BRENO MENDES CARDOSO FRAGA

INTEGRAÇÃO DE REDES NEURAIS DO TIPO LSTM EM OTIMIZAÇÃO DE PORTFÓLIOS DE INVESTIMENTO

São Paulo
2021

BRENO MENDES CARDOSO FRAGA

INTEGRAÇÃO DE REDES NEURAIS DO TIPO LSTM EM OTIMIZAÇÃO DE PORTFÓLIOS DE INVESTIMENTO

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro de Produção.

São Paulo
2021

BRENO MENDES CARDOSO FRAGA

INTEGRAÇÃO DE REDES NEURAIS DO TIPO LSTM EM OTIMIZAÇÃO DE PORTFÓLIOS DE INVESTIMENTO

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro de Produção.

Orientadora:

Profª. Dra. Celma de Oliveira Ribeiro

São Paulo
2021

Catalogação-na-publicação

Fraga, Breno
Integração de Redes Neurais do tipo LSTM em Otimização de Portfólios de Investimento / B. Fraga -- São Paulo, 2021.
97 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Produção.

1.LSTM 2.ARCH/GARCH 3.Fronteira Eficiente 4.Previsão de retornos 5.Investimentos Quantitativos I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Produção II.t.

Agradecimentos

Gostaria de agradecer meus pais, Eduardo e Terezinha, que sempre tiveram como maior prioridade de suas vidas fornecer uma educação de qualidade a mim e a meus irmãos, passando por incontáveis sacrifícios para que isso se realizasse.

Agradeço ao meu irmão João Vitor, quem eu considero ser a pessoa mais inteligente e curiosa que conheço, me ajudando incontáveis vezes a entender conceitos complicados de matemática, física e afins. Juntos desde pequenos, sempre esteve presente para me apoiar.

À minha orientadora, a Professora Dra. Celma, por toda a dedicação e atenção que deu ao meu projeto, guiando-me a algo factível e interessante. Tenho certeza de que, sem ela, meu projeto não teria se concretizado e eu não teria metade dos conhecimentos que adquiri ao longo deste Trabalho de Formatura.

Aos meus amigos do prédio, que há muitos anos acreditam em mim mais do que eu acredito em mim mesmo. Aos meus amigos da faculdade, que juntos atravessamos essa jornada que é o curso de engenharia, atravessando incontáveis problemas e momentos de felicidade. A todos quem tive o prazer de me aproximar durante o meu intercâmbio na França, que me marcaram para o resto da minha vida e com certeza ajudaram a moldar quem eu sou hoje.

A todos os professores da Escola Politécnica que ajudaram no meu desenvolvimento como aluno para chegar onde cheguei.

À equipe com a qual trabalhei e que tanto me ensinou sobre conceitos que são abordados neste trabalho, me propiciando conhecimentos que ficarão comigo para o resto da minha vida.

Resumo

Este trabalho tem como contexto o trabalho em uma gestora de fundos de investimento, tendo como objetivo suprir esta gestora com o desenvolvimento de uma estratégia de investimentos no mercado acionário brasileiro utilizando-se modelos de redes neurais artificiais do tipo LSTM. Estas foram utilizadas para elaborar um modelo de previsão de retornos de 20 dias de um conjunto pré-selecionado de ativos. Em seguida, modelos do tipo ARCH/GARCH foram utilizados em conjunto a redes neurais LSTM para a previsão de covariâncias dos ativos. Por fim, técnicas de otimização de alocação dos recursos financeiros nos mesmos ativos com o objetivo de maximizar os retornos esperados dado um limite pré-estabelecido de risco foram utilizados para, enfim, a construção da estratégia de investimento. Esta se mostrou rentável no período de teste e superou *benchmarks* relevantes no mercado, como o Ibovespa.

Palavras-Chave – LSTM, ARCH/GARCH, Fronteira Eficiente, Previsão de retornos, Investimentos Quantitativos.

Abstract

This work's main focus is the development of an investment strategy on the brazilian stock market using LSTM type artificial neural networks for a mutual funds managing company. These networks were used to create a model that predicts the 20-day future returns of a previously established stock group. Afterwards, ARCH and GARCH models were combined with LSTM neural networks to build a predictor for the stocks' covariance matrix. Finally, optimization techniques for distributing financial resources throughout investments on each stock with the objective of maximizing returns given a maximum acceptable risk were used to generate the final investment strategy. Such strategy has shown positive returns within the test sample, beating relevant market *benchmarks*, such as Ibovespa.

Keywords – LSTM, ARCH/GARCH, Efficient frontier, Return prediction, Quantitative Investment.

Lista de Figuras

1 Representação do funcionamento de uma rede neural básica 23

2 Representação do funcionamento de um neurônio artificial básico 24

3 Representação gráfica da função sigmoidal com números reais 24

4 Representação gráfica da função tangente hiperbólico com números reais . . . 25

5 Representação ilustrativa dos efeitos de diferentes taxas de aprendizado 27

6 Representação ilustrativa dos efeitos do uso de Adam sobre o aprendizado da rede 28

7 Representação ilustrativa dos efeitos de sobreajuste sobre um mesmo conjunto de pontos 29

8 Representação da técnica de parada antecipada 30

9 Representação ilustrativa de uma rede neural recorrente simples 31

10 Representação de uma célula LSTM 33

11 Ativos fictícios cujos retornos e volatilidades esperadas são conhecidas 39

12 Representação dos retornos e volatilidades de uma combinação linear de dois ativos cujos retornos são normalmente distribuídos 40

13 Exemplo de uma regressão linear aplicada a uma função não linear 42

14 Retornos diários da ação COCE5 42

15 Funções FAC e FACP aplicadas nas variâncias dos retornos de VALE5 44

16 Preços diários de ITUB4 com suas médias móveis de 5 e 30 dias 48

17 Bandas de Bollinger aplicadas aos preços diários de ITUB4 ao longo de 140 dias úteis, em que t=20 e n=2 50

18 Preços diários das ações do banco Goldman Sachs 52

19 Representação do cálculo de uma taxa média de juros com base em períodos de diferentes taxas de juros 54

20 Metodologia de abordagem do presente trabalho 57

21 Representação do modelo de previsão de retornos por LSTMs 59

22 FACP aplicado aos valores diários das variâncias de 20 dias dos retornos de PETR4 61

23 Retornos acumulados da estratégia resultante do presente trabalho e da taxa básica de juros brasileira no período avaliado 65

24 Comparação entre os retornos da estratégia proposta e de dois dos principais índices da bolsa de valores brasileira 66

25 Comparação entre os retornos da estratégia proposta e os retornos de uma carteira feita com os mesmos ativos em pesos iguais 67

26 Comparação entre os retornos da estratégia proposta e os retornos de uma carteira utilizando a previsão ingênua 67

Lista de Tabelas

1 Ativos pré selecionados para análise de retornos e variâncias 52

2 Taxa média de juros de contratos futuros de DI no dia 15 de junho de 2015 para certas datas de vencimento 55

3 Valores dos hiperparâmetros da rede neural para previsão de retornos futuros . . 62

4 Ativos selecionados como variáveis explicatórias das redes neurais 75

Sumário

1 Introdução **17**

1.1 Contexto e Definição do Problema 17

1.2 Organização do Trabalho 18

2 Aprendizado de Máquina (Machine Learning) **21**

2.1 Princípios básicos do ML 21

2.2 Funcionamento básico de uma rede neural artificial 22

2.3 Gradiente descendente e *backpropagation* 26

2.4 Ciclos de treino em redes do tipo *feedforward* 27

2.5 Regularização e erro de validação 28

2.6 RNN 30

2.7 LSTM 32

3 Portfólios de Investimento **35**

3.1 Conceitos básicos 35

3.1.1 Ativos financeiros 35

3.1.2 Mercado de capitais 35

3.1.3 Índices 36

3.1.4 Taxa básica de juros 36

3.1.5 Estratégias de investimentos com ações 37

3.1.6 Risco 37

3.2 Fronteira Eficiente de Markowitz 39

4 Heterocedasticidade condicional auto-regressiva **41**

4.1 Variâncias não constantes 41

4.2 Funções de autocorrelação 42

4.3 Modelos ARCH e GARCH 44

5 Indicadores técnicos **47**

5.1 Média móvel 47

5.2 EWMA 48

5.3 MACD 48

5.4 Bandas de Bollinger 49

6 Ativos Estudados **51**

6.1 Os ativos do portfólio 51

6.2 Ativos correlacionados 52

7 Desenvolvimento **57**

7.1 Previsão de retornos 58

7.2 Previsão de matrizes de covariâncias 60

7.3 Otimização do portfólio 62

8 Resultados **65**

8.1 Retorno da estratégia 65

9 Conclusão **69**

Referências **71**

Apêndice A – Lista de ativos utilizados como variáveis explicatórias **75**

Apêndice B – Rede Neural Previsora de Retornos **77**

Apêndice C – Rede Neural Previsora de Covariâncias **83**

Apêndice D – Modelos ARCH/GARCH **89**

Apêndice E – Rebalanceador de Portfólio **93**

1.0 Introdução

1.1 Contexto e Definição do Problema

Fundos de investimento são ”uma comunhão de recursos, captados de pessoas físicas ou jurídicas, com o objetivo de obter ganhos financeiros a partir da aplicação em títulos e valores mobiliários”(Comissão de Valores Mobiliários). As gestoras de fundos se utilizam de capital adquirido de clientes para aplicá-los em diversos ativos financeiros e obter lucros para os clientes, arrecadando no processo um percentual do total de recursos financeiros. Os fundos de investimento podem ser divididos em duas categorias, como segue.

- Fundos passivos - fundos nos quais são aplicadas estratégias simples, normalmente replicando um coletivo de ativos de referência, como por exemplo os que compõem o índice Ibovespa (para definição, ler seção 3.1).
- Fundos ativos - fundos nos quais são aplicadas estratégias de maior sofisticação com o objetivo de maximizar os lucros. Normalmente retornam lucros superiores aos fundos passivos, no entanto, normalmente suas taxas são também superiores.

O percentual dos recursos monetários arrecadados pela gestora é definido de acordo com duas taxas, como segue.

- Taxa de administração - arrecadado para financiar os custos operacionais da gestora, como salários e sistemas de informação, e propiciar lucros
- Taxa de performance - percentuais dos lucros que excedem um *benchmark*. Assim, estimulam as gestoras a almejarem a maximização dos lucros obtidos. Apenas gestoras de fundos ativos coletam a taxa de performance.

O autor deste trabalho esteve presente em uma gestora de fundos ativos, nos quais os recursos financeiros são fracionados para serem aplicados de acordo com diferentes estratégias como forma de diversificação da alocação de seus recursos. Atualmente, a gestora compra e

vende ativos de acordo com estratégias fundamentalistas, que são estratégias nas quais os ativos comprados e vendidos são definidos com base em análises de fatores econômicos e financeiros relacionados ao ativo. Tem-se como exemplo a compra de propriedade sobre uma empresa justificada pela expectativa da governança ser eficiente em sua gestão, impulsionando lucros a serem crescentes ao longo do tempo.

A gestora na qual o autor trabalhou possui a intenção de alocar uma fração de seus recursos financeiros em estratégias quantitativas/sistêmicas, que são estratégias nas quais conceitos qualitativos de difícil mensuração quantitativa – como por exemplo a eficiência gerencial da governança corporativa de uma empresa – são desconsiderados, dando-se preferência a métodos computacionais, matemáticos e estatísticos para tomar decisões de investimento. Dessa forma, o objetivo de tal trabalho é a criação de uma estratégia quantitativa de investimentos que traga retornos superiores a *benchmarks* como o Ibovespa e, para tanto, o autor buscou na literatura métodos de realização de investimentos quantitativos.

Este trabalho baseia-se na utilização de técnicas modernas de aprendizado de máquina (para definições, ver capítulo 2) para criação de estratégias quantitativas de investimento. Três obras na literatura se apresentam como pilares fundamentais deste trabalho, sendo elas:

- ”Portfolio optimization with return prediction using deep learning and machine learning” por Ma, Han e Wang (2021), no qual os autores propõem um modelo para estimar as variações percentuais dos preços de um grupo de ativos com base em redes neurais artificiais
- ”Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models” por Kim e Won (2018), no qual os autores propõem a integração de modelos clássicos de previsão de variâncias futuras com redes neurais artificiais do tipo LSTM
- ”Portfolio selection : efficient diversification of investment” por Markowitz (1959), no qual o autor apresenta técnicas de diversificação e alocação eficiente de recursos financeiros.

1.2 Organização do Trabalho

Para a elaboração estruturada do trabalho, este está dividido em 9 capítulos, sendo eles:

1. **Introdução** - apresentação do problema e dos objetivos.

2. **Aprendizado de Máquina (Machine Learning)** - apresentação dos conceitos de aprendizado de máquina que serão utilizados no desenvolvimento dos modelos preditivos.

3. **Portfólios de Investimento** - capítulo dedicado à apresentação dos conceitos financeiros necessários ao entendimento da estratégia desenvolvida, bem como a apresentação de técnicas de alocação de recursos financeiros utilizadas.

4. **Heterocedasticidade condicional auto-regressiva** - apresentação dos modelos utilizados em conjunto às redes neurais para previsão de variâncias futuras de preços de ativos.

5. **Indicadores técnicos** - uma revisão sobre alguns dos conceitos amplamente utilizados dentre investidores para técnicas de investimentos quantitativos.

6. **Ativos Estudados** - uma breve descrição de todos os ativos cujas características foram utilizadas como variáveis explicatórias nos modelos preditivos

7. **Desenvolvimento** - apresentação dos passos tomados, baseados na literatura, para se chegar ao modelo proposto neste trabalho.

8. **Resultados** - uma comparação entre o modelo proposto e alguns *benchmarks*, como forma de verificar a eficiência e eficácia do modelo.

9. **Conclusão** - breve descrição das conclusões constatadas pelo autor ao longo do desdobramento deste trabalho, bem como sugestões de próximos passos a serem tomados.

2.0 Aprendizado de Máquina (Machine Learning)

2.1 Princípios básicos do ML

De acordo com James et al. (2013), o aprendizado estatístico é um conjunto de ferramentas para a compreensão de dados, entendendo compreensão como previsão, classificação, identificação de padrões, e outros. Seja Y uma resposta quantitativa e p previsores, X_1, X_2, ..., X_p, assumindo-se que exista alguma relação entre Y e $X = (X_1,\ X_2,\ ...,\ X_p)$. O autor define então um modelo estatístico entre Y e X na equação 2.1, em que ε corresponde ao erro do modelo.

$$Y = f(X) + \varepsilon \tag{2.1}$$

Neste caso, não é conhecida a função f, mas assume-se que ela exista. Dessa forma, o aprendizado estatístico pode ser compreendido como o conjunto de ferramentas para se obter um modelo que aproxime f.

Dentre os campos de aprendizado estatístico, encontra-se o ML (”Aprendizado de Máquina”, do inglês ”Machine Learning”), que desenvolve programas que são capazes de se autodesenvolverem a cumprir uma dada tarefa sem que recebam as ordens explícitas na sintaxe computacional para cumprir tal tarefa. Como definição, tem-se que ”Um programa de computador é dito para aprender com a experiência E com a relação a alguma classe de tarefas T e medida de desempenho P, se o seu desempenho em tarefas em T, medida pelo P, melhora com a experiência E” (MITCHELL, 1997).

Um exemplo é uma regressão linear simples como visto na equação 2.2, na qual $\hat{y}$ corresponde a uma previsão de y em função de um certo valor de x e ε ao erro da previsão, definido na equação 2.3. No evento de existir uma relação de linearidade entre as duas variáveis (em que $\varepsilon \neq 0$), é esperado que o aumento do número de indivíduos da amostra aumente a probabilidade da capacidade preditiva do modelo, tornando-o mais próximo da f que deseja-se descobrir.

Os valores dos parâmetros β_0 e β_1 são determinados de forma a minimizar o erro quadrático médio (EQM) da previsão na amostra de tamanho m. Dessa forma, a regressão melhora seu desempenho na tarefa T (prever o valor de y) medido por P (o EQM) com a experiência E (as

amostras).

$$\begin{aligned} y_i &= \hat{y}_i + \varepsilon_i \\ \hat{y}(x) &= \beta_0 + \beta_1 \cdot x \end{aligned} \tag{2.2}$$

$$J = \sum_{i=1}^{m} \frac{(\hat{y}_i - y_i)^2}{m} \tag{2.3}$$

É possível constatar que o valor resultante da equação 2.3 é dependente dos parâmetros α, β_0 e β_1 que são utilizados. A função que descreve o valor do erro total em função dos parâmetros é chamada de *Função Custo*, com notação $J(\theta)$, em que θ são os parâmetros do modelo de previsão. A maneira como se define o erro resultante acaba por definir também o comportamento de $J(\theta)$.

De maneira análoga à regressão linear, o ML consiste em utilizar uma amostra e criar equações que possam descrever a(s) variável(eis) dependente(s), também minimzando seus valores de erros de previsão.

Apesar de existirem outros grandes métodos de aprendizado de máquina, este trabalho abordará apenas o ”Aprendizado Supervisionado”, o qual funciona de maneira análoga à regressão linear, no qual um modelo estima um valor para a variável independente com base em variáveis dependentes sabendo-se, para uma dada amostra, o valor da variável independente.

2.2 Funcionamento básico de uma rede neural artificial

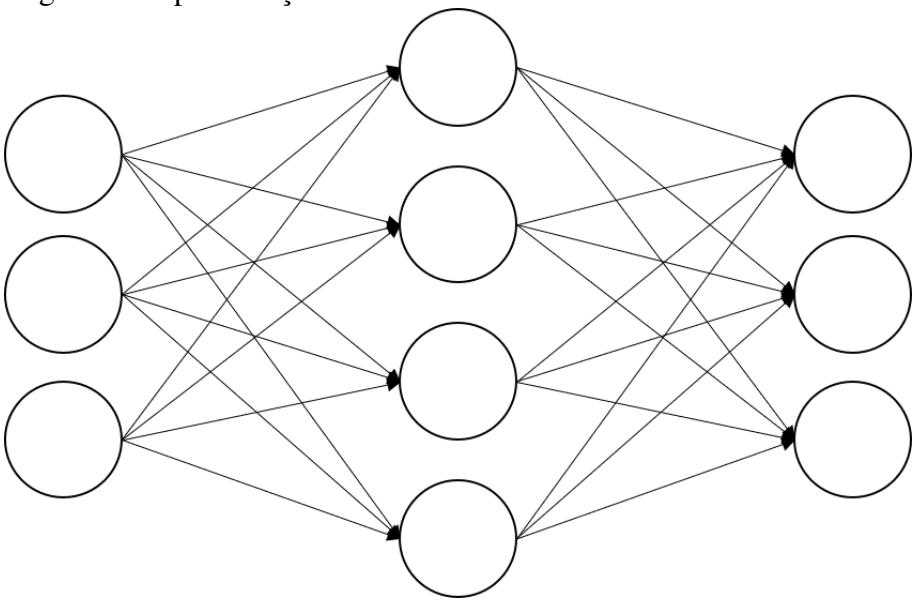
As redes neurais artificiais originaram-se com inspiração nas redes neurais biológicas, nas quais neurônios recebem sinais de outros neurônios através de seus dendritos, processam tais sinais e podem ou não serem ativados, emitindo um sinal de saída a outros neurônios. No trabalho de Fiesler (1994), o autor formaliza a definição de redes neurais artificiais, descrevendo que são compostas por dois componentes: sua estrutura e seu esquema de interconexões.

Segundo o autor, a estrutura é o número de grupos que a rede possui e o número de neurônios em cada grupo. No evento desses grupos estarem linearmente ordenados, são denominados como camadas. O esquema de interconexões se refere aos tipos de conexões usadas, podendo ser simétricas ou assimétricas, a ordem dessas conexões e a conectividade (quais neurônios estão conectados). Assim, um exemplo de uma rede neural com três camadas pode ser observado na Figura 1, sendo a conectividade representada pelas setas. Esta rede pode receber, por exemplo, os valores de preços de três produtos distintos e retornando, como saída, os

valores de preços previstos de tais produtos em um ano no futuro.

Redes que possuem camadas linearmente ordenadas como a da Figura 1 são denominadas redes do tipo *feedforward*. Em tais redes, a primeira camada é a camada de entrada, a última é a camada de saída e todas as outras são denotadas camadas ocultas.

Figura 1: Representação do funcionamento de uma rede neural básica



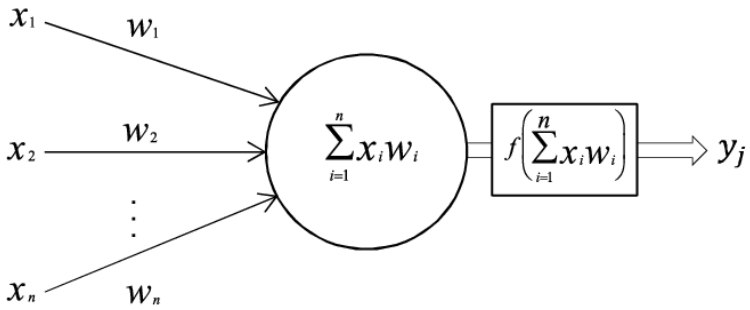
Fonte: elaborado pelo autor

O *forward propagation* (do inglês, "propagação dianteira") é iniciado pela inserção de valores nos neurônios da camada de entrada. Cada nó da rede irá transformar o valor recebido e transferi-los a todos os neurônios da camada seguinte, e assim sucessivamente até que se chegue à camada de saída. Essa transformação que ocorre em cada neurônio se dá por duas partes: primeiramente, soma os valores recebidos de todos os nós da camada anterior, multiplicados pelos pesos de cada interconexão w; em seguida, é aplicada uma função não linear sobre o resultado, obtendo-se o valor que o neurônio emite a todos da camada seguinte, multiplicando-os pelos pesos de suas respectivas interconexões. O funcionamento destes neurônios pode ser observado na Figura 2.

A função de ativação é uma função que executa uma transformação não linear nos dados que recebe. Tais funções são muito úteis para que redes neurais possam aprender padrões e correlações não lineares através de entradas lineares (NWANKPA et al., 2020). Algumas das mais comuns e que serão as usadas no presente trabalho são a função sigmoidal e a função tangente hiperbólica.

A função sigmoidal recebe quaisquer valores reais e os transforma em algum valor entre 0 e 1, não incluindo-os. É formulada pela equação 2.4, em que e corresponde ao número de Euler.

Figura 2: Representação do funcionamento de um neurônio artificial básico

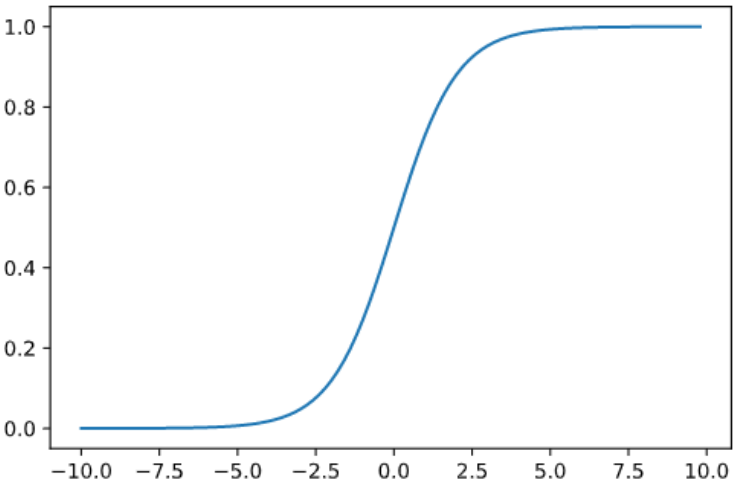


Fonte: Vieira, Pinayab e Mechelli (2017)

Possui um formato em ”S”, como visto na Figura 3.

$$S(x) = \frac{1}{1+e^{-x}} \tag{2.4}$$

Figura 3: Representação gráfica da função sigmoidal com números reais

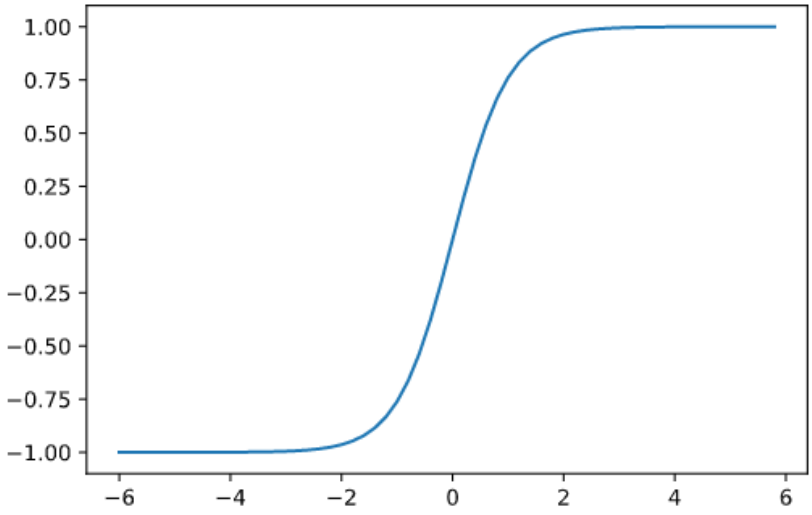


Fonte: elaborado pelo autor

Outra transformação não linear amplamente utilizada em redes neurais é a função tangente hiperbólica, que recebe entradas reais e os transforma a um valor entre -1 e 1, não incluindo-os, com uma forma em ”S” semelhante ao da função sigmoidal, como visto na Figura 4. É formulada de acordo com a equação 2.5.

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.5}$$

Figura 4: Representação gráfica da função tangente hiperbólico com números reais



Fonte: elaborado pelo autor

Pode-se então explicar o valor da variável de saída como função das variáveis de entrada. Sejam:

- $a_i^{(k)}$ o neurônio i da camada k
- $w_{i,j}^{(k)}$ o peso da conexão entre $a_i^{(k)}$ e $a_j^{(k+1)}$
- $\theta^{(k)} = \begin{bmatrix} w_{1,1}^{(k)} & \cdots & w_{1,j}^{(k)} \\ \vdots & \ddots & \vdots \\ w_{i,1}^{(k)} & \cdots & w_{i,j}^{(k)} \end{bmatrix}$ a matriz com os pesos $w_{i,j}$ da camada k
- g a função de ativação de cada neurônio
- v o vetor de valores de saída de uma camada

O vetor de valores de saída v da camada k é, portanto, definido pela equação 2.6.

$$v^{(k)} = g((\theta^{(k-1)})^T v^{(k-1)}) \tag{2.6}$$

O método *forward propagation* é, então, a utilização iterativa da equação 2.6 para se obter o vetor da camada de saída, sendo $v^{(0)}$ o vetor de valores das variáveis explicatórias.

O aprendizado (ou treino) é a redução do valor de erro calculado com base no vetor de saídas, comparando-o ao valor real da variável independente da amostra. Após o cálculo do vetor de saídas, calcula-se o termo de erro em relação à variável independente da amostra, e ocorre alteração dos pesos $w_{i,j}^{(k)}$ para a redução dos valores de erros. Em outras palavras, o

aprendizado é a redução de $J(\theta)$ variando-se θ, sendo θ os pesos das interconexões da rede neural.

2.3 Gradiente descendente e *backpropagation*

Uma derivada parcial é a inclinação de uma função de múltiplas variáveis em função de uma única variável em um dado ponto (STEWART, 2013). Isso pode ser definido de acordo com a equação 2.7.

$$\frac{\partial}{\partial a} f(a,b) = \lim_{h \to 0} \frac{f(a+h,b) - f(a,b)}{h} \tag{2.7}$$

Se f é uma função de n variáveis, então o gradiente de f é a função vetorial ∇f definida por2.8.

$$\nabla f(x_1, x_2, \ldots, x_n) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(x_1, x_2, \ldots, x_n) \\ \frac{\partial f}{\partial x_2}(x_1, x_2, \ldots, x_n) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x_1, x_2, \ldots, x_n) \end{bmatrix} \tag{2.8}$$

Como tal vetor representa a variação de f em função da alteração de cada uma des suas variáveis, o gradiente representa a direção vetorial no espaço definido por $(x_1, x_2, ..., x_n)$ na qual a função f possui o maior crescimento. Portanto, muitos dos algoritmos de ML recorrem ao algoritmo 2.1 de Gradiente Descendente, no qual o algoritmo calcula o gradiente do custo $J(\theta)$ ao se utilizando-se parâmetros θ e altera-os no sentido contrário ao do gradiente, a fim de reduzir o valor de J iterativamente.

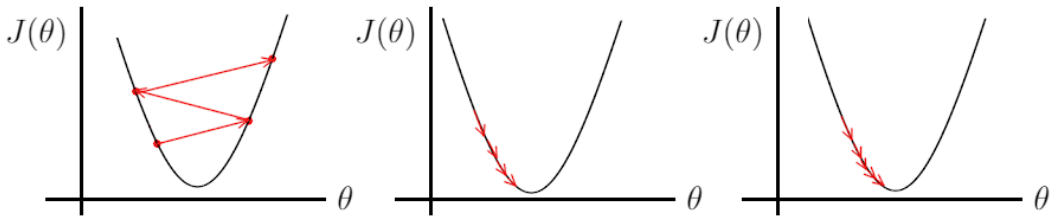
Algoritmo 2.1 Gradiente Descendente

Requer: $J(\theta)$ diferenciável para todos os θ_i
  while $J(\theta) > J_{objetivo}$ **do**
    Calcular $\nabla J(\theta)$
    $\theta \leftarrow \theta - \alpha \cdot \nabla J(\theta)$
    Calcular $J(\theta)$
  end while

No caso de tal algoritmo, utiliza-se uma constante de aprendizado α para regular o quanto deve ser subtraído do vetor θ no sentido do gradiente. Caso o valor de tal constante seja muito pequeno, o algoritmo pode ser muito custoso computacionalmente e demorado até atingir um

mínimo local. Caso o oposto seja verdade, o valor pode nunca chegar a um mínimo local por conta do algoritmo constantemente ultrapassar tal valor, entrando em um ciclo. Como exemplo ilustrativo, tem-se a Figura 5.

Figura 5: Representação ilustrativa dos efeitos de diferentes taxas de aprendizado



Fonte: elaborado pelo autor

No trabalho de Rumelhart, Hinton e Williams (1986), os autores propõem o método de *backpropagation* (do inglês, "retropropagação"), por meio do qual se calcula o valor de $\nabla J(\theta)$ em redes do tipo *feedforward* de forma computacionalmente eficiente, aplicando em seguida o gradiente descendente. Dessa forma, consolidou-se o uso do gradiente descendente como técnica amplamente utilizada em ML para alteração dos pesos das interconexões definidos por θ.

2.4 Ciclos de treino em redes do tipo *feedforward*

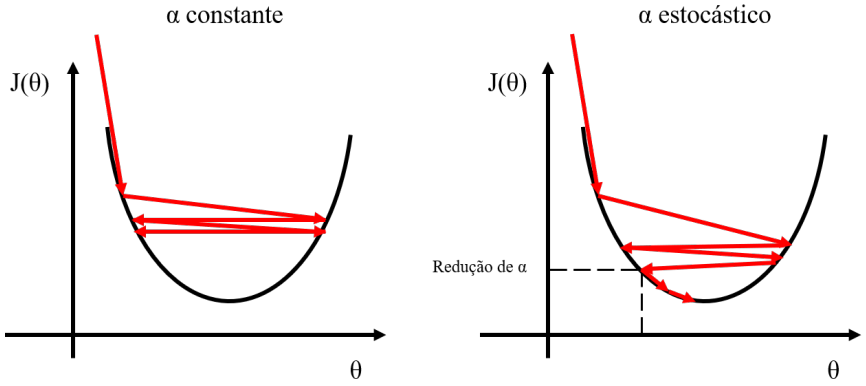
Para que seja efetuado o aprendizado (ou treino) da rede neural, se faz necessário primeiramente uma amostra de variáveis explicatórias $X = (X_1, X_2, ..., X_m)$ com seus correspondentes $Y \in R^n$. Sendo isso assegurado, cria-se uma rede neural com m nós na camada de entrada, n neurônios na camada de saída e camadas ocultas com uma quantidade pré determinada de nós. Em seguida, os valores dos pesos w das interconexões da rede são definidos aleatoriamente.

O treino consiste em inserir um vetor da amostra - X_i - obter-se o valor de previsão da rede através do método *forward propagation*, subsequente cálculo do erro J com base no valor de Y_i, e, por fim, uso das técnicas de *backpropagation* e gradiente descendente para alteração dos parâmetros θ. O aprendizado se dá de forma iterativa, aplicando este mesmo método para todas as unidades da amostra.

Ao aplicar tal método por toda a amostra, é dito que a rede completou um ciclo de treino. Múltiplos ciclos podem ser executados para melhoria dos parâmetros preditivos da rede. Com o objetivo de aperfeiçoar o treinamento da rede, Kingma e Ba (2014) propuseram "Adam", uma técnica em que a constante de aprendizado α é estocástica para se evitar flutuações indesejavelmente grandes de θ, mostrando que redes nas quais α é constante tendem a iterar o processo

na forma de um ciclo, no qual os parâmetros θ são alterados sem que isto mude o custo. Além disso, utilizaram também uma redução drástica não estocástica de α quando ciclos de treinos adicionais não apresentam suficiente redução do erro J. O efeito da utilização de tais técnicas é observável na Figura 6.

Figura 6: Representação ilustrativa dos efeitos do uso de Adam sobre o aprendizado da rede



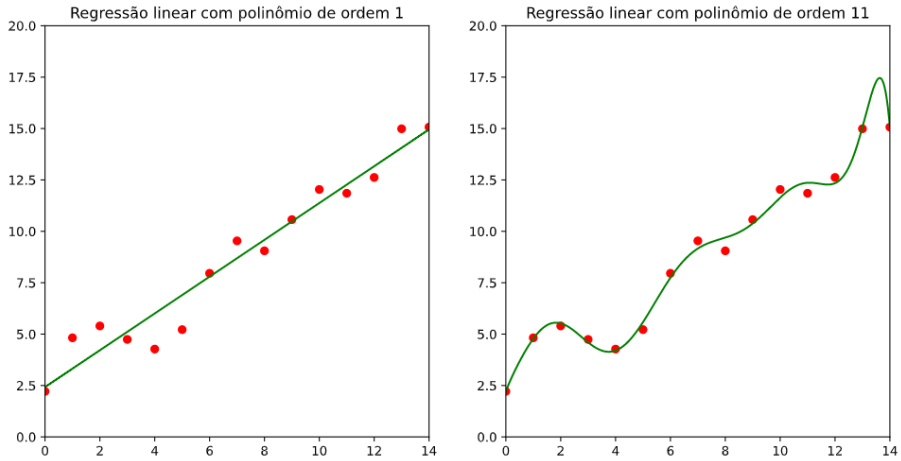
Fonte: elaborado pelo autor

2.5 Regularização e erro de validação

De acordo com Yeom et al. (2020), um modelo sofre de sobreajuste quando sua performance sobre dados não presentes na amostra diverge da performance na amostra, significando que o erro de generalização de tal modelo é grande. Como exemplo, toma-se a função $f(x) = x + 2 + \varepsilon_i$, sendo $\varepsilon_i \sim N(0,1)$. Podem ser utilizados pontos resultantes desta função para criar um modelo polinomial e, quanto maior a ordem do polinômio, maior a flexibilidade do modelo para se diminuir o EQM sobre os pontos da amostra. Apesar de ordens maiores levarem a erros menores, isto é realidade apenas para os pontos que fazem parte da amostra e o oposto se torna verdade para pontos que não o fazem. Isto pode ser observado na Figura 7, na qual dois modelos polinomiais são criados para explicarem a função $f(x)$.

Ying (2019) destacou alguns métodos de mitigação dos efeitos de sobreajuste, entre os quais regularização e parada antecipada. De acordo com o autor, a regularização – no contexto de ML – consiste em adicionar, na função de custo $J(\theta)$, uma penalidade aos coeficientes de cada variável explicatória. Isto tem a funcionalidade de, primeiramente, reduzir a relevância preditória de variáveis que não sejam fortemente correlacionadas ao objeto de previsão e, segundamente, para lidar com a multicolinearidade entre variáveis explicatórias, o que, segundo Reynaldo (1997), torna instável a estimativa dos coeficientes, sendo sensível a pequenas alterações nos valores das variáveis independentes.

Figura 7: Representação ilustrativa dos efeitos de sobreajuste sobre um mesmo conjunto de pontos



Fonte: elaborado pelo autor

Para o caso de um modelo polinomial de ordem n aplicado a uma amostra de tamanho m, a nova função de custo pode ser então observada na equação 2.9, na qual λ é o coeficiente de regularização. A regularização que penaliza o quadrado dos coeficientes é conhecida como do tipo L2. Em contraste, a regularização do tipo L1 penaliza os valores absolutos dos coeficientes.

$$\begin{aligned} \hat{y}(x) &= \theta_0 + \sum_{j=1}^{n} \theta_j x^j \\ J(\theta) &= \left(\sum_{i=1}^{m} \frac{(\hat{y}_i - y_i)^2}{m}\right) + \lambda \sum_{j=1}^{n} \theta_j^2 \end{aligned} \tag{2.9}$$

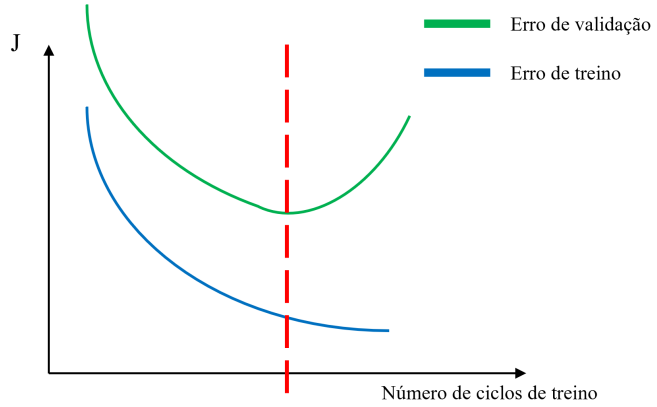
De maneira análoga ao de modelos polinomiais, a regularização é aplicada também a redes *feedforward*. Neste caso, a regularização pode ser aplicada tanto aos pesos das interconexões – penalizando a quantidade de conexões relevantes ao modelo – quanto às saídas de cada neurônio – penalizando a quantidade de nós relevantes ao modelo. Tal função custo pode ser visto na equação 2.10.

$$J = \left(\sum_{i=1}^{m} \frac{(\hat{y}_i - y_i)^2}{m}\right) + \lambda_1 \left(\sum_i \sum_j \sum_k w_{i,j}^{(k)}\right) + \lambda_2 \left(\sum_k \mathbf{1} v^{(k)}\right) \tag{2.10}$$

A técnica de parada antecipada consiste em dividir a amostra em amostras de treino, de validação e de teste, completar ciclos de treino da rede utilizando-se apenas a amostra de treino (reduzindo o erro de treino), calcular o erro do modelo quando aplicado à amostra de validação (erro de validação) e interromper os ciclos de treino quando este erro não estiver decrescendo consistentemente. Esta técnica se mostra eficiente por interromper o treino da rede antes que altere os parâmetros a fim de reduzir erros devidos a ruído estatístico (YING, 2019). Por fim, a capacidade de generalização do modelo é averiguado por meio de sua aplicação na amostra de

teste, obtendo-se o erro de teste.

Figura 8: Representação da técnica de parada antecipada



Fonte: elaborado pelo autor

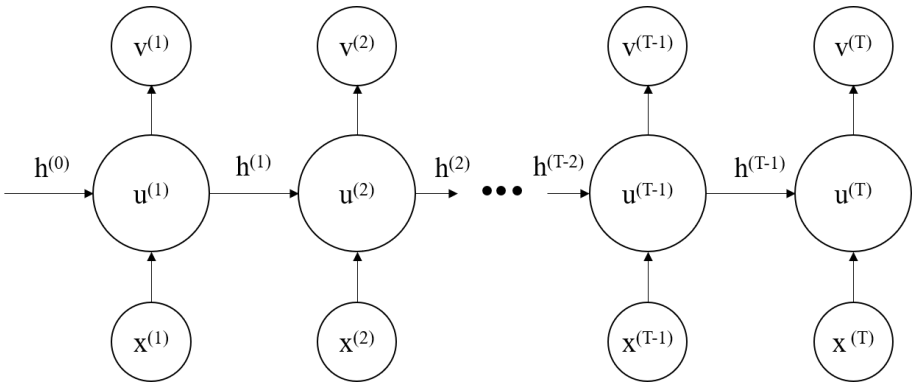
2.6 RNN

RNNs (Redes Neurais Recorrentes, do inglês ”Recurrent Neural Netwoorks”) são uma família de redes neurais para processamento de dados sequenciais (GOODFELLOW; BENGIO; COURVILLE, 2016). Foram propostas por Rumelhart, Hinton e Williams (1986) e são úteis para a criação de modelos que dependam de uma sequência ordenada, como um modelo que complete uma frase dadas as primeiras palavras. A diferença principal entre RNNs e redes do tipo *feedforward* consiste no fato de cada camada receber como entrada não somente os valores das variáveis explicatórias mas também do *estado oculto h*, um vetor que atravessa as unidades da RNN sofrendo transformações. Uma camada, no contexto de RNNs, é composta por T camadas de redes *feedforward* – ou unidades/células u – que se conectam sequencialmente na mesma ordem da sequência de entrada $(x^{(1)}, x^{(2)}, ..., x^{(T)})$, em que cada unidade recebe o vetor de saída (estado oculto) da camada anterior, conforme representado pela Figura 9.

Sejam:

- $y = [y_1\ y_2\ ...\ y_m]$ uma variável de dimensão m que deseja-se prever
- $x^{(t)} = [x_1^{(t)}\ x_2^{(t)}\ ...\ x_n^{(t)}]$ uma variável explicatória de dimensão n na ordem t de uma sequência de T termos
- $u^{(t)}$ a unidade do RNN na ordem t
- $a_i^{(t)}$ o nó i do estado oculto t

Figura 9: Representação ilustrativa de uma rede neural recorrente simples



Fonte: elaborado pelo autor

- $h^{(t)} = [a_1^{(t)}\ a_2^{(t)}\ ...\ a_m^{(t)}]$ o estado oculto (ou vetor de saída) da unidade t
- $wh_{i,j}$ o peso da conexão entre o nó i de um estado oculto ao nó j do estado oculto seguinte (este peso é igual ao longo de toda a RNN)
- $\theta_{hh} = \begin{bmatrix} wh_{1,1} & \cdots & wh_{1,m} \\ \vdots & \ddots & \vdots \\ wh_{m,1} & \cdots & wh_{m,m} \end{bmatrix}$ a matriz de pesos da conexão do estado oculto de uma unidade à próxima
- $wx_{i,j}$ o peso da conexão entre os nó i da camada de entrada ($x^{(t)}$) ao nó j do estado oculto (t) (este peso é igual ao longo de toda a RNN)
- $\theta_{hx} = \begin{bmatrix} wx_{1,1} & \cdots & wx_{1,n} \\ \vdots & \ddots & \vdots \\ wx_{m,1} & \cdots & wx_{m,n} \end{bmatrix}$ a matriz de pesos $wx_{i,j}$
- g_h a função de ativação usada entre as unidades da camada RNN
- g_v a função de ativação usada entre unidades e as saídas da camada RNN

Os valores de $h^{(t)}$ e $v^{(t)}$ podem então serem descritos pelas equações 2.11 e 2.12, respectivamente.

$$h^{(t)} = g_h(\theta_{hh}h^{(t-1)} + \theta_{hx}x^{(t)}) \tag{2.11}$$

$$v^{(t)} = g_v(\theta_{hh}h^{(t-1)} + \theta_{hx}x^{(t)}) \tag{2.12}$$

Caso g_h e g_v sejam a mesma função de ativação, $h^{(t)} = v^{(t)}$, e ambas foram originalmente propostas como sendo a tangente hiperbólica definida pela equação 2.5. Inicia-se computando

um valor arbitrário de $h^{(0)}$ (é comum o emprego de $h^{(0)} = [0\ 0\ ...\ 0]$) e prosseguindo-se com a propagação dos valores até a obtenção de $v^{(T)}$, com o subsequente aprendizado através do BPTT ("Retropropagação Pelo Tempo", do inglês "Backpropagation Through Time"). Normalmente, essa é a variável que será usada como previsão do modelo, já que leva em consideração todos os termos $x^{(t)}$ da sequência ordenada.

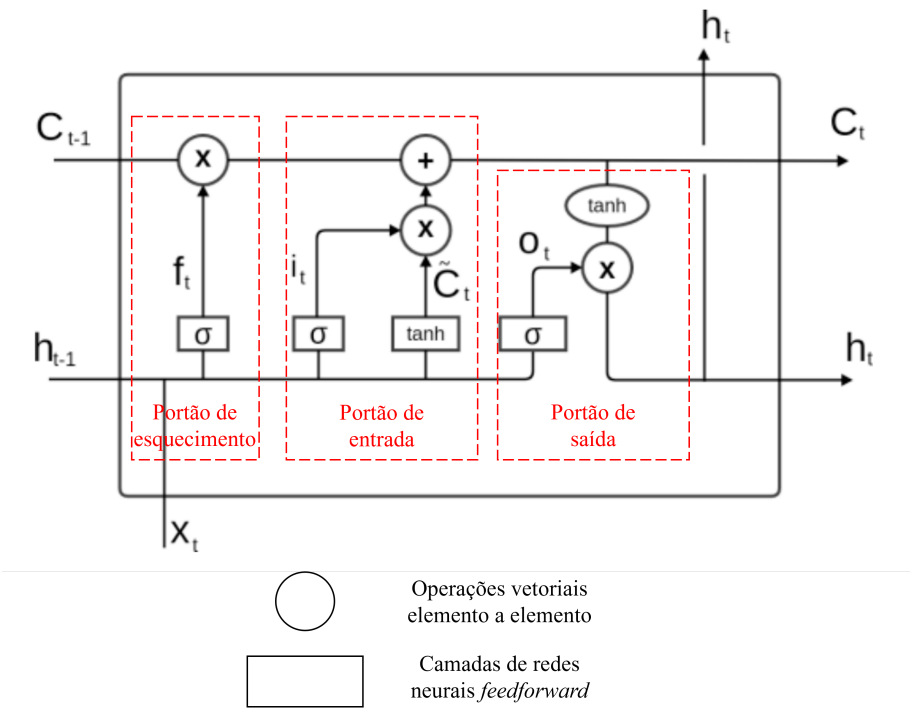
Como exemplo ilustrativo, tem-se uma rede RNN que recebe n dados de T períodos (sendo $t = (t_0,\ t_1,\ ...,\ t_T)$) referentes a m ativos, com o objetivo de prever-se o preço destes ativos em $t = t_{(T+1)}$. A ordem dos valores é de extrema importância para que o modelo preveja corretamente, como por exemplo em um período de tendência de subida dos preços destes ativos. A sequencialidade dos preços crescentes permite que o modelo preveja um provável crescimento em $t = t_{(T+1)}$.

2.7 LSTM

As RNNs tradicionais sofrem principalmente do problema denominado dissipação/explosão do gradiente. Como mencionado, as RNNs se utilizam do BPTT, e no cálculo da derivada parcial $\frac{\partial}{\partial \theta_{hh}} J$ no vetor de saída $v^{(T)}$, através da regra da cadeia, há um produtório das derivadas parciais $\frac{\partial}{\partial h^{(1)}} h^{(2)}$, $\frac{\partial}{\partial h^{(2)}} h^{(3)}$, ... $\frac{\partial}{\partial h^{(T-1)}} h^{(T)}$. Hochreiter e Schmidhuber (1997) observaram que em modelos RNN tradicionais em que as derivadas parciais destes termos são menores que um, o produtório é levado a próximo de zero – o que faria com que o gradiente descendente tivesse pouco efeito sobre as matrizes de pesos de períodos distantes de T. O oposto ocorre quando as derivadas parciais são maiores do que um, levando o produtório a tender a infinito. Dito de outra forma, a informação de $x^{(t)}$ tende a perder relevância rapidamente no modelo preditivo nas unidades posteriores da camada RNN. Por esse motivo, é dito que as RNNs tradicionais possuem memória de curto prazo.

Hochreiter e Schmidhuber (1997) não só observaram este problema como propuseram um modelo alternativo, a LSTM ("Memória de Curto e Longo Prazo", do inglês "Long Short-Term Memory"). São redes recorrentes nas quais cada unidade possui, no lugar de uma rede *feedforward*, quatro destas. Além disso, possuem não só um estado oculto h_t mas também um estado de célula C_t. Este possui a funcionalidade de manter informação de longo prazo sendo transmitida pela rede, não passando pelas mesmas operações que o estado oculto passa. As células LSTMs possuem três do que os autores chamaram de portões, que são redes *feedforward* em combinação com operações vetoriais elemento a elemento com o propósito de filtrar a informação que terá efeito sobre os estados oculto e de célula.

Figura 10: Representação de uma célula LSTM



Fonte: adaptado de Hiransha et al. (2018)

O estado de célula observado na Figura 10 representa um fluxo de informação contínuo que atravessa todas as unidades da camada LSTM. O primeiro portão, o de esquecimento, regula a informação da célula de estado anterior C_{t-1} que será mantida em C_t. O portão de entrada insere novas informações que serão adicionadas ao estado de célula após o primeiro portão, retornando então o novo estado de célula C_t. Por último, o portão de saída se utiliza do estado de célula atual C_t, do estado oculto anterior h_{t-1} e do vetor das variáveis de entrada x_t para formar o estado oculto da célula e que será passado às unidades seguintes da camada LSTM (HIRANSHA et al., 2018).

3.0 Portfólios de Investimento

3.1 Conceitos básicos

3.1.1 Ativos financeiros

Ativos financeiros são ativos que surgem de um acordo contratual de futuros fluxos de dinheiro ou de propriedade sobre uma instituição (CORPORATE FINANCE INSTITUTE, 2021). Como exemplo de ativo financeiro, tem-se o empréstimo, modalidade na qual aquele que emprestou o dinheiro possui um contrato assegurando que receberá de volta a mesma quantidade emprestada com um excedente – o juros. Outro ativo financeiro amplamente conhecido são as ações, que representam uma fração de propriedade do capital de uma empresa. O proprietário de uma ação de uma empresa tem direito a uma fração dos lucros distribuídos – os *dividendos*.

Neste trabalho, os únicos ativos financeiros que serão abordados são as ações. Existem várias no mercado brasileiro que possuem uma elevada liquidez (facilidade e velocidade em transformar o ativo em valor monetário), além de altas flutuações de preço quando comparados a produtos vendidos fisicamente.

3.1.2 Mercado de capitais

O mercado de capitais é todo o sistema no qual são comprados e vendidos os ativos financeiros – ações, dívidas, etc. – referentes a entidades privadas. O mercado acionário faz parte do de capitais e representa o sistema pelo qual se negociam participações em empresas. No presente trabalho, o único mercado de capitais que será abordado é o acionário, e portanto os termos serão utilizados como sinônimos. O mercado acionário é composto por:

- Mercado primário – sistema no qual a empresa vende suas participações de propriedade sobre a empresa ao público pela primeira vez, por meio de um IPO (Oferta Pública Inicial, do inglês ”Initial Public Offer”).
- Mercado secundário – sistema no qual ações que já foram vendidas ao público maior

podem ser negociadas entre quaisquer integrantes do mercado, sendo eles a empresa representada pela ação ou não.

É importante ressaltar que o crescimento saudável do mercado secundário desenvolve nos investidores a confiança de que suas ações serão valiosas no futuro, valorizando o mercado primário e, portanto, aumentando os arrecadamentos advindos de IPOs por parte das empresas.

O mercado acionário no Brasil se dá de forma centralizada através da bolsa de valores B3 (”Brasil, Bolsa, Balcão”). Assim, tanto o mercado primário quanto o secundário acontecem nesta bolsa. Os proprietários de diversos ativos financeiros podem negociá-los livremente com outros integrantes do mercado ao preço que as duas partes concordarem.

3.1.3 Índices

Os *índices* são, no contexto financeiro, métricas obtidas a partir de alguma medida aplicada considerando-se um conjunto de ativos pré selecionados de acordo com um critério. Um exemplo é o índice Ibovespa: ”É composto pelas ações [...] de companhias listadas na B3 que atendem aos critérios descritos na sua metodologia, correspondendo a cerca de 80% do número de negócios e do volume financeiro do nosso mercado de capitais” (B3, 2021).

O índice Ibovespa é a soma dos produtos dos preços de determinadas ações por seus pesos, definidos por critérios como seus volumes de negociação. Os critérios de cálculo dos pesos e de quais ações compõem o índice Ibovespa estão no site da B3.

3.1.4 Taxa básica de juros

De acordo com Securato et al. (2003), juros é a remuneração pelo uso do capital. Essa remuneração varia principalmente em função de dois fatores:

- Risco de não pagamento – maiores expectativas de que o devedor não cumprirá o pagamento de sua dívida levam a cobranças adicionais de juros
- Maturidade – maiores juros são aplicados quando a duração de tempo para devolver o capital aumenta

Em diversas economias, governos tomam empréstimos para financiar seus investimentos e pagar suas dívidas, prometendo pagá-los até uma determinada data. No Brasil e em outros países, isso se dá principalmente por ”Títulos Públicos”. Como o governo, quando comparado

a empresas, indivíduos, etc. representa menores chances de falir e não pagar suas dívidas, é considerado o órgão de menor risco no país. Portanto, os juros dos títulos públicos são os menores no país e definem a taxa de juros livre de risco, R_f. Servem como base para o restante das modalidades de empréstimo do país, nas quais as taxas tendem a serem maiores do que a básica. No Brasil, tal taxa recebe o nome de Selic.

3.1.5 Estratégias de investimentos com ações

Os preços de ações estão comumente atrelados a uma expectativa do quanto esta retornará de dinheiro ao proprietário, justificando seu preço. Existem no mercado diversos integrantes procurando comprar ações por um valor que acreditam ser menor do que o valor total que essa empresa o retornará ao longo do tempo, bem como integrantes procurando vendê-las por um preço que acreditam estar acima de tal valor.

Existem empresas cujo foco é comprar ações e desfrutar dos lucros distribuídos ao longo do tempo de maneira rentável. Há também instituições que operam comprando ações que acreditam que o preço negociado no mercado irá aumentar – independentemente do motivo – e que poderão vendê-las futuramente, auferindo lucros. O presente trabalho foca em estratégias de obtenção de renda deste tipo, puramente atreladas à flutuação dos preços das ações.

Define-se o retorno r de um ativo i no período Δt, de t_0 a t_f, como sendo sua flutuação percentual de t_0 para t_f no preço p no qual foi negociado na bolsa de valores, como indica a equação 3.1.

$$r_{i,\Delta t} = \frac{p_{i,t_f} - p_{i,t_0}}{p_{i,t_0}} \tag{3.1}$$

3.1.6 Risco

Um portfólio de investimentos, ou carteira, é um conjunto de ativos financeiros em posse de um indivíduo ou instituição financeira com o objetivo de auferir lucros. Seu valor financeiro total é a soma dos valores financeiros de cada ativo unitário. O retorno R de um porfólio P no período Δt composto por m ativos com pesos constantes $w_{i,\Delta t}$ (quantidade financeira no portfólio do ativo i no momento t_0) pode ser decrito pela equação 3.2. Em outras palavras, seu retorno é a média ponderada dos retornos dos ativos que o compõem.

$$R_{P,\Delta t} = \sum_{i=1}^{m} r_{i,\Delta t} w_{i,\Delta t} \tag{3.2}$$

Jorion (2007) define o risco como a variância de resultados inesperados. Em seu trabalho,

o autor cita diversos tipos de risco, como o de negócios (tomada de decisões que pode levar a resultados negativos inesperados) bem como o risco financeiro, como por exemplo perdas decorrentes de uma queda nos preços de um ativo em posse, e também o risco de mercado/sistêmico, como os resultados inesperados advindos de uma crise econômica. Uma das métricas desenvolvidas para se controlarem e mitigarem os riscos foi o VaR ("Valor em Risco", do inglês "Value At Risk"), que é "o quão grande pode ser a perda dado um horizonte de tempo e a uma dada probabilidade" (ROMAN; MITRA, 2009).

A métrica de risco de principal foco neste trabalho é a volatilidade do retorno dos ativos e do portfólio em um período Δt, sendo volatilidade o termo designado para se referir ao desvio padrão, visto para uma única ação i pela equação 3.3, em que $\bar{r}_{i,\Delta t}$ representa a média dos retornos r_i no período Δt.

$$\sigma_{i,\Delta t} = \sqrt{\frac{\sum_{t=t_0}^{t_f} (r_{i,\Delta t} - \bar{r}_{i,\Delta t})^2}{n-1}} \tag{3.3}$$

A volatilidade representa uma medição de dispersão em torno da média. Assim, quanto maior a volatilidade de um investimento, maior a exposição do investidor a grandes lucros mas também a grandes perdas. Além disso, os ativos são correlacionados entre si, sendo a covariância dos retornos de um ativo x e de outro y no período Δt de n intervalos de tempo definida pela equação 3.4.

$$cov(x,y)_{\Delta t} = \frac{\sum_{t=t_0}^{t_f} (r_{x,t} - \bar{r}_{x,\Delta t}) \cdot (r_{y,t} - \bar{r}_{y,\Delta t})}{n-1} \tag{3.4}$$

Assim, é possível representar a covariância entre todos os pares de ativos (i, j) de um portfólio no período Δt através da matriz de covariâncias Σ, definida pela equação 3.5.

$$\Sigma = \begin{bmatrix} cov(1,1)_{\Delta t} & cov(1,2)_{\Delta t} & \cdots & cov(1,m)_{\Delta t} \\ cov(2,1)_{\Delta t} & cov(2,2)_{\Delta t} & \cdots & cov(2,m)_{\Delta t} \\ \vdots & & \ddots & \vdots \\ cov(m,1)_{\Delta t} & cov(m,2)_{\Delta t} & \cdots & cov(m,m)_{\Delta t} \end{bmatrix} \tag{3.5}$$

Pelas equações 3.3 e 3.4, é possível observar que $cov(i,i) = \sigma_i^2$. Além disso, $cov(1,2) = cov(2,1)$, o que torna a matriz Σ simétrica. Assumindo para cada ativo i que seus retornos $r_{i,\Delta t}$ são variáveis aleatórias, a variância γ dos retornos de um portfólio com pesos $w^T = [w_1\ w_2\ ...,w_n]$

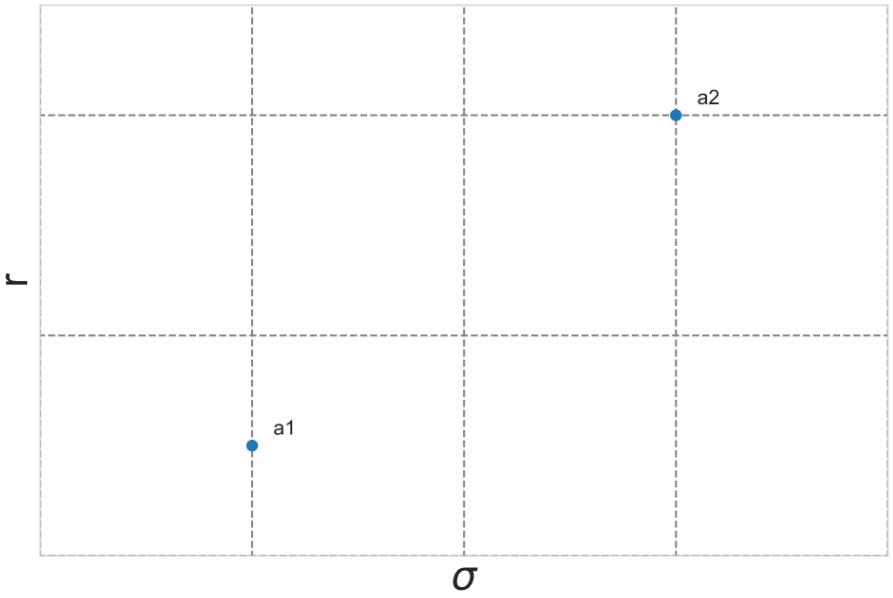
no período Δt é definida pela equação 3.6.

$$\gamma_{\Delta t} = w^T \cdot \Sigma \cdot w \tag{3.6}$$

3.2 Fronteira Eficiente de Markowitz

Sejam a_1 e a_2 dois investimentos cujos retornos ao longo de um período Δt são variáveis aleatórias independentes normalmente distribuídas, de valores esperados R_1 e R_2 e desvios padrões σ_1 e σ_2 ($r_{i,\Delta t} \sim \mathcal{N}(R_i, \sigma_i^2)$), como visto na Figura 11.

Figura 11: Ativos fictícios cujos retornos e volatilidades esperadas são conhecidas



Fonte: elaborado pelo autor

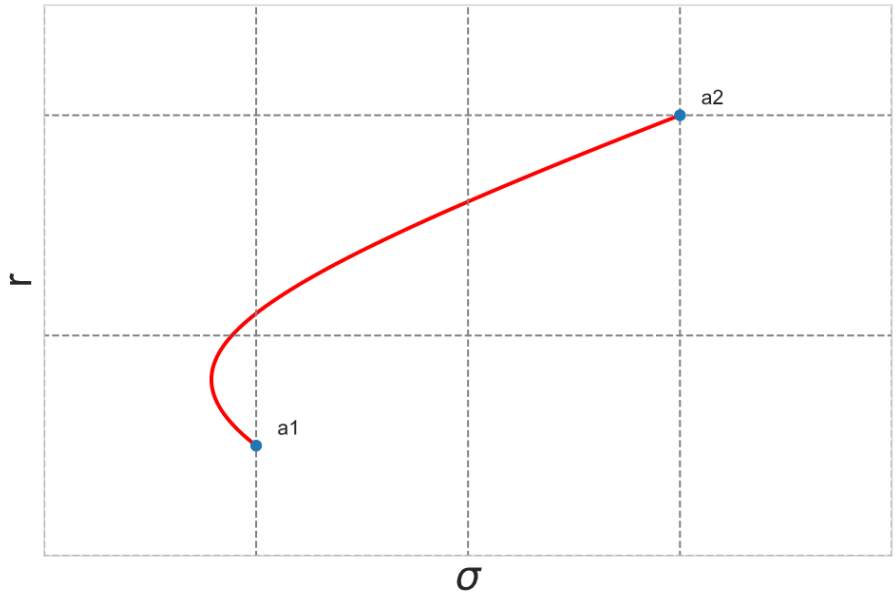
Seja P um portfólio com tais investimentos e $w^T = [w_1 \; w_2]$ o vetor de pesos de cada investimento dentro do portfólio no período Δt. Uma combinação linear destes dois investimentos não resulta em um investimento cujos ponto definido pelos valores de retorno esperado e devio padrão se encontre na reta entre os pontos destacados na Figura 11. Isto se dá pois o desvio padrão não é formado linearmente pelos desvios padrões destes ativos, como indicado pela equação 3.6. Por tal equação, tem-se definida a variância γ do portfólio P na equação 3.7.

$$\gamma = \begin{bmatrix} w_1 & w_2 \end{bmatrix} \cdot \begin{bmatrix} \sigma_1^2 & cov(1,2) \\ cov(2,1) & \sigma_2^2 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = w_1^2\sigma_1^2 + 2w_1w_2cov(1,2) + w_2^2\sigma_2^2 \tag{3.7}$$

Assim, uma combinação linear de a_1 e a_2 teria uma relação de risco e retorno descrito na Figura 12.

Dessa forma, chega-se no modelo de otimização de portfólios proposto por Markowitz

Figura 12: Representação dos retornos e volatilidades de uma combinação linear de dois ativos cujos retornos são normalmente distribuídos



Fonte: elaborado pelo autor

(1959). Em seu trabalho, o autor chama de fronteira eficiente o conjunto de portfólios que resultam no maior valor de retorno esperado dado um risco limite – expresso em variância – ou então portfólios que resultam no menor valor de volatilidade esperada dado um retorno mínimo aceitável. Dito isso, ampliou o conceito de diversificação dos investimentos, que ocorre naturalmente na descoberta da fronteira eficiente de um portfólio.

Tendo-se valores de retornos esperados R_i dos ativos a_i para o período Δt, suas variâncias e covariâncias esperadas Σ para o mesmo período, a forma generalizada de resolução da fronteira eficiente pode ser vista na equação 3.8, sujeito às restrições vistas em 3.9, nas quais FO denota a função objetivo. A forma apresentada é a que será utilizada no presente trabalho, na qual fixa-se um valor de risco máximo desejado σ_m e maximiza-se os retornos esperados através de m ativos no período Δt.

$$FO: \ max(\sum_{i=1}^{m} r_{i,\Delta t} w_{i,\Delta t}) \tag{3.8}$$

$$\begin{aligned} s.a. \ \ & w^T \cdot \Sigma \cdot w && \leq \sigma_m^2 \\ & \sum_{i=1}^{m} w_{i,\Delta t} && \leq 1 \\ & w_{i,\Delta t} && \geq 0 \end{aligned} \tag{3.9}$$

4.0 Heterocedasticidade condicional auto-regressiva

4.1 Variâncias não constantes

Toma-se um modelo de regressão linear simples, em que $\hat{y}_i = \beta_0 + \beta_1 \cdot x_i$, sendo $\varepsilon_i = y_i - \hat{y}_i$ o erro do modelo. Para que tal modelo represente bem a variável dependente, o erro deve seguir alguns princípios:

1. Os erros devem ser independentes para quaisquer pares ε_i e ε_j, com $i \neq j$
2. Os erros devem ser distribuídos através de uma curva normal com média zero e desvio padrão σ
3. O desvio padrão de ε deve ser constante para todos os valores das variáveis explicatórias

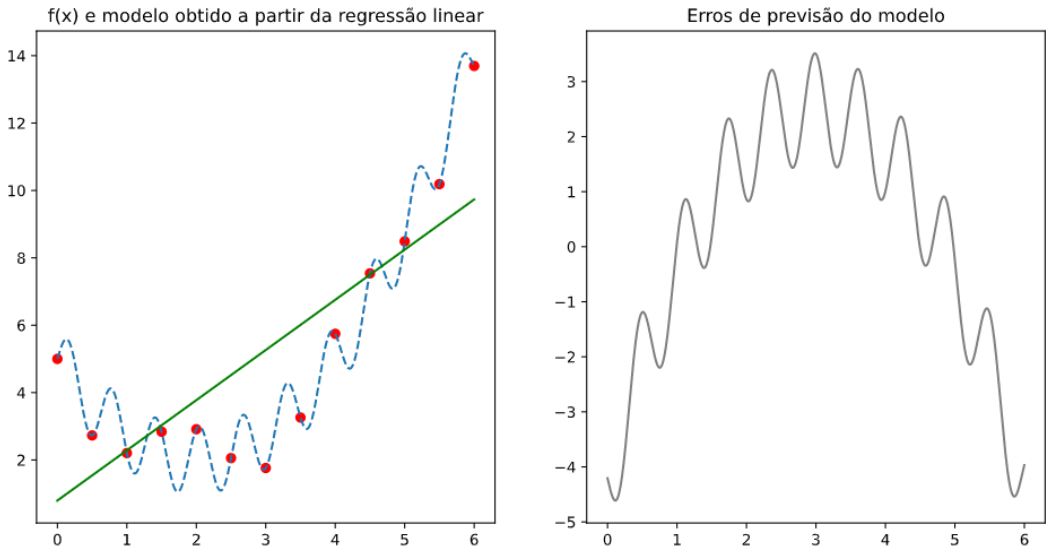
Este último princípio é o que define a homocedasticidade. Quando a variância dos termos de erro mudam concomitantemente com alterações nos valores das variáveis explicatórias, essa condição se torna de heterocedasticidade. Toma-se como exemplo a função não linear da equação 4.1. Os valores da função foram tomados nos pontos $x \in \{0, 0.5, 1.0, ..., 5.5, 6.0\}$ e, através do método dos Mínimos Quadrados, chegou-se no modelo representado pela equação 4.2.

$$f(x) = \frac{3x^2}{4} - 3x + 5 + sen(10x) \tag{4.1}$$

$$\hat{y}(x) = 0,79 + 1,49x \tag{4.2}$$

É possível observar na Figura 13 que a variância dos erros não é homogênea, configurando a condição como de heterocedasticidade. Isto é comum na volatilidade dos preços de ativos financeiros, nos quais períodos de alta volatilidade tendem a serem seguidos por períodos de alta volatilidade, bem como períodos de baixa volatilidade tendem a serem seguidos por períodos de baixa volatilidade (MANDELBROT, 1967). Este fenômeno é chamado de agrupamento de volatilidade. Pode ser observado, por exemplo, na Figura 14, representando a variação percentual dos preços diários da ação COCE5, ação da Companhia Energética do Ceará. Há uma tendência

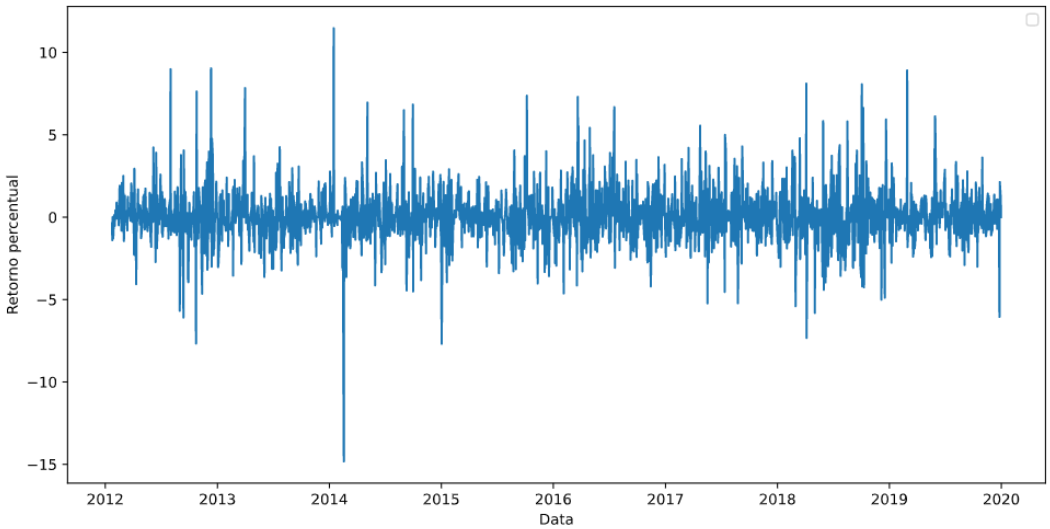
Figura 13: Exemplo de uma regressão linear aplicada a uma função não linear



Fonte: elaborado pelo autor

dentre os ativos financeiros a haver alguma forma de correlação entre a variância dos preços em um período e a variância dos preços no período seguinte.

Figura 14: Retornos diários da ação COCE5



Fonte: adaptado de Yahoo! Finance (2021)

4.2 Funções de autocorrelação

Uma das formas de se averiguar a independência entre duas variáveis aleatórias x_1 e x_2 é através do coeficiente de correlação linear de Pearson, definida pela equação 4.3 para um

período Δt.

$$corr(x_1, x_2)_{\Delta t} = \frac{cov(x_1, x_2)_{\Delta t}}{\sigma_{1,\Delta t} \cdot \sigma_{2,\Delta t}} \tag{4.3}$$

Este coeficiente está por definição no intervalo $[-1, 1]$, -1 representando uma forte correlação linear negativa, +1 representando uma forte correlação linear positiva e 0 representando nenhuma correlação linear.

Assim, define-se a função de autocorrelação (FAC) K de uma variável temporal X_t como sendo a correlação entre a variável X_t e ela mesma defasada de um certo período, $X_{t+\tau}$ (GUBNER, 2006). Assim sendo, define-se então a equação 4.4.

$$K_{XX}(t, t+\tau)_{\Delta t} = \frac{cov(X_t, X_{t+\tau})_{\Delta t}}{\sigma_{X,\Delta t} \cdot \sigma_{X,\Delta t+\tau}} \tag{4.4}$$

No entanto, o coeficiente de Pearson pode levar a uma conclusão de forte correlação entre duas variáveis quando de fato estas são correlacionadas a uma terceira variável, e não entre si. Dessa forma, define-se a função de autocorrelação parcial (FACP) de ordem (defasagem) 2 na equação 4.5.

$$\beta_X(2) = \frac{cov((X_t|X_{t-1}), (X_{t-2}|X_{t-1}))}{\sigma_{t|X_{t-1}} \cdot \sigma_{t-2|X_{t-1}}} \tag{4.5}$$

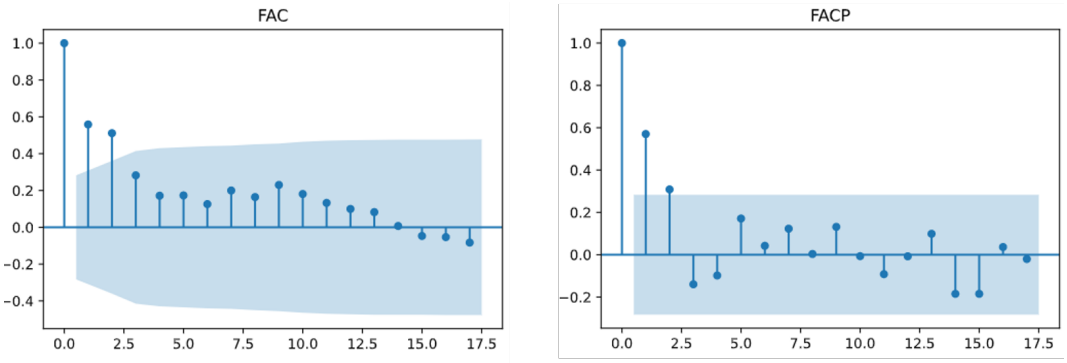
No caso da equação 4.5, $X_t|X_{t-1}$ é a série de resíduos após a criação de um modelo de regressão linear para explicar X_t em função de X_{t-1}, e $X_{t-2}|X_{t-1}$ é a série de resíduos após a criação de uma regressão linear para explicar X_{t-2} em função de X_{t-1}. Uma visualização intuitiva da autoregressão parcial é a tentativa de se usar X_{t-2} para explicar a variância de Xt que não é explicada pela variância de X_{t-1}, procurando entender os efeitos de X_{t-2} sobre X_t sem que a correlação entre X_{t-1} e X_{t-2} seja levada em consideração.

Define-se ainda a FACP de ordem k de acordo com a equação 4.6

$$\beta_X(k) = \frac{cov((X_t|X_{t-1}, X_{t-2}, ..., X_{t-k+1}), (X_{t-k}|X_{t-1}, X_{t-2}, ..., X_{t-k+1}))}{\sigma_{t|X_{t-1}, X_{t-2}, ..., X_{t-k+1}} \cdot \sigma_{t-k|X_{t-1}, X_{t-2}, ..., X_{t-k+1}}} \tag{4.6}$$

Como exemplo de visualização, tais funções foram aplicadas nas variâncias de períodos de 20 dias nos retornos da ação VALE5 (representativa da empresa Vale S.A.), entre os períodos de janeiro de 2012 a dezembro de 2019, observável na Figura 15. No caso, foram representados os valores de tais funções quando $p \in \{0, 1, 2, ..., 17\}$.

Figura 15: Funções FAC e FACP aplicadas nas variâncias dos retornos de VALE5



Fonte: elaborado pelo autor

4.3 Modelos ARCH e GARCH

Seja um modelo autoregressivo linear (AR) de ordem p – para modelar a variável X – definido pela equação 4.7. Através do método dos Mínimos Quadrados, minimiza-se o termo quadrático de erro ε.

$$X_t = \alpha_0 + \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + \ldots + \alpha_p X_{t-p} + \varepsilon_t = \alpha_0 + (\sum_{j=1}^{p} \alpha_j X_{t-j}) + \varepsilon_t \tag{4.7}$$

Engle (1982) propôs um modelo que chamou de ARCH (”Heterocedasticidade Condicional Autoregressiva”, do inglês ”Autoregressive Conditional Heteroskedasdicity”), com o objetivo de explicar a mudança dos resíduos de um modelo AR em função do tempo (em outras palavras, a heterocedasticidade). O autor separa a variância de um modelo em dois tipos: a condicional (que varia no tempo e depende dos erros anteriores) e a incondicional. O autor define a variância condicional por meio dos resíduos conforme a equação 4.8, em que ε_t são os resíduos encontrados no modelo AR.

$$\varepsilon_t(p) = \alpha_0 + (\sum_{j=1}^{p} \alpha_j \varepsilon_{t-j}) \tag{4.8}$$

Assim, o autor propõe a criação de modelos de previsão de variâncias de uma variável X no tempo de acordo com dois passos:

1. Encontrar o melhor modelo AR para representar a variável X

$$X_t(p) = \hat{\alpha_0} + (\sum_{j=1}^{p} \hat{\alpha_j} X_{t-j}) + \varepsilon_t$$

2. Encontrar os parâmetros α do modelo ARCH com base no modelo AR que minimizem o

erro da previsão de resíduos

$$\varepsilon_t(p) = \alpha_0 + (\sum_{j=1}^{p} \alpha_j \varepsilon_{t-j})$$

Pela proposição de tal modelo, aprofundamento de suas utilizações e suas estimações de eficiência/eficácia, Robert Engle recebeu o prêmio Nobel de Ciências Econômicas em 2003. Além disso, Bollerslev, Chou e Kroner (1992) notaram centenas de trabalhos acadêmicos que utilizaram modelos ARCH para previsão de volatilidades em diferentes mercados financeiros, obtendo previsões suficientemente precisas em diversos casos.

Bollerslev (1986) propôs uma generalização do modelo ARCH, que foi então denominado GARCH. De maneira análoga ao da equação 4.8, o autor tentou explicar a variância em t de acordo não somente com os p erros anteriores do modelo AR mas também das q variâncias anteriores da variável independente. Assim, a generalização pode ser vista na equação 4.9. O autor mostrou também que tal modelo é melhor adequado para previsão de variâncias de variáveis que apresentam o agrupamento de volatilidades.

$$\sigma_t(p,q) = \Omega + (\sum_{j=1}^{p} \alpha_j \varepsilon_{t-j}) + (\sum_{j=1}^{q} \beta_j \sigma_{t-j}) \tag{4.9}$$

Como estes modelos são usados para explicarem variâncias de um período com base em variâncias e resíduos anteriores, utiliza-se a FACP para se determinar quantas amostras anteriores devem serem levadas em consideração para a construção de um modelo ARCH/GARCH.

5.0 Indicadores técnicos

Indicadores técnicos, no contexto financeiro, são métricas matemáticas usando valores do passado com o objetivo de explicar o movimento do preço de uma ação. Neely et al. (2011) mostraram que indicadores técnicos têm um alto poder preditivo do movimento dos preços de ações, e, portanto, serão usados como variáveis explicatórias no modelo criado pelas redes neurais que farão as previsões dos retornos.

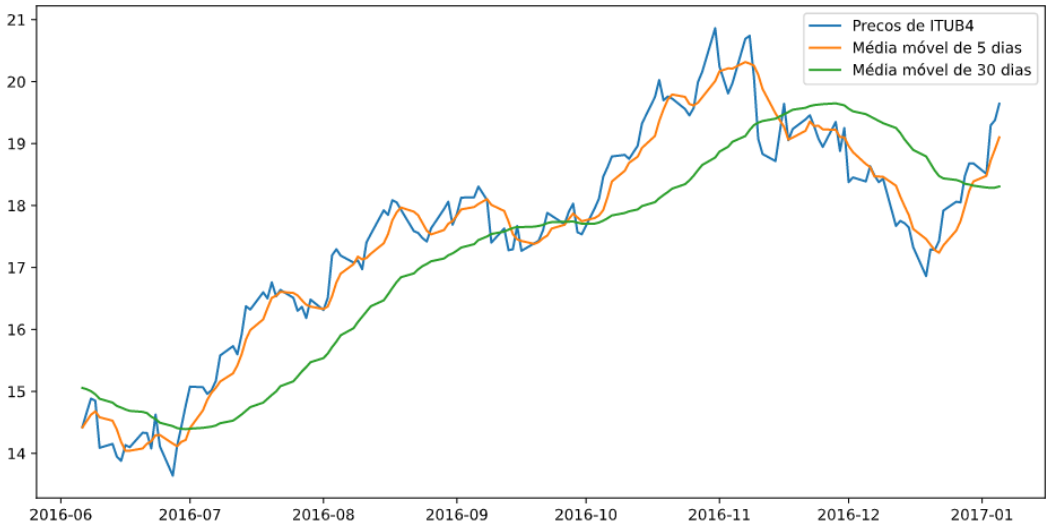
5.1 Média móvel

A média móvel consiste em calcular, em cada momento (t), a média do preço dos ativos nos k períodos anteriores. Uma técnica amplamente utilizada para prever tendências é a análise do cruzamento de médias móveis do preço de um ativo, uma curta – k_1 – e outra longa – k_2. Quando o valor da média móvel curta atravessa o valor da longa, pode-se interpretar tal movimentação como o valor do preço esteve recentemente se tornando maior do que o valor médio a longo prazo, indicando uma tendência de subida nos preços. O oposto pode ser interpretado quando o valor de k_1 se torna menor que o de k_2. Tal capacidade preditiva justifica a entrada dos valores de médias móveis de diferentes durações como variáveis explicatórias do modelo desenvolvido pelas redes neurais neste trabalho.

Para cada momento t, o valor da média móvel m dos preços p de k períodos é definida pela equação 5.1.

$$m_{k,t} = \frac{1}{k} \cdot \sum_{i=0}^{k-1} p_{t-i} \tag{5.1}$$

Figura 16: Preços diários de ITUB4 com suas médias móveis de 5 e 30 dias



Fonte: adaptado de Yahoo! Finance (2021)

5.2 EWMA

O EWMA (”Média Móvel Ponderada Exponencialmente”, do inglês ”Exponentially Weighted Moving Average”) é um indicador semelhante ao das médias móveis, na qual os pesos de valores mais antigos decrescem numa medida exponencial. Isto significa que a razão entre os pesos w_i de valores adjacentes temporalmente é igual para todos os termos. Primeiro, decide-se o fator de decrescimento dos pesos, chamado de α. Então, pode se calcular o valor da média móvel exponencial dos preços p no momento t de um período Δt de n dias de acordo com equação 5.2.

$$EWMA_{k,t}(\alpha) = \alpha \cdot p_t + \sum_{i=1}^{k-1} (1-\alpha)^i \cdot p_{t-i} \tag{5.2}$$

É possível observar que valores maiores de fatores de decrescimento α levam a maiores pesos para preços mais recentes. Dessa forma, diferentes valores de $alpha$ foram utilizados para que a própria rede pudesse aprender quais levar em consideração com maior importância para a previsão de retornos futuros. Os valores utilizados foram 25%, 50% e 75%.

5.3 MACD

O MACD (”Convergência-divergência de médias móveis”, do inglês ”Moving average convergence divergence”) foi desenvolvido em 1970 por Gerald Appel para identificar tendências de subida ou descida dos preços de um ativo. O MACD é composto pela diferença das médias

exponenciais de 26 dias e de 12 dias – diferença denotada por ”valor MACD” – bem como pela média exponencial de um período mais curto (como 9 dias) desta difererença. Durante momentos de tendência de movimentação dos preços, nos quais aumentam ou descrescem rapidamente, o ”valor MACD” tende a se afastar da média exponencial de tal valor, o que configura uma divergência. Assim, o indicador identifica momentos de tendência com a convergência e divergência de tais médias móveis.

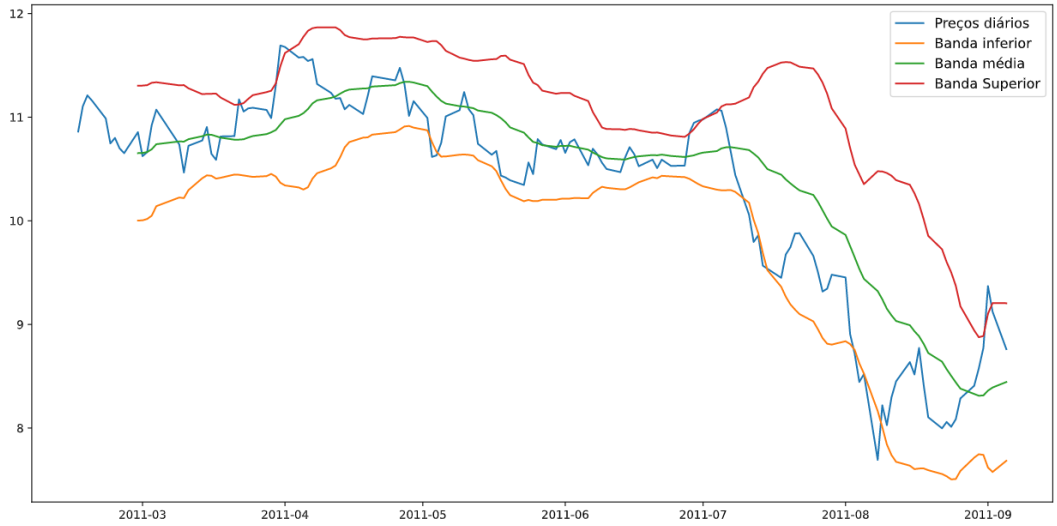
Colby e Meyers (1988) demonstrou que, comprando-se ativos atrelados ao índice Dow Jones (composto pelos preços dos 30 ativos de maior relevância no mercado acionário norte-americano) quando a linha MACD estivesse acima da média exponencial de período curto, e vendendo-os quando este chegasse a ser maior do que aquele, o investidor possuiria retornos positivos ao longo de décadas na bolsa de valores americana. Dessa forma, seus componentes apresentam-se como preditivas do movimento dos preços das ações e, portanto, serão utilizadas como variáveis explicatórias da rede neural deste trabalho.

5.4 Bandas de Bollinger

As Bandas de Bollinger são indicadores que compõem uma técnica desenvolvida na década de 1980 por John Bollinger. A técnica é composta por três indicadores: uma média móvel de k dias (a banda média), a média móvel de k dias somado n desvios padrões referente ao período de k dias (a banda superior) e, por último, a média móvel subtraindo-se n desvios padrões (a banda inferior). Bollinger demonstrou que os preços de vários ativos – principalmente para commodities como o milho – tendem a reverter a um valor médio. Assim, quando o preço estivesse mais próximo do terceiro indicador, haveriam evidências de que o preço do ativo voltaria a subir nos períodos seguintes para retornar a seu valor médio.

Colby e Meyers (1988) notou que as bandas de Bollinger são versáteis, sendo úteis para vários mercados diferentes e reagindo rapidamente a mudanças rápidas de preços. Observou também que a alteração dos parâmetros k e n fornecem previsões realistas de tendências de diferentes durações. No entanto, concluiu que as bandas tendem a não fornecer bons sinais de compra e venda quando utilizadas como indicador único, indicando que devem ser usadas como suporte a indicadores de causalidade. Em outras palavras, no evento do preço de um ativo atingir a banda superior, outras métricas devem também serem analisadas para se ter uma previsão certeira do movimento futuro do preço; na circunstância de haverem poucos motivos para o preço estar historicamente elevado, se torna provável que o preço cairá novamente, seguindo as bandas.

Figura 17: Bandas de Bollinger aplicadas aos preços diários de ITUB4 ao longo de 140 dias úteis, em que t=20 e n=2



Fonte: adaptado de Yahoo! Finance (2021)

6.0 Ativos Estudados

6.1 Os ativos do portfólio

Neste trabalho, múltiplos ativos terão seus retornos, variâncias e covariâncias estimadas para um período de 20 dias. Faz-se necessário, então, formalizar quais ativos terão tais valores estimados para futura formação de um portfólio composto exclusivamente destes.

Como constatado previamente neste trabalho, o mercado de capitais se dá de forma centralizada no Brasil através da bolsa de valores B3. Nem todos os ativos negociados nesta são de alta liquidez, havendo ativos que não possuem uma negociação sequer ao longo de diversos dias. Assim, a lei da oferta e da demanda indicaria que esses ativos seriam facilmente sujeitos a alterações nos preços por negociações únicas feitas num dia (MANKIW, 2005). Para se fazer uma análise dos retornos dos ativos, foi adotada a hipótese de que as operações indicadas pelo presente trabalho não alterariam os valores dos preços dos ativos.

Para mitigar as mudanças de preços devido a compras/vendas pontuais, o espaço amostral de ações analisadas foi selecionado a partir do conjunto de ações com maior volume de negociação financeira na bolsa de valores. Assim, assegura-se que um investidor individual ou institucional de pequeno a médio porte teria pouco efeito sobre o preço de um ativo com suas operações.

Portanto, os ativos selecionados para a análise em questão são aqueles que compunham o índice IBRX50 no início do período de análise (1º de janeiro de 2012), uma carteira teórica criada pela B3 composta pelos 50 ativos de maior volume diário de negociação por trimestre. A informação de quais ativos compunham tal grupo em determinada data foi extraída a partir da plataforma Bloomberg Terminal.

Por falta de disponibilidade de alguns preços de alguns ativos na plataforma utilizada para captá-los (Yahoo! Finance, em 2021), apenas 47 ativos foram pré selecionados dentro desta carteira. Todos os ativos analisados podem ser observados na Tabela 1.

Tabela 1: Ativos pré selecionados para análise de retornos e variâncias

PETR4	PETR3	VALE5	VALE3
ITUB4	ABEV3	BBDC4	CSNA3
ITSA4	GGBR4	CMIG4	USIM5
BRKM5	BBDC3	ELET3	GOAU4
ELET6	KLBN4	BRAP4	OIBR4
SBSP3	BBAS3	CCRO3	LAME4
EMBR3	UNIP6	VIVT3	CPLE6
CTNM4	TRPL4	EGIE3	CMIG3
CGAS5	RAPT4	TIMS3	CLSC4
POMO4	CESP5	ETER3	COCE5
CPLE3	FESA4	BOBR4	SAPR4
TNCP4	TASA4	INEP	

Fonte: elaborado pelo autor

6.2 Ativos correlacionados

Os ativos são muitas vezes correlacionados – em diferentes graus – a outros ativos, ou a situações econômicas. Seus preços podem ser descritos como uma função de seus próprios desempenhos mas também da situação econômica, tanto global, nacional ou industrial. Um exemplo seria o desempenho dos retornos de ações a eventos macroeconômicos mundiais. Uma crise mundial pode levar ao decréscimo de valores de ações, como pode ser visto pela Figura 18, na qual a ação do banco Goldman Sachs na bolsa de Nova York (NYSE, *New York Stock Exchange*) aparenta ter caído como consequência da crise financeira de 2008.

Figura 18: Preços diários das ações do banco Goldman Sachs

Fonte: adaptado de Yahoo! Finance (2021)

Dessa forma, para a previsão de retornos, a rede neural recebeu como entradas os valores de ativos correlacionados aos que queremos prever os retornos, o preço das ações de bancos internacionais em bolsas americanas de valores, de empresas listadas na bolsa mas que não compunham o índice IBRX50, dentre outros.

Referente a situações macroeconômicas, foram captados os valores diários de moedas em relação ao dólar. Uma queda no valor da moeda nacional pode indicar um fraco desempenho da economia do país, o que levaria a um decréscimo amplo nos preços da maioria dos ativos negociados na bolsa de valores.

Além disso, a taxa de juros pode indicar também o movimento do mercado. Quando tal taxa é elevada, os investidores possuem menores incentivos para tomarem riscos maiores no mercado acionário para obterem retornos que podem pouco ultrapassarem – ou não ultrapassarem – os retornos de investimentos em títulos públicos de renda fixa, que representam menor risco. Assim, o preço de ações e a taxa básica de juros de um país tendem a se movimentar em direções opostas.

É importante notar que não só a taxa de juros é importante mas também a percepção do mercado de seu movimento futuro. Quando o mercado espera um aumento, espera também – por consequência – uma diminuição generalizada no valor das ações. No caso, previsões futuras de preços refletem em preços presentes, já que assume-se que o investidor racional não deseja possuir ativos que espera desvalorizarem. Dessa forma, não só a taxa básica de juros vigente foi utilizada como variável explicatória na rede neural mas também a percepção de mercado de qual seria a taxa média no futuro.

Isso pôde ser verificado através dos contratos futuros de DI (Depósito Interbancário). Tais contratos remuneram seus proprietários no valor de R$100 mil e, assim, os participantes do mercado sobre este ativo negociam estes contratos por um valor que acreditam que seja o valor destes R$100 mil trazidos a valor presente de acordo com a taxa básica de juros brasileira, a taxa DI. O valor de uma quantia monetária futura pode ser trazida a valor presente de acordo com a equação 6.1, sendo NPV o valor presente, i a taxa média de juros anuais, FV o valor futuro e t o número de dias úteis entre o período presente e o período de recebimento do valor. No caso da taxa brasileira, esta é acruada por dia útil, e foi considerado que o ano possui 252 dias úteis – como a própria instituição B3 o considera. No caso, o preço de negociação do contrato é o NPV, e o valor futuro FV equivale aos R$100 mil recebidos na data de vencimento.

$$NPV = \frac{FV}{(1+i)^{(\frac{t}{252})}} \tag{6.1}$$

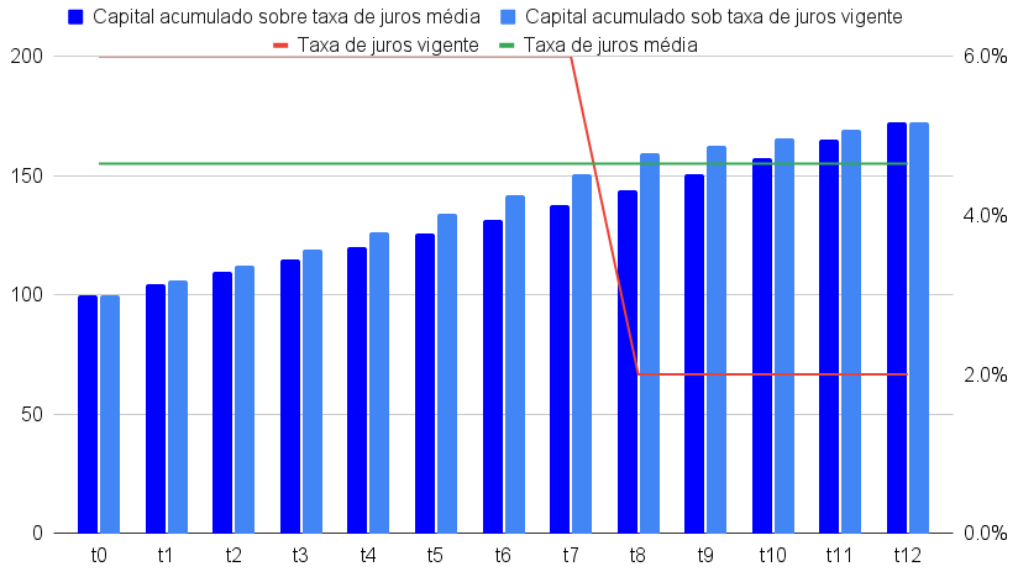
Reescrevendo-se a equação 6.1, pode-se obter a taxa de juros i média em função das outras

variáveis, visto na equação 6.2.

$$i = (\frac{FV}{NPV})^{(-\frac{t}{252})} - 1 \tag{6.2}$$

Uma visualização da taxa média pode ser obtida através da Figura 19, na qual o acruo de capital através da taxa de juros vigente de cada período leva a um valor final, que pode então ter sua taxa média calculada de acordo com a equação 6.2.

Figura 19: Representação do cálculo de uma taxa média de juros com base em períodos de diferentes taxas de juros



Fonte: elaborado pelo autor

Estes contratos têm como data de vencimento o primeiro dia útil do mês para o qual está sendo negociado. Logo, se uma instituição adquire um contrato deste tipo para o mês de julho de 2030, irá pagar no momento presente o valor negociado e receber o valor de R$100 mil no primeiro dia útil do mês de julho de 2030. Como os preços diários de contratos com diferentes datas de vencimento são divulgadas pela B3, se fez possível obter a expectativa consolidada do mercado sobre a taxa de juros acumulada até a data de vencimento de cada um destes contratos disponíveis, de acordo com a equação 6.2.

Como exemplo, alguns dos valores podem ser observados na Tabela 2, obtidos a partir do site da B3. No caso da rede neural, objetiva-se explicar o valor de variáveis independentes a partir de dependentes, sendo que as mesmas variáveis devem ser usadas em todas as amostras – de maneira análoga a uma regressão linear multivariada. Assim, não seria possível usar a taxa média de juros prevista pelo mercado para um contrato a Δt dias do vencimento e, no dia seguinte, usar como valor da mesma variável o valor de tal taxa média para um contrato a $\Delta t - -1$ dias para vencer, já que configuram variáveis diferentes e em diferentes datas. Assim,

fez-se necessário o uso de interpolações lineares.

Tabela 2: Taxa média de juros de contratos futuros de DI no dia 15 de junho de 2015 para certas datas de vencimento

Dias até o vencimento do contrato	Taxa média de juros prevista pelo mercado
78	13,82%
107	14,01%
137	14,15%
198	14,31%

Fonte: adaptado de B3

Para uma função $f(x)$ com valores conhecidos em $f(x_1)$ e $f(x_2)$, define-se então uma interpolação linear $f^*(x^*)$ de dois pontos $(x_1, f(x_1))$ e $(x_2, f(x_2))$, com $x_1 < x^* < x_2$ como a média dos valores de $f(x_i)$ ponderados pela distância de x^* a cada um dos pontos x_i. Sua formulação matemática pode ser vista de duas formas nas equações 6.3 e 6.4.

$$f^*(x^*) = \frac{(f(x_2) \cdot (x_2 - x^*)) + (f(x_1) * (x^* - x_1))}{x_2 - x_1} \tag{6.3}$$

$$f^*(x^*) = f(x_1) + \frac{f(x_2) - f(x_1)}{x_2 - x_1} \cdot (x^* - x_1) \tag{6.4}$$

Após a obtenção dos valores diários das expectativas das taxas de juros médias para diferentes datas de vencimento, foram feitas interpolações lineares de acordo com a equação 6.4 para se obter a previsão do mercado para a taxa de juros média para contratos que venceriam em 1, 2, ..., 3000 dias. Assim, foram obtidos valores diários para todas essas variáveis, que puderam então serem inseridas como variáveis de entrada na rede neural deste trabalho.

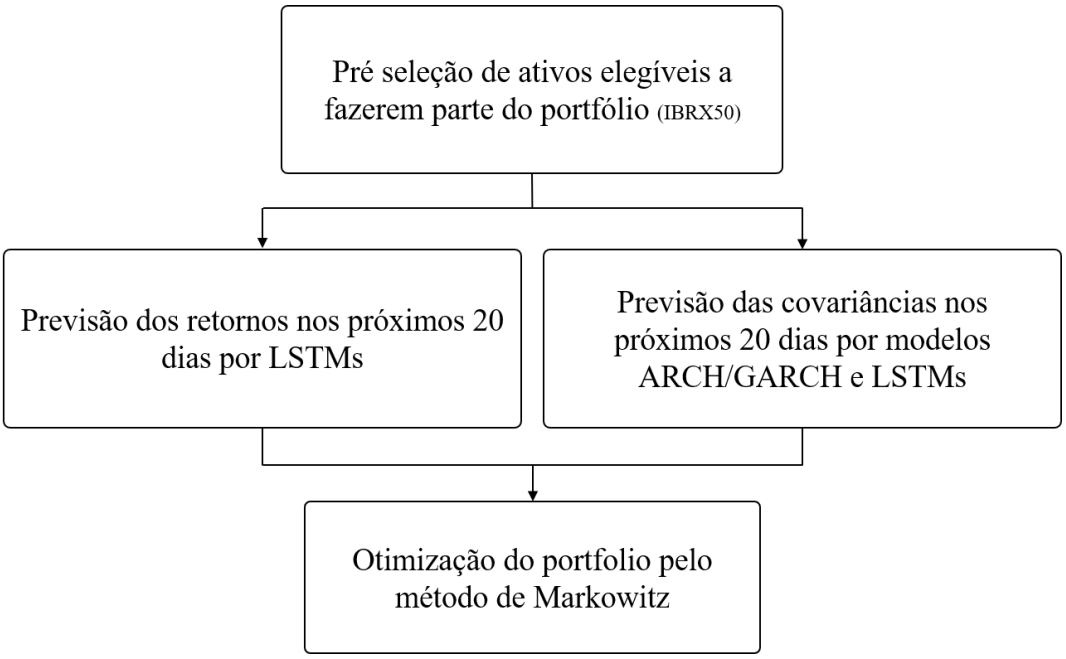
Ademais, fazendo o Brasil parte de uma economia globalizada, acontecimentos internacionais têm muitas vezes efeitos sobre os preços locais de diversos ativos. Portanto, outros ativos correlacionados selecionados foram os de índices de bolsas nacional e internacionais, bem como os preços diários internacionais de empresas multinacionais que possuem operações no Brasil, como Goldman Sachs, JPMorgan Chase e outros. Além disso, como existe uma relação de causalidade em que o aumento de volatilidade nos mercados acionários norte-americanos levaram, historicamente, ao aumento de volatilidade do mercado acionário brasileiro (ALMEIDA, 2009), foi utilizada como variável explicatória também o índice de volatilidade da bolsa americana VIX (*Volatility Index*, índice feito com base em opções sobre empresasas americanas para representar o sentimento do mercado em relação à volatilidade dos ativos, sendo a opção um contrato que confere a seu proprietário o direito de compra/venda de um ativo por um determinado preço

e em determinadas datas no futuro). Tais preços e índices foram obtidos gratuitamente através da plataforma Yahoo! Finance (2021), e sua lista completa está indicada em apêndice.

7.0 Desenvolvimento

Como anunciado anteriormente neste trabalho, objetivou-se a criação de um portfólio otimizado por meio da fronteira eficiente de Markowitz, integrado a previsões de retornos e de matrizes de covariâncias por redes neurais do tipo LSTM. Essa abordagem pode ser observada no fluxograma da Figura 20. Para fins de simplificação, foi adotada a pressuposição de que os preços são discretos, tendo sido selecionados apenas os preços de fechamento p_t (em outras palavras, o último preço no qual o ativo foi negociado na bolsa de valores em um dado dia (t)), sendo considerado que todo (t) é um dia útil no qual há negociações na bolsa B3.

Figura 20: Metodologia de abordagem do presente trabalho



Fonte: elaborado pelo autor

Tendo os ativos elegíveis a fazerem parte do portfólio determinados na seção 6.1, o próximo passo foi a criação de modelos de previsão de retornos e covariâncias.

Um modelo de previsão pode ser representado através da notação $Y = f(X) + \varepsilon$, no qual f representa a função de previsão, Y as variáveis que deseja-se prever (em um primeiro momento,

os retornos dos ativos e, em seguida, as matrizes de covariâncias), X as variáveis explicatórias e $\hat{Y}$ a previsão obtida. Conforme descrito na seção 2.5, faz-se necessário para a criação de tal modelo, primeiramente, a obtenção de uma amostra sobre a qual o modelo aprenderá, outra sobre a qual haverá validação e outra com a qual será feita a verificação de sua capacidade de generalização. Dessa forma, os dados utilizados para previsão foram separadas de acordo com suas datas, como segue:

- 20 de janeiro de 2012 a 28 de fevereiro de 2018 – 20% das datas foram selecionadas aleatoriamente para comporem a amostra de validação e as remanescentes foram selecionadas para comporem a amostra de treino
- 1 de março de 2018 a 26 de dezembro de 2019 – amostra de teste/verificação de capacidade de generalização

Como descrito anteriormente no trabalho, as redes LSTMs se utilizam de sequências e, portanto, sequências contendo dados de $((t-n_d),\ (t-n_d+1),\ ...,\ (t))$ foram criadas para cada dia de análise (t), em que n_d=40 corresponde ao número de dias anteriores observados. Seja m o número de variáveis explicatórias observadas nos n_d dias anteriores em cada dia (t) para previsão dos retornos/matrizes de covariâncias de n_A ativos entre (t) e $(t+20)$.

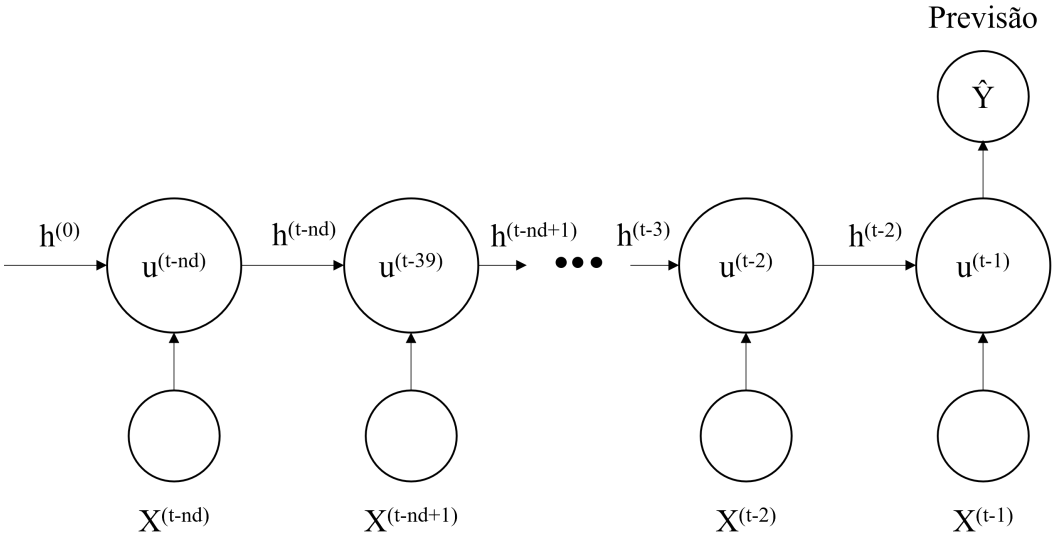
7.1 Previsão de retornos

Para cada dia t na amostra de treino, foram criadas matrizes $R^{m\ x\ n_A}$ com os valores das variáveis explicatórias de $(t-n_d)$ a (t), e, além disso, o valor do retorno no período Δt de (t) a $(t+20)$ – a variável que deseja-se prever – foi adicionado ao vetor Y, que é utilizado para treinar a rede. A representação de tal rede pode ser vista na Figura 21.

Quanto às variáveis explicatórias, foram utilizadas:

- Preços diários de todos os ativos, considerando-se tanto os elegíveis a fazerem parte do portfólio quanto os correlacionados, conforme descrito no capítulo 6;
- valores diários de índices de bolsas de valores, conforme explicado na seção 3.1;
- retornos de $(t-20)$ a (t) dos ativos elegíveis a fazerem parte do portfólio;
- retornos de $(t-1)$ a (t) dos ativos elegíveis a fazerem parte do portfólio;
- indicadores técnicos de cada um dos ativos elegíveis a fazerem parte do portfólio, conforme explicado no capítulo 5;

Figura 21: Representação do modelo de previsão de retornos por LSTMs



Fonte: elaborado pelo autor

- taxa básica de juros anual (Selic) vigente em (t), conforme explicado na seção 3.1;
- expectativa de mercado para os valores da taxa básica média de juros anual para $(t+30)$, $(t+90)$, $(t+180)$, $(t+360)$, $(t+720)$, $(t+1800)$ e $(t+3000)$, conforme explicado na seção 6.2 (obs.: para esta variável, as defasagens referem-se a dias absolutos, diferentemente das outras, em que a defasagem de dias é calculada em dias úteis);
- valores de câmbio entre diversas moedas, conforme explicado na seção 6.2;
- variações percentuais de $(t-1)$ a (t) dos valores de câmbio.

A função de erro $J(\theta)$ utilizada foi o EQM (equação 2.3), sendo priorizada a função dos erros quadráticos no lugar dos erros absolutos com o intuito de se aumentar a penalidade no modelo de erros advindos de *outliers* ("Um outlier é uma observação que se diferencia tanto das demais observações que levanta suspeitas de que aquela observação foi gerada por um mecanismo distinto" (HAWKINS, 1980)) como forma de identificar ativos cujos retornos de 20 dias estão crescendo de forma discrepante aos restantes.

Os hiperparâmetros – os parâmetros que não são modificados pelo treino da rede, como número de nós, número de camadas, etc. – foram testados iterativamente a fim de se reduzir o erro de validação. Como explicado na seção 2.3, é necessária a definição de uma taxa de aprendizado para a execução do gradiente descendente. Foi utilizado o valor inicial de $0,0001$, sendo reduzido pela metade quando 10 ciclos de treino consecutivos mostrassem redução de erro de validação menor que 0,01%. Foi utilizada apenas uma camada LSTM, de 145 neurônios

nos estados oculto e de célula. Os coeficientes de regularização foram de $\lambda_1 = 0{,}010$ – correspondente ao coeficiente de regularização dos pesos das interconexões da rede – e $\lambda_2 = 0{,}001$ – correspondente ao coeficiente de regularização das saídas dos neurônios.

7.2 Previsão de matrizes de covariâncias

A previsão das matrizes de covariâncias foi realizada através de uma integração de modelos ARCH, GARCH e LSTMs, conforme proposto por Kim e Won (2018). Objetivou-se criar um modelo do tipo $Y = f(X) + \varepsilon$, no qual Y representa a matriz de covariâncias dos ativos a_i do portfólio na data (t) no período de (t) a $(t+20)$. Entretanto, redes neurais retornam vetores e cada dimensão está associada a uma variável independente, e não matrizes. Portanto, para cada dia (t), as matrizes de covariâncias foram vetorizadas, tornando

$$\begin{bmatrix} cov(a_1,a_1)_{\Delta t} & cov(a_1,a_2)_{\Delta t} & \cdots & cov(a_1,a_{n_A})_{\Delta t} \\ cov(a_2,a_1)_{\Delta t} & cov(a_2,a_2)_{\Delta t} & \cdots & cov(a_2,a_{n_A})_{\Delta t} \\ \vdots & & \ddots & \vdots \\ cov(a_m,a_1)_{\Delta t} & cov(a_m,a_2)_{\Delta t} & \cdots & cov(a_{n_A},a_{n_A})_{\Delta t} \end{bmatrix}$$

em

$$\begin{bmatrix} cov(a_1,a_1)_{\Delta t} & cov(a_1,a_2)_{\Delta t} & \cdots & cov(a_{n_A-1},a_{n_A})_{\Delta t} & cov(a_{n_A},a_{n_A})_{\Delta t} \end{bmatrix}$$

Assim, foram obtidos vetores de dimensão n_A^2 para que a rede previsse, sendo cada dimensão representativa de uma variável dependente.

Em seguida, fez-se necessário criar os modelos de ARCH e GARCH de acordo com as seguintes equações, respectivamente.

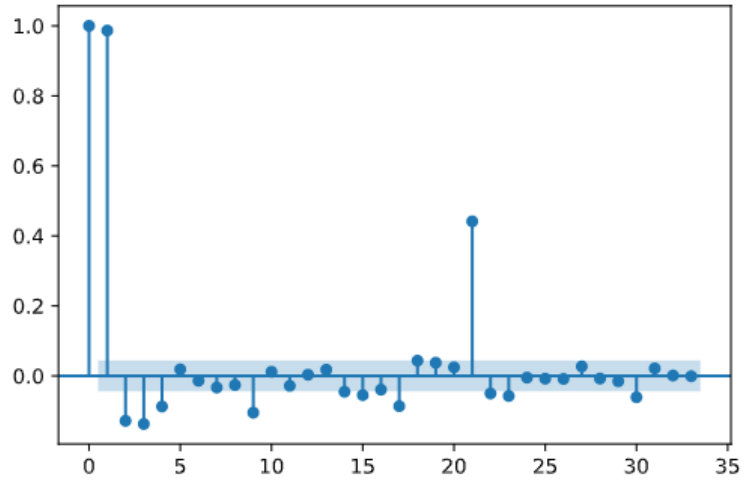
$$\sigma_t(p) = \alpha_0 + (\sum_{j=1}^{p} \alpha_j \varepsilon_{t-j})$$

$$\sigma_t(p,q) = \Omega + (\sum_{j=1}^{p} \alpha_j \varepsilon_{t-j}) + (\sum_{j=1}^{q} \beta_j \sigma_{t-j})$$

Os valores de p e q utilizados para a criação dos modelos foram definidos com base na utilização do PACF, que revelou forte correlação das variâncias em até 3 períodos anteriores para diversos ativos, como mostrado na Figura 22.

Como redes neurais são capazes de modelar relações de extrema complexidade, não foram selecionados apenas um valor para p e q. Foram criados os modelos de ARCH e GARCH para cada um dos ativos do portfólio utilizando-se $p,\ q \in \{1,\ 2,\ 3\}$, possível através da metodologia

Figura 22: FACP aplicado aos valores diários das variâncias de 20 dias dos retornos de PETR4



Fonte: elaborado pelo autor

discutida na seção 4.3.

Para cada dia (t) e para cada ativo do portfólio, foram utilizados 200 dias de amostra $((t-199),\ (t-198),\ ...,\ (t))$ para serem criados modelos autoregressivos de ordem $p \in \{1,\ 2,\ 3\}$ para preverem os retornos de cada ativo. Seus resíduos foram, então, utilizados para a criação dos ARCH/GARCH de ordens p e $q \in \{1,\ 2,\ 3\}$. Em seguida, os valores dos parâmetros de cada modelo de todo ativo, bem como suas previsões de variância em 20 dias, foram inseridos como variáveis explicatórias da rede neural utilizada para a previsão final.

Por fim, foram inseridas como variáveis explicatórias, também, os p-valores de cada variável de todo modelo (isto é, a probabilidade de que a amostra apresentaria tal comportamento dado que o coeficiente α/β desta variável é zero).

Dessa forma, as variáveis explicatórias utilizadas na rede neural para previsão de matrizes de covariâncias foram:

- Preços diários de todos os ativos, considerando-se tanto os elegíveis a fazerem parte do portfólio quanto os correlacionados, conforme descrito no capítulo 6;
- valores diários de índices de bolsas de valores, conforme explicado na seção 3.1;
- retornos de $(t-20)$ a (t) dos ativos elegíveis a fazerem parte do portfólio;
- retornos de $(t-1)$ a (t) dos ativos elegíveis a fazerem parte do portfólio;
- indicadores técnicos de cada um dos ativos elegíveis a fazerem parte do portfólio, conforme explicado no capítulo 5;

- taxa básica de juros anual (Selic) vigente em (t), conforme explicado na seção 3.1;
- expectativa de mercado para os valores da taxa básica média de juros anual para $(t+30)$, $(t+90)$, $(t+180)$, $(t+360)$, $(t+720)$, $(t+1800)$ e $(t+3000)$, conforme explicado na seção 6.2 (obs.: para esta variável, as defasagens referem-se a dias absolutos, diferentemente das outras, em que a defasagem de dias é calculada em dias úteis);
- valores de câmbio entre diversas moedas, conforme explicado na seção 6.2;
- variações percentuais de $(t-1)$ a (t) dos valores de câmbio;
- matrizes de covariâncias entre todos os ativos elegíveis a fazerem parte do portfólio no período de $(t-20)$ a (t);
- parâmetros dos modelos de ARCH/GARCH, bem como seus p-valores

Os hiperparâmetros foram testados iterativamente a fim de se reduzir o erro de validação. A taxa de aprendizado para a execução do gradiente descendente utilizada foi de $0,0001$, sendo reduzido pela metade quando 10 ciclos de treino consecutivos mostrassem redução de erro de validação menor que 0,01%. Diferentemente da rede concebida para a previsão de retornos, esta rede foi composta duas camadas LSTM, sendo seus hiperparâmetros observados na Tabela 3, em que λ_1 corresponde ao coeficiente de regularização dos pesos das interconexões da rede e λ_2 corresponde ao coeficiente de regularização das saídas dos neurônios.

Tabela 3: Valores dos hiperparâmetros da rede neural para previsão de retornos futuros

Camada	Número de neurônios dos estados oculto e de célula	λ_1	λ_2
1	218	0,010	0,001
2	114	0,00001	0,000

Fonte: elaborado pelo autor

7.3 Otimização do portfólio

Tendo por fim os retornos r de cada ativo i previstos no período Δt, de (t) e $(t+20)$, bem como suas covariâncias, o último passo foi a criação do portfólio. Para cada momento, os pesos w_i de cada ativo i no portfólio foram definidos de acordo com a maximização do retorno pelo modelo de otimização a seguir. É importante notar que foi inserida a restrição de que todos os pesos $w_{i,\Delta t}$ devem ser obrigatoriamente menores que $w_{max} = 0{,}4$, seguindo uma prática comum

nas empresas do setor: a diversificação obrigatória de ativos no fundo, estabelecendo um limite no qual pode-se investir em um mesmo ativo.

$$FO: \ max(\sum_{i=1}^{m} r_{i,\Delta t} w_{i,\Delta t})$$

$$\begin{aligned} s.a. \ \ w^T \cdot \Sigma \cdot w &\leq \sigma_m^2 \\ \sum_{i=1}^{m} w_{i,\Delta t} &\leq 1 \\ w_{i,\Delta t} &\geq 0 \\ w_{i,\Delta t} &\leq w_{max} \end{aligned}$$

A variância da rentabilidade do portfólio pode ser medida diariamente mas pode, também, ser medida anualmente, como é prática comum dentre as gestoras de fundos de investimento. A conversão entre volatilidade diária e anualizada se dá por meio da equação 7.1. Como diversas gestoras de fundos de investimento de alta volatilidade – dentre as quais, aquela na qual trabalhou o autor deste trabalho – adotam como objetivo atingirem uma volatilidade anual de seus retornos entre 12% e 16%, o valor de 14% foi utilizado como volatilidade anualizada máxima aceitável. Assim, o valor de variância máxima σ_m^2 adotado foi de $\frac{0,14^2}{252}$.

$$\sigma_{anualizada} = \sigma_{diario} \cdot \sqrt{252} \tag{7.1}$$

Por fim, a estratégia adotada foi aplicar a mencionada maximização no dia 1 de março de 2018 e, a cada 20 dias úteis, reaplicar a maximização, alterando o valor dos pesos w_i de cada ativo do portfólio. Os retornos foram calculados diariamente com base nas equações 3.1 e 3.2.

É importante notar que não há restrição de soma total dos pesos $w_i, \Delta t$ ser 1, tornando possível não investir inteiramente todos os recursos do portfólio. A parte não investida é considerada como a parte que possui retorno igual a zero. Dessa forma, em períodos de maior volatilidade no mercado, a maximização dos retornos pelo processo descrito pode levar a pequenos investimentos como forma de proteção a altas flutuações nos preços.

8.0 Resultados

8.1 Retorno da estratégia

O retorno total da estratégia proposta no presente trabalho foi de 70,94% no período de 1 de março de 2018 a 26 de dezembro de 2019 obtendo como volatilidade anualizada o valor de 18,16%, enquanto que a taxa Selic apresentou uma rentabilidade final de 11,54% no mesmo período. A comparação entre os retornos acumulados pode ser observada na Figura 23.

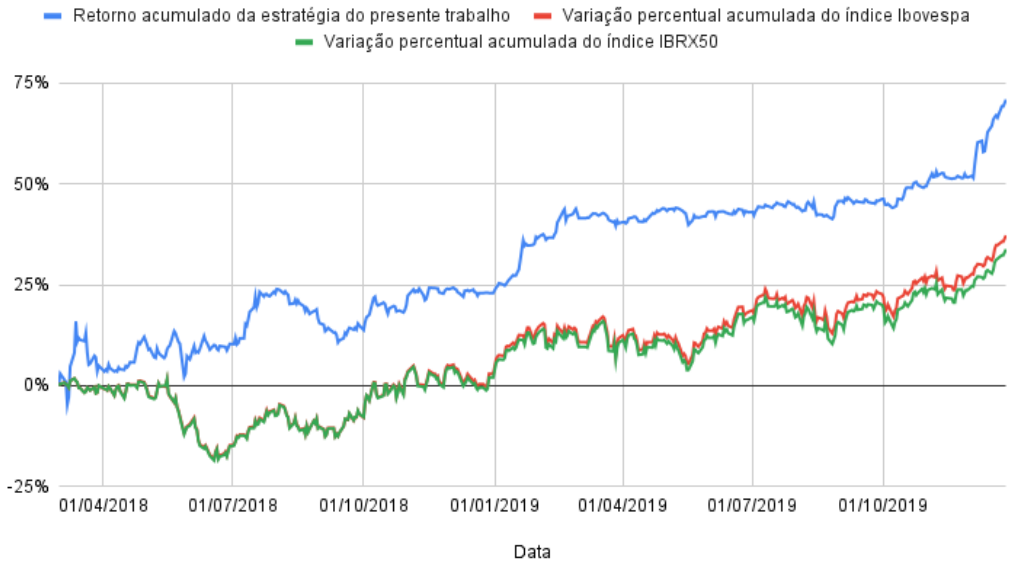
Figura 23: Retornos acumulados da estratégia resultante do presente trabalho e da taxa básica de juros brasileira no período avaliado

Fonte: elaborado pelo autor

Para verificar a eficiência da estratégia, foi realizada a comparação entre os retornos dessa estratégia com a do índice Ibovespa, já que serve de métrica para o desempenho da bolsa de valores agregada. Um retorno similar de ambos poderia indicar que o modelo proposto não está de fato estruturando um bom portfólio, mas sim que os retornos das ações da bolsa de maneira ampla estão demonstrando bons resultados. No entanto, é possível observar na Figura 24 que as variações percentuais acumuladas de tal índice atingiram 37,28%, com uma volatilidade anualizada de 19,94%, representando resultados inferiores com um risco maior. Além disso, a mesma

comparação foi efetuada com o índice IBRX50, dado que os ativos selecionados para comporem o portfólio deste trabalho faziam parte de tal índice. É importante notar que o IBRX50 tem sua composição reavaliada quadrimestralmente (nos dias 1 de janeiro, 1 de maio e 1 de setembro) e, sendo assim, os ativos do índice, ao longo do tempo, não são os mesmos dos ativos que foram selecionados no portfólio do modelo proposto.

Figura 24: Comparação entre os retornos da estratégia proposta e de dois dos principais índices da bolsa de valores brasileira

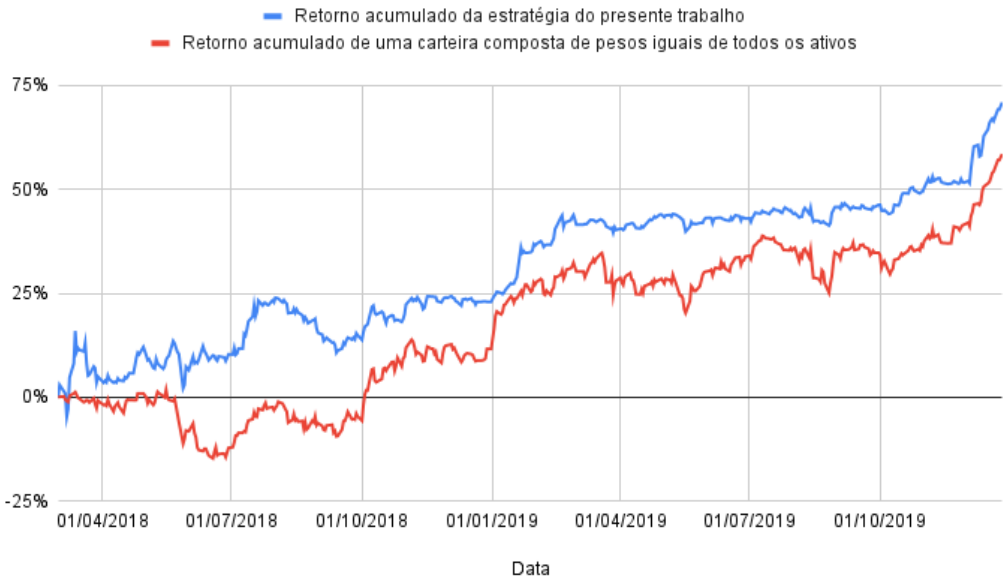


Fonte: elaborado pelo autor

Realizou-se também a comparação entre o retorno do modelo proposto com o retorno de uma carteira hipotética composta pelos mesmos 47 ativos em proporções iguais. Esta obteve um retorno de 58,51% sob uma volatilidade de 18,28%. Assim, o modelo proposto mostrou-se superior à uma estratégia de alocações iguais de recursos em todos os ativos.

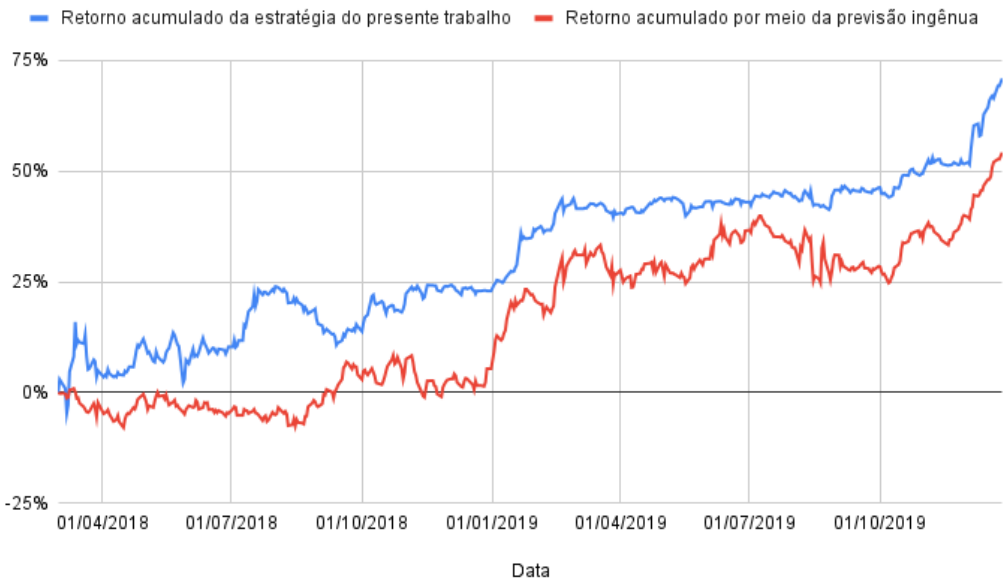
Por último, foi efetuada uma comparação entre o modelo proposto e um modelo de otimização do portfólio por meio da previsão *ingênua*, nome dado ao método que consiste em utilizar o último valor realizado como previsão para o futuro. Em outros termos, uma carteira hipotética com os mesmos ativos definidos no modelo proposto foi otimizada pela fronteira de eficiência em que seus valores de retornos esperados de (t) a $(t+20)$ foram definidos como os retornos realizados no período de $(t-20)$ a (t), bem como a matriz de covariâncias esperadas entre os ativos no período de (t) a $(t+20)$ foi definida como sendo a matriz de covariâncias no período de $(t-20)$ a (t). A estratégia que se utilizou da previsão ingênua apresentou um retorno total de 55,42%, possuindo uma volatilidade anualizada de 18,61%. Dessa forma, o uso de previsões por meio de redes neurais do tipo LSTM se mostrou superior à utilização da previsão ingênua.

Figura 25: Comparação entre os retornos da estratégia proposta e os retornos de uma carteira feita com os mesmos ativos em pesos iguais



Fonte: elaborado pelo autor

Figura 26: Comparação entre os retornos da estratégia proposta e os retornos de uma carteira utilizando a previsão ingênua



Fonte: elaborado pelo autor

9.0 Conclusão

A utilização de modelos de redes neurais do tipo LSTM para construção de uma estratégia quantitativa de investimentos baseados em ativos acionários de alto volume de transações no mercado brasileiro se mostra uma estratégia de alta rentabilidade. Uma pré seleção dos ativos foi efetuada, ativos correlacionados selecionados, modelos de previsão de retorno de 20 dias construído a partir de redes neurais do tipo LSTM, modelos de previsão de volatilidades integrando ARCH/GARCH e redes LSTMs foram elaborados e, por fim, um otimizador de alocações financeiros nos ativos selecionados foi concebido por meio das técnicas elaboradas por Markowitz (1959).

A estratégia proposta obteve retornos hipotéticos de 70,94%, sendo assim, ultrapassam os de *benchmarks* tipicamente utilizados como base de comparação para fundos acionários, notadamente Ibovespa e IBRX50, e se mostram superior a uma carteira formada pelos mesmos ativos em proporções iguais, obtendo retornos maiores sob um risco menor – medido pela volatilidade anualizada, que foi de 18,16%. Portanto, os objetivos deste trabalho foram alcançados na medida em que foi criada uma estratégia quantitativa cujos retornos supere aqueles de *benchmarks*.

Apesar de o modelo demonstrar ser eficiente, diversas melhorias podem ser desenvolvidas, entre as quais, destacam-se:

- **Uso de redes neurais do tipo NLP (Processamento de Língua Natural, do inglês "Natural Language Processing") em combinação com programas que automaticamente captam manchetes e notícias do dia sobre um determinado tema** - As redes NLPs são muito utilizadas para análise de sentimento, classificando textos de acordo com o sentimento que transmitem, podendo esse ser negativo, neutro ou positivo. Como manchetes jornalísticas possuem alto efeito sobre os retornos de períodos curtos de ações (ORMOS; VáZSONYI, 2011), a utilização de análise de sentimento como variável explicatória nas redes neurais de previsões de preços e volatilidades poderia trazer ganhos nas previsões.
- **Adoção de estratégias que envolvam *vendas a descoberto*, ou *short*** - Tal modalidade consiste na venda de uma ação que o vendedor não possui. Para tanto, o vendedor pri-

meiro efetua a venda, aluga uma ação para fornecê-la ao comprador e, futuramente, deve comprar a ação de um terceiro para entregá-la ao comprador inicial, retornando a ação alugada a seu proprietário de origem. Dessa forma, os retornos do ”short” são advindos do decréscimo do preço da ação, já que o vendedor inicialmente vendeu o ativo a um preço negociado e, cajo haja queda no preço, pagará um valor menor para obtê-lo. Dessa forma, poderia se incluir na otimização de portfólios a possibilidade de venda a descoberto de ações que a rede neural prevê haver retornos de 20 dias negativos.

- **Integração dos custos transacionais na maximização dos retornos** - Existem diversos custos operacionais envolvidos na compra e venda de ativos, que dependem de diversos fatores diferentes. Tais custos, como a corretagem, podem variar de acordo com o ativo e também com a operação (compra, venda ou aluguel de ações). Assim, pode-se desenvolver um modelo de otimização no qual a função objetivo considere os custos transacionais.

Referências

ALMEIDA, M. F. de. ***Uma análise comparativa da volatilidade dos mercados acionários do Brasil e EUA em tempos de crise.*** 2009.

B3 - Ibovespa. Disponível em: ⟨http://www.b3.com.br/pt_br/market-data-e-indices/indices/indices-amplos/ibovespa.htm⟩. Acesso em: 29/06/2021.

BOLLERSLEV, T. ***Generalized autoregressive conditional heteroskedasticity***. *Journal of Econometrics*, v. 31, n. 3, p. 307–327, April 1986.

BOLLERSLEV, T.; CHOU, R. Y.; KRONER, K. F. ***ARCH modelling in finance: a review of the theory and empirical evidence***. *Journal of Econometrics*, v. 52, p. 5–59, 1992.

COLBY, R.; MEYERS, T. **The Encyclopedia of Technical Market Indicators**. [S.l.]: Dow Jones-Irwin, 1988. ISBN 9781556230493.

CORPORATE Finance Institute - Financial Assets. Disponível em: ⟨https://corporatefinanceinstitute.com/resources/knowledge/accounting/financial-assets/⟩. Acesso em: 05/07/2021.

CVM - Portal do Investidor - Fundos de Investimento. Disponível em: ⟨https://www.investidor.gov.br/menu/primeiros_passos/Investindo/Tipos_Investimento/Fundos_Investimento.html⟩. Acesso em: 06/07/2021.

ENGLE, R. F. ***Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation***. *Econometrica*, v. 50, n. 4, p. 987–1007, July 1982.

FIESLER, E. ***Neural network classification and formalization***. *Computer Standards & Interfaces*, v. 16, p. 231–239, 1994.

GOODFELLOW, I. J.; BENGIO, Y.; COURVILLE, A. C. **Deep Learning**. MIT Press, 2016. (Adaptive computation and machine learning). ISBN 978-0-262-03561-3. Disponível em: ⟨http://www.deeplearningbook.org/⟩.

GUBNER, J. A. **Probability and Random Processes for Electrical and Computer Engineers**. USA: Cambridge University Press, 2006. ISBN 0521864704.

HAWKINS, D. M. **Identification of outliers**. [S.l.]: Chapman and Hall, 1980. (Monographs on applied probability and statistics).

HIRANSHA, M. et al. ***NSE Stock Market Prediction Using Deep-Learning Models***. *Procedia Computer Science*, v. 132, p. 1351–1362, 2018. ISSN 1877-0509. International Conference on Computational Intelligence and Data Science.

HOCHREITER, S.; SCHMIDHUBER, J. ***Long Short-Term Memory***. *Neural Computation*, v. 9, n. 8, p. 1735–1780, 1997.

JAMES, G. et al. **An Introduction to Statistical Learning: with Applications in R**. [S.l.]: Springer, 2013.

JORION, P. **Value at Risk – The New Benchmark for Managing Financial Risk**. [S.l.: s.n.], 2007. v. 21. 397-398 p.

KIM, H. Y.; WON, C. H. ***Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models***. *Expert Systems With Applications*, 2018.

KINGMA, D. P.; BA, J. **Adam: A Method for Stochastic Optimization**. 2014. Disponível em: ⟨http://arxiv.org/abs/1412.6980⟩.

MA, Y.; HAN, R.; WANG, W. ***Portfolio optimization with return prediction using deep learning and machine learning***. *Expert Systems With Applications*, 2021.

MANDELBROT, B. ***The Variation of Some Other Speculative Prices***. *The Journal of Business*, University of Chicago Press, v. 40, n. 4, p. 393–413, 1967. ISSN 00219398, 15375374. Disponível em: ⟨http://www.jstor.org/stable/2351623⟩.

MANKIW, N. G. **Introdução à Economia**. [S.l.]: Pioneira Thomson Learning, 2005. ISBN 978-8522127917.

MARKOWITZ, H. M. **Portfolio selection : efficient diversification of investment**. Oxford: [s.n.], 1959. ISBN 1-55786-108-0.

MITCHELL, T. M. **Machine Learning**. New York: McGraw-Hill, 1997. ISBN 978-0-07-042807-2.

NEELY, C. J. et al. ***Forecasting the Equity Risk Premium: The Role of Technical Indicators***. *SSRN Electronic Journal*, 2011.

NWANKPA, C. E. et al. ***Activation Functions: Comparison of Trends in Practice and Research for Deep Learning***. 2020.

ORMOS, M.; VáZSONYI, M. ***Impacts of Public News on Stock Market Prices: Evidence from S&P(500)***. *Interdisciplinary Journal of Research in Business*, v. 1, p. 1–17, 2011.

REYNALDO, C. ***Regressão 'Ridge': Um Método Alternativo para o Mal Condicionamento da Matriz das Regressoras***. 1997.

ROMAN, D.; MITRA, G. ***Portfolio selection models: a review and new directions***. *Wilmott Journal*, v. 1, p. 69–85, 2009.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. ***Learning Representations by Back-propagating Errors***. *Nature*, v. 323, n. 6088, p. 533–536, 1986.

SECURATO, J. R. et al. **Cálculo financeiro das tesourarias: bancos e empresas**. [S.l.]: Saint Paul, 2003.

STEWART, J. **Calculus : early transcendentals**. Belmont, Cal.: Brooks/Cole, Cengage Learning, 2013. ISBN 978-1285741550.

VIEIRA, S.; PINAYAB, W. H.; MECHELLI, A. ***Using deep learning to investigate the neuroimaging correlates ofpsychiatric and neurological disorders: Methods and applications***. *Neuroscience and Biobehavioural reviews*, v. 74, p. 58–75, 2017.

YEOM, S. et al. ***Overfitting, robustness and malicious algorithms: A study of potential causes of privacy risk in machine learning***. *Journal of Computer Security*, v. 28, n. 1, p. 35–70, 2020.

YING, X. ***An Overview of Overfitting and its Solutions***. *Journal of Physics: Conference Series*, v. 1168, 2019.

APÊNDICE A – LISTA DE ATIVOS UTILIZADOS COMO VARIÁVEIS EXPLICATÓRIAS

Tabela 4: Ativos selecionados como variáveis explicatórias das redes neurais

PETR4.SA	PETR3.SA	VALE5.SA	VALE3.SA
ITUB4.SA	ABEV3.SA	BBDC4.SA	TNLP4.SA
EMBR4.SA	UBBR11.SA	CSNA3.SA	ITSA4.SA
GGBR4.SA	CMIG4.SA	USIM5.SA	BRKM5.SA
AMBV3.SA	BBDC3.SA	ARCZ6.SA	BRTP4.SA
ELET3.SA	VIVO4.SA	VCPA4.SA	GOAU4.SA
CMET4.SA	TMAR5.SA	CRUZ3.SA	TNLP3.SA
ELET6.SA	BELG4.SA	KLBN4.SA	BRAP4.SA
OIBR4.SA	CPSL3.SA	SDIA4.SA	SBSP3.SA
CSTB4.SA	PCAR4.SA	BBAS3.SA	VIVT4.SA
CCRO3.SA	TCSL4.SA	LAME4.SA	WEGE4.SA
TCOC4.SA	EMBR3.SA	SUZB5.SA	BRTP3.SA
UNIP6.SA	VIVT3.SA	EBTP4.SA	CPLE6.SA
CTNM4.SA	TRPL4.SA	UGPA4.SA	PTIP4.SA
EGIE3.SA	TMCP4.SA	FFTL4.SA	DURA4.SA
CMIG3.SA	CNFB4.SA	RPSA4.SA	CGAS5.SA
RAPT4.SA	ELPL5.SA	TMCP3.SA	TIMS3.SA
CRTP5.SA	CLSC4.SA	POMO4.SA	CESP5.SA
RIPI4.SA	TBLE6.SA	MAGS5.SA	NETC4.SA
CSPC4.SA	ETER3.SA	COCE5.SA	CPLE3.SA
FESA4.SA	MYPK4.SA	TPRC6.SA	PMAM4.SA
BOBR4.SA	TSEP4.SA	TCOC3.SA	VIVO3.SA
SAPR4.SA	LIGH3.SA	TLCP4.SA	TNCP4.SA
SALM4.SA	TASA4.SA	INEP4.SA	ACES4.SA
ACES3.SA	PRGA4.SA	USDBRL	USDMXN
USDCOP	EURBRL	BRLARS	GBPBRL
EURUSD	IBOV	UVXY	VIXM
GS	JPM	BAC	C
C	MS	WFC	

Fonte: elaborado pelo autor

APÊNDICE B - REDE NEURAL PREVISORA DE RETORNOS

```
from utils import *

import time
import pandas as pd
import xlwings as xw
import numpy as np
import scipy as sp
from scipy.fft import fft, ifft

from mxnet import nd, autograd, gluon
from mxnet.gluon import nn, rnn
import mxnet as mx
import datetime as dt
import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler

from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau, Model
Checkpoint, TensorBoard

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import regularizers

import xgboost as xgb
from sklearn.metrics import accuracy_score

print("------- Imports concluídos com sucesso -------")

# Pegando os valores dos preços diários

caminho = "C:\\Users\\breno\\Desktop\\Faculdade2\\TF\\TF\\Coletador de dados\\
"
filename = "Dados-v2.xlsx"
```

```
dfPrecos = pd.read_excel(caminho+filename, sheet_name="Precos", engine='openpy
xl')
ativos_portfolio = ["PETR4.SA", "PETR3.SA", "VALE5.SA", "VALE3.SA", "ITUB4.SA"
, "ABEV3.SA", "BBDC4.SA", "CSNA3.SA", "ITSA4.SA", "GGBR4.SA", "CMIG4.SA", "USI
M5.SA", "BRKM5.SA", "BBDC3.SA", "ELET3.SA", "GOAU4.SA", "ELET6.SA", "KLBN4.SA"
, "BRAP4.SA", "OIBR4.SA", "SBSP3.SA", "BBAS3.SA", "CCRO3.SA", "LAME4.SA", "EMB
R3.SA", "UNIP6.SA", "VIVT3.SA", "CPLE6.SA", "CTNM4.SA", "TRPL4.SA", "EGIE3.SA"
, "CMIG3.SA", "CGAS5.SA", "RAPT4.SA", "TIMS3.SA", "CLSC4.SA", "POMO4.SA", "CES
P5.SA", "ETER3.SA", "COCE5.SA", "CPLE3.SA", "FESA4.SA", "BOBR4.SA", "SAPR4.SA"
, "TNCP4.SA", "TASA4.SA", "INEP4.SA"]



################ Criando indicadores tecnicos para todos os ativos que podem c
ompor a carteira
def get_technical_indicators(dataset, ativo):
    # Retorno do ativo em 1, 5 e 20 dias
    for n in [1,5,20]:
        dataset[ativo + '_' + str(n) + '_prev_days_returns'] = (dfPrecos[ativo
]/dfPrecos[ativo].shift(n))-1

    # Médias móveis de 2, 7 e 21 dias
    dataset[ativo+'_ma2'] = dataset[ativo].rolling(window=2).mean()
    dataset[ativo + '_ma7'] = dataset[ativo].rolling(window=7).mean()
    dataset[ativo + '_ma21'] = dataset[ativo].rolling(window=21).mean()

    # MACD
    dataset[ativo + '_26ema'] = dataset[ativo].ewm(span=26).mean()
    dataset[ativo + '_12ema'] = dataset[ativo].ewm(span=12).mean()

    dataset[ativo + '_MACD'] = (dataset[ativo + '_12ema']-
dataset[ativo + '_26ema'])

    # Bollinger Bands
    dataset[ativo + '_20sd'] = dataset[ativo].rolling(window=20).std()
    dataset[ativo + '_upper_band'] = dataset[ativo + '_ma21'] + (dataset[ativo
 + '_20sd']*2)
    dataset[ativo + '_lower_band'] = dataset[ativo + '_ma21'] - (dataset[ativo
 + '_20sd']*2)

    # EWMA
    dataset[ativo + '_ema'] = dataset[ativo].ewm(com=0.5).mean()


    return dataset

for ativo in ativos_portfolio:
    dfPrecos = get_technical_indicators(dfPrecos, ativo)
```

```
'''
Definindo o que será previsto: o retorno do ativo em vinte dias
'''
n_dias = 20

dfResults = pd.DataFrame()
dfResults['Date'] = dfPrecos['Date']
for ativo in ativos_portfolio:
    dfResults[ativo+'_20_day_return'] = (dfPrecos[ativo].shift(-
n_dias)/dfPrecos[ativo])-1

dfPrecos.drop(dfPrecos.index[:n_dias], inplace=True) # retirando o que os indi
cadores técnicos não conseguiam calcular até certa data
dfResults.drop(dfResults.index[:n_dias], inplace=True)

dfPrecos.drop(dfPrecos.index[-
n_dias:], inplace=True) # retirando o que os dias que não podemos calcular os
retornos de 20 dias depois
dfResults.drop(dfResults.index[-n_dias:], inplace=True)


############################### Preparando os dados ##########################
#####

n_treino = int(0.75*len(dfPrecos['ITUB4.SA']))
n_teste = len(dfPrecos['ITUB4.SA']) - n_treino
print(f"Dias para treino do modelo: {n_treino}")
print(f"Dias para se testar o modelo treinado: {n_teste}")
df_training = dfPrecos.iloc[:n_treino]

cols = list(df_training)[1:]

colsResults = []
for ativo in ativos_portfolio:
    colsResults.append(ativo+'_20_day_return')

datelist_train = list(df_training['Date'])

if type(datelist_train[0]) == str:
    datelist_train = [dt.datetime.strptime(date, '%Y-%m-
%d').date() for date in datelist_train]

for col in df_training.columns[1:]:
     df_training[col] = df_training[col].astype(float)

training_set = df_training[cols].values
results_set = dfResults[colsResults].values
```

```
'''
Standard scaling
Um para todas as variáveis independentes e outro para todas as dependentes
'''
sc = StandardScaler()
training_set_scaled = sc.fit_transform(training_set)

sc_predict = StandardScaler()
sc_predict.fit_transform(results_set)

x_train = []
y_train = []

n_dias = 20 # numero de dias no futuro que se quer prever o retorno dos ativos
n_passado = 40 # numero de dias no passado que se olhará para prever tais reto
rnos

for i in range(n_passado, len(training_set_scaled)+1):
    x_train.append(training_set_scaled[i-n_passado : i][:])
    y_train.append([results_set[i-1]])

x_train = np.array(x_train)
y_train = np.array(y_train)



################################ Criando a rede LSTM ############################
#####

# Decidindo um valor incial de neurônios na camada oculta como sendo o valor e
ntre o número de neurônios na entrada e os de saída.
n_input_nodes = len(cols)
n_output_nodes = len(ativos_portfolio)
n_hidden_nodes = int((n_input_nodes + n_output_nodes)/5)

# iniciando a rede neural sequencial
model = Sequential()

# primeira camada da LSTM
model.add(LSTM(units=n_hidden_nodes, return_sequences=False, input_shape=(n_pa
ssado, n_input_nodes), activity_regularizer=regularizers.l1(1e-2)))

# adicionando dropout para evitar overfitting
model.add(Dropout(0.25))

# Adicionando vetores de saída
model.add(Dense(units=n_output_nodes, activation="linear"))

# Adicionando um otimizador
```

```
model.compile(optimizer = Adam(learning_rate=0.0001), loss="mean_squared_error
")

#%%
############################## Treinando a rede ############################
##
print("--------------- Starting ---------------")
start_time = time.time()
print(time.asctime(time.localtime(start_time)))
print("----------------------------------------")

caminho = f"{os.path.dirname(__file__)}\\ResultadosDoModelo\\"
weights_filename = caminho + 'weights3.h5'
es = EarlyStopping(monitor='val_loss', min_delta=1e-
10, patience=15, verbose=1)
rlr = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=10, verbose=1
)
mcp = ModelCheckpoint(filepath=weights_filename, monitor='val_loss', verbose=1
, save_best_weight=True, save_weights_only=True)

tb = TensorBoard('logs')

history = model.fit(x_train, y_train, shuffle=True, epochs=500, callbacks=[es,
 rlr, mcp, tb], validation_split=0.2, verbose=1, batch_size=14)


print("---------------- Ending ----------------")
end_time = time.time()
print(time.asctime(time.localtime(end_time)))
print(f'Total time taken : {end_time-start_time}')
print("----------------------------------------")



################## Fazendo previsões de retornos futuros na amostra de teste #
#################
df_test = dfPrecos.iloc[n_treino-n_passado:]

cols = list(df_test)[1:]


datelist_test = list(df_test['Date'])

if type(datelist_test[0]) == str:
    datelist_test = [dt.datetime.strptime(date, '%Y-%m-
%d').date() for date in datelist_test]

for col in df_test.columns[1:]:
    df_test[col] = df_test[col].astype(float)
```

```
test_set = df_test[cols].values

# aplicando a mesma normalização aplicada na amostra de treino para a amostra
de teste
test_set_scaled = sc.transform(test_set)

x_test = []

for i in range(n_passado+1, len(test_set_scaled)+1):
    x_test.append(test_set_scaled[i-n_passado : i][:])

x_test = np.array(x_test)

predictions_future = model.predict(x_test)

y_pred_future = sc_predict.inverse_transform(predictions_future)

dfPredictions = pd.DataFrame()
dfPredictions = pd.DataFrame(y_pred_future, columns=ativos_portfolio)
dfPredictions['Date'] = df_test['Date'].iloc[n_passado:].values



ativo = 'ITUB4.SA'

plt.figure(figsize=(14, 5), dpi=100)
plt.plot(dfResults['Date'], dfResults[ativo + '_20_day_return'], label = 'Verd
adeiros retornos de ' + ativo)
plt.plot(dfPredictions['Date'], dfPredictions[ativo], label = 'Retornos previs
tos para ' + ativo)
plt.vlines(df_training['Date'].iloc[-1], -0.5, 0.5, linestyles='--
', colors='gray', label='Início das previsões')
plt.hlines(0, dfResults['Date'].iloc[0], dfResults['Date'].iloc[-
1], linestyles='--', colors='gray', label='Início das previsões')
plt.xlabel('Data')
plt.ylabel('Retorno de 20 dias')
plt.ylim(-0.5, 0.5)
plt.legend()
plt.show()


################################ Salvando as previsões futuras ################
################
filename = caminho+'previsaoRetornos.pkl'
dfPredictions.to_pickle(filename)
```

APÊNDICE C - REDE NEURAL PREVISORA DE COVARIÂNCIAS

```
from numpy.core.defchararray import index
from tensorflow.python.keras.engine import training
from utils import *

import time
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt

from sklearn.decomposition import PCA
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler

from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau, Model
Checkpoint, TensorBoard
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import regularizers
from tensorflow.python.keras.backend import shape

from sklearn.metrics import accuracy_score

print("------- Imports concluídos com sucesso -------")



# Pegando os valores dos preços diários

caminho = "C:\\Users\\breno\\Desktop\\Faculdade2\\TF\\TF\\Coletador de dados\\
"
filename = "PrecosComGarch.xlsx"
dfPrecos = pd.read_excel(caminho+filename, sheet_name="Precos com GARCH", engi
ne='openpyxl')
ativos_portfolio = ["PETR4.SA", "PETR3.SA", "VALE5.SA", "VALE3.SA", "ITUB4.SA"
, "ABEV3.SA", "BBDC4.SA", "CSNA3.SA", "ITSA4.SA", "GGBR4.SA", "CMIG4.SA", "USI
M5.SA", "BRKM5.SA", "BBDC3.SA", "ELET3.SA", "GOAU4.SA", "ELET6.SA", "KLBN4.SA"
, "BRAP4.SA", "OIBR4.SA", "SBSP3.SA", "BBAS3.SA", "CCRO3.SA", "LAME4.SA", "EMB
```

```
R3.SA", "UNIP6.SA", "VIVT3.SA", "CPLE6.SA", "CTNM4.SA", "TRPL4.SA", "EGIE3.SA"
, "CMIG3.SA", "CGAS5.SA", "RAPT4.SA", "TIMS3.SA", "CLSC4.SA", "POMO4.SA", "CES
P5.SA", "ETER3.SA", "COCE5.SA", "CPLE3.SA", "FESA4.SA", "BOBR4.SA", "SAPR4.SA"
, "TNCP4.SA", "TASA4.SA", "INEP4.SA"]
n_ativos = len(ativos_portfolio)




'''
Definindo o que será previsto: a covariancia dos ativos nos próximos 20 dias
'''

n_dias = 20

dfResults = pd.DataFrame()
dfResults['Date'] = dfPrecos['Date']
dfResults['Covariancias'] = np.nan
dfResults['Covariancias'] = dfResults['Covariancias'].astype(object)

for i in range(len(dfPrecos['Date'])-n_dias):
    dfResults['Covariancias'].iloc[i] = np.cov(dfPrecos[ativos_portfolio].iloc
[i+1:i+n_dias+1].values.transpose()).ravel()

for i in range(n_ativos**2):
    dfPrecos[len(dfPrecos.columns)] = np.nan

for i in range(n_dias, len(dfPrecos['Date'])):
    covariancias = dfResults['Covariancias'].iloc[i-n_dias]
    dfPrecos.loc[i, dfPrecos.columns[-(n_ativos**2):]] = covariancias
    if i in range(0,3000,100):
        print(i)



n_dias_garch = 200 # numero de dias que o garch esteve usando para fazer as es
timativas, nao há dados de GARCH para valores abaixo disso
dfPrecos.drop(dfPrecos.index[:n_dias_garch], inplace=True) # retirando o que o
s indicadores técnicos não conseguiam calcular até certa data
dfResults.drop(dfResults.index[:n_dias_garch], inplace=True)

dfPrecos.drop(dfPrecos.index[-
n_dias:], inplace=True) # retirando o que os dias que não podemos calcular os
retornos de 20 dias depois
dfResults.drop(dfResults.index[-n_dias:], inplace=True)

dfPrecos.reset_index(inplace=True)
dfResults.reset_index(inplace=True)
```

```
dfPrecos.drop('index', inplace=True, axis=1)
dfResults.drop('index', inplace=True, axis=1)

dfPrecos = dfPrecos.ffill()
dfPrecos = dfPrecos.fillna(0)


lista_tamanhos = []
for i in range(len(dfResults['Covariancias'])):
    if dfResults['Covariancias'].iloc[i].shape[0] not in lista_tamanhos:
        lista_tamanhos.append(dfResults['Covariancias'].iloc[i].shape[0])
print(lista_tamanhos)


############################### Preparando os dados ##########################
#####

n_treino = int(0.75*len(dfPrecos['ITUB4.SA']))
n_teste = len(dfPrecos['ITUB4.SA']) - n_treino
print(f"Dias para treino do modelo: {n_treino}")
print(f"Dias para se testar o modelo treinado: {n_teste}")
df_training = dfPrecos.iloc[:n_treino]

cols = list(df_training)[1:]

datelist_train = list(df_training['Date'])

if type(datelist_train[0]) == str:
    datelist_train = [dt.datetime.strptime(date, '%Y-%m-
%d').date() for date in datelist_train]

for col in df_training.columns[1:]:
     df_training[col] = df_training[col].astype(float)

training_set = df_training[cols].values
results_set = dfResults['Covariancias'].iloc[:n_treino].values
a = []
for line in results_set:
    a.append(line.tolist())
results_set = np.array(a)



'''
Standard scaling
Um para todas as variáveis independentes e outro para todas as dependentes
'''
sc = StandardScaler()
training_set_scaled = sc.fit_transform(training_set)
```

```
sc_predict = StandardScaler()
sc_predict.fit_transform(results_set)

x_train = []
y_train = []

n_dias = 20 # numero de dias no futuro que se quer prever o retorno dos ativos
n_passado = 40 # numero de dias no passado que se olhará para prever tais reto
rnos

for i in range(n_passado, len(training_set_scaled)+1):
    x_train.append(training_set_scaled[i-n_passado : i][:])
    y_train.append([results_set[i-1]])

x_train = np.array(x_train)
y_train = np.array(y_train)



############################### Criando a rede LSTM ##########################
#####

# Decidindo um valor incial de neurônios na camada oculta como sendo o valor e
ntre o número de neurônios na entrada e os de saída.
n_input_nodes = len(cols)
n_output_nodes = n_ativos**2 # matriz de covariancias n_ativos x n_ativos
n_hidden_nodes = int((n_input_nodes + n_output_nodes)/30)

# iniciando a rede neural sequencial
model = Sequential()

# primeira camada da LSTM
model.add(LSTM(units=n_hidden_nodes, return_sequences=True, input_shape=(n_pas
sado, n_input_nodes), kernel_regularizer=regularizers.l2(1e-
3), activity_regularizer=regularizers.l1(1e-2)))

# segunda camada de LSTM
n_second_layer_nodes = int(n_hidden_nodes/2)
model.add(LSTM(units=n_second_layer_nodes, return_sequences=False, activity_re
gularizer=regularizers.l2(5e-1)))

# adicionando dropout para evitar overfitting
model.add(Dropout(0.25))

# Adicionando vetores de saída
model.add(Dense(units=n_output_nodes, activation="linear"))

# Adicionando um otimizador
```

```
model.compile(optimizer = Adam(learning_rate=0.0001), loss="mean_squared_error
")


############################## Treinando a rede ##############################
##
print("--------------- Starting ---------------")
start_time = time.time()
print(time.asctime(time.localtime(start_time)))
print("----------------------------------------")

caminho = f"{os.path.dirname(__file__)}\\ResultadosDoModelo\\"
weights_filename = caminho + 'weightsCovariance.h5'
es = EarlyStopping(monitor='val_loss', min_delta=1e-
10, patience=15, verbose=1)
rlr = ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=10, verbose=1
)
mcp = ModelCheckpoint(filepath=weights_filename, monitor='val_loss', verbose=1
, save_best_weight=True, save_weights_only=True)

tb = TensorBoard('logs')

history = model.fit(x_train, y_train, shuffle=True, epochs=50, callbacks=[es,
rlr, mcp, tb], validation_split=0.2, verbose=1, batch_size=14)



print("---------------- Ending ----------------")
end_time = time.time()
print(time.asctime(time.localtime(end_time)))
print(f'Total time taken : {end_time-start_time}')
print("----------------------------------------")



############################## Fazendo previsões na amostra de teste ########
########################
df_test = dfPrecos.iloc[n_treino-n_passado:]

cols = list(df_test)[1:]


datelist_test = list(df_test['Date'])

if type(datelist_test[0]) == str:
    datelist_test = [dt.datetime.strptime(date, '%Y-%m-
%d').date() for date in datelist_test]

for col in df_test.columns[1:]:
```

```
    df_test[col] = df_test[col].astype(float)

test_set = df_test[cols].values

# aplicando a mesma normalização aplicada na amostra de treino para a amostra
de teste
test_set_scaled = sc.transform(test_set)

x_test = []

for i in range(n_passado+1, len(test_set_scaled)+1):
    x_test.append(test_set_scaled[i-n_passado : i][:])

x_test = np.array(x_test)

predictions_future = model.predict(x_test)

y_pred_future = sc_predict.inverse_transform(predictions_future)


################## Recriando as matrizes #################
dfPredictionMatrices = pd.DataFrame()
dfPredictionMatrices['Date'] = dfPrecos['Date'].loc[n_treino:]
dfPredictionMatrices['Covariancias'] = np.nan
dfPredictionMatrices['Covariancias'] = dfPredictionMatrices['Covariancias'].as
type(object)
for i in range(y_pred_future.shape[0]):
    dfPredictionMatrices['Covariancias'].iloc[i] = y_pred_future[i].reshape(n_
ativos, n_ativos)

dfPredictionMatrices['Covariancias'].iloc[-1].shape



################################ Salvando as previsões futuras ################
################
filename = caminho+'previsaoCovariancias.pkl'
dfPredictionMatrices.to_pickle(filename)
```

APÊNDICE D - MODELOS ARCH/GARCH

```
######################## Imports ########################
import time
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import xlwings as xw

from arch import arch_model
from arch.__future__ import reindexing
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

pd.options.mode.chained_assignment = None  # default='warn'

def findTerm(term, array):
    count = 0
    while array[count] != term:
        count += 1
    if count == len(array):
        return('invalid' )
    else:
        return(count)

print("------- Imports concluídos com sucesso -------")


######################## Pegando os dados de preços dos ativos ###############
#########
path = "C:\\Users\\breno\\Desktop\\Faculdade2\\TF\\TF\\Coletador de dados\\"
filename = "Dados-v2.xlsx"
sheetname = "Precos"



dfPrecos = pd.read_excel(path+filename, sheetname, engine='openpyxl', index_co
l='Date')
sheetname = "Precos com GARCH"
df = pd.read_excel(path+filename, sheetname, engine='openpyxl', index_col='Dat
e')
```

```
filenameConsertado = r'C:\Users\breno\Desktop\Faculdade2\TF\TF\Coletador de da
dos\PrecosComGarch.xlsx'
wb = xw.Book(filenameConsertado)
sht = wb.sheets("Precos com GARCH")
sht.range('A1').value = df



######################### Criando as colunas dos parâmetros ARCH/GARCH #######
##################

listaTickers = ["PETR4.SA", "PETR3.SA", "VALE5.SA", "VALE3.SA", "ITUB4.SA", "A
BEV3.SA", "BBDC4.SA", "CSNA3.SA", "ITSA4.SA", "GGBR4.SA", "CMIG4.SA", "USIM5.S
A", "BRKM5.SA", "BBDC3.SA", "ELET3.SA", "GOAU4.SA", "ELET6.SA", "KLBN4.SA", "B
RAP4.SA", "OIBR4.SA", "SBSP3.SA", "BBAS3.SA", "CCRO3.SA", "LAME4.SA", "EMBR3.S
A", "UNIP6.SA", "VIVT3.SA", "CPLE6.SA", "CTNM4.SA", "TRPL4.SA", "EGIE3.SA", "C
MIG3.SA", "CGAS5.SA", "RAPT4.SA", "TIMS3.SA", "CLSC4.SA", "POMO4.SA", "CESP5.S
A", "ETER3.SA", "COCE5.SA", "CPLE3.SA", "FESA4.SA", "BOBR4.SA", "SAPR4.SA", "T
NCP4.SA", "TASA4.SA", "INEP4.SA"]

n_samples = 200
p_list = range(1, 3)
q_list = range(1, 3)

n_dias = 20 # numero de dias no futuro que queremos prever


wb = xw.Book(path+filename)
sht = wb.sheets("Precos com GARCH")


for ativo in listaTickers:
    if (ativo != 'Date'):
        first_index = df[ativo].first_valid_index()
        line = np.where(df.index == first_index)[0][0]
        first_line = line + n_samples
        last_line = len(df[ativo])
        print(f'Iniciando {ativo}\n')

        if first_line < last_line:
            for i in range(first_line, last_line):
                returns_sample = df[ativo].iloc[i-n_samples:i]
                for p in p_list:
                    for q in q_list:
                        model = arch_model(returns_sample, p=p, q=q)
                        model_fit = model.fit(disp='off')
                        resulting_parameters = model_fit.params
                        pred = model_fit.forecast(horizon=20)
```

```
                        for param in resulting_parameters.keys():
                            col_name = ativo + "-garch-p" + str(p) + "-
q" + str(q) + "-" + str(param)
                            if col_name not in df.columns:
                                df[col_name] = np.nan
                            df[col_name].iloc[i] = resulting_parameters[param]

                        col_name = ativo + "-garch-p" + str(p) + "-
q" + str(q) + "-prediction"
                        if col_name not in df.columns:
                            df[col_name] = np.nan
                        df[col_name].iloc[i] = pred.variance.values[-
1,:][0]

                        model = arch_model(returns_sample, p=p, q=q, vol='ARCH
')
                        model_fit = model.fit(disp='off')
                        resulting_parameters = model_fit.params
                        pred = model_fit.forecast(horizon=1)

                        for param in resulting_parameters.keys():
                            col_name = ativo + "-arch-p" + str(p) + "-
q" + str(q) + "-" + str(param)
                            if col_name not in df.columns:
                                df[col_name] = np.nan
                            df[col_name].iloc[i] = resulting_parameters[param]

                        col_name = ativo + "-arch-p" + str(p) + "-
q" + str(q) + "-prediction"
                        if col_name not in df.columns:
                            df[col_name] = np.nan

                        df[col_name].iloc[i] = pred.variance.values[-1,:][0]

    time.sleep(2)
    df.to_pickle(path+'dfGarchs.pkl')
    sht.range('A1').value = df


print("---------- Fim do cálculo dos modelos ----------")



returns = np.array(dfItau['Returns'])
plot_pacf(returns**2)

n_treino = int(0.75*len(dfItau['ITUB4.SA']))
n_teste = len(dfItau['ITUB4.SA']) - n_treino
```

```
print(f'n_treino={n_treino:.0f} -- n_teste={n_teste:.0f}')

rolling_predictions = []


for i in range(n_teste):
    treino = returns[:-(n_teste+i)]
    model = arch_model(treino, p=2, q=2)
    model_fit = model.fit(disp='off')
    pred = model_fit.forecast(horizon = 20)
    rolling_predictions.append(np.sqrt(pred.variance.values[-1,:][0]))


rolling_std = pd.Series(dfItau['20sd'][-
n_teste:].values, index = dfItau['Date'][-n_teste:])
rolling_predictions_pd = pd.Series(rolling_predictions, index = dfItau['Date']
[-n_teste:])
real_returns = pd.Series(dfItau['Returns'][-
n_teste:].values, index = dfItau['Date'][-n_teste:])

plt.figure(figsize=(10, 4))
true, = plt.plot(real_returns)
preds, = plt.plot(rolling_predictions_pd)
stds, = plt.plot(rolling_std)
plt.title('Previsão de volatilidade das ações do Itaú (ITUB4)')
plt.legend(['Retorno real diário', 'Volatilidade prevista', 'Vol realizada (20
 dias)'], fontsize=12)



real_returns = pd.Series(dfItau['Returns'].iloc[-
n_teste:].values, index = dfItau['Date'][-n_teste:])
print(real_returns)
```

APÊNDICE E - REBALANCEADOR DE PORTFÓLIO

```
from numpy.lib.utils import byte_bounds
import pandas as pd
import numpy as np
import time
import datetime as dt
from scipy.optimize import minimize
import xlwings as xw



############################ Recuperando os dados de previsão ################
##########

caminho = "C:\\Users\\breno\\Desktop\\Faculdade2\\TF\\TF\\Coletador de dados\\
"
filename = "Dados-v2.xlsx"
dfPrecos = pd.read_excel(caminho+filename, sheet_name="Precos", engine='openpy
xl')
ativos_portfolio = ["PETR4.SA", "PETR3.SA", "VALE5.SA", "VALE3.SA", "ITUB4.SA"
, "ABEV3.SA", "BBDC4.SA", "CSNA3.SA", "ITSA4.SA", "GGBR4.SA", "CMIG4.SA", "USI
M5.SA", "BRKM5.SA", "BBDC3.SA", "ELET3.SA", "GOAU4.SA", "ELET6.SA", "KLBN4.SA"
, "BRAP4.SA", "OIBR4.SA", "SBSP3.SA", "BBAS3.SA", "CCRO3.SA", "LAME4.SA", "EMB
R3.SA", "UNIP6.SA", "VIVT3.SA", "CPLE6.SA", "CTNM4.SA", "TRPL4.SA", "EGIE3.SA"
, "CMIG3.SA", "CGAS5.SA", "RAPT4.SA", "TIMS3.SA", "CLSC4.SA", "POMO4.SA", "CES
P5.SA", "ETER3.SA", "COCE5.SA", "CPLE3.SA", "FESA4.SA", "BOBR4.SA", "SAPR4.SA"
, "TNCP4.SA", "TASA4.SA", "INEP4.SA"]
n_ativos = len(ativos_portfolio)

naiveForecasting = True

if naiveForecasting == False:
    caminho = "C:\\Users\\breno\\Desktop\\Faculdade2\\TF\\TF\\Code\\Resultados
DoModelo\\"
    dfPredictionReturns = pd.DataFrame()
    dfPredictionCovariances = pd.DataFrame()
    dfPredictionReturns = pd.read_pickle(caminho+'previsaoRetornos.pkl')
    dfPredictionCovariances = pd.read_pickle(caminho+'previsaoCovariancias.pkl
')
else:
    dfPredictionReturns = pd.DataFrame()
    dfPredictionReturns['Date'] = dfPrecos['Date']
```

```
    for ativo in ativos_portfolio:
        dfPredictionReturns[ativo] = (dfPrecos[ativo]/dfPrecos[ativo].shift(20
))-1
    dfPredictionCovariances = pd.DataFrame()
    dfPredictionCovariances['Date'] = dfPrecos['Date']
    dfPredictionCovariances['Covariancias'] = np.nan
    dfPredictionCovariances['Covariancias'] = dfPredictionCovariances['Covaria
ncias'].astype(object)
    tamanho = int(len(dfPrecos['Date']))
    for i in range(int(tamanho/2), tamanho):
        dfPredictionCovariances['Covariancias'].iloc[i] = np.cov(dfPredictionR
eturns[ativos_portfolio].iloc[i-19:i+1].values.transpose())



variancia_maxima = (0.14/(252**0.5))**2

maxima_alocacao = 0.40
n_dias = 20


# Setando os períodos de previsão iguais para ambos os dataframes de previsão
first_day = max(dfPredictionReturns['Date'].iloc[0], dfPredictionCovariances['
Date'].iloc[0])
first_day
i_start = 0
if dfPredictionCovariances['Date'].iloc[0] > dfPredictionReturns['Date'].iloc[
0]:
    while dfPredictionReturns['Date'].iloc[i_start] != first_day:
        i_start += 1
    dfPredictionReturns = dfPredictionReturns.iloc[i_start:]

elif dfPredictionReturns['Date'].iloc[0] > first_day:
    while dfPredictionCovariances['Date'].iloc[i_start] != dfPredictionReturns
['Date'].iloc[0]:
        i_start += 1
    dfPredictionCovariances = dfPredictionCovariances.iloc[i_start:]

if dfPrecos['Date'].iloc[0] != first_day:
    i_start = 0
    while dfPrecos['Date'].iloc[i_start] != first_day:
        i_start += 1
    dfPrecos = dfPrecos.iloc[i_start:]

dfPrecos = dfPrecos.iloc[:len(dfPredictionReturns['Date']) + n_dias]


dfPredictionReturns.reset_index(drop=True, inplace=True)
dfPredictionCovariances.reset_index(drop=True, inplace=True)
```

```
dfPrecos.reset_index(drop=True, inplace=True)


bound = [0.0, 1.0]
bounds = []
for i in range(n_ativos):
    bounds.append(bound)

def otimizaPeriodo(ativos_portfolio, retornos, covariancias):
    n_ativos = len(ativos_portfolio)
    if len(retornos) != n_ativos or len(covariancias) != n_ativos:
        return('Erro: tamanho dos vetores não são iguais')
    for i in range(len(covariancias)):
        if len(covariancias[i]) != n_ativos:
            return('Erro: tamanho dos vetores não são iguais')
    retornos = np.array(retornos)

    global bounds

    x0 = [(1/n_ativos) for i in range(n_ativos)] # estimativa inicial de aloca
ção

    con1 = {'type': 'ineq', 'fun': constraint1}
    con2 = {'type': 'ineq', 'fun': constraint2}
    con3 = {'type': 'ineq', 'fun': constraint3}

    cons = [con1, con2, con3]

    sol = minimize(funcaoObjetivo, x0, method='SLSQP', bounds=bounds, constrai
nts=cons)
    return(sol.x)


def funcaoObjetivo(x):
    # Minimizando o inverso do retorno (o mesmo que maximizando o retorno)
    global retornos
    npRetornos = np.array(retornos)
    x = np.array(x)
    return(-1 * np.matmul(npRetornos.transpose(), x))

def constraint1(x):
    global covariancias
    global variancia_maxima
    x = np.array(x)
    var = np.matmul(np.matmul(x.transpose(), covariancias), x)
    # return(var-variancia_maxima)
    return(variancia_maxima - var)

def constraint2(x):
```

```
    # A soma das alocacoes em cada ativo nao pode ultrapassar 100%
    soma = -1
    for termo in x:
        soma += termo
    # return(soma)
    return(-1*soma)

def constraint3(x):
    # Estabelecendo limite maximo de quanto pode ser alocado em cada ativo
    global maxima_alocacao
    maximo = 0
    for termo in x:
        if termo > maximo:
            maximo = termo
    return(maxima_alocacao - maximo)



retornoEsperado = dfPredictionReturns.values[0][:-1]
covEsperada = dfPredictionCovariances.values[0][1:][0]
for i in range(47):
    covEsperada[i][i] = abs(covEsperada[i][i])
covariancias = covEsperada
retornos = retornoEsperado
peso1 = otimizaPeriodo(ativos_portfolio, retornoEsperado, covariancias=covEspe
rada)


alocacoes = []
datas = []


for i in range(0, len(dfPredictionReturns), n_dias):
    retornos = dfPredictionReturns.values[i][:-1]
    covariancias = dfPredictionCovariances.values[i][1:][0]
    alocacao = otimizaPeriodo(ativos_portfolio, retornos, covariancias)

    datas.append(dfPredictionReturns['Date'].iloc[i])
    alocacoes.append(alocacao)


valoresFinais = np.array(alocacoes)



############################## Projetando os retornos ######################
########
dfRetornosReais = pd.DataFrame()
```

```
for ativo in ativos_portfolio:
    dfRetornosReais[ativo] = dfPrecos[ativo]/dfPrecos[ativo].shift(1)

dfRetornosReais = dfRetornosReais.fillna(value=1)

dfRetornosReais['NaoAlocado'] = 1

retornos_reais = dfRetornosReais.values
retornos_reais.shape

alocado_por_data = []
periodos = valoresFinais.shape[0] # periodos de rebalanceamento a cada n_dias
sobras = []

totais_alocados = valoresFinais.sum(axis=1)

for i in range(periodos):
    total_alocado = totais_alocados[i]
    sobra = max(0, 1-total_alocado)
    for j in range(n_dias):
        alocado_por_data.append(valoresFinais[i])
        sobras.append([sobra])

alocado_por_data = np.array(alocado_por_data)
sobras = np.array(sobras)

alocado_por_data = np.append(alocado_por_data, sobras, axis=1)

n_total_dias = alocado_por_data.shape[0]

retornos_reais = retornos_reais[:n_total_dias]


# Retornos diários da estratégia por ativo
retornos_estrategias_por_ativo = np.multiply(retornos_reais, alocado_por_data)

# Retorno diário da estratégia
retornos_estrategias = retornos_estrategias_por_ativo.sum(axis=1)

# Retorno total da estratégia ao longo de todo o período
retorno_final = np.prod(retornos_estrategias)

print (f'volatilidade anualizada = {100*retornos_estrategias.std()*(252**0.5):
.2f}%')
print (f'retorno_final = {retorno_final} = {100*(retorno_final - 1):.2f}%')
```

```
#################### Passando alocacoes ao Excel ####################
if naiveForecasting == False:
    caminho = "C:\\Users\\breno\\Desktop\\Faculdade2\\TF\\TF\\Code\\AlocacoesE
strategia.xlsx"
else:
    caminho = "C:\\Users\\breno\\Desktop\\Faculdade2\\TF\\TF\\Code\\AlocacoesE
strategiaNaiveForecasting.xlsx"
wb = xw.Book(caminho)
shtPrecos = wb.sheets('Precos diarios')
shtAlocacoes = wb.sheets('Alocacoes diarias')
shtRetornos = wb.sheets('Retornos diarios')

shtPrecos.range('A1').value = dfPrecos
shtAlocacoes.range('B2').value = alocado_por_data
shtRetornos.range('B2').value = retornos_estrategias_por_ativo
```