PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Ciência de Dados e Big Data

Alison Diego Harka Machado

PREDIÇÃO DO PREÇO DE AÇÃO NO MERCADO FINANCEIRO UTILIZANDO REGRESSÃO LINEAR, ARIMA E MLP

Belo Horizonte
2021

Alison Diego Harka Machado

PREDIÇÃO DO PREÇO DE AÇÃO NO MERCADO FINANCEIRO UTILIZANDO REGRESSÃO LINEAR, ARIMA E MLP

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados e Big Data como requisito parcial à obtenção do título de especialista.

Belo Horizonte

2021

LISTA DE FIGURAS

SUMÁRIO

1. Introdução

Nesta seção será apresentada uma contextualização sobre mercado financeiro brasileiro, principais tipos de investimentos, mercado de ações, bolsa de valores, uso do aprendizado de máquina para finanças, bem como quais ferramentas que serão utilizadas no decorrer do trabalho. Além disso, será definido o problema e objetivos que este trabalho se propõe a realizar.

1.1. Contextualização

O Mercado Financeiro Brasileiro tem se tornado cada vez mais uma oportunidade para quem deseja obter novas rendas por meio de investimentos sobre as negociações de ativos de valor financeiro. Com a avanço da tecnologia e da Internet ao longo dos anos, essas negociações foram impulsionadas devido a facilidade de acesso a informações de qualquer lugar do mundo.

O Mercado Financeiro é voltado para a realização de atividades de transferências de recursos entre agentes econômicos, em que se efetuam operações ou transações com títulos de curto, médio, e longo prazo (INFOMONEY, 2021).

Para quem deseja investir no Mercado Financeiro Brasileiro, é importante conhecer alguns conceitos que são aplicados na área. Neste caso, podemos citar dois conceitos iniciais sobre duas principais modalidades de investimentos: renda fixa e renda variável.

A renda fixa são títulos que pagam, em períodos definidos, uma quantia de remuneração, que pode ser determinada no momento da aplicação ou no momento do resgate (no final da aplicação). Quando você compra um título de renda fixa está emprestando dinheiro ao emissor do título, que pode ser um banco, uma empresa ou o governo. Dessa maneira, na renda fixa os juros do título são a remuneração que você recebe por emprestar seu dinheiro.

Já a renda variável são títulos que não apresentam um índice de rentabilidade fixo no momento da aplicação, ou seja, a remuneração não é discriminada anteriormente, como acontece com os títulos de renda fixa. Sendo assim, a rentabilidade destas aplicações depende das condições de mercado, que podem sofrer variações para mais ou para menos, e neste caso, uma maior exposição a riscos que podem gerar prejuízos financeiros.

Dentre os principais tipos de investimentos em renda fixa pode-se destacar: Tesouro Direto, Certificado de Depósito Bancário (CDB), Letra de Crédito Imobiliário (LCI), Letra de Crédito do Agronegócio (LCA), entre outros. Já na renda variável destaca-se: Commodities, Fundos de Investimentos imobiliários e Ações.

Para estudo desse trabalho, ressalta-se as Ações, pois visa um alto potencial de retorno financeiro. Além disso, investir em ações tem chamado a atenção de muitos brasileiros, conforme site da INFOMONEY (2021), o número de pessoas físicas que iniciaram investimentos em ações passou de 619.625 em 2017 para 3.065.775 até setembro de 2020, uma variação de 395%, e como consequência disso, o volume de negociações e valores envolvidos aumentaram.

Em contrapartida, investir em ações não é uma tarefa simples para quem deseja ter lucratividade, isso ocorre, pois, existe uma oscilação alta nos preços devido a vários fatores de mercado, como por exemplo, a política, a desvalorização de uma empresa, pandemia, entre outros.

Além disso, a falta de conhecimento técnico sobre o assunto e o alto volume de dados para analisar são fatores que podem gerar dúvidas para os investidores. No entanto, existem técnicas e ferramentas disponíveis, inclusive gratuitamente, para auxiliar nesse processo, conforme será demonstrado mais adiante nesse trabalho.

Nesse contexto, Ações são papéis emitidos pelas empresas de sociedades anônimas que representa a menor fração do capital dessas empresas, ou seja, representa uma parte dessa empresa. Dessa forma, ao adquirir esses papéis, os acionistas (pessoa titular que detém as ações) se tornam sócios da empresa. Para cada ação é estimado um valor, que representa o que a empresa pode oferecer para os acionistas. Esses valores dependerão de vários fatores, tais como: oferta e procura, pagamentos de dividendos (parte do lucro da empresa paga a seus acionistas), projeções de crescimento, entre outros.

As Ações podem ser classificadas em dois tipos principais: ordinárias ou preferenciais. Ações ordinárias são aquelas que permitem direito a voto nas assembleias, ou seja, permite a participação nas decisões da companhia, como por exemplo, eleger membros do conselho de administração da empresa. Já as ações preferenciais permitem o recebimento de dividendos em valor superiores às ações ordinárias, bem como a prioridade no recebimento de reembolso do capital.

Importante salientar que o Mercado de Ações está diretamente relacionado com a economia de um país, pois, quanto mais desenvolvida é a economia, mais ativo

é seu mercado de capitais, o que se traduz em mais oportunidades para as pessoas, empresas e instituições aplicarem seus patrimônios. Uma empresa abre seu capital para obter uma fonte de captação de recursos permanente. Isso acontece quando a empresa lança suas ações ao público, ou seja, emite Ações e as negocia na bolsa de valores.

A bolsa de valores é um ambiente de negociação entre quem quer vender Ações e quem deseja comprá-las. No Brasil, a instituição responsável pelo ambiente dessas negociações é denominada B3, antiga Bovespa. As principais Ações que são negociadas na bolsa de valores formam o índice IBOVESPA, que é um indicador do desempenho das principais Ações no país, sendo uma espécie de “termômetro” do Mercado Financeiro Brasileiro, indicando se a bolsa está em alta ou baixa.

Para quem deseja ingressar os investimentos em ações ou já é acionista, é importante conhecer e aplicar estratégias para maximizar os lucros e mitigar os riscos. Nesse contexto, técnicas computacionais de Aprendizado de Máquina ou Machine Learning (ML) ganham espaço, pois auxiliam os acionistas ao lidar com fluxos de dados.

Segundo MITCHELL (1997), Machine Learning (ML) pode ser definido como o estudo de algoritmos de computador que se aprimoram automaticamente por meio da experiência e do uso de dados. Com Machine Learning é possível, por exemplo, criar modelos preditivos, modelos de agrupamento de dados por meio de características semelhantes, entre outros.

Um exemplo da aplicação de ML em finanças é o uso de dados de mídias sociais para análise de sentimento dos usuários sobre notícias financeiras. Empresas conhecidas no ramo financeiro, como a Bloomberg, Thomson Reuters e RavenPack estão fornecendo dados de sentimento de notícias financeiras sob medida para suporte dos acionistas nas negociações das ações.

Nesse exemplo, a técnica explorada de ML foi o processamento de linguagem natural (PLN), que também pode ser aplicado a documentos não estruturados, por exemplo, relatórios de anúncio de lucros, relatórios SEC 10K (informações sobre empresas). Esses relatórios podem servir de insumo que permitirá extrair informações de forma rápida e automatizada.

Para este trabalho, foram criados modelos de ML capazes de realizar predição do preço da ação no mercado financeiro. Para tanto, foi utilizada as seguintes ferramentas criação dos modelos e suporte:

Ferramentas utilizadas para execução dos Modelos de ML		
Ferramenta	**Versão**	**Descrição**
Python	3.7.6	Linguagem de programação de alto nível, interpretada de script, imperativa, orientada a objetos, funcional, de tipagem dinâmica e forte (PYTHON, 2021).
Plataforma Anaconda3	1.9.12	Distribuição das linguagens de programação Python e R para computação científica, que visa simplificar o gerenciamento e implantação de pacotes. A distribuição inclui pacotes de ciência de dados adequados para Windows, Linux e macOS. (ANACONDA, 2021).
Jupyter notebook	6.0.3	Interface gráfica onde permite a criação e execução de códigos de uma linguagem de programação, incluindo a exibição de gráficos (JUPYTER, 2021).

Essas três ferramentas foram utilizadas para o desenvolvimento das etapas deste trabalho. YUXING (2017) destaca alguns motivos para escolha da linguagem Python para análise de dados financeiros:

- É gratuito em termos de licença e está disponível para todos os principais sistemas operacionais, como Windows, Linux, Unix, Mac. Ser gratuito possui benefícios, por exemplo, quando alunos se formam, eles podem aplicar o que aprenderam aonde quer que vão. Isso também é válido para a comunidade financeira.
- É poderoso, flexível e fácil de aprender. É capaz de resolver quase todas as estimativas financeiras e econômicas.
- Está diretamente relacionado com Big Data. As linguagens Python e R são duas das linguagens de programação de código aberto mais populares do mundo para análise de dados.
- Existem muitas bibliotecas/módulos prontos para serem utilizados, que simplificam o processo de codificação ou programação.

Na imagem seguinte, é apresentado o fluxo das etapas de execução dos modelos de Machine Learning que será utilizado nesse trabalho.

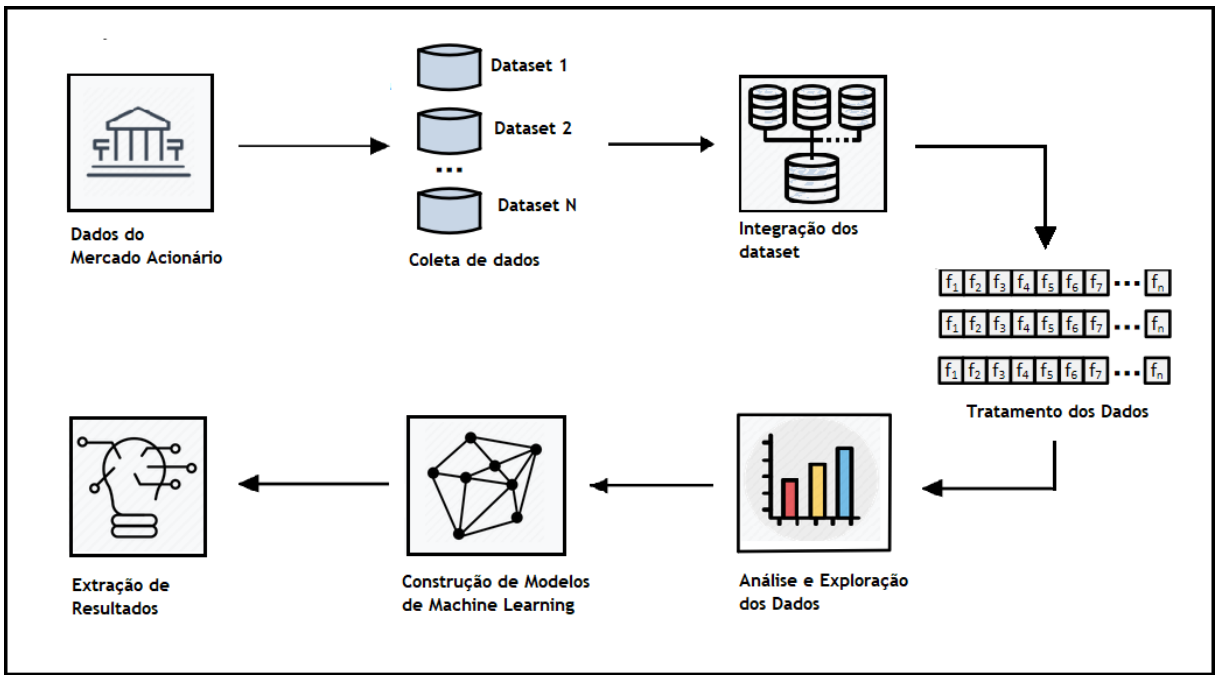

Figura 1: Fluxo de Execução dos Modelos de Machine Learning

O fluxo se inicia com os dados do mercado acionário onde é realizada uma consulta geral sobre fontes de dados que podem ser utilizadas para compor a análise. Em seguida, há a etapa de coleta de dados, onde serão utilizados métodos de extração e importação para as ferramentas. Em seguida, é realizado a integração dos conjuntos de dados. Após isso, realiza-se a etapa de tratamento de dados, que tem o objetivo enriquecer o conjunto de dados. Depois, é realizado a análise e exploração dos dados por meio de gráficos para verificar padrões e insights. Seguidamente, é realizado a construção dos modelos aplicando técnicas e algoritmos para resolução do problema, e por fim a extração dos resultados encontrados.

1.2. O problema proposto

A capacidade de prever o futuro apuradamente é fundamental para muitos processos de decisões em planejamento e formulações de estratégias. A utilização de boas ferramentas que auxiliem a tomada de decisão em investimentos no mercado de ações é indispensável e pode representar o sucesso ou o fracasso em uma negociação.

Em função disso, e com o desafio de maximizar os lucros para os acionistas, o presente trabalho busca resolver a seguinte pergunta ou problema:

Qual abordagem, utilizando Machine Learning, apresenta melhor resultado na predição de preço da ação, de modo que seja uma ferramenta útil para auxiliar no processo de tomada de decisão para compra ou venda de ações de curto prazo?

A seguir, será utilizada a técnica dos 5-Ws para definição do plano de ação de resolução do problema. Essa técnica permite dar clareza do que será feito, do porquê, de qual período se trata, de quem, e de onde serão extraídos os dados.

(*Why* – Por quê?): Riscos envolvidos em aplicações financeiras que podem gerar perdas financeiras. Nesse caso, uma ferramenta que auxilie nesse processo de compra ou venda de ações passa a ser um diferencial.

(*Who* – De quem?): Os dados se referem aos ativos negociados na bolsa de valores do Brasil – B3.

(*What* - O que?): Elaborar uma ferramenta preditiva que auxilie no processo de tomada de decisão para compra ou venda de ações.

(*Where* – De onde?): Os dados são disponibilizados publicamente nos sites Yahoo Finance e Investing.

(*When* – De quando?): O período de análise é 01.01.2016 a 30.06.2021.

1.3. Objetivos

Este trabalho tem como objetivo geral a elaboração de uma estratégia, utilizando modelos de ML, que seja capaz de realizar predições de curto prazo, para auxiliar o investidor no processo de tomada de decisão sobre a compra ou venda de determinada ação.

Os objetivos específicos que se propõe são:

- Buscar dados de diferentes fontes;
- Realizar a integração e tratamento dos dados;
- Realizar a análise e exploração de dados utilizando recursos gráficos;
- Aplicação dos modelos de Regressão Linear, ARIMA e MLP para realizar predição da ação um dia a frente;
- Comparar os resultados de cada modelo.

2. Coleta de Dados

Nesta seção será apresentado os conjuntos de dados (Datasets) escolhidos, a descrição de cada Dataset, data de coleta, bem como as bibliotecas que serão utilizadas em todas as etapas do trabalho.

Foram utilizados dois Datasets de fontes distintas com séries históricas, conforme período 01.01.2016 a 30.06.2021.

Os Datasets são:

- Ação ITUB4, da empresa Itaú Unibanco;
- Dólar USDBRL.

A ação (ITUB4) foi escolhida por se tratar do maior banco do Brasil, o Itaú, e que em julho de 2019 seu valor de mercado representava cerca de US$ 92,4 bilhões (INFOMONEY, 2021).

A utilização do Dataset Dólar (USDBRL) servirá para analisar as correlações com a ação, e principalmente, para agregar no processo de *Feature Selection/Engineering* (método para seleção das melhores características dos conjuntos de dados).

Destaca-se que, além da ação escolhida, os modelos de Machine Learning serão treinados de modo que outras ações possam ser analisadas.

A seguir apresentamos os dois métodos utilizados na extração dos dados:

1) **download()**, da biblioteca yfinance;
2) **get_currency_cross_historical_data()**, da biblioteca investpy.

Em ambos os métodos, o acesso é realizado remotamente por meio de API (*Application Programming Interface*), em que no primeiro caso, a fonte de dados é do site Yahoo Finance (YAHOO, 2021), já no segundo caso é referente ao site Investing (INVESTPY, 2021), que podem ser localizados nos seguintes endereços:

https://finance.yahoo.com/quotes/API,Documentation/view/v1/

https://investpy.readthedocs.io/api.html

Os dados foram coletados no início de realização desse trabalho em 02/07/2021.

A seguir, trechos do código utilizados para extração dos dados:

```
#Obtendo dataset da ação (01/01/2016 até 30/06/21)
ticker = "ITUB4.SA"
df = yfinance.download(ticker, start="2016-01-01", end="2021-06-30")
```

```
[**********100%**********]  1 of 1 completed
```

```
#Obtendo dataset do dolar (01/01/2016 até 30/06/21)
dolar = investpy.get_currency_cross_historical_data(currency_cross='USD/BRL', from_date='01/01/2016', to_date='30/06/2021')
```

Figura 2: Coleta de dados

2.1. Datasets

Nas tabelas a seguir são apresentadas as descrições dos Datasets:

DATASET DA AÇÃO ITUB4		
Nome da coluna	**Descrição**	**Tipo**
Date	Dia referência dos preços.	Datetime
High	Preço máximo que a ação chegou no dia.	Float
Low	Preço mínimo que a ação chegou no dia.	Float
Open	Preço de abertura da ação no dia.	Float
Close	Preço de fechamento da ação no dia.	Float
Volume	Volume de negociações no dia.	Int
Adj Close	Preço de fechamento ajustado no dia após ser contabilizado as ações corporativas.	Float

DATASET DO DÓLAR USDBRL		
Nome da coluna	**Descrição**	**Tipo**
Date	Dia referência dos preços.	Datetime
High	Preço máximo que o dólar chegou no dia.	Float
Low	Preço mínimo que o dólar chegou no dia.	Float
Open	Preço de abertura do dólar no dia.	Float
Close	Preço de fechamento do dólar no dia.	Float
Currency	Unidade monetária	Object

2.2. Bibliotecas

A seguir, são apresentadas as bibliotecas utilizadas nas etapas de coleta, processamento, tratamento, análise e exploração de dados, bem como a criação dos modelos de ML:

```
import yfinance
import investpy

import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import plotly.graph_objects as go

from sklearn.feature_selection import SelectKBest
from sklearn.preprocessing import MinMaxScaler

from sklearn import linear_model
from statsmodels.tsa.arima_model import ARIMA
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import GridSearchCV

from math import sqrt
from sklearn.metrics import mean_squared_error, r2_score
```

Figura 3: Bibliotecas utilizadas

3. Processamento/Tratamento de Dados

Nesta seção será apresentado o processamento e tratamento dos dados, destacando a quantidade de registros obtidos, a presença de valores ausentes e as justificativas de alteração ou remoção de valores. Esta etapa tem por finalidade a limpeza, enriquecimento, codificação, padronização e seleção de características ou atributos.

Inicialmente, foi verificado algumas estatísticas de cada Dataset, com seus devidos atributos.

```
df.describe()
```

	Open	High	Low	Close	Adj Close	Volume
count	1362.000000	1362.000000	1362.000000	1362.000000	1362.000000	1.362000e+03
mean	28.084639	28.446573	27.710245	28.072364	25.506462	2.467253e+07
std	5.737441	5.759889	5.689683	5.726834	6.195710	1.471685e+07
min	13.939393	14.072727	13.824242	13.981818	10.908615	0.000000e+00
25%	23.988333	24.326666	23.660000	24.006666	21.227223	1.466430e+07
50%	27.891666	28.240000	27.520000	27.896667	25.650890	2.105080e+07
75%	33.355000	33.807501	33.009165	33.401666	30.934887	3.063832e+07
max	38.669998	39.790001	38.400002	39.689999	36.459167	1.049172e+08

Figura 4: Estatísticas do Dataset ITUB4.

```
dolar.describe()
```

	Open	High	Low	Close
count	1430.000000	1430.000000	1430.000000	1430.000000
mean	4.018076	4.048761	3.989179	4.018819
std	0.825682	0.836519	0.814115	0.826104
min	3.056800	3.079600	3.038700	3.057500
25%	3.282400	3.301850	3.263500	3.282925
50%	3.803350	3.833550	3.783300	3.806150
75%	4.315825	4.331325	4.295325	4.323675
max	5.924800	5.971800	5.817600	5.885600

Figura 5: Estatísticas do Dataset Dólar.

Além disso, foi verificado as quantidades totais de registros compostos nos Datasets, incluindo valores nulos se houver, pois serão tratados posteriormente.

```
# quantidade de registros do dataset da ação
df.count()
```

```
Date         1362
Open         1362
High         1362
Low          1362
Close        1362
Adj Close    1362
Volume       1362
dtype: int64
```

Figura 6: Quantidade Registros Dataset Ação

```
# quantidade de registros do dataset da dólar
dolar.count()
```

```
Date     1430
Open     1430
High     1430
Low      1430
Close    1430
dtype: int64
```

Figura 7: Quantidade Registros Dataset Dólar

Conforme é observado, a quantidade de registros para cada atributo do Dataset Ação é de 1362, enquanto no Dataset do dólar é de 1460. Essa diferença é devido à ausência de registros na fonte dos dados da ação. Essa ausência poderia ser suprida com preenchimento desses registros utilizando diversas estratégias de *oversampling* (valores aleatórios, média de valores, outros), entretanto, neste trabalho optou-se por não modificar o Dataset inserindo valores que possam se diferenciar do comportamento real da ação.

A próxima verificação diz respeito a presença de valores nulos, conforme é apresentado a seguir:

```
df.isnull().sum()
```

```
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

Figura 8: Verificação de valores nulos do Dataset ITUB4

```
dolar.isnull().sum()
```

```
Open         0
High         0
Low          0
Close        0
Currency     0
dtype: int64
```

Figura 9: Verificação de valores nulos do Dataset Dólar

Conforme é observado nas figuras 8 e 9, não há presença de qualquer valor nulo, logo, não será necessário nenhum tratamento em relação a isso.

Antes de realizar a integração dos Datasets, alguns ajustes foram realizados no Dataset do dólar.

A primeira modificação foi a eliminação da coluna "Currency", pois apresenta apenas a informação da unidade monetária (no caso, BRL) e neste trabalho essa informação não será necessária.

A segunda modificação foi a renomeação dos atributos, pois alguns atributos possuem o mesmo nome que atributos do dataset da ação.

A terceira modificação foi utilizado a integração dos datasets por meio do método merge(), sendo o modo como os dados são integrados é utilizando o "inner join".

A seguir os códigos dos ajustes realizados:

```
dolar = dolar.drop('Currency', axis=1)
```

```
df_dolar = dolar.rename(columns={"Open": "Open_dolar",
                                 "High": "High_dolar",
                                 "Low": "Low_dolar",
                                 "Close": "Close_dolar"})
df_dolar.head()
```

	Date	Open_dolar	High_dolar	Low_dolar	Close_dolar
0	2016-01-01	3.9608	3.9608	3.9608	3.9608
1	2016-01-04	3.9608	4.0709	3.9590	4.0402
2	2016-01-05	4.0403	4.0590	3.9883	4.0077
3	2016-01-06	4.0071	4.0570	4.0040	4.0244
4	2016-01-07	4.0248	4.0749	4.0216	4.0448

```
df_full = pd.merge(df, df_dolar, on='Date', how='inner')
df_full.head()
```

	Date	Open	High	Low	Close	Adj Close	Volume	Open_dolar	High_dolar	Low_dolar	Close_dolar
0	2016-01-04	15.709090	15.812121	15.230303	15.230303	11.882679	38071770	3.9608	4.0709	3.9590	4.0402
1	2016-01-05	15.393939	15.624242	15.200000	15.357575	11.981978	21093105	4.0403	4.0590	3.9883	4.0077
2	2016-01-06	15.151515	15.654545	15.030303	15.296969	11.934696	33520575	4.0071	4.0570	4.0040	4.0244
3	2016-01-07	15.090909	15.327272	15.006060	15.006060	11.707721	28766430	4.0248	4.0749	4.0216	4.0448
4	2016-01-08	15.121212	15.303030	15.012121	15.012121	11.712455	18087465	4.0453	4.0569	4.0050	4.0245

Figura 10: Integração dos dados

Outra modificação realizada no dataset foi o deslocamento da coluna "Adj close" para cima, para que os modelos de Machine Learning possam desenvolver o aprendizado baseados nas caraterísticas do dia atual, de modo a prever o valor do "Adj close" do dia seguinte. Dessa forma, as características do último registro do dataset (referente ao dia 29/06/21) foi retirado e utilizado ao final para validar o treinamento realizado pelos algoritmos.

```
#Deslocando os valores da coluna 'Adj Close' para cima
df_full['Adj Close'] = df_full['Adj Close'].shift(-1)
```

Figura 11: Deslocamento da coluna Adj Close

4. Análise e Exploração dos Dados

Nesta seção será apresentada a etapa de análise e exploração dos dados. Para fins de desenvolvimento dessa etapa, será utilizado recursos gráficos para visualizar o comportamento dos dados.

Existem diversas possibilidades de análises, porém, será destacado a seguir algumas que foram consideradas importante nesse processo e que também são conhecidas por investidores.

a) Análise de cotação ao longo do tempo

Os dados presentes nos datasets referem-se às informações de cotação dos ativos, no caso, a ação e o dólar. Tais informações são importantes para o investidor, pois permite que ele tenha uma visão mais clara a respeito dos movimentos do mercado. Dessa forma, o gráfico de Candlesticks costuma ser bastante utilizado pelo fato de apresentar de forma intuitiva as cotações.

O gráfico de Candlesticks utiliza quatro parâmetros: data, preço de abertura, preço de fechamento, preço máximo e preço mínimo.

A área entre o preço de abertura e o preço de fechamento indicam o corpo do Candle. Os valores de máxima e mínima representam, respectivamente, o maior e o menor valor que o ativo alcançou durante a negociação. Graficamente todo Candle possui um corpo, mas não necessariamente as linhas que indicam máxima e mínima, pois o máximo e o mínimo podem ser valores entre valor de abertura e de fechamento.

Se no final de um período de negociação o valor do fechamento for maior que o da abertura, o Candle será de alta. Entretanto, se o valor de fechamento for menor que o de abertura, o Candle será de baixa.

Na próxima página, é apresentado um exemplo representativo de um Candle de alta e de baixa.

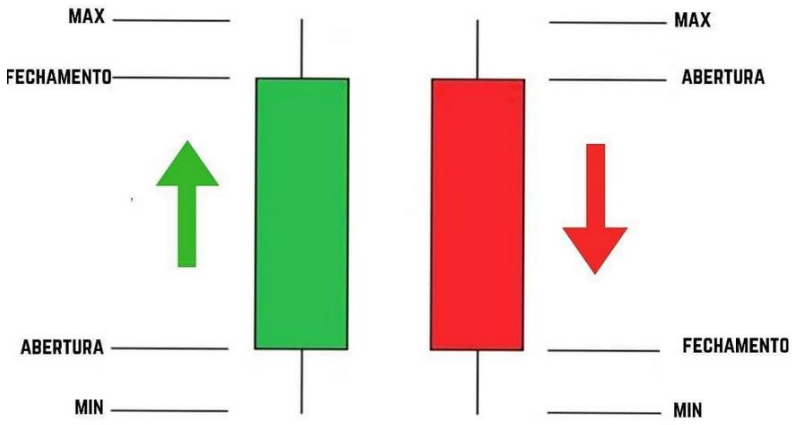

Figura 12: Exemplo de Candle

Nas seguintes imagens são apresentados os códigos para exibir os gráficos de Candlesticks da ação e do dólar, utilizando a biblioteca “plotly”.

```
# gráfico de candlestick para ação
# criamos uma variável, do tipo dicionário (chave e valor), para personalizar a lista de impressão no gráfico.
trace1 = {
    'x': df_full.Date,
    'open': df_full.Open,
    'close': df_full.Close,
    'high': df_full.High,
    'low': df_full.Low,
    'type': 'candlestick',
    'name': 'ITUB4.SA',
    'showlegend': True
}

data = [trace1]
layout = go.Layout()

fig1 = go.Figure(data=data, layout=layout)
fig1.show()
```

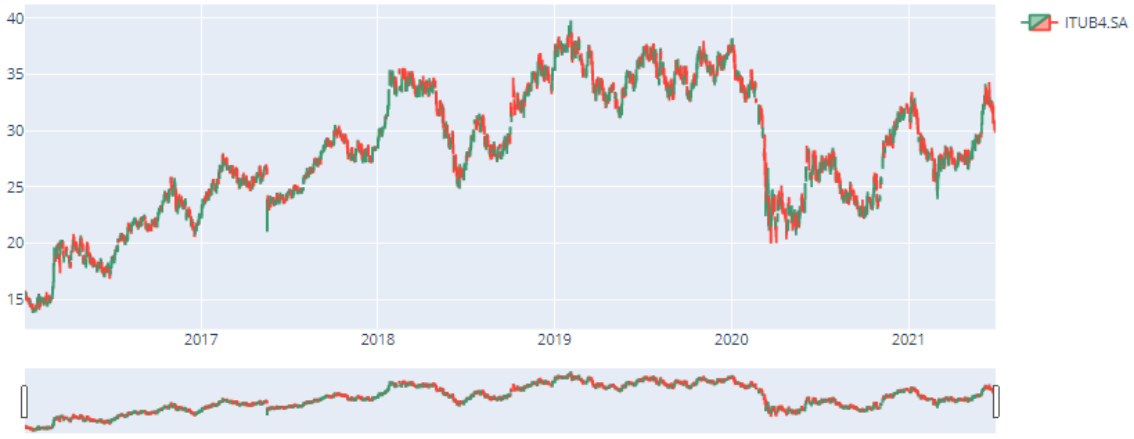

Figura 13: Gráfico de Candlesticks da Ação

```
# gráfico de candlestick do dolar
trace2 = {
    'x': df_full.Date,
    'open': df_full.Open_dolar,
    'close': df_full.Close_dolar,
    'high': df_full.High_dolar,
    'low': df_full.Low_dolar,
    'type': 'candlestick',
    'name': 'Dólar',
    'showlegend': True
}

data = [trace2]
layout = go.Layout()

fig2 = go.Figure(data=data, layout=layout)
fig2.show()
```

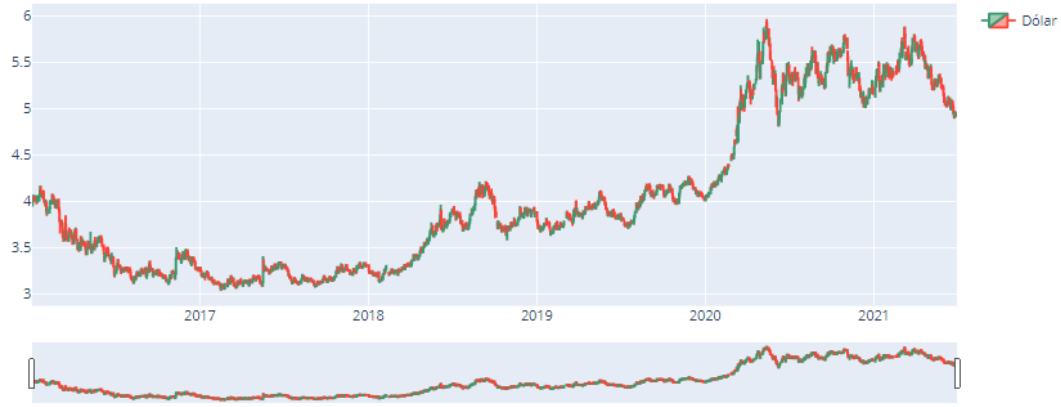

Figura 14: Gráfico de Candlesticks do Dólar

Além dos gráficos de Candlesticks, será apresentado a seguir um gráfico de linha com a ação e dólar. Nesse caso, foi utiliza a biblioteca matplotlib.

```
plt.figure(figsize=(16,6)) #altura e largura do gráfico
plt.plot(df_full['Date'], df_full['Adj Close'])
plt.plot(df_full['Date'], df_full['Close_dolar'])
plt.legend(['Adj Close', 'Close_dolar'])
plt.grid()
plt.title("Cotação da ação e dólar x Tempo", fontsize = 20)
plt.show()
```

Figura 15: Gráfico do preço de fechamento da ação e dólar.

Conforme informação do gráfico anterior, é possível notar que no período entre janeiro de 2020 e julho de 2020 houve forte desvalorização da ação, isso aconteceu devido aos impactos causados pela pandemia, referente ao coronavírus (COVID-19).

b) Análise da correlação entre os atributos

Outro tipo de visualização relevante para este trabalho é a análise correlação entre todos os atributos do Dataset. Essa visualização é verificada com mapa de calor, que facilita a leitura das correlações entre variáveis, pois representa a escala dos números em cores. As áreas mais escuras se referem a fraca relação entre os dados, já as áreas mais claras se referem a relação mais forte dos dados.

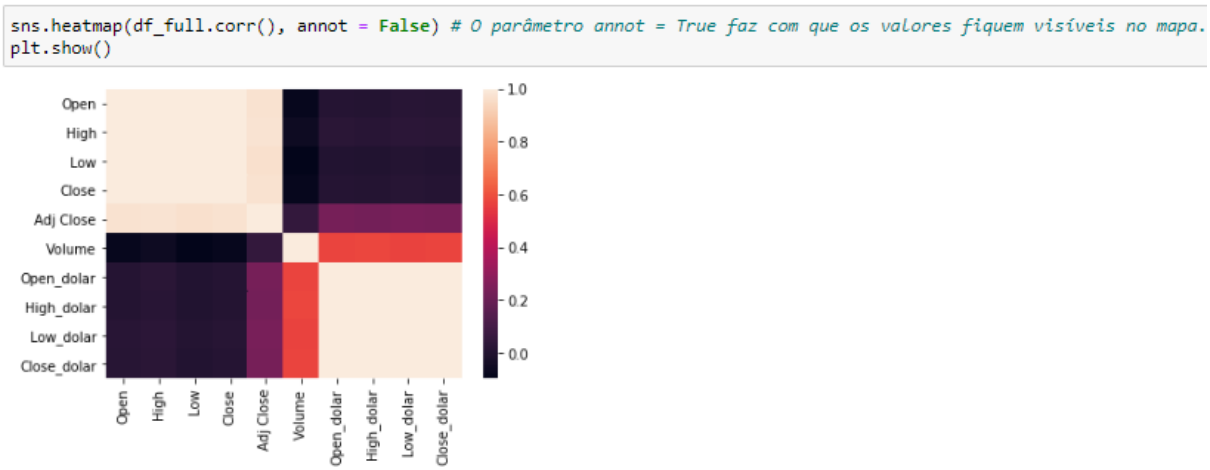

Figura 16: Mapa de Calor do Dataset Completo.

Nesse mapa de calor, pode-se perceber que os atributos possuem forte correlação com o atributo "volume". Isso se deve por conta dos valores de volume estarem em uma escala muito superior em relação aos demais. Tal fato, justifica a necessidade de normalizar o Dataset para que essa discrepância de escala não interfira no processo de treinamento dos modelos de Machine Learning.

5. Criação de Modelos de Machine Learning

Nesta seção serão apresentados os conceitos relacionados ao aprendizado de máquina, tipos de aprendizado, pré-processamento dos dados, treinamento dos modelos, métricas de avaliação de desempenho, e algoritmos utilizados.

5.1. Tipos de Aprendizado de Máquina

Aprendizado de máquina é uma subárea da inteligência artificial que apresenta métodos e algoritmos que possuem a capacidade de aprender por meio dos dados apresentados. Os modelos de aprendizado de máquina utilizam os padrões presentes em um conjunto de dados para adquirir experiências. O processo de aprendizado pode ser classificado como supervisionado, não supervisionado, semi-supervisionado e por reforço (MITCHELL, 1999).

5.1.1. Aprendizado Supervisionado

Processo de aprendizagem no qual há um "supervisor" que sabe previamente qual é a resposta esperada dado um valor de entrada. Para efetuar o aprendizado, o algoritmo compara sua saída com a deste supervisor, para que assim, se ajuste em busca de minimizar o erro de sua resposta. Nesse tipo de aprendizado pode ser encontrado modelos com a finalidade de classificação e previsão (DECKER & FOCARDI, 1995). Dentre os exemplos de classificação incluem: árvores de decisão, redes neurais e Naive-bayes. Já para os modelos de previsão incluem: regressão linear, ARIMA, LSTM, entre outros.

5.1.2. Aprendizado Não-Supervisionado

Processo de aprendizagem em que os algoritmos operam sem um "supervisor", portanto não se sabe qual deverá ser a saída desejada. Dessa forma, o conhecimento obtido a partir dos algoritmos refere-se à detecção de características e similaridades na base de dados. Assim, podemos dizer que esse tipo de aprendizagem não há um resultado conhecido, mas sim que irá ser descoberto (DECKER & FOCARDI, 1995).

Dentre os exemplos de modelos citamos: agrupamento ou clustering, detecção de anomalias, redes neurais geradoras adversárias.

5.1.3. Aprendizado Semi-Supervisionado

Processo de aprendizagem que utiliza a combinação das categorias de aprendizagem supervisionada e não supervisionada, A idéia do aprendizado semi-supervisionado é então utilizar os exemplos rotulados para se obter informações sobre o problema e utilizá-las para guiar o processo de aprendizado a partir dos exemplos não rotulados (BRUCE, 2001). Dentre os exemplos de modelos incluem: baseados em grafos e métodos heurísticos.

5.1.4. Aprendizado por Reforço

Segundo SUTTON & BARTO (1998) Processo de aprendizagem onde é realizada uma sequência de decisões por um “agente” que aprende a atingir uma meta em um ambiente incerto e complexo. Esse processo se baseia em tentativa e erro para encontrar uma solução, e para que a máquina se aperfeiçoe a cada tentativa, é realizado um sistema de recompensa e punição para aperfeiçoar as ações do agente, melhorando assim o resultado.

Conforme foi possível observar, o tipo de aprendizado de máquina mais apropriado para a resolução do problema é o aprendizado supervisionado, pois o atributo alvo de predição (adj close) da ação é conhecido, permitindo comparar o valor real com o valor predito.

Na próxima seção, será demonstrado a etapa de pré-processamento dos dados cujo objetivo é melhorar a qualidade dos dados que serão transferidos para os algoritmos de Machine Learning.

5.2. Pré-processamento dos Dados

Antes de submeter o conjunto de dados a um algoritmo é necessário realizar algumas modificações/transformações para otimizar o processo de aprendizado, nesse trabalho será realizada a normalização dos dados e seleção de features, descritas logo a seguir.

5.2.1. Normalização dos Dados

Prática cujo objetivo é alterar os valores das colunas numéricas no conjunto de dados para usar uma escala comum, sem distorcer diferenças nos intervalos de valores ou perda de informações. Neste caso, há duas fórmulas que podem ser utilizadas: Z-score ou Min-Max. Para este trabalho, será utilizado a função Min-Max, por meio dos métodos *MinMaxScaler()*, *fit_transform()*, e *transform ()*, todos da biblioteca *sklearn.*

```
scaler = MinMaxScaler()
X_train_scale = scaler.fit_transform(X_train)  # Normalizando os dados de entrada(treinamento)
X_test_scale  = scaler.transform(X_test)       # Normalizando os dados de entrada(teste)
```

Figura 17: Normalização de dados

5.2.2. Seleção de características/atributos (Feature selection)

É uma prática cujo objetivo é selecionar as características ou atributos mais relevantes para resolução do problema. Essa etapa é importante pois permite que o algoritmo trabalhe apenas com as características mais relevantes.

Existem diversos métodos que podem auxiliar nessa atividade, porém, foram utilizados neste trabalho os seguintes método: SelectKBest() e fit_transform(), da biblioteca sklearn. Após execução do código, as seguintes features foram utilizadas:

```
features.head()
```

	Open	High	Low	Close	Volume	Open_dolar	High_dolar	Low_dolar	Close_dolar
0	15.709090	15.812121	15.230303	15.230303	38071770	3.9608	4.0709	3.9590	4.0402
1	15.393939	15.624242	15.200000	15.357575	21093105	4.0403	4.0590	3.9883	4.0077
2	15.151515	15.654545	15.030303	15.296969	33520575	4.0071	4.0570	4.0040	4.0244
3	15.090909	15.327272	15.006060	15.006060	28766430	4.0248	4.0749	4.0216	4.0448
4	15.121212	15.303030	15.012121	15.012121	18087465	4.0453	4.0569	4.0050	4.0245

```
features_list = ('Open','High','Low','Close','Volume', 'Open_dolar','High_dolar','Low_dolar','Close_dolar')

k_best_features = SelectKBest(k='all')
k_best_features.fit_transform(features, label)
k_best_features_scores = k_best_features.scores_
raw_pairs = zip(features_list[1:], k_best_features_scores)
ordered_pairs = list(reversed(sorted(raw_pairs, key=lambda x: x[1])))

k_best_features_final = dict(ordered_pairs[:15])
best_features = k_best_features_final.keys()
print ('')
print ("Melhores features:")
print (k_best_features_final)
```

```
Melhores features:
{'Low_dolar': 131.08881549215477, 'Close_dolar': 105.83746189837713, 'Close': 101.6582093090275, 'Volume': 92.58984766588192,
'High_dolar': 79.34910714173184, 'Low': 75.48825504270295, 'High': 52.87722607609511, 'Open_dolar': 2.1021895286229775}
```

```
#separando as features escolhidas
features = df_full.loc[:,['Low_dolar', 'Close_dolar', 'Close','Volume', 'High_dolar']]
```

Figura 18: Seleção de Features

5.3. Treinamento de modelos

Além dos diversos tratamentos realizados no dataset para uma melhor performance dos algoritmos, a forma como os dados são apresentados ao processo de treinamento também pode influenciar o resultado. Dessa forma, dois métodos muito utilizados nesse processo são Holdout e K-Fold Cross-Validation. Esses métodos são utilizados de forma estratégica para evitar problemas como Overfitting e Underfitting.

5.3.1. Holdout

O método Holdout consiste em dividir o conjunto total de dados em dois subconjuntos mutuamente exclusivos, um para treinamento e outro para teste. O conjunto de dados pode ser dividido em quantidades iguais ou não (HAN & KAMBER, 1999). Uma proporção muito comum é considerar uma divisão de 70/30 ou 80/20 porcento, de modo que a maior parte fique para a etapa de treinamento.

Na imagem seguinte, é demonstrado um exemplo de funcionamento do método Holdout:

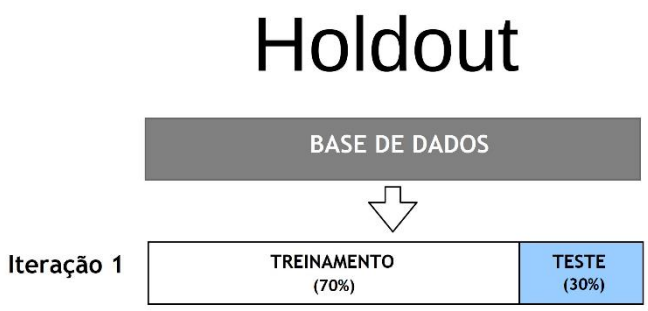

Figura 19: Exemplo de funcionamento do método Holdout

5.3.2. Validação Cruzada

O método "K-Fold Cross-validation" consiste em dividir a base de dados de forma aleatória em K subconjuntos (em que K é definido previamente) com aproximadamente a mesma quantidade de amostras em cada um deles. A cada iteração, treino e teste, um conjunto formado por K-1 subconjuntos são utilizados para treinamento e o subconjunto restante será utilizado para teste, gerando um resultado de métrica para avaliação, por exemplo, acurácia. Esse processo garante que cada subconjunto será utilizado para teste em algum momento da avaliação do modelo (MEDIUM, 2021).

Na próxima imagem, é apresentado um exemplo de funcionamento do método Cross-validation:

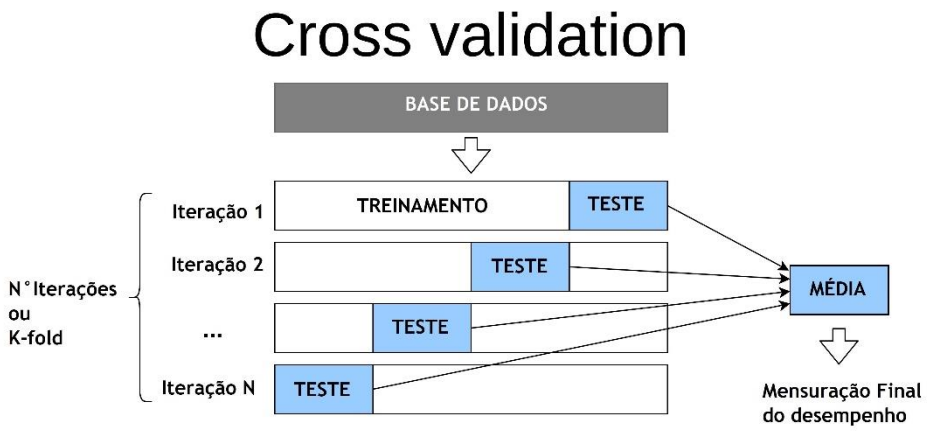

Figura 20: Exemplo de funcionamento do método Cross-validation

Para esse trabalho, será utilizado o método Holdout, com a separação do dataset em 70% para treino e 30% para teste, por meio dos seguintes comandos:

```
qtd_linhas = len(df_full)

qtd_linhas_treino= round(.70 * qtd_linhas)
qtd_linhas_teste= qtd_linhas - qtd_linhas_treino

print("Quant linhas de treino: 0:"+ str(qtd_linhas_treino))
print("Quant linhas de teste: " + str(qtd_linhas_treino) + ":" + str(qtd_linhas_treino + qtd_linhas_teste -1))
```

```
Quant linhas de treino: 0:953
Quant linhas de teste: 953:1360
```

```
#Separa os dados de treino e teste
X_train = features[:qtd_linhas_treino]
X_test = features[qtd_linhas_treino:qtd_linhas_treino + qtd_linhas_teste -1]

y_train = label[:qtd_linhas_treino]
y_test = label[qtd_linhas_treino:qtd_linhas_treino + qtd_linhas_teste -1]

print(len(X_train), len(y_train))

print(len(X_test), len(y_test))
```

```
953 953
407 407
```

Figura 21: Aplicação método holdout

5.4. Métricas de avaliação de desempenho

Para cada categoria de aprendizado, é fundamental que exista uma etapa de avaliação do modelo. Nesta etapa, podem ser utilizadas algumas técnicas, por exemplo, a avaliação por métricas a fim de maximizar a qualidade e minimizar o erro do modelo. A seguir, duas métricas que foram utilizadas neste trabalho.

5.4.1. Raiz do Erro Quadrático Médio (RMSE)

Essa métrica serve para verificar o erro entre o valor real e valor predito. Quanto mais próximo de zero for o valor obtido por esta métrica, melhor é o resultado da predição (HYNDMAN; KOEHLER, 2006).

Na imagem seguinte, é apresentado a fórmula do RMSE com a representação de cada valor.

$$RMSE = \sqrt{\frac{\sum(Q_{obs} - Q_{cal})^2}{N}}$$

Figura 22: Fórmula RMSE

Onde:

$Qobs$ representa o valor observado

$Qcal$ representa o valor calculado

N número de amostras utilizadas

5.4.2. Coeficiente de determinação

O Coeficiente de Determinação ou Coeficiente R², diz quanto o modelo está prevendo corretamente. O cálculo de Coeficiente R² envolve duas medidas:

MEDIDAS DO CÁLCULO DO COEFICIENTE R²		
Medida	**Descrição**	**Fórmula**
Soma Total dos Quadrados (STQ)	Mostra a variação de y em torno da própria média. É o somatório do quadrado das diferenças entre o valor real e a média dos valores.	$\sum(y - \bar{y})^2$ y é o valor real, y traço é a média
Soma dos Quadrados dos Resíduos (SQU)	É o somatório do quadro das diferenças entre o valor predito e o valor real.	$\sum(y - \hat{y})^2$ y é o valor real, y com chapéu é o valor predito

Ao final, a fórmula de determinação R² é o próprio SQR dividido pelo SQT (Figura 21). O resultado de R² varia entre 0 e 1, por vezes sendo expresso em termos percentuais (WIKIPEDIA, 2021).

$$R^2 = \frac{\sum(y - \bar{y})}{\sum(y - \hat{y})} = \frac{SQU}{SQT}$$

Figura 23: Fórmula do Coeficiente de Determinação

5.5. Algoritmos utilizados

Serão utilizados três modelos distintos para comparação e extração dos resultados: Regressão Linear, ARIMA e MLP.

A seguir, um breve resumo sobre dos conceitos sobre cada algoritmo de Machine Learning escolhido e justificativas de utilização.

5.5.1. Regressão Linear

Regressão linear é a verificação da existência de um relacionamento entre duas ou mais variáveis. Por exemplo, dado X e Y, quanto que X explica Y. Para isso, a regressão linear utiliza os pontos de dados para encontrar a melhor reta de ajuste para modelar essa relação (MONTGOMERY et al., 2001).

Com a reta encontrada é possível realizar a previsão do próximo ponto conforme distribuição dos dados. Na terminologia da análise de regressão, a variável que é prevista é dita variável dependente. Já a variável ou variáveis usadas para prever o valor da variável dependente são denominadas variáveis independentes (MONTGOMERY et al., 2001).

A regressão linear pode ser simples ou múltipla, e a diferença é que na regressão linear simples há apenas uma variável independente (X) ou explicativa para estimar a previsão da variável dependente (Y) ou prevista. Já na regressão linear múltipla utiliza-se mais de uma variável independente para realizar a previsão (MONTGOMERY E RUNGER, 2012).

A seguir as fórmulas de cada tipo de Regressão:

$$y = \beta_0 + \beta_1 x_1 + \varepsilon$$

Figura 24: Fórmula de regressão linear simples

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \varepsilon$$

Figura 25: Fórmula de regressão linear múltipla

onde:

y = variável dependente

β_i = parâmetros estimadores

x_i = variável independente

ε = erro referente a variabilidade em que y não é explicado por x

Nesse trabalho, será aplicado o modelo de Regressão Linear Múltipla, pois serão utilizados mais de uma variável independente para previsão da variável dependente, definidos no método K-Best. Além disso, é um método que realiza projeções e estuda a relação entre duas variáveis, que é aplicado ao problema proposto.

Código para execução do algoritmo e avaliação por métricas:

```
#treinamento usando regressão linear
lr = linear_model.LinearRegression()
lr.fit(X_train_scale, y_train)
pred_lr= lr.predict(X_test_scale)
cd =r2_score(y_test, pred_lr)
rmse_lr = sqrt(mean_squared_error(y_test, pred_lr))

print('------Linear Regression------')
print(f'Coeficiente de determinação:{cd * 100:.2f}')
print(f'RMSE: {rmse_lr}')
```

Figura 26: Algoritmo de Regressão Linear

Resultados obtidos com Regressão Linear:

Coeficiente de Determinação	RMSE
93.66%	1.03

5.5.2. ARIMA

O ARIMA (*Auto Regressive Integrated Moving Averages*) é um algoritmo matemático proposto por Box-Jenkins que visa captar o comportamento da correlação seriada ou autocorrelação entre os valores de série temporal, e com base nesse comportamento realizar previsões futuras (WERNER; RIBEIRO, 2003).

Segundo FAVA (2000), os modelos ARIMA resultam da combinação de três componentes denominados "filtros", sendo: componente auto-regressivo (AR), filtro de integração (I) e componente de médias móveis (MA)", tomando-se a forma usual ARIMA, representados pelos parâmetros (p, d, q), em que:

p – AR (autorregressão)

d – I (integração)

q – MA (média móvel)

O ARIMA possui algumas variações que utilizam outros parâmetros, por exemplo, os modelos Média Móvel Integrada Regressiva Sazonal (SARIMA), Média Móvel de Vetores (VMA), entre outros (FAVA, 2000).

Código para execução do algoritmo e avaliação por métricas:

```
# Divide o dataset em 70:30
training_data = y_train.values
test_data = y_test.values

history = [x for x in training_data]
model_predictions = []
N_test_observations = len(test_data)
for time_point in range(N_test_observations):
    model = ARIMA(history, order=(6,1,0))
    model_fit = model.fit(disp=0)
    output = model_fit.forecast()
    yhat = output[0]
    model_predictions.append(yhat)
    true_test_value = test_data[time_point]
    history.append(true_test_value)
```

```
cd_arima =r2_score(test_data, np.array(model_predictions))
RMSE_error = sqrt(mean_squared_error(test_data, np.array(model_predictions)))
print('------ARIMA------')
print(f'Coeficiente de determinação: {cd_arima * 100:.2f}')
print(f'RMSE: {RMSE_error}')
```

Figura 27: Algoritmo ARIMA

Resultados obtidos com ARIMA:

Coeficiente de Determinação	RMSE
97.26%	0.68

5.5.3. Redes Neurais Artificiais

Redes Neurais Artificiais (RNA) são técnicas computacionais que apresentam um modelo matemático inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento através da experiência. Uma RNA possui muitas unidades de processamento conectadas umas às outras. Essas unidades de processamento, geralmente, são conectadas por canais de comunicação que estão associados a determinado peso. As unidades fazem operações apenas sobre seus dados locais, que são entradas recebidas pelas suas conexões. O comportamento inteligente de uma RNA vem das interações entre as unidades de processamento da rede (FERNANDES, 2003).

A operação de uma unidade de processamento, proposta por MCCULLOCK E PITTS (1943), pode ser resumida da seguinte forma:

- Sinais são apresentados à entrada;
- Cada sinal é multiplicado por um número, ou peso, que indica a sua influência na saída da unidade;
- É feita a soma ponderada dos sinais que produz um nível de atividade;
- Se este nível de atividade exceder um certo limite (*threshold*) a unidade produz uma determinada resposta de saída (variável classificada ou predita).

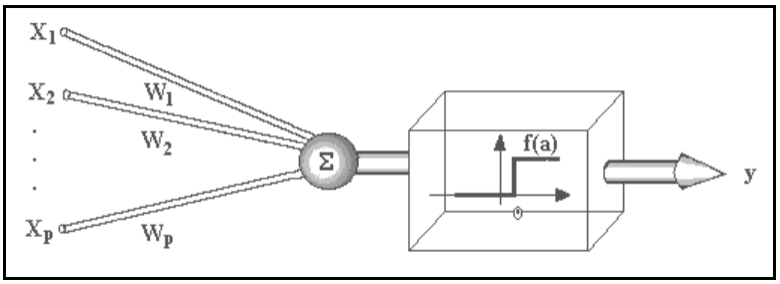

Figura 28: Esquema de unidade de processamento proposto por McCullock e Pitts

A arquitetura das redes neurais artificiais pode ser formada por uma ou mais camadas de neurônios. Estas camadas estão formadas por neurônios enfileirados e ligados entre si.

Apesar de ser conhecida como rede de uma camada, as redes neurais de uma camada são constituídas da camada de entrada (input layer) e camada de saída (ouput layer). Isso porque somente na camada de saída é que há o processamento das unidades neuronais. A camada de entrada é responsável apenas pelo envio das informações.

Na imagem seguinte é apresentado um exemplo com 4 neurônios na camada de entrada e 4 neurônios na camada de saída.

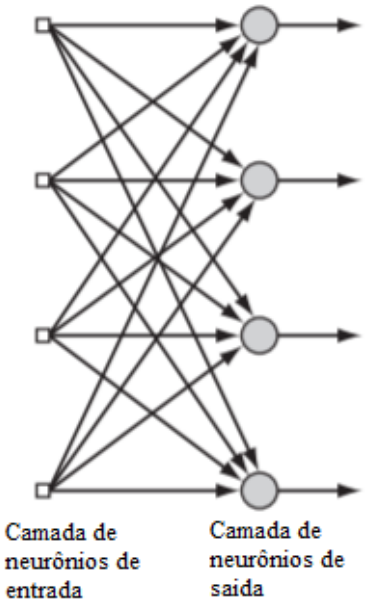

Figura 29: Exemplo de arquitetura de Rede Neural com uma camada

Já as redes neurais Multilayer Perceptron (MLP) são constituídas de três ou mais camadas, sendo a primeira denominada camada de entrada (input layer). Da segunda camada em diante, tem-se a camada escondida ou oculta (hidden layers). A última camada é denominada camada de saída (ouput layer). Este tipo de rede tem sido aplicado com sucesso em diversas áreas na solução de problemas complexos, desempenhando tarefas de classificação de padrões, reconhecimento de voz, reconhecimento de imagem e predição de valores.

Na imagem seguinte é apresentado um exemplo de arquitetura de redes neurais MLP.

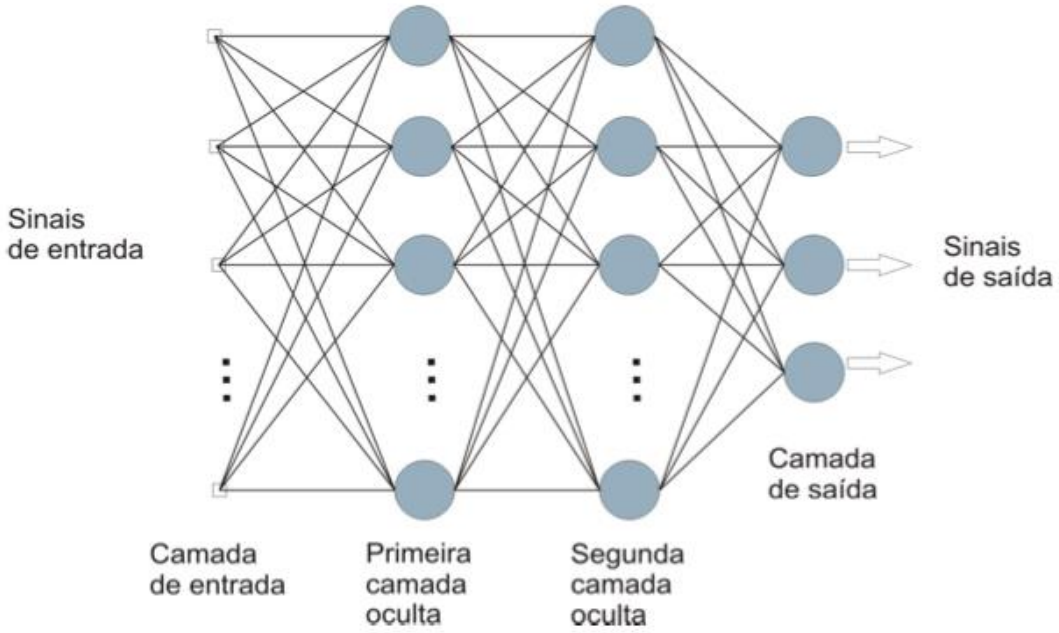

Figura 30: Exemplo de Arquitetura Multilayer Perceptron (MLP)

Nesse trabalho, será utilizado o MLP com uma configuração padrão de rede MLP e outra com ajuste de hiperparâmetros.

5.5.3.1. MLP com configuração padrão

Código para execução do algoritmo com a configuração padrão:

```
#rede neural
rn = MLPRegressor(max_iter=200)

rn.fit(X_train_scale, y_train)
pred_rn= rn.predict(X_test_scale)
```

```
cd_mlp1 = rn.score(X_test_scale, y_test)
rmse_mlp1 = sqrt(mean_squared_error(pred_rn, y_test))

print('------MLP------')
print(f'Coeficiente de determinação:{cd * 100:.2f}')
print(f'Root Mean Squared Error is {rmse_mlp1}')
```

Figura 31: Algoritmo MLP

Pelo fato de os valores de pesos serem obtidos de forma aleatória no início da execução do algoritmo, para diferentes execuções pode-se obter diferentes resultados. Dessa forma, a tabela a seguir apresenta os resultados de cinco execuções, bem como a média das métricas obtidas.

Resultados obtidos com MLP:

N° execução	Coeficiente de Determinação	RMSE
1	93.66%	1.36
2	93.66%	1.06
3	93.66%	1.09
4	93.66%	1.15
5	93.66%	1.69
Média:	**93.66%**	**1.27**

5.5.3.2. MLP com ajuste de hiperparâmetros

Para obter uma melhor performance da execução do algoritmo, modificou-se alguns hiperparâmetros: “hidden_layer_sizes”, “activation”, “solver”, “alpha”, “learning_rate”. A seguir, código para execução do algoritmo com a configuração ajustada:

```
rn_mlp = MLPRegressor(max_iter=1000)

parameter_space = {
        'hidden_layer_sizes': [(i,) for i in list(range(1, 21))],
        'activation': ['tanh', 'relu'],
        'solver': ['sgd', 'adam', 'lbfgs'],
        'alpha': [0.0001, 0.05],
        'learning_rate': ['constant', 'adaptive'],
    }

search = GridSearchCV(rn_mlp, parameter_space, n_jobs=-1, cv=5)


search.fit(X_train_scale,y_train)
clf = search.best_estimator_
pred_mlp= search.predict(X_test_scale)

cd = search.score(X_test_scale, y_test)
rmse_mlp2 = sqrt(mean_squared_error(pred_mlp, y_test))

print(f'Coeficiente de determinação:{cd * 100:.2f}')
print(f'Root Mean Squared Error is {rmse_mlp2}')
```

Figura 32: Algoritmo MLP com ajustes de hiperparâmetros

Novamente, pelo fato de os valores de pesos serem obtidos de forma aleatória no início da execução do algoritmo, para diferentes execuções pode-se obter diferentes resultados. Dessa forma, a tabela a seguir apresenta os resultados de cinco execuções, bem como a média das métricas obtidas para o MLP ajustado.

Resultados obtidos com MLP ajustado dos hiperparâmetros:

Execução	Coeficiente de Determinação	RMSE
1	76.82%	1.98
2	92.27%	1.14
3	67.16%	2.35
4	77.18%	1.80
5	91.20%	1.09
Média:	**68,68%**	**1.67**

6. Interpretação dos Resultados

Nesta seção são apresentados os resultados obtidos na etapa de análise e exploração de dados, vantagens e desvantagens de cada algoritmo escolhido, e os resultados da execução dos algoritmos de Machine Learning.

Na etapa de análise e exploração dos dados foram observados os comportamentos dos ativos utilizados (ITUB4 e dólar). Alguns comportamentos foram verificados, como por exemplo a desvalorização da ação no período inicial da pandemia, no primeiro semestre de 2020. No mesmo período, verificou-se por meio de recursos gráficos o aumento do dólar que passou a assumir valores acima de 5 reais. Esse aumento, dentre outras razões, também foi influenciado por motivos de pandemia.

Dessa forma, podemos constatar que diversas variáveis influenciam o comportamento dos ativos, por isso é de grande importância sempre relacionar os dados com outras informações. No caso deste trabalho, utilizou-se o dólar como um dado extra que pudesse trazer informações relevantes para a tarefa de predição executada.

Com relação à tarefa de predição, foram utilizados os algoritmos Regressão Linear, ARIMA e MLP, mas outros algoritmos poderiam ser utilizados para fins comparativos. A seguir tem-se uma tabela destacando os pontos principais de vantagens e desvantagens de cada técnica.

VANTAGENS E DESVANTAGENS DOS MODELOS DE MACHINE LEARNING		
Modelo	**Vantagens**	**Desvantagens**
Regressão Linear	✓ Rápida execução; ✓ Requer pequeno número de suposições; ✓ Alto grau de confiabilidade.	✓ Os dados devem ser independentes; ✓ Limitada a relacionamentos lineares.
ARIMA	✓ Excelente para tarefas de previsões; ✓ Flexível e pode representar inúmeras séries.	✓ Assumem uma relação linear, o que nem sempre acontece; ✓ não pode lidar com dados sazonais.
Redes MLP	✓ Capacidade de lidar com informações incompletas e ruidosas; ✓ Não é necessária informação sobre o ambiente a *priori*, pois o aprendizado é feito através da apresentação de padrões à rede; ✓ Processamento paralelo; ✓ Habilidade de aprender por meio de exemplos.	✓ Tempo de treinamento é o mais demorado; ✓ Dificuldade em determinar a arquitetura ideal; ✓ Alto consumo de recurso computacional.

Cada algoritmo possui um processo de treinamento e por isso obteve-se resultados diferentes. Na tabela a seguir é demonstrado os resultados da avaliação por métricas de cada modelo:

Modelo	Coeficiente de Determinação	RMSE
Regressão linear	93.66%	1.03
ARIMA	97.26%	0.68
MLP(*)	93.66%	1.27
MLP c/ ajustes(*)	68,68%	1.67

(*) resultados com base na média de 5 execuções.

Conforme tabela acima, obteve-se um melhor desempenho ao utilizar o algoritmo ARIMA, com RMSE de 0.68 e Coeficiente de Determinação de 97,26%. Importante ressaltar que esse resultado diz respeito a avaliação do modelo com o subconjunto do dataset destinado a teste.

Para validar as estratégias e os algoritmos utilizados, realizou-se a predição do dia seguinte, a fim de verificar como seria o resultado dos algoritmos após colocá-los em produção.

Na próxima imagem é apresentado o resultado da etapa de validação de cada modelo em relação à tarefa de predição.

```
#executando a predição

print("Valor real: 30.119904")

previsao=scaler.transform(valor_novo)

pred=lr.predict(previsao)
print(f'Valor predito pela Regressão Linear {pred}')

print(f'Valor predito pelo ARIMA {yhat}')

pred=rn.predict(previsao)
print(f'Valor predito pela MLP {pred}')

pred=search.predict(previsao)
print(print(f'Valor predito pela MLP Ajustado {pred}'))
```

```
Valor real: 30.119904
Valor predito pela Regressão Linear [29.7992835]
Valor predito pelo ARIMA [30.92646894]
Valor predito pela MLP [31.22327348]
Valor predito pela MLP Ajustado [29.54557311]
None
```

Figura 33: Previsão do preço da ação

Conforme resultado da validação dos modelos, é possível observar que o algoritmo que melhor teve desempenho foi a Regressão linear, pois o valor real é 30.11 e o valor predito foi de 29.79. Essa diferença é obtida devido à dificuldade que o modelo tem em realizar uma predição exata. Inclusive, essa dificuldade já é apresentada na etapa de teste onde há a presença de um erro.

A seguir é apresentado um gráfico comparando desempenho na etapa de validação de cada algoritmo.

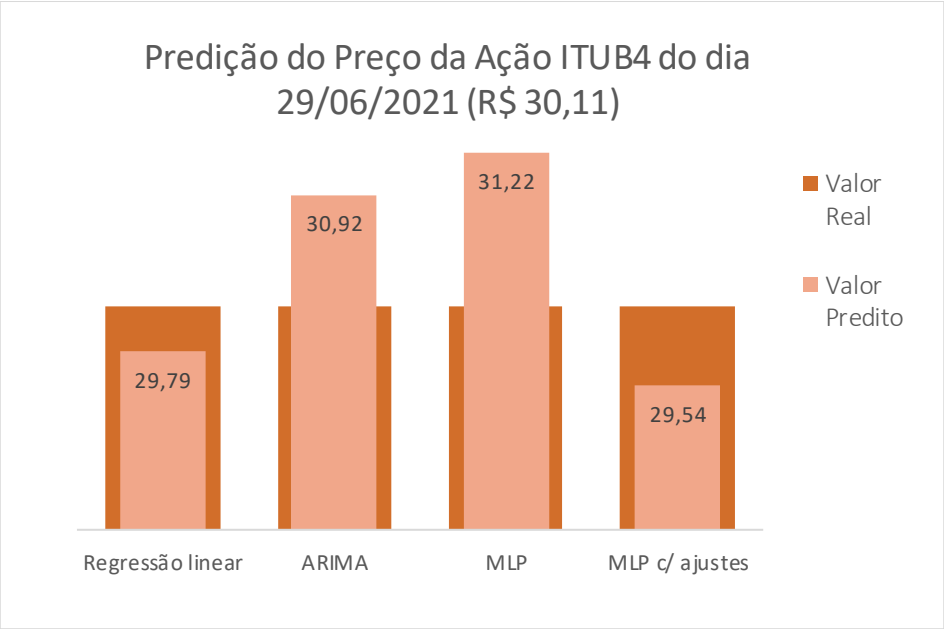

Figura 34: Comparação da Predição dos Algoritmos

Conforme foi possível observar no decorrer do trabalho, os modelos desenvolvidos e testados podem auxiliar os investidores, sendo uma ferramenta útil no processo de compra e venda de ações no curto prazo. Recomenda-se que esses modelos utilizados sejam complementados com mais informações que subsidiem a tomada de decisão, por exemplo, o enriquecimento da base de dados, inclusão de mais fontes de dados, a inclusão de mais features para treinamento e teste, e até mesmo o uso de dados de mídias sociais para análise de sentimento dos usuários sobre notícias financeiras.

Dessa forma, este trabalho não tem como objetivo indicar a compra ou venda de determinada ação, também não recomenda que a tomada de decisão seja realizada baseada nas abordagens desenvolvidas nesse trabalho. Todas as abordagens executadas são apenas para fins didáticos.

7. Apresentação dos Resultados

Para a apresentação dos resultados obtidos, foi utilizado o modelo de Canvas, proposto por Vasandani, que resume os resultados encontrados nesse trabalho de Ciência de dados:

<table>
<tr><td colspan="3">Título: Predição do preço de Ação na Bolsa de Valores – B3</td></tr>
<tr>
<td>1 Descrição do Problema/pergunta

Qual abordagem, utilizando Machine Learning, apresenta melhor resultado na predição de preço da ação, de modo que seja uma ferramenta útil para auxiliar no processo de tomada de decisão para compra ou venda de ações de curto prazo?</td>
<td>2 Resultados e Predições

Predição do atributo “adj close” um dia a frente.
• Valor real: 30.11
• Valores preditos:
Regressão linear: 29.79
ARIMA: 30.92
MLP: 31.22
MLP ajustado: 29.54</td>
<td>4 Aquisição dos Dados

Fontes de dados:
- Yahoo Finance
- Investing

Métodos de extração:
• download(), da biblioteca yfinance.
• get_currency_cross_historical_data(), da biblioteca investpy.</td>
</tr>
<tr>
<td>4 Modelos

Aprendizado supervisionado, Utilizando os algoritmos:

- Regressão Linear;

- ARIMA;

- MLP;

- MLP c/ ajuste hiperparâmetros.</td>
<td>5 Avaliação do Modelo

Avaliação segundo as métricas:
• RMSE:
Regressão linear: 1.03
ARIMA: 0.68
MLP: 1.27
MLP ajustado: 1.67

• Coeficiente de Correlação:
Regressão linear: 93,66%
ARIMA: 97,26%
MLP: 93,66%
MLP ajustado: 68,68%</td>
<td>6 Preparação dos Dados

Remoção de valores nulos e atributos não utilizados;
Isnull().sum()
Drop()

Integração dos Datasets:
merge();

Seleção de Features:
SelectKBest();

Normalização dos Dados:
MinMaxScaler(),
fit_transform(),
transform ().</td>
</tr>
</table>

8. Links

Link para o vídeo: https://www.youtube.com/watch?v=zTSEo4MjNOc

Link para o repositório: https://github.com/alisondhm/DataScience.git

REFERÊNCIAS

ANACONDA. **Anaconda Navigator**. Apresenta conteúdo da ferramenta Anaconda Navigator. Disponível em: https://docs.anaconda.com/anaconda/navigator/ Acesso em: 12/08/2021.

BRUCE, Rebbeca. **A bayesian approach to semi-supervised learning.** In M. Ishizuka & A. Satter (Eds.), Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium — NLPRS-2001, Tokyo, Japan. Springer Verlag, 2001.

DECKER, Karsten M. & FOCARDI, Sergio. **Technology overview: A report on data mining.** Technical Report CSCS TR-95-02, CSCS-ETH, Swiss Scientific Computing Center, 1995.

FERNANDES, Anita Maria da Rocha. **Inteligência Artificial Noções Gerais.** Florianópolis: Visual Books, 2003.

FAVA, V. **Manual de econometria**. In: VASCONCELOS, M. A. S.; D. São Paulo: Editora Atlas, 2000.

JUPYTER. **Projeto Jupyter**. Apresenta conteúdo do Projeto Jupyter. Disponível em: https://jupyter.org/ . Acesso em: 12/08/2021.

HAN, Jiawie e KAMBER, Micheline. **Data mining: concepts and techniques.** The Morgan Kaufmann Series in Data Management Systems, Jim Gray, Series Editor Morgan Kaufmann Publishers, 1999.

HAYKIN, S. **Redes neurais princípios e prática**. 2. ed. Porto Alegre, RS, Brasil: Bookman Companhia, 2001.

HAYKIN, S. **Neural networks and learning machines**. McMaster University, Canadá: Pearson Prentice Hall, 2008.

HYNDMAN, R. J.; KOEHLER, A. B. **Another look at measures of forecast accuracy. International Journal of Forecasting,** v. 22, no. 4, p. 679-688, 2006.

INFOMONEY. **Infomoney**. Empresa especializada em conteúdo sobre investimentos, mercado e negócios. Disponível em: https://www.infomoney.com.br/. Acesso em: 02/07/2021.

INVESTPY. **Investing**: API Disponível em: https://investpy.readthedocs.io/api.html. Acesso em: 03/07/2021.

MCCULLOCH, Warren s. & WALTER, Pitts. **A logical calculus of the ideas immanent in nervous activity.** The Bulletin of Mathematical Biophysics, vol. 5, no. 4, 1 Dec. 1943, pp. 115-33, doi:10.1007/BF02478259.

MEDIUM. Medium. Disponível em: https://medium.com/@edubrazrabello/cross-validation-avaliando-seu-modelo-de-machine-learning-1fb70df15b78. Acesso em: 10/07/2021.

MITCHELL, TOM M. **Machine Learning**. Cidade de New York, EUA, Editora McGraw-Hill, 1997.

MITCHELL, TOM M. **The role of unlabeled data in supervised learning.** In Proceedings of the Sixth International Colloquium on Cognitive Science, San Sebastian, Spain, 1999.

MONTGOMERY, D.; VINING, G., PECK, A. **Introduction to Linear Regression Analysis.** John Wiley & Sons New York, New York, 2001.

MONTGOMERY, D.; RUNGER G. **Estatística aplicada e probabilidade para engenheiros.** LTC, São Paulo, 2012.

PYTHON. **Python**. Apresenta conteúdo sobre Python. Disponível em: https://www.python.org/ . Acesso em: 12/08/2021.

SUTTON, Richard S. & BARTO, Andrew G. **Reinforcement Learning: An Introduction.** Cambridge: MIT Press, 1998.

WERNER, L; RIBEIRO, J. L. D. **Previsão de demanda: uma aplicação de modelos Box-Jenkins na área de assistência técnica de computadores pessoais.** Gestão & Produção, v.10, n.1, p. 47-67, 2003.

WIKIPEDIA. **Coeficiente de Determinação**. Apresenta conteúdo o método de coeficiente de determinação. Disponível em: https://pt.wikipedia.org/w/index.php?title=Coeficiente_de_determina%C3%A7%C3%A3o&oldid=58684266. Acesso em: 12/08/2021.

YAHOO. **Yahoo! Finance**. Apresenta conteúdo da API da Yahoo Finance Disponível em: https://finance.yahoo.com/quotes/API,Documentation/view/v1/. Acesso em: 02/07/2021.

YUXING, YAN. **Python for Finance Second Edition**. Cidade de Birmingham, UK, Editora Packt, 2017.