

ANÁLISE DA INTELIGÊNCIA ARTIFICIAL APLICADA PARA OTIMIZAÇÃO DA CARTEIRA DE AÇÕES PREVISTAS NO IBOVESPA

Viviane Alves de Lima - Universidade Anhembi Morumbi - vnlvih@gmail.com
Bruno M.C. Duarte - Universidade Anhembi Morumbi - brunomanoel.cduarte@gmail.com
Ana Clara S. H. de Andrade - Universidade Anhembi Morumbi - anaclarsh@gmail.com
Beatriz Martins Silva - Universidade Anhembi Morumbi - beeeatrizmartinssilv@gmail.com
Alef Ribeiro dos Santos - Universidade Anhembi Morumbi - felars@gmail.com

Resumo

Considerando a instabilidade econômica atual no Brasil, juntamente com a crise política e social, os investidores estão expostos a um ambiente de incertezas ainda maiores. Dessa maneira, precisa-se analisar criteriosamente as situações que estão sujeitos antes de tomar a decisão de investir e qual aplicação realizar. Por isto, é de suma necessidade avaliar valores a serem investidos, nível de risco que estão dispostos a correr, retorno que estão almejando, prazo de investimento e ainda, estar ciente da volatilidade desses investimentos. Perante a necessidade de realizar estes passos, os investidores possuem ferramentas de análise de investimentos confiáveis – uma vez que se encontram em um ambiente de incertezas, juntamente com os riscos oferecidos – faz com que estudos como o de Markowitz (1952) sejam base para o desenvolvimento de métodos de análise e tomada de decisão no mercado financeiro. Após a contextualização do estudo, definiu-se, como principal objetivo da pesquisa, criar uma carteira de investimentos visando a análise retroativa de resultados de ações, decidiu-se otimizar a carteira de ações presentes no IBOVESPA, entre os anos de 2020 a 2021 utilizando o modelo *machine learning*. Acredita-se na importância do uso de tecnologias relacionadas com a inteligência artificial que corroboram com a tomada de decisão mais assertiva e com mitigação de risco quando se trata de ações e para analisar automaticamente os dados, analisando um grande volume de informações para encontrar a melhor combinação de ações em uma carteira de investimento. Deve-se destacar que o presente estudo assim como outros efetuados na área apresentam limitações para o método de otimização de carteira, uma delas tendo em vista que o retorno histórico pode não vir a acontecer no futuro. O presente artigo contribuiu para a apresentação e aplicação da ferramenta de auxílio na análise do mercado financeiro, tendo em vista que vários investidores não possuem conhecimento de práticas como essa e há a necessidade em possuir ferramentas que auxiliam na tomada de decisão.

Palavras-Chaves: IBOVESPA, *MACHINE LEARNING*, INTELIGÊNCIA ARTIFICIAL E MERCADO FINANCEIRO.

1. Introdução

Há muitos séculos, a sociedade cria maneiras de fazer com que o dinheiro gere mais dinheiro. Segundo o livro “Introdução à economia” (2013), estima-se que há mais de cinco mil anos, na antiga Mesopotâmia, estas atividades já ocorriam. Em 1500, os navios mercantes levantaram fundos para dividir os lucros obtidos com investidores que aplicavam dinheiro em suas viagens. As recompensas poderiam ser enormes, mas o risco desse negócio também era muito alto.

Dentro deste cenário, a solução foi compartilhar o risco com mais investidores, e foi esta necessidade que impulsionou o surgimento das empresas com capital aberto. Neste novo modelo, os investidores aplicavam dinheiro em uma empresa e tornavam-se co-titulares do seu estoque e direito de uma porção dos lucros. Entre 1552 a 1571 ocorreu a construção da Bolsa da Antuérpia e da Bolsa de Londres para que os acionistas pudessem comprar e vender ações de capital aberto.

Com o avanço do mercado e crescimento de investidores em potencial, o mercado financeiro ganhou forma e hoje possui-se diversas Bolsas de Valores ao redor no mundo. Em 1967, com intuito de aprimorar a estrutura do mercado financeiro no Brasil e a demanda de se ter mais instrumentos para avaliação de mercado, foi criado o Ibovespa. O índice Ibovespa é composto por cinquenta ações negociadas na Bolsa de Valores, Mercadorias e Futuros de São Paulo. É o mais antigo e tradicional indicador do Brasil sobre o comportamento do preço médio das ações. Segundo a Associação Nacional dos Bancos de Investimentos (ANBID), o volume financeiro atingiu em 2007, R\$ 1 trilhão, representando mais de 60% do total de captação no sistema financeiro, considerando os depósitos a prazo, à vista e poupança, tornando o país emergente em 10º lugar no ranking mundial.

Ao se interessar por investimentos, deve-se compreender sobre as carteiras de ações. Vargas e Leal Cassettari, A., (2006) dizem que “uma das mais importantes questões das teorias financeiras diz respeito à administração eficiente de carteiras”. A carteira de ações é um conjunto de ações das quais se investe como o propósito de lucrar a partir da valorização do preço das ações de uma série de empresas. Ao construir uma carteira de ações é importante que uma análise crítica seja executada para obter-se maior assertividade na tomada de decisão e conseqüentemente lucratividade.

Um outro pilar valioso do mercado financeiro é a inteligência artificial que é usada como ferramenta para tomadas de decisões, uma vez que possui uma incrível capacidade na análise de dados e fornecimento de insights precisos e confiáveis. Segundo Guitta Pessis-Pasternak (2001) suas percepções são úteis para estratégias de investimentos e por isto cada vez mais procura-se modos de utilizar os benefícios desta avançada tecnologia para otimização de resultados e lucros.

Além de analisar um grande volume de dados é interessante que os sistemas possam aprender com estas informações. O conceito de *Machine Learning* (Aprendizado de Máquina) foi criado por Arthur Samuel em 1959, de forma que ele afirma: “aprendizado de máquina é um campo de estudo que dá aos computadores a capacidade de aprender sem ser programado de forma explícita”. Neste contexto, a inteligência artificial realiza a análise de dados e automatiza a construção de modelos analíticos, identificando padrões e possíveis tomadas de decisão com o mínimo de intervenção humana. Este modelo deve ser criado a partir do tratamento dos dados, criação de algoritmos e modelagem.

Dentro desta abordagem, este artigo propõe-se analisar a aplicação da inteligência artificial para a otimização da carteira de ações previstas no IBOVESPA.

2. Referencial teórico

2.1. Carteira de ações

Na literatura econômica é comum ler-se sobre as carteiras de investimentos. Seja carteira de fundos de investimentos ou carteira de investidores pessoas físicas. Segundo o BTG Pactual (2017) Uma Carteira de investimentos é o conjunto de aplicações do investidor, seja pessoa física ou jurídica. Também chamada de cesta e portfólio de investimentos, ela reúne todos os ativos financeiros que você escolhe para fazer seu dinheiro crescer, tanto em renda fixa quanto variável.

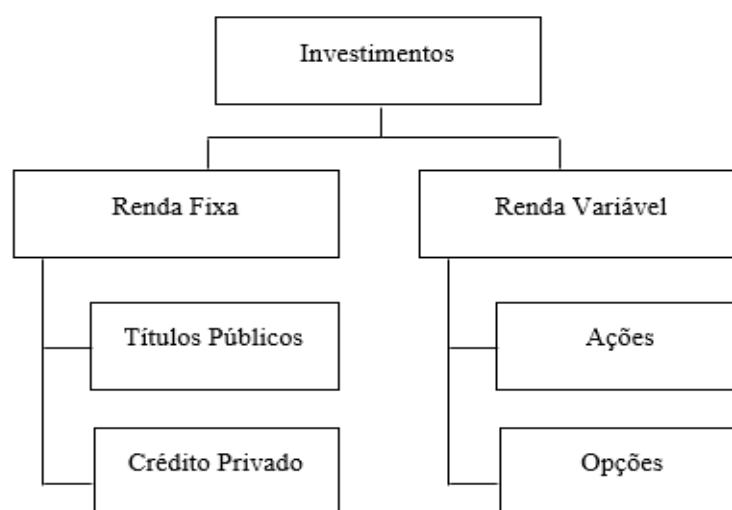
Esta pesquisa irá focar no mercado de Renda Variável. Mais especificamente nas ações de empresas pertencentes ao IBOVESPA.

Quadro 1: Segmentos do mercado financeiro

Tipo	Descrição	Exemplo
Renda Fixa	Segundo a XP Investimentos (2021) Renda Fixa é o investimento realizado diretamente em Títulos Públicos e Privados de Renda Fixa. Quando se compra um título de Renda Fixa, está emprestando dinheiro ao emissor do papel, que pode ser um banco, uma empresa ou mesmo o Governo	Títulos Públicos LFT NTN-B Títulos Privados CDB Debêntures CRI / CRA LCI / LCA
Renda variável	Segundo a Genial Investimentos (2018) Renda Variável é um tipo de investimento no qual não é possível definir o quanto seu dinheiro vai render em determinado período. Por esse motivo, costumam ser aplicações de maior risco, mas, por outro lado, o rendimento pode ser muito maior e em um espaço de tempo mais curto	Ações Opções

Fonte: Elaborado pelos autores (2021)

Figura 1 – Tipos de investimentos



Fonte: Elaborado pelos autores (2021)

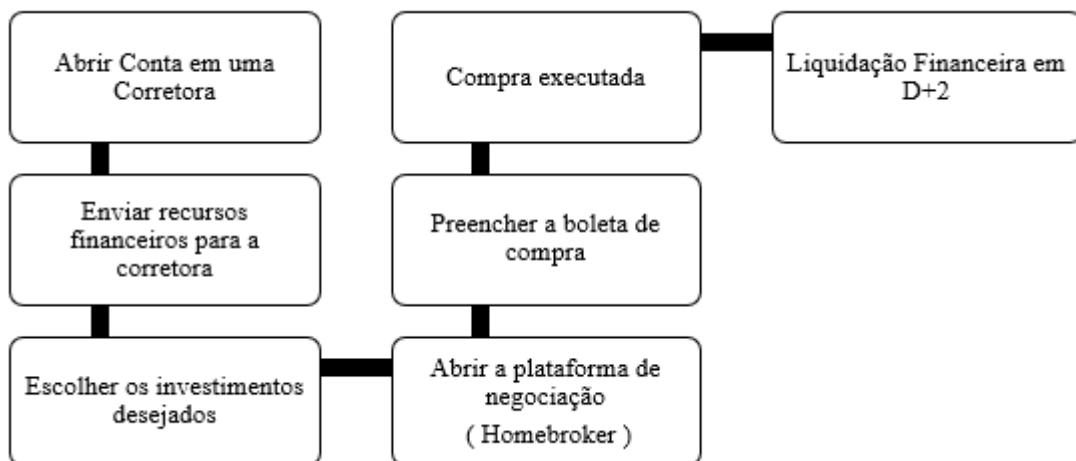
No Brasil pode-se comprar e vender ações através da Bolsa de Valores Brasileira, chamada B3. E segundo a própria B3 (2021), ações são valores mobiliários emitidos por sociedades anônimas representativos de uma parcela do seu capital social. Em outras palavras, são títulos de propriedade que conferem a seus detentores a participação na sociedade da empresa.

Quando as ações são emitidas por companhias abertas ou assemelhadas, são negociados em bolsa de valores ou no mercado de balcão. Atualmente existem mais de 400 empresas de

capital aberto na bolsa de valores brasileira. São empresas de diversos tamanhos e setores de atuação.

Ao investir em algumas dessas empresas, é formada uma carteira de investimentos ou carteira de ações.

Figura 2: Processo de investimento



Fonte: Elaborado pelos autores (2021)

2.2.1. Teoria do portfólio

Segundo Markowitz (1952) o procedimento necessário para a escolha de um portfólio ou carteira de investimentos é baseado na diversificação das ações que compõem determinada carteira, sendo composta por ações cujos retornos esperados têm baixa correlação entre si, garantindo um desvio padrão ou risco baixo e maior retorno esperado.

2.2.2. Índice sharpe

O Índice Sharpe é utilizado para avaliar o quão bem um investimento terá de retorno considerando um determinado risco. De modo que, quanto maior o valor desse índice mais atrativo é o investimento. O cálculo deste índice é apresentado abaixo:

$$\text{Índice Sharpe} = \frac{(R_p - R_f)}{\sigma_P} \quad (1)$$

Onde:

Rp: retorno do portfólio;

Rf: taxa livre de risco;

σ_P : desvio padrão do ativo ou volatilidade do ativo.

Este índice é utilizado para fazer a seleção da carteira mais estável, que irá apresentar menor volatilidade e maior retorno esperado.

2.3. Inteligência artificial

De acordo com Luger & Stubblefield (1993), Inteligência Artificial é uma vertente da Ciência da Computação, onde se estuda os meios de inteligência articulado por máquinas. Tem como princípio e base imitar ou simular a inteligência e o comportamento humano, como a percepção da linguagem, a aprendizagem, o raciocínio etc., simulando nossa forma de assimilar informações (FEIGENBAUM, 1981). O Quadro 2 apresenta as principais técnicas de IA segundo Hurwitz e Kirsch (2018):

Quadro 2 - Técnicas de Inteligência Artificial

	Técnica	Definição/aplicação
1.	Rede bayesiana ou “ <i>bayesian network</i> ”	Matematicamente uma rede bayesiana é uma representação compacta de uma tabela de conjunção de probabilidades do universo do problema. Do ponto de vista de um especialista, redes bayesianas constituem um modelo gráfico que representa de forma simples as relações de causalidade das variáveis de um sistema (KORB e NICHOLSON, 2010). A técnica de IA rede bayesiana apresenta-se como uma técnica promissora para os processos de gestão hospitalar em razão da capacidade de analisar sistemas complexos e suas inter-relações.

2.	<i>Clustering</i>	A técnica de “ <i>clustering</i> ” que consiste em um agrupamento automático de dados segundo a sua semelhança segundo Strickland (2014).
3.	Mineração dos dados ou “ <i>data mining</i> ”	<i>Data mining</i> ou mineração de dados é a técnica de IA que consiste na organização dos dados na busca por padrões, associações, mudanças e/ou anomalias relevantes. A aplicação da mineração de dados se dá em grandes bases de dados contidos dentro do <i>data warehouse</i> e pode contribuir com a tomada de decisão pelas instituições de forma mais rápida e com maior grau de confiança (CARDOSO e MACHADO, 2008).
4.	Aprendizagem Profunda ou “ <i>deep learning</i> ”	A técnica de “ <i>deep learning</i> ” ou aprendizagem profunda refere-se ao número de camadas através das quais os dados são transformados, onde cada camada sucessiva usa a camada anterior como entrada formando uma hierarquia de conceitos, 2014 (DENG et al., 2014).
5.	Sistemas especialistas ou “ <i>expert system</i> ”	Os sistemas especialistas ou “ <i>expert system</i> ” são programas com o objetivo de simular o raciocínio de um profissional “ <i>expert</i> ” em alguma área específica do conhecimento (HJERPPE e OLANDER, 1989)
6.	Lógica difusa ou “ <i>fuzzy logic</i> ”	A lógica difusa ou “ <i>fuzzy logic</i> ” é a técnica de IA empregada para descrever a inter-relação entre os processos hospitalares e graduar a sua relevância na qual os valores lógicos das variáveis podem ser qualquer número real entre 0, correspondente ao valor falso e 1, correspondente ao valor verdadeiro. A lógica difusa lida com o conceito de verdade parcial, onde o valor verdade pode compreender entre o valor completamente verdadeiro e completamente falso (GOMIDE e GUDWIN, 1994).
7.	Análise de imagens ou “ <i>image analysis</i> ”	A análise de imagem ou “ <i>image analysis</i> ” é a técnica de IA que estuda automaticamente uma imagem com o objetivo de extrair informações úteis e relevantes a partir dela. Para este a composição deste trabalho, não foram identificados artigos dentro do escopo na literatura. Sua aplicação concentra-se nos processos de diagnósticos diversos (SOLOMON e BRECKON, 2011).

8.	Aprendizado de máquina ou " <i>machine learning</i> "	A técnica " <i>machine learning</i> " ou aprendizado de máquina é a técnica de IA onde o aprendizado automático da máquina sem ser programada explora o estudo e a construção de algoritmos capazes de aprender com seus erros e fazer previsão sobre dados. Tais algoritmos operam construindo um modelo a partir de inputs amostrais a fim de fazer previsões ou decisões guiadas pelos dados ao invés de simplesmente seguindo inflexíveis e estáticas instruções programadas (MONARD e BARANAUSKAS, 2003).
9.	Redes Neurais ou " <i>neural networks</i> "	Redes neurais artificiais (RNAs) ou " <i>neural network</i> " é a técnica de IA inspirada no sistema nervoso central de um animal (em particular o cérebro) que são capazes de realizar o aprendizado de máquina bem como o reconhecimento de padrões. Redes neurais artificiais geralmente são apresentadas como sistemas de "neurônios interconectados, que podem computar valores de entradas", simulando o comportamento de redes neurais biológicas (VAZ, 2019).
10.	Floresta aleatória ou " <i>random forest</i> "	A técnica de IA <i>random forest</i> ou floresta aleatória é um algoritmo de aprendizagem supervisionada que cria múltiplas árvores de decisão que podem ser utilizadas para tanto para tarefas de classificação quanto para regressão (HO, 1995).
11.	Máquina de vetores de suporte ou " <i>support vector machine</i> "	O <i>support vector machine</i> (SVM) ou máquina de vetores de suporte é a técnica de IA para um conjunto de métodos do aprendizado supervisionado que analisam os dados e reconhecem padrões usados para classificação e análise de regressão. O SVM padrão toma como entrada um conjunto de dados e prediz, para cada entrada dada, qual de duas possíveis classes a entrada faz parte, o que faz do SVM um classificador linear binário não probabilístico (CORTES e VAPNIK, 1995).

Fonte: Adaptado de Franzini Filho (2020).

2.3.1. *Machine learning* no mercado de ações

O mercado de ações é passível a rápidas mudanças e variações aleatórias, tornando a tarefa de tentar entender como determinadas ações irão se comportar no futuro uma tarefa difícil. Embora exista essa barreira, nos últimos anos, muitos estudiosos vêm buscando estratégias relacionadas a aprendizado de máquina para tentar prever o comportamento das ações e buscam com essa ferramenta melhorar a tomada de decisão dos investidores.

Gu et al. (2018) em seus trabalhos testou diversos modelos de aprendizado de máquina como os de regressão, árvores de decisão e redes neurais e utilizou diversos indicadores financeiros para melhorar o seu modelo a fim de demonstrar que existem benefícios em utilizar essas ferramentas para tentar entender o comportamento futuro das ações no mercado norte-americano.

No estudo de Khaidem, Saha e Dey (2016), a técnica escolhida foi a da floresta aleatória para prever os preços das ações AAPL(*Apple*), GE (*General Eletric*) e *Samsung Electronics Co. Ltd* os modelos trouxeram acuracidades que ficaram entre 84 e 94% de acurácia, dependendo da janela de tempo que foi analisada, foram feitas previsões para o período de 1, 2 e 3 meses.

No trabalho de Barbosa, Kimura e Altman (2017), a floresta aleatória ou “*random forest*” apresentou elevado desempenho quando comparado a outras técnicas de aprendizado de máquina, sendo o modelo que trouxe menores erros e maior acuracidade na análise de risco e crédito, indicando que este é um algoritmo que consegue solucionar problemas de múltiplas áreas.

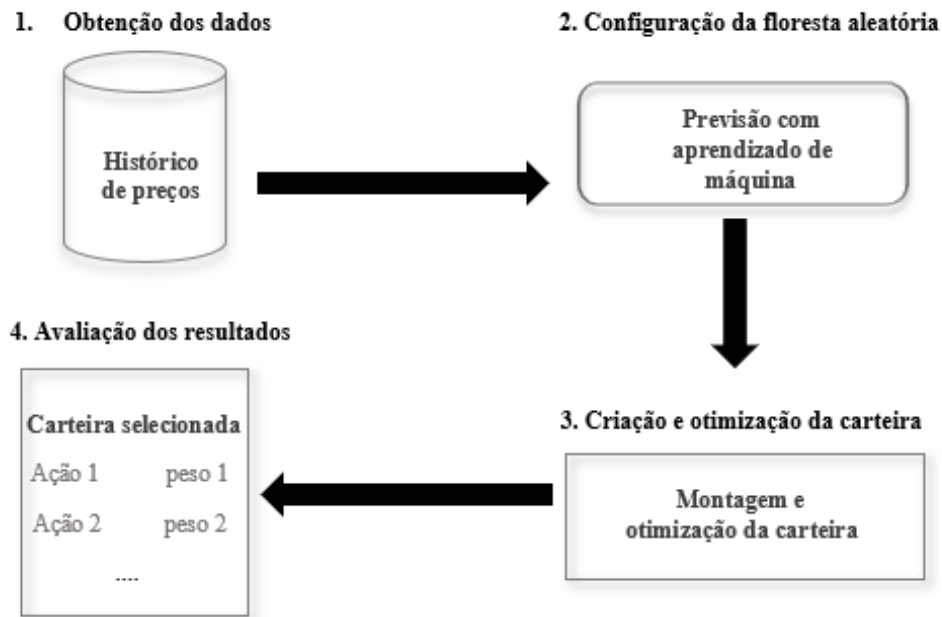
Santos (2020) em seu estudo testou prever as ações do B3 com três técnicas diferentes *Random Forest*, Redes Neurais Artificiais e *Support Vector Machines* empregando alguns indicadores técnicos como o Índice de Força Relativa, o oscilador estocástico etc, para treinar os modelos, os resultados obtidos variaram conforme a técnica de separação dos dados entre treino e teste e os modelos usados, tendo maior acuracidade com o modelo SVM.

3. Metodologia

Os algoritmos apontados neste estudo foram executados através da linguagem *Python*. A linguagem foi escolhida como ferramenta de execução por possuir uma construção simples e ser composta por diversas bibliotecas que contribuíram na elaboração do trabalho, como a biblioteca *pandas* para a manipulação dos dados, a biblioteca *numpy* para operações matemáticas, a *scikit-learn* para a implementação do algoritmo *Random Forest* e as bibliotecas *matplotlib* e *plotly* para visualização dos resultados em forma de gráficos.

O presente estudo empírico pode ser dividido em quatro etapas que são apresentadas na Figura 3:

Figura 3: Estrutura do modelo proposto



Fonte: Elaborado pelos autores (2021)

3.1. Obtenção dos dados

O presente trabalho é baseado nas ações listadas no Ibovespa no período que vai de 1 de janeiro de 2020 até 31 de março de 2021. Ao todo foram analisadas 82 ações e os dados foram obtidos através da API *pandas_datareader* contida na própria biblioteca *pandas* que permite extrair os dados de diários de fechamento, abertura, mínimo, máximo e volume das 82 ações analisadas.

Os dados foram divididos entre treino e teste, sendo que foram utilizados 75% para treinar o algoritmo, enquanto os 25% restantes de teste foram empregados para validar o modelo.

3.2. Configuração da floresta aleatória e previsão das ações:

Para que as previsões das ações fossem consideradas um horizonte preditivo de 30 dias. Alguns atributos técnicos, usados por muitos investidores a fim de identificar tendências de queda e alta nos preços, foram considerados no modelo além da série histórica dos preços. Abaixo os atributos utilizados no presente trabalho e suas definições segundo Murphy (1999):

- Índice de Força Relativa ou *Relative Strength Index* (RSI):

$$RSI = 100 - \frac{100}{1 + RS} \quad (2)$$

$$RS = \frac{G_{14}}{P_{14}} \quad (3)$$

Onde,

G14: Ganhos médios nos últimos 14 dias

P14: Perdas médias nos últimos 14 dias

RSI é um indicador de *momentum* popular que determina se a ação está sobrecomprado ou sobrevendido. Diz-se que uma ação está sobrecomprada quando a demanda injustificadamente empurra o preço para cima. Essa condição é geralmente interpretada como um sinal de que a ação está sobrevalorizada e que o preço provavelmente cairá. Diz-se que uma ação está sobrevendida quando o preço cai drasticamente para um nível abaixo de seu valor real. Este é um resultado causado devido ao pânico de vendas. O RSI varia de 0 a 100 e, geralmente, quando o RSI está acima de 70, pode indicar que a ação está sobrecomprada e quando o RSI está abaixo de 30, pode indicar que a ação está sobrevendida.

- Oscilador Estocástico ou *Stochastic Oscillator* (%K):

$$\%K = 100 \times \frac{(C - L_{14})}{(H_{14} - L_{14})} \quad (4)$$

Onde,

C: Preço de fechamento;

L₁₄: Menor preço nos últimos 14 dias.

H₁₄: Maior preço nos últimos 14 dias.

O oscilador estocástico segue a velocidade ou o *momentum* do preço. Como regra, o momentum muda antes que o preço mude. Ele mede o nível do preço de fechamento em relação à faixa baixo-alto durante um período.

- *Williams% R*

$$\%R = \frac{(H_{14} - C)}{(H_{14} - L_{14})} \times (-100) \quad (5)$$

Onde,

C: Preço de fechamento;

L₁₄: Menor preço nos últimos 14 dias.

H₁₄: Maior preço nos últimos 14 dias.

Williams% R varia de -100 a 0. Quando seu valor está acima de -20, indica um sinal de venda e quando seu valor está abaixo de -80, indica um sinal de compra.

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

$$MACD = EMA_{12}(C) - EMA_{26}(C) \quad (6)$$

$$SignalLine = EMA_9(MACD) \quad (7)$$

Onde,

C: Preços de fechamento;

EMA_n: Média móvel exponencial de n dias.

EMA significa *Exponential Moving Average* ou Média Móvel exponencial. Quando o MACD vai abaixo do *SignalLine*, isso indica um sinal de venda. Quando ultrapassa o *SignalLine*, indica um sinal de compra.

- Taxa de variação do preço ou *Price Rate of Change* (PROC_t):

$$PROC_t = \frac{C_t - C_{t-n}}{C_{t-n}} \quad (8)$$

Onde,

C_t: Preço de fechamento em um tempo t

Ele mede a alteração mais recente no preço em relação ao preço em n dias atrás.

- Saldo do Volume ou *On-Balance Volume* (OBV):

$$OBV(t) = \begin{cases} OBV(t-1) + Vol(t) & \text{if } C(t) > C(t-1) \\ OBV(t-1) - Vol(t) & \text{if } C(t) < C(t-1) \\ OBV(t-1) & \text{if } C(t) = C(t-1) \end{cases} \quad (9)$$

Onde,

OBV(t): Saldo do Volume no tempo t;

Vol(t): Volume comercializado no tempo t;

C(t) – Preço de fechamento no tempo t.

Utiliza mudanças no volume para estimar as mudanças nos preços das ações. Este indicador técnico é usado para as tendências de compra e venda de um estoque, considerando o volume acumulado, ele soma cumulativamente os volumes nos dias em que os preços se agrupam e subtrai o volume nos dias em que os preços caem, em comparação com os preços de o dia anterior.

3.2.1. Avaliação das previsões

Diferentemente do trabalho de Santos (2020) citado anteriormente, onde o autor prever os valores das ações utilizando a floresta aleatória como um problema de classificação o presente trabalho utiliza o mesmo algoritmo, porém tratando-o como um problema de regressão, nesse tipo de problema para se garantir a qualidade do modelo e saber o quanto o modelo erra uma medida comumente utilizada para é o Erro Médio Percentual Absoluto ou *MAPE* que foi utilizado no presente trabalho. A fórmula e definição desse parâmetro é apresentado de acordo com (MONTGOMERY; RUNGER, 2009):

- *MAPE - Mean Absolute Percentage Error* ou Erro Médio Percentual Absoluto:

Definição: É a média de erro em percentual absoluto

$$\frac{1}{n} \sum_{i=1}^n \frac{|y_i - \hat{y}_i|}{y_i} \cdot 100\% \quad (10)$$

Onde,

y_i : valores reais;

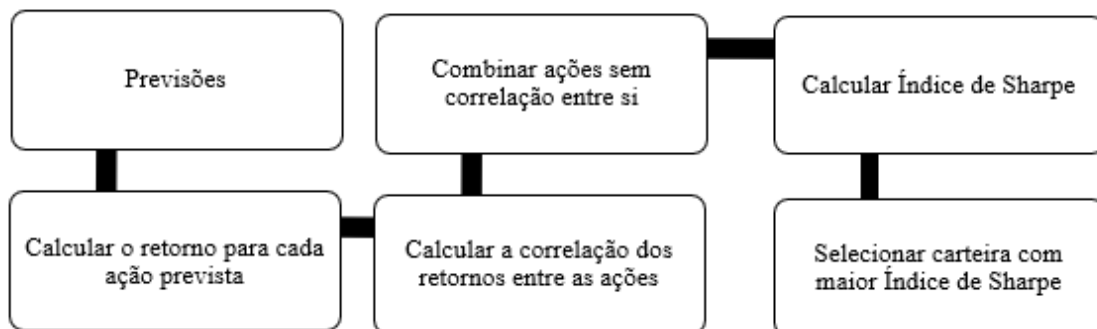
\hat{y} : valores previstos

Um valor de *MAPE* aceitável é de 10%, isso significa que o modelo previu errado 10% dos dados, esse valor foi utilizado como critério após as previsões serem feitas para definir qual ação entraria ou não na carteira, ou seja, considerou-se para os próximos passos apenas ações cujas previsões tiveram *MAPE* menor ou igual a 10%.

3.3. Criação e otimização da carteira:

Para compor a carteira foram utilizados os conceitos da Teoria dos portfólios apresentados anteriormente, analisando-se os retornos dos preços previstos e formando combinações de carteiras compostas por ações que não possuam correlação entre si, também foram atribuídos pesos aleatórios a cada ação que iria compor a carteira, a seleção da carteira mais atrativa é embasada na carteira que apresenta o maior Índice Sharpe e para o cálculo deste foi utilizada a taxa livre de risco de 2%. A Figura 4 apresenta as etapas para a composição da carteira.

Figura 4: Etapa da composição da carteira



Fonte: Elaborado pelos autores (2021)

3.4. Avaliação dos resultados

Logo após a carteira ótima ser selecionada foi feita uma simulação de estratégia (*backtesting*) considerando os pesos da carteira selecionada para ver como ela se comportou no passado em comparação ao portfólio da Ibovespa. Para a simulação foi considerado o valor de

R\$10.000,00. Por meio dessa avaliação pode-se definir se o portfólio selecionado poderá trazer potenciais ganhos ou perdas e entender como essa estratégia se comportou no passado.

4. Análise dos resultados

Das 82 duas ações analisadas do Ibovespa, utilizando o algoritmo *Random Forest* apenas 35 delas tiveram *MAPE* menor ou igual a 10%. Portanto, foram essas 35 ações apresentadas na Figura 5 que foram combinadas entre si para formarem diferentes carteiras.

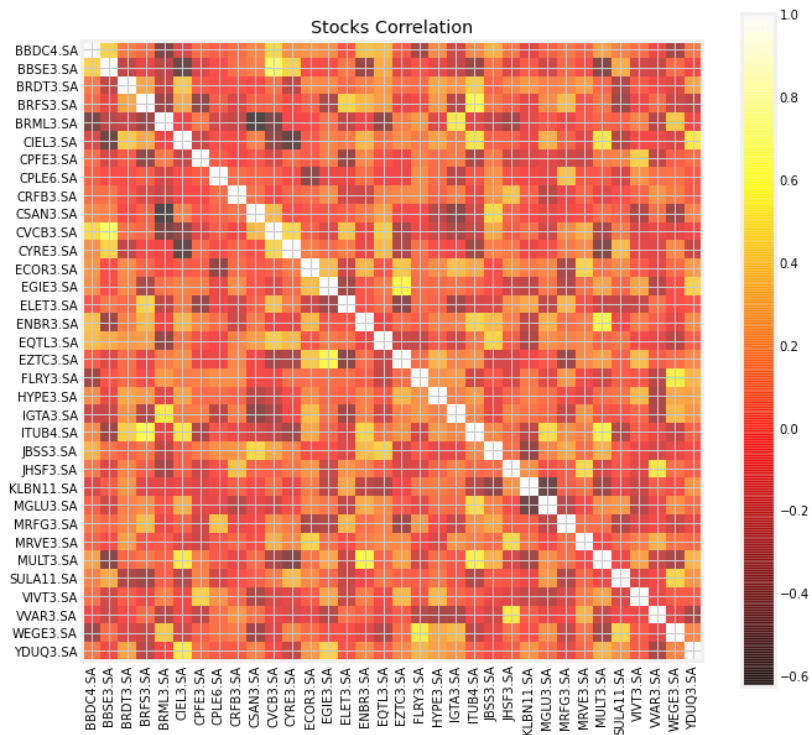
Figura 5: Ações selecionadas

```
[ 'BBSE3.SA',  
  'BBDC4.SA',  
  'BRFS3.SA',  
  'BRML3.SA',  
  'CRFB3.SA',  
  'CIEL3.SA',  
  'CPLE6.SA',  
  'CSAN3.SA',  
  'CPFE3.SA',  
  'CVCB3.SA',  
  'CYRE3.SA',  
  'ECOR3.SA',  
  'ELET3.SA',  
  'ENBR3.SA',  
  'EGIE3.SA',  
  'EQL3.SA',  
  'EZTC3.SA',  
  'FLRY3.SA',  
  'HYPE3.SA',  
  'IGTA3.SA',  
  'ITUB4.SA',  
  'JBSS3.SA',  
  'JHSF3.SA',  
  'KLBN11.SA',  
  'MGLU3.SA',  
  'MRFG3.SA',  
  'MRVE3.SA',  
  'MULT3.SA',  
  'RAIL3.SA',  
  'SULA11.SA',  
  'VIVT3.SA',  
  'UGPA3.SA',  
  'VVAR3.SA',  
  'WEGE3.SA',  
  'YDUQ3.SA' ]
```

Fonte: Elaborada pelos autores (2021)

Analisando as previsões para os próximos 30 dias a partir de 1 de abril calculou-se os retornos esperados dessas ações e posteriormente analisou-se as correlações entre esses retornos. A Figura 6 mostra o resultado das correlações entre os retornos das 35 ações.

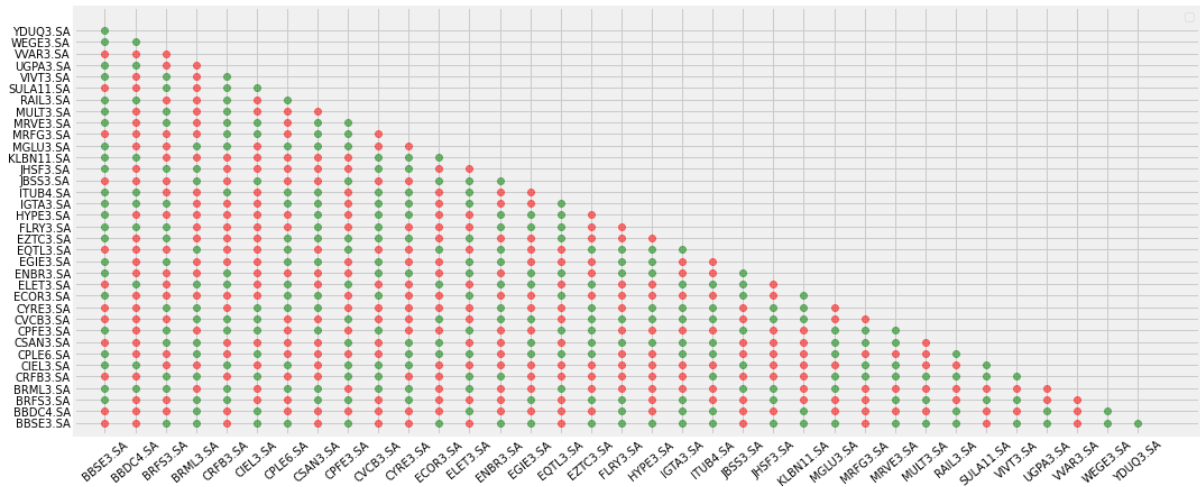
Figura 6: Correlações entre os retornos de cada ação



Fonte: Elaborada pelos autores (2021)

Posteriormente analisou-se as ações que poderiam ou não ser combinadas em uma carteira considerando que em uma mesma carteira não se pode ter ações com correlação forte entre si, a Figura 7 abaixo mostra em verde as ações que podem ser combinadas, enquanto as ações que não podem ser combinadas estão representadas pela cor vermelha. Ao total foram combinadas 595 ações em carteiras distintas onde foram atribuídos pesos aleatórios para cada ação em cada carteira, neste caso houve carteiras com ações iguais, porém com pesos diferentes.

Figura 7: Ações que podem ou não ser combinadas



Fonte: Elaborada pelos autores (2021)

Depois de compor as carteiras calculou-se o Índice Sharpe de cada uma e os resultados foram ordenados da carteira com maior Índice Sharpe para a menor, a Figura 8 mostra as primeiras 10 carteiras com maiores índices.

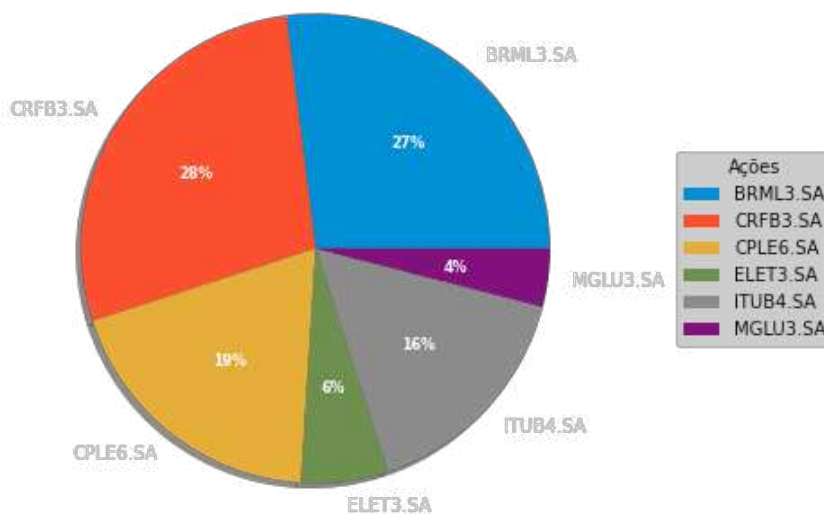
Figura 8: Carteiras montadas

Portfolio	Weights	Sharpe Ratio	Portfolio Return	Portfolio Volatility
[BRML3.SA, CRFB3.SA, CPLE6.SA, ELET3.SA, ITUB4...	[0.27, 0.28, 0.19, 0.06, 0.16, 0.04]	2.857310	18.07	1.116236
[BBDC4.SA, BRML3.SA, CRFB3.SA, CPLE6.SA, ELET3...	[0.05, 0.19, 0.16, 0.26, 0.11, 0.22]	2.730307	23.70	1.686046
[BBSE3.SA, BRML3.SA, CPLE6.SA, CYRE3.SA, ELET3...	[0.07, 0.15, 0.23, 0.1, 0.07, 0.13, 0.04, 0.21]	2.438648	20.57	1.519802
[BBSE3.SA, BRML3.SA, CRFB3.SA, CPLE6.SA, CPFE3...	[0.04, 0.22, 0.14, 0.26, 0.04, 0.14, 0.11, 0.04]	2.282902	18.06	0.908686
[BBDC4.SA, BRML3.SA, CPLE6.SA, CYRE3.SA, ELET3...	[0.08, 0.16, 0.22, 0.04, 0.14, 0.14, 0.21]	2.260140	24.30	2.058347
[CPLE6.SA, ECOR3.SA, ELET3.SA, EQTL3.SA, EZTC3...	[0.38, 0.1, 0.19, 0.05, 0.17, 0.1]	2.240211	24.67	1.583847
[BBDC4.SA, BRML3.SA, CRFB3.SA, ELET3.SA, ITUB4...	[0.08, 0.31, 0.24, 0.14, 0.24]	2.148977	22.31	1.929795
[BBSE3.SA, BRML3.SA, CPLE6.SA, ECOR3.SA, ELET3...	[0.08, 0.11, 0.36, 0.09, 0.07, 0.28]	2.117493	14.41	0.587416
[ECOR3.SA, ELET3.SA, CPLE6.SA, EGIE3.SA]	[0.09, 0.16, 0.47, 0.27]	2.088546	21.92	1.074330
[CPLE6.SA, CPFE3.SA, ELET3.SA, EQTL3.SA, EZTC3...	[0.39, 0.08, 0.19, 0.12, 0.13, 0.09]	2.047829	22.76	1.426359

Fonte: Elaborada pelos autores (2021)

Por fim, foi escolhida a carteira com maior Índice Sharpe com o valor aproximado de 2,85 para analisar sua estratégia no passado em relação ao portfólio total do Ibovespa contendo as 82 ações analisadas inicialmente. A Figura 9 apresenta a carteira selecionada composta pelas ações previstas e seus respectivos pesos.

Figura 9: Peso por ação da carteira selecionada



Fonte: Elaborada pelos autores (2021)

Com um montante de R\$10.000,00 buscou-se aqui analisar como seriam os ganhos e perdas da carteira analisada em comparação ao portfólio do Ibovespa com as 82 duas ações analisadas inicialmente. As Figuras 10 (a) e (b) apresentam respectivamente os retornos acumulados em valor absoluto e percentual. A linha azul representa a carteira selecionada enquanto a vermelha representa as ações do Ibovespa.

Figura 10: (a) Retornos acumulados em valores absolutos.

(b) Retornos acumulados em %

Figura (a)

Portfolio Value



Figura (b)

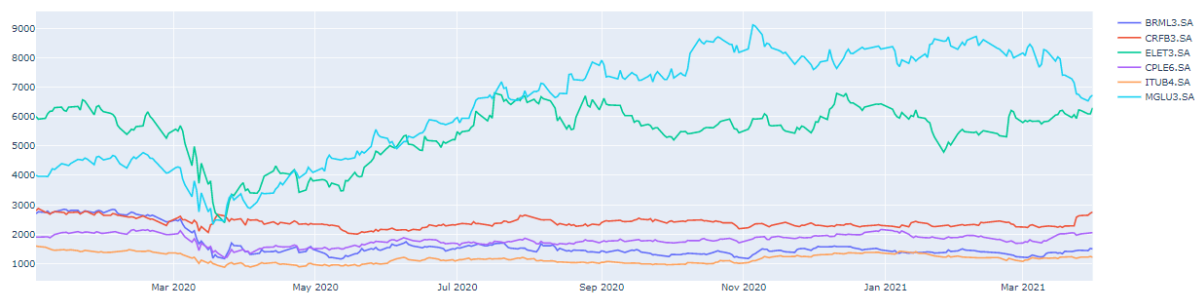
Cumulative Return % (Portfolio vs Ibovespa)



Fonte: Elaborada pelos autores (2021)

Aqui a carteira selecionada teve um comportamento bem parecido com as outras ações do Ibovespa, a correlação entre esta carteira e o Ibovespa é de 0,88 o que é considerado um valor alto e que justifica a semelhança entre os comportamentos, embora os retornos acumulados em valores absolutos sejam bem maiores da carteira em relação ao Ibovespa, quando se analisa esses valores em % nota-se que a diferença entre ambos diminui, porém, a carteira ainda apresenta maior desempenho. A Figura 11 apresenta os retornos individuais de cada ação que compõem a carteira.

Figura 11: Retornos acumulados de cada ação da carteira selecionada



Fonte: Elaborada pelos autores (2021)

Na Figura 11 nota-se que as ações MGLU3.SA e ELET3.SA foram as ações, baseando-se nessa estratégia, que trariam maiores retornos, também sendo as responsáveis para que os retornos acumulados em valores absolutos da carteira selecionada serem maiores que os do Ibovespa como um todo.

5. Considerações finais

O objetivo deste trabalho foi analisar a aplicação da Inteligência Artificial para otimização da carteira de ações previstas no IBOVESPA com a aplicação do *machine learning*. Como resultado dessa análise, conclui-se que a Inteligência Artificial pôde contribuir para escolha da carteira mais vantajosa.

Os resultados alcançados mostram que com o algoritmo escolhido não foi possível obter resultados satisfatórios ao se prever todas as 82 ações, sendo possível analisar 35 delas. Este fato pode ser oriundo tanto da alta volatilidade dos preços, sendo difícil para o algoritmo prever corretamente todas as ações, bem como pelo método de floresta aleatória escolhido.

A principal contribuição deste trabalho baseia-se na carteira otimizada selecionada contendo as ações BRML3.SA, CRFB3.SA, ELET3.SA, CPLE6.SA, ITUB4.SA e MGLU3.SA que apresentaram um equilíbrio entre os retornos e riscos considerados no estudo.

Como sugestão de futuros trabalhos a ideia seria de testar algoritmos de aprendizado profundo (*deep learning*) como as Redes LSTM e que considerem outros indicadores financeiros além dos utilizados no presente trabalho.

REFERÊNCIAS

AÇÕES. B3. 2021 Disponível em: <http://www.b3.com.br/pt_br/produtos-e-servicos/negociacao/renda-variavel/acoes.htm?csrt=263549456621472766>. Acesso em: 25 abr. 2021.

BARBOZA, Flavio, KIMURA, Herbert e ALTMAN, Edward. **Machine learning models and bankruptcy prediction**. Expert Systems with Applications. Nova York, 15 out. 2017.

CARTEIRA DE INVESTIMENTOS: o que é, como montar e diversificar. BTG Pactual. 2017. Disponível em: <<https://www.btgpactualdigital.com/blog/investimentos/carteira-de-investimentos>> Acesso em: 25 abr. 2021.

FEIGENBAUM, Armand. **The handbook of artificial intelligence**. Nova Jersey: Elsevier Inc. Butherworth- Heinemann. 1981

Franzini Filho, Carlos Roberto; Souza Filho, Adiloderne Nogueira; **TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL APLICADAS AOS PROCESSOS DE GESTÃO HOSPITALAR**, p. 2852-2863. In: Anais do VIII Simpósio de Engenharia de Produção. São Paulo: Blucher, 2020. Disponível em: <http://pdf.blucher.com.br.s3-sa-east-1.amazonaws.com/engineeringproceedings/viisimep/315846.pdf>. Acesso em: 26 mar. 2021. ISSN 2357-7592, DOI 10.5151/viisimep-315846

GU, Shihao, KELLY, Bryane e XIU, Dacheng. **Empirical asset pricing via machine learning**. NBER. Technical report, National Bureau of Economic Research. Cambridge, 28 dez. 2018

HURWITZ, Judith; KIRSCH, Daniel. **Machine Learning for dummies**. IBM. 2018. Disponível em: <<https://www.ibm.com/downloads/cas/GB8ZMQZ3>> Acesso em: 7 abr. 2021.

IZBICKI, Rafael, MENDONÇA, Tiago. **Aprendizado de máquina: uma abordagem estatística**. 1e. ed. São Carlos: [s. n.], 2020. 272 p. ISBN 978-65-00-02410-4. Disponível em: <<http://www.rizbicki.ufscar.br/AME.pdf>> Acesso em: 6 abr. 2021.

KACZMAREK, Tomasz. **Building portfolios based on machine learning predictions**. Economic Research-Ekonomska Istraživanja, Poland, p. 1-20, 1 out. 2020. Disponível em: <<https://doi.org/10.1080/1331677X.2021.1875865>> Acesso em: 13 abr. 2021.

KHAIDEM, Luckyson, SAHA, Snehanthu. e DEY, Sudeepa. **Predicting the direction of stock market prices using random forest**. Arxiv. Cornell University. Nova York, 29 abr. 2016.

LUGER, George. **Artificial intelligence**. 6. ed. Mexico: Editora Person University. 1993.

MANKIWI, N. Gregory. **Introdução à economia**. São Paulo: Cengage Learning, 2013

MARKOWITZ, Harry Max. **Portfolio selection** *Journal of Finance*, v.07, p.77-91, mar. 1952.

MITCHELL, Tom. Michael. **Machine Learning**. Pennsylvania. McGraw-Hill. 1997.

MONTGOMERY, Douglas; RUNGER, Geoger. **Estatística aplicada e probabilidade para engenheiros**. 4°. ed. 464 p. [S. l.]: LTC, 2009.

MURPHY, Joseph. **Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications**. New York Institute: Prentice Hall Press, 1999.

NOGUEIRA, Carolina Calío. **Previsibilidade No Mercado Acionário Utilizando Machine Learning**. 2019. Dissertação (Mestre em Economia) - A Fundação Getúlio Vargas, [S. l.], 2019. Disponível em:
<http://bibliotecadigital.fgv.br/dspace/bitstream/handle/10438/27999/carolina_calio_nogueira_VF.pdf?sequence=1&isAllowed=y> Acesso em: 10 abr. 2021.

O QUE É RENDA FIXA E POR QUE INVESTIR? XP Investimentos. 2021. Disponível em:
<<https://www.xpi.com.br/investimentos/renda-fixa/>> Acesso em: 25 abr. 2021.

O QUE É RENDA VARIÁVEL? **Aprenda tudo sobre essa forma de investimento**. Genial Investimentos. 2018. Disponível em: <<https://blog.genialinvestimentos.com.br/o-que-e-renda-variavel/>> Acesso em: 25 abr. 2021.

SANTOS, Gustavo Carvalho. **Algoritmos de Machine Learning Para Previsão De Ações Da B3**. Dissertação (Mestre em Ciências) - Universidade Federal de Uberlândia, [S. l.], 2020. Disponível em:
<<https://repositorio.ufu.br/bitstream/123456789/29897/7/AlgoritmosMachineLearning.pdf>> Acesso em: 6 abr. 2021.

VARGAS, G.; LEAL, R. **Gestão de Investimento e Fundos**. São Paulo: Atlas, 2006. 480 p.

APÊNDICES

CODIGO FONTE

A1. Obtendo os dados

```
import pandas_datareader as pdr
import datetime
import pandas as pd
import numpy as np
```

```
tickers=['ABEV3.SA',
'ASAI3.SA',
'AZUL4.SA',
'BTOW3.SA',
'B3SA3.SA',
'BBSE3.SA',
'BBDC3.SA',
'BBDC4.SA',
'BRAP4.SA',
'BBAS3.SA',
'BRKM5.SA',
'BRFS3.SA',
'BRML3.SA',
'BPAC11.SA',
'CRFB3.SA',
'CCRO3.SA',
'CMIG4.SA',
'HGTX3.SA',
'CIEL3.SA',
'COGN3.SA',
'CPLE6.SA',
'CSAN3.SA',
'CPFE3.SA',
'CVCB3.SA',
'CYRE3.SA',
'ECOR3.SA',
'ELET3.SA',
'ELET6.SA',
```


'EMBR3.SA',
'ENBR3.SA',
'ENGI11.SA',
'ENEV3.SA',
'EGIE3.SA',
'EQTL3.SA',
'EZTC3.SA',
'FLRY3.SA',
'GOAU4.SA',
'GGBR4.SA',
'GOLL4.SA',
'NTCO3.SA',
'HAPV3.SA',
'HYPE3.SA',
'IGTA3.SA',
'GNDI3.SA',
'IRBR3.SA',
'ITSA4.SA',
'ITUB4.SA',
'JBSS3.SA',
'JHSF3.SA',
'KLBN11.SA',
'RENT3.SA',
'LCAM3.SA',
'LAME4.SA',
'LREN3.SA',
'MGLU3.SA',
'MRFG3.SA',
'BEEF3.SA',
'MRVE3.SA',
'MULT3.SA',
'PCAR3.SA',
'BRDT3.SA',

'PETR3.SA',
'PETR4.SA',
'PRIO3.SA',
'QUAL3.SA',
'RADL3.SA',
'RAIL3.SA',
'SBSP3.SA',
'SANB11.SA',
'CSNA3.SA',
'SULA11.SA',
'SUZB3.SA',
'TAE11.SA',
'VIVT3.SA',
'TIMS3.SA',
'TOTS3.SA',
'UGPA3.SA',
'USIM5.SA',
'VALE3.SA',
'VVAR3.SA',
'WEGE3.SA',
'YDUQ3.SA']

```

def get(tickers, startdate, enddate):
    def data(ticker):
        return (pdr.get_data_yahoo(ticker, start=startdate, end=enddate))
    datas = map (data, tickers)
    return(pd.concat(datas, keys=tickers, names=['Ticker', 'Date']))

start_date = datetime.datetime(2019, 1, 1)
end_date = datetime.datetime(2021, 4, 29)

all_data = get(tickers, start_date, end_date)

all_data.columns

all_data.shape

#salvar os resultados em arquivos csv
all_data.to_csv('/content/drive/MyDrive/Colab Notebooks/TCC/TCC_v2/ibovespa.csv')

```

A2. Montar previsões

```

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_log_error, mean_absolute_error, mean_squared_error
from sklearn.ensemble import RandomForestRegressor

### **Carregando o histórico de preço**

price_data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/TCC/TCC_v2/ibovespa.csv')
price_data.head()

price_data = price_data[(price_data['Date']>= '2020-01-01') & (price_data['Date']<= '2021-03-31')]
price_data['Date'] = pd.to_datetime(price_data['Date'])

# sort the values by symbol and then date
price_data.sort_values(by = ['Ticker', 'Date'], inplace = True)

# calculate the change in price
price_data['change_in_price'] = price_data['Close'].diff()

# identify rows where the symbol changes
mask = price_data['Ticker'] != price_data['Ticker'].shift(1)

# For those rows, let's make the value null
price_data['change_in_price'] = np.where(mask == True, np.nan, price_data['change_in_price'])

# Calculate the 14 day RSI
n = 14

# First make a copy of the data frame twice
up_df, down_df = price_data[['Ticker', 'change_in_price']].copy(), price_data[['Ticker', 'change_in_price']].copy()

# For up days, if the change is less than 0 set to 0.
up_df.loc['change_in_price'] = up_df.loc[(up_df['change_in_price'] < 0), 'change_in_price'] = 0

```

```

# For down days, if the change is greater than 0 set to 0.
down_df.loc['change_in_price'] = down_df.loc[(down_df['change_in_price'] > 0), 'change_in_price'] = 0

# We need change in price to be absolute.
down_df['change_in_price'] = down_df['change_in_price'].abs()

# Calculate the EWMA (Exponential Weighted Moving Average), meaning older values are given less weight compared to newer values.
ewma_up = up_df.groupby('Ticker')['change_in_price'].transform(lambda x: x.ewm(span = n).mean())
ewma_down = down_df.groupby('Ticker')['change_in_price'].transform(lambda x: x.ewm(span = n).mean())

# Calculate the Relative Strength
relative_strength = ewma_up / ewma_down

# Calculate the Relative Strength Index
relative_strength_index = 100.0 - (100.0 / (1.0 + relative_strength))

# Add the info to the data frame.
price_data['down_days'] = down_df['change_in_price']
price_data['up_days'] = up_df['change_in_price']
price_data['RSI'] = relative_strength_index

# Calculate the Stochastic Oscillator
n = 14

# Make a copy of the high and low column.
low_14, high_14 = price_data[['Ticker', 'Low']].copy(), price_data[['Ticker', 'High']].copy()

# Group by symbol, then apply the rolling function and grab the Min and Max.
low_14 = low_14.groupby('Ticker')['Low'].transform(lambda x: x.rolling(window = n).min())
high_14 = high_14.groupby('Ticker')['High'].transform(lambda x: x.rolling(window = n).max())

# Calculate the Stochastic Oscillator.
k_percent = 100 * ((price_data['Close'] - low_14) / (high_14 - low_14))

# Add the info to the data frame.
price_data['low_14'] = low_14
price_data['high_14'] = high_14
price_data['k_percent'] = k_percent

# Calculate the Williams %R
n = 14

# Make a copy of the high and low column.
low_14, high_14 = price_data[['Ticker', 'Low']].copy(), price_data[['Ticker', 'High']].copy()

# Group by symbol, then apply the rolling function and grab the Min and Max.
low_14 = low_14.groupby('Ticker')['Low'].transform(lambda x: x.rolling(window = n).min())
high_14 = high_14.groupby('Ticker')['High'].transform(lambda x: x.rolling(window = n).max())

# Calculate William %R indicator.
r_percent = ((high_14 - price_data['Close']) / (high_14 - low_14)) * -100

# Add the info to the data frame.
price_data['r_percent'] = r_percent

# Calculate the MACD
ema_26 = price_data.groupby('Ticker')['Close'].transform(lambda x: x.ewm(span = 26).mean())
ema_12 = price_data.groupby('Ticker')['Close'].transform(lambda x: x.ewm(span = 12).mean())
macd = ema_12 - ema_26

# Calculate the EMA
ema_9_macd = macd.ewm(span = 9).mean()

# Store the data in the data frame.
price_data['MACD'] = macd
price_data['MACD_EMA'] = ema_9_macd

# Calculate the Price Rate of Change
n = 9

# Calculate the Rate of Change in the Price, and store it in the Data Frame.
price_data['Price_Rate_Of_Change'] = price_data.groupby('Ticker')['Close'].transform(lambda x: x.pct_change(periods = n))

```

```

def obv(group):

    # Grab the volume and close column.
    volume = group['Volume']
    change = group['Close'].diff()

    # initialize the previous OBV
    prev_obv = 0
    obv_values = []

    # calculate the On Balance Volume
    for i, j in zip(change, volume):

        if i > 0:
            current_obv = prev_obv + j
        elif i < 0:
            current_obv = prev_obv - j
        else:
            current_obv = prev_obv

        # OBV.append(current_OBV)
        prev_obv = current_obv
        obv_values.append(current_obv)

    # Return a panda series.
    return pd.Series(obv_values, index = group.index)

# apply the function to each group
obv_groups = price_data.groupby('Ticker').apply(obv)

# add to the data frame, but drop the old index, before adding it.
price_data['On Balance Volume'] = obv_groups.reset_index(level=0, drop=True)

def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred = np.array(y_true), np.array(y_pred)
    return np.mean(np.abs((y_true - y_pred) / y_true)) * 100

df = df_filter = pd.DataFrame()
previsoes = {
    'previsao': [],
    'Ticker': []
}
validacao = {
    #'rmse': [],
    #'mae': [],
    'mape': [],
    'Ticker': []
}

for item in tickers: # loop para fazer previsão para cada uma das ações

    df_filter = price_data[price_data['Ticker'] == item] #filtrar a ação
    # Grab our X & Y Columns.
    df_filter.set_index(df_filter['Date'], inplace=True)
    X_Cols = df_filter[['RSI', 'k_percent', 'r_percent', 'Price_Rate_Of_Change', 'MACD', 'On Balance Volume']]
    Y_Cols = df_filter['Adj Close']

    # Split X and y into X_
    train_size = int(len(Y_Cols) * 2 / 3)
    X_train = X_Cols[:train_size]
    X_test = X_Cols[train_size:]
    y_train = Y_Cols[:train_size]
    y_test = Y_Cols[train_size:]
    # Create a Random Forest Classifier
    rand_frst = RandomForestRegressor(n_estimators = 100, oob_score = True, criterion = "mse", random_state = 0)

    # Fit the data to the model
    rand_frst.fit(X_train, y_train)

    # Make predictions
    y_pred = rand_frst.predict(X_test)

    #previsão dos ultimos 30 dias
    pred = pd.DataFrame(data=y_pred[-30:], columns=['previsao'])
    pred['Ticker'] = item
    d.append(pred)

```

```

#validações
#rmse = np.sqrt(mean_squared_error(y_test,y_pred))
#mae = mean_absolute_error(y_test,y_pred)
mape = mean_absolute_percentage_error(y_test, y_pred)
#validacao['rmse'].append(rmse)
#validacao['mae'].append(mae)
validacao['mape'].append(mape)
validacao['Ticker'].append(item)

previsoes = pd.concat(d)

validacaoDF = pd.DataFrame(validacao)
validacaoDF

validacaoDF.sort_values(by='mape',ascending=True)

stocks_to_keep = validacaoDF[validacaoDF['mape'] <=10]
stocks_to_keep = stocks_to_keep['Ticker'].tolist()
#stocks_to_keep.reset_index(drop=True,inplace=True)
print(f'Total de ações com previsões boas:{len(stocks_to_keep)}')

stocks_to_keep

previsoes = previsoes[previsoes['Ticker'].isin(stocks_to_keep)]

previsoes = previsoes.pivot_table(columns='Ticker', values='previsao',index=previsoes.index)

previsoes.index = pd.date_range('2021-04-01', freq='D', periods=30)

previsoes.to_csv('/content/drive/MyDrive/Colab Notebooks/TCC/TCC_v2/previsoes.csv')

```

A3. Montando o portfolio

```

import pandas as pd
import numpy as np
from datetime import datetime
import matplotlib.pyplot as plt
from matplotlib import cm as cm
plt.style.use('fivethirtyeight')

df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/TCC/TCC_v2/previsoes.csv')

```

```

df=df[['BBSE3.SA
      'BBDC4.SA',
      'BRFS3.SA',
      'BRML3.SA',
      'CRFB3.SA',
      'CIEL3.SA',
      'CPL6.SA',
      'CSAN3.SA',
      'CPFE3.SA',
      'CVCB3.SA',
      'CYRE3.SA',
      'ECOR3.SA',
      'ELET3.SA',
      'ENBR3.SA',
      'EGIE3.SA',
      'EQTL3.SA',
      'EZTC3.SA',
      'FLRY3.SA',
      'HYPE3.SA',
      'IGTA3.SA',
      'ITUB4.SA',
      'JBSS3.SA',
      'JHSF3.SA',
      'KLBN11.SA',
      'MGLU3.SA',
      'MRFG3.SA',
      'MRVE3.SA',
      'MULT3.SA',
      'RAIL3.SA',
      'SULA11.SA',
      'VIVT3.SA',
      'UGPA3.SA',
      'VVAR3.SA',
      'WEGE3.SA',
      'YDUQ3.SA']]

```

```

#Calculando o retorno para os preços previstos
data = df.copy()
returns = data.pct_change()[1:]
len(returns)

```

```

Covariance = returns.cov()
Correlation = returns.corr()
#np.round(Correlation,3)

```

```

def correlation_matrix_plot(matrix, maptype):
    fig = plt.figure(figsize=(10,10))
    ax1 = fig.add_subplot(111)
    cmap = cm.get_cmap('jet', 30)
    if (maptype == 'cmap'):
        cax = ax1.imshow(matrix, interpolation="nearest", cmap=cmap, alpha=0.7)
    else:
        cax = ax1.imshow(matrix, interpolation="nearest", cmap=maptype, alpha=0.7)
    ax1.grid(True)
    plt.title('Stocks Correlation')
    ax1.set_xticks(np.arange(len(companies)))
    ax1.set_yticks(np.arange(len(companies)))
    ax1.set_xticklabels(companies,fontsize=10,rotation=90)
    ax1.set_yticklabels(companies,fontsize=10)
    ax1.set_alpha(0.4)
    ticks = np.arange(0,9,1)
    fig.colorbar(cax)
    plt.show()

correlation_matrix_plot(Correlation,'hot')

```

```

mean_Covariance = Covariance.mean()

#Função que retornará as ações que podem ser combinadas considerando a não correlação entre elas
Covariance = returns.cov()
Correlation = returns.corr()
mean_Covariance = Covariance.mean()

def get_pairable(data, Covariance, Correlation):
    Pairable = np.zeros(Covariance.shape)
    plt.figure(figsize=(16,7))

    for i in range(len(companies)):
        for j in range(len(companies)-i):
            if(Covariance[i,j] > mean_Covariance[i] or Covariance[i,j] > mean_Covariance[j] or Correlation[i][j]>0.5):
                plt.plot(i, j, 'o', color='red', alpha=0.5)
            else:
                plt.plot(i, j, 'o', color='green', alpha=0.5)
                Pairable[i,j] = 1

    plt.xlim(-1,len(companies)+1)
    plt.ylim(-1,len(companies)+1)
    plt.xticks(range(len(companies)), companies, rotation=40)
    plt.yticks(range(len(companies)), companies)
    #plt.set_xticklabels(companies,fontsize=10,rotation=40)
    plt.legend()
    return Pairable

#Verde ações que se combinam
#Vermelho ações que não se combinam
Pairable = get_pairable(data,np.array(Covariance), np.array(Correlation))

#Calculando o Índice Sharpe
Rf = 2/100 #risk_free_rate
stocks_rng = range(len(companies))

def sharpe_ratio(pair, portfolio_weights, meanR, cov):
    Rp = portfolio_weights.dot(meanR.T)
    SigmaP = portfolio_weights.dot(cov.dot(portfolio_weights.T)) * len(returns)
    s_ratio = (Rp - Rf)/np.sqrt(SigmaP)
    return s_ratio

#Função para formar os pares que não tem correlação entre si
def check_pairs(pair):
    not_in_pair = []
    for i in stocks_rng:
        if(i not in pair):
            not_in_pair.append(i)
        else:
            continue

    for tick in not_in_pair:
        total_pair = len(pair)
        for i in pair:
            if(Pairable[tick,i]!=1 or Pairable[i,tick]!=1):
                total_pair -= 1
        if(float(total_pair)/len(pair) > 0.5):
            pair.add(tick)
    return pair

#Função que irá formar os portfolios segunda o Índice Sharpe

portfolio_collection = []
optim = {'good':[],'better':[],'best': []}

```

```

def select(combo):
    pair = set(combo)
    pair = check_pairs(pair)
    #print("new pair by checking pairable: ", pair)
    if pair in portfolio_collection:
        #print('returning')
        return
    portfolio_collection.append(pair)
    #print("portfolio: ", portfolio_collection)
    sharpe_r = 0
    eff_weights = np.ones(len(pair))

    symbols = [companies[s] for s in pair]
    mean_returns = np.array(returns[symbols].mean()) * len(returns)
    sub_cov_mat = np.array(returns[symbols].cov())

    for _ in range(200):
        weights = [np.random.randint(50,500) for _ in pair]
        weights = np.array(weights, dtype=float).round(2)
        weights /= round(weights.sum(), 2)
        s_r = sharpe_ratio(pair, weights, mean_returns, sub_cov_mat)
        if (s_r > sharpe_r):
            sharpe_r = s_r
            eff_weights = weights.round(2)

    if (sharpe_r >= 1 and sharpe_r < 2) :
        optim['good'].append([[companies[s] for s in pair], eff_weights, sharpe_r])
    if (sharpe_r >= 2 and sharpe_r < 3) :
        optim['better'].append([[companies[s] for s in pair], eff_weights, sharpe_r])
    if (sharpe_r >= 3) :
        optim['best'].append([[companies[s] for s in pair], eff_weights, sharpe_r])

for row in better.iterrows():
    mean = np.array(returns[row[1][0]].mean()) * len(returns)
    portfolio_weights = np.array((row[1][1]))
    Rp = portfolio_weights.dot(mean.T)
    portfolio_return.append(round(Rp * 100, 2))

better['Portfolio Return'] = portfolio_return
better

portfolio_volatility = []

for row in better.iterrows():
    portfolio = row[1][0]
    portfolio_weights = (row[1][1])
    portfolio_data = data[portfolio]
    portfolio = portfolio_data.mul(portfolio_weights,axis=1).sum(axis=1)
    volatility = np.std(portfolio)
    portfolio_volatility.append(volatility)

better['Portfolio Volatility'] = portfolio_volatility
better.sort_values(by='Sharpe Ratio', ascending=False)

better['PortfolioL'] = better['Portfolio'].apply(lambda x: ', '.join(map(str, x)))

plt.figure(figsize=(14,5))
fig, ax1 = plt.subplots(figsize=(14,6))

ax2 = ax1.twinx()
ax2.plot(better['PortfolioL'], better['Sharpe Ratio'], 'g-', alpha=0.6, marker='o',
         linestyle='dashed', linewidth=2, markersize=6)
ax1.plot(better['PortfolioL'], better['Portfolio Return'], 'b-', alpha=0.6, marker='o',
         linestyle='dashed', linewidth=2, markersize=6)
ax1.plot(better['PortfolioL'], better['Portfolio Volatility'], 'r-', alpha=0.6, marker='o',
         linestyle='dashed', linewidth=2, markersize=6)

ax1.set_xlabel('Portfolios with Sharpe ratio >= 2 and < 3 (BETTER)')
ax1.set_ylabel('Return and Volatility')
ax2.set_ylabel('Sharpe Ratio')
ax1.xaxis.set_tick_params(rotation=90)
ax1.legend(loc='best')
ax2.legend(loc='best')

plt.show()

```



```

"""### **Analisando o melhor portfolio**"""
portfolio = better[better.index == 2]
for ticker in portfolio['Portfolio']:
    print(ticker)

"""### **Backtesting**"""

price_data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/TCC/TCC_v2/ibovespa.csv')
price_data = price_data[(price_data['Date']>= '2020-01-01') & (price_data['Date']<= '2021-03-31')]
price_data.set_index(price_data['Date'], inplace=True)
price_data.head()

tickers = ['BRML3.SA', 'CRFB3.SA', 'CPLE6.SA', 'ELET3.SA', 'ITUB4.SA', 'MGLU3.SA']

#price_data = price_data[price_data['Ticker'].isin(tickers)]

weights = [27, 28, 19, 6, 16, 4]

fig, ax = plt.subplots(figsize=(6,6), subplot_kw=dict(aspect="equal"))
wedges, texts, autotexts = ax.pie(weights, labels=tickers, autopct="%1.0f%%",
    shadow=True, textprops=dict(color="w"))

ax.legend(wedges, tickers,
          title="Ações",
          loc="center",
          bbox_to_anchor=(1,0,0.5,1))

plt.setp(autotexts, size=8, weight="bold")

ax.set_title("Peso por ação da cateira selecionada:")

plt.show()

import plotly.express as px
import plotly.graph_objects as go

BRML = price_data[price_data['Ticker'] == 'BRML3.SA']
CRFB = price_data[price_data['Ticker'] == 'CRFB3.SA']
CPLE = price_data[price_data['Ticker'] == 'CPLE6.SA']
ELET = price_data[price_data['Ticker'] == 'ELET3.SA']
ITUB = price_data[price_data['Ticker'] == 'ITUB4.SA']
MGLU = price_data[price_data['Ticker'] == 'MGLU3.SA']

price_data.reset_index(drop=True, inplace=True)
price_data = price_data.groupby('Date').sum()
price_data.head()

# Calculate the cumulative return
for df in (BRML, CRFB, CPLE, ELET, ITUB, MGLU, price_data):
    df['Cum Return'] = df['Adj Close']/df.iloc[0]['Adj Close']

# Calculate value of initial investment of 10K in the Portfolio
init_investment = 10000
for df, alloc in zip((BRML, CRFB, CPLE, ELET, ITUB, MGLU), [.27, .28, .19, .6, .16, .4]):
    df['Value'] = alloc*init_investment*df['Cum Return']

# Calculate value of initial investment of 10K in the SPY
price_data['Ibovespa Total'] = init_investment*price_data['Cum Return']
price_data['Ibovespa Total'] = round(price_data['Ibovespa Total'])

# Combine All dataframes
all_vals = [BRML['Value'], CRFB['Value'], CPLE['Value'], ELET['Value'], ITUB['Value'], MGLU['Value']]
portfolio_val = pd.concat(all_vals, axis=1)
portfolio_val.columns = ['BRML3.SA', 'CRFB3.SA', 'CPLE6.SA', 'ELET3.SA', 'ITUB4.SA', 'MGLU3.SA']

# Calculate Cumulative Return and Cumulative Return %
portfolio_val['Portfolio Total'] = round(portfolio_val.sum(axis=1))
portfolio_val['Cum Return'] = portfolio_val['Portfolio Total']/portfolio_val.iloc[0]['Portfolio Total']

portfolio_val['Cum Return %'] = (portfolio_val['Cum Return'] - 1) * 100
price_data['Cum Return %'] = (price_data['Cum Return'] - 1) * 100
price_data.tail()

```

```

"""**Now, let's visualize the Growth of 10K initial investment using**"""

# Visualize Portfolio Value and Compare it to price_data benchmark
import plotly.graph_objects as go
fig = go.Figure()
fig.add_trace(go.Scatter(x=portfolio_val.index, y=portfolio_val['Portfolio Total'], name='Portfolio Total'))
fig.add_trace(go.Scatter(x=price_data.index, y=price_data['Ibovespa Total'],name='Ibovespa Total'))
fig.update_layout(title="Portfolio Value")

# Visualize Return %
fig = go.Figure()
fig.add_trace(go.Scatter(x=portfolio_val.index, y= portfolio_val['Cum Return %'], name='Portfolio Cumulative Return %'))
fig.add_trace(go.Scatter(x=price_data.index, y=price_data['Cum Return %'],name='Ibovespa Cumulative Return %'))
fig.update_layout(title="Cumulative Return % (Portfolio vs Ibovespa)")

print("Portfolio Correlation with Ibovespa")
round(portfolio_val['Portfolio Total'].corr(price_data['Ibovespa Total']),2)

# Visualize Value of Individual Stocks
fig = go.Figure()
fig.add_trace(go.Scatter(x=portfolio_val.index, y= portfolio_val['BRML3.SA'], name='BRML3.SA'))
fig.add_trace(go.Scatter(x=portfolio_val.index, y= portfolio_val['CRFB3.SA'],name='CRFB3.SA'))
fig.add_trace(go.Scatter(x=portfolio_val.index, y= portfolio_val['ELET3.SA'],name='ELET3.SA'))
fig.add_trace(go.Scatter(x=portfolio_val.index, y= portfolio_val['CPLE6.SA'],name='CPLE6.SA'))
fig.add_trace(go.Scatter(x=portfolio_val.index, y= portfolio_val['ITUB4.SA'],name='ITUB4.SA'))
fig.add_trace(go.Scatter(x=portfolio_val.index, y= portfolio_val['MGLU3.SA'],name='MGLU3.SA'))

```