR BRIGANTI GINI

AVALIAÇÃO DE MÉTODOS PREDITIVOS APLICADOS AO ÍNDICE IBOVESPA

LIMEIRA
2022

ARTHUR BRIGANTI GINI

AVALIAÇÃO DE MÉTODOS PREDITIVOS APLICADOS AO ÍNDICE IBOVESPA

Monografia apresentada à Faculdade de Tecnologia da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Guilherme Palermo Coelho

Este exemplar corresponde à versão final da monografia defendida por Arthur Briganti Gini e orientada pelo Prof. Dr. Guilherme Palermo Coelho.

LIMEIRA
2022

FOLHA DE APROVAÇÃO

Abaixo se apresentam os membros da comissão julgadora desta monografia para o Título de Bacharel em Sistemas de Informação, a que submeteu o aluno Arthur Briganti Gini, na Faculdade de Tecnologia – FT/UNICAMP, em Limeira/SP.

BANCA EXAMINADORA

Prof. Dr. Guilherme Palermo Coelho
FT/UNICAMP - Presidente da Comissão Julgadora

Prof. Dr João Roberto Bertini Júnior
FT/UNICAMP

Prof. Dr. Ulísses Martins Dias
FT/UNICAMP

A ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Trabalhos de Conclusão de Curso e na Secretaria de Graduação da FT.

” O autoconhecimento é o começo da sabedoria, em cuja tranquilidade e silêncio encontra o imensurável.”

(Jiddu Krishnamurti)

AGRADECIMENTOS

Agradeço primeiramente a minha família que sempre me desafiou dentro de casa, me levando a dedicação integral aos estudos. Graças a ETEC Professora Anna De Oliveira Ferraz e o cursinho Geração NEAR da UNESP, pude me dedicar manhã, tarde e noite para que meu sonho de fazer faculdade e me tornar independente fosse realizado.

Gratidão a UNICAMP por acreditar no meu potencial e proporcionar oportunidades incríveis de crescimento pessoal e profissional. Agradeço principalmente ao SAE, por me apoiar financeiramente durante os estudos, sem esse suporte eu não conseguiria finalizar a graduação.

Agradeço aos meus Professores e Orientadores Guilherme Palermo e Ana Estela por me desafiar constantemente e me acolher no HIGHPIDS, essa oportunidade foi essencial para que eu junto com meus amigos, Larissa Benevides, Gabriel Domingues e Juliana Morroni, pudéssemos focar todos os períodos do dia nos estudos da universidade.

Agradeço especialmente aos meus amigos Matheus Cumpian e Arthur Guedes, por realizarem projetos incríveis comigo. Graças a essa união, conseguimos criar inovações promissoras que fazem acreditarem cada vez mais em nós.

Gostaria de agradecer a todas as repúblicas em que morei, pois fazer parte de tantas famílias maravilhosas foi parte essencial para o crescimento incrível que tive durante esses 5 anos.

Graças a toda dedicação e apoio durante anos, me encontro hoje no melhor momento da minha vida, com trabalho e pessoas ao meu redor que me orgulham muito, sinto também que isso é só o começo, nós somos o próximo mundo!

Por fim, agradeço a você, que dedicou tempo para ler meu trabalho e agora conhece um pouco da minha história, espero que goste do conteúdo. Fique à vontade para enviar qualquer comentário pelas minhas redes sociais.

RESUMO

A quantidade de investidores no mercado brasileiro de ações tem crescido em um ritmo acelerado nos últimos anos. Dado esse cenário, é cada vez mais relevante o desenvolvimento de ferramentas e metodologias de suporte à decisão. Tendo isso em vista, o presente trabalho avaliou diferentes metodologias preditivas a fim de identificar quais métodos de Aprendizado de Máquina conseguem prever com menor erro o índice IBOVESPA. Para isso, utilizamos os dados públicos do índice e das ações mais relevantes que o compõem, organizando-os de forma que a entrada do modelo preditivo fossem os valores de cotação de uma semana para predizer a variação do IBOVESPA no início da semana seguinte. Para treinamento dos modelos utilizamos a biblioteca KT (*Keras Tuner*), otimizando os hiper parâmetros de forma automática e independente para cada modelo. Como método de avaliação calculamos as métricas de MAPE e RMSE, dispondo de suas médias e desvio-padrão. Concluímos que, dentre os 30 modelos gerados no trabalho, utilizando as arquiteturas de Redes Neurais Artificias (RNA), MLP (Multilayer Perceptron), CNN (do inglês, *Convolution Neural Network*) e LSTM (*Long Short-Term Memory*), o modelo que leva aos menores erros de predição é a LSTM.

Palavras-chave: Redes Neurais Artificiais, IBOVESPA, MLP, CNN, LSTM.

ABSTRACT

The number of investors in the Brazilian stock market has grown at a rapid pace in recent years. Given this scenario, the development of decision support tools and methodologies is increasingly relevant. With that in mind, the present work evaluated different predictive methodologies in order to identify which Machine Learning methods can predict the IBOVESPA index with less error. For this, we use the most relevant stocks from the index, and organize the stock price of a week as the input of each model to predict the opening price of the IBOVESPA at the beginning of the following week. For training, we used the KT library (Keras Tuner), optimizing the hyperparameters of the models automatically and independently for each model. To evaluate each forecasting model, we used MAPE and RMSE. We conclude that, among the 30 models generated in this work using the MLP (Multilayer Perceptron), CNN (Convolution Neural Network), and LSTM (Long Short-Term Memory) architectures, the model that leads to the smallest prediction errors is the LSTM.

Keywords: Artificial Neural Networks, IBOVESPA, MLP, CNN, LSTM.

LISTA DE TABELAS:

LISTA DE FIGURAS

LISTA DE SIGLAS E ABREVIAÇÕES

TI	Tecnologia da Informação
TCC	Trabalho de Conclusão de Curso
ID	Identificador
RNA	Rede Neural Artificial
KT	*Keras Tuner*
RMSE	*Root Mean Squared Error*
MAPE	*Mean Absolute Percentage Error*
RNN	*Recurrent Neural Network*
LSTM	*Long Short-Term Memory*
B3	Brasil Bolsa Balcão
MLP	*Multilayer Perceptron*
MSE	*Mean Squared Error*
MAE	*Mean Absolute Error*
CNN	*Convolutional Neural Network*
API	*Application Programming Interface*
QTD	Quantidade

SUMÁRIO

1. INTRODUÇÃO

O valor das ações no mercado é um tema que constantemente tem recebido atenção da comunidade científica, principalmente se tratando da previsão desses valores. Considerando diversos métodos preditivos, foram encontrados melhores resultados utilizando-se Aprendizado de Máquina comparado com modelos estatísticos clássicos (Coelho et al., 2008).

Para compreender os preços do mercado de ações, segundo a teoria de Dow (Rhea, 1993) os movimentos são tendenciais. Desta forma, os investidores desenvolveram duas estratégias principais, a Análise Técnica e a Fundamentalista. Enquanto a primeira utiliza o preço passado e indicadores técnicos, o segundo se baseia nas condições financeiras como balanços patrimoniais e indicadores macroeconômicos para extrair a informação se determinada ação pode ter seu preço apreciado ou depreciado.

Como demonstra a literatura, a análise técnica vem sendo muito utilizada em conjunto com diferentes arquiteturas de Redes Neurais Artificiais (RNAs), como a Rede Neural do tipo Perceptron Multicamadas (do inglês, *Multilayer Perceptrons* – MLP), Redes Neurais Convolucionais (do inglês, *Convolution Neural Network* - CNN) e Redes Neurais Recorrentes (do inglês, *Recorrent Neural Network* – RNN) (Mirete-Ferrer et al., 2022).

Tendo em vista a revisão da literatura de (Ozbayoglu, Gudelek, Sezer, 2020), (Bustos, Pomares-Quimbaya, 2020) e (Kumbure et al., 2022), a maioria dos trabalhos científicos utilizam dados dos mercados de ações dos Estados Unidos e asiático. Diante disso, neste trabalho pretendemos contribuir para a literatura com o estudo de RNAs aplicadas ao mercado brasileiro de ações.

Desta forma, propomos aqui aplicar técnicas de predição com o objetivo de antecipar a variação da abertura do mercado brasileiro, através do índice IBOVESPA. Para isso, três sistemas foram criados, aplicando diferentes arquiteturas de Redes Neurais Artificiais (RNAs), buscando identificar qual arquitetura leva às melhores diferenças entre o valor real e o predito.

[1]https://keras.io/keras_tuner/

Utilizamos um conjunto de dados coletados entre junho de 2010 e novembro de 2022, para algumas ações brasileiras: Ambev (ABEV3), JBS (JBSAY), Petrobras (PETR4) e Vale (VALE3). Além disso, utilizamos também a cotação do dólar frente ao real. O primeiro passo para tratar a base de dados foi a normalização das cotações, para isso, calculamos a variação do valor de abertura de um dia para o outro. O segundo passo foi a separação dos dados em 80% para treinamento e 20% para os testes. Para transformarmos as séries temporais em dados de entrada para os modelos, geramos uma janela nos dados de 6 dias, dos quais os 5 primeiros são utilizados como entrada e o último dia é o objetivo da predição.

Com a base de dados pronta, treinamos 10 Redes Neurais Artificiais (RNAs) de cada uma das três arquiteturas estudadas. Para otimizar os hiper-parâmetros, utilizamos o *framework Keras Tuner*[1] (KT), aplicando o método *Hyperband*. Compilamos então todos os parâmetros ótimos, em busca de minimizar os efeitos das configurações internas das redes na realização das predições.

Para fim de avaliação das trinta Redes Neurais Artificiais (RNAs) geradas, utilizamos como métricas de erro o *Root Mean Squared Error* (RMSE) (em português, *Erro Quadrático Médio)* e o *Mean Absolute Percentage Error* (MAPE) (em português, Erro Percentual Absoluto Médio), sendo que o primeiro indica a média do quanto o valor da predição pode variar para mais ou menos do valor real, enquanto o segundo demonstra a porcentagem média do erro predito em relação ao valor real.

Nossos resultados mostram que, na média, a rede do tipo *Recurrent Neural Network* (RNN), com a arquitetura *Long Short-Term Memory* (LSTM), se saiu melhor, atingindo os respectivos valores de 30,59% para o *Mean Absolute Percentage Error* (MAPE) e 0,337 para o *Root Mean Squared Error* (RMSE).

Esta monografia está organizada da seguinte forma. No Capítulo 2 será apresentada a Fundamentação Teórica que guiou o desenvolvimento do trabalho. No Capítulo 3 será apresentada a Metodologia Experimental. Os resultados serão apresentados e discutidos no Capítulo 4. Por fim, no Capítulo 5 são feitas as conclusões e indicações para possíveis trabalhos futuros.

2. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo abordaremos as principais áreas do aprendizado de máquina e discutiremos as características das arquiteturas de Redes Neurais Artificiais utilizadas no trabalho.

2.1 Aprendizado de Máquina

O aprendizado de máquina explora o estudo de sistemas que, segundo Simon (2013), busca dar aos computadores a capacidade de aprender sem serem explicitamente programados para isso.

As atividades do aprendizado de máquina podem ser classificadas em 3 diferentes tipos (Russell, 2003), são elas o *aprendizado supervisionado*, no qual o sistema recebe as entradas e saídas desejadas e assim busca aprender uma regra geral que mapeie as relações dos dados; o *aprendizado não supervisionado*, onde o sistema recebe dados sem rótulo e busca sozinho as estruturas e relações existentes entre os dados; e o *aprendizado por reforço* onde o sistema está num ambiente dinâmico que recebe premiações ou punições na medida em que tenta solucionar o problema.

No presente trabalho utilizaremos a abordagem de aprendizado supervisionado para a realização das predições do índice IBOVESPA.

2.2 Redes Neurais Artificiais

As Redes Neurais Artificiais são modelos matemáticos baseados no funcionamento do cérebro humano, tendo sua função adquirida pelo reconhecimento de padrões para depois generalizar o aprendizado.

Uma RNA é um conjunto de vários neurônios artificiais, no qual uma unidade recebe entradas e produz saídas, representando assim não só a comunicação entre neurônios biológicos, mas também o comportamento de dendritos e axônios presentes em tais neurônios.

Em 1958, Rosenblatt (1958) introduziu o modelo de neurônio conhecido como *Perceptron*, que tem como sua principal característica a representação de sinapses por pesos, que podem ser ajustados iterativamente para que o modelo passe a reconhecer padrões.

Tendo isso em vista, podemos também utilizar a saída de um neurônio do tipo *Perceptron* como a entrada de um neurônio seguinte, o que nos leva às RNAs divididas em camadas, tradicionalmente organizadas em camada de entrada, com os *inputs* do modelo, as camadas intermediárias ou ocultas e a camada de saída com o objetivo da rede.

Com o intuito de aprofundar as arquiteturas utilizadas no trabalho, a seguir apresentamos as seguintes Redes Neurais Artificiais (RNAs): MLP (*Multilayer Perceptron), CNN (Convolutional Neural Network) e LSTM (Long Short-Term Memory).*

2.2.1 MLP

As Redes Neurais Artificiais do tipo *Multi-layer Perceptron* são baseadas no *Perceptron* apresentado em 1958 por Rosenblatt (1958), onde a saída de um neurônio está conectada com a entrada de outro do mesmo tipo, criando uma rede organizada em camadas, com ao menos uma camada oculta entre a camada de entrada e saída da rede. Esse tipo de rede é considerado como um aproximador universal de funções (Haykin, 2008).

A Figura 2.1 apresenta uma RNA do tipo MLP com a seguinte configuração: a primeira camada provê os dados utilizados na rede, sendo a entrada; temos também duas camadas em sequência, sendo consideradas intermediárias (camadas ocultas), em que os pesos das sinapses são aplicados às suas entradas e o resultado passa por suas respectivas funções de ativação. Ao final, temos uma camada de saída com um neurônio aplicando também os pesos e funções de ativação, para assim, apresentar como resultado final da saída da Rede Neural Artificial (RNA).

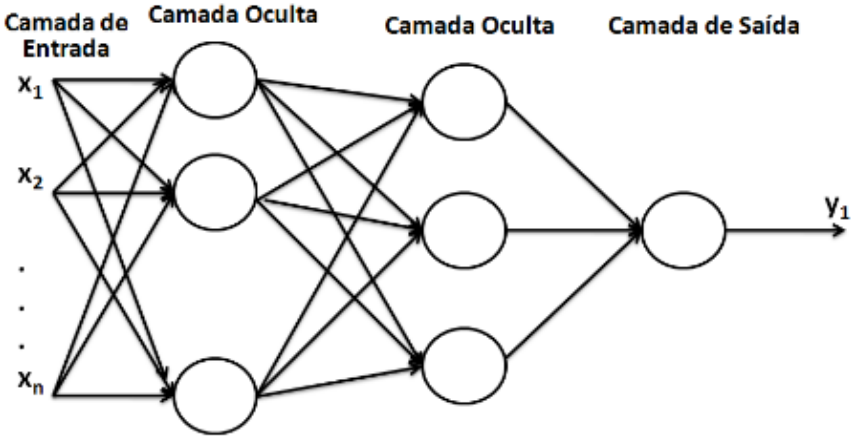

Figura 2.1: RNA do tipo MLP. Adaptado de (Carosia et al., 2019).

Em 1986, Rumelhart et al. (1986) formalizaram o termo *backpropagation* para um dos algoritmos de treinamento de RNAs, capaz de auxiliar muito na assertividade das Redes Neurais Artificiais. Ele é composto de duas fases, a primeira sendo a *Forward* (em português, propagação adiante), onde uma amostra dos dados é passada pela rede produzindo uma saída. Tal saída é comparada com a saída desejada do modelo, gerando um sinal de erro representativo da efetividade da RNA. Na segunda etapa, denominada *Backward* (em português, retropropagação), os neurônios têm seus pesos ajustados utilizando o gradiente da função de erro.

Dessa forma, esse processo utiliza a retropropagação do erro para ajuste dos pesos das sinapses. Esta etapa se baseia na taxa de aprendizado (*Learning Rate*), variando de 1 a próximo de 0, que busca garantir que a mudança dos pesos seja gradual.

Esse processo de treinamento ocorre de forma iterativa, podendo o número de iterações (do inglês, *epoch*) ser pré-determinado ou livre, ou seja, o treinamento é executado até que não haja significativa redução do erro.

Para o cálculo do erro da predição do modelo podemos utilizar diversas métricas que se baseiam na diferença entre o valor real e o predito. Para problemas

de predição geralmente utilizamos o *Mean Squared Error* (MSE) e/ou o *Mean Absolute Error* (MAE).

2.2.2 CNN

As Redes Neurais Convolucionais foram inicialmente desenvolvidas por Fukushima (1975). Assim como as redes convolucionais atuais elas possuíam diversas camadas organizadas hierarquicamente, o que permitia o uso de dados com características espaciais.

Uma RNA com arquitetura CNN é considerada uma rede *feedforward*, recebendo dados via uma série de camadas convolucionais, de subamostragem (p*ooling*) e, por fim, funções de ativação, essas em que suas saídas são entregues a camadas totalmente conectadas similares a uma MLP, para assim, produzir a saída final da rede.

Na Figura 2.2 pode ser vista uma arquitetura genérica de uma Rede Neural Convolucional, que possui uma camada de convolução, gerando um mapa de características servindo de entrada para a camada de *pooling*, que, por sua vez, alimenta a camada totalmente conectada similar a MLP, produzindo o resultado do modelo.

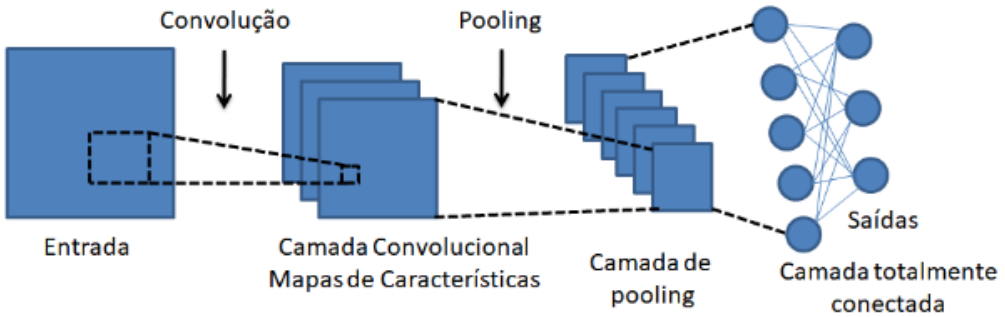

Figura 2.2: Representação de RNA do tipo CNN. Adaptado de (Faria, 2018).

O treinamento da RNA do tipo CNN utiliza o algoritmo *backpropagation*, passando pelas épocas (*epoch*) de forma iterativa com o objetivo de minimizar a diferença entre as saídas geradas pela rede e as saídas conhecidas.

2.2.3 LSTM

As Redes Neurais Artificiais podem também ser organizadas em redes do tipo recorrente, introduzidas por Jordan (1986), esse tipo de rede permite a realimentação das camadas ocultas a partir da sua camada de saída. Esse tipo de arquitetura permite que a rede aprenda a partir do contexto, tendo a percepção de tempo, o que as torna indicadas para predição de séries temporais e processamento de linguagem natural (Lipton, Berkowitz, Elkan, 2015).

As RNN mais simples têm memória de longo prazo na forma de pesos transmitidos através do tempo nas sinapses dos neurônios. Esses pesos são alterados lentamente durante o treinamento. Além dessa memória de longo prazo, as redes possuem memória de curto prazo mantida pelas portas internas da célula.

A arquitetura RNN *Long Short-Term Memory* (LSTM), proposta por Hochreiter & Schmidhuber (1997), tem como seu principal diferencial a substituição dos nós tradicionais das RNAs para uma unidade de computação que possui célula de memória (do inglês, *memory cell*). Tal célula que, por sua vez, gera um terceiro tipo de memória, carregando um tipo intermediário de armazenamento de conhecimento pela célula.

Na Figura 2.3 temos a essência da LSTM, sua célula de memória, composta por portas que funcionam como comportas de dados, controlando o fluxo do dado pelo neurônio e definindo o que a rede deve guardar e o que deve esquecer ao longo do tempo.

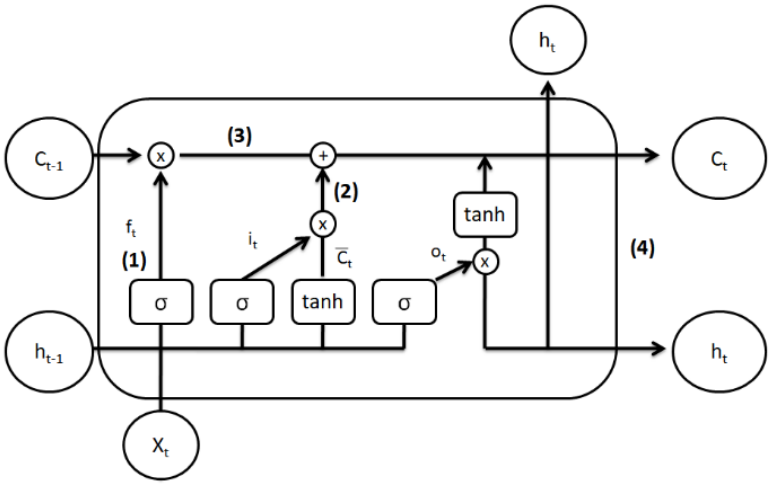

Figura 2.3: Célula de memória de uma LSTM. Adaptado de (Zhang, Wang, Liu, 2018).

A Rede Neural Recorrente LSTM, assim como as demais empregadas neste trabalho, utilizam o algoritmo de treinamento *backpropagation*, que busca otimizar a rede para que a distância entre o valor real e predito seja mínimo.

3. METODOLOGIA

A metodologia adotada neste trabalho será discutida nesse capítulo, explicitando desde a coleta e pré-processamento dos dados até a geração dos modelos utilizados. Todos os códigos-fonte do trabalho foram desenvolvidos utilizando Python 3, por englobar as bibliotecas que auxiliam nas etapas do trabalho. Todo material gerado está disponível no Github[2].

3.1 Sobre os dados

A base de dados construída para os experimentos realizados neste trabalho contou com dados das seguintes ações do negociadas na B3: Ambev (ABEV3), JBS (JBSAY), Petrobras (PETR4) e Vale (VALE3). Além disso, utilizamos também a cotação do dólar frente ao real.

O período utilizado na construção da base de dados foi entre junho de 2010 e novembro de 2022, e foram coletadas as cotações de abertura das ações indicadas. Todos os dados foram extraídos utilizando a biblioteca *pandas data_reader*[3] e obtidos com a API (do inglês, *Application Programming Interface*) do *Yahoo Finanças.*

Uma vez coletada a base de dados, várias etapas de pré-processamento se seguiram. A primeira foi a normalização das cotações, quando pegamos o valor em *t* e subtraímos do valor em t+1, extraindo o *Rate of change* (ROC) do valor da ação, que corresponde à variação de abertura de um dia para outro.

O segundo processamento na base de dados foi a separação entre 80% da base para o conjunto de Treinamento e 20% para o conjunto de Teste. Desta forma, garantimos que nenhum dado utilizado no teste fosse visto pelo modelo na etapa de treinamento e ajuste de hiper-parâmetros.

O terceiro passo foi a geração da janela utilizada para entrada no modelo. Para isso, pegamos os conjuntos de treinamento e teste e criamos 6 colunas para cada cotação, na qual a primeira coluna tem o valor da variação da cotação em t - 6 (6 dias

[2] https://github.com/ArthurGini/TCC_STOCK_PREDICTIONS
[3] https://pandas-datareader.readthedocs.io/en/latest/

atrás), a segunda coluna com dados de t - 5 (5 dias atrás) e assim por diante até a última coluna ter o valor em t, com o valor do dia atual.

Ao final, temos a base de dados de treinamento com 1912 registros e 36 colunas correspondentes à variação da cotação de 6 ativos brasileiros, sendo os valores de ROC dos últimos seis dias das cotações da ABEV3, JBSAY, PETR4, VALE3 e a cotação do dólar frente ao real. Ao final temos o conjunto de teste com 474 tuplas.

Para validação, durante o processo treinamento dos modelos, utilizamos o *Tensorflow*[4] para selecionar 20% de forma aleatória da base de treinamento, assim, calculamos as métricas de MSE e MAE identificando a assertividade na geração dos modelos.

3.2 Modelos

Utilizamos a mesma base experimental para gerar e avaliar diferentes arquiteturas de RNAs. Dessa forma, abordaremos nessa seção todos os processos em comum na geração dos diferentes modelos e, nas sessões seguintes, especificaremos o que difere na definição de cada arquitetura.

Os modelos estudados no trabalho foram desenvolvidos utilizando a biblioteca *Tensorflow* e otimizados pela biblioteca *Keras Tuner*[5], ambas disponíveis em Python.

Todos os modelos possuem alguns parâmetros que foram definidos de forma absoluta e outros de forma automática, utilizando a biblioteca *Keras Tuner* (KT). Os hiper-parâmetros numéricos definidos automaticamente têm seus valores mínimo e máximo determinados e variam dentro deste intervalo durante o ajuste. Já para os hiper-parâmetros categóricos (como a função de ativação, por exemplo), foram estabelecidos os valores possíveis, dentre os quais foi escolhido o melhor pela biblioteca KT.

Para treinar os modelos estudados aplicamos o otimizador *Adam* do *Keras*, tendo seu *learning rate* definido pelo KT com sua taxa variando dentre os valores de 0,1;

[4] https://www.tensorflow.org/api_docs
[5] https://keras.io/keras_tuner/

0,01 e 0,001. A métrica utilizada para a função de avaliação de perda foi o MSE e as métricas extraídas durante o treinamento foram MAE e MSE.

Utilizando as métricas, aplicamos o método *Hyperband* do KT por 10 vezes com o objetivo de maximizar o MSE do conjunto de validação. Para a seleção dos melhores hiper-parâmetros utilizamos o método *get_best_hyperparameters* do KT buscando nas repetições realizadas o modelo com o menor valor de MSE para o conjunto de validação.

Com os melhores hiper parâmetros encontrados, configuramos os modelos e executamos o treinamento por uma quantidade máxima de 500 épocas. Durante esse treinamento, utilizamos um *checkpoint* automático identificando qual o menor erro de validação da métrica MSE para salvar o modelo no seu estado atual, assim temos então, os melhores hiper parâmetros e configurações possíveis do modelo gerado.

O processo citado acima foi executado de forma completa 10 vezes para cada arquitetura de RNA estudada, dessa forma, obtivemos diferentes modelos configurados de forma automática e independente.

Os pontos específicos de cada arquitetura das RNAs, incluindo os parâmetros considerados na etapa de ajuste de hiper-parâmetros, são discutidos nas sessões a seguir.

3.2.1 MLP

As MLP foram configuradas com 5 camadas, sendo a primeira delas com uma quantidade fixa de 36 neurônios para a entrada dos dados. Temos também três camadas ocultas com quantidades variáveis de neurônios, sendo no mínimo 4 e no máximo 64 definidos pelo KT. A camada final possui um neurônio de saída utilizando a função de ativação linear.

A função de ativação foi escolhida dentre os valores (*sigmoid, softplus, softsign, tanh, selu, elu, relu*) uma vez pelo KT, sendo aplicada em todas as camadas

da RNA. O *learning rate* teve seu valor escolhido categoricamente entre os valores de 0,1; 0,01 e 0,001.

3.2.2 CNN

Configuramos o modelo CNN com 3 camadas. A primeira delas é uma *Conv1D* do *Tensorflow,* com número de filtros definidos pelo KT, em um intervalo entre 4 e 16. Selecionamos também o *kernel size* de forma automática, dentre um valor mínimo de 1 e no máximo 15, incrementando os valores em 2 a cada tentativa.

A segunda camada realiza a operação de *Flatten*, na qual os mapas de características gerados na camada anterior são transpostos para forma vetorial de apenas uma dimensão. Deste modo, o resultado das camadas é levado para a última que possui um neurônio com a função de ativação linear, gerando a saída final do modelo CNN.

3.2.3 LSTM

Para a construção do modelo LSTM utilizamos a primeira camada com a *layer* LSTM do *Tensorflow*, com *input shape* fixo em 36 pontos de entrada. A quantidade de nós nessa camada foi definida pelo KT, variando entre 16 e 64, e o hiper parâmetro de *return sequences* foi habilitado.

Com a segunda camada realizamos a operação de *Flatten*, tornando a saída do modelo com a mesma quantidade de linhas que a base de dados de teste, assim utilizamos na camada final um neurônio com função de ativação linear para obter a saída final do modelo de predição.

3.3 Avaliação

A avaliação dos modelos foi realizada de forma automática calculando para cada execução das diferentes arquiteturas duas principais métricas, a primeira é o *Mean absolute percentage error* (MAPE), que utiliza o MAE para o cálculo, representando o percentual médio absoluto do erro do modelo.

A segunda métrica calculada utiliza o MSE para o cálculo do *Root Mean Squared Error* (RMSE), onde representamos numericamente o quanto a predição do modelo varia para mais ou menos do valor real.

Todos os dados de avaliação foram compilados em tabelas que serão apresentadas e discutidas no capítulo a seguir.

4. RESULTADOS

Nesta seção apresentamos os resultados experimentais obtidos no trabalho. Abordamos as diferentes arquiteturas e seus modelos gerados na Seção 4.1, suas respectivas métricas de erro são discutidas na Seção 4.2.

4.1 Modelos Gerados

Como resultado da metodologia discutida obtivemos 30 modelos de RNAs gerados automaticamente e de forma independente. Nesta seção, apresentamos todos esses modelos, com suas características e resultados obtidos com as métricas de avaliação, a fim de identificar o que melhor prevê o índice Ibovespa.

Para avalição dos modelos utilizamos em conjunto as duas métricas, MAPE e RMSE, sendo que a primeira métrica representa a quantidade percentual do erro do modelo frente ao valor real e a segunda o quanto o modelo está errando em valores absolutos.

4.1.1 MLP

Na Tabela 4.1 temos os resultados dos modelos de RNA gerados utilizando a arquitetura MLP. Apresentamos as configurações dos hiper-parâmetros e os resultados obtidos com as duas métricas de qualidade, MAPE e RMSE, para o conjunto de teste.

Pelos resultados da Tabela 4.1 podemos notar que alguns hiper-parâmetros definidos pelo KT tendem a aparecer com maior frequência que outros, o *Learning Rate* por exemplo, varia menos em comparação à escolha da função de ativação. A quantidade de neurônios varia muito entre as camadas de diferentes modelos, apesar de seu máximo ser igual a 63 em todas as camadas.

Podemos observar que o Modelo 2 apresentou os menores erros tendo seu MAPE igual a 29,08%, e seu RMSE de 0,3698. Tendo isso em vista, é notável que esse é o melhor modelo gerado utilizando a arquitetura MLP.

			Quantidade de Neurônios					
Modelo	**Função de Ativação**	**Época**	**Camada 1**	**Camada 2**	**Camada 3**	***Learning rate***	**MAPE**	**RMSE**
1	softplus	301	14	56	61	0,01	42,69	0,5018
2	elu	88	34	18	52	0,01	29,08	0,3698
3	tanh	258	29	63	63	0,001	56,21	0,5823
4	softplus	480	59	47	37	0,01	41,13	0,4846
5	elu	412	57	17	40	0,001	35,03	0,4617
6	elu	101	63	50	48	0,001	34,23	0,4583
7	softsign	77	53	21	18	0,001	30,22	0,3703
8	softplus	103	5	17	42	0,01	43,59	0,5149
9	softplus	444	53	24	60	0,01	37,95	0,4621
10	relu	324	11	20	22	0,01	31,36	0,4513

Tabela 4.1: Hiper parâmetros de cada modelo MLP.

4.1.2 CNN

A Tabela 4.2 apresenta os valores dos hiper parâmetros e resultados obtidos utilizando os dados de teste nos modelos da arquitetura CNN.

É notável que alguns hiper parâmetros definidos pelo KT possuem uma variância diferente tendo em vista os resultados das demais arquiteturas. Observando a Tabela 4.2 temos que a função de ativação convergiu, na maioria dos modelos, para a *tanh*, modelos estes que não levaram aos menores valores de MAPE e RMSE.

O Modelo 4 possui o menor valor para RMSE e MAPE na série, indicando que é o que conseguiu predizer os valores da forma mais próxima ao valor real. Esse é o único modelo que utilizou a função de ativação *elu*, demonstrando que não é sempre que os hiper-parâmetros que mais convergem no KT produzem os melhores modelos.

Modelo	Função de Ativação	Época	*Kernel size*	*Learning Rate*	Filtro	MAPE	RMSE
1	tanh	30	1	0,001	16	34,51	0,3584
2	tanh	24	7	0,001	14	38,54	0,3929
3	tanh	374	1	0,01	5	33,65	0,3344
4	elu	96	9	0,01	10	31,73	0,3098
5	softsign	43	3	0,01	9	34,24	0,3487
6	tanh	41	3	0,001	10	32,06	0,3246
7	tanh	470	3	0,01	15	33,79	0,3374
8	tanh	256	11	0,01	5	38,24	0,3641
9	softsign	6	7	0,01	7	32,96	0,3280
10	tanh	384	7	0,01	9	32,93	0,3262

Tabela 4.2: Hiper parâmetros de cada modelo CNN.

4.1.3 LSTM

A Tabela 4.3 a seguir contém todos os modelos de RNAs gerados utilizando a arquitetura LSTM, consolidando as características definidas e suas respectivas métricas de qualidade geradas utilizando a base de dados de teste, MAPE e RMSE.

Podemos observar, na Tabela 4.3, que todos os modelos convergiram para o mesmo valor de *Learning Rate*, mostrando que, apesar deste hiper parâmetros ser o mesmo, os resultados produzidos pelos modelos podem ser diferentes.

É possível notar que o Modelo 10 é o que obteve as melhores métricas calculadas, com os valores de MAPE e RMSE sendo 27,99% e 0,3126, respectivamente.

Modelo	Função de Ativação	Épocas	Qtd. Neurônios	*Learning Rate*	MAPE	RMSE
1	sigmoid	16	18	0.01	30,79	0,3632
2	tanh	10	55	0.01	28,91	0,3323
3	tanh	27	18	0.01	31,60	0,3466
4	tanh	14	43	0.01	29,72	0,3228
5	tanh	8	56	0.01	34,55	0,3349
6	softsign	17	29	0.01	31,60	0,3407
7	tanh	12	23	0.01	34,29	0,3457
8	softsign	11	18	0.01	28,34	0,3414
9	softsign	11	22	0.01	28,09	0,3370
10	tanh	11	33	0.01	27,99	0,3126

Tabela 4.3: Hiper parâmetros de cada modelo LSTM.

4.2 Considerações finais

Para analisar as arquiteturas em conjunto a Tabela 4.4 apresenta as médias e desvio padrão das duas métricas utilizadas para a avaliação dos modelos estudados no trabalho.

É notável que as RNAs de arquitetura MLP obtiveram a maior média e desvio-padrão para todas as métricas, demonstrando ser o tipo de modelo que mais variou seus resultados nas 10 repetições realizadas.

Podemos observar que as arquiteturas CNN e LSTM tiveram seu desvio padrão do MAPE aproximadamente 4 vezes menor do que a calculada utilizando a rede MLP, indicando que, no geral, as arquiteturas mais complexas ficaram com os resultados variando menos tendo em vista essa métrica.

É possível notar que a rede LSTM possui a menor média para a métrica MAPE. Contudo, seu desvio-padrão é maior que a obtida na rede CNN. Para as métricas de RMSE a rede LSTM se saiu melhor perante as demais estudadas, demonstrando que esse tipo de rede erra com uma diferença menor entre o valor real e o predito.

Arquitetura Utilizada	**Média do MAPE**	**Desvio Padrão MAPE**	**Média do RMSE**	**Desvio Padrão RMSE**
MLP	38,15	8,1753	0,4657	0,0635
CNN	34,27	2,3438	0,3424	0,0242
LSTM	30,59	2,4296	0,3377	0,0137

Tabela 4.4: Média e Desvio padrão das arquiteturas.

5. CONCLUSÃO

O trabalho desenvolvido teve como resultado (i) a criação de uma base de dados do mercado brasileiro de ações, (ii) a realização de um estudo comparativo entre diferentes tipos de RNAs. O foco do trabalho foi avaliar qual tipo de arquitetura consegue predizer melhor o índice IBOVESPA.

Utilizamos as arquiteturas de RNAs MLP, CNN e LSTM, a entrada para os modelos consistiu em uma base de dados com 6 ativos, sendo eles Ambev (ABEV3), JBS (JBSAY), Petrobras (PETR4), Vale (VALE3) e a cotação do dólar frente ao real.

Para o treinamento dos modelos utilizamos a biblioteca KT para otimizar os hiper parâmetros de forma automática. Realizamos 10 repetições do treinamento dos modelos de RNAs para cada arquitetura utilizada.

Tendo em vista os resultados obtidos, avaliamos os 30 modelos gerados automaticamente utilizando as métricas de MAPE e RMSE, em busca de identificar qual arquitetura de RNA melhor prediz o índice IBOVESPA.

Considerando as médias e desvio-padrão das principais métricas utilizadas, podemos concluir que, as predições que a rede do tipo LSTM gera varia menos considerando o valor predito e o real.

Para trabalhos futuros, recomenda-se um estudo mais abrangente de outras arquiteturas de RNAs. Ademais, pode-se também construir uma estratégia de investimento que utilize os modelos de predição como os gerados neste trabalho.

6. REFERÊNCIAS

BUSTOS, Oscar; POMARES-QUIMBAYA, Alexandra. Stock market movement forecast: A systematic review. Expert Systems with Applications, v. 156, p. 113464, 2020.

COELHO, Leandro dos Santos; SANTOS, André Alves Portela; COSTA JR, Newton Carneiro Affonso da. Podemos prever a taxa de cambio brasileira? Evidência empírica utilizando inteligência computacional e modelos econométricos. Gestão & Produção, v. 15, p. 635-647, 2008.

DE O. CAROSIA, Arthur E.; COELHO, Guilherme P.; SILVA, Ana EA da. The influence of tweets and news on the brazilian stock market through sentiment analysis. In: Proceedings of the 25th Brazillian Symposium on Multimedia and the Web. 2019. p. 385-392.

FARIA, EL de. Redes neurais convolucionais e mÁquinas de aprendizado extremo aplicadas ao mercado financeiro brasileiro. Programa de Pós-graduação em Engenharia Civil, COPPE, da Universidade Federal do Rio de Janeiro, 2018.

HAYKIN, Simon. Neural networks and learning machines, 3/E. Pearson Education India, 2008.

HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long short-term memory. Neural computation, v. 9, n. 8, p. 1735-1780, 1997.

JORDAN, M. I. Serial order: a parallel distributed processing approach. technical report, june 1985-march 1986. California Univ., San Diego, La Jolla (USA). Inst. for Cognitive Science, 1986.

KUMBURE, Mahinda Mailagaha et al. Machine learning techniques and data for stock market forecasting: a literature review. Expert Systems with Applications, p. 116659, 2022.

LIPTON, Zachary C.; BERKOWITZ, John; ELKAN, Charles. A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019, 2015.

MIRETE-FERRER, Pedro M. et al. A Review on Machine Learning for Asset Management. Risks, v. 10, n. 4, p. 84, 2022.

NGARAMBE, Jack et al. A review on the current usage of machine learning tools for daylighting design and control. Building and Environment, p. 109507, 2022.

OZBAYOGLU, Ahmet Murat; GUDELEK, Mehmet Ugur; SEZER, Omer Berat. Deep learning for financial applications: A survey. Applied Soft Computing, v. 93, p. 106384, 2020.

RHEA, Robert. The Dow theory: An explanation of its development and an attempt to define its usefulness as an aid in speculation. Fraser Publishing Company, 1993.

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

RUMELHART, David E. Learning representations by back-propagation errors. nature, v. 323, p. 533-536, 1986.

RUSSELL, Stuart; NORVIG, Peter. Artificial intelligence: A modern approach. Pretice Hall Series in Artificial Intelligence, v. 1, p. 649-789, 2003.

SIMON, Phil. Too big to ignore: the business case for big data. John Wiley & Sons, 2013.

ZHANG, Lei; WANG, Shuai; LIU, Bing. Deep learning for sentiment analysis: survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, v. 8, n. 4, p. e1253, 2018.