

UNIVERSIDADE FEDERAL DO PARANÁ

BARBARA SIMOES DE CARVALHO

MEMORIAL DE PROJETOS: STORYTELLING EM INTELIGÊNCIA ARTIFICIAL - A  
NARRATIVA GUIADA POR DADOS

CURITIBA

2025

BARBARA SIMOES DE CARVALHO

MEMORIAL DE PROJETOS: STORYTELLING EM INTELIGÊNCIA ARTIFICIAL - A  
NARRATIVA GUIADA POR DADOS

Memorial de Projetos apresentado ao curso de Especialização em Inteligência Artificial Aplicada, Setor de Educação Profissional e Tecnológica, Universidade Federal do Paraná, como requisito parcial à obtenção do título de Especialista em Inteligência Artificial Aplicada.

Orientadora: Profa. Dra. Rafaela Mantovani Fontana

CURITIBA

2025



## TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação Inteligência Artificial Aplicada da Universidade Federal do Paraná foram convocados para realizar a arguição da Monografia de Especialização de **BARBARA SIMOES DE CARVALHO**, intitulada: **MEMORIAL DE PROJETOS: STORYTELLING EM INTELIGÊNCIA ARTIFICIAL - A NARRATIVA GUIADA POR DADOS**, que após terem inquirido a aluna e realizada a avaliação do trabalho, são de parecer pela sua aprovação no rito de defesa.

A outorga do título de especialista está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 08 de Agosto de 2025.



RAFAELA MANTOVANI FONTANA  
Presidente da Banca Examinadora



RAZER ANTHOM NIZER ROJAS MONTAÑO  
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

## RESUMO

O presente trabalho explora o conceito de *storytelling* em ambientes mediados por inteligência artificial (IA), com foco específico na análise e visualização de dados. Em um contexto cada vez mais orientado por dados, torna-se essencial compreender como a narrativa baseada em informações quantitativas pode ser potencializada com o uso de tecnologias avançadas. A proposta deste estudo é discutir como a integração entre storytelling, ferramentas estatísticas, visualizações interativas e algoritmos inteligentes pode resultar em processos de comunicação mais eficazes, personalizados e acessíveis. O uso da IA permite automatizar a extração de padrões, adaptar narrativas ao perfil do público e facilitar a tomada de decisões em diferentes contextos, como ambientes corporativos e sistemas informacionais. Além disso, destaca-se o papel da narrativa como ponte entre dados complexos e interpretação humana, tornando mais intuitiva a compreensão de fenômenos diversos. Assim, o trabalho enfatiza a importância do storytelling como uma estratégia central para transformar dados em conhecimento relevante e aplicável.

**Palavras-chave:** storytelling, inteligência artificial, visualização de dados, análise de dados, business intelligence.

## ABSTRACT

This work explores the concept of storytelling in environments mediated by artificial intelligence (AI), with a specific focus on data analysis and visualization. In a world increasingly driven by data, it is essential to understand how narratives based on quantitative information can be enhanced through the use of advanced technologies. The aim of this study is to discuss how the integration of storytelling with statistical tools, interactive visualizations, and intelligent algorithms can lead to more effective, personalized, and accessible communication processes. AI enables the automation of pattern extraction, the adaptation of narratives to audience profiles, and facilitates decision-making in various contexts, such as corporate environments and informational systems. Additionally, the narrative plays a key role as a bridge between complex data and human interpretation, making it easier to understand diverse phenomena. Thus, this work highlights the importance of storytelling as a central strategy for transforming data into relevant and actionable knowledge.

**Keywords:** storytelling, artificial intelligence, data visualization, data analysis, business intelligence.



## SUMÁRIO

|                                                                 |            |
|-----------------------------------------------------------------|------------|
| <b>1 PARECER TÉCNICO.....</b>                                   | <b>7</b>   |
| <b>REFERÊNCIAS.....</b>                                         | <b>10</b>  |
| <b>APÊNDICE 1 – INTRODUÇÃO À INTELIGÊNCIA ARTIFICIAL.....</b>   | <b>11</b>  |
| <b>APÊNDICE 2 – LINGUAGEM DE PROGRAMAÇÃO APLICADA.....</b>      | <b>19</b>  |
| <b>APÊNDICE 3 – LINGUAGEM R.....</b>                            | <b>30</b>  |
| <b>APÊNDICE 4 – ESTATÍSTICA APLICADA I.....</b>                 | <b>36</b>  |
| <b>APÊNDICE 5 – ESTATÍSTICA APLICADA II.....</b>                | <b>41</b>  |
| <b>APÊNDICE 6 – ARQUITETURA DE DADOS.....</b>                   | <b>43</b>  |
| <b>APÊNDICE 7 – APRENDIZADO DE MÁQUINA.....</b>                 | <b>51</b>  |
| <b>APÊNDICE 8 – DEEP LEARNING.....</b>                          | <b>56</b>  |
| <b>APÊNDICE 9 – BIG DATA.....</b>                               | <b>80</b>  |
| <b>APÊNDICE 10 – VISÃO COMPUTACIONAL.....</b>                   | <b>82</b>  |
| <b>APÊNDICE 11 – ASPECTOS FILOSÓFICOS E ÉTICOS DA IA.....</b>   | <b>105</b> |
| <b>APÊNDICE 12 – GESTÃO DE PROJETOS DE IA.....</b>              | <b>113</b> |
| <b>APÊNDICE 13 – FRAMEWORKS DE INTELIGÊNCIA ARTIFICIAL.....</b> | <b>117</b> |
| <b>APÊNDICE 14 – VISUALIZAÇÃO DE DADOS E STORYTELLING.....</b>  | <b>135</b> |
| <b>APÊNDICE 15 – TÓPICOS EM INTELIGÊNCIA ARTIFICIAL.....</b>    | <b>145</b> |

## 1 PARECER TÉCNICO

Em uma sociedade orientada por dados, a habilidade de contar histórias com base em informações quantitativas é uma competência valiosa. O *storytelling* é a habilidade de comunicar com ajuda de narrativas que sejam capazes de envolver e emocionar o público alvo a fim de que a mensagem seja recebida com entusiasmo (Arruda, 2019). Quando potencializada por sistemas de inteligência artificial (IA), essa prática se torna poderosa, possibilitando a extração automatizada de padrões, a criação de visualizações adaptativas e a personalização de narrativas. Este parecer técnico visa discutir como a IA contribui para o fortalecimento do *storytelling* em visualização de dados.

A inteligência artificial permite a automação de tarefas analíticas, tornando o processo de extração de *insights* mais ágil e preciso. Segundo Bruce e Bruce (2020), ferramentas como R e Python facilitam a aplicação de conceitos estatísticos essenciais para o desenvolvimento de narrativas quantitativas, como regressões, agrupamentos e análises preditivas.

Essas análises, quando inseridas em uma narrativa, facilitam a compreensão de fenômenos complexos e embasam decisões estratégicas. Em ambientes corporativos, algoritmos de IA são capazes de gerar relatórios interativos e *dashboards* que se ajustam automaticamente ao perfil do usuário, promovendo *storytelling* personalizado.

A visualização de dados desempenha um papel fundamental na mediação entre dados e audiência. A IA pode agregar nessa mediação permitindo representações visuais interativas e em tempo real. No atual cenário corporativo orientado por dados, o uso da inteligência artificial integrada a ferramentas de *business intelligence* (BI) tem se consolidado como uma abordagem estratégica. O objetivo central dessas aplicações é transformar grandes volumes de dados brutos em *insights* acionáveis que auxiliem gestores na tomada de decisões informadas, ágeis e baseadas em evidências (Sharda; Delen; Turban, 2019).

As plataformas de BI evoluíram significativamente com a incorporação de recursos de IA, como aprendizado de máquina, processamento de linguagem natural

(PLN) e automação de relatórios. Esses recursos permitem que os sistemas aprendam com os próprios dados históricos da organização, identifiquem padrões e anomalias, e sugiram caminhos otimizados para alcançar metas estratégicas.

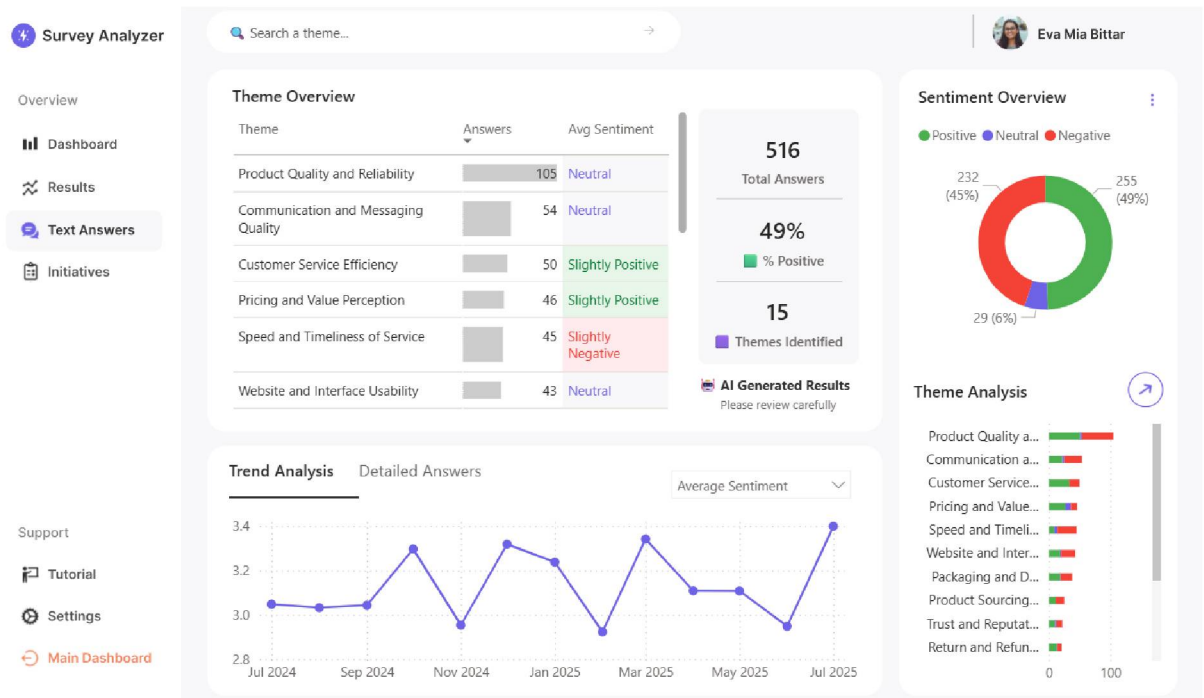
A integração entre BI e *storytelling* visual é essencial para a disseminação do conhecimento dentro das organizações. Por meio de painéis interativos (*dashboards*), é possível apresentar informações complexas de forma simplificada, destacando as métricas mais relevantes e conduzindo o usuário por uma jornada lógica de análise. A IA, nesse contexto, atua como um catalisador do processo, oferecendo recomendações automáticas e gerando análises preditivas que enriquecem a narrativa.

O dashboard *Analyzing Survey Comments*, disponível na *Microsoft Fabric Community* (Mehta, 2024), exemplifica a integração entre IA, *storytelling* e BI. A solução utiliza algoritmos de PLN para analisar comentários textuais de pesquisas, identificando automaticamente sentimentos, tópicos e termos mais frequentes, conforme mostra a FIGURA 1. Esses dados são apresentados por meio de visualizações interativas, como gráficos e nuvens de palavras, o que facilita a interpretação dos resultados. A narrativa é construída com base na disposição lógica e sequencial das informações, conduzindo o usuário em uma jornada de descoberta orientada por dados. Dessa forma, o *dashboard* transforma dados não estruturados em *insights* acionáveis, unindo a precisão analítica da IA, os recursos visuais do BI e a clareza comunicativa do *storytelling*, promovendo a tomada de decisões mais rápida, personalizada e baseada em evidências (Mehta, 2024).

A IA aplicada ao BI também pode contribuir para a democratização dos dados dentro das empresas. Ferramentas de *self-service* BI, empoderadas por assistentes inteligentes, permitem que qualquer colaborador, independentemente de seu nível hierárquico, possa explorar os dados e gerar *insights* próprios, promovendo uma cultura organizacional orientada à informação (Sharda; Delen; Turban, 2019).

Dessa forma, o *storytelling* com dados, quando aliado à inteligência artificial no contexto de *business intelligence*, se revela uma ferramenta poderosa para transformar dados em ações concretas, contribuindo significativamente para a competitividade e inovação das organizações.

FIGURA 1 – DASHBOARD “ANALYZING SURVEY COMMENTS”



FONTE: Mehta (2024).

A construção de narrativas com apoio de inteligência artificial representa uma convergência entre ciência de dados, comunicação visual e automação. O uso de IA no *storytelling* não apenas potencializa a análise e visualização de dados, mas também personaliza e dinamiza a forma como essas histórias são contadas. Ao unir estatística aplicada, design visual e inteligência computacional, é possível gerar narrativas que informam, engajam e influenciam decisões.

O futuro do *storytelling* em IA aponta para uma comunicação mais eficiente, baseada em dados confiáveis, apresentados de maneira acessível e adaptável às necessidades do público. Nesse cenário, a narrativa deixa de ser apenas um recurso retórico e passa a ser um componente estratégico essencial.

## REFERÊNCIAS

ARRUDA, Rafael Ferraz. **Comunicação Inteligente e Storytelling: para alavancar negócios e carreiras**. 1. ed. São Paulo: Alta Books, 2019. 256 p. ISBN 978-8550807836

BRUCE, Peter; BRUCE, Andrew. **Estatística prática para cientistas de dados: 50 + Essential Concepts Using R and Python**. 2. ed. Sebastopol: O'Reilly Media, 2020. 360 p. ISBN 978-1492072942

MEHTA, Amit. **Analyzing Survey Comments**. Microsoft Fabric Community, 23 jul. 2024. Disponível em: <https://community.fabric.microsoft.com/t5/Themes-Gallery/Analyzing-Survey-Comments/td-p/4761183>. Acesso em: 01 ago. 2025.

SHARDA, Ramesh; DELEN, Dursun; TURBAN, Efraim. **Business intelligence e análise de dados para gestão do negócio**. 4. ed. Porto Alegre: Bookman, 2019.

## APÊNDICE 1 – INTRODUÇÃO À INTELIGÊNCIA ARTIFICIAL

### A – ENUNCIADO

#### 1 ChatGPT

- (6,25 pontos)** Pergunte ao ChatGPT o que é Inteligência Artificial e cole aqui o resultado.
- (6,25 pontos)** Dada essa resposta do ChatGPT, classifique usando as 4 abordagens vistas em sala. Explique o porquê.
- (6,25 pontos)** Pesquise sobre o funcionamento do ChatGPT (sem perguntar ao próprio ChatGPT) e escreva um texto contendo no máximo 5 parágrafos. Cite as referências.
- (6,25 pontos)** Entendendo o que é o ChatGPT, classifique o próprio ChatGPT usando as 4 abordagens vistas em sala. Explique o porquê.

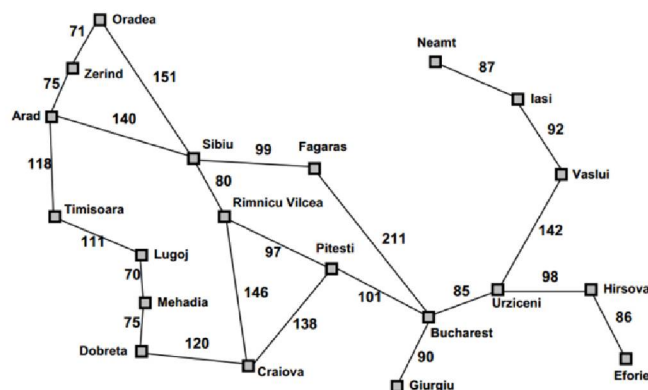
#### 2 Busca Heurística

Realize uma busca utilizando o algoritmo A\* para encontrar o melhor caminho para chegar a **Bucharest** partindo de **Lugoj**. Construa a árvore de busca criada pela execução do algoritmo apresentando os valores de  $f(n)$ ,  $g(n)$  e  $h(n)$  para cada nó. Utilize a heurística de distância em linha reta, que pode ser observada na tabela abaixo.

Essa tarefa pode ser feita em uma **ferramenta de desenho**, ou até mesmo no **papel**, desde que seja digitalizada (foto) e convertida para PDF.

- (25 pontos)** Apresente a árvore final, contendo os valores, da mesma forma que foi apresentado na disciplina e nas práticas. Use o formato de árvore, não será permitido um formato em blocos, planilha, ou qualquer outra representação.

**NÃO É NECESSÁRIO IMPLEMENTAR O ALGORITMO.**





|           |     |                |     |
|-----------|-----|----------------|-----|
| Arad      | 366 | Mehadia        | 241 |
| Bucareste | 0   | Neamt          | 234 |
| Craiova   | 160 | Oradea         | 380 |
| Drobeta   | 242 | Pitesti        | 100 |
| Eforie    | 161 | Rimnicu Vilcea | 193 |
| Fagaras   | 176 | Sibiu          | 253 |
| Giurgiu   | 77  | Timisoara      | 329 |
| Hirsova   | 151 | Urziceni       | 80  |
| Iasi      | 226 | Vaslui         | 199 |
| Lugoj     | 244 | Zerind         | 374 |

Figura 3.22 Valores de  $hDLR$  — distâncias em linha reta para Bucareste.

### 3 Lógica

Verificar se o argumento lógico é válido.

Se as uvas caem, então a raposa as come

Se a raposa as come, então estão maduras

As uvas estão verdes ou caem

Logo

A raposa come as uvas se e somente se as uvas caem

Deve ser apresentada uma prova, no mesmo formato mostrado nos conteúdos de aula e nas práticas.

#### Dicas:

1. Transformar as afirmações para lógica:

$p$ : as uvas caem

$q$ : a raposa come as uvas

$r$ : as uvas estão maduras

2. Transformar as três primeiras sentenças para formar a base de conhecimento

R1:  $p \rightarrow q$

R2:  $q \rightarrow r$

R3:  $\neg r \vee p$

3. Aplicar equivalências e regras de inferência para se obter o resultado esperado. Isto é, com essas três primeiras sentenças devemos derivar  $q \leftrightarrow p$ . Cuidado com a ordem em que as fórmulas são geradas.

**Equivalência Implicação:**  $(\alpha \rightarrow \beta)$  equivale a  $(\neg \alpha \vee \beta)$

**Silogismo Hipotético:**  $\alpha \rightarrow \beta, \beta \rightarrow \gamma \vdash \alpha \rightarrow \gamma$

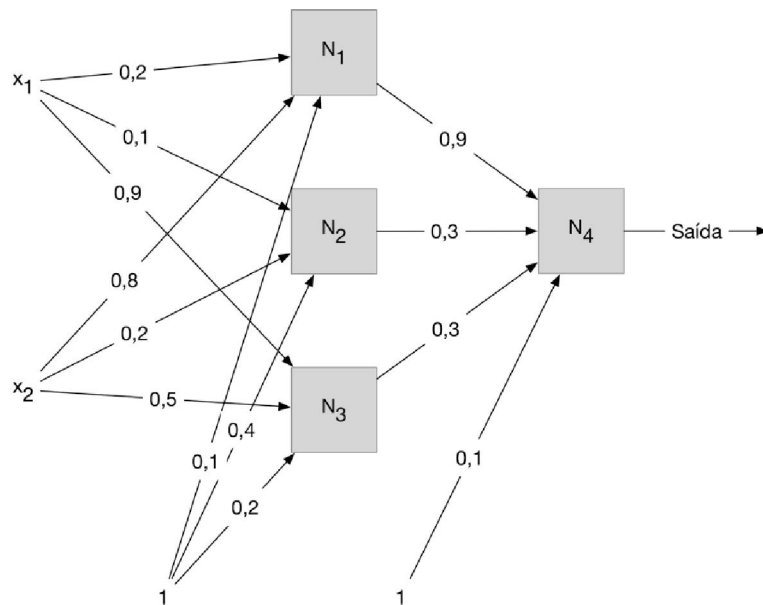
**Conjunção:**  $\alpha, \beta \vdash \alpha \wedge \beta$

**Equivalência Bicondicional:**  $(\alpha \leftrightarrow \beta)$  equivale a  $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$

- a) **(25 pontos)** Deve-se mostrar todos os passos e regras aplicadas, **no mesmo formato apresentado nas aulas e nas práticas**. As equivalências e regras necessárias estão descritas acima e no material.

#### 4 Redes Neurais Artificiais

Seja a RNA da figura abaixo.



Os neurônios  $N_1$ ,  $N_2$  e  $N_3$  possuem função de ativação linear. Já  $N_4$  possui função de ativação tangente hiperbólica (pesquise a fórmula e aplique).

Dada a entrada  $x_1 = -3$ ,  $x_2 = 1$ , dê:

- (6,25 pontos)** Valor de saída do neurônio  $N_1$
- (6,25 pontos)** Valor de saída do neurônio  $N_2$
- (6,25 pontos)** Valor de saída do neurônio  $N_3$
- (6,25 pontos)** Valor de saída da rede como um todo

## B – RESOLUÇÃO

### 1. ChatGPT

#### a)

Inteligência Artificial Inteligência artificial (IA) é um campo da ciência da computação que se concentra no desenvolvimento de sistemas e máquinas capazes de realizar tarefas que normalmente exigiriam inteligência humana. Isso pode incluir aprendizado, raciocínio, resolução de problemas, reconhecimento de padrões, tomada de decisões e muito mais. A IA tenta simular a inteligência humana através de algoritmos e modelos de dados. Existem diferentes abordagens para a IA, incluindo: 1. **IA Fraca**: Também conhecida como inteligência artificial estreita, é a IA especializada em tarefas específicas, como reconhecimento de voz, reconhecimento facial, diagnóstico médico, jogos de xadrez, entre outros. A IA fraca não possui autonomia ou consciência. 2. **IA Forte**: Também chamada de inteligência artificial geral ou verdadeira IA, é uma IA que possuiria a capacidade de realizar qualquer tarefa intelectual que um ser humano possa realizar. Esta é uma área de pesquisa teórica e não está atualmente presente em sistemas práticos. 3. **Aprendizado de Máquina**: Um subcampo da IA que se concentra em construir sistemas que podem aprender com dados. Esses sistemas podem fazer previsões ou tomar decisões sem serem explicitamente programados para fazer isso. 4. **Redes Neurais Artificiais (RNAs)**: Um modelo computacional inspirado no sistema nervoso humano que é usado em muitas aplicações de aprendizado de máquina. As RNAs são compostas por muitos neurônios artificiais interconectados que processam informações. 5. **Processamento de Linguagem Natural (NLP)**: Um campo da IA que permite que os computadores entendam, interpretem e gerem linguagem humana de maneira natural. A IA está se tornando cada vez mais presente em nossas vidas, com aplicativos em áreas como assistentes pessoais, automóveis autônomos, diagnóstico médico, jogos e muito mais. No entanto, também levanta questões éticas, como a privacidade dos dados e o potencial de deslocamento de empregos.

#### b)

A resposta dada pelo ChatGPT pode ser classificada dentro da abordagem “Agir racionalmente”. Dentro desta abordagem existem agentes que avaliam as situações e tentam reagir da melhor forma possível. E podemos ver isso na resposta dada, pois além de uma breve descrição do que é a inteligência artificial ele também responde diferentes abordagens da IA, buscando dar a melhor resposta possível para a pergunta (situação). Não poderíamos classificar o ChatGPT dentro das teorias “Pensar como humano” e “Agir como humano” pois ele não tenta se encaixar dentro da forma de agir ou pensar humana, mas sim ele se adequa para a dar a resposta mais correta e racional dentro do contexto. Quanto à abordagem “Pensar racionalmente” acredito que o ChatGPT vá um pouco além dessa abordagem, pois ele não fica preso a racionalidade perfeita e não há necessidade de enunciar problemas de forma perfeita, ele se adapta com a situação.

c)

ChatGPT é um sistema criado pela empresa OpenIA com o objetivo de ser um chatbot que mantém um diálogo em uma linguagem natural, e que cria imagens e vídeos baseadas nos textos. Essa ferramenta também utiliza as interações com os usuários para aprimorar a suas respostas e ficando cada vez mais precisa com o passar do tempo. Ele cria as respostas baseada na modelagem de linguagem. No treinamento desta inteligência artificial foram utilizados uma enorme base de dados contendo textos sobre diversas áreas de conhecimento. E por meio da fase de aprendizado, a ferramenta foi capaz de identificar padrões e estruturas da linguagem, assim como a relação entre palavras, frases e contextos. Ao interagir com o ChatGPT, ele analisa o texto e tenta produzir uma resposta coerente levando em consideração as informações que ele tem sobre o assunto perguntado. Isso permite que ele responda de forma natural e relevante para o contexto. No entanto, como qualquer IA em desenvolvimento, ele ainda pode cometer erros e na verdade o aprendizado também inclui os erros, para que ele possa refinar os parâmetros que são usados.

Referências bibliográficas:

<https://openai.com/chatgpt>

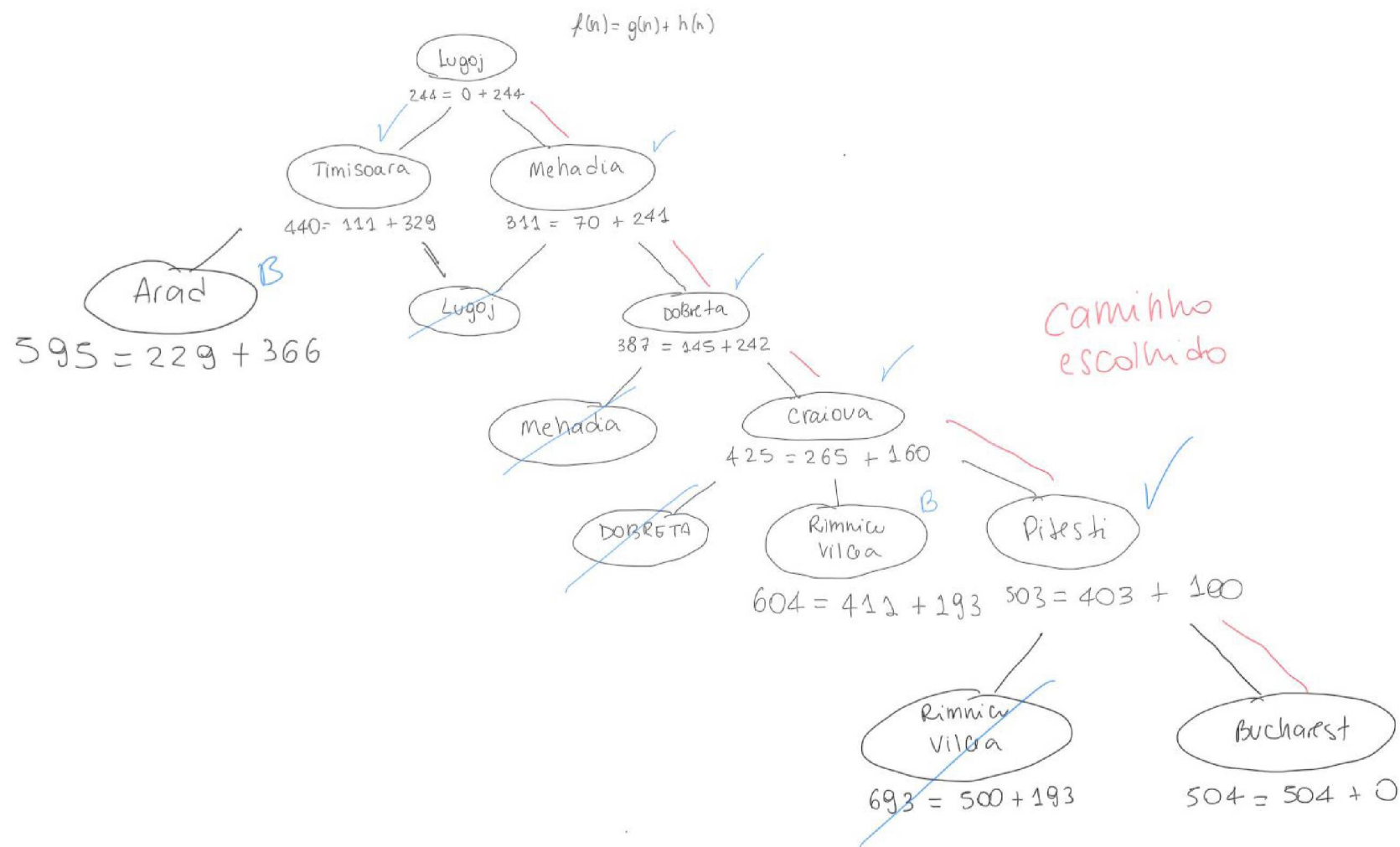
<https://br.hubspot.com/blog/marketing/chatgpt>

d)

O ChatGPT também pode ser classificado como “Agir Racionalmente”. Pois ele é um agente que atua de forma autônoma, se adequando a situação em que se encontra (as perguntas e interações dos usuários). Ele é capaz de adaptar sua resposta de acordo com o contexto e dar a resposta da forma mais natural de acordo com a linguagem do usuário. Similarmente a classificação da resposta dada pela ferramenta, ele não se encaixaria em um contexto de “Pensar Racionalmente” pois é capaz de se adaptar e não precisa de uma elaboração detalhada do problema. Essa ferramenta também não se encaixaria na abordagem “Agir como Humanos” pois ela ainda não passa em um teste de Turing, embora já obtenha resultados mais satisfatórios que a maioria dos chatbots de hoje em dia. A abordagem “Pensar como humanos”, pois ele não visa ter a forma de pensar de um humano, mas sim responder adequadamente às solicitações feitas.

## **2. Busca Heurística**

a)



### 3. Lógica

#### a) Transformar as afirmações para lógica

$p$  = as uvas caem  
 $q$  = a raposa come as uvas  
 $r$  = as uvas estão maduras

Se as uvas caem, então a raposa as come  
 $p \rightarrow q$

Se a raposa as come, então estão maduras  
 $q \rightarrow r$

As uvas estão verdes ou caem  
 $\neg r \vee p$

b)  $R1: p \rightarrow q$   
 $R2: q \rightarrow r$   
 $R3: \neg r \vee p$

c)

|                                                  |                                  |
|--------------------------------------------------|----------------------------------|
| $R4: r \rightarrow p$                            | Equivalência Implicação em R3    |
| $R5: q \rightarrow p$                            | Silogismo Hipotético em R2 e R4  |
| $R6: (p \rightarrow q) \wedge (q \rightarrow p)$ | Conjunção em R1 e R5             |
| $R7: (q \rightarrow p) \wedge (p \rightarrow q)$ | Comutatividade em R6             |
| $R8: q \leftrightarrow p$                        | Equivalência bicondicional em R7 |

### 4. Redes Neurais Artificiais

Dados  $X1 = -3$  e  $X2 = 1$

- a) Valor de saída do neurônio N1  
 $u = (-3 * 0,2) + (1 * 0,8) + (1 * 0,1) = 0,3$   
 $f(u) = 0,3$
- b) Valor de saída do neurônio N2  
 $u = (-3 * 0,1) + (1 * 0,2) + (1 * 0,4) = 0,3$   
 $f(u) = 0,3$
- c) Valor de saída do neurônio N3  
 $u = (-3 * 0,9) + (1 * 0,5) + (1 * 0,2) = -2$   
 $f(u) = -2$
- d) Valor de saída da rede como um todo  
 $u = (0,3 * 0,9) + (0,3 * 0,3) + (-2 * 0,3) + (1 * 0,1) = -0,14$

$$\tanh(u) = -0,1390$$

$$N1 = (0,3 * 0,9) = 0,27$$

$$N2 = (0,3 * 0,3) = 0,09$$

$$N3 = (-2 * 0,3) = -0,6$$

$$\text{Bias} = (1 * 0,1) = 0,1$$

$$\text{Somatório } N4 = 0,27 + 0,09 + (-0,6) + 0,1$$

$$\text{Somatório } N4 = 0,37 + 0,09 + (-0,6)$$

$$\text{Somatório } N4 = 0,46 + (-0,6)$$

$$\text{Somatório } N4 = -0,14$$

$$\tanh(u) = -0,1390....$$

## APÊNDICE 2 – LINGUAGEM DE PROGRAMAÇÃO APLICADA

### A – ENUNCIADO

**Nome da base de dados do exercício:** *precos\_carros\_brasil.csv*

**Informações sobre a base de dados:**

Dados dos preços médios dos carros brasileiros, das mais diversas marcas, no ano de 2021, de acordo com dados extraídos da tabela FIPE (Fundação Instituto de Pesquisas Econômicas). A base original foi extraída do site Kaggle ([Acesse aqui a base original](#)). A mesma foi adaptada para ser utilizada no presente exercício.

Observação: As variáveis *fuel*, *gear* e *engine\_size* foram extraídas dos valores da coluna *model*, pois na base de dados original não há coluna dedicada a esses valores. Como alguns valores do modelo não contêm as informações do tamanho do motor, este conjunto de dados não contém todos os dados originais da tabela FIPE.

**Metadados:**

| Nome do campo      | Descrição                                                                                         |
|--------------------|---------------------------------------------------------------------------------------------------|
| year_of_reference  | O preço médio corresponde a um mês de ano de referência                                           |
| month_of_reference | O preço médio corresponde a um mês de referência, ou seja, a FIPE atualiza sua tabela mensalmente |
| fipe_code          | Código único da FIPE                                                                              |
| authentication     | Código de autenticação único para consulta no site da FIPE                                        |
| brand              | Marca do carro                                                                                    |
| model              | Modelo do carro                                                                                   |
| fuel               | Tipo de combustível do carro                                                                      |
| gear               | Tipo de engrenagem do carro                                                                       |
| engine_size        | Tamanho do motor em centímetros cúbicos                                                           |



|            |                                                                    |
|------------|--------------------------------------------------------------------|
| year_model | Ano do modelo do carro. Pode não corresponder ao ano de fabricação |
| avg_price  | Preço médio do carro, em reais                                     |

**Atenção:** ao fazer o download da base de dados, selecione o formato .csv. É o formato que será considerado correto na resolução do exercício.

## 1 Análise Exploratória dos dados

A partir da base de dados **precos\_carros\_brasil.csv**, execute as seguintes tarefas:

- Carregue a base de dados media\_precos\_carros\_brasil.csv
- Verifique se há valores faltantes nos dados. Caso haja, escolha uma tratativa para resolver o problema de valores faltantes
- Verifique se há dados duplicados nos dados
- Crie duas categorias, para separar colunas numéricas e categóricas. Imprima o resumo de informações das variáveis numéricas e categóricas (estatística descritiva dos dados)
- Imprima a contagem de valores por modelo (model) e marca do carro (brand)
- Dê um breve explicação (máximo de quatro linhas) sobre os principais resultados encontrados na Análise Exploratória dos dados

## 2 Visualização dos dados

A partir da base de dados **precos\_carros\_brasil.csv**, execute as seguintes tarefas:

- Gere um gráfico da distribuição da quantidade de carros por marca
- Gere um gráfico da distribuição da quantidade de carros por tipo de engrenagem do carro
- Gere um gráfico da evolução da média de preço dos carros ao longo dos meses de 2022 (variável de tempo no eixo X)
- Gere um gráfico da distribuição da média de preço dos carros por marca e tipo de engrenagem
- Dê uma breve explicação (máximo de quatro linhas) sobre os resultados gerados no item d
- Gere um gráfico da distribuição da média de preço dos carros por marca e tipo de combustível
- Dê uma breve explicação (máximo de quatro linhas) sobre os resultados gerados no item f

## 3 Aplicação de modelos de machine learning para prever o preço médio dos carros

A partir da base de dados **precos\_carros\_brasil.csv**, execute as seguintes tarefas:

- Escolha as variáveis **numéricas** (modelos de Regressão) para serem as variáveis independentes do modelo. A variável target é **avg\_price**. **Observação:** caso julgue necessário, faça a transformação de variáveis categóricas em variáveis numéricas para inputar no modelo. Indique **quais variáveis** foram transformadas e **como** foram transformadas
- Crie partições contendo 75% dos dados para treino e 25% para teste
- Treine modelos RandomForest (biblioteca RandomForestRegressor) e XGBoost (biblioteca XGBRegressor) para predição dos preços dos carros. **Observação:** caso julgue necessário, mude os parâmetros dos modelos e rode novos modelos. Indique quais parâmetros foram inputados e indique o treinamento de cada modelo
- Grave os valores preditos em variáveis criadas

- e. Realize a análise de importância das variáveis para estimar a variável target, **para cada modelo treinado**
- f. Dê uma breve explicação (máximo de quatro linhas) sobre os resultados encontrados na análise de importância de variáveis
- g. Escolha o melhor modelo com base nas métricas de avaliação MSE, MAE e R<sup>2</sup>
- h. Dê uma breve explicação (máximo de quatro linhas) sobre qual modelo gerou o melhor resultado e a métrica de avaliação utilizada

## B - RESOLUÇÃO

1. a)

```
import pandas as pd
```

```
import warnings
warnings.filterwarnings("ignore")
```

```
df = pd.read_csv('precos_carros_brasil.csv', low_memory=False)
```

b) Existem 65245 linhas no final do arquivo com valores nulos para todas as colunas

```
# Verificando se existem valores faltantes nos dados
df.isna().any()
df.isna().sum()
```

```
year_of_reference      65245
month_of_reference     65245
fipecode               65245
authentication         65245
brand                  65245
model                  65245
fuel                   65245
gear                   65245
engine_size            65245
year_model             65245
avg_price_brl          65245
dtype: int64
```

```
# Substituir os valores faltantes
df['year_of_reference'].fillna('Não informado', inplace=True)
df['month_of_reference'].fillna('Não informado', inplace=True)
df['fipecode'].fillna('Não informado', inplace=True)
df['authentication'].fillna('Não informado', inplace=True)
df['brand'].fillna('Não informado', inplace=True)
df['model'].fillna('Não informado', inplace=True)
df['fuel'].fillna('Não informado', inplace=True)
df['gear'].fillna('Não informado', inplace=True)
df['engine_size'].fillna('Não informado', inplace=True)
df['year_model'].fillna('Não informado', inplace=True)
df['avg_price_brl'].fillna('Não informado', inplace=True)
```

c)

```
# Verificando se temos valores duplicados
df.duplicated().sum()
```

```
65246
```

```
# Converter e limpar dados conforme necessário
```

```
df['engine_size'] = df['engine_size'].apply(lambda x:
str(x).replace(',', '.'))
for col in ['year_of_reference', 'year_model', 'avg_price_brl',
'engine_size']:
    df[col] = pd.to_numeric(df[col], errors='coerce')
df_cleaned = df.dropna().drop_duplicates()
```

d)

```
# Separar colunas numéricas e categóricas e gerar estatísticas
descritivas
numeric_cols = df_cleaned.select_dtypes(include=['number']).columns
categorical_cols = df_cleaned.select_dtypes(include=['object']).columns
numeric_summary = df_cleaned[numeric_cols].describe()
categorical_summary = df_cleaned[categorical_cols].describe()
print(numeric_summary)
print(categorical_summary)
```

|       | year_of_reference | engine_size   | year_model    | avg_price_brl |
|-------|-------------------|---------------|---------------|---------------|
| count | 202295.000000     | 202295.000000 | 202295.000000 | 202295.000000 |
| mean  | 2021.564695       | 1.822302      | 2011.271514   | 52756.765713  |
| std   | 0.571904          | 0.734432      | 6.376241      | 51628.912116  |
| min   | 2021.000000       | 1.000000      | 2000.000000   | 6647.000000   |
| 25%   | 2021.000000       | 1.400000      | 2006.000000   | 22855.000000  |
| 50%   | 2022.000000       | 1.600000      | 2012.000000   | 38027.000000  |
| 75%   | 2022.000000       | 2.000000      | 2016.000000   | 64064.000000  |
| max   | 2023.000000       | 6.200000      | 2023.000000   | 979358.000000 |

|        | month_of_reference | fipe_code | authentication | brand \ |
|--------|--------------------|-----------|----------------|---------|
| count  | 202295             | 202295    | 202295         | 202295  |
| unique | 12                 | 2091      | 202295         | 6       |
| top    | January            | 003281-6  | cfzlcztzfwrcp  | Fiat    |
| freq   | 24260              | 425       | 1              | 44962   |

|        | model                                  | fuel     | gear   |
|--------|----------------------------------------|----------|--------|
| count  | 202295                                 | 202295   | 202295 |
| unique | 2112                                   | 3        | 2      |
| top    | Palio Week. Adv/Adv TRYON 1.8 mpi Flex | Gasoline | manual |
| freq   | 425                                    | 168684   | 161883 |

e)

```
##Contagem de valores por modelo e marca
model_count = df_cleaned['model'].value_counts().head(10) # Top 10
modelos
brand_count = df_cleaned['brand'].value_counts().head(10) # Top 10
marcas

print(model_count)
print("#####")
print(brand_count)
```

| model                                  |     |
|----------------------------------------|-----|
| Palio Week. Adv/Adv TRYON 1.8 mpi Flex | 425 |
| Focus 1.6 S/SE/SE Plus Flex 8V/16V 5p  | 425 |
| Focus 2.0 16V/SE/SE Plus Flex 5p Aut.  | 400 |
| Saveiro 1.6 Mi/ 1.6 Mi Total Flex 8V   | 400 |
| Corvette 5.7/ 6.0, 6.2 Targa/Stingray  | 375 |
| Golf 2.0/ 2.0 Mi Flex Aut/Tiptronic.   | 375 |

```

Doblo Adv/Adv TRYON/LOCKER 1.8 Flex      375
Kombi Escolar 1.6 MPi                     350
Courier 1.6 L/ 1.6 Flex                   350
Courier XL/XL-RS 1.6/ XL 1.6 Flex         350
Name: count, dtype: int64
#####
brand
Fiat      44962
VW - Volkswagen  44312
GM - Chevrolet  38590
Ford       33150
Renault    29191
Nissan     12090
Name: count, dtype: int64

```

f) Os modelos mais comuns incluem o "Palio Week. Adv/Adv TRYON 1.8 mpi Flex" e o "Focus 1.6 S/SE/SE Plus Flex 8V/16V 5p", cada um aparecendo 425 vezes.

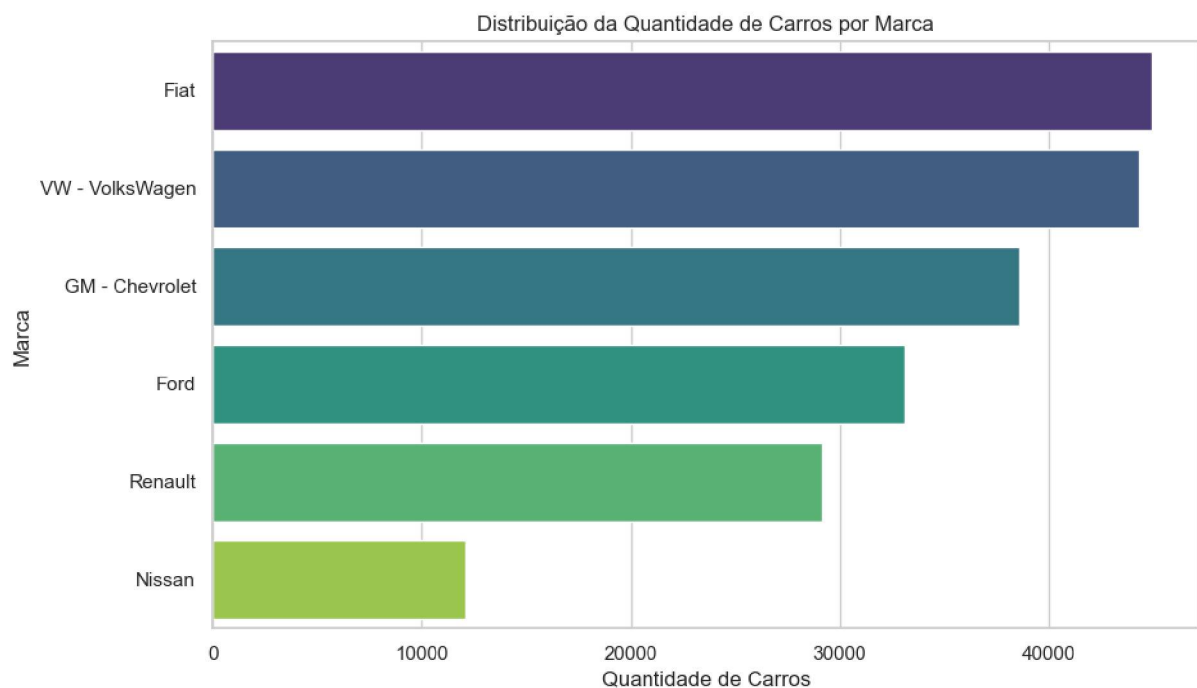
As marcas mais representadas são Fiat, VW - Volkswagen, GM - Chevrolet, e Ford, com a Fiat tendo o maior número de registros.

2. a)

```

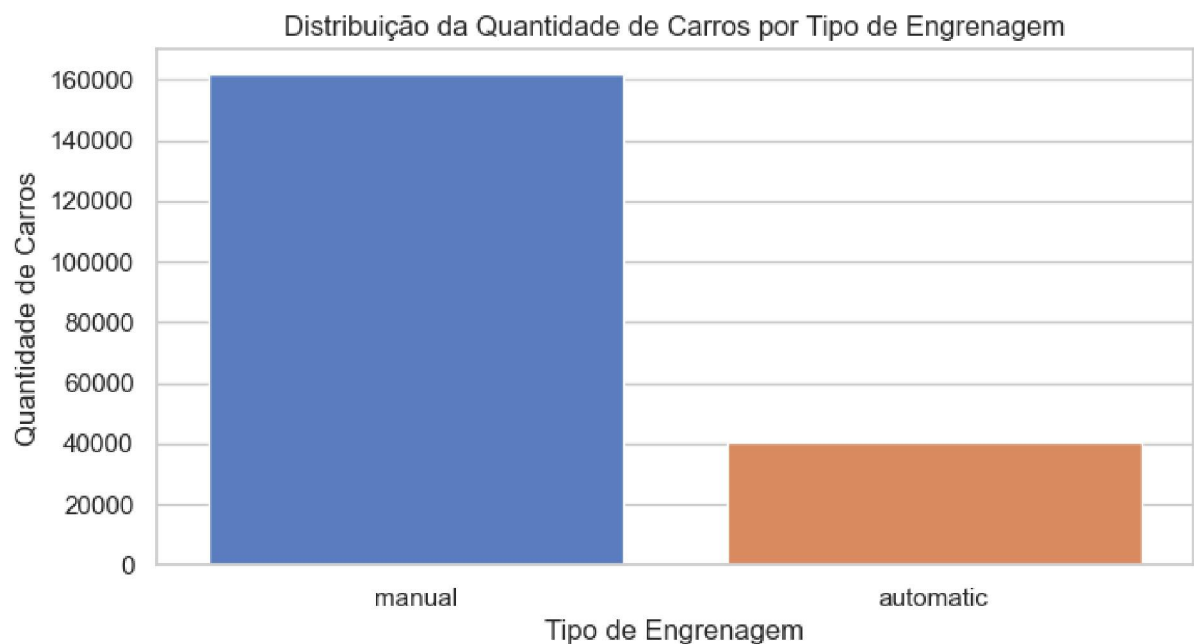
# Distribuição da quantidade de carros por marca
plt.figure(figsize=(10, 6))
brand_counts = df_cleaned['brand'].value_counts()
sns.barplot(x=brand_counts.values, y=brand_counts.index,
            palette="viridis")
plt.title('Distribuição da Quantidade de Carros por Marca')
plt.xlabel('Quantidade de Carros')
plt.ylabel('Marca')
plt.show()

```



b)

```
#Distribuição da Quantidade de Carros por Tipo de Engrenagem do Carro
plt.figure(figsize=(8, 4))
gear_counts = df_cleaned['gear'].value_counts()
sns.barplot(x=gear_counts.index, y=gear_counts.values, palette="muted")
plt.title('Distribuição da Quantidade de Carros por Tipo de Engrenagem')
plt.xlabel('Tipo de Engrenagem')
plt.ylabel('Quantidade de Carros')
plt.show()
```



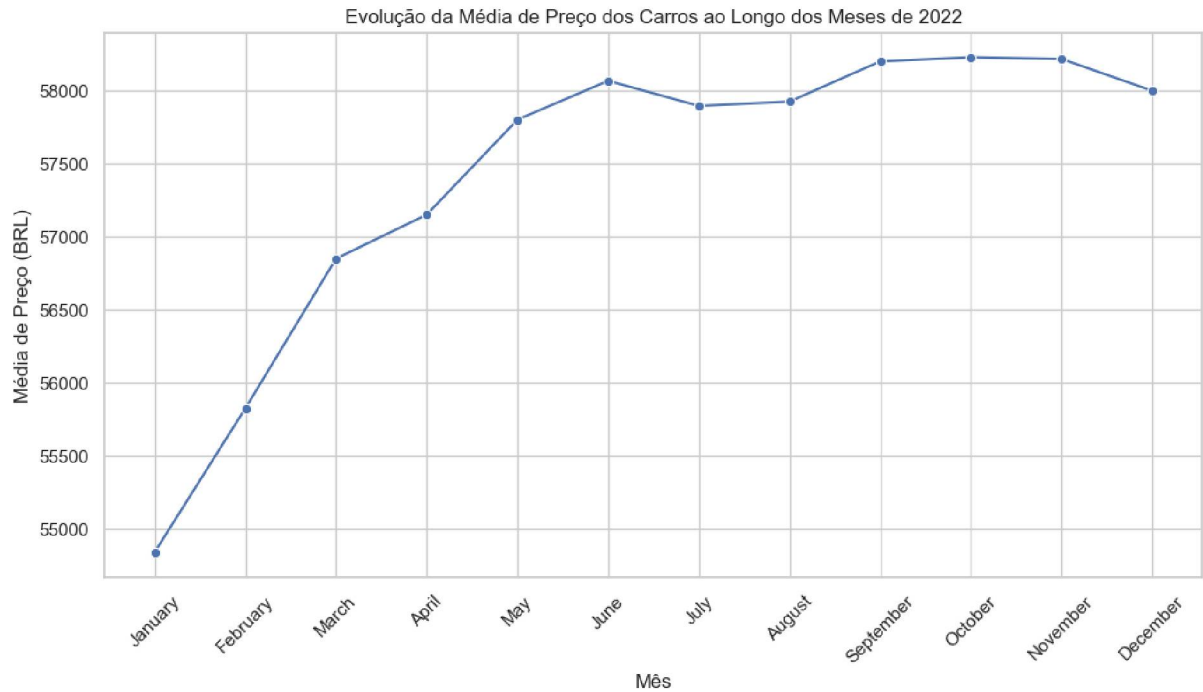
c)

```
# Filtrando dados para 2022
df_2022 = df_cleaned[df_cleaned['year_of_reference'] == 2022]
# Convertendo 'month_of_reference' para tipo categórico com ordenação dos meses
months_order = ["January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"]
df_2022['month_of_reference'] = pd.Categorical(df_2022['month_of_reference'], categories=months_order, ordered=True)

# Agrupando por mês e calculando a média do preço
monthly_avg_price = df_2022.groupby('month_of_reference')['avg_price_brl'].mean().reset_index()

# Gráfico da evolução da média de preço ao longo dos meses de 2022
plt.figure(figsize=(12, 6))
sns.lineplot(x='month_of_reference', y='avg_price_brl', data=monthly_avg_price, marker='o')
```

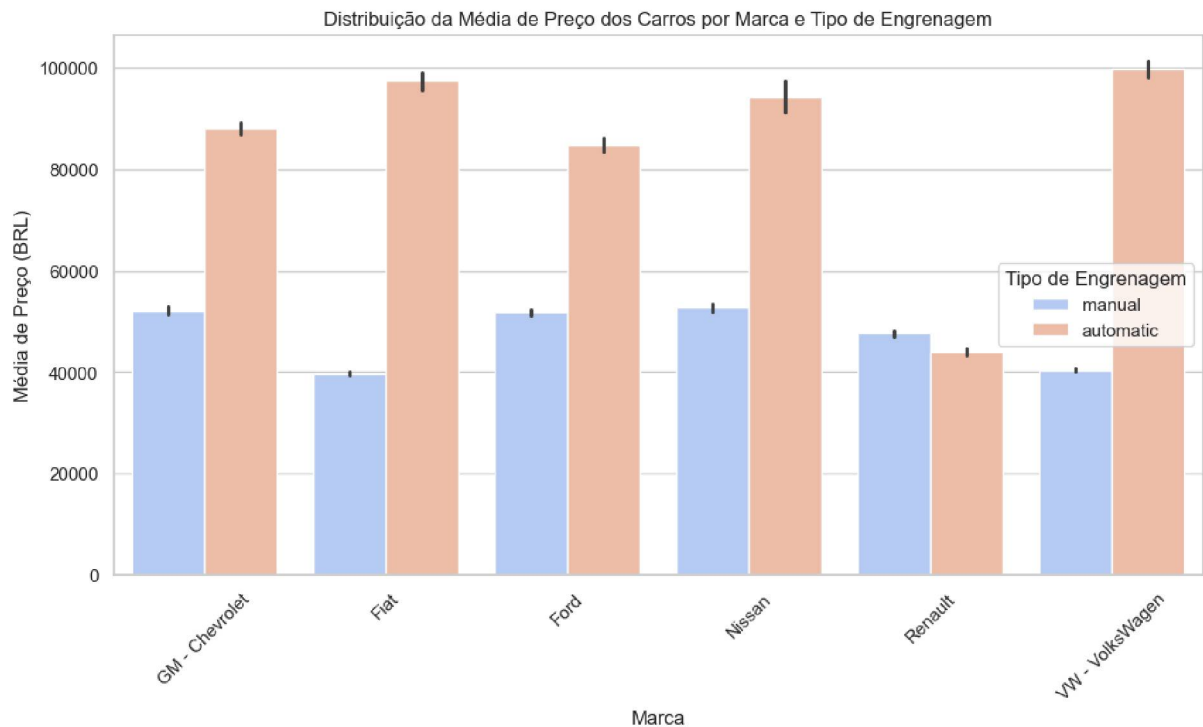
```
plt.title('Evolução da Média de Preço dos Carros ao Longo dos Meses de 2022')
plt.xlabel('Mês')
plt.ylabel('Média de Preço (BRL)')
plt.xticks(rotation=45)
plt.show()
```



d)

```
# Distribuição da Média de Preço dos Carros por Marca e Tipo de Engrenagem
plt.figure(figsize=(12, 6))
sns.barplot(x='brand', y='avg_price_brl', hue='gear', data=df_cleaned,
palette="coolwarm")
plt.title('Distribuição da Média de Preço dos Carros por Marca e Tipo de Engrenagem')
plt.xlabel('Marca')
plt.ylabel('Média de Preço (BRL)')
plt.xticks(rotation=45)
plt.legend(title='Tipo de Engrenagem')
plt.show()
```

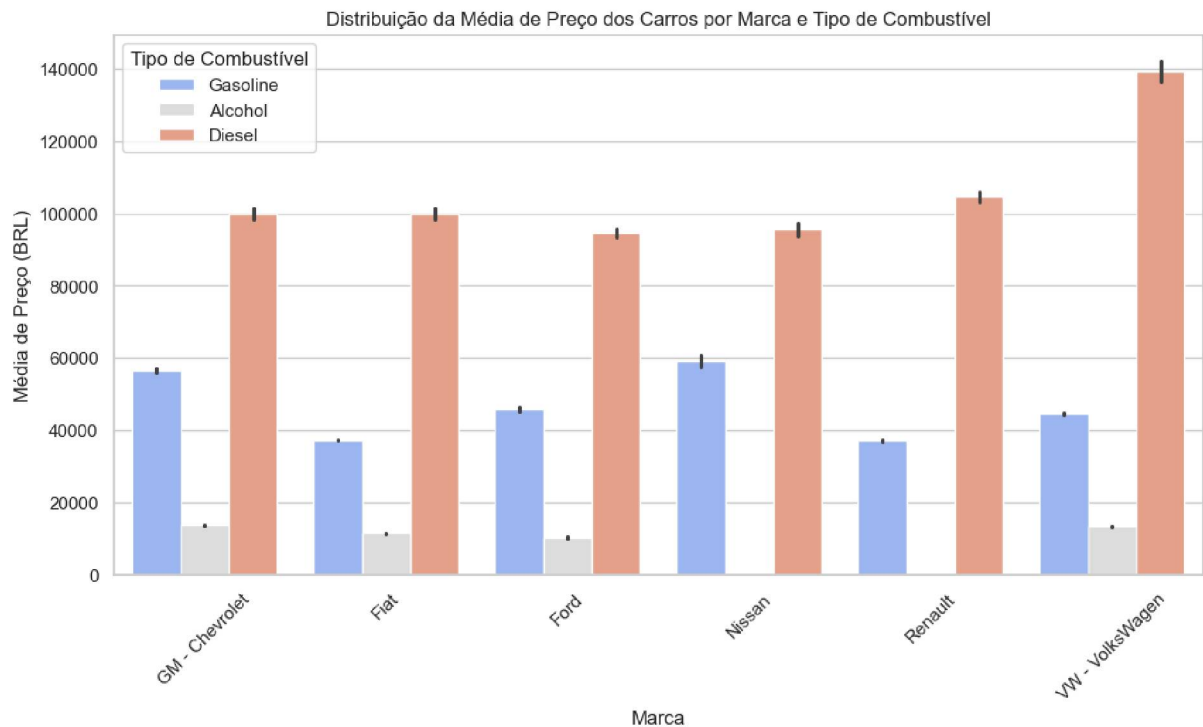




e) Carros com engrenagem automática têm uma média de preço mais elevada em comparação com carros de engrenagem manual para todas as marcas. A diferença é consistente através das marcas, sugerindo que a transmissão automática é um fator que contribui para o custo mais alto dos veículos, independente da marca.

f)

```
#Distribuição da Média de Preço dos Carros por Marca e Tipo de Combustível
plt.figure(figsize=(12, 6))
sns.barplot(x='brand', y='avg_price_brl', hue='fuel', data=df_cleaned,
palette="coolwarm")
plt.title('Distribuição da Média de Preço dos Carros por Marca e Tipo de Combustível')
plt.xlabel('Marca')
plt.ylabel('Média de Preço (BRL)')
plt.xticks(rotation=45)
plt.legend(title='Tipo de Combustível')
plt.show()
```



g) Independentemente do tipo de combustível, a VW - Volkswagen e a GM - Chevrolet apresentam as médias de preços mais elevadas.

Por outro lado, a Ford, Fiat e Nissan mostram médias de preços menores para os três tipos de combustível.

Isso pode indicar uma estratégia de posicionamento de mercado dessas marcas ou refletir diferentes categorias de veículos oferecidos por elas.

3. a)

```
# Convertendo a variável 'engine_size' para numérica
dados['engine_size'] = dados['engine_size'].str.replace(',', '.').astype(float)

# Codificando variáveis categóricas e convertendo para tipo 'int'
categorical_vars = ['month_of_reference', 'fuel', 'gear']
for col in categorical_vars:
    dados[col] = LabelEncoder().fit_transform(dados[col].astype(str)).astype(int)

# Selecionando as variáveis independentes e a variável target
X = dados[['year_of_reference', 'month_of_reference', 'fuel', 'gear', 'engine_size', 'year_model']]
y = dados['avg_price_brl']

# Removendo linhas com valores nulos
X = X.dropna()
y = y.loc[X.index]
```

b)

```
# Dividindo os dados em conjuntos de treino e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
```



c)

```
# Inicializando os modelos
rf = RandomForestRegressor(random_state=42)
xgb = XGBRegressor(random_state=42, n_estimators=100, max_depth=3)
# Treinando os modelos
rf.fit(X_train, y_train)
xgb.fit(X_train, y_train)
```

d)

```
# Fazendo previsões
y_pred_rf = rf.predict(X_test)
y_pred_xgb = xgb.predict(X_test)
```

e)

```
# Avaliando os modelos
metrics_rf = {
    "MSE": mean_squared_error(y_test, y_pred_rf),
    "MAE": mean_absolute_error(y_test, y_pred_rf),
    "R^2": r2_score(y_test, y_pred_rf)
}

metrics_xgb = {
    "MSE": mean_squared_error(y_test, y_pred_xgb),
    "MAE": mean_absolute_error(y_test, y_pred_xgb),
    "R^2": r2_score(y_test, y_pred_xgb)
}

print("RandomForest Metrics:", metrics_rf)
print("XGBoost Metrics:", metrics_xgb)
```

```
RandomForest Metrics: {'MSE': 214511591.53775933, 'MAE': 7272.201350817884,
'R^2': 0.9191415741559127}
XGBoost Metrics: {'MSE': 185360830.5689201, 'MAE': 7621.348730526677, 'R^2':
0.9301297199582}
```

```
# Analisando a importância das variáveis
importances_rf = rf.feature_importances_
importances_xgb = xgb.feature_importances_

print("Importância das variáveis no RandomForest:", importances_rf)
print("Importância das variáveis no XGBoost:", importances_xgb)
```

```
Importância das variáveis no RandomForest: [0.01433986 0.01371423
0.0349384 0.04101832 0.48478858 0.4112006 ]
Importância das variáveis no XGBoost: [0.03193877 0.00563701 0.13987061
0.16708949 0.30171058 0.3537536 ]
```

f) A análise de importância das variáveis revelou que, para ambos os modelos RandomForest e XGBoost, o tamanho do motor (engine\_size) e o ano do modelo (year\_model) são as variáveis mais influentes na previsão do preço médio dos carros. Isso sugere que características intrínsecas do veículo, como sua capacidade e modernidade, desempenham papéis cruciais na determinação de seu valor no mercado. Enquanto o RandomForest dá uma ênfase um pouco maior ao engine\_size, o XGBoost distribui a importância de forma mais balanceada entre outras variáveis, como gear e fuel, indicando uma abordagem mais holística na avaliação dos fatores que influenciam o preço.

g) Com base nas métricas de avaliação analisadas, o modelo XGBoost é identificado como o melhor modelo para a previsão do preço médio dos carros.

Apesar de o XGBoost apresentar um MAE ligeiramente superior ao do RandomForest, ele alcançou um menor MSE e um maior coeficiente  $R^2$ .

Especificamente, o  $R^2$  do XGBoost é 0.9301, indicando que cerca de 93% da variância nos preços dos carros é explicada pelo modelo,

o que é superior ao  $R^2$  de 0.9191 alcançado pelo RandomForest. Além disso, o menor MSE indica que as previsões do XGBoost estão mais próximas dos valores reais em comparação com as do RandomForest.

h). O modelo XGBoost gerou o melhor resultado na tarefa de prever o preço médio dos carros. A decisão baseia-se principalmente no coeficiente de determinação  $R^2$  e no Erro Quadrático Médio (MSE). O XGBoost apresentou um  $R^2$  de 0.9301, sugerindo uma alta capacidade do modelo de explicar a variância observada nos preços dos carros, além de um MSE menor, indicando previsões mais precisas em comparação com o RandomForest.

Essas métricas demonstram a eficácia do XGBoost em capturar as relações complexas dentro dos dados e em fornecer previsões acuradas sobre o preço dos veículos."

## APÊNDICE 3 – LINGUAGEM R

### A – ENUNCIADO

#### 1 Pesquisa com Dados de Satélite (Satellite)

O banco de dados consiste nos valores multiespectrais de pixels em vizinhanças 3x3 em uma imagem de satélite, e na classificação associada ao pixel central em cada vizinhança. O objetivo é prever esta classificação, dados os valores multiespectrais.

Um quadro de imagens do Satélite Landsat com MSS (*Multispectral Scanner System*) consiste em quatro imagens digitais da mesma cena em diferentes bandas espectrais. Duas delas estão na região visível (correspondendo aproximadamente às regiões verde e vermelha do espectro visível) e duas no infravermelho (próximo). Cada pixel é uma palavra binária de 8 bits, com 0 correspondendo a preto e 255 a branco. A resolução espacial de um pixel é de cerca de 80m x 80m. Cada imagem contém 2340 x 3380 desses pixels. O banco de dados é uma subárea (minúscula) de uma cena, consistindo de 82 x 100 pixels. Cada linha de dados corresponde a uma vizinhança quadrada de pixels 3x3 completamente contida dentro da subárea 82x100. Cada linha contém os valores de pixel nas quatro bandas espectrais (convertidas em ASCII) de cada um dos 9 pixels na vizinhança de 3x3 e um número indicando o rótulo de classificação do pixel central.

As classes são: solo vermelho, colheita de algodão, solo cinza, solo cinza úmido, restolho de vegetação, solo cinza muito úmido.

Os dados estão em ordem aleatória e certas linhas de dados foram removidas, portanto você não pode reconstruir a imagem original desse conjunto de dados. Em cada linha de dados, os quatro valores espectrais para o pixel superior esquerdo são dados primeiro, seguidos pelos quatro valores espectrais para o pixel superior central e, em seguida, para o pixel superior direito, e assim por diante, com os pixels lidos em sequência, da esquerda para a direita e de cima para baixo. Assim, os quatro valores espectrais para o pixel central são dados pelos atributos 17, 18, 19 e 20. Se você quiser, pode usar apenas esses quatro atributos, ignorando os outros. Isso evita o problema que surge quando uma vizinhança 3x3 atravessa um limite.

O banco de dados se encontra no pacote **mlbench** e é completo (não possui dados faltantes).

Tarefas:

1. Carregue a base de dados Satellite
2. Crie partições contendo 80% para treino e 20% para teste
3. Treine modelos RandomForest, SVM e RNA para predição destes dados.
4. Escolha o melhor modelo com base em suas matrizes de confusão.
5. Indique qual modelo dá o melhor o resultado e a métrica utilizada

## 2 Estimativa de Volumes de Árvores

Modelos de aprendizado de máquina são bastante usados na área da engenharia florestal (mensuração florestal) para, por exemplo, estimar o volume de madeira de árvores sem ser necessário abatê-las.

O processo é feito pela coleta de dados (dados observados) através do abate de algumas árvores, onde sua altura, diâmetro na altura do peito (dap), etc, são medidos de forma exata. Com estes dados, treina-se um modelo de AM que pode estimar o volume de outras árvores da população.

Os modelos, chamados alométricos, são usados na área há muitos anos e são baseados em regressão (linear ou não) para encontrar uma equação que descreve os dados. Por exemplo, o modelo de Spurr é dado por:

$$\text{Volume} = b_0 + b_1 * \text{dap}^2 * H_t$$

Onde dap é o diâmetro na altura do peito (1,3metros), Ht é a altura total. Tem-se vários modelos alométricos, cada um com uma determinada característica, parâmetros, etc. Um modelo de regressão envolve aplicar os dados observados e encontrar b0 e b1 no modelo apresentado, gerando assim uma equação que pode ser usada para prever o volume de outras árvores.

Dado o arquivo **Volumes.csv**, que contém os dados de observação, escolha um modelo de aprendizado de máquina com a melhor estimativa, a partir da estatística de correlação.

### Tarefas

1. Carregar o arquivo Volumes.csv (<http://www.razer.net.br/datasets/Volumes.csv>)
2. Eliminar a coluna NR, que só apresenta um número sequencial
3. Criar partição de dados: treinamento 80%, teste 20%
4. Usando o pacote "caret", treinar os modelos: Random Forest (rf), SVM (svmRadial), Redes Neurais (neuralnet) e o modelo alométrico de SPURR

- O modelo alométrico é dado por:  $\text{Volume} = b_0 + b_1 * \text{dap}^2 * H_t$

```
alom <- nls(VOL ~ b0 + b1*DAP*DAP*HT, dados, start=list(b0=0.5, b1=0.5))
```

5. Efetue as predições nos dados de teste
6. Crie suas próprias funções (UDF) e calcule as seguintes métricas entre a predição e os dados observados

- Coeficiente de determinação:  $R^2$

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

onde  $y_i$  é o valor observado,  $\hat{y}_i$  é o valor predito e  $\bar{y}$  é a média dos valores  $y_i$  observados.

Quanto mais perto de 1 melhor é o modelo;

- Erro padrão da estimativa:  $S_{yx}$

$$S_{yx} = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n-2}}$$

esta métrica indica erro, portanto quanto mais perto de 0 melhor é o modelo;

- $S_{yx}\%$

$$S_{yx}\% = \frac{S_{yx}}{\bar{y}} * 100$$

esta métrica indica porcentagem de erro, portanto quanto mais perto de 0 melhor é o modelo;

7. Escolha o melhor modelo.

## B – RESOLUÇÃO

- 1.

```
#Instala os pacotes necessários para solução do exercício
install.packages("mlbench")
install.packages("e1071")
install.packages("randomForest")
install.packages("kernlab")
install.packages("caret")

#Importa os pacotes para uso
library(mlbench)
library(caret)

#Carrega a base de dados Sattelite, mantém apenas os atributos 17,18,19
e 20 e mostra o summary
data(Satellite)
Satellite <- subset(Satellite,select = c(x.17,x.18,x.19,x.20,classes))
```

```

head(Satellite)
summary(Satellite)

#Cria partição de treinamento e testes
set.seed(7)
indices <- createDataPartition(Satellite$classes, p=0.80, list=FALSE)
treino <- Satellite[indices,]
teste <- Satellite[-indices,]

#Treina os modelos Random Forest, svm e rna
rf <- train(classes~., data=treino, method="rf")
svm <- train(classes~., data=treino, method="svmRadial")
rna <- train(classes~., data=treino, method="nnet", trace=FALSE)

#Efetua as predições
predicoes.rf <- predict(rf, teste)
predicoes.svm <- predict(svm, teste)
predicoes.rna <- predict(rna, teste)

#Gera as matrizes de confusão
print("Matriz de confusão RF")
confusionMatrix(predicoes.rf, teste$classes)
print("Matriz de confusão SVM")
confusionMatrix(predicoes.svm, teste$classes)
print("Matriz de confusão rna")
confusionMatrix(predicoes.rna, teste$classes)

```

O modelo escolhido com base nas matrizes de confusão impressas foi o SVM.

Entre os três modelos treinados, o SVM foi o modelo que apresentou o melhor resultado.

A métrica utilizada foi a acurácia, conforme o resultados abaixo extraídos da saída do comando `confusionMatrix`.

1. Random forest: 0.8419
2. svm: 0.8707
3. rna: 0.8084

2.

```

#Importa os pacotes para uso
library(mlbench)
library(caret)
library(RSNNS)

getwd() #verifica o diretório de trabalho atual

#Carrega a base de dados do arquivo csv
volumes <- read.csv2("http://www.razer.net.br/datasets/Volumes.csv")
summary(volumes)

# Remove a coluna NR
volumes$NR <- NULL
head(volumes)

```

```

#Cria partição de treinamento e testes
set.seed(7)
indices <- createDataPartition(volumes$VOL, p=0.80, list=FALSE)
treino <- volumes[indices,]
teste <- volumes[-indices,]

#Treina os modelos Random Forest, svm e rna
rf <- caret::train(VOL~., data=treino, method="rf")
svm <- caret::train(VOL~., data=treino, method="svmRadial")
rna <- caret::train(VOL~., data=treino, method="nnet", linout = TRUE)
alom <- nls(VOL ~ b0 + b1*DAP*DAP*HT, data=treino, start=list(b0=0.5,
b1=0.5))

#Efetua as predições
predicoes.rf <- predict(rf, teste)
predicoes.svm <- predict(svm, teste)
predicoes.rna <- predict(rna, teste)
predicoes.alom <- predict(alom, teste)

# Função para cálculo do Coeficiente de determinação R2
RSQUARE = function(observados,preditos){
  cor(observados,preditos)^2
}

r2.rf = RSQUARE(teste$VOL,predicoes.rf)
r2.svm = RSQUARE(teste$VOL,predicoes.svm)
r2.rna = RSQUARE(teste$VOL,predicoes.rna)
r2.alom = RSQUARE(teste$VOL,predicoes.alom)

# Função para cálculo do Syx
syx <- function(observados, preditos, n, p)
{
  sqrt((sum((observados-preditos)^2))/(n-p-1))
}

n <- nrow(treino)
p <- nrow(teste)

syx.rf <- syx(teste$VOL,predicoes.rf,n,p)
syx.svm <- syx(teste$VOL,predicoes.svm,n,p)
syx.rna <- syx(teste$VOL,predicoes.rna,n,p)
syx.alom <- syx(teste$VOL,predicoes.alom,n,p)

# Função para cálculo do Syx%
syxP <- function(syx,observados)
{
  (syx / mean(observados)) * 100
}

syxP.rf <- syxP(syx.rf,teste$VOL)
syxP.svm <- syxP(syx.svm,teste$VOL)
syxP.rna <- syxP(syx.rna,teste$VOL)
syxP.alom <- syxP(syx.alom,teste$VOL)

print("Métricas R²")
r2.rf
r2.alom
r2.rna

```

```

r2.svm

print("Métricas Syx")
syx.rf
syx.alom
syx.rna
syx.svm

print("Métricas SyxP")
syxP.rf
syxP.alom
syxP.rna
syxP.svm

```

De acordo com as métricas calculadas, o melhor modelo foi o RNA, o qual apresentou o maior coeficiente de variação  $R^2$  e o menor erro padrão da estimativa Syx e Syx%:

```

print("Métricas R²")
[1] "Métricas R²"
> r2.rf
[1] 0.8589349
> r2.alom
[1] 0.8546826
> r2.rna
[1] 0.881327
> r2.svm
[1] 0.856365
>
> print("Métricas Syx")
[1] "Métricas Syx"
> syx.rf
[1] 0.07984293
> syx.alom
[1] 0.08695543
> syx.rna
[1] 0.0719159
> syx.svm
[1] 0.08122125
>
> print("Métricas SyxP")
[1] "Métricas SyxP"
> syxP.rf
[1] 6.118093
> syxP.alom
[1] 6.6631
> syxP.rna
[1] 5.510672
> syxP.svm
[1] 6.223709

```



## APÊNDICE 4 – ESTATÍSTICA APLICADA I

### A – ENUNCIADO

#### 1) Gráficos e tabelas

(15 pontos) Elaborar os gráficos box-plot e histograma das variáveis “age” (idade da esposa) e “husage” (idade do marido) e comparar os resultados

(15 pontos) Elaborar a tabela de frequências das variáveis “age” (idade da esposa) e “husage” (idade do marido) e comparar os resultados

#### 2) Medidas de posição e dispersão

(15 pontos) Calcular a média, mediana e moda das variáveis “age” (idade da esposa) e “husage” (idade do marido) e comparar os resultados

(15 pontos) Calcular a variância, desvio padrão e coeficiente de variação das variáveis “age” (idade da esposa) e “husage” (idade do marido) e comparar os resultados

#### 3) Testes paramétricos ou não paramétricos

(40 pontos) Testar se as médias (se você escolher o teste paramétrico) ou as medianas (se você escolher o teste não paramétrico) das variáveis “age” (idade da esposa) e “husage” (idade do marido) são iguais, construir os intervalos de confiança e comparar os resultados.

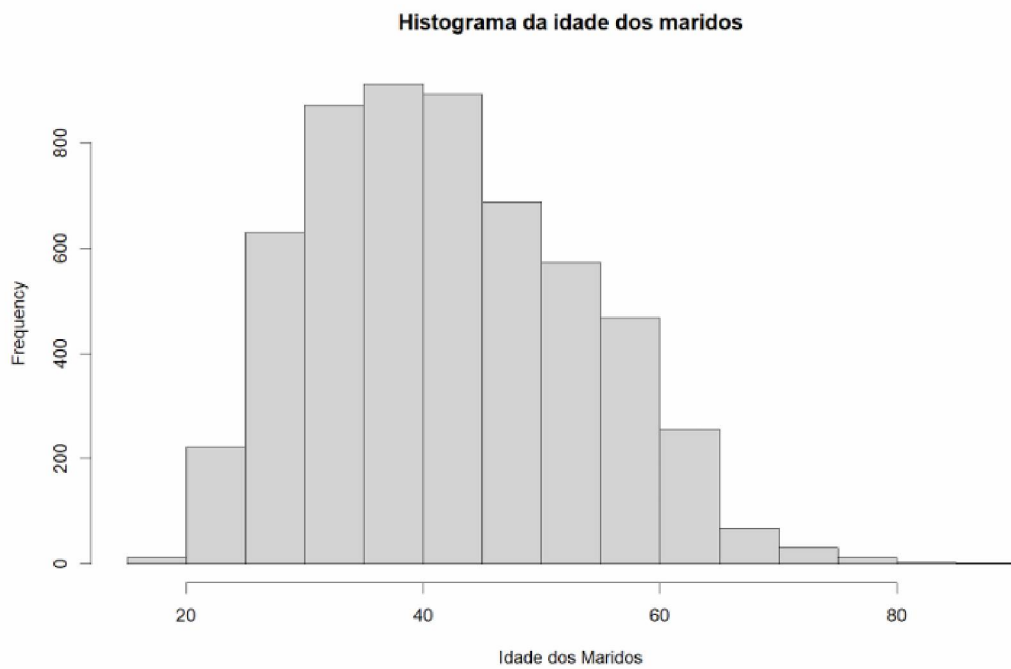
Obs:

Você deve fazer os testes necessários (e mostra-los no documento pdf) para saber se você deve usar o unpaired test (paramétrico) ou o teste U de Mann-Whitney (não paramétrico), justifique sua resposta sobre a escolha.

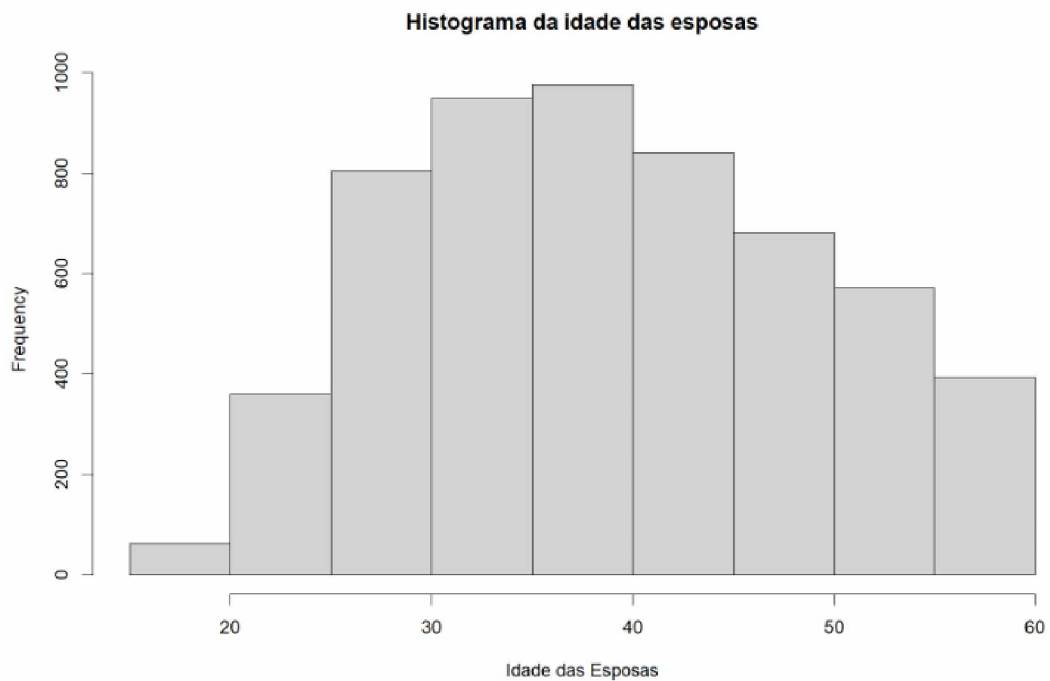
Lembre-se de que os intervalos de confiança já são mostrados nos resultados dos testes citados no item 1 acima.

### B – RESOLUÇÃO

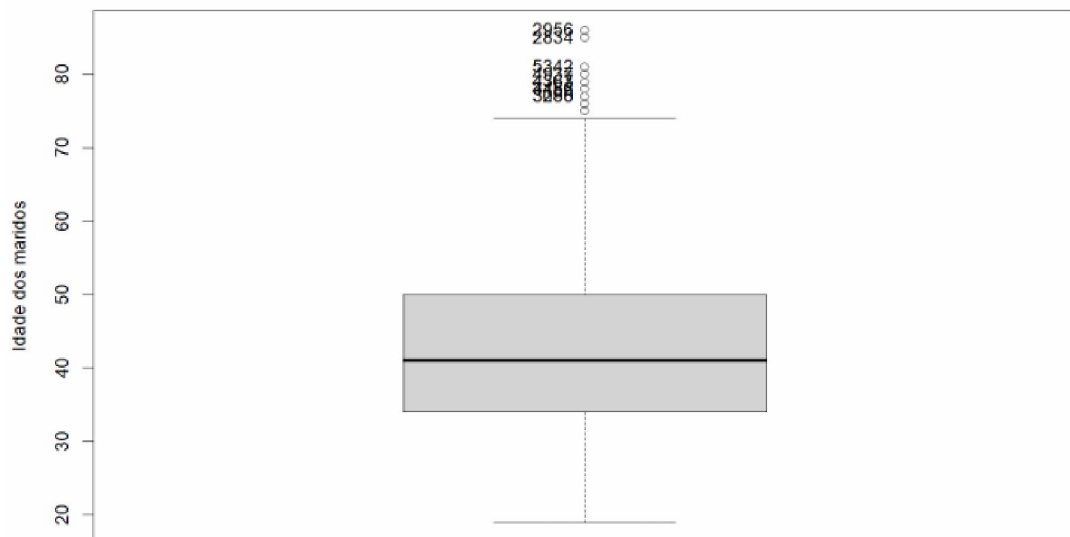
1.



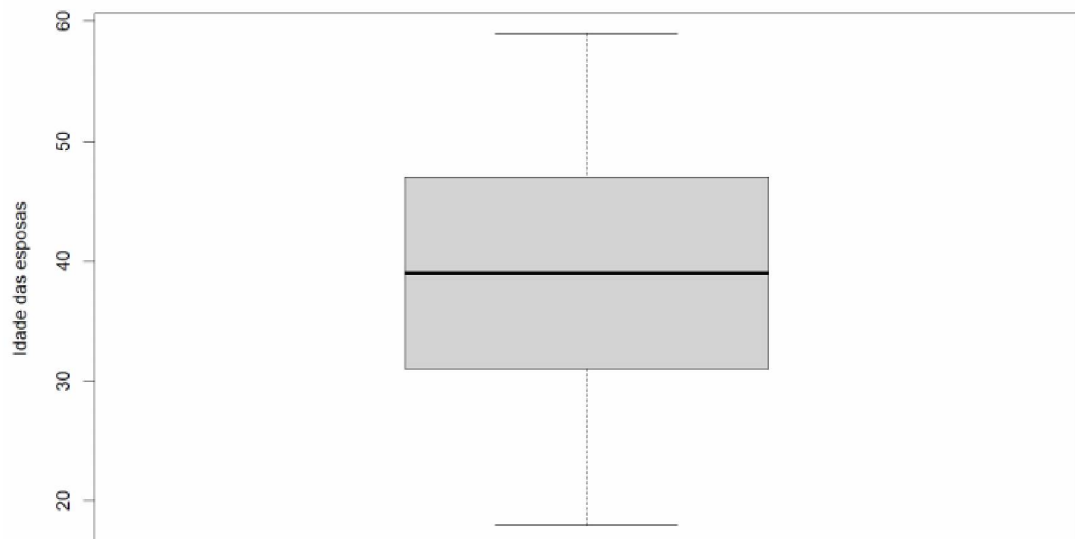
A partir desse gráfico podemos deduzir que a maior quantidade dos maridos possui entre 30 e 50 anos. Existe também uma menor parte dos maridos que possui menos de 20 anos e acima de 70 anos.



A partir desse gráfico podemos deduzir que a maior quantidade das esposas possui a idade entre 30 e 40 anos. E que apenas uma pequena parte das esposas possui idade inferior a 20 anos.



A partir desse gráfico podemos concluir que a idade dos maridos apresenta outliers e podemos concluir que os maridos possuem idade máxima maior que a idade das esposas.



A partir desse gráfico podemos concluir que a idade das esposas não apresenta outliers e o gráfico não possui valores discrepantes em relação à mediana.

Tabela de Frequência para a idade dos maridos:

| Class limits    | f   | rf   | rf(%) | cf   | cf(%) |
|-----------------|-----|------|-------|------|-------|
| [18.81,23.671)  | 102 | 0.02 | 1.81  | 102  | 1.81  |
| [23.671,28.531) | 466 | 0.08 | 8.27  | 568  | 10.08 |
| [28.531,33.392) | 809 | 0.14 | 14.36 | 1377 | 24.44 |
| [33.392,38.253) | 895 | 0.16 | 15.89 | 2272 | 40.33 |

```
[38.253,43.114) 917 0.16 16.28 3189 56.60
[43.114,47.974) 629 0.11 11.16 3818 67.77
[47.974,52.835) 649 0.12 11.52 4467 79.29
[52.835,57.696) 541 0.10 9.60 5008 88.89
[57.696,62.556) 394 0.07 6.99 5402 95.88
[62.556,67.417) 152 0.03 2.70 5554 98.58
[67.417,72.278) 51 0.01 0.91 5605 99.49
[72.278,77.139) 21 0.00 0.37 5626 99.86
[77.139,81.999) 6 0.00 0.11 5632 99.96
[81.999,86.86) 2 0.00 0.04 5634 100.00
```

Tabela de Frequência para a idade das esposas:

```
Class limits f rf rf(%) cf cf(%)
[17.82,20.804) 61 0.01 1.08 61 1.08
[20.804,23.787) 161 0.03 2.86 222 3.94
[23.787,26.771) 312 0.06 5.54 534 9.48
[26.771,29.754) 505 0.09 8.96 1039 18.44
[29.754,32.738) 562 0.10 9.98 1601 28.42
[32.738,35.721) 571 0.10 10.13 2172 38.55
[35.721,38.705) 624 0.11 11.08 2796 49.63
[38.705,41.689) 510 0.09 9.05 3306 58.68
[41.689,44.672) 542 0.10 9.62 3848 68.30
[44.672,47.656) 432 0.08 7.67 4280 75.97
[47.656,50.639) 389 0.07 6.90 4669 82.87
[50.639,53.623) 358 0.06 6.35 5027 89.23
[53.623,56.606) 304 0.05 5.40 5331 94.62
[56.606,59.59) 303 0.05 5.38 5634 100.00
```

Pode-se concluir que a idade das esposas é mais igualmente distribuída que a idade dos maridos e que não possui tanta diferença entre os valores com a mediana.

2.

A média da idade dos maridos é cerca de 7.67% maior do que a média da idade das esposas A mediana da idade dos maridos é cerca de 5.13% maior do que a mediana da idade das esposas A moda da idade dos maridos é cerca de 18.92% maior do que a moda da idade das esposas.

A variância da idade dos maridos é cerca de 26.41% maior do que a variância da idade das esposas. O desvio padrão da idade dos maridos é cerca de 12.42% maior do que o desvio padrão da idade das esposas. A variação na idade dos maridos é ligeiramente maior do que a das esposas, pois o CV dos maridos é maior. Em outras palavras, as idades dos maridos estão mais dispersas em torno da média do que as idades das esposas.

3.

Op-valor de ambas as amostras é menor que o nível de significância 0.05. Então rejeita-se H0, concluindo que a distribuição dos dados é estatisticamente diferente da distribuição normal Para distribuições não normais, utilizaremos então o teste não-paramétrico.

```
# Avaliando o teste não paramétrico Mann-Whitney "U" test
# Sumario estatistico
group_by(idades, categoria) %>%
```

```

    summarise( count = n(), median = median(idade, na.rm = TRUE), IQR
= IQR(idade, na.rm = TRUE) )

    # Testar se a idade mediana dos maridos é igual a idade mediana
das esposas
    # Hipoteses:
    # H0: A idade mediana dos maridos é igual estatisticamente a idade
mediana das esposas;
    # Ha: A idade mediana dos maridos nao é estatisticamente igual a
idade mediana das esposas;
    wilcox.test(idade ~ categoria, data = idades, exact = FALSE,
conf.int=TRUE)
    ### p < 0,05- H0 é rejeitado- a mediana dos maridos não é
estatisticamente igual a idade mediana das esposas

    # Testar se a idade mediana dos maridos é menor que a idade
mediana das esposas # Hipoteses do teste:
    # H0: A idade mediana dos maridos nao é estatisticamente menor que
a idade mediana das esposas;
    # Ha: A idade mediana dos maridos é estatisticamente menor que a
idade mediana das esposas
    wilcox.test(idade ~ categoria, data = idades, exact = FALSE,
alternative = "less", conf.int=TRUE)
    # p-valor > 0,05- H0 é aceita- idade mediana dos maridos não é
estatisticamente menor que a idade mediana das esposas
    # Testar se a idade mediana dos maridos é maior que a idade
mediana das esposas

    # Hipoteses do teste:
    # H0: A idade mediana dos maridos nao é estatisticamente maior que
a idade mediana das esposas;
    # Ha: A idade mediana dos maridos é estatisticamente maior que a
idade mediana das esposas
    wilcox.test(idade ~ categoria, data = idades, exact = FALSE,
alternative = "greater", conf.int=TRUE)
    ### p < 0,05- H0 é rejeitado- a mediana dos maridos não é
estatisticamente menor a idade mediana das esposas

```

A escolha foi pelo teste não paramétrico, uma vez que, no teste paramétrico, o p-valor de ambas as amostras é menor que o nível de significância de 0.05, indicando que a distribuição dos dados é estatisticamente diferente da distribuição normal. Seguindo para o teste não paramétrico U de Mann-Whitney, foi possível concluir que idade mediana dos maridos não é estatisticamente menor que a idade mediana das esposas, uma vez que o p-valor do teste foi maior que o nível de significância. Foi possível concluir ainda que o intervalo de confiança da diferença entre as medianas é um valor menor que 3,00, com uma mediana de 2,99.

## APÊNDICE 5 – ESTATÍSTICA APLICADA II

### A – ENUNCIADO

#### Regressões Ridge, Lasso e ElasticNet

**(100 pontos)** Fazer as regressões Ridge, Lasso e ElasticNet com a variável dependente “lwage” (salário-hora da esposa em logaritmo neperiano) e todas as demais variáveis da base de dados são variáveis explicativas (todas essas variáveis tentam explicar o salário-hora da esposa). No pdf você deve colocar a rotina utilizada, mostrar em uma tabela as estatísticas dos modelos (RMSE e  $R^2$ ) e concluir qual o melhor modelo entre os três, e mostrar o resultado da predição com intervalos de confiança para os seguintes valores:

|                |                                             |
|----------------|---------------------------------------------|
| husage = 40    | (anos – idade do marido)                    |
| husunion = 0   | (marido não possui união estável)           |
| husearns = 600 | (US\$ renda do marido por semana)           |
| huseduc = 13   | (anos de estudo do marido)                  |
| husblack = 1   | (o marido é preto)                          |
| hushisp = 0    | (o marido não é hispânico)                  |
| hushrs = 40    | (horas semanais de trabalho do marido)      |
| kidge6 = 1     | (possui filhos maiores de 6 anos)           |
| age = 38       | (anos – idade da esposa)                    |
| black = 0      | (a esposa não é preta)                      |
| educ = 13      | (anos de estudo da esposa)                  |
| hispanic = 1   | (a esposa é hispânica)                      |
| union = 0      | (esposa não possui união estável)           |
| exper = 18     | (anos de experiência de trabalho da esposa) |
| kidlt6 = 1     | (possui filhos menores de 6 anos)           |

obs: lembre-se de que a variável dependente “lwage” já está em logaritmo, portanto você não precisa aplicar o logaritmo nela para fazer as regressões, mas é necessário aplicar o antilog para obter o resultado da predição.

### B – RESOLUÇÃO

Métricas do modelo sem a utilização da variável “earnings”:

| Modelo                   | RMSE      | R <sup>2</sup> |
|--------------------------|-----------|----------------|
| Ridge (treinamento)      | 0.8312971 | 0.3086094      |
| Ridge (teste)            | 0.9814711 | 0.2060657      |
| Lasso (treinamento)      | 0.832615  | 0.3064155      |
| Lasso (teste)            | 0.9833673 | 0.2029951      |
| ElasticNet (treinamento) | 0.8319181 | 0.307576       |
| ElasticNet (teste)       | 0.9824075 | 0.2045501      |

Métricas do modelo com a utilização da variável “earnings”:

| Modelo                   | RMSE      | R <sup>2</sup> |
|--------------------------|-----------|----------------|
| Ridge (treinamento)      | 0.5225343 | 0.7268253      |
| Ridge (teste)            | 0.7205827 | 0.5720459      |
| Lasso (treinamento)      | 0.522357  | 0.7270106      |
| Lasso (teste)            | 0.7218243 | 0.5705699      |
| ElasticNet (treinamento) | 0.5223142 | 0.7270553      |
| ElasticNet (teste)       | 0.7219057 | 0.570473       |

De acordo com as métricas acima, o melhor modelo é o Ridge, uma vez que a métrica R<sup>2</sup> deste modelo possui um valor mais próximo de 1 se comparada com os demais, e a métrica RMSE é mais próxima de zero se comparada com as demais- apesar da diferença ser mínima entre os três modelos. Vale considerar ainda que as métricas dos modelos sem o uso da variável “earnings” possuem valores abaixo dos ideais, indicando que o poder explicativo destes modelos é baixo. É possível perceber também que a variável “earnings” impacta significativamente no modelo, uma vez que a inclusão desta variável afeta positivamente nos valores das métricas.

As previsões foram feitas utilizando o modelo sem a inclusão da variável “earnings”, conforme orientação recebida, e os valores preditos em cada modelo estão descritos abaixo:

- Regressão Ridge Valor do salário-hora predito: 7.89 Intervalo de confiança: 7.72 a 8.06
- Regressão Lasso Valor do salário-hora predito: 8.21 Intervalo de confiança: 8.03 a 8.39
- Regressão ElasticNet Valor do salário-hora predito: 8.12 Intervalo de confiança: 7.95 a 8.30

## APÊNDICE 6 – ARQUITETURA DE DADOS

### A – ENUNCIADO

#### 1 Construção de Características: Identificador automático de idioma

O problema consiste em criar um modelo de reconhecimento de padrões que dado um texto de entrada, o programa consegue classificar o texto e indicar a língua em que o texto foi escrito.

Parta do exemplo (notebook produzido no Colab) que foi disponibilizado e crie as funções para calcular as diferentes características para o problema da identificação da língua do texto de entrada.

Nessa atividade é para "construir características".

Meta: a acurácia deverá ser maior ou igual a 70%.

Essa tarefa pode ser feita no Colab (Google) ou no Jupiter, em que deverá exportar o notebook e imprimir o notebook para o formato PDF. Envie no UFPR Virtual os dois arquivos.

#### 2 Melhore uma base de dados ruim

Escolha uma base de dados pública para problemas de classificação, disponível ou com origem na UCI Machine Learning.

Use o mínimo de intervenção para rodar a SVM e obtenha a matriz de confusão dessa base.

O trabalho começa aqui, escolha as diferentes tarefas discutidas ao longo da disciplina, para melhorar essa base de dados, até que consiga efetivamente melhorar o resultado.

Considerando a acurácia para bases de dados balanceadas ou quase balanceadas, se o percentual da acurácia original estiver em até 85%, a meta será obter 5%. Para bases com mais de 90% de acurácia, a meta será obter a melhora em pelo menos 2 pontos percentuais (92% ou mais).

Nessa atividade deverá ser entregue o script aplicado (o notebook e o PDF correspondente).

### B – RESOLUÇÃO



1.

```

# Construção dos atributos
import re

def tamanhoMedioFrases(texto):
    palavras = re.split("\s", texto)
    tamanhos = [len(s) for s in palavras if len(s)>0]
    soma = 0
    for t in tamanhos:
        soma=soma+t
    return soma / len(tamanhos)

# Função para verificar se a frase possui caracteres/dígrafos/sufixos
do idioma portugues
def possui_caracteres_ptbr(frase):
    if 'ã' in frase or 'â' in frase or 'ê' in frase or 'ú' in frase or
    'ç' in frase or 'Ç' in frase or 'nh' in frase or 'lh' in frase:
        return 1
    else:
        return 0

# Função para verificar se a frase possui caracteres/dígrafos/sufixos
do idioma ingles
def possui_caracteres_en(frase):
    if 'ing ' in frase or 'eat' in frase:
        return 1
    else:
        return 0

# Função para verificar se a frase possui caracteres/dígrafos/sufixos
do idioma espanhol
def possui_caracteres_es(frase):
    if 'ñ' in frase or 'ía' in frase or 'ón' in frase:
        return 1
    else:
        return 0

# Função para verificar se a frase possui termos exatos(preposições,
conjunções, artigos) de acordo com os vetores de cada idioma
def possui_termos(text,dic):
    r = re.search(r'\b({})\b'.format("|".join(dic)), text)
    if r is None:
        return 0
    else:
        return 1

termos_En = ['the', 'on', 'is', 'to', 'you', 'for', 'with', 'She', 'was', 'can']
termos_Es = ['el', 'El', 'Ella', 'la', 'es', 'un', 'una', 'uns', 'unas', 'con', 'Me', 'más', 'algunas', 'las', 'al', 'Quiero', 'cómo']
termos_Pt = ['é', 'para', 'um', 'uma', 'com', 'às', 'aos', 'esse', 'no', 'na', 'que']

def extraiCaracteristicas(frase):
    # frase é um vetor [ 'texto', 'lingua' ]
    texto = frase[0]
    pattern_regex = re.compile('[^\w+]', re.UNICODE)
    texto = re.sub(pattern_regex, ' ', texto)

```

```

    caracteristica1=tamanhoMedioFrases(texto) # Foi removida do modelo,
    pois trouxe um pequeno impacto negativo na acurácia
    caracteristica2=possui_termos(texto,termos_En)
    caracteristica3=possui_termos(texto,termos_Es)
    caracteristica4=possui_termos(texto,termos_Pt)
    caracteristica5=possui_caracteres_en(texto)
    caracteristica6=possui_caracteres_es(texto)
    caracteristica7=possui_caracteres_ptbr(texto)

    # acrescenta as suas funcoes no vetor padrao
    padrao = [caracteristica2,
caracteristica3,caracteristica4,caracteristica5,caracteristica6,caracte
ristica7,frase[1] ]
    return padrao

def geraPadroes(frases):
    padroes = []
    for frase in frases:
        padrao = extraiCaracteristicas(frase)
        padroes.append(padrao)
    return padroes

# converte o formato [frase classe] em
# [caracteristica_1, caracteristica_2,... caracteristica n, classe]
padroes = geraPadroes(pre_padroes)

#
# apenas para visualizacao
print(padroes)

dados = pd.DataFrame(padroes)
dados

# treinando modelo com SVM
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
import numpy as np

vet = np.array(padroes)
classes = vet[:, -1] # classes = [p[-1] for p in padroes]
padroes_sem_classe = vet[:, 0:-1]
X_train, X_test, y_train, y_test = train_test_split(padroes_sem_classe,
classes, test_size=0.25, stratify=classes)

treinador = svm.SVC() #algoritmo escolhido
modelo = treinador.fit(X_train, y_train)

#
# score com os dados de treinamento
acuracia = modelo.score(X_train, y_train)
print("Acurácia nos dados de treinamento: {:.2f}%".format(acuracia *
100))

```

```
#
# melhor avaliar com a matriz de confusão
y_pred = modelo.predict(X_train)
cm = confusion_matrix(y_train, y_pred)
print(cm)
print(classification_report(y_train, y_pred))

#
# com dados de teste que não foram usados no treinamento
print('métricas mais confiáveis')
y_pred2 = modelo.predict(X_test)
cm = confusion_matrix(y_test, y_pred2)
print(cm)
print(classification_report(y_test, y_pred2))
```

Acurácia nos dados de treinamento: 97.10%

```
[[20  0  2]
 [ 0 23  0]
 [ 0  0 24]]
```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| espanhol     | 1.00      | 0.91   | 0.95     | 22      |
| inglês       | 1.00      | 1.00   | 1.00     | 23      |
| português    | 0.92      | 1.00   | 0.96     | 24      |
| accuracy     |           |        | 0.97     | 69      |
| macro avg    | 0.97      | 0.97   | 0.97     | 69      |
| weighted avg | 0.97      | 0.97   | 0.97     | 69      |

métricas mais confiáveis

```
[[8 0 0]
 [0 7 0]
 [0 0 8]]
```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| espanhol     | 1.00      | 1.00   | 1.00     | 8       |
| inglês       | 1.00      | 1.00   | 1.00     | 7       |
| português    | 1.00      | 1.00   | 1.00     | 8       |
| accuracy     |           |        | 1.00     | 23      |
| macro avg    | 1.00      | 1.00   | 1.00     | 23      |
| weighted avg | 1.00      | 1.00   | 1.00     | 23      |

## 2.

```
# Tratamentos realizados
# Normalização das colunas, remoção da coluna de identificação e
# separação da classe dos atributos
# Tratamento dos dados: remoção de outliers e correção da prevalência.

X = mhr.iloc[:,0:6]
cols = mhr_colunas[0:6]

Y = mhr['RiskLevel']
Y_orig = mhr['RiskLevel']
from sklearn.preprocessing import scale
from sklearn.preprocessing import minmax_scale
import pandas as pd
```

```

import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
import imblearn
from imblearn.over_sampling import RandomOverSampler
from collections import Counter

# Identificando outliers
threshold = 3

z = np.abs(stats.zscore(X['Age']))
outliers = X[z > threshold]
print(outliers)

z = np.abs(stats.zscore(X['SystolicBP']))
outliers = X[z > threshold]
print(outliers)

z = np.abs(stats.zscore(X['DiastolicBP']))
outliers = X[z > threshold]
print(outliers)

z = np.abs(stats.zscore(X['BS']))
outliers = X[z > threshold]
print(outliers)

z = np.abs(stats.zscore(X['BodyTemp']))
outliers = X[z > threshold]
print(outliers)

z = np.abs(stats.zscore(X['HeartRate']))
outliers = X[z > threshold]
print(outliers)

X_orig = X.copy()

# Splitting Dataset
#from sklearn.tree import DecisionTreeClassifier
#dt=DecisionTreeClassifier(max_depth=300)
#dtPrel=dt.fit(X, Y).predict_proba(X)
#X=pd.DataFrame(dtPrel)

#Foram encontrados registros com batimentos cardíacos com valores
compatíveis com outliers (7) - os valores serão corrigidos pela média
X.loc[z > threshold, 'HeartRate'] = X['HeartRate'].median()

#Avaliando a distribuição dos dados
class_count = Y.value_counts().head
print(class_count)

#As classes estão desbalanceadas, portanto optou-se por fazer o
oversampling
oversampler = RandomOverSampler(sampling_strategy='auto',
random_state=42)
X_resampled, y_resampled = oversampler.fit_resample(X, Y)

class_count = y_resampled.value_counts().head
print(class_count)

```

```
# normalização min-max
X = pd.DataFrame(minmax_scale(X_resampled))
Y = y_resampled
```

```
Empty DataFrame
Columns: [Age, SystolicBP, DiastolicBP, BS, BodyTemp, HeartRate]
Index: []
Empty DataFrame
Columns: [Age, SystolicBP, DiastolicBP, BS, BodyTemp, HeartRate]
Index: []
Empty DataFrame
Columns: [Age, SystolicBP, DiastolicBP, BS, BodyTemp, HeartRate]
Index: []
```

|           | Age | SystolicBP | DiastolicBP | BS   | BodyTemp | HeartRate |
|-----------|-----|------------|-------------|------|----------|-----------|
| 122       | 40  | 160        | 100         | 19.0 | 98.0     | 77        |
| 126       | 55  | 140        | 95          | 19.0 | 98.0     | 77        |
| 129       | 40  | 160        | 100         | 19.0 | 98.0     | 77        |
| 136       | 55  | 140        | 95          | 19.0 | 98.0     | 77        |
| 165       | 40  | 160        | 100         | 19.0 | 98.0     | 77        |
| 181       | 55  | 140        | 95          | 19.0 | 98.0     | 77        |
| 261       | 40  | 160        | 100         | 19.0 | 98.0     | 77        |
| 277       | 55  | 140        | 95          | 19.0 | 98.0     | 77        |
| 318       | 35  | 85         | 60          | 19.0 | 98.0     | 86        |
| 361       | 40  | 160        | 100         | 19.0 | 98.0     | 77        |
| 372       | 55  | 140        | 95          | 19.0 | 98.0     | 77        |
| 537       | 40  | 160        | 100         | 19.0 | 98.0     | 77        |
| 553       | 55  | 140        | 95          | 19.0 | 98.0     | 77        |
| 582       | 40  | 160        | 100         | 19.0 | 98.0     | 77        |
| 598       | 55  | 140        | 95          | 19.0 | 98.0     | 77        |
| ...       |     |            |             |      |          |           |
| high risk |     | 406        |             |      |          |           |
| low risk  |     | 406        |             |      |          |           |
| mid risk  |     | 406        |             |      |          |           |

```
Name: count, dtype: int64>
```

```
# trata a construção do modelo, os testes e as métricas da matriz de
confusão
from sklearn.model_selection import train_test_split
import numpy as np

# com os dados originais
X_orig_train, X_orig_test, y_orig_train, y_orig_test =
train_test_split(X_orig,
                  Y_orig, test_size=0.25,
                  stratify=Y_orig, random_state=10)

# com os dados tratados
X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.25,
stratify=Y, random_state=10)
# Treina o modelo com base nos dados originais (SVM)
from sklearn import svm
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report

treinador = svm.SVC() #algoritmo escolhido
```



```

modelo_orig = treinador.fit(X_oring_train, y_oring_train)

# predição com os mesmos dados usados para treinar
y_orig_pred = modelo_orig.predict(X_oring_train)
cm_orig_train = confusion_matrix(y_oring_train, y_orig_pred)
print('Matriz de confusão - com os dados ORIGINAIS usados no TREINAMENTO')
print(cm_orig_train)
print(classification_report(y_oring_train, y_orig_pred))

# predição com os mesmos dados usados para testar
print('Matriz de confusão - com os dados ORIGINAIS usados para TESTES')
y2_orig_pred = modelo_orig.predict(X_orig_test)
cm_orig_test = confusion_matrix(y_orig_test, y2_orig_pred)
print(cm_orig_test)
print(classification_report(y_orig_test, y2_orig_pred))

```

Matriz de confusão - com os dados ORIGINAIS usados no TREINAMENTO

```

[[111  63  29]
 [  0 272  32]
 [ 10 161  81]]

```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| high risk    | 0.92      | 0.55   | 0.69     | 203     |
| low risk     | 0.55      | 0.89   | 0.68     | 304     |
| mid risk     | 0.57      | 0.32   | 0.41     | 252     |
| accuracy     |           |        | 0.61     | 759     |
| macro avg    | 0.68      | 0.59   | 0.59     | 759     |
| weighted avg | 0.65      | 0.61   | 0.59     | 759     |

Matriz de confusão - com os dados ORIGINAIS usados para TESTES

```

[[43 12 13]
 [ 1 89 12]
 [ 5 58 21]]

```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| high risk    | 0.88      | 0.63   | 0.74     | 68      |
| low risk     | 0.56      | 0.87   | 0.68     | 102     |
| mid risk     | 0.46      | 0.25   | 0.32     | 84      |
| accuracy     |           |        | 0.60     | 254     |
| macro avg    | 0.63      | 0.58   | 0.58     | 254     |
| weighted avg | 0.61      | 0.60   | 0.58     | 254     |

```

from sklearn import svm
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report

treinador = svm.SVC() #algoritmo escolhido

modelo = treinador.fit(X_train, y_train)

# predição com os mesmos dados usados para treinar
y_pred = modelo.predict(X_train)
cm_train = confusion_matrix(y_train, y_pred)

```

```

print('Matriz de confusão - com os dados TRATADOS usados no
TREINAMENTO')
print(cm_train)
print(classification_report(y_train, y_pred))

# predição com os mesmos dados usados para testar
print('Matriz de confusão - com os dados TRATADOS usados para TESTES')
y2_pred = modelo.predict(X_test)
cm_test = confusion_matrix(y_test, y2_pred)
print(cm_test)
print(classification_report(y_test, y2_pred))

```

Matriz de confusão - com os dados TRATADOS usados no TREINAMENTO

```

[[262  11  32]
 [  8 266  30]
 [ 31 143 130]]

```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| high risk    | 0.87      | 0.86   | 0.86     | 305     |
| low risk     | 0.63      | 0.88   | 0.73     | 304     |
| mid risk     | 0.68      | 0.43   | 0.52     | 304     |
| accuracy     |           |        | 0.72     | 913     |
| macro avg    | 0.73      | 0.72   | 0.71     | 913     |
| weighted avg | 0.73      | 0.72   | 0.71     | 913     |

Matriz de confusão - com os dados TRATADOS usados para TESTES

```

[[85  2 14]
 [ 4 83 15]
 [13 49 40]]

```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| high risk    | 0.83      | 0.84   | 0.84     | 101     |
| low risk     | 0.62      | 0.81   | 0.70     | 102     |
| mid risk     | 0.58      | 0.39   | 0.47     | 102     |
| accuracy     |           |        | 0.68     | 305     |
| macro avg    | 0.68      | 0.68   | 0.67     | 305     |
| weighted avg | 0.68      | 0.68   | 0.67     | 305     |

## APÊNDICE 7 – APRENDIZADO DE MÁQUINA

### A – ENUNCIADO

Para cada uma das tarefas abaixo (Classificação, Regressão etc.) e cada base de dados (Veículo, Diabetes etc.), fazer os experimentos com todas as técnicas solicitadas (KNN, RNA etc.) e preencher os quadros com as estatísticas solicitadas, bem como os resultados pedidos em cada experimento.

### B – RESOLUÇÃO

Para o experimento de Classificação:

| Veículo        |                         |          |                                                                                                               |
|----------------|-------------------------|----------|---------------------------------------------------------------------------------------------------------------|
| Técnica        | Parâmetro               | Acurácia | Matriz de Confusão                                                                                            |
| SVM – Hold-out | C=1<br>Sigma=0.0653699  | 0.7882   | Reference<br>Prediction bus opel saab van<br>bus 42 0 0 0<br>opel 0 18 10 0<br>saab 0 21 32 0<br>van 2 2 1 42 |
| SVM – CV       | C=1<br>Sigma=0.05755071 | 0.7882   | Reference<br>Prediction bus opel saab van<br>bus 42 0 0 0<br>opel 0 19 11 0<br>saab 0 20 31 0<br>van 2 2 1 42 |
| RF – CV        | mtry=10                 | 0.7765   | Reference<br>Prediction bus opel saab van<br>bus 43 0 1 0<br>opel 0 21 11 0<br>saab 0 18 28 2<br>van 1 2 3 40 |
| RF – Hold-out  | mtry=2                  | 0.7706   | Reference<br>Prediction bus opel saab van<br>bus 44 0 0 1<br>opel 0 19 13 0<br>saab 0 19 27 0<br>van 0 3 3 41 |
| RNA – CV       | size=21 decay=0.9       | 0.7529   | Reference<br>Prediction bus opel saab van<br>bus 40 2 2 0<br>opel 0 11 5 0<br>saab 1 28 35 0<br>van 3 0 1 42  |
| RNA – Hold-out | size=5 decay=0.1        | 0.7353   | Reference<br>Prediction bus opel saab van<br>bus 40 1 3 0<br>opel 1 14 10 0<br>saab 1 25 29 0<br>van 2 1 1 42 |
| KNN            | k=1                     | 0.6471   | Reference<br>Prediction bus opel saab van<br>bus 37 2 3 4<br>opel 0 14 16 2<br>saab 4 22 24 1<br>van 3 3 0 35 |



## Diabetes

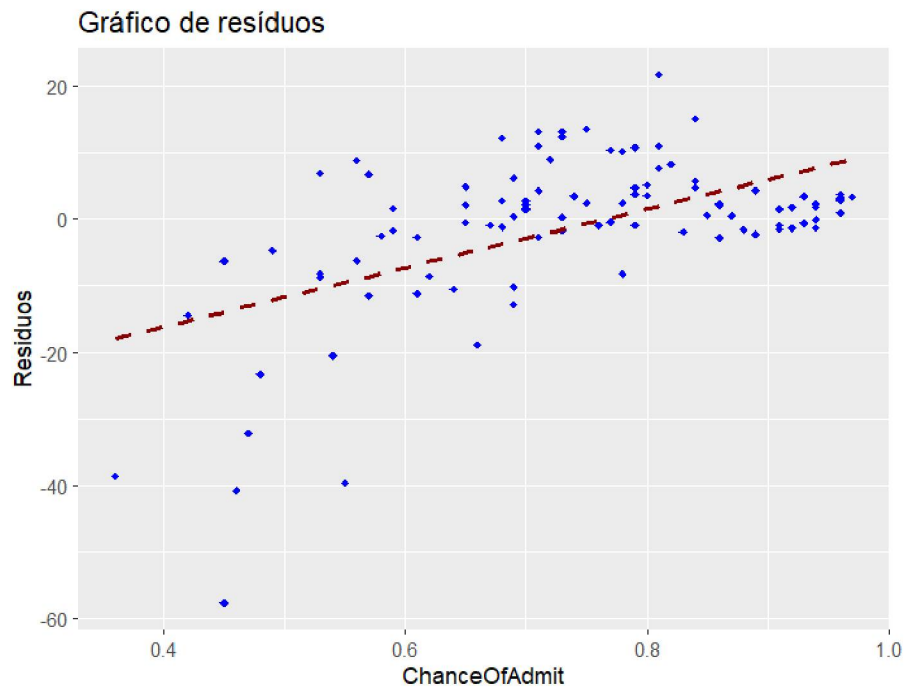
| Técnica               | Parâmetro                 | Acurácia | Matriz de Confusão                                        |
|-----------------------|---------------------------|----------|-----------------------------------------------------------|
| <b>SVM – Hold-out</b> | C=0.25<br>Sigma=0.1234769 | 0.7532   | Reference<br>Prediction neg pos<br>neg 88 27<br>pos 11 28 |
| <b>SVM – CV</b>       | C=0.25<br>Sigma=0.1410459 | 0.7597   | Reference<br>Prediction neg pos<br>neg 89 27<br>pos 10 28 |
| <b>RF – CV</b>        | mtry=8                    | 0.7338   | Reference<br>Prediction neg pos<br>neg 83 25<br>pos 16 30 |
| <b>RF – Hold-out</b>  | mtry=2                    | 0.7706   | Reference<br>Prediction neg pos<br>neg 86 24<br>pos 13 31 |
| <b>RNA – CV</b>       | size=1 decay=0.7          | 0.7403   | Reference<br>Prediction neg pos<br>neg 84 25<br>pos 15 30 |
| <b>RNA – Hold-out</b> | size=3 decay=0.1          | 0.7338   | Reference<br>Prediction neg pos<br>neg 81 23<br>pos 18 32 |
| <b>KNN</b>            | k=9                       | 0.6753   | Reference<br>Prediction neg pos<br>neg 78 29<br>pos 21 26 |

Para o experimento de Regressão:

## Admissão

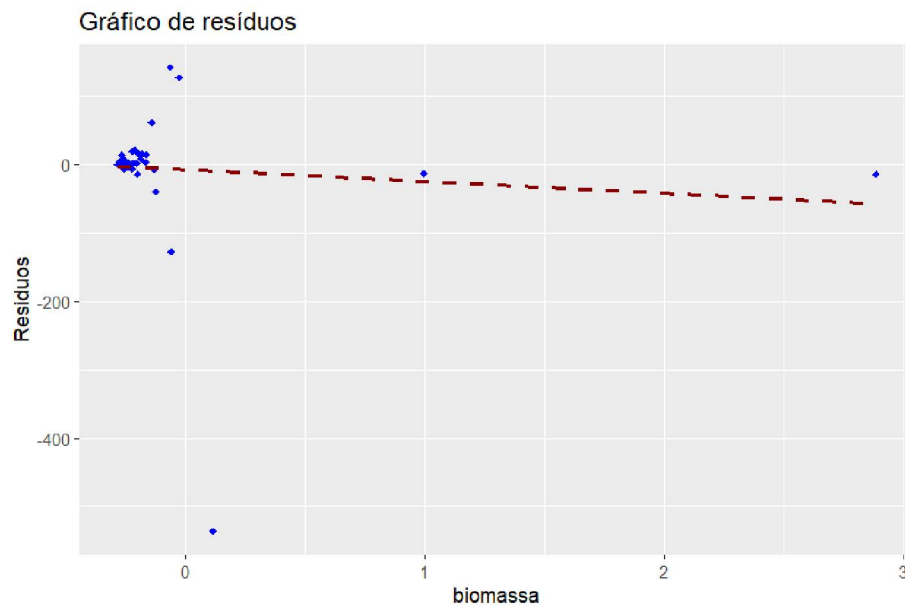
| Técnica               | Parâmetro                | R2        | Syx        | Pearson   | Rmse       | MAE        |
|-----------------------|--------------------------|-----------|------------|-----------|------------|------------|
| <b>RF – CV</b>        | mtry=2                   | 0.7954605 | 0.03872289 | 0.8924344 | 0.06695813 | 0.04657689 |
| <b>RF – Hold-out</b>  | mtry=2                   | 0.7950212 | 0.03876445 | 0.8920844 | 0.06703    | 0.04677648 |
| <b>SVM – CV</b>       | C=1<br>Sigma=0.1887577   | 0.7877206 | 0.03944873 | 0.8897533 | 0.06821323 | 0.04707763 |
| <b>RNA – Hold-out</b> | size=5<br>decay=0.0001   | 0.7866996 | 0.03954349 | 0.8873683 | 0.06837708 | 0.04606429 |
| <b>SVM – Hold-out</b> | C=0.5<br>Sigma=0.2278116 | 0.7844834 | 0.03974838 | 0.8890079 | 0.06873138 | 0.04794774 |
| <b>KNN</b>            | K=9                      | 0.7255268 | 0.04485689 | 0.8520559 | 0.07756481 | 0.05700556 |

|                 |                      |          |           |           |            |            |
|-----------------|----------------------|----------|-----------|-----------|------------|------------|
| <b>RNA – CV</b> | size=21<br>decay=0.1 | 0.610896 | 0.0534087 | 0.7842805 | 0.09235227 | 0.07102196 |
|-----------------|----------------------|----------|-----------|-----------|------------|------------|



### Biomassa

| <b>Técnica</b>        | <b>Parâmetro</b>       | <b>R2</b> | <b>Syx</b> | <b>Pearson</b> | <b>Rmse</b> | <b>MAE</b> |
|-----------------------|------------------------|-----------|------------|----------------|-------------|------------|
| <b>RF – Hold-out</b>  | mtry=2                 | 0.9524863 | 0.05434587 | 0.992358       | 0.09386797  | 0.03131866 |
| <b>RF – CV</b>        | mtry=3                 | 0.9434469 | 0.05929054 | 0.9876647      | 0.1024086   | 0.03406133 |
| <b>KNN</b>            | K=1                    | 0.8825298 | 0.08545173 | 0.9714224      | 0.147595    | 0.04077413 |
| <b>SVM – Hold-out</b> | C=1<br>Sigma=0.9910463 | 0.8472997 | 0.09742646 | 0.9615432      | 0.1682782   | 0.07168103 |
| <b>SVM – CV</b>       | C=1<br>Sigma=1.058521  | 0.8366469 | 0.1007675  | 0.9589976      | 0.174049    | 0.07256467 |
| <b>RNA – CV</b>       | size=31<br>decay=0.1   | 0.3929778 | 0.1942492  | 0.924422       | 0.3355136   | 0.2518463  |
| <b>RNA – Hold-out</b> | size=5<br>decay=0.1    | 0.3834227 | 0.1957721  | 0.9219961      | 0.338144    | 0.2558137  |



Para o experimento de agrupamento:

Lista de Clusters gerados para a base veículos:

Clustering vector:

```
[1] 9 5 4 5 2 3 9 6 10 7 6 10 2 2 7 4 10 7 4 4 2 10 5 5 4 2 10 1 8 2 2 5 5 4
[35] 9 2 2 9 1 2 4 10 2 9 4 2 6 2 2 9 6 5 4 5 3 6 7 2 4 10 8 6 6 2 7 5 2 1
[69] 5 8 4 1 7 10 2 8 9 2 4 2 2 4 6 5 5 3 2 10 7 10 1 9 8 2 2 4 5 2 8 2 9 5
[103] 6 10 8 1 8 5 2 7 5 2 5 10 2 2 8 1 2 7 6 2 5 10 2 2 5 5 9 8 8 5 7 6 4 9
[137] 2 6 10 5 9 2 4 2 9 1 9 5 5 6 7 7 5 4 5 3 5 6 5 2 6 8 9 6 4 1 5 4 2 6
[171] 4 3 5 4 2 9 7 5 5 6 4 10 5 10 8 7 9 9 4 9 7 5 2 3 9 2 1 2 2 2 2 9 1 4
[205] 10 2 9 2 2 4 9 6 5 4 6 9 2 1 2 10 1 2 7 2 6 5 1 6 4 2 10 10 2 4 6 2 9 2
[239] 1 2 9 9 2 1 2 6 9 9 4 10 6 1 2 5 2 4 5 5 8 1 10 7 10 10 7 2 2 9 10 10 2 7
[273] 5 5 3 5 2 6 4 9 7 2 2 1 2 9 5 2 8 6 6 9 2 6 3 5 8 6 7 7 4 2 9 6 2 2
[307] 4 4 8 8 7 2 5 8 2 6 6 7 10 1 1 3 2 8 9 2 7 2 5 9 5 7 1 2 3 1 2 4 5 5
[341] 5 6 6 1 1 3 4 5 5 6 8 2 2 2 7 7 2 4 2 7 4 8 9 2 2 8 2 6 2 9 7 9 2 9
[375] 10 1 1 6 10 1 6 3 2 1 5 2 7 2 9 1 2 5 5 2 7 9 3 5 9 6 4 5 4 5 8 5 6 5
[409] 2 9 9 10 2 6 2 4 5 5 2 6 6 4 5 2 7 6 1 9 4 6 1 4 5 2 4 6 6 2 7 8 8 2
[443] 10 7 3 2 7 4 4 5 9 9 9 5 8 2 6 5 1 7 6 8 9 2 7 2 2 4 1 6 6 1 1 1 2 4
[477] 1 2 7 2 4 1 9 5 2 2 1 9 5 4 1 9 2 1 3 9 4 2 10 1 1 1 6 5 8 1 4 5 9 4
[511] 2 9 1 9 6 6 5 1 2 6 7 6 5 9 1 9 7 4 9 2 7 3 4 10 6 9 1 10 8 3 9 6 9 9
[545] 5 2 6 5 9 4 10 6 7 2 8 7 3 6 2 1 1 9 4 5 9 7 4 9 2 6 1 2 2 2 4 4 1 8
[579] 9 10 2 6 1 4 4 10 1 6 5 1 2 10 2 5 6 7 7 10 7 6 9 5 1 9 7 1 6 10 6 2 4 2
[613] 10 9 10 6 6 10 6 4 1 2 7 1 6 4 5 5 7 6 6 4 2 8 6 6 5 2 7 1 8 9 8 7 2 2
[647] 9 5 4 5 4 6 9 7 9 10 5 2 6 7 6 8 2 9 6 9 7 4 2 2 8 7 8 5 5 1 2 8 2 9
[681] 9 9 2 3 9 2 6 3 10 8 9 2 8 2 7 6 5 9 5 9 1 4 9 6 4 4 5 10 2 7 1 8 8 1
[715] 9 1 7 9 1 4 9 1 2 8 2 6 8 9 10 4 1 3 5 3 6 5 8 1 1 5 1 5 6 8 2 2 6 2
[749] 2 7 7 10 9 5 6 10 8 2 2 2 7 8 6 4 4 2 9 5 8 5 2 8 8 2 6 7 8 3 4 6 8 6
[783] 7 1 10 2 1 10 1 9 2 6 4 8 5 6 10 4 4 5 5 6 9 7 8 2 2 4 2 6 4 6 9 2 2 2
[817] 2 1 7 7 6 1 5 8 1 2 7 1 6 2 5 9 4 9 6 3 6 9 2 6 6 9 2 1 6 10
```

within cluster sum of squares by cluster:

```
[1] 90565.80 175854.74 115198.75 110896.81 129929.63 121479.98 128496.26 86449.16 319987.98 46182.38
(between_ss / total_ss = 95.7 %)
```

Available components:

```
[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```

10 primeiras linhas do cluster correspondente:

|    | skewMaxis | skewmaxis | kurtmaxis | kurtMaxis | HolRa | tipo | dados\$tipo |
|----|-----------|-----------|-----------|-----------|-------|------|-------------|
| 1  | 70        | 6         | 16        | 187       | 197   | van  | van         |
| 2  | 72        | 9         | 14        | 189       | 199   | van  | van         |
| 3  | 73        | 14        | 9         | 188       | 196   | saab | saab        |
| 4  | 63        | 6         | 10        | 199       | 207   | van  | van         |
| 5  | 127       | 9         | 11        | 180       | 183   | bus  | bus         |
| 6  | 85        | 5         | 9         | 181       | 183   | bus  | bus         |
| 7  | 66        | 13        | 1         | 200       | 204   | bus  | bus         |
| 8  | 67        | 3         | 3         | 193       | 202   | van  | van         |
| 9  | 64        | 2         | 14        | 200       | 208   | van  | van         |
| 10 | 64        | 4         | 14        | 195       | 204   | saab | saab        |

Para o experimento de regras de associação:

### Musculação

```
> ### Extrai as regras
> rules <- apriori(dados, parameter = list(supp = 0.4, conf = 0.8, target = "rules"))
Apriori

Parameter specification:
confidence minval smax arem originalsupport maxtime support minlen maxlen target ext
0.8 0.1 1 none FALSE TRUE 5 0.4 1 10 rules TRUE

Algorithmic control:
filter tree heap memopt load sort verbose
0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 10

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[11 item(s), 26 transaction(s)] done [0.00s].
sorting and recoding items ... [5 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 done [0.00s].
writing ... [5 rule(s)] done [0.00s].
creating s4 object ... done [0.00s].
> inspect(rules)
  lhs      rhs      support  confidence coverage lift    count
[1] {}      => {LegPress} 0.8076923 0.8076923 1.0000000 1.000000 21
[2] {Esteira} => {Extensor} 0.4230769 0.9166667 0.4615385 1.833333 11
[3] {Extensor} => {Esteira} 0.4230769 0.8461538 0.5000000 1.833333 11
[4] {Extensor} => {Bicicleta} 0.4615385 0.9230769 0.5000000 1.714286 12
[5] {Bicicleta} => {Extensor} 0.4615385 0.8571429 0.5384615 1.714286 12
```

## APÊNDICE 8 – DEEP LEARNING

### A – ENUNCIADO

#### 1 Classificação de Imagens (CNN)

Implementar o exemplo de classificação de objetos usando a base de dados CIFAR10 e a arquitetura CNN vista no curso.

#### 2 Detector de SPAM (RNN)

Implementar o detector de spam visto em sala, usando a base de dados SMS Spam e a arquitetura de RNN vista no curso.

#### 3 Gerador de Dígitos Fake (GAN)

Implementar o gerador de dígitos *fake* usando a base de dados MNIST e arquitetura GAN vista no curso.

#### 4 Tradutor de Textos (Transformer)

Implementar o tradutor de texto do português para o inglês, usando a base de dados e a arquitetura Transformer vista no curso.

### B – RESOLUÇÃO

#### 1. CNN

```
cifar10 = tf.keras.datasets.cifar10
(x_train, y_train), (x_test, y_test) = cifar10.load_data()
# Imagens em pixels de 0 -255
# / 255.0 transforma em 0 -1
x_train, x_test = x_train / 255.0, x_test / 255.0

# O dado y é a classe a qual faz parte
# O flatten torna os dados vetorizados
y_train, y_test = y_train.flatten(), y_test.flatten()

# Dimensão dos dados
print("x_train.shape: ", x_train.shape)
print("y_train.shape: ", y_train.shape)
print("x_test.shape: ", x_test.shape)
print("y_test.shape: ", y_test.shape)
x_train.shape: (50000, 32, 32, 3)
y_train.shape: (50000,)
```



```
x_test.shape: (10000, 32, 32, 3)
y_test.shape: (10000,)
```

```
K = len(set(y_train))
# Aqui começa o Estágio 1
i = Input(shape=x_train[0].shape)
x = Conv2D(32, (3, 3), strides=2, activation="relu")(i)
x = Conv2D(64, (3, 3), strides=2, activation="relu")(x)
x = Conv2D(128, (3, 3), strides=2, activation="relu")(x)
# Todas as imagens são do mesmo tamanho, não precisa de Global Pooling
x = Flatten()(x)
# Aqui começa o Estágio 2
x = Dropout(0.5)(x)
x = Dense(1024, activation="relu")(x)
x = Dropout(0.2)(x)
x = Dense(K, activation="softmax")(x)
# Model ( lista entrada, lista saída)
model = Model(i, x)
model.summary()
# Compilar o modelo
model.compile(optimizer="adam",
loss="sparse_categorical_crossentropy", metrics=["accuracy"])
# Treinar o modelo
r= model.fit(x_train, y_train, validation_data=(x_test, y_test),
epochs=15)
```

Epoch 1/15

```
1563/1563 _____ 60s 36ms/step - accuracy:
0.3427 - loss: 1.7828 - val_accuracy: 0.5221 - val_loss: 1.3203
```

Epoch 2/15

```
1563/1563 _____ 78s 34ms/step - accuracy:
0.5172 - loss: 1.3323 - val_accuracy: 0.5913 - val_loss: 1.1452
```

Epoch 3/15

```
1563/1563 _____ 54s 34ms/step - accuracy:
0.5790 - loss: 1.1829 - val_accuracy: 0.6283 - val_loss: 1.0554
```

Epoch 4/15

```
1563/1563 _____ 56s 36ms/step - accuracy:
0.6117 - loss: 1.0859 - val_accuracy: 0.6361 - val_loss: 1.0310
```

Epoch 5/15

```
1563/1563 _____ 55s 35ms/step - accuracy:
0.6381 - loss: 1.0104 - val_accuracy: 0.6623 - val_loss: 0.9676
```

Epoch 6/15

```
1563/1563 _____ 54s 35ms/step - accuracy:
0.6621 - loss: 0.9506 - val_accuracy: 0.6569 - val_loss: 0.9798
```

Epoch 7/15

```
1563/1563 _____ 55s 35ms/step - accuracy:
0.6856 - loss: 0.8885 - val_accuracy: 0.6764 - val_loss: 0.9037
```

Epoch 8/15

```
1563/1563 _____ 82s 35ms/step - accuracy:
0.7034 - loss: 0.8400 - val_accuracy: 0.6791 - val_loss: 0.9235
```

Epoch 9/15

```
1563/1563 _____ 56s 36ms/step - accuracy:
0.7148 - loss: 0.8002 - val_accuracy: 0.7059 - val_loss: 0.8652
```

Epoch 10/15

```
1563/1563 _____ 54s 35ms/step - accuracy:
0.7220 - loss: 0.7780 - val_accuracy: 0.7042 - val_loss: 0.8417
```

Epoch 11/15

```
1563/1563 _____ 83s 36ms/step - accuracy:
0.7383 - loss: 0.7302 - val_accuracy: 0.7029 - val_loss: 0.8430
```

Epoch 12/15

**1563/1563** ————— **54s** 35ms/step - accuracy:  
0.7491 - loss: 0.7052 - val\_accuracy: 0.7108 - val\_loss: 0.8478

Epoch 13/15

...

Epoch 14/15

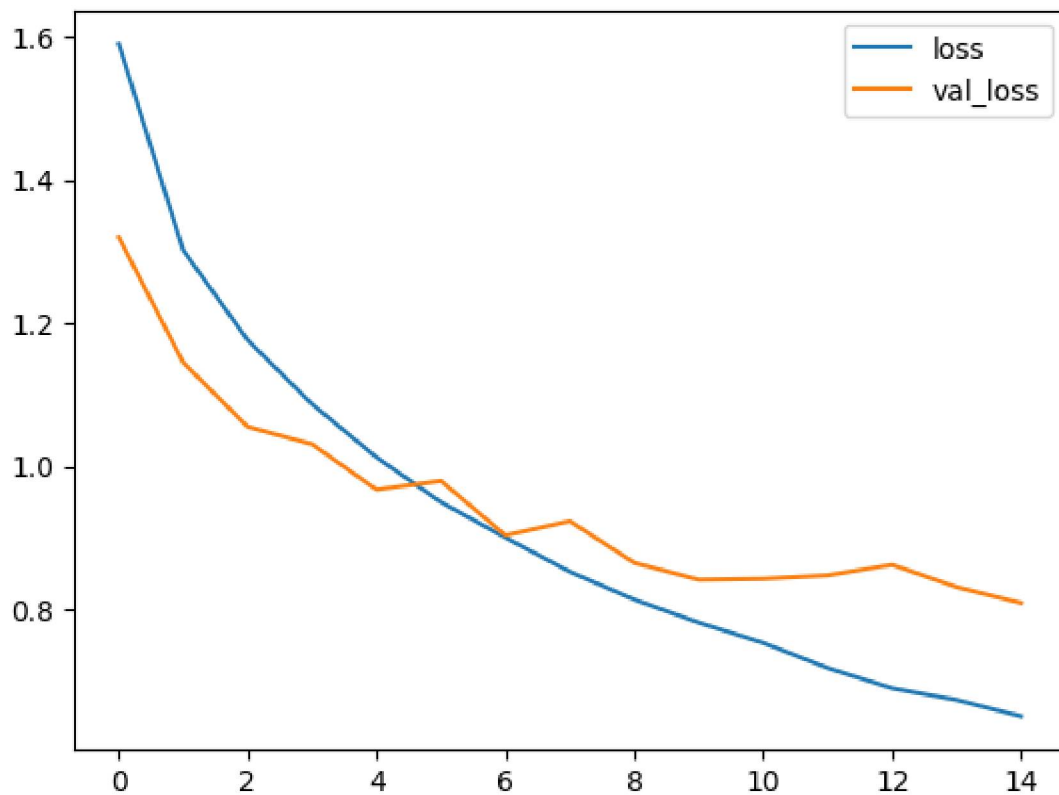
**1563/1563** ————— **84s** 36ms/step - accuracy:  
0.7663 - loss: 0.6589 - val\_accuracy: 0.7135 - val\_loss: 0.8314

Epoch 15/15

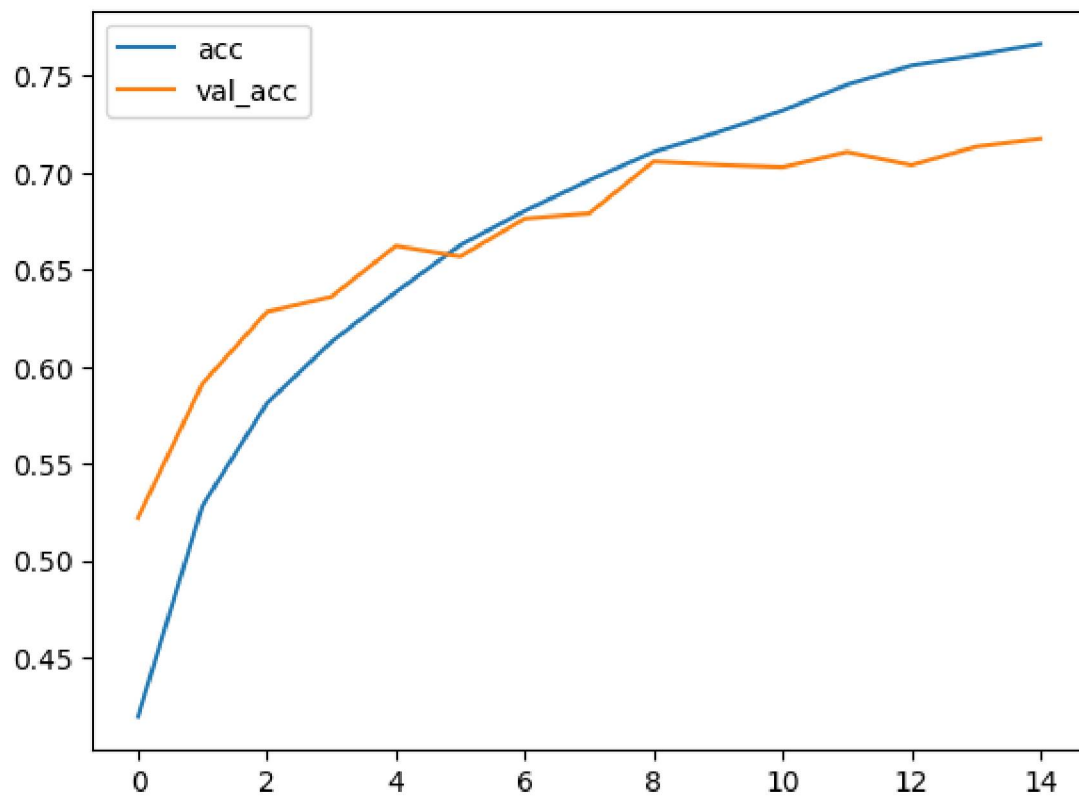
**1563/1563** ————— **80s** 35ms/step - accuracy:  
0.7729 - loss: 0.6298 - val\_accuracy: 0.7177 - val\_loss: 0.8092

```
#Plotar a função de perda, treino e validação
plt.plot(r.history["loss"], label="loss")
plt.plot(r.history["val_loss"], label="val_loss")
plt.legend()
plt.show()

#Plotar a acurácia, treino e validação
plt.plot(r.history["accuracy"], label="acc")
plt.plot(r.history["val_accuracy"], label="val_acc")
plt.legend()
plt.show()
```







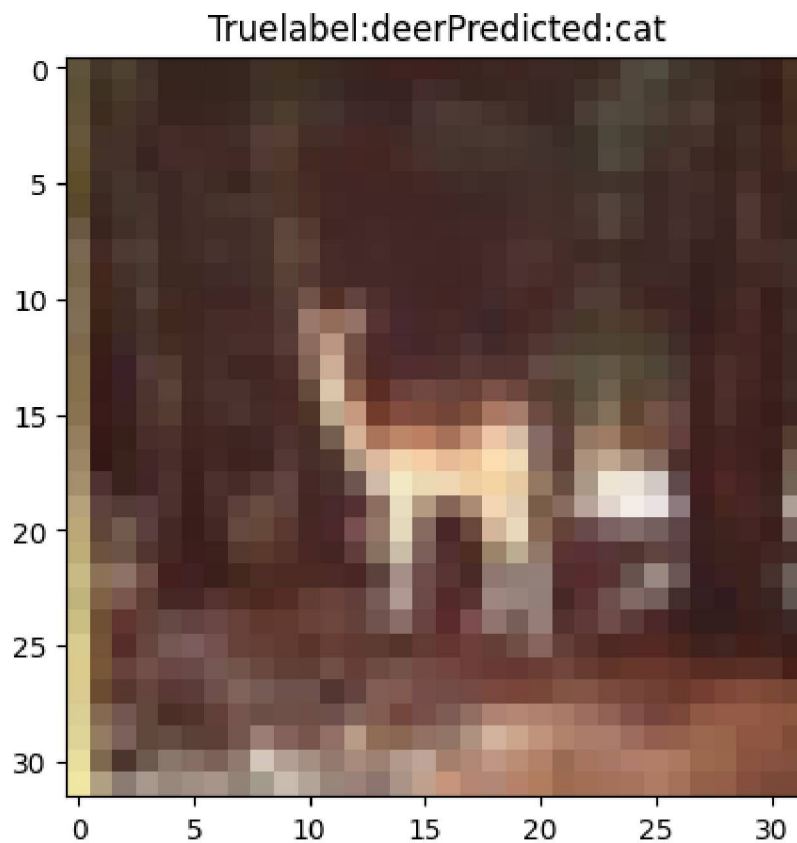
```
# Efetuar predições na base de teste
# argmax é usado pois a função de ativação da saída é softmax
# argmax pega o neurônio que deu o maior resultado, isto é,
# a maior probabilidade de saída
y_pred= model.predict(x_test).argmax(axis=1)
#Mostrar a matriz de confusão
cm = confusion_matrix(y_test, y_pred)
plot_confusion_matrix(conf_mat=cm, figsize=(7, 7),
show_normed=True)
```

|            |   |                 |               |               |               |               |               |               |               |               |               |
|------------|---|-----------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
| true label | 0 | 720<br>(0.72)   | 41<br>(0.04)  | 46<br>(0.05)  | 11<br>(0.01)  | 20<br>(0.02)  | 8<br>(0.01)   | 10<br>(0.01)  | 17<br>(0.02)  | 81<br>(0.08)  | 46<br>(0.05)  |
|            |   | 9<br>(0.01)     | 858<br>(0.86) | 3<br>(0.00)   | 3<br>(0.00)   | 3<br>(0.00)   | 2<br>(0.00)   | 13<br>(0.01)  | 1<br>(0.00)   | 15<br>(0.01)  | 93<br>(0.09)  |
|            | 2 | 56<br>(0.06)    | 10<br>(0.01)  | 559<br>(0.56) | 45<br>(0.04)  | 118<br>(0.12) | 59<br>(0.06)  | 88<br>(0.09)  | 41<br>(0.04)  | 10<br>(0.01)  | 14<br>(0.01)  |
|            |   | 19<br>(0.02)    | 13<br>(0.01)  | 65<br>(0.07)  | 437<br>(0.44) | 78<br>(0.08)  | 200<br>(0.20) | 108<br>(0.11) | 48<br>(0.05)  | 10<br>(0.01)  | 22<br>(0.02)  |
|            | 4 | 15<br>(0.01)    | 4<br>(0.00)   | 54<br>(0.05)  | 43<br>(0.04)  | 681<br>(0.68) | 28<br>(0.03)  | 84<br>(0.08)  | 71<br>(0.07)  | 12<br>(0.01)  | 8<br>(0.01)   |
|            |   | 9<br>(0.01)     | 6<br>(0.01)   | 43<br>(0.04)  | 133<br>(0.13) | 54<br>(0.05)  | 625<br>(0.62) | 54<br>(0.05)  | 60<br>(0.06)  | 4<br>(0.00)   | 12<br>(0.01)  |
|            | 6 | 4<br>(0.00)     | 7<br>(0.01)   | 25<br>(0.03)  | 47<br>(0.05)  | 29<br>(0.03)  | 25<br>(0.03)  | 841<br>(0.84) | 8<br>(0.01)   | 7<br>(0.01)   | 7<br>(0.01)   |
|            |   | 12<br>(0.01)    | 4<br>(0.00)   | 27<br>(0.03)  | 24<br>(0.02)  | 57<br>(0.06)  | 38<br>(0.04)  | 13<br>(0.01)  | 812<br>(0.81) | 3<br>(0.00)   | 10<br>(0.01)  |
|            | 8 | 40<br>(0.04)    | 44<br>(0.04)  | 15<br>(0.01)  | 7<br>(0.01)   | 6<br>(0.01)   | 7<br>(0.01)   | 7<br>(0.01)   | 5<br>(0.01)   | 830<br>(0.83) | 39<br>(0.04)  |
|            |   | 18<br>(0.02)    | 96<br>(0.10)  | 8<br>(0.01)   | 14<br>(0.01)  | 7<br>(0.01)   | 3<br>(0.00)   | 7<br>(0.01)   | 12<br>(0.01)  | 21<br>(0.02)  | 814<br>(0.81) |
|            |   | predicted label |               |               |               |               |               |               |               |               |               |

```

labels= ["airplane", "automobile", "bird", "cat", "deer", "dog",
"frog", "horse", "ship", "truck"]
misclassified= np.where(y_pred!= y_test)[0]
i= np.random.choice(misclassified)
plt.imshow(x_test[i], cmap="gray")
plt.title("Truelabel:%sPredicted:%s" % (labels[y_test[i]],
labels[y_pred[i]]))

```



## 2. RNN

```
# carrega e arruma a base
# !wget http://www.razer.net.br/datasets/spam.csv
df = pd.read_csv("spam.csv", encoding="ISO-8859-1")
df.head()
df = df.drop(["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"], axis=1)
df.columns = ["labels", "data"]
df["b_labels"] = df["labels"].map({ "ham": 0, "spam": 1})
y = df["b_labels"].values
# Separa a base em treino e teste
x_train, x_test, y_train, y_test = train_test_split(df["data"], y,
test_size=0.33)
# Número máximo de palavras para considerar
# São consideradas as mais frequentes, as demais são
# ignoradas
num_words = 20000
tokenizer = Tokenizer(num_words=num_words)
tokenizer.fit_on_texts(x_train)
sequences_train = tokenizer.texts_to_sequences(x_train)
sequences_test = tokenizer.texts_to_sequences(x_test)
word2index = tokenizer.word_index
V = len(word2index)
print("%s tokens" % V)
```

7142 tokens

```
# Acerta o tamanho das sequências (padding)
data_train = pad_sequences(sequences_train) # usa o tamanho da maior
seq.
```

```
T = data_train.shape[1] # tamanho da sequência
data_test = pad_sequences(sequences_test, maxlen=T)
print("data_train.shape: ", data_train.shape)
print("data_test.shape: ", data_test.shape)
```

```
data_train.shape: (3733, 162)
```

```
data_test.shape: (1839, 162)
```

```
# Define o modelo
D = 20 # tamanho do embedding, hiperparâmetro que pode ser escolhido
M = 5 # tamanho do hidden state, quantidade de unidades LSTM
i = Input(shape=(T,)) # Entra uma frase inteira
x = Embedding(V+1, D)(i)
x = LSTM(M)(x)
x = Dense(1, activation="sigmoid")(x) # Sigmoide pois só tem 2 valores
model = Model(i, x)
model.summary()
# Compila e treina o modelo
model.compile(loss="binary_crossentropy", optimizer="adam",
metrics=["accuracy"])
epochs = 5
r = model.fit(data_train, y_train, epochs=epochs,
validation_data=(data_test,
y_test))
```

```
Epoch 1/5
```

```
117/117 ----- 6s 12ms/step - accuracy: 0.8671 -
loss: 0.5510 - val_accuracy: 0.8989 - val_loss: 0.3147
```

```
Epoch 2/5
```

```
117/117 ----- 1s 10ms/step - accuracy: 0.9537 -
loss: 0.2485 - val_accuracy: 0.9565 - val_loss: 0.1832
```

```
Epoch 3/5
```

```
117/117 ----- 1s 10ms/step - accuracy: 0.9798 -
loss: 0.1462 - val_accuracy: 0.9750 - val_loss: 0.1300
```

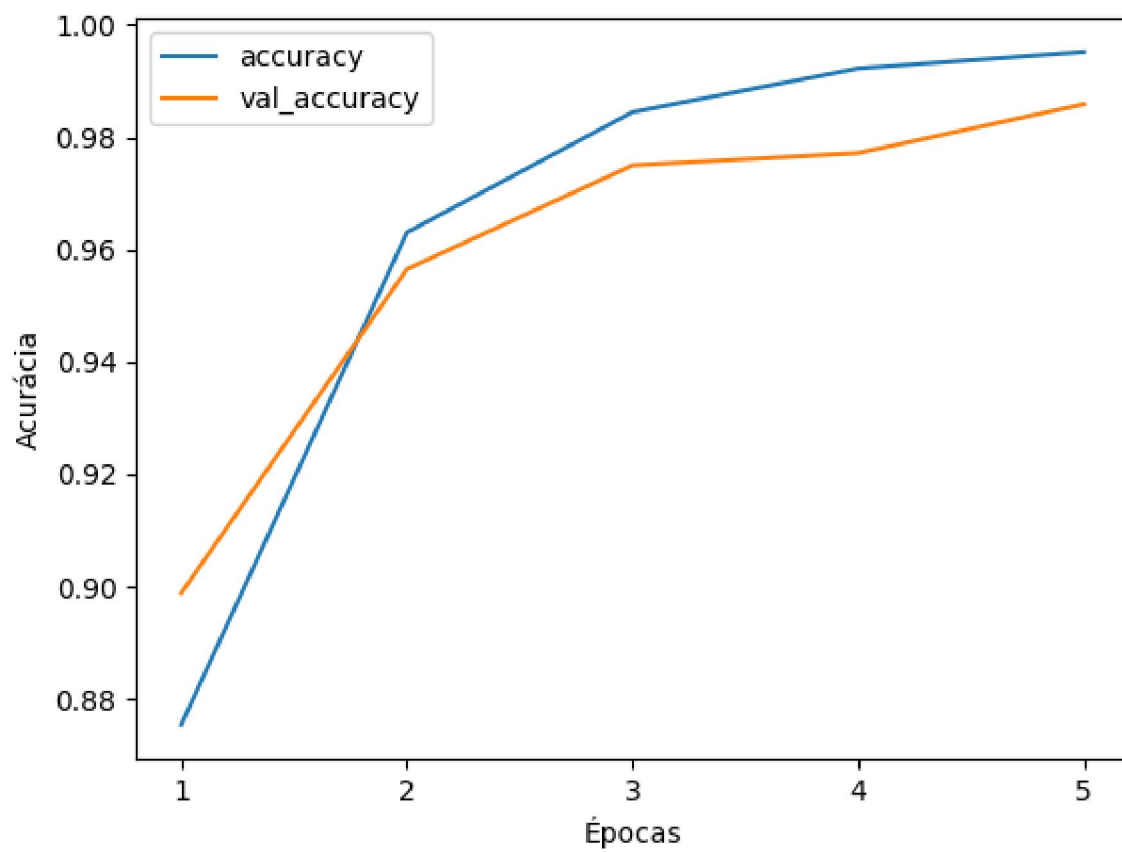
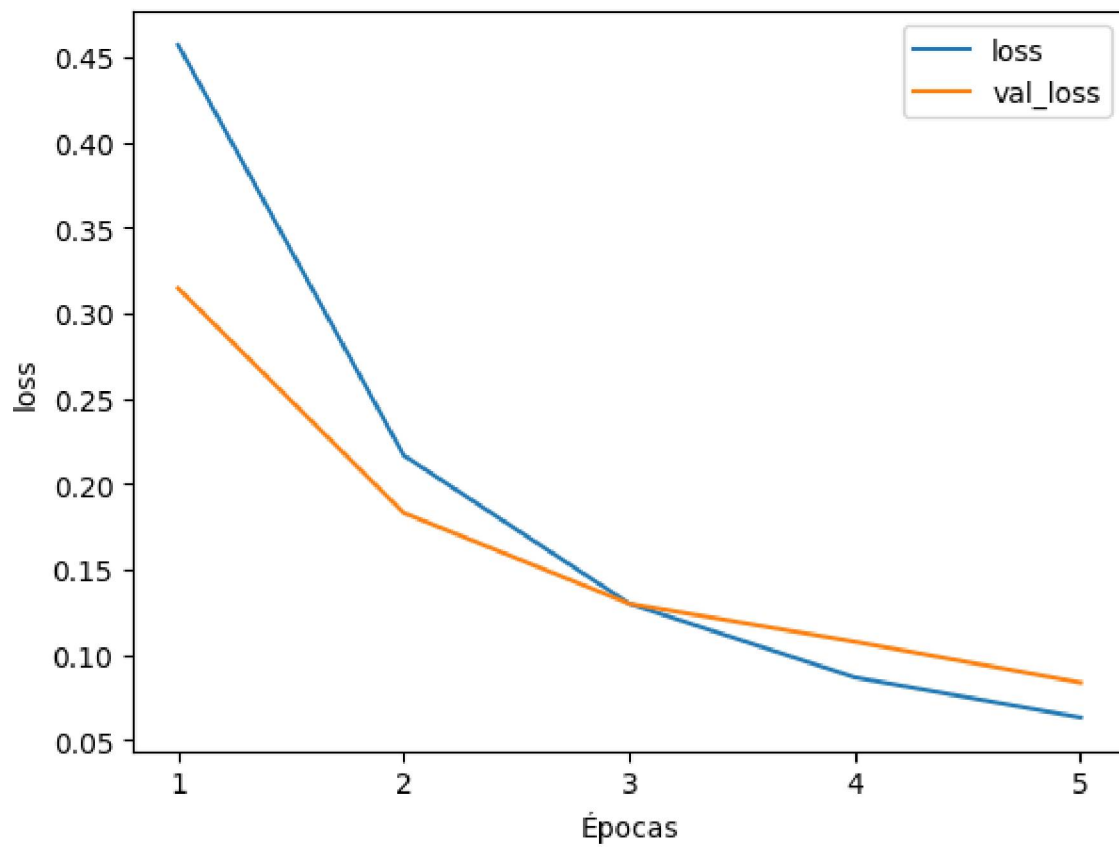
```
Epoch 4/5
```

```
117/117 ----- 1s 9ms/step - accuracy: 0.9903 -
loss: 0.0936 - val_accuracy: 0.9772 - val_loss: 0.1077
```

```
Epoch 5/5
```

```
117/117 ----- 1s 9ms/step - accuracy: 0.9966 -
loss: 0.0651 - val_accuracy: 0.9859 - val_loss: 0.0838
```

```
# Plota função de perda e acurácia
plt.plot(r.history["loss"], label="loss")
plt.plot(r.history["val_loss"], label="val_loss")
plt.xlabel("Épocas")
plt.ylabel("loss")
plt.xticks(np.arange(0, epochs, step=1), labels=range(1, epochs+1))
plt.legend()
plt.show()
plt.plot(r.history["accuracy"], label="accuracy")
plt.plot(r.history["val_accuracy"], label="val_accuracy")
plt.xlabel("Épocas")
plt.ylabel("Acurácia")
plt.xticks(np.arange(0, epochs, step=1), labels=range(1, epochs+1))
plt.legend()
plt.show()
```



```
# Efetua a predição de um texto novo
#texto = "Hi, my name is Razer and want to tell you something."
```



```

texto = "Is your car dirty? Discover our new product. Free for all.
Click the link."
seq_texto = tokenizer.texts_to_sequences([texto]) # Tokeniza
data_texto = pad_sequences(seq_texto, maxlen=T)   # Padding
pred = model.predict(data_texto) # Predição
print(pred)
print ("SPAM" if pred >= 0.5 else "OK")

[[0.6847701]]
SPAM

```

### 3. GAN

```

# Carregar a base de dados
(train_images, train_labels), (_, _) = tf.keras.datasets.mnist.load_data()

# Normalização
train_images = train_images.reshape(train_images.shape[0], 28, 28, 1).astype('float32')
train_images = (train_images-127.5) / 127.5 # Normaliza entre [-1, 1]

# Gera o banco em partes e randomiza
BUFFER_SIZE = 60000
BATCH_SIZE = 256

# Cria o dataset (from_tensor_slices)
# Randomiza (shuffle)
# Combina elementos consecutivos em lotes (batch)
train_dataset = tf.data.Dataset.from_tensor_slices(train_images).shuffle(BUFFER_SIZE).batch(BATCH_SIZE)

# Cria o GERADOR
def make_generator_model():
    model = tf.keras.Sequential()
    model.add(layers.Dense(7*7*256, use_bias=False, input_shape=(100,)))
    model.add(layers.BatchNormalization())
    model.add(layers.LeakyReLU())

    model.add(layers.Reshape((7, 7, 256)))
    assert model.output_shape == (None, 7, 7, 256)
    # Note: None is the batch size

    model.add(layers.Conv2DTranspose(128, (5, 5), strides=(1, 1), padding='same', use_bias=False))
    assert model.output_shape == (None, 7, 7, 128)

    model.add(layers.BatchNormalization())
    model.add(layers.LeakyReLU())
    model.add(layers.Conv2DTranspose(64, (5, 5), strides=(2, 2), padding='same', use_bias=False))
    assert model.output_shape == (None, 14, 14, 64)

    model.add(layers.BatchNormalization())
    model.add(layers.LeakyReLU())

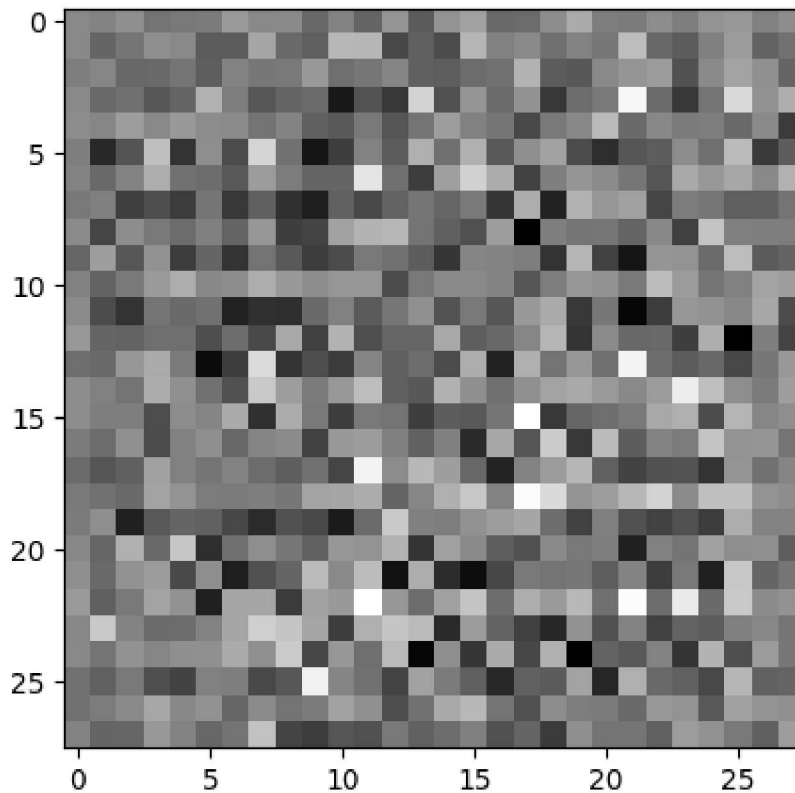
```

```

        model.add(layers.Conv2DTranspose(1, (5, 5), strides=(2, 2),
padding='same', use_bias=False, activation='tanh'))
        assert model.output_shape == (None, 28, 28, 1)

        return model
generator = make_generator_model()
noise = tf.random.normal([1, 100])
generated_image = generator(noise, training=False)
plt.imshow(generated_image[0, :, :, 0], cmap='gray')

```



```

# Cria o DISCRIMINADOR
def make_discriminator_model():
    model = tf.keras.Sequential()
    model.add(layers.Conv2D(64, (5, 5), strides=(2, 2), padding='same',
input_shape=[28, 28, 1]))

    model.add(layers.LeakyReLU())
    model.add(layers.Dropout(0.3))

    model.add(layers.Conv2D(128, (5, 5), strides=(2, 2), padding='same'))
    model.add(layers.LeakyReLU())
    model.add(layers.Dropout(0.3))

    model.add(layers.Flatten())
    model.add(layers.Dense(1))

    return model
discriminator = make_discriminator_model()
decision = discriminator(generated_image)
print (decision)
super().__init__(activity_regularizer=activity_regularizer, **kwargs)

tf.Tensor([[ -0.00226097]], shape=(1, 1), dtype=float32)

```



```

# Define as funções de perda
cross_entropy = tf.keras.losses.BinaryCrossentropy(from_logits=True)

# Perda do DISCRIMINADOR
def discriminator_loss(real_output, fake_output):
    real_loss = cross_entropy(tf.ones_like(real_output), real_output)
    fake_loss = cross_entropy(tf.zeros_like(fake_output), fake_output)
    total_loss = real_loss + fake_loss
    return total_loss

# Perda do GERADOR
def generator_loss(fake_output):
    return cross_entropy(tf.ones_like(fake_output), fake_output)

# Cria os otimizadores para o gerador e discriminador
generator_optimizer = tf.keras.optimizers.Adam(1e-4)
discriminator_optimizer = tf.keras.optimizers.Adam(1e-4)
# Cria checkpoints para salvar modelos ao longo do tempo
# Úteis em tarefas longas, para se recuperar de um desligamento
checkpoint_dir = './training_checkpoints'
checkpoint_prefix = os.path.join(checkpoint_dir, "ckpt")
checkpoint = tf.train.Checkpoint(generator_optimizer=generator_optimizer,
                                discriminator_optimizer=discriminator_optimizer,
                                generator=generator,
                                discriminator=discriminator)

# Configura o Loop de treinamento
EPOCHS = 100
noise_dim = 100
num_examples_to_generate = 16

# You will reuse this seed overtime (so it's easier)
# to visualize progress in the animated GIF)
seed = tf.random.normal([num_examples_to_generate, noise_dim])
# Função que faz um passo de treinamento
# É uma `tf.function`, que compila essa função
@tf.function
def train_step(images):
    noise= tf.random.normal([BATCH_SIZE, noise_dim])

    with tf.GradientTape() as gen_tape, tf.GradientTape() as disc_tape:
        generated_images= generator(noise, training=True)

        real_output= discriminator(images, training=True)
        fake_output= discriminator(generated_images, training=True)

        gen_loss= generator_loss(fake_output)
        disc_loss= discriminator_loss(real_output, fake_output)

        gradients_of_generator= gen_tape.gradient(gen_loss,
generator.trainable_variables)
        gradients_of_discriminator= disc_tape.gradient(disc_loss,
discriminator.trainable_variables)

        generator_optimizer.apply_gradients(zip(gradients_of_generator,
generator.trainable_variables))

```

```

discriminator_optimizer.apply_gradients(zip(gradients_of_discriminator,
discriminator.trainable_variables))
# Treinamento completo/laço
def train(dataset, epochs):
    for epoch in range(epochs):
        start = time.time()

        for image_batch in dataset:
            train_step(image_batch)

        # Produce images for the GIF as you go
        display.clear_output(wait=True)
        generate_and_save_images(generator, epoch+ 1, seed)

        # Save the model every 15 epochs
        if(epoch+ 1) % 15 == 0:
            checkpoint.save(file_prefix= checkpoint_prefix)

            print ('Time for epoch{} is{} sec'.format(epoch+ 1,
time.time()-start))

        # Generate after the final epoch
        display.clear_output(wait=True)
        # Gerar e salvar imagens
def generate_and_save_images(model, epoch, test_input):
    # Notice `training` is set to False.
    # This is so all layers run in inference mode (batchnorm).
    predictions = model(test_input, training=False)

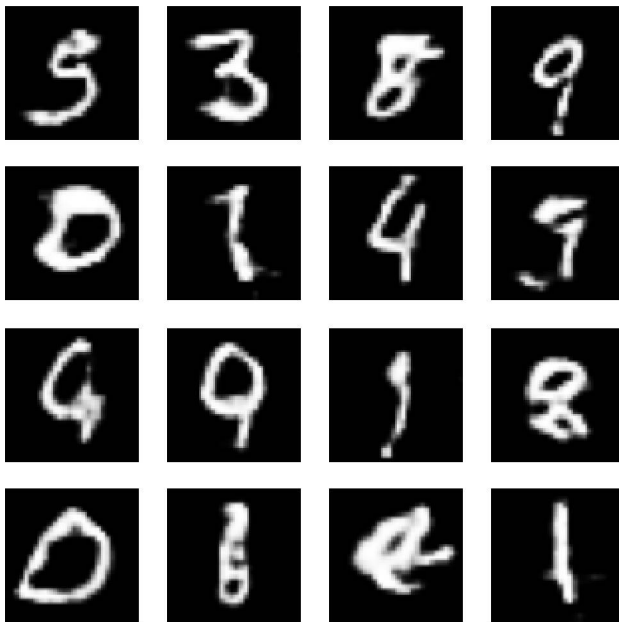
    fig = plt.figure(figsize=(4, 4))

    for i in range(predictions.shape[0]):
        plt.subplot(4, 4, i+1)
        plt.imshow(predictions[i, :, :, 0] * 127.5 + 127.5, cmap='gray')
        plt.axis('off')

    plt.savefig('image_at_epoch_{:04d}.png'.format(epoch))
    plt.show()

    # Treinar o modelo e restaurar o último ponto de verificação
    train(train_dataset, EPOCHS)
    checkpoint.restore(tf.train.latest_checkpoint(checkpoint_dir))

```

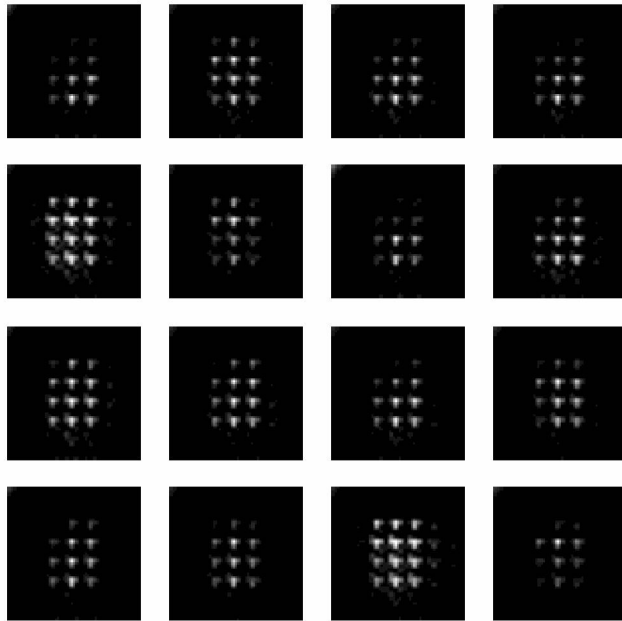


```
# Criar um GIF
# Display a single image using the epoch number
def display_image(epoch_no):
    return PIL.Image.open('image_at_epoch_{:04d}.png'.format(epoch_no))

display_image(EPOCHS)

anim_file= 'dcgan.gif'

with imageio.get_writer(anim_file, mode='I') as writer:
    filenames= glob.glob('image*.png')
    filenames= sorted(filenames)
    for filename in filenames:
        image= imageio.imread(filename)
        writer.append_data(image)
        image= imageio.imread(filename)
        writer.append_data(image)
import tensorflow_docs.vis.embed as embed
embed.embed_file(anim_file)
```



#### 4. Transformer

```
examples, metadata= tfds.load('ted_hrlr_translate/pt_to_en',
with_info=True, as_supervised=True)

train_examples, val_examples= examples['train'], examples['validation']
#Verificar o dataset
for pt_examples, en_examples in train_examples.batch(3).take(1):
    for pt in pt_examples.numpy():
        print(pt.decode('utf-8'))
    print()

for en in en_examples.numpy():
    print(en.decode('utf-8'))
```

e quando melhoramos a procura , tiramos a única vantagem da impressão , que é a serendipidade .

mas e se estes fatores fossem ativos ?

mas eles não tinham a curiosidade de me testar .

and when you improve searchability , you actually take away the one advantage of print , which is serendipity .

but what if it were active ?

but they did n't test for curiosity .

```
# Tokenização e Destokenização do texto
model_name = "ted_hrlr_translate_pt_en_converter"
tf.keras.utils.get_file(f"{model_name}.zip",
f"https://storage.googleapis.com/download.tensorflow.org/models/{model_name}.zip", cache_dir='.', cache_subdir='', extract=True)
```

```

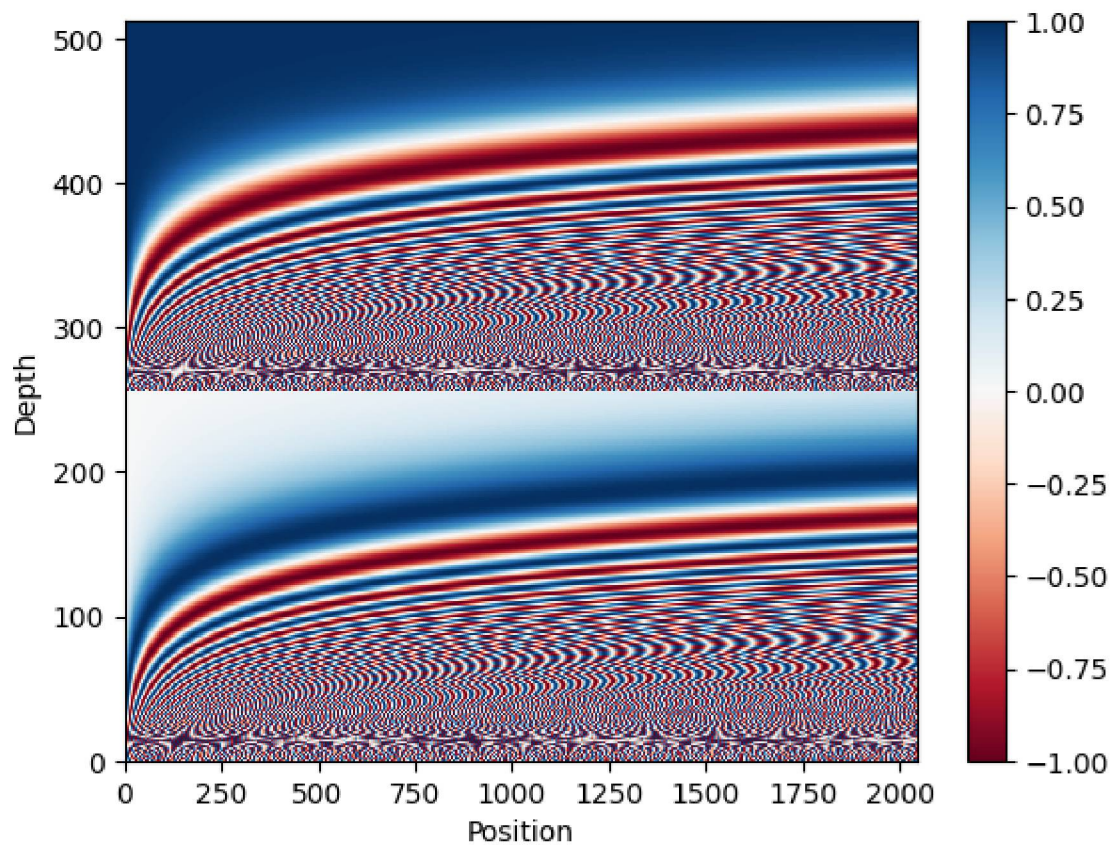
# Tem 2 tokenizers: um ptoutro em en
# tokenizers.entokeniza e detokeniza
tokenizers= tf.saved_model.load(model_name)
# PIPELINE DE ENTRADA
# Codificar/tokenizar lotes de texto puro
def tokenize_pairs(pt, en):
    pt= tokenizers.pt.tokenize(pt)
    # Converte ragged(irregular, tamvariável) para dense
    # Faz padding com zeros.
    pt= pt.to_tensor()
    en= tokenizers.en.tokenize(en)
    # ragged-> dense
    en= en.to_tensor()
    return pt, en
# Pipeline simples: processa, embaralha, agrupa os dados, prefetch
# Datasets de entrada terminam com prefetch
BUFFER_SIZE = 20000
BATCH_SIZE = 64
def make_batches(ds):
    return(
        ds
        .cache()
        .shuffle(BUFFER_SIZE)
        .batch(BATCH_SIZE)
        .map(tokenize_pairs, num_parallel_calls=tf.data.AUTOTUNE)
        .prefetch(tf.data.AUTOTUNE))

train_batches= make_batches(train_examples)
val_batches= make_batches(val_examples)
# CODIFICAÇÃO POSICIONAL
def get_angles(pos, i, d_model):
    angle_rates = 1 / np.power(10000, (2 * (i//2)) / np.float32(d_model))
    return pos * angle_rates

def positional_encoding(position, d_model):
    angle_rads = get_angles(np.arange(position)[:, np.newaxis],
                             np.arange(d_model)[np.newaxis, :], d_model)
    # sin em índices pares no array; 2i
    angle_rads[:, 0::2] = np.sin(angle_rads[:, 0::2])
    # cos em índices ímpares no array; 2i+1
    angle_rads[:, 1::2] = np.cos(angle_rads[:, 1::2])
    # newaxis, aumenta a dimensão [] -> [ [] ]
    pos_encoding = angle_rads[np.newaxis, ...]
    return tf.cast(pos_encoding, dtype=tf.float32)
# CODIFICAÇÃO POSICIONAL
n, d= 2048, 512
pos_encoding= positional_encoding(n, d)
print(pos_encoding.shape)
pos_encoding= pos_encoding[0]
# Arrumar as dimensões
pos_encoding= tf.reshape(pos_encoding, (n, d//2, 2))
pos_encoding= tf.transpose(pos_encoding, (2, 1, 0))
pos_encoding= tf.reshape(pos_encoding, (d, n))
plt.pcolormesh(pos_encoding, cmap='RdBu')
plt.ylabel('Depth')
plt.xlabel('Position')
plt.colorbar()
plt.show()

```





```
# Cria uma máscara de 0 e 1, 0 para quando há valor e 1 quando não há
def create_padding_mask(seq):
    seq= tf.cast(tf.math.equal(seq, 0), tf.float32)
    # addextra dimensionstoaddthepadding
    # totheattentionlogits.
    return seq[:, tf.newaxis, tf.newaxis, :] # (batch_size, 1, 1, seq_len)
# Máscara futura, usada no decoder
def create_look_ahead_mask(size):
    # zera o triângulo inferior
    mask= 1 -tf.linalg.band_part(tf.ones((size, size)), -1, 0)
    return mask # (seq_len, seq_len)
# Função de Atenção
def scaled_dot_product_attention(q, k, v, mask):
    # Q K^T
    matmul_qk = tf.matmul(q, k, transpose_b=True) # (... , seq_len_q, seq_len_k)

    # converte matmul_qk para float32
    dk = tf.cast(tf.shape(k)[-1], tf.float32)

    # divide por sqrt(d_k)
    scaled_attention_logits = matmul_qk / tf.math.sqrt(dk)

    # Soma a máscara, e os valores faltantes serão um número próximo a
    -inf
    if mask is not None:
        scaled_attention_logits += (mask * -1e9)

    # softmax normaliza os dados, soman 1. // (... , seq_len_q, seq_len_k)
```

```

attention_weights = tf.nn.softmax(scaled_attention_logits, axis=-1)

output = tf.matmul(attention_weights, v) # (... , seq_len_q, depth_v)

return output, attention_weights
# Atenção Multi-cabeças
class MultiHeadAttention(tf.keras.layers.Layer):
    def __init__(self, d_model, num_heads):
        super(MultiHeadAttention, self).__init__()
        self.num_heads = num_heads
        self.d_model = d_model

        assert d_model % self.num_heads == 0

        self.depth = d_model // self.num_heads

        self.wq = tf.keras.layers.Dense(d_model)
        self.wk = tf.keras.layers.Dense(d_model)
        self.wv = tf.keras.layers.Dense(d_model)

        self.dense = tf.keras.layers.Dense(d_model)

    def split_heads(self, x, batch_size):
        x = tf.reshape(x, (batch_size, -1, self.num_heads, self.depth))
        return tf.transpose(x, perm=[0, 2, 1, 3])

    def call(self, v, k, q, mask):
        batch_size = tf.shape(q)[0]
        q = self.wq(q) # (batch_size, seq_len, d_model)
        k = self.wk(k) # (batch_size, seq_len, d_model)
        v = self.wv(v) # (batch_size, seq_len, d_model)

        q = self.split_heads(q, batch_size) # (batch_size, num_heads, seq_len_q, depth)
        k = self.split_heads(k, batch_size) # (batch_size, num_heads, seq_len_k, depth)
        v = self.split_heads(v, batch_size) # (batch_size, num_heads, seq_len_v, depth)

        # Calcula a atenção para cada cabeça (de forma matricial)
        # scaled_attention.shape== (batch_size, num_heads, seq_len_q, depth)
        # attention_weights.shape== (batch_size, num_heads, seq_len_q, seq_len_k)
        scaled_attention, attention_weights = scaled_dot_product_attention(q, k, v, mask)

        # Troca a dimensão 2 com 1, para acertar o num_heads
        # (batch_size, seq_len_q, num_heads, depth)
        scaled_attention = tf.transpose(scaled_attention, perm=[0, 2, 1, 3])

        # Concatena os valores em: (batch_size, seq_len_q, d_model)
        concat_attention = tf.reshape(scaled_attention, (batch_size, -1, self.d_model))

        output = self.dense(concat_attention) # (batch_size, seq_len_q, d_model)

```



```

        return output, attention_weights
def point_wise_feed_forward_network(d_model, dff):
    return tf.keras.Sequential([
        tf.keras.layers.Dense(dff, activation='relu'), # (batch_size,
seq_len, dff)
        tf.keras.layers.Dense(d_model) # (batch_size, seq_len, d_model)
    ])
class EncoderLayer(tf.keras.layers.Layer):
    def __init__(self, d_model, num_heads, dff, rate=0.1):
        super(EncoderLayer, self).__init__()

        self.mha = MultiHeadAttention(d_model, num_heads)
        self.ffn = point_wise_feed_forward_network(d_model, dff)

        self.layernorm1 = tf.keras.layers.LayerNormalization(epsilon=1e-6)
        self.layernorm2 = tf.keras.layers.LayerNormalization(epsilon=1e-6)

        self.dropout1 = tf.keras.layers.Dropout(rate)
        self.dropout2 = tf.keras.layers.Dropout(rate)

    def call(self, x, training, mask):
        attn_output, _ = self.mha(x, x, x, mask) # (batch_size,
input_seq_len, d_model)
        attn_output = self.dropout1(attn_output, training=training)
        out1 = self.layernorm1(x + attn_output) # (batch_size,
input_seq_len, d_model)

        ffn_output = self.ffn(out1) # (batch_size, input_seq_len, d_model)
        ffn_output = self.dropout2(ffn_output, training=training)
        out2 = self.layernorm2(out1 + ffn_output) # (batch_size,
input_seq_len, d_model)

        return out2
class DecoderLayer(tf.keras.layers.Layer):
    def __init__(self, d_model, num_heads, dff, rate=0.1):
        super(DecoderLayer, self).__init__()

        self.mha1 = MultiHeadAttention(d_model, num_heads)
        self.mha2 = MultiHeadAttention(d_model, num_heads)

        self.ffn = point_wise_feed_forward_network(d_model, dff)

        self.layernorm1 = tf.keras.layers.LayerNormalization(epsilon=1e-6)
        self.layernorm2 = tf.keras.layers.LayerNormalization(epsilon=1e-6)
        self.layernorm3 = tf.keras.layers.LayerNormalization(epsilon=1e-6)

        self.dropout1 = tf.keras.layers.Dropout(rate)
        self.dropout2 = tf.keras.layers.Dropout(rate)
        self.dropout3 = tf.keras.layers.Dropout(rate)

    def call(self, x, enc_output, training, look_ahead_mask,
padding_mask):
        # enc_output.shape == (batch_size, input_seq_len, d_model)

        # (batch_size, target_seq_len, d_model)
        attn1, attn_weights_block1 = self.mha1(x, x, x, look_ahead_mask)
        attn1 = self.dropout1(attn1, training=training)
        out1 = self.layernorm1(attn1 + x)

```

```

        # (batch_size, target_seq_len, d_model)
        attn2, attn_weights_block2 = self.mha2(enc_output, enc_output,
        out1, padding_mask)
        attn2 = self.dropout2(attn2, training=training)
        out2 = self.layernorm2(attn2 + out1) # (batch_size,
        target_seq_len, d_model)

        ffn_output = self.ffn(out2) # (batch_size, target_seq_len,
        d_model)
        ffn_output = self.dropout3(ffn_output, training=training)
        out3 = self.layernorm3(ffn_output + out2) # (batch_size,
        target_seq_len, d_model)

        return out3, attn_weights_block1, attn_weights_block2
class Encoder(tf.keras.layers.Layer):
    def __init__(self, num_layers, d_model, num_heads, dff,
        input_vocab_size, maximum_position_encoding, rate=0.1):
        super(Encoder, self).__init__()
        self.d_model = d_model
        self.num_layers = num_layers
        self.embedding = tf.keras.layers.Embedding(input_vocab_size,
        d_model)
        self.pos_encoding = positional_encoding(maximum_position_encoding,
        self.d_model)
        self.enc_layers = [EncoderLayer(d_model, num_heads, dff, rate) for _
        in range(num_layers)]
        self.dropout = tf.keras.layers.Dropout(rate)

    def call(self, x, training, mask):
        seq_len = tf.shape(x)[1]
        # adding embedding and position encoding.
        x = self.embedding(x) # (batch_size, input_seq_len, d_model)
        x *= tf.math.sqrt(tf.cast(self.d_model, tf.float32))
        x += self.pos_encoding[:, :seq_len, :]
        x = self.dropout(x, training=training)
        for i in range(self.num_layers):
            x = self.enc_layers[i](x, training, mask)
        return x # (batch_size, input_seq_len, d_model)
class Decoder(tf.keras.layers.Layer):
    def __init__(self, num_layers, d_model, num_heads, dff,
        target_vocab_size, maximum_position_encoding, rate=0.1):
        super(Decoder, self).__init__()

        self.d_model = d_model
        self.num_layers = num_layers

        self.embedding = tf.keras.layers.Embedding(target_vocab_size,
        d_model)
        self.pos_encoding = positional_encoding(maximum_position_encoding,
        d_model)

        self.dec_layers = [DecoderLayer(d_model, num_heads, dff, rate) for
        _ in range(num_layers)]
        self.dropout = tf.keras.layers.Dropout(rate)

    def call(self, x, enc_output, training, look_ahead_mask,
        padding_mask):

```

```

seq_len= tf.shape(x)[1]
attention_weights= {}

x= self.embedding(x) # (batch_size, target_seq_len, d_model)
x*= tf.math.sqrt(tf.cast(self.d_model, tf.float32))
x+= self.pos_encoding[:, :seq_len, :]

x= self.dropout(x, training=training)

for i in range(self.num_layers):
    x, block1, block2 = self.dec_layers[i](x, enc_output, training,
look_ahead_mask, padding_mask)
    attention_weights[f'decoder_layer{i+1}_block1'] = block1
    attention_weights[f'decoder_layer{i+1}_block2'] = block2

    # x.shape== (batch_size, target_seq_len, d_model)
    return x, attention_weights
class Transformer(tf.keras.Model):
    def __init__(self, num_layers, d_model, num_heads, dff,
input_vocab_size, target_vocab_size, pe_input, pe_target, rate=0.1):
        super().__init__()
        self.encoder = Encoder(num_layers, d_model, num_heads, dff,
input_vocab_size, pe_input, rate)
        self.decoder = Decoder(num_layers, d_model, num_heads, dff,
target_vocab_size, pe_target, rate)
        self.final_layer = tf.keras.layers.Dense(target_vocab_size)

    def call(self, inputs, training):
        # Keras models prefer if you pass all your inputs in the first
argument
        inp, tar = inputs

        enc_padding_mask, look_ahead_mask, dec_padding_mask =
self.create_masks(inp, tar)

        # (batch_size, inp_seq_len, d_model)
        enc_output = self.encoder(inp, training, enc_padding_mask)

        # dec_output.shape == (batch_size, tar_seq_len, d_model)
        dec_output, attention_weights = self.decoder(tar, enc_output,
training, look_ahead_mask, dec_padding_mask)

        # (batch_size, tar_seq_len, target_vocab_size)
        final_output = self.final_layer(dec_output)

        return final_output, attention_weights

    def create_masks(self, inp, tar):
        # Encoder padding mask
        enc_padding_mask = create_padding_mask(inp)

        # Used in the 2nd attention block in the decoder.
        # This padding mask is used to mask the encoder outputs.
        dec_padding_mask = create_padding_mask(inp)

        # Used in the 1st attention block in the decoder.
        # It is used to pad and mask future tokens in the input received by
        # the decoder.

```



```

        look_ahead_mask = create_look_ahead_mask(tf.shape(tar)[1])
        dec_target_padding_mask = create_padding_mask(tar)
        look_ahead_mask = tf.maximum(dec_target_padding_mask,
look_ahead_mask)
        return enc_padding_mask, look_ahead_mask, dec_padding_mask
# Hiperparâmetros
num_layers = 4
d_model = 128
dff = 512
num_heads = 8
dropout_rate = 0.1
class
CustomSchedule(tf.keras.optimizers.schedules.LearningRateSchedule):
    def __init__(self, d_model, warmup_steps=4000):
        super(CustomSchedule, self).__init__()
        self.d_model = d_model
        self.d_model = tf.cast(self.d_model, tf.float32)
        self.warmup_steps = warmup_steps

    def __call__(self, step):
        step = tf.cast(step, tf.float32) # Adicionado para evitar ERRO
        arg1 = tf.math.rsqrt(step)
        arg2 = step * (self.warmup_steps ** -1.5)
        return tf.math.rsqrt(self.d_model) * tf.math.minimum(arg1, arg2)

learning_rate = CustomSchedule(d_model)
optimizer = tf.keras.optimizers.Adam(learning_rate, beta_1=0.9,
beta_2=0.98, epsilon=1e-9)
loss_object =
tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True,
reduction='none')
def loss_function(real, pred):
    mask = tf.math.logical_not(tf.math.equal(real, 0))
    loss_ = loss_object(real, pred)
    mask = tf.cast(mask, dtype=loss_.dtype)
    loss_ *= mask
    return tf.reduce_sum(loss_) / tf.reduce_sum(mask)

def accuracy_function(real, pred):
    accuracies = tf.equal(real, tf.argmax(pred, axis=2))
    mask = tf.math.logical_not(tf.math.equal(real, 0))
    accuracies = tf.math.logical_and(mask, accuracies)
    accuracies = tf.cast(accuracies, dtype=tf.float32)
    mask = tf.cast(mask, dtype=tf.float32)
    return tf.reduce_sum(accuracies) / tf.reduce_sum(mask)

train_loss = tf.keras.metrics.Mean(name='train_loss')
train_accuracy = tf.keras.metrics.Mean(name='train_accuracy')
transformer = Transformer(
    num_layers=num_layers,
    d_model=d_model,
    num_heads=num_heads,
    dff=dff,
    input_vocab_size=tokenizers.pt.get_vocab_size().numpy(),
    target_vocab_size=tokenizers.en.get_vocab_size().numpy(),
    pe_input=1000,
    pe_target=1000,
    rate=dropout_rate)

```

```

# Checkpoint
checkpoint_path = "./checkpoints/train"
ckpt = tf.train.Checkpoint(transformer=transformer,
optimizer=optimizer)
ckpt_manager = tf.train.CheckpointManager(ckpt, checkpoint_path,
max_to_keep=5)
# if a checkpoint exists, restore the latest checkpoint.
if ckpt_manager.latest_checkpoint:
    ckpt.restore(ckpt_manager.latest_checkpoint)
    print('Latest checkpoint restored!!')
EPOCHS = 20
train_step_signature= [
    tf.TensorSpec(shape=(None, None), dtype=tf.int64),
    tf.TensorSpec(shape=(None, None), dtype=tf.int64),
]

@tf.function(input_signature=train_step_signature)
def train_step(inp, tar):
    tar_inp= tar[:, :-1]
    tar_real= tar[:, 1:]
    with tf.GradientTape() as tape:
        predictions, _ = transformer([inp, tar_inp], training = True)
        loss= loss_function(tar_real, predictions)

        gradients= tape.gradient(loss, transformer.trainable_variables)
        optimizer.apply_gradients(zip(gradients,
transformer.trainable_variables))
        train_loss(loss)
        train_accuracy(accuracy_function(tar_real, predictions))

for epoch in range(EPOCHS):
    start = time.time()
    train_loss.reset_state()
    train_accuracy.reset_state()
    # inp-> portuguese, tar-> english
    for(batch, (inp, tar)) in enumerate(train_batches):
        train_step(inp, tar)
        if batch % 50 == 0:
            print(f'Epoch{epoch+ 1} Batch {batch}
Loss{train_loss.result():.4f} Accuracy {train_accuracy.result():.4f}')

        if(epoch + 1) % 5 == 0:
            ckpt_save_path= ckpt_manager.save()
            print(f'Savingcheckpoint for epoch{epoch+1} at{ckpt_save_path}')

            print(f'Epoch{epoch+ 1} Loss{train_loss.result():.4f} Accuracy
{train_accuracy.result():.4f}')
            print(f'Timetakenfor 1 epoch: {time.time() -start:.2f} secs\n')

```

```

Epoch1 Batch 0 Loss8.8749 Accuracy 0.0000
Epoch1 Batch 50 Loss8.8154 Accuracy 0.0026
Epoch1 Batch 100 Loss8.7097 Accuracy 0.0175
Epoch1 Batch 150 Loss8.5960 Accuracy 0.0276
Epoch1 Batch 200 Loss8.4555 Accuracy 0.0329
Epoch1 Batch 250 Loss8.2860 Accuracy 0.0374
Epoch1 Batch 300 Loss8.0958 Accuracy 0.0408
Epoch1 Batch 350 Loss7.8987 Accuracy 0.0442
Epoch1 Batch 400 Loss7.7113 Accuracy 0.0529
Epoch1 Batch 450 Loss7.5427 Accuracy 0.0608

```

```
Epoch1 Batch 500 Loss7.3946 Accuracy 0.0688
Epoch1 Batch 550 Loss7.2615 Accuracy 0.0768
Epoch1 Batch 600 Loss7.1382 Accuracy 0.0846
Epoch1 Batch 650 Loss7.0203 Accuracy 0.0924
Epoch1 Batch 700 Loss6.9122 Accuracy 0.0997
Epoch1 Batch 750 Loss6.8119 Accuracy 0.1062
Epoch1 Batch 800 Loss6.7179 Accuracy 0.1124
Epoch1 Loss6.7020 Accuracy 0.1135
Timetakenfor 1 epoch: 166.49 secs
```

```
Epoch2 Batch 0 Loss5.0856 Accuracy 0.2240
Epoch2 Batch 50 Loss5.2126 Accuracy 0.2136
Epoch2 Batch 100 Loss5.1797 Accuracy 0.2184
Epoch2 Batch 150 Loss5.1590 Accuracy 0.2209
Epoch2 Batch 200 Loss5.1262 Accuracy 0.2248
```

```
...
Savingcheckpoint for epoch20 at./checkpoints/train/ckpt-4
Epoch20 Loss1.4604 Accuracy 0.6782
Timetakenfor 1 epoch: 95.31 secs
```

```
class Translator(tf.Module):
    def __init__(self, tokenizers, transformer):
        self.tokenizers = tokenizers
        self.transformer = transformer

    def __call__(self, sentence, max_length=20):
        # input sentence is portuguese, hence adding the start and end
        token
        assert isinstance(sentence, tf.Tensor)
        if len(sentence.shape) == 0:
            sentence = sentence[tf.newaxis]
        sentence = self.tokenizers.pt.tokenize(sentence).to_tensor()
        encoder_input = sentence
        # as the target is english, the first token to the transformer
        should be the
        # english start token.
        start_end = self.tokenizers.en.tokenize([''])[0]
        start = start_end[0][tf.newaxis]
        end = start_end[1][tf.newaxis]
        output_array = tf.TensorArray(dtype=tf.int64, size=0,
dynamic_size=True)
        output_array = output_array.write(0, start)
        for i in tf.range(max_length):
            output = tf.transpose(output_array.stack())
            predictions, _ = self.transformer([encoder_input, output],
training=False)
            predictions = predictions[:, -1:, :] # (batch_size, 1,
vocab_size)
            predicted_id = tf.argmax(predictions, axis=-1)
            output_array = output_array.write(i+1, predicted_id[0])
            if predicted_id == end:
                break

        output = tf.transpose(output_array.stack())
        # output.shape (1, tokens)
        text = tokenizers.en.detokenize(output) [0]
        tokens = tokenizers.en.lookup(output) [0]
        _, attention_weights = self.transformer([encoder_input,
output[:, :-1]],
```



```
training=False)
    return text, tokens, attention_weights
translator = Translator(tokenizers, transformer)
sentence = "Eu li sobre triceratops na enciclopédia."
translated_text, translated_tokens, attention_weights = translator(
    tf.constant(sentence))

print(f'"Prediction":15s}: {translated_text}')
```

```
Prediction      : b'i read about triumphat in ency .'
```

## APÊNDICE 9 – BIG DATA

### A – ENUNCIADO

Enviar um arquivo PDF contendo uma descrição breve (2 páginas) sobre a implementação de uma aplicação ou estudo de caso envolvendo Big Data e suas ferramentas (NoSQL e NewSQL). Caracterize os dados e Vs envolvidos, além da modelagem necessária dependendo dos modelos de dados empregados.

### B – RESOLUÇÃO

Estudo de caso: Uso de Big Data para análise de dados de telemetria em sistemas de armazenamento e distribuição de água.

Objetivo: Este trabalho tem o objetivo de descrever e avaliar a arquitetura de implementação de um ambiente BIG Data, voltado para a análise de dados de telemetria em um sistema de armazenamento e distribuição de água, demonstrando a correlação com os conceitos discutidos durante as aulas da disciplina.

Arquitetura de referência: Na arquitetura de referência a seguir, utilizada no respectivo estudo de caso, é possível destacar a adoção de ferramentas para cada etapa da construção do ambiente de Big Data voltado para análise de dados:

- a) Carga/ingestão de dados em batch e stream.
- b) Armazenamento de dados em um Data Lake.
- c) Transformação e carga de dados para bases analíticas- Data Warehouse.
- d) Machine learning.
- e) Visualização de dados.

A arquitetura escolhida é executada no ambiente Azure da Microsoft pelo fato de ser o provedor de nuvem contratado pela empresa escolhida para o estudo de caso.

Caracterização dos dados: Os dados têm origem em sensores instalados nos equipamentos das plantas industriais que armazenam e distribuem a água já tratada para centenas de municípios. Estes sensores geram dados em intervalos de 15 minutos com medidas de vazão, pressão, nível do reservatório, potência e estado das bombas, entre outros. Os dados são coletados em todas as plantas espalhadas pelo estado e centralizados em no ambiente de nuvem para processamento e análise. A arquitetura está aderente ao conceito de Big Data, uma vez que os seus componentes dão suporte aos “3 V’s do Big Data”:

1. Volume: Um sistema de armazenamento e distribuição de água possui milhares de sensores distribuídos em suas diversas plantas e equipamentos, gerando um imenso volume de dados.
2. Velocidade: Os dados são coletados e ingeridos pela arquitetura de Big Data praticamente em tempo real (streaming), possibilitando análises e tomadas de decisão rápidas e assertivas.
3. Variedade: A arquitetura de referência proposta permite a coleta e armazenamento de dados dos mais diversos tipos, sejam eles estruturados, semi-estruturados ou não estruturados.

Modelos de dados: Para que o processamento e análise dos dados coletados pelos sensores seja possível, podem ser utilizados alguns tipos de modelagem:

1. Time-series: permite fazer análise histórica e previsões, como por exemplo, previsão de demanda na produção e distribuição de água para os próximos meses ou anos, previsão do nível dos reservatórios para determinados períodos do dia, etc.
2. Column-store: permite fazer análises com maior desempenho e utilizando menos recursos.
3. Dados geográficos: permite fazer análises relacionadas à posição geográfica das plantas e equipamentos, possibilitando visualizar dados em mapas para melhor tomada de decisão

## APÊNDICE 10 – VISÃO COMPUTACIONAL

### A – ENUNCIADO

#### 1) Extração de Características

Os bancos de imagens fornecidos são conjuntos de imagens de 250x250 pixels de imuno-histoquímica (biópsia) de câncer de mama. No total são 4 classes (0, 1+, 2+ e 3+) que estão divididas em diretórios. O objetivo é classificar as imagens nas categorias correspondentes. Uma base de imagens será utilizada para o treinamento e outra para o teste do treino.

As imagens fornecidas são recortes de uma imagem maior do tipo WSI (*Whole Slide Imaging*) disponibilizada pela Universidade de Warwick ([link](#)). A nomenclatura das imagens segue o padrão XX\_HER\_YYYY.png, onde XX é o número do paciente e YYYY é o número da imagem recortada. Separe a base de treino em 80% para treino e 20% para validação. **Separe por pacientes (XX), não utilize a separação randômica! Pois, imagens do mesmo paciente não podem estar na base de treino e de validação, pois isso pode gerar um viés.** No caso da CNN VGG16 remova a última camada de classificação e armazene os valores da penúltima camada como um vetor de características. Após o treinamento, os modelos treinados devem ser validados na base de teste.

Tarefas:

- Carregue a base de dados de **Treino**.
- Crie partições contendo 80% para treino e 20% para validação (atenção aos pacientes).
- Extraia características utilizando LBP e a CNN VGG16 (gerando um csv para cada extrator).
- Treine modelos Random Forest, SVM e RNA para predição dos dados extraídos.
- Carregue a base de **Teste** e execute a tarefa 3 nesta base.
- Aplique os modelos treinados nos dados de treino
- Calcule as métricas de Sensibilidade, Especificidade e F1-Score com base em suas matrizes de confusão.
- Indique qual modelo dá o melhor o resultado e a métrica utilizada

#### 2) Redes Neurais

Utilize as duas bases do exercício anterior para treinar as Redes Neurais Convolucionais VGG16 e a Resnet50. Utilize os pesos pré-treinados (*Transfer Learning*), refaça as camadas *Fully Connected* para o problema de 4 classes. Compare os treinos de 15 épocas com e sem *Data Augmentation*. Tanto a VGG16 quanto a Resnet50 têm como camada de entrada uma imagem 224x224x3, ou seja, uma imagem de 224x224 pixels coloridos (3 canais de cores). Portanto, será necessário fazer uma transformação de 250x250x3 para 224x224x3. Ao fazer o *Data Augmentation* **cuidado** para não alterar demais as cores das imagens e atrapalhar na classificação.

Tarefas:

- a) Utilize a base de dados de **Treino** já separadas em treino e validação do exercício anterior
- b) Treine modelos VGG16 e Resnet50 adaptadas com e sem *Data Augmentation*
- c) Aplique os modelos treinados nas imagens da base de **Teste**
- d) Calcule as métricas de Sensibilidade, Especificidade e F1-Score com base em suas matrizes de confusão.
- e) Indique qual modelo dá o melhor o resultado e a métrica utilizada

## B – RESOLUÇÃO

### 1. Extração de características

```
import numpy as np, cv2, sys, pylab, matplotlib, PIL, scipy, os
from numpy import histogram
from scipy import stats
from PIL import Image
import matplotlib.pyplot as plt
import skimage
from skimage.feature import local_binary_pattern
from skimage import data, morphology

from skimage.transform import rotate
from skimage import data
from skimage.color import label2rgb
from skimage import data, exposure, util

from skimage.util import dtype

from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.datasets import load_svmlight_file
from sklearn.metrics import f1_score
from sklearn import preprocessing

!pip install keras
!pip install tensorflow
import tensorflow as tf
import keras, os
from keras.applications.vgg16 import VGG16
from keras.models import Sequential
from keras.applications.vgg16 import preprocess_input
from keras.layers import Dense, Conv2D, MaxPool2D, Flatten
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import Model
from tensorflow.keras.preprocessing import image

from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier

import numpy as np
```

```

from sklearn.metrics import confusion_matrix
import seaborn as sns
import pandas as pd

# Módulo para imprimir os gráficos de treinamento de forma dinâmica.
!pip install livelossplot
#Monta o Drive Google para ler os arquivos
from google.colab import drive
drive.mount('/content/drive')

!mkdir Features
!mkdir probabilidades
#Cria as funcoes para extracao de caracteristicas LBP
def lbp_riu(npmat_img, label, arq):

    radius = 1
    n_points = 8
    lbp = local_binary_pattern(npmat_img, n_points, radius,
method='uniform')

    array_lbp = np.zeros(11, dtype=int) #Dez posições do P+2 e mais uma
para o label.
    rows = npmat_img.shape[0]
    cols = npmat_img.shape[1]
    for ci in range (0, rows):
        for cj in range (0, cols):
            aux = int(lbp[ci][cj])
            array_lbp[aux+1] += 1

    array_lbp[0] = label #Label da imagem
    array_str = '%s' % (label) #[0]

    for i in range (1, len(array_lbp)):
        str_aux = " %d:%d" % (i, array_lbp[i])
        array_str = array_str + str_aux #[i]
    array_str = array_str + '\n'

    nome_arq = 'Features/lbp_riu_8_1_'+arq

    arquivo = open(nome_arq, 'a')
    arquivo.write(array_str)
    arquivo.close()
def lbp_tophat(npmat_img, label, arq):

    radius = 1
    n_points = 8
    lbp = local_binary_pattern(npmat_img, n_points, radius,
method='uniform')
    array_lbp = np.zeros(11, dtype=int) #Dez posições do P+2 e mais uma
para o label.

    rows = npmat_img.shape[0]
    cols = npmat_img.shape[1]
    for ci in range (0, rows):
        for cj in range (0, cols):
            aux = int(lbp[ci][cj])
            array_lbp[aux+1] += 1

```



```

array_lbp[0] = label #Label da imagem
array_str = '%s' % (label) #[0]

for i in range (1, len(array_lbp)):
    str_aux = " %d:%d" % (i, array_lbp[i])
    array_str = array_str + str_aux #[i]

se = morphology.disk(5)
white = morphology.white_tophat(npmat_img, se)
black = morphology.black_tophat(npmat_img, se)
wmean = np.mean(white)
wsd = np.std(white)
bmean = np.mean(black)
bsd = np.std(black)

    str_aux = " 11:%.10f 12:%.10f 13:%.10f 14:%.10f" % (wmean, wsd,
bmean, bsd)
    array_str = array_str + str_aux
    array_str = array_str + '\n'

nome_arq = 'Features/lbp_tophat'+arq

arquivo = open(nome_arq, 'a')
arquivo.write(array_str)
arquivo.close()
def tophat(npmat_img, label, arq):

    se = morphology.disk(5)
    white = morphology.white_tophat(npmat_img, se)
    black = morphology.black_tophat(npmat_img, se)
    wmean = np.mean(white)
    wsd = np.std(white)
    bmean = np.mean(black)
    bsd = np.std(black)

    array_str = '%s' % (label)
    str_aux = " 1:%.10f 2:%.10f 3:%.10f 4:%.10f" % (wmean, wsd, bmean,
bsd)
    array_str = array_str + str_aux
    array_str = array_str + '\n'

    nome_arq = 'Features/tophat_'+arq

    arquivo = open(nome_arq, 'a')
    arquivo.write(array_str)
    arquivo.close()
#Extração de características usando CNN VGG16
trdata = ImageDataGenerator()
traindata =
trdata.flow_from_directory(directory="/content/drive/MyDrive/IAA011/Tra
in_Warwick/Train_4cls_amostra", target_size=(224,224))

tsdata = ImageDataGenerator()
testdata =
tsdata.flow_from_directory(directory="/content/drive/MyDrive/IAA011/Tes
t_Warwick/Test_4cl_amostra", target_size=(224,224))

```

```

vldata = ImageDataGenerator()
validation_data =
vldata.flow_from_directory(directory="/content/drive/MyDrive/IAA011/Tra
in_Warwick/Test_4cls_amostra", target_size=(224,224))
model = VGG16()
model = Model(inputs=model.inputs, outputs=model.layers[-2].output)
model.summary()
def load_image(path):
    img = image.load_img(path, target_size=model.input_shape[1:3])
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
    x = preprocess_input(x)
    return img, x
def vgg_16(npmat_img, label, arq):

    img = image.load_img(npmat_img, target_size=model.input_shape[1:3])
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
    x = preprocess_input(x)

    feat_extractor = Model(inputs=model.input,
outputs=model.get_layer("fc2").output)
    feat = feat_extractor.predict(x)

    array_str = '%s' % (label) #[0]

    for ci in range (0, 4095):
        aux = float(feat[0][ci])
        str_aux = " %d:%.10f" % (ci+1, aux)
        array_str = array_str + str_aux

    nome_arq = 'Features/vgg16'+arq

    arquivo = open(nome_arq, 'a')
    arquivo.write(array_str)
    arquivo.write('\n')
    arquivo.close()
def Calcula_caracteristicas(arq):

    img_path = open(arq, 'r')

    for line in img_path:

        image_block = line.rstrip('\n')
        nums = image_block.split('.')
        nums = nums[0].split('/')
        label = nums[7]

        #Le imagem original
        img = cv2.imread(image_block,0)

        #print(image_block)

        #Chama as funcoes de caracteristicas
        lbp_tophat(img, label, arq)
        lbp_riu(img, label, arq)
        tophat(img, label, arq)
        vgg_16(image_block, label, arq)

```

```

Calcula_caracteristicas("Train.txt")
Calcula_caracteristicas("Test.txt")
Calcula_caracteristicas("Validation.txt")
def imprime_acuracia(y_test, y_pred, modelo):

    matrix = confusion_matrix(y_test, y_pred)

    labels = ['0', '1', '2', '3']

    plt.figure(figsize=(10,7))
    sns.heatmap(matrix, annot=True, fmt="d", cmap='Blues',
xticklabels=labels, yticklabels=labels)

    plt.xlabel('Predito')
    plt.ylabel('Atual')
    plt.title('Matriz de confusão ' + modelo)
    plt.show()

    total1=sum(sum(matrix));

    accuracy1=(matrix[0,0]+matrix[1,1]+matrix[2,2]+matrix[3,3])/total1
    print ('Acuracia {:.2f}%'.format(accuracy1 * 100))

    sensitivity1 = matrix[0,0]/(matrix[0,0]+matrix[0,1])
    print ('Sensibilidade {:.2f}%'.format(sensitivity1 * 100))

    specificity1 = matrix[1,1]/(matrix[1,0]+matrix[1,1])
    print ('Especificidade {:.2f}%'.format(specificity1 * 100))

    f1 = f1_score(y_test, y_pred, average='weighted')
    print ('F1 Score {:.2f}%'.format(f1 * 100))
# LBP RIU
X, y = load_svmlight_file("Features/lbp_riu_8_1_Train.txt")
X_valid, y_valid = load_svmlight_file("Features/lbp_riu_8_1_Validation.txt")

#Random forest
rflbpriu = RandomForestClassifier()
rflbpriu.fit(X, y)
print ('Acuracia LBP RIU com RF {:.2f}%'.format(rflbpriu.score(X_valid,
y_valid) * 100))

#SVM
svmlbpriu = svm.SVC(kernel='linear')
svmlbpriu.fit(X, y)
svmlbpriu.fit(X, y)
print ('Acuracia LBP RIU com SVM {:.2f}%'.format(svmlbpriu.score(X_valid, y_valid) * 100))

#RNA
rnalbpriu = MLPClassifier()
rnalbpriu.fit(X, y)
print ('Acuracia LBP RIU com RNA {:.2f}%'.format(rnalbpriu.score(X_valid, y_valid) * 100))

```

Acurácia LBP RIU com RF 67.80%

Acurácia LBP RIU com SVM 52.54%

Acurácia LBP RIU com RNA 50.85%

# LBP Tophat

```

X, y = load_svmlight_file("Features/lbp_tophatTrain.txt")
X_valid, y_valid = load_svmlight_file("Features/lbp_tophatValidation.txt")

#Random forest
rflbptop = RandomForestClassifier()
rflbptop.fit(X, y)
print ('Acuracia LBP Tophat com RF {:.2f}%'.format(rflbptop.score(X_valid, y_valid) * 100))

#SVM
svmlbptop = svm.SVC(kernel='linear')
svmlbptop.fit(X, y)
print ('Acuracia LBP Tophat com SVM {:.2f}%'.format(svmlbptop.score(X_valid, y_valid) * 100))

#RNA
rnalbptop = MLPClassifier()
rnalbptop.fit(X, y)
print ('Acuracia LBP Tophat com RNA {:.2f}%'.format(rnalbptop.score(X_valid, y_valid) * 100))

```

Acurácia LBP Tophat com RF 67.80%  
 Acurácia LBP Tophat com SVM 60.17%  
 Acurácia LBP Tophat com RNA 50.00%

```

# Tophat
X, y = load_svmlight_file("Features/tophat_Train.txt")
X_valid, y_valid = load_svmlight_file("Features/tophat_Validation.txt")

#Random forest
rftophat = RandomForestClassifier()
rftophat.fit(X, y)
print ('Acuracia Tophat com RF {:.2f}%'.format(rftophat.score(X_valid, y_valid) * 100))

#SVM
svmtophat = svm.SVC(kernel='linear')
svmtophat.fit(X, y)
print ('Acuracia Tophat com SVM {:.2f}%'.format(svmtophat.score(X_valid, y_valid) * 100))

#RNA
rnatophat = MLPClassifier()
rnatophat.fit(X, y)
print ('Acuracia Tophat com RNA {:.2f}%'.format(rnatophat.score(X_valid, y_valid) * 100))

```

Acurácia Tophat com RF 61.86%  
 Acurácia Tophat com SVM 59.32%  
 Acurácia Tophat com RNA 59.32%

```

# VGG16
X, y = load_svmlight_file("Features/vgg16Train.txt")
X_valid, y_valid = load_svmlight_file("Features/vgg16Validation.txt")

#Random forest
rfvgg = RandomForestClassifier()
rfvgg.fit(X, y)
print ('Acuracia VGG16 com RF {:.2f}%'.format(rfvgg.score(X_valid, y_valid) * 100))

```

```

#SVM
svmvvgg = svm.SVC(kernel='linear')
svmvvgg.fit(X, y)
print ('Acuracia VGG16 com SVM {:.2f}%'.format(svmvgg.score(X_valid,
y_valid) * 100))

#RNA
rnavgg = MLPClassifier()
rnavgg.fit(X, y)
print ('Acuracia VGG16 com RNA {:.2f}%'.format(rnavgg.score(X_valid,
y_valid) * 100))
Acurácia VGG16 com RF 93.22%
Acurácia VGG16 com SVM 94.07%
Acurácia VGG16 com RNA 92.37%

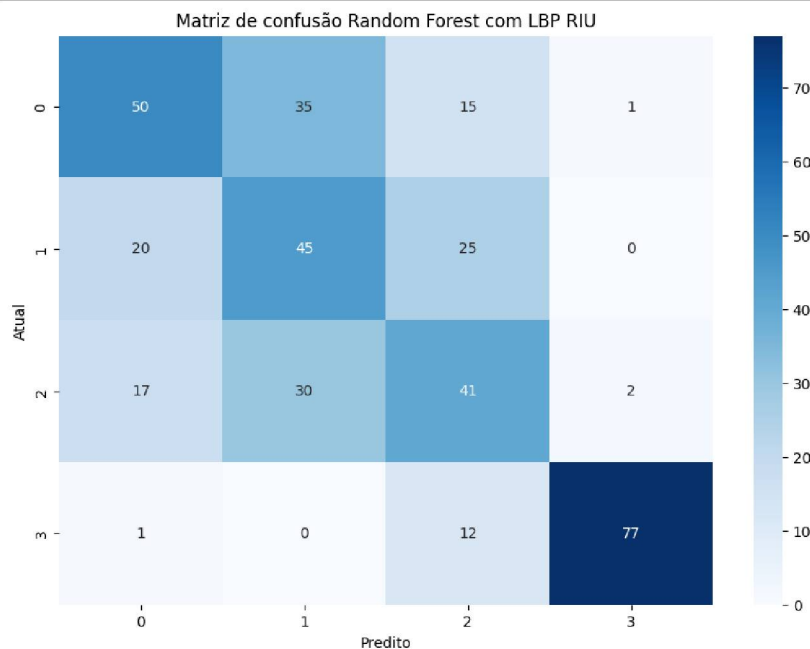
#Rodar os modelos na base de teste
#LBP RIU
X_test, y_test = load_svmlight_file("Features/lbp_riu_8_1_Test.txt")

#Random forest
y_pred = rflbpriu.predict(X_test)
imprime_acuracia(y_test, y_pred, "Random Forest com LBP RIU")

#SVM
y_pred = svmlbpriu.predict(X_test)
imprime_acuracia(y_test, y_pred, "SVM com LBP RIU")

#RNA
y_pred = rnalbprriu.predict(X_test)
imprime_acuracia(y_test, y_pred, "RNA com LBP RIU")

```



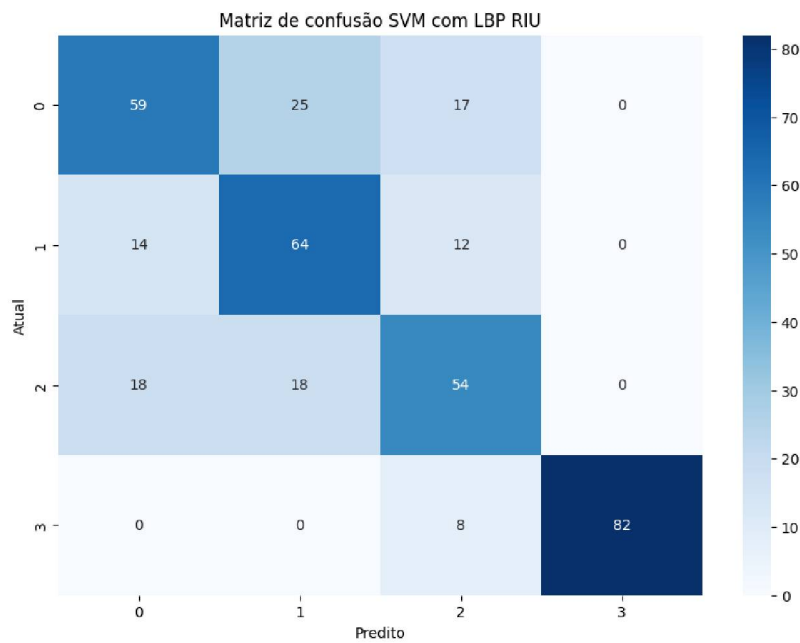
Acurácia 57.41%

Sensibilidade 58.82%

Especificidade 69.23%

F1 Score 58.17%



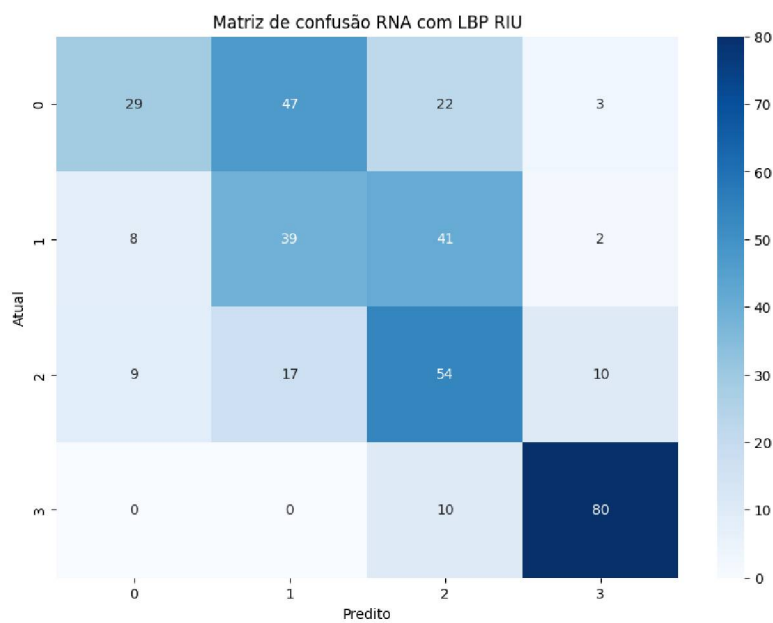


Acurácia 69.81%

Sensibilidade 70.24%

Especificidade 82.05%

F1 Score 70.10%



Acurácia 54.45%

Sensibilidade 38.16%

Especificidade 82.98%

F1 Score 53.60%

```
# LBP Tophat
X_test, y_test = load_svmlight_file("Features/lbp_tophatTest.txt")
```



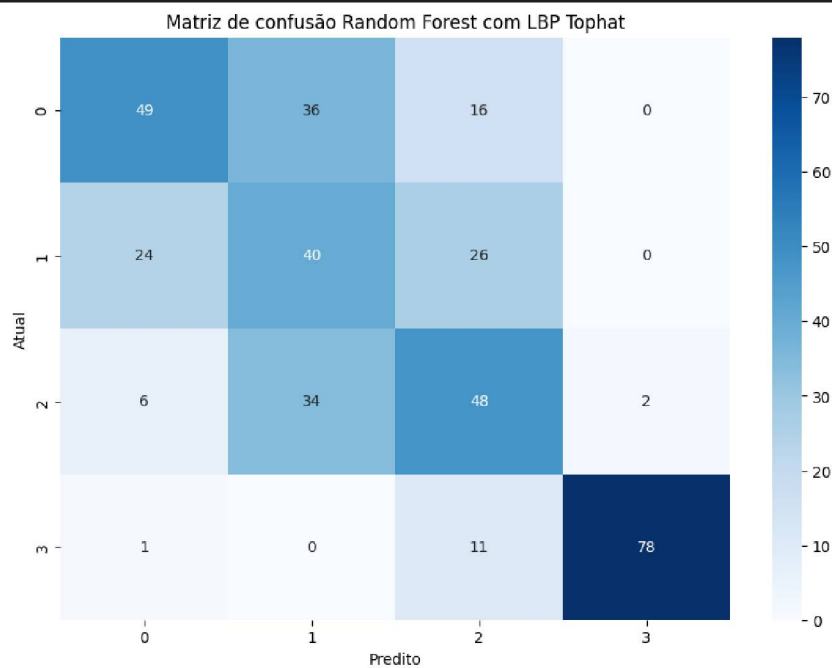
```

#Random forest
y_pred = rflbptop.predict(X_test)
imprime_acuracia(y_test, y_pred, 'Random Forest com LBP Tophat')

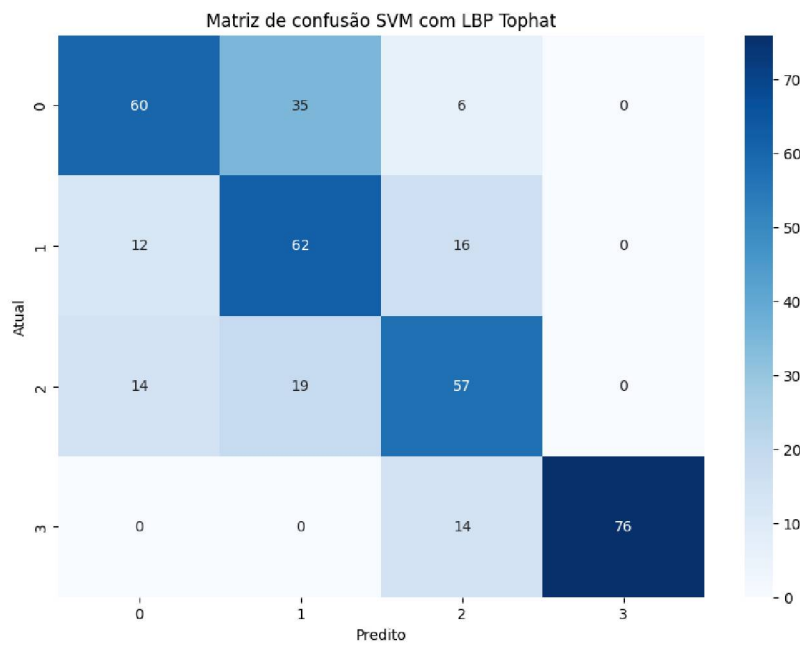
#SVM
y_pred = svmlbptop.predict(X_test)
imprime_acuracia(y_test, y_pred, 'SVM com LBP Tophat')

#RNA
y_pred = rnalbptop.predict(X_test)
imprime_acuracia(y_test, y_pred, 'RNA com LBP Tophat')

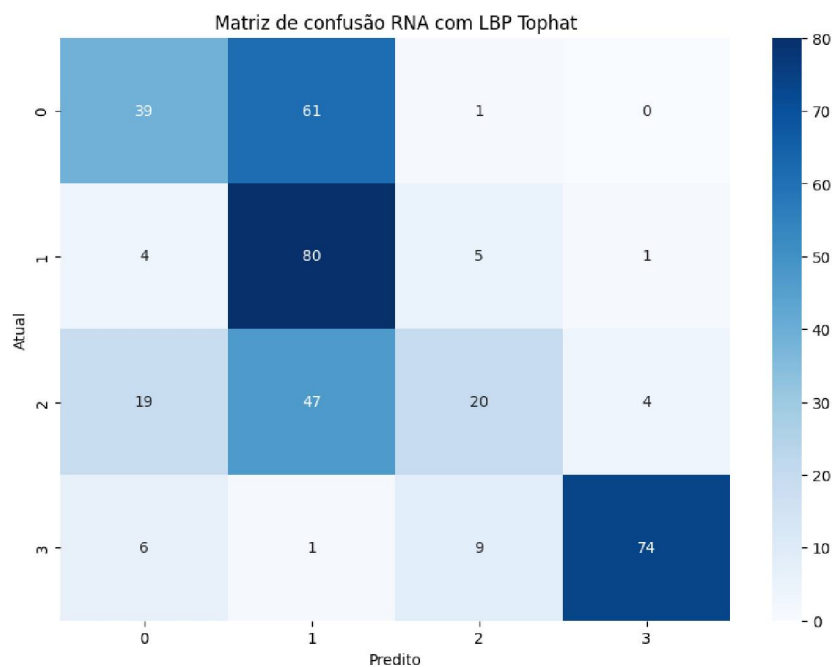
```



Acurácia 57.95%  
 Sensibilidade 57.65%  
 Especificidade 62.50%  
 F1 Score 58.90%



Acurácia 68.73%  
 Sensibilidade 63.16%  
 Especificidade 83.78%  
 F1 Score 69.40%



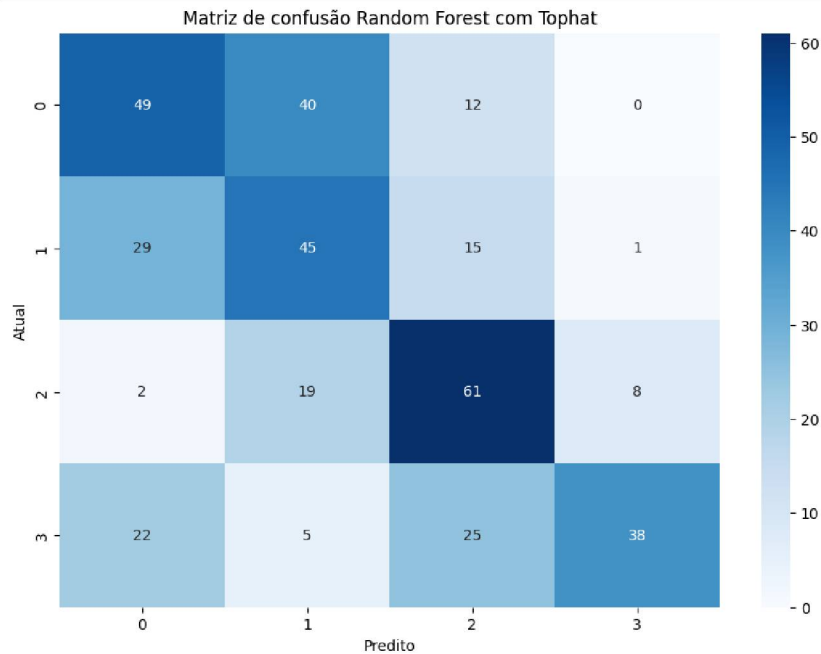
Acurácia 57.41%  
 Sensibilidade 39.00%  
 Especificidade 95.24%  
 F1 Score 55.48%

```
#Tophat
X_test, y_test = load_svmlight_file("Features/tophat_Test.txt")

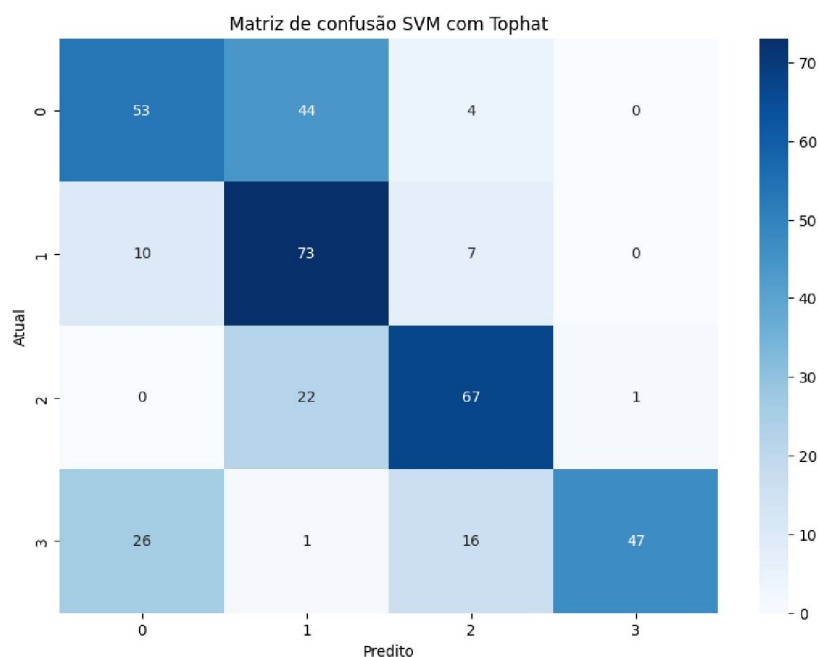
#Random forest
y_pred = rftophat.predict(X_test)
imprime_acuracia(y_test, y_pred, 'Random Forest com Tophat')
```

```
#SVM
y_pred = svmtophat.predict(X_test)
imprime_acuracia(y_test, y_pred, 'SVM com Tophat')

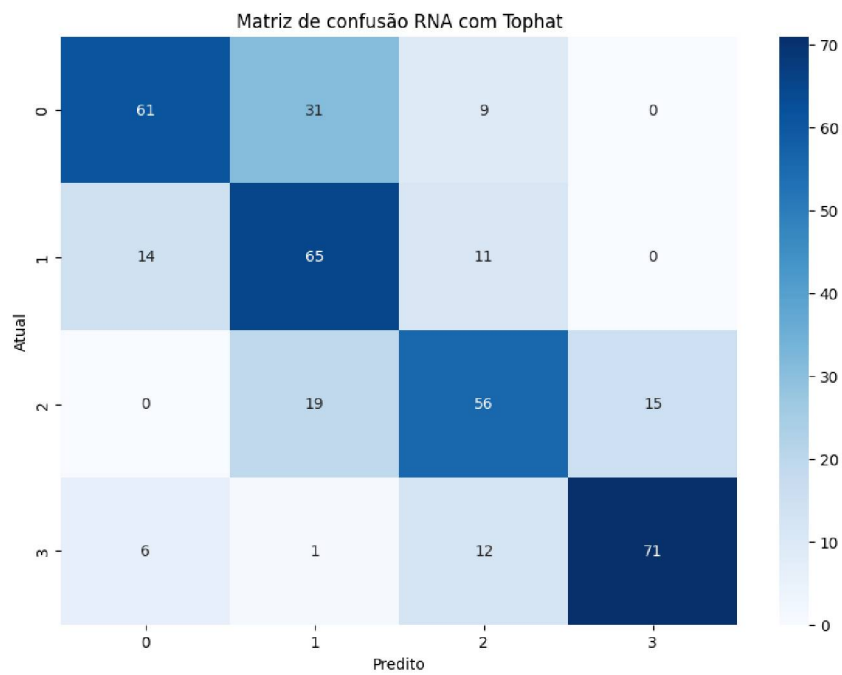
#RNA
y_pred = rnatophat.predict(X_test)
imprime_acuracia(y_test, y_pred, 'RNA com Tophat')
```



Acurácia 52.02%  
 Sensibilidade 55.06%  
 Especificidade 60.81%  
 F1 Score 52.15%



Acurácia 64.69%  
 Sensibilidade 54.64%  
 Especificidade 87.95%  
 F1 Score 64.78%



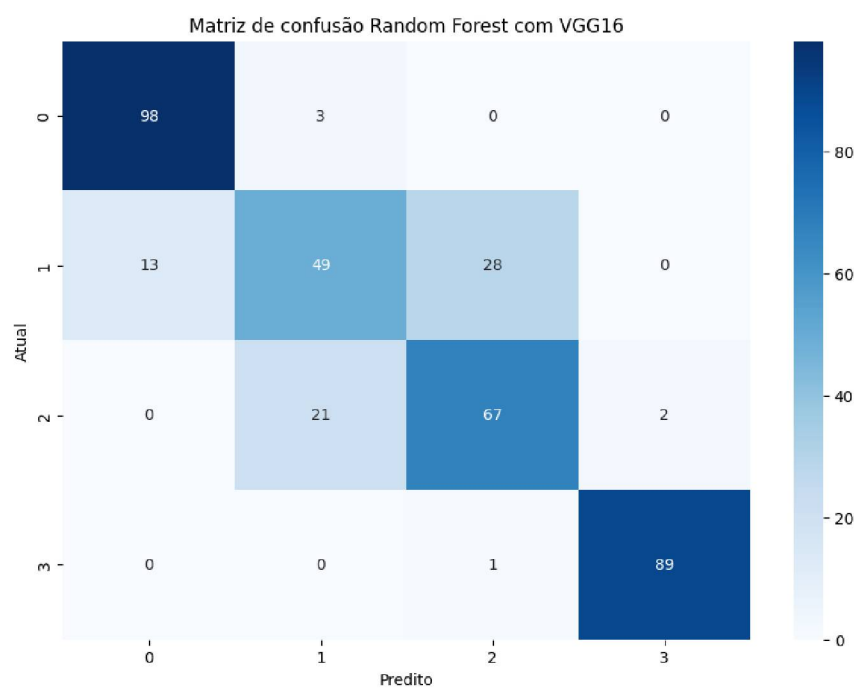
Acuracia 68.19%  
 Sensibilidade 66.30%  
 Especificidade 82.28%  
 F1 Score 68.39%

```
#VGG16
X_test, y_test = load_svmlight_file("Features/vgg16Test.txt")

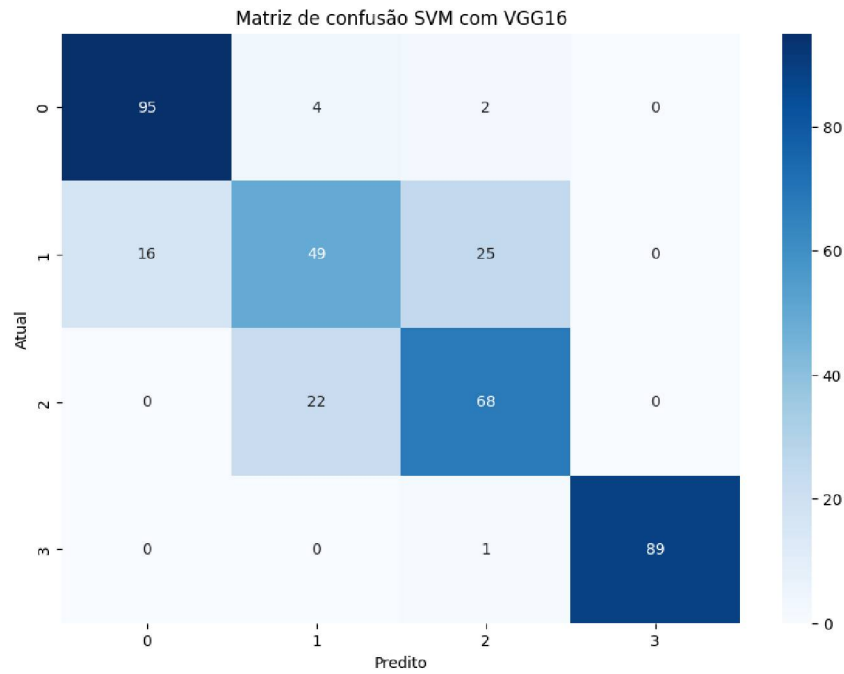
#Random forest
y_pred = rfvgg.predict(X_test)
imprime_acuracia(y_test, y_pred, 'Random Forest com VGG16')

#SVM
y_pred = svmvgg.predict(X_test)
imprime_acuracia(y_test, y_pred, 'SVM com VGG16')

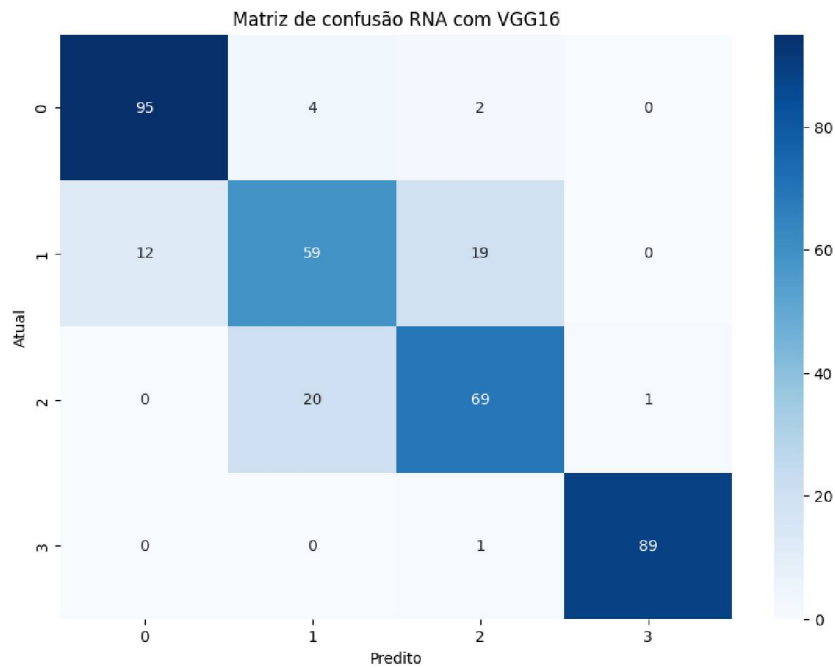
#RNA
y_pred = rnavgg.predict(X_test)
imprime_acuracia(y_test, y_pred, 'RNA com VGG16')
```



Acurácia 81.67%  
Sensibilidade 97.03%  
Especificidade 79.03%  
F1 Score 81.09%



Acurácia 81.13%  
Sensibilidade 95.96%  
Especificidade 75.38%  
F1 Score 80.67%



Acurácia 84.10%  
 Sensibilidade 95.96%  
 Especificidade 83.10%  
 F1 Score 83.90%

## 2. Redes Neurais

```
!pip install livelossplot
%pip install keras
import os
from keras.models import Model
from keras.optimizers import Adam
from tensorflow.keras.applications.resnet50 import preprocess_input
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.layers import Dense, Dropout, Flatten
from pathlib import Path
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import f1_score

from tensorflow.keras.applications.resnet50 import ResNet50
from keras.applications.vgg16 import VGG16

from keras.optimizers import RMSprop
from keras.callbacks import ModelCheckpoint, EarlyStopping, TensorBoard
from livelossplot import PlotLossesKeras

from sklearn.metrics import confusion_matrix
import seaborn as sns
#Monta o Drive Google para ler os arquivos
from google.colab import drive
drive.mount('/content/drive')
#Busca as classes dinamicamente (util para casos com muitas classes
#onde é necessário limitar)
```





```

shuffle=True,
seed=42)

testgen =
test_generator.flow_from_directory('/content/drive/MyDrive/IAA011/Test_
Warwick/Test_4cl_amostra',
                                   target_size=(224, 224),
                                   batch_size=BATCH_SIZE,
                                   class_mode=None,
                                   classes=class_subset,
                                   shuffle=False,
                                   seed=42)

# Resnet50 COM transfer learning
resnet_tl = ResNet50(input_shape=(224,224,3), weights='imagenet',
include_top=False)

#Treinar apenas as novas camadas
for layer in resnet_tl.layers:
    layer.trainable = False
# Refaça as camadas Fully Connected para o problema de 4 classes.
x_tl = Flatten()(resnet_tl.output)

# Camada de classificação com as 4 classes
prediction = Dense(len(class_subset), activation='softmax')(x_tl)

# Criação do Objeto Modelo
model_tlres = Model(inputs=resnet_tl.input, outputs=prediction)
%%time
steps_per_epoch = traingen.samples // BATCH_SIZE
val_steps = validgen.samples // BATCH_SIZE

n_epochs = 10
optimizer = RMSprop(learning_rate=0.0001)

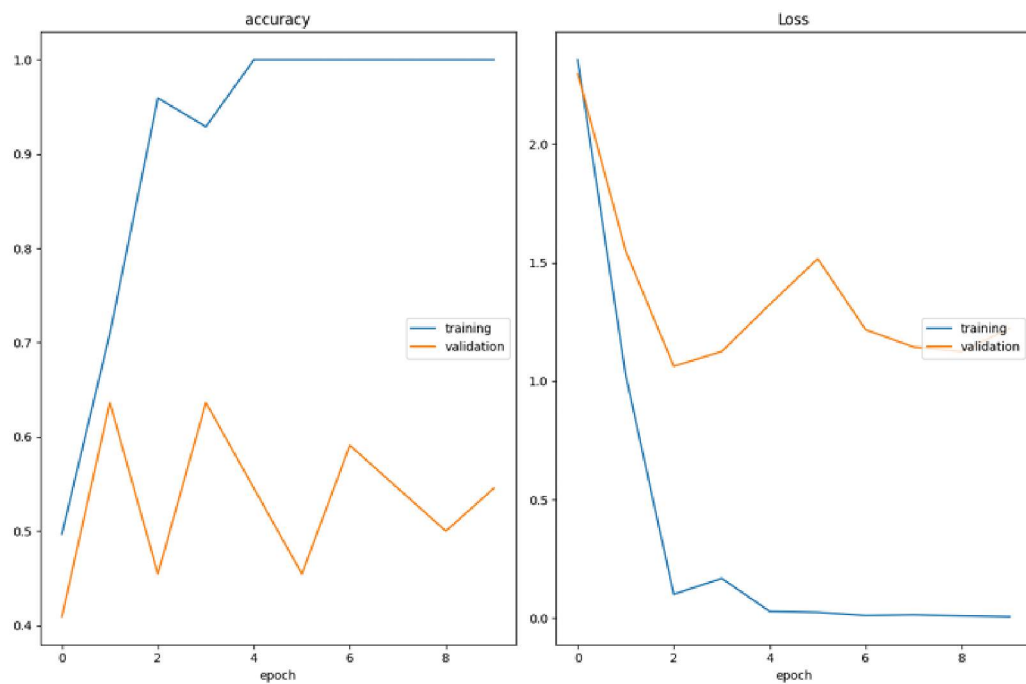
model_tlres.compile(loss='categorical_crossentropy',
optimizer=optimizer, metrics=['accuracy'])

# Salva o modelo Keras após cada época, porém só o de melhor resultado
checkpointer =
ModelCheckpoint(filepath='img_model_tlres.weights.best.keras',
                verbose=1,
                save_best_only=True)

# Para evita overfitting no treinamento
early_stop = EarlyStopping(monitor='val_loss',
                           patience=10,
                           restore_best_weights=True,
                           mode='min')

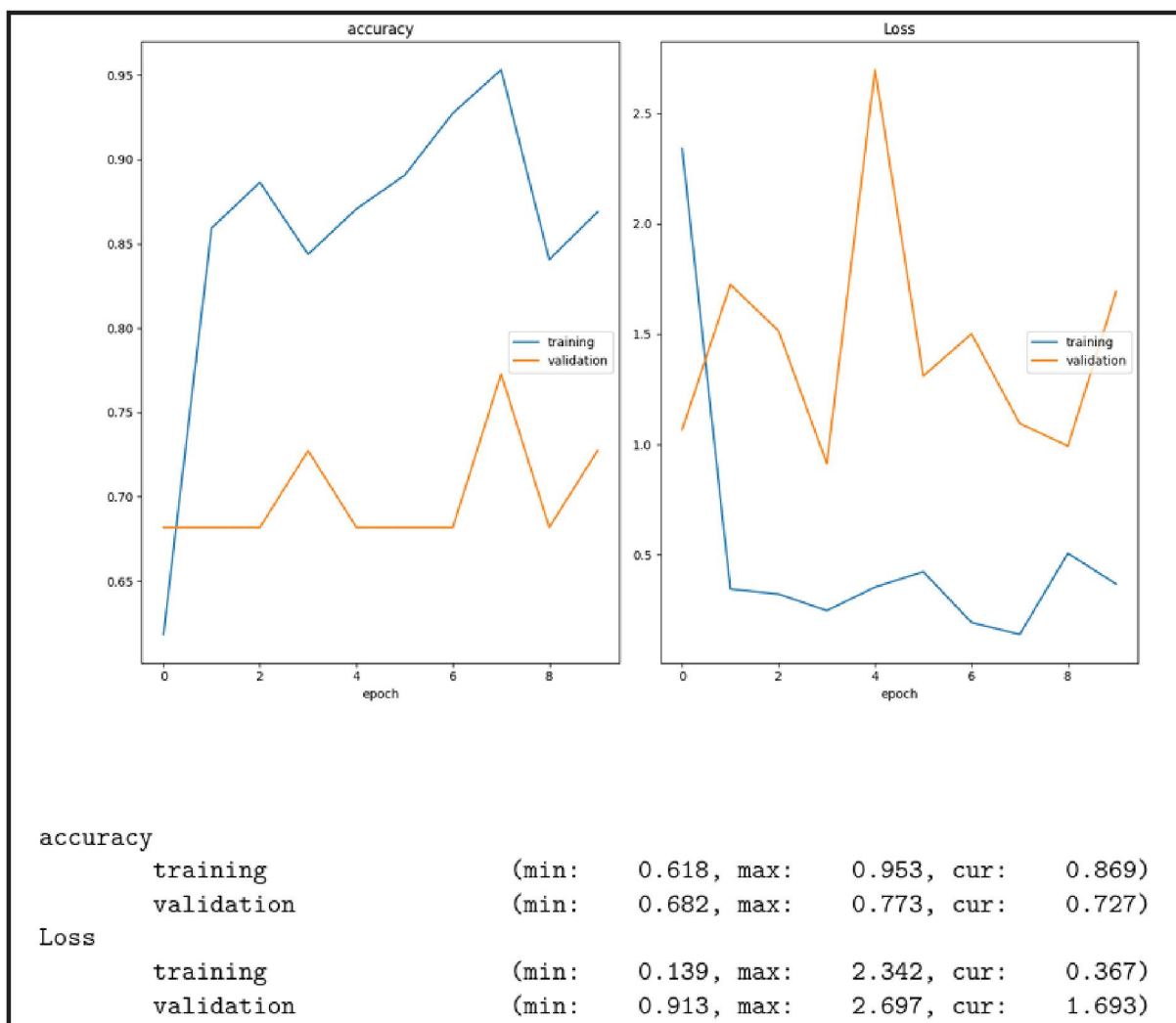
# Treinamento do Modelo resnet SEM Data Augmentation COM Transfer
learning
history_tlresorig = model_tlres.fit(traingenorig,
                                   epochs=n_epochs,
                                   steps_per_epoch=steps_per_epoch,
                                   validation_data=validgen,
                                   validation_steps=val_steps,
                                   callbacks=[early_stop, checkpointer,
PlotLossesKeras()],
                                   verbose=False)

```



```
accuracy
  training      (min: 0.497, max: 1.000, cur: 1.000)
 validation     (min: 0.409, max: 0.636, cur: 0.545)
Loss
  training      (min: 0.006, max: 2.355, cur: 0.006)
 validation     (min: 1.063, max: 2.297, cur: 1.223)
```

```
# Treinamento do Modelo resnet com Data Augmentation COM Transfer
learning
history_tlres = model_tlres.fit(training,
                                epochs=n_epochs,
                                steps_per_epoch=steps_per_epoch,
                                validation_data=validgen,
                                validation_steps=val_steps,
                                callbacks=[early_stop, checkpointer,
PlotLossesKeras()],
                                verbose=False)
```



```
# VGG16 Com transfer learning
vgg16_tl = VGG16(input_shape=(224,224,3), weights='imagenet',
include_top=False)

#Treinar apenas as novas camadas
for layer in resnet_tl.layers:
    layer.trainable = False
# Refaça as camadas Fully Connected para o problema de 4 classes.
x_tl = Flatten()(vgg16_tl.output)

# Camada de classificação com as 4 classes
prediction = Dense(len(class_subset), activation='softmax')(x_tl)

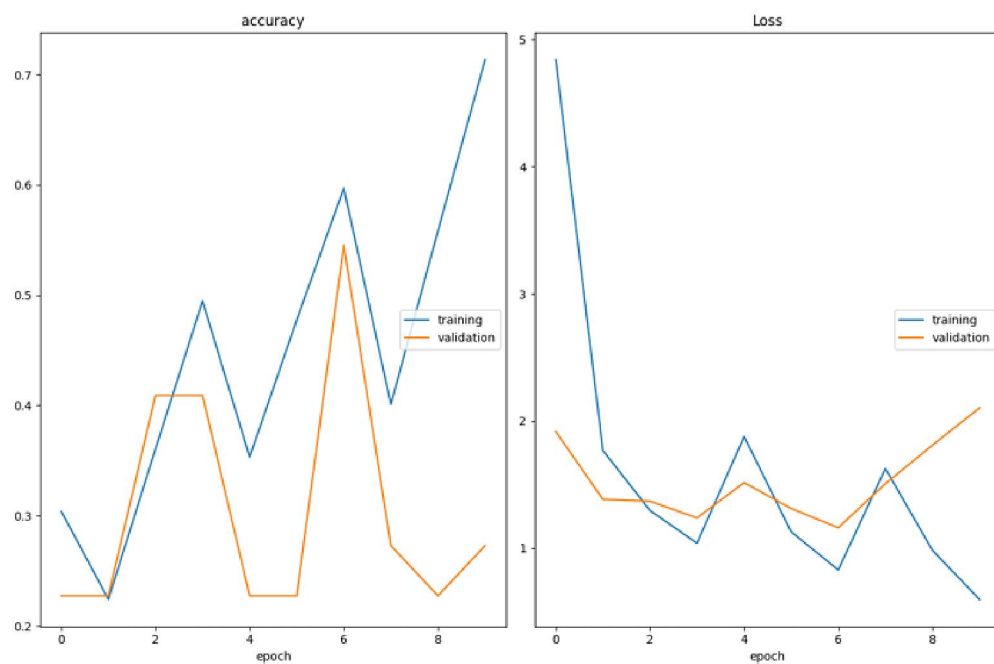
# Criação do Objeto Modelo
model_tlvvg = Model(inputs=vgg16_tl.input, outputs=prediction)
optimizer = RMSprop(learning_rate=0.0001)

model_tlvvg.compile(loss='categorical_crossentropy',
optimizer=optimizer, metrics=['accuracy'])

checkpointer =
ModelCheckpoint(filepath='img_model_tlvvg.weights.best.keras',
verbose=1,
save_best_only=True)
```

```
# Para evita overfitting no treinamento
early_stop = EarlyStopping(monitor='val_loss',
                           patience=10,
                           restore_best_weights=True,
                           mode='min')

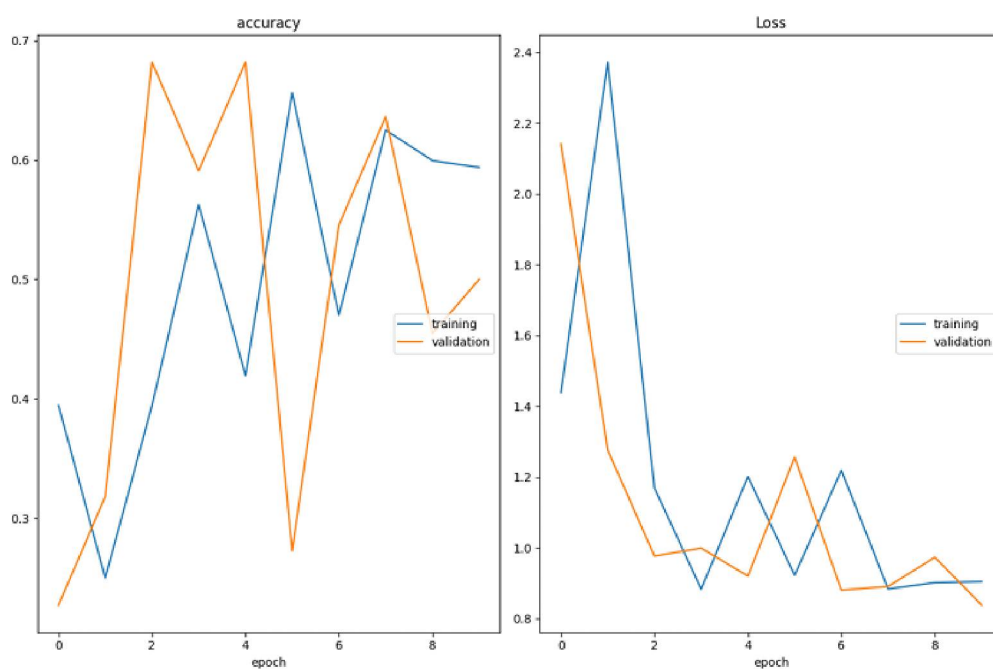
# Treinamento do Modelo VGG16 SEM Data Augmentation COM Transfer
learning
history_vggorig = model_tlvvgg.fit(traingenorig,
                                   epochs=n_epochs,
                                   steps_per_epoch=steps_per_epoch,
                                   validation_data=validgen,
                                   validation_steps=val_steps,
                                   callbacks=[early_stop, checkpointer,
PlotLossesKeras()],
                                   verbose=False)
```



|            |       |        |      |        |      |        |
|------------|-------|--------|------|--------|------|--------|
| accuracy   |       |        |      |        |      |        |
| training   | (min: | 0.224, | max: | 0.714, | cur: | 0.714) |
| validation | (min: | 0.227, | max: | 0.545, | cur: | 0.273) |
| Loss       |       |        |      |        |      |        |
| training   | (min: | 0.594, | max: | 4.842, | cur: | 0.594) |
| validation | (min: | 1.159, | max: | 2.102, | cur: | 2.102) |

```
# Treinamento do Modelo VGG16 COM Data Augmentation COM Transfer
learning
history_tlvvgg = model_tlvvgg.fit(traingen,
                                   epochs=n_epochs,
                                   steps_per_epoch=steps_per_epoch,
                                   validation_data=validgen,
                                   validation_steps=val_steps,
                                   callbacks=[early_stop, checkpointer,
PlotLossesKeras()],
                                   verbose=False)
```





|            |       |        |      |                    |
|------------|-------|--------|------|--------------------|
| accuracy   |       |        |      |                    |
| training   | (min: | 0.250, | max: | 0.656, cur: 0.594) |
| validation | (min: | 0.227, | max: | 0.682, cur: 0.500) |
| Loss       |       |        |      |                    |
| training   | (min: | 0.883, | max: | 2.372, cur: 0.904) |
| validation | (min: | 0.838, | max: | 2.141, cur: 0.838) |

```
def imprime_acuracia(y_test, y_pred, modelo):

    matrix = confusion_matrix(y_test, y_pred)

    labels = ['0', '1', '2', '3']

    plt.figure(figsize=(10,7))
    sns.heatmap(matrix, annot=True, fmt="d", cmap='Blues',
xticklabels=labels, yticklabels=labels)

    plt.xlabel('Predito')
    plt.ylabel('Atual')
    plt.title('Matriz de confusão ' + modelo)
    plt.show()

    totall=sum(sum(matrix));

    accuracy1=(matrix[0,0]+matrix[1,1]+matrix[2,2]+matrix[3,3])/totall
    print ('Acuracia {:.2f}%'.format(accuracy1 * 100))

    sensitivity1 = matrix[0,0]/(matrix[0,0]+matrix[0,1])
    print ('Sensibilidade {:.2f}%'.format(sensitivity1 * 100))

    specificity1 = matrix[1,1]/(matrix[1,0]+matrix[1,1])
    print ('Especificidade {:.2f}%'.format(specificity1 * 100))
```



```

f1 = f1_score(y_test, y_pred, average='weighted')
print ('F1 Score {:.2f}%'.format(f1 * 100))
from sklearn.metrics import accuracy_score

# Aplique os modelos treinados nas imagens da base de Teste
model_tlres.load_weights('img_model_tlres.weights.best.keras')
model_tlvvg.load_weights('img_model_tlvvg.weights.best.keras')

true_classes = testgen.classes
class_indices = traingen.class_indices
class_indices = dict((v,k) for k,v in class_indices.items())

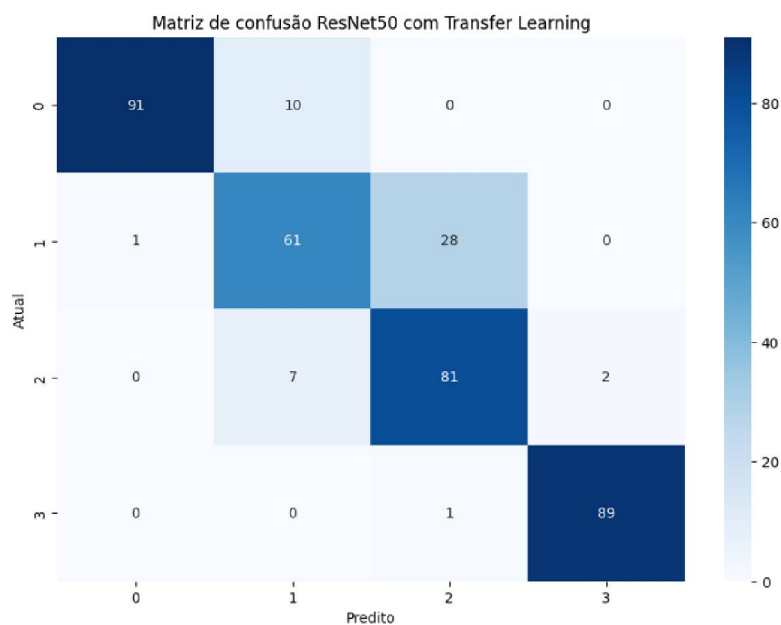
preds_tlres = model_tlres.predict(testgen)
pred_classes_tlres = np.argmax(preds_tlres, axis=1)

preds_tlvvg = model_tlvvg.predict(testgen)
pred_classes_tlvvg = np.argmax(preds_tlvvg, axis=1)

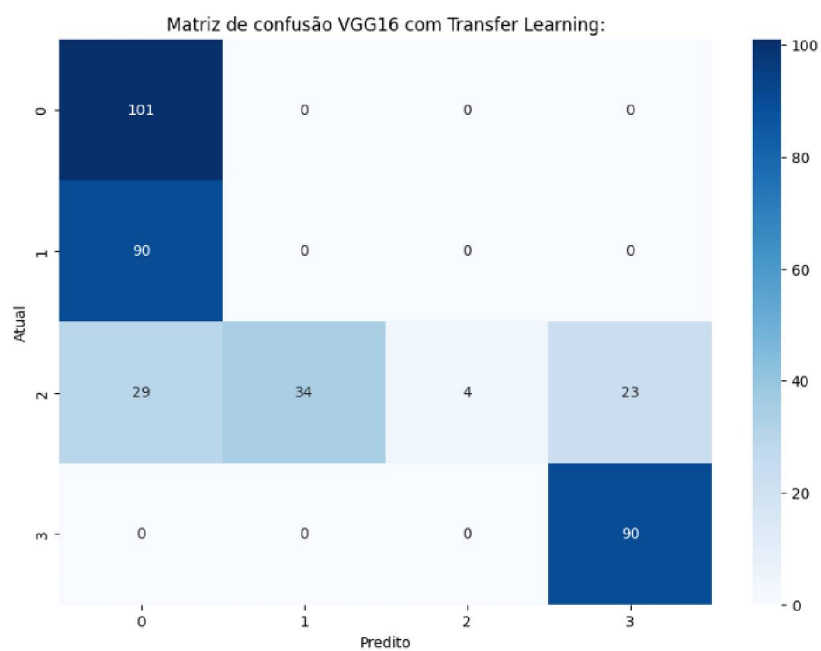
#Calcule as métricas de Sensibilidade, Especificidade e F1-Score com
base em suas matrizes de confusão.
imprime_acuracia(true_classes, pred_classes_tlres, "ResNet50 com
Transfer Learning")

imprime_acuracia(true_classes, pred_classes_tlvvg, "VGG16 com Transfer
Learning:")

```



Acurácia 86.79%  
 Sensibilidade 90.10%  
 Especificidade 98.39%  
 F1 Score 86.79%



Acurácia 52.56%  
Sensibilidade 100.00%  
Especificidade 0.00%  
F1 Score 40.71%

## APÊNDICE 11 – ASPECTOS FILOSÓFICOS E ÉTICOS DA IA

### A – ENUNCIADO

Título do Trabalho: "Estudo de Caso: Implicações Éticas do Uso do ChatGPT"

Trabalho em Grupo: O trabalho deverá ser realizado em grupo de alunos de no máximo seis (06) integrantes.

Objetivo do Trabalho: Investigar as implicações éticas do uso do ChatGPT em diferentes contextos e propor soluções responsáveis para lidar com esses dilemas.

Parâmetros para elaboração do Trabalho:

- 1. Relevância Ética:** O trabalho deve abordar questões éticas significativas relacionadas ao uso da inteligência artificial, especialmente no contexto do ChatGPT. Os alunos devem identificar dilemas éticos relevantes e explorar como esses dilemas afetam diferentes partes interessadas, como usuários, desenvolvedores e a sociedade em geral.
- 2. Análise Crítica:** Os alunos devem realizar uma análise crítica das implicações éticas do uso do ChatGPT em estudos de caso específicos. Eles devem examinar como o algoritmo pode influenciar a disseminação de informações, a privacidade dos usuários e a tomada de decisões éticas. Além disso, devem considerar possíveis vieses algorítmicos, discriminação e questões de responsabilidade.
- 3. Soluções Responsáveis:** Além de identificar os desafios éticos, os alunos devem propor soluções responsáveis e éticas para lidar com esses dilemas. Isso pode incluir sugestões para políticas, regulamentações ou práticas de design que promovam o uso responsável da inteligência artificial. Eles devem considerar como essas soluções podem equilibrar os interesses de diferentes partes interessadas e promover valores éticos fundamentais, como transparência, justiça e privacidade.
- 4. Colaboração e Discussão:** O trabalho deve envolver discussões em grupo e colaboração entre os alunos. Eles devem compartilhar ideias, debater diferentes pontos de vista e chegar a conclusões informadas através do diálogo e da reflexão mútua. O estudo de caso do ChatGPT pode servir como um ponto de partida para essas discussões, incentivando os alunos a aplicar conceitos éticos e legais aprendidos ao analisar um caso concreto.
- 5. Limite de Palavras:** O trabalho terá um limite de 6 a 10 páginas teria aproximadamente entre 1500 e 3000 palavras.
- 6. Estruturação Adequada:** O trabalho siga uma estrutura adequada, incluindo introdução, desenvolvimento e conclusão. Cada seção deve ocupar uma parte proporcional do total de páginas, com a introdução e a conclusão ocupando menos espaço do que o desenvolvimento.
- 7. Controle de Informações:** Evitar incluir informações desnecessárias que possam aumentar o comprimento do trabalho sem contribuir significativamente para o conteúdo. Concentre-se em informações relevantes, argumentos sólidos e evidências importantes para apoiar sua análise.

**8. Síntese e Clareza:** O trabalho deverá ser conciso e claro em sua escrita. Evite repetições desnecessárias e redundâncias. Sintetize suas ideias e argumentos de forma eficaz para transmitir suas mensagens de maneira sucinta.

**9. Formatação Adequada:** O trabalho deverá ser apresentado nas normas da ABNT de acordo com as diretrizes fornecidas, incluindo margens, espaçamento, tamanho da fonte e estilo de citação. Deve-se seguir o seguinte template de arquivo: <https://bibliotecas.ufpr.br/wp-content/uploads/2022/03/template-artigo-de-periodico.docx>

## B – RESOLUÇÃO

### IMPLICAÇÕES ÉTICAS DO USO DO CHATGPT

#### Resumo

Este estudo explora as implicações éticas do uso do ChatGPT, uma ferramenta de inteligência artificial desenvolvida pela OpenAI, com foco em questões como a disseminação de informações falsas, a privacidade dos usuários, a tomada de decisões éticas e os vieses algorítmicos. Por meio de uma análise crítica de estudos de caso, são identificados os impactos negativos do ChatGPT em áreas como suporte médico, educação e recrutamento, onde a confiança cega na IA pode levar a diagnósticos errados, discriminação e perpetuação de desigualdades. As soluções responsáveis propostas incluem a implementação de políticas de transparência, regulamentações que garantam a responsabilidade pelos resultados gerados pela IA e práticas de design ético que incorporem revisão humana em processos críticos. Além disso, destaca-se a importância da educação e conscientização sobre os limites e capacidades da IA, juntamente com a necessidade de colaboração entre desenvolvedores, usuários e legisladores para promover o uso ético da tecnologia. Assim, enfrentar os desafios éticos do ChatGPT requer uma abordagem multifacetada que combine análise crítica, propostas de políticas e colaboração entre as partes interessadas para garantir que o uso da inteligência artificial seja benéfico e justo, equilibrando a inovação tecnológica com valores éticos fundamentais.

Palavras-chave: ChatGPT, ética, inteligência artificial, privacidade, responsabilidade.

#### Abstract

This study explores the ethical implications of using ChatGPT, an artificial intelligence tool developed by OpenAI, focusing on issues such as the dissemination of false information, user privacy, ethical decision-making, and algorithmic biases. Through a critical analysis of case studies, it identifies the negative impacts of ChatGPT in areas like medical support, education, and recruitment, where blind trust in AI can lead to incorrect diagnoses, discrimination, and the perpetuation of inequalities. Proposed responsible solutions include implementing transparency policies, regulations ensuring accountability for AI-generated results, and ethical design practices incorporating human review in critical processes. Additionally, the importance of education and awareness about AI's limits and capabilities is emphasized, along with the need for collaboration among developers, users, and

legislators to promote ethical technology use. Thus, addressing the ethical challenges of ChatGPT requires a multifaceted approach that combines critical analysis, policy proposals, and stakeholder collaboration to ensure the use of artificial intelligence is beneficial and fair, balancing technological innovation with fundamental ethical values.

Keywords: ChatGPT, ethics, artificial intelligence, privacy, responsibility.

## **1. Introdução**

A inteligência artificial tem transformado diversas áreas do conhecimento e da prática humana, trazendo benefícios significativos, mas também desafios éticos que precisam ser cuidadosamente considerados. Entre as diversas aplicações de IA, o ChatGPT, um modelo de linguagem desenvolvido pela OpenAI, tem ganhado destaque por sua capacidade de gerar texto de maneira autônoma e interagir com usuários de forma natural. No entanto, o uso de tecnologias como o ChatGPT levanta questões éticas significativas, que vão desde a disseminação de informações até a privacidade dos usuários e a responsabilidade pelas ações automatizadas.

O objetivo deste trabalho é investigar as implicações éticas do uso do ChatGPT em diferentes contextos, identificando dilemas éticos relevantes e propondo soluções responsáveis para lidar com esses desafios. Para isso, realizaremos uma análise crítica de estudos de caso específicos, examinando como o algoritmo pode influenciar a disseminação de informações, a privacidade dos usuários e a tomada de decisões éticas. Além disso, discutiremos possíveis vieses algorítmicos e questões de responsabilidade, propondo soluções que promovam o uso responsável da inteligência artificial.

### **Relevância Ética**

O ChatGPT, desenvolvido pela OpenAI, é uma ferramenta poderosa que utiliza modelos de linguagem avançados para gerar respostas automatizadas a perguntas de usuários. Apesar de sua utilidade, levanta dilemas éticos significativos. Um dos principais problemas é a disseminação de informações falsas ou enganosas. Como o ChatGPT gera respostas baseadas em vastas quantidades de dados, há o risco de perpetuar desinformação, impactando negativamente a sociedade. Além disso, a privacidade dos usuários é uma preocupação constante, já que a interação com a IA pode envolver dados pessoais sensíveis (Bostrom e Yudkowsky, 2014). A relevância ética também abrange a transparência no funcionamento dos algoritmos. Usuários muitas vezes não têm acesso a como as respostas são geradas ou quais dados foram utilizados, criando uma "caixa preta" que dificulta a accountability. Esta falta de transparência pode resultar em desconfiança e mal-entendidos sobre as capacidades e limitações do ChatGPT.

Questões Éticas Significativas

O uso de modelos de linguagem como o ChatGPT envolve diversos dilemas éticos. Entre os principais, destacam-se:

- **Disseminação de Informações:** A capacidade do ChatGPT de gerar textos pode ser usada tanto para informar quanto para desinformar, levantando questões sobre a veracidade e a precisão das informações disseminadas.
- **Privacidade dos Usuários:** A interação com o ChatGPT pode envolver a coleta e o armazenamento de dados pessoais, o que pode comprometer a privacidade dos usuários se não for gerido de maneira adequada.
- **Responsabilidade:** Quem deve ser responsabilizado por decisões ou ações tomadas com base nas interações com o ChatGPT? Esta questão é crucial, especialmente em contextos onde as consequências podem ser significativas.

#### Impacto nas Partes Interessadas

- **Usuários:** Podem ser impactados pela qualidade e pela veracidade das informações recebidas, bem como pela segurança dos seus dados pessoais.
- **Desenvolvedores:** Devem considerar as implicações éticas durante o desenvolvimento e a implementação de modelos de IA, adotando práticas que minimizem vieses e protejam a privacidade.
- **Sociedade em Geral:** A proliferação de informações geradas por IA pode influenciar a opinião pública e afetar decisões em diversas áreas, desde a política até a saúde.

### Análise Crítica

#### Estudos de Caso Específicos

##### **Caso 1:** Uso do ChatGPT em Atendimento ao Cliente

Análise de como o ChatGPT é utilizado para interagir com clientes, potencialmente substituindo funcionários humanos e levantando questões sobre a qualidade do atendimento e a responsabilidade por erros.

##### **Caso 2:** ChatGPT em Educação

Discussão sobre o uso do ChatGPT como ferramenta educacional, incluindo questões sobre a qualidade das respostas fornecidas e a potencial dependência dos alunos na tecnologia.

##### **Caso 3:** ChatGPT na produção de conteúdo

Discussão sobre o uso do ChatGPT na criação de conteúdo, como textos e imagens, incluindo questões éticas sobre direitos autorais, uma vez que o conteúdo gerado pode ser resultado do aprendizado de máquina executado a partir de outras obras, as quais possuem direitos autorais.

##### **Caso 4:** ChatGPT na Medicina



Em um cenário de suporte médico, por exemplo, a confiança cega nas respostas do ChatGPT pode levar a diagnósticos errados ou tratamentos inadequados. Este aspecto é agravado pela falta de responsabilidade direta, uma vez que a IA é um sistema sem consciência ou moralidade intrínseca (Floridi, 2019).

### **Vieses Algorítmicos e Discriminação**

Os algoritmos de IA, incluindo o ChatGPT, podem refletir e amplificar vieses presentes nos dados de treinamento. Isso pode resultar em discriminação contra certos grupos e na perpetuação de estereótipos. Uma análise crítica desses vieses é essencial para desenvolver soluções que promovam a equidade e a justiça.

Estudo de caso sobre a utilização do ChatGPT em plataformas de recrutamento revelou que o algoritmo tendia a preferir candidatos de um certo perfil demográfico, refletindo vieses históricos presentes nos dados utilizados para seu treinamento (O'Neil, 2016). Esses vieses podem perpetuar desigualdades sociais e econômicas, criando barreiras injustas para grupos minoritários.

A influência do ChatGPT na disseminação de desinformação também merece atenção. Em um cenário político, por exemplo, a capacidade do ChatGPT de gerar conteúdos textuais convincentes pode ser usada para espalhar fake news ou manipular a opinião pública. A falta de um mecanismo robusto para verificação de fatos e a possibilidade de utilização mal-intencionada por terceiros exacerbam o problema.

### **Responsabilidade**

A responsabilidade pelo uso do ChatGPT deve ser compartilhada entre desenvolvedores, reguladores e usuários. Desenvolvedores devem implementar mecanismos para minimizar riscos e aumentar a transparência, enquanto reguladores precisam estabelecer diretrizes claras para o uso ético da IA. Usuários também devem ser educados sobre os potenciais riscos e usos responsáveis da tecnologia.

### **Soluções Responsáveis**

#### **Desafios Éticos**

Os desafios éticos identificados incluem a necessidade de garantir a veracidade das informações geradas, proteger a privacidade dos usuários e estabelecer mecanismos claros de responsabilidade.

Além disso, práticas de design ético devem ser adotadas. Isso inclui a incorporação de mecanismos de revisão humana em processos críticos, como suporte médico ou recrutamento, para mitigar o impacto de possíveis erros ou vieses. O desenvolvimento de sistemas de IA deve seguir princípios éticos que priorizem a justiça, a equidade e a privacidade (Jobin, Ienca e Vayena, 2019).

A educação e a conscientização sobre os limites e capacidades da IA são igualmente importantes. Usuários e desenvolvedores devem ser capacitados para entender os possíveis riscos e tomar decisões informadas sobre o uso da tecnologia. Programas de treinamento e workshops podem ser implementados para promover uma compreensão mais profunda e crítica da IA.

Políticas e regulamentações governamentais também desempenham um papel crucial. Leis que exigem a transparência dos algoritmos e a responsabilidade por seus resultados podem ajudar a proteger os direitos dos usuários e garantir um uso mais ético da IA. Organizações internacionais podem colaborar para estabelecer diretrizes e padrões globais, promovendo uma abordagem harmoniosa e coerente.

Adicionalmente, a criação de comitês de ética em IA nas empresas e instituições de pesquisa pode fornecer um fórum para discutir e avaliar os impactos éticos de novas tecnologias. Esses comitês podem atuar como intermediários entre desenvolvedores, usuários e reguladores, garantindo que diferentes perspectivas sejam consideradas e que decisões informadas sejam tomadas.

### **Propostas de Soluções**

**1. Políticas e Regulamentações:** Desenvolvimento de políticas que promovam a transparência e a responsabilidade no uso do ChatGPT, incluindo diretrizes para a proteção de dados e a mitigação de vieses. Como exemplo podemos citar a proposta europeia de regulamentação, Artificial Intelligence Act.

**2. Práticas de Design Ético:** Implementação de práticas de design que considerem os impactos éticos desde a fase de desenvolvimento, incluindo a realização de auditorias de vieses e testes de equidade.

### **Exemplos Práticos Adicionais**

**1. Automação de Tarefas Administrativas:** O uso do ChatGPT para automatizar tarefas administrativas em empresas pode aumentar a eficiência, mas também pode levar à perda de empregos e à necessidade de requalificação dos trabalhadores afetados.

**2. Suporte Técnico:** Utilizar o ChatGPT como suporte técnico em empresas de tecnologia pode melhorar o atendimento ao cliente, mas levanta questões sobre a precisão das informações fornecidas e a responsabilidade por erros técnicos.

### **Discussão de Impactos Econômicos**

A automação por meio do ChatGPT pode ter impactos significativos no mercado de trabalho, substituindo funções que antes eram realizadas por humanos. Isso pode levar a uma redistribuição das oportunidades de emprego, exigindo que os trabalhadores adquiram novas habilidades para se adaptarem ao novo cenário tecnológico. A discussão ética deve considerar os efeitos econômicos dessa transição e as medidas necessárias para apoiar os trabalhadores afetados.

## **Comparação com Outras Tecnologias**

Comparar as implicações éticas do ChatGPT com outras tecnologias de IA, como assistentes virtuais (e.g., Siri, Alexa) e sistemas de recomendação (e.g., Netflix, Amazon), pode fornecer um contexto mais amplo para a discussão. Essas comparações podem ajudar a identificar melhores práticas e lições aprendidas, destacando como diferentes abordagens podem mitigar riscos éticos e promover o uso responsável da IA.

## **Equilíbrio de Interesses**

As soluções propostas devem equilibrar os interesses de desenvolvedores, usuários e a sociedade em geral. Transparência, justiça e privacidade são valores fundamentais que devem ser promovidos para garantir o uso ético do ChatGPT. Estudos de caso específicos, como a utilização do ChatGPT em plataformas de mídia social, podem servir como pontos de partida para essas discussões, incentivando uma abordagem colaborativa e informada.

O diálogo aberto e contínuo entre as partes interessadas pode facilitar a troca de ideias e a identificação de melhores práticas. Fóruns e conferências sobre ética em IA podem proporcionar espaços para essas interações, promovendo a conscientização e a ação coletiva. A criação de redes de colaboração entre instituições acadêmicas, organizações não governamentais e empresas pode fortalecer os esforços para um uso ético e responsável da IA.

## **Conclusão**

O uso do ChatGPT apresenta diversos dilemas éticos que precisam ser cuidadosamente considerados. As principais questões incluem a disseminação de informações falsas, a privacidade dos usuários e a responsabilidade pelas decisões tomadas com base nas interações com a IA. Uma análise crítica desses dilemas é essencial para desenvolver soluções que promovam o uso responsável da inteligência artificial.

As soluções propostas incluem o desenvolvimento de políticas e regulamentações que promovam a transparência e a responsabilidade, bem como a implementação de práticas de design ético. A educação e conscientização sobre os limites e capacidades da IA são igualmente importantes, capacitando usuários e desenvolvedores a entender os possíveis riscos e a tomar decisões informadas sobre o uso da tecnologia. Além disso, a criação de comitês de ética e a colaboração entre diversas partes interessadas, como desenvolvedores, usuários e legisladores, são passos fundamentais para garantir um uso ético e justo do ChatGPT.

É crucial também considerar os impactos econômicos e sociais da automação proporcionada pelo ChatGPT, especialmente em relação ao mercado de trabalho e à redistribuição das oportunidades de emprego. A comparação com outras tecnologias de IA pode oferecer lições valiosas e destacar melhores práticas para mitigar riscos éticos.

Enfrentar os desafios éticos do ChatGPT requer uma abordagem multifacetada que combine análise crítica, propostas de políticas e colaboração contínua. Somente assim será possível garantir que o uso da inteligência artificial seja benéfico para todos os envolvidos e para a sociedade como um todo, equilibrando a inovação tecnológica com valores éticos fundamentais como transparência, justiça e privacidade. Ao adotar essas medidas, podemos promover um futuro onde a IA contribua positivamente para o desenvolvimento humano e social, respeitando e preservando os direitos e a dignidade de todas as pessoas.

### Referências

- BOSTROM, Nick; YUDKOWSKY, Eliezer. The Ethics of Artificial Intelligence. In: Cambridge Handbook of Artificial Intelligence. Cambridge: Cambridge University Press, 2014.
- DOE, A. Privacy Concerns in AI. International Journal of Data Privacy, 38(2), 89-101, 2021.
- FLORIDI, Luciano. What the Near Future of Artificial Intelligence Could Be. Philosophy & Technology, v. 32, n. 1, p. 1-15, 2019.
- JOBIN, Anna; IENCA, Marcello; VAYENA, Effy. The global landscape of AI ethics guidelines. Nature Machine Intelligence, v. 1, n. 9, p. 389-399, 2019.
- O'NEIL, Cathy. Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy. New York: Crown Publishing Group, 2016.
- OpenAI. ChatGPT: Language Model. Retrieved from <https://www.openai.com/research/chatgpt>, 2023.
- SMITH, J. Ethical Implications of AI. Journal of AI Research, 45(3), 123-145, 2022.
- Artificial Intelligence Act. Retrieved from <https://artificialintelligenceact.eu/>, 2024.

## APÊNDICE 12 – GESTÃO DE PROJETOS DE IA

### A – ENUNCIADO

#### 1 Objetivo

Individualmente, ler e resumir – seguindo o *template* fornecido – um dos artigos abaixo:

AHMAD, L.; ABDELRAZEK, M.; ARORA, C.; BANO, M.; GRUNDY, J. Requirements practices and gaps when engineering human-centered Artificial Intelligence systems. *Applied Soft Computing*. 143. 2023. DOI <https://doi.org/10.1016/j.asoc.2023.110421>

NAZIR, R.; BUCAIONI, A.; PELLICCIONE, P.; Architecting ML-enabled systems: Challenges, best practices, and design decisions. *The Journal of Systems & Software*. 207. 2024. DOI <https://doi.org/10.1016/j.jss.2023.111860>

SERBAN, A.; BLOM, K.; HOOS, H.; VISSER, J. Software engineering practices for machine learning – Adoption, effects, and team assessment. *The Journal of Systems & Software*. 209. 2024. DOI <https://doi.org/10.1016/j.jss.2023.111907>

STEIDL, M.; FELDERER, M.; RAMLER, R. The pipeline for continuous development of artificial intelligence models – Current state of research and practice. *The Journal of Systems & Software*. 199. 2023. DOI <https://doi.org/10.1016/j.jss.2023.111615>

XIN, D.; WU, E. Y.; LEE, D. J.; SALEHI, N.; PARAMESWARAN, A. Whither AutoML? Understanding the Role of Automation in Machine Learning Workflows. In *CHI Conference on Human Factors in Computing Systems (CHI'21)*, Maio 8-13, 2021, Yokohama, Japão. DOI <https://doi.org/10.1145/3411764.3445306>

#### 2 Orientações adicionais

Escolha o artigo que for mais interessante para você. Utilize tradutores e o Chat GPT para entender o conteúdo dos artigos – caso precise, mas escreva o resumo em língua portuguesa e nas suas palavras.

Não esqueça de preencher, no trabalho, os campos relativos ao seu nome e ao artigo escolhido.

No *template*, você deverá responder às seguintes questões:

- Qual o objetivo do estudo descrito pelo artigo?
- Qual o problema/oportunidade/situação que levou a necessidade de realização deste estudo?
- Qual a metodologia que os autores usaram para obter e analisar as informações do estudo?
- Quais os principais resultados obtidos pelo estudo?

Responda cada questão utilizando o espaço fornecido no *template*, sem alteração do tamanho da fonte (Times New Roman, 10), nem alteração do espaçamento entre linhas (1.0).

Não altere as questões do template.

Utilize o editor de textos de sua preferência para preencher as respostas, mas entregue o trabalho em PDF.

## **B – RESOLUÇÃO**



Nome do artigo escolhido: STEIDL FELDERER RAMLER - The pipeline for continuous development of artificial intelligence models

| Qual o <b>objetivo</b> do estudo descrito pelo artigo?                                                                                                                                                                                              | Qual o <b>problema/oportunidade/situação</b> que levou à necessidade de realização desse estudo?                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | Qual a <b>metodologia</b> que os autores usaram para obter e analisar as informações do estudo?                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Quais os <b>principais resultados</b> obtidos pelo estudo?                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>O objetivo principal do artigo é identificar, sintetizar e estruturar sistematicamente ideias conceituais relevantes relacionadas ao Pipeline para desenvolvimento contínuo dentro do desenvolvimento de modelos de inteligência artificial.</p> | <p>As empresas enfrentam continuamente problemas para desenvolver e entregar modelos de inteligência artificial em ambientes de produção complexos enquanto também levam em consideração a qualidade da entrega.</p> <p>Esse estudo teve como oportunidade a sumarização de muitos outros estudos que embora abordassem o Pipeline para desenvolvimento contínuo não tinham como foco o desenvolvimento contínuo para aplicações de inteligência artificial.</p> <p>Há 3 perguntas principais que esse estudo procurou responder:</p> <ol style="list-style-type: none"> <li>1. Quais são os termos mais comumente usados para descrever um Pipeline de desenvolvimento contínuo para IA? E</li> </ol> | <p>A principal metodologia usada para esse estudo foi a MLR ou Multivocal Literature Review. Essa metodologia de estudo permite não apenas usar como referência estudos publicados na literatura, mas também o que foi chamado de “literatura cinza”, que são pesquisas simples realizadas em sites de busca.</p> <p>Essa abordagem foi importante para esse estudo porque o foco do estudo é um assunto relativamente novo e que não possui muitos artigos publicados. Foram analisadas no total 151 fontes relevantes sendo 79 de fontes formais e 71 de fontes informais derivadas da literatura cinza.</p> <p>Além da MLR foi usado também como metodologia a criação taxonômica para categorizar os dados levantados e a análise qualitativa que foi realizada através de entrevistas com 9</p> | <p>Os resultados do estudo foram divididos em três subgrupos remetendo as perguntas que deveriam ser respondidas ao final do estudo.</p> <p>Sobre os termos que são mais usados foram analisados: DevOps para IA, CI/CD para IA, MLOps, Lifecycle Management e CD4ML.</p> <p>Os entrevistados indicaram que as diferenças entre os termos são na verdade desconhecidas na prática. Para o estudo literário a semelhança para todos os termos foi o desenvolvimento contínuo e aprimoramento dos modelos de IA por meio de pipelines.</p> <p>Sobre as principais tarefas que o Pipeline deve executar e os gatilhos para que elas sejam executadas foram identificados no estudo 4 gatilhos: Feedback e sistema de alerta, orquestração de serviços e agendamentos, repositórios e outros. Os modelos de IA precisam ser adaptáveis e treinados para providenciar qualidade por um longo período de tempo. E por isso os gatilhos de contexto específico dependem do modelo de IA, dos requisitos de negócio e de estratégias de treinamento para iniciar e reiniciar o Pipeline.</p> <p>O Pipeline também não é linear e precisa de feedbacks ao longo do processo. E isso é uma característica chave dentro do desenvolvimento agile que permite a melhora contínua e a coleta de dados relevantes.</p> <p>Existem 4 tarefas que são realizadas pelos pipelines: data handling, que usa em média 7% do tempo total de execução. Mas esse tempo pode ser reduzido se for usado uma estratégia de paralelismo. Essa etapa é importante pois os resultados influenciam diretamente na próxima etapa, o treinamento do modelo.</p> <p>O treinamento do modelo é a etapa principal do pipeline, pois é nessa etapa que os dados coletados no data handling serão aplicados à IA.</p> <p>Após o modelo ser desenvolvido ele deve ser preparado para deploy. E para isso existe a terceira tarefa: desenvolvimento de software. Essa etapa inclui tarefas como empacotamento da aplicação, testes de qualidade e versionamento de código. Nessa fase também temos a seleção do melhor modelo de IA.</p> |

|  |                                                                                                                                                                                                                                                                                                                                                                       |                                                                                                                                                                                                                                                                                                                                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <p>como esses termos se diferenciam, nos seus significados específicos?</p> <p>2. Quais tarefas tem que ser conduzidas pelos pipelines de desenvolvimento contínuo da IA? Quais são os possíveis triggers para iniciar o pipeline?</p> <p>3. Quais são os potenciais desafios quando implementar, adaptar e usar um Pipeline de desenvolvimento contínuo para IA?</p> | <p>profissionais que trabalham na área.</p> <p>As entrevistas foram conduzidas de forma semi estruturada e consistem em perguntas introdutórias, perguntas sobre diferentes definições dos pipelines, tarefas que são atribuídas aos pipelines, avaliação da taxonomia proposta e também os desafios que são enfrentados ao usar os pipelines.</p> | <p>A última tarefa ou estágio do pipeline é o sistema de operação. Nessa fase o pipeline executa a entrega do software em ambiente de produção e lida com o contínuo monitoramento dos dados, modelo e sistema.</p> <p>Finalmente para o último tópico abordado, foram identificados 25 possíveis desafios focados na implementação, adaptação e uso dos pipelines para desenvolvimento contínuo de IA. Cada um desses desafios recebeu um peso diferente baseado na quantidade de vezes que foi citado pelas fontes.</p> <p>Para cada uma das tarefas do pipeline foram identificados possíveis desafios como: a integração de dados coletados na etapa de data handling e tempo ideal para atualizar o modelo de treinamento na etapa de treinamento do modelo.</p> <p>Na tarefa de desenvolvimento de software um desafio citado foi a dificuldade de empacotar os modelos de software de forma independente para promover escalabilidade. Por fim, na etapa de operações de sistema foi identificado como desafio garantir que o modelo é compatível com a aplicação consumidora e modelos de comparação.</p> <p>Desafios gerais em relação ao pipeline consistem em garantir a flexibilidade, customização e reutilização, assim como torná-lo mais tolerante a falhas.</p> |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## APÊNDICE 13 – FRAMEWORKS DE INTELIGÊNCIA ARTIFICIAL

### A – ENUNCIADO

#### 1 Classificação (RNA)

Implementar o exemplo de Classificação usando a base de dados Fashion MNIST e a arquitetura RNA vista na aula **FRA - Aula 10 - 2.4 Resolução de exercício de RNA - Classificação**. Além disso, fazer uma breve explicação dos seguintes resultados:

- Gráficos de perda e de acurácia;
  - Imagem gerada na seção “**Mostrar algumas classificações erradas**”, apresentada na aula prática.
- Informações:

- **Base de dados:** Fashion MNIST Dataset
- **Descrição:** Um dataset de imagens de roupas, onde o objetivo é classificar o tipo de vestuário. É semelhante ao famoso dataset MNIST, mas com peças de vestuário em vez de dígitos.
- **Tamanho:** 70.000 amostras, 784 features (28x28 pixels).
- **Importação do dataset:** Copiar código abaixo.

```
data = tf.keras.datasets.fashion_mnist
(x_train, y_train), (x_test, y_test) = fashion_mnist.load_data()
```

#### 2 Regressão (RNA)

Implementar o exemplo de Classificação usando a base de dados Wine Dataset e a arquitetura RNA vista na aula **FRA - Aula 12 - 2.5 Resolução de exercício de RNA - Regressão**. Além disso, fazer uma breve explicação dos seguintes resultados:

- Gráficos de avaliação do modelo (loss);
- Métricas de avaliação do modelo (pelo menos uma entre MAE, MSE,  $R^2$ ).

Informações:

- **Base de dados:** Wine Quality
- **Descrição:** O objetivo deste dataset prever a qualidade dos vinhos com base em suas características químicas. A variável target (y) neste exemplo será o score de qualidade do vinho, que varia de 0 (pior qualidade) a 10 (melhor qualidade)
- **Tamanho:** 1599 amostras, 12 features.
- **Importação:** Copiar código abaixo.

```
url =
"https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv"
data = pd.read_csv(url, delimiter=';')
```

Dica 1. Para facilitar o trabalho, renomeie o nome das colunas para português, dessa forma:

```
data.columns = [
    'acidez_fixa',          # fixed acidity
    'acidez_volatil',       # volatile acidity
    'acido_citrico',        # citric acid
    'acucar_residual',      # residual sugar
    'cloretos',             # chlorides
    'dioxido_de_enxofre_livre', # free sulfur dioxide
    'dioxido_de_enxofre_total', # total sulfur dioxide
    'densidade',            # density
    'pH',                   # pH
    'sulfatos',             # sulphates
    'alcool',               # alcohol
    'score_qualidade_vinho' # quality
]
```

Dica 2. Separe os dados (x e y) de tal forma que a última coluna (índice -1), chamada score\_qualidade\_vinho, seja a variável target (y)

### 3 Sistemas de Recomendação

Implementar o exemplo de Sistemas de Recomendação usando a base de dados Base\_livros.csv e a arquitetura vista na aula **FRA - Aula 22 - 4.3 Resolução do Exercício de Sistemas de Recomendação**. Além disso, fazer uma breve explicação dos seguintes resultados:

- Gráficos de avaliação do modelo (loss);
- Exemplo de recomendação de livro para determinado Usuário.

Informações:

- **Base de dados:** Base\_livros.csv
- **Descrição:** Esse conjunto de dados contém informações sobre avaliações de livros (Notas), nomes de livros (Titulo), ISBN e identificação do usuário (ID\_usuario)
- **Importação:** Base de dados disponível no Moodle (UFPR Virtual), chamada Base\_livros (formato .csv).

## 4 Deepdream

Implementar o exemplo de implementação mínima de Deepdream usando uma imagem de um felino - retirada do site Wikipedia - e a arquitetura Deepdream vista na aula **FRA - Aula 23 - Prática Deepdream**. Além disso, fazer uma breve explicação dos seguintes resultados:

- Imagem onírica obtida por *Main Loop*;
- Imagem onírica obtida ao levar o modelo até uma oitava;
- Diferenças entre imagens oníricas obtidas com *Main Loop* e levando o modelo até a oitava.

Informações:

- **Base de dados:** [https://commons.wikimedia.org/wiki/File:Felis\\_catus-cat\\_on\\_snow.jpg](https://commons.wikimedia.org/wiki/File:Felis_catus-cat_on_snow.jpg)
- **Importação da imagem:** Copiar código abaixo.

```
url =
"https://commons.wikimedia.org/wiki/Special:FilePath/Felis\_catus-cat\_on\_snow.jpg"
```

Dica: Para exibir a imagem utilizando display (display.html) use o link [https://commons.wikimedia.org/wiki/File:Felis\\_catus-cat\\_on\\_snow.jpg](https://commons.wikimedia.org/wiki/File:Felis_catus-cat_on_snow.jpg)

## B – RESOLUÇÃO

### 1. Classificação RNA

```
data = tf.keras.datasets.fashion_mnist
(x_train, y_train), (x_test, y_test) = data.load_data()
x_train, x_test = x_train/255.0, x_test/255.0
i = tf.keras.layers.Input(shape=(28, 28))
x = tf.keras.layers.Flatten()(i)
x = tf.keras.layers.Dense(128, activation="relu")(x)
x = tf.keras.layers.Dropout(0.2)(x)
x = tf.keras.layers.Dense(10, activation="softmax")(x)

model = tf.keras.models.Model(i, x)
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
```

Epoch 1/12

1875/1875 ————— 6s 2ms/step - accuracy: 0.7677 - loss: 0.6725 - val\_accuracy: 0.8473 - val\_loss: 0.4200

Epoch 2/12

1875/1875 ————— 4s 2ms/step - accuracy: 0.8518 - loss: 0.4089 - val\_accuracy: 0.8598 - val\_loss: 0.3856

Epoch 3/12

1875/1875 ————— 4s 2ms/step - accuracy: 0.8672 - loss: 0.3684 - val\_accuracy: 0.8671 - val\_loss: 0.3695

```

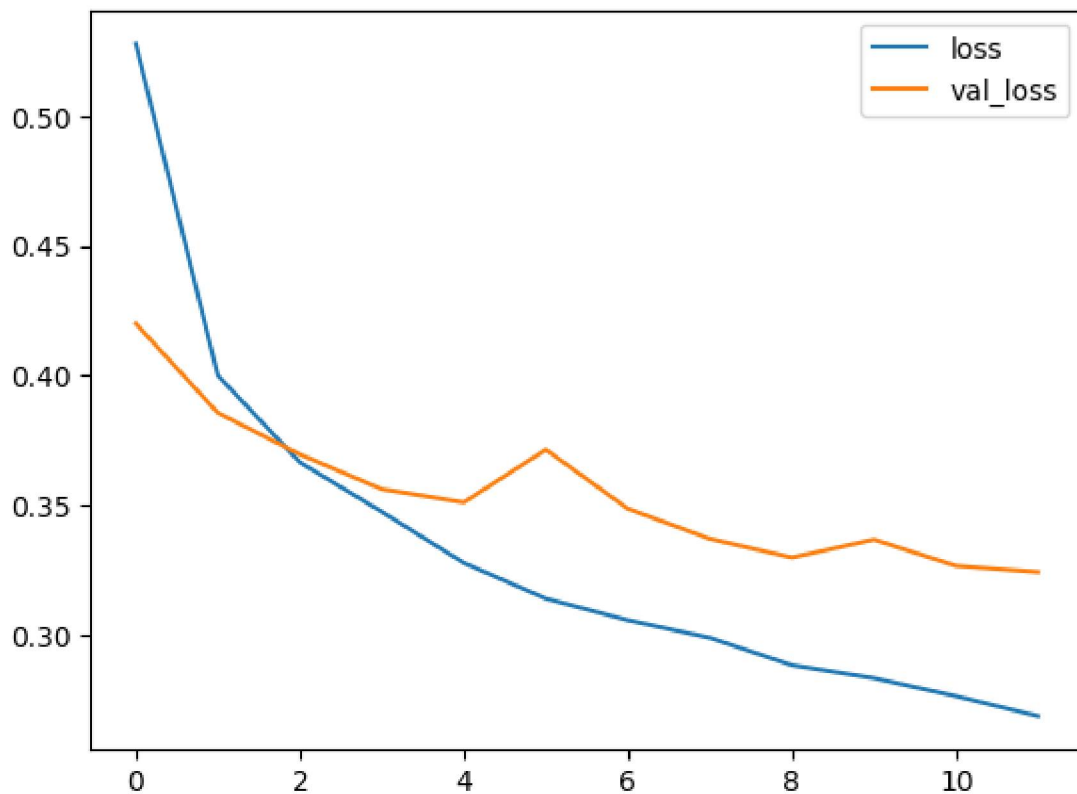
Epoch 4/12
1875/1875 ————— 5s 3ms/step - accuracy: 0.8731 -
loss: 0.3474 - val_accuracy: 0.8704 - val_loss: 0.3561
Epoch 5/12
1875/1875 ————— 6s 3ms/step - accuracy: 0.8782 -
loss: 0.3273 - val_accuracy: 0.8754 - val_loss: 0.3510
Epoch 6/12
1875/1875 ————— 4s 2ms/step - accuracy: 0.8815 -
loss: 0.3160 - val_accuracy: 0.8681 - val_loss: 0.3713
Epoch 7/12
1875/1875 ————— 3s 2ms/step - accuracy: 0.8861 -
loss: 0.3071 - val_accuracy: 0.8784 - val_loss: 0.3485
Epoch 8/12
1875/1875 ————— 6s 2ms/step - accuracy: 0.8915 -
loss: 0.2939 - val_accuracy: 0.8773 - val_loss: 0.3369
Epoch 9/12
1875/1875 ————— 4s 2ms/step - accuracy: 0.8957 -
loss: 0.2819 - val_accuracy: 0.8808 - val_loss: 0.3297
Epoch 10/12
1875/1875 ————— 3s 2ms/step - accuracy: 0.8949 -
loss: 0.2799 - val_accuracy: 0.8808 - val_loss: 0.3366
Epoch 11/12
1875/1875 ————— 4s 2ms/step - accuracy: 0.8988 -
loss: 0.2726 - val_accuracy: 0.8857 - val_loss: 0.3264
Epoch 12/12
1875/1875 ————— 4s 2ms/step - accuracy: 0.8975 -
loss: 0.2684 - val_accuracy: 0.8849 - val_loss: 0.3241

```

```

# Função de perda
plt.plot(r.history["loss"], label="loss")
plt.plot(r.history["val_loss"], label="val_loss")
plt.legend()

```

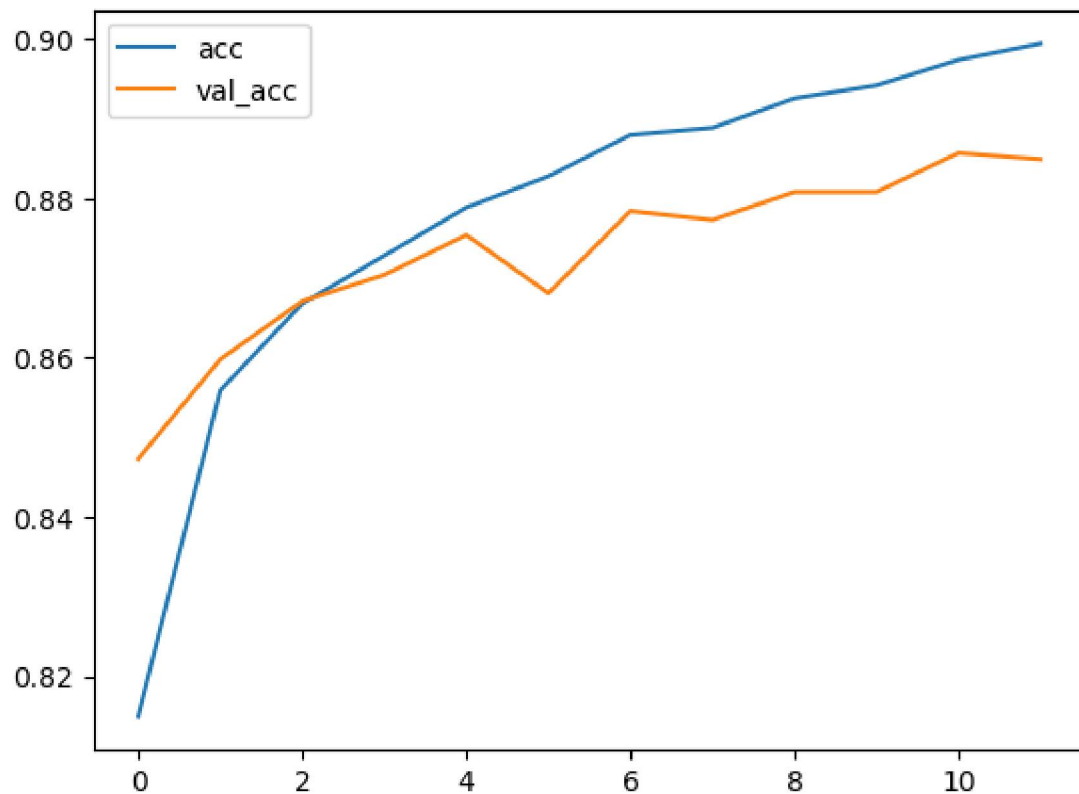


```

# Acurácia

```

```
plt.plot(r.history["accuracy"], label="acc")
plt.plot(r.history["val_accuracy"], label="val_acc")
plt.legend()
```



```
# Avaliar o modelo com a base de teste
print( model.evaluate(x_test, y_test) )
```

```
313/313 ————— 0s 1ms/step - accuracy: 0.8854 -
loss: 0.3246
```

```
y_pred = model.predict(x_test).argmax(axis=1)
print(y_pred)
```

```
[9 2 1 ... 8 1 5]
```

```
# Matriz de confusão
cm = confusion_matrix(y_test, y_pred)
plot_confusion_matrix(conf_mat=cm, figsize=(7, 7),
                      show_normed=True)
```



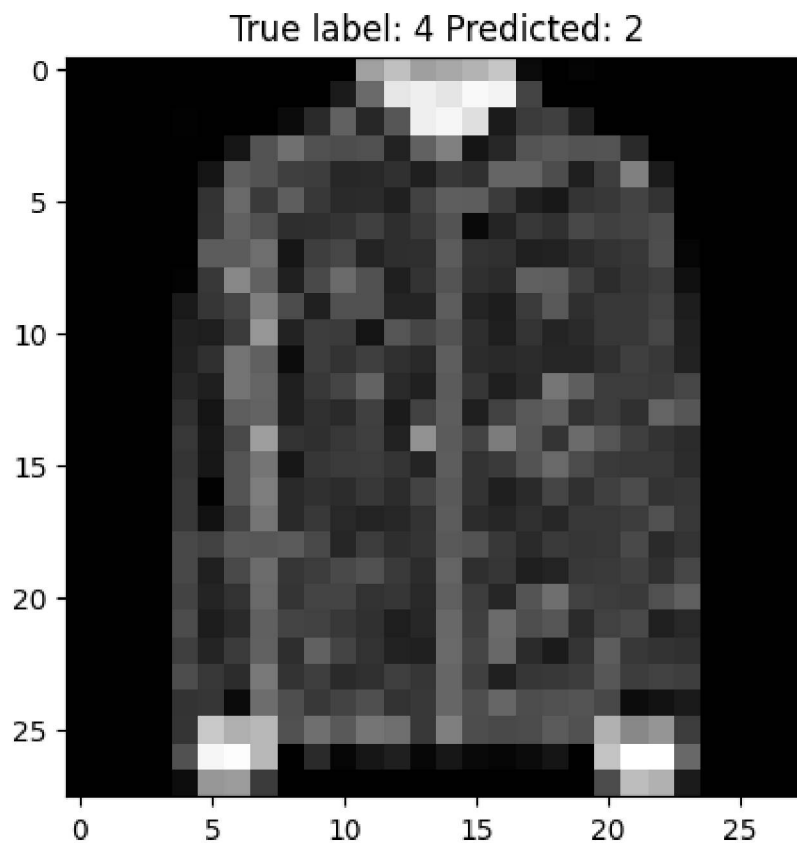
|            |   |                 |               |               |               |               |               |               |               |               |               |
|------------|---|-----------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
| true label | 0 | 877<br>(0.88)   | 4<br>(0.00)   | 19<br>(0.02)  | 14<br>(0.01)  | 0<br>(0.00)   | 1<br>(0.00)   | 77<br>(0.08)  | 0<br>(0.00)   | 8<br>(0.01)   | 0<br>(0.00)   |
|            | 1 | 3<br>(0.00)     | 975<br>(0.97) | 1<br>(0.00)   | 14<br>(0.01)  | 2<br>(0.00)   | 0<br>(0.00)   | 4<br>(0.00)   | 0<br>(0.00)   | 1<br>(0.00)   | 0<br>(0.00)   |
|            | 2 | 21<br>(0.02)    | 3<br>(0.00)   | 828<br>(0.83) | 8<br>(0.01)   | 74<br>(0.07)  | 0<br>(0.00)   | 66<br>(0.07)  | 0<br>(0.00)   | 0<br>(0.00)   | 0<br>(0.00)   |
|            | 3 | 23<br>(0.02)    | 20<br>(0.02)  | 11<br>(0.01)  | 884<br>(0.88) | 34<br>(0.03)  | 0<br>(0.00)   | 26<br>(0.03)  | 0<br>(0.00)   | 2<br>(0.00)   | 0<br>(0.00)   |
|            | 4 | 0<br>(0.00)     | 1<br>(0.00)   | 121<br>(0.12) | 26<br>(0.03)  | 772<br>(0.77) | 0<br>(0.00)   | 80<br>(0.08)  | 0<br>(0.00)   | 0<br>(0.00)   | 0<br>(0.00)   |
|            | 5 | 0<br>(0.00)     | 0<br>(0.00)   | 0<br>(0.00)   | 1<br>(0.00)   | 0<br>(0.00)   | 957<br>(0.96) | 0<br>(0.00)   | 22<br>(0.02)  | 0<br>(0.00)   | 20<br>(0.02)  |
|            | 6 | 161<br>(0.16)   | 1<br>(0.00)   | 102<br>(0.10) | 27<br>(0.03)  | 45<br>(0.04)  | 0<br>(0.00)   | 656<br>(0.66) | 0<br>(0.00)   | 8<br>(0.01)   | 0<br>(0.00)   |
|            | 7 | 0<br>(0.00)     | 0<br>(0.00)   | 0<br>(0.00)   | 0<br>(0.00)   | 0<br>(0.00)   | 11<br>(0.01)  | 0<br>(0.00)   | 969<br>(0.97) | 0<br>(0.00)   | 20<br>(0.02)  |
|            | 8 | 4<br>(0.00)     | 1<br>(0.00)   | 5<br>(0.01)   | 3<br>(0.00)   | 3<br>(0.00)   | 1<br>(0.00)   | 2<br>(0.00)   | 4<br>(0.00)   | 977<br>(0.98) | 0<br>(0.00)   |
|            | 9 | 0<br>(0.00)     | 0<br>(0.00)   | 0<br>(0.00)   | 0<br>(0.00)   | 0<br>(0.00)   | 6<br>(0.01)   | 1<br>(0.00)   | 39<br>(0.04)  | 0<br>(0.00)   | 954<br>(0.95) |
|            |   | predicted label |               |               |               |               |               |               |               |               |               |

```

misclassified = np.where(y_pred != y_test)[0]

i = np.random.choice(misclassified)
plt.imshow(x_test[i].reshape(28, 28), cmap="gray")
plt.title("True label: %s Predicted: %s" % (y_test[i], y_pred[i]))

```



Gráficos de perda e de acurácia:

Os gráficos de perda e acurácia apresentam uma pequena diferença entre os dados de validação e os dados de treinamento apresenta grande divergência. Tanto os valores de perda quanto de acurácia chegam perto dos valores ideais a partir de 12 epochs, porém ainda podem ser melhorados.

Imagem gerada na seção “Mostrar algumas classificações erradas”, apresentada na aula prática:

A imagem classificada de forma errada foi categorizada como label 2 (puloover), quando deveria ter sido classificada como 4 (coat).

## 2. Regressão RNA

```
url
="https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv"
data = pd.read_csv(url, delimiter=';')

data.columns = [
'acidez_fixa', # fixed acidity
'acidez_volatil', # volatile acidity
'acido_citrico', # citric acid
'acucar_residual', # residual sugar
'cloretos', # chlorides
'dioxido_de_enxofre_livre', # free sulfur dioxide
'dioxido_de_enxofre_total', # total sulfur dioxide
```

```

'densidade', # density
'pH', # pH
'sulfatos', # sulphates
'alcool', # alcohol
'score_qualidade_vinho' # quality
]
data = shuffle(data, random_state=4)
y = data.score_qualidade_vinho
X = data.drop('score_qualidade_vinho', axis=1).astype(float)
X = StandardScaler().fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.25,
                                                    random_state=42)

def modelo(input_shape):
    model = Sequential()

    model.add(Dense(64, activation='relu', input_shape=(input_shape,)))
    model.add(Dense(32, activation='relu'))
    model.add(Dense(16, activation='relu'))
    model.add(Dense(1))

    return model

# Criação de funções para as métricas R2 e RMSE serem inseridas no
modelo
def rmse(y_true, y_pred):
    y_true = tf.cast(y_true, dtype=tf.float32)
    return backend.sqrt(backend.mean( backend.square(y_pred - y_true),
axis=-1) )

def r2(y_true, y_pred):
    y_true = tf.cast(y_true, dtype=tf.float32)
    media = backend.mean(y_true)
    num = backend.sum (backend.square(y_true - y_pred))
    den = backend.sum (backend.square(y_true - media))
    return (1.0 - num/den)

#optimizer=tf.keras.optimizers.Adam(learning_rate=0.01)
#optimizer=tf.keras.optimizers.SGD(learning_rate=0.001, momentum=0.5)
optimizer=tf.keras.optimizers.RMSprop(0.001)

r = modelo(X.shape[1])
r.compile(optimizer=optimizer, loss='mse', metrics=[rmse,r2])
# Early stop para epochs
early_stop = tf.keras.callbacks.EarlyStopping(
                                monitor='val_loss',
                                patience=20,
                                restore_best_weights=True)
rf = r.fit(X_train, y_train, epochs=50, validation_data=(X_test,
y_test), verbose=1, batch_size=32, validation_split=0.1,
callbacks=[early_stop])

```

Epoch 1/50

**38/38** ————— **2s** 14ms/step - loss: 27.9201 - r2: -42.0333 - rmse: 5.1883 - val\_loss: 11.3992 - val\_r2: -16.7498 - val\_rmse: 3.2062

Epoch 2/50

**38/38** ————— **0s** 2ms/step - loss: 8.0132 - r2: -11.7250 - rmse: 2.5300 - val\_loss: 2.5625 - val\_r2: -2.8855 - val\_rmse: 1.2871

Epoch 3/50

```

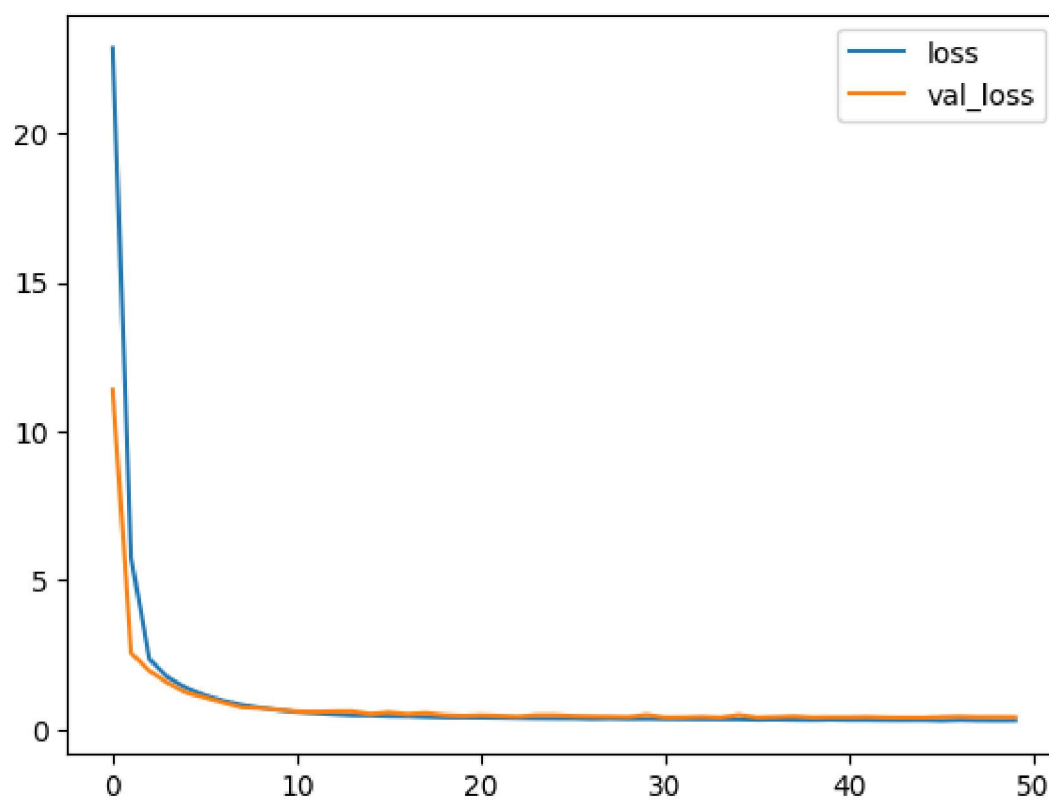
38/38 ----- 0s 2ms/step - loss: 2.4152 - r2:
-2.9216 - rmse: 1.2291 - val_loss: 1.9616 - val_r2: -1.9341 - val_rmse:
1.0730
Epoch 4/50
38/38 ----- 0s 3ms/step - loss: 1.8582 - r2:
-2.2753 - rmse: 1.0725 - val_loss: 1.5563 - val_r2: -1.3211 - val_rmse:
0.9613
Epoch 5/50
38/38 ----- 0s 3ms/step - loss: 1.5017 - r2:
-1.5115 - rmse: 0.9694 - val_loss: 1.2303 - val_r2: -0.8362 - val_rmse:
0.8751
Epoch 6/50
38/38 ----- 0s 3ms/step - loss: 1.0801 - r2:
-0.9427 - rmse: 0.8127 - val_loss: 1.0732 - val_r2: -0.5939 - val_rmse:
0.7948
Epoch 7/50
38/38 ----- 0s 3ms/step - loss: 1.0575 - r2:
-0.7507 - rmse: 0.8187 - val_loss: 0.9038 - val_r2: -0.3392 - val_rmse:
0.7339
Epoch 8/50
38/38 ----- 0s 3ms/step - loss: 0.8625 - r2:
-0.4885 - rmse: 0.7367 - val_loss: 0.7454 - val_r2: -0.1078 - val_rmse:
0.6845
Epoch 9/50
38/38 ----- 0s 3ms/step - loss: 0.7502 - r2:
-0.2072 - rmse: 0.6744 - val_loss: 0.7107 - val_r2: -0.0516 - val_rmse:
0.6648
Epoch 10/50
38/38 ----- 0s 3ms/step - loss: 0.6289 - r2:
-0.0569 - rmse: 0.6226 - val_loss: 0.6625 - val_r2: 0.0268 - val_rmse: 0.6240
Epoch 11/50
38/38 ----- 0s 3ms/step - loss: 0.6491 - r2:
0.0266 - rmse: 0.6196 - val_loss: 0.6018 - val_r2: 0.1172 - val_rmse: 0.6006
Epoch 12/50
38/38 ----- 0s 3ms/step - loss: 0.5317 - r2:
0.0884 - rmse: 0.5630 - val_loss: 0.5869 - val_r2: 0.1484 - val_rmse: 0.5846
Epoch 13/50
...
Epoch 49/50
38/38 ----- 0s 4ms/step - loss: 0.3010 - r2:
0.5038 - rmse: 0.4262 - val_loss: 0.4228 - val_r2: 0.3875 - val_rmse: 0.5030
Epoch 50/50
38/38 ----- 0s 3ms/step - loss: 0.2911 - r2:
0.5150 - rmse: 0.4137 - val_loss: 0.4192 - val_r2: 0.3946 - val_rmse: 0.4930

```

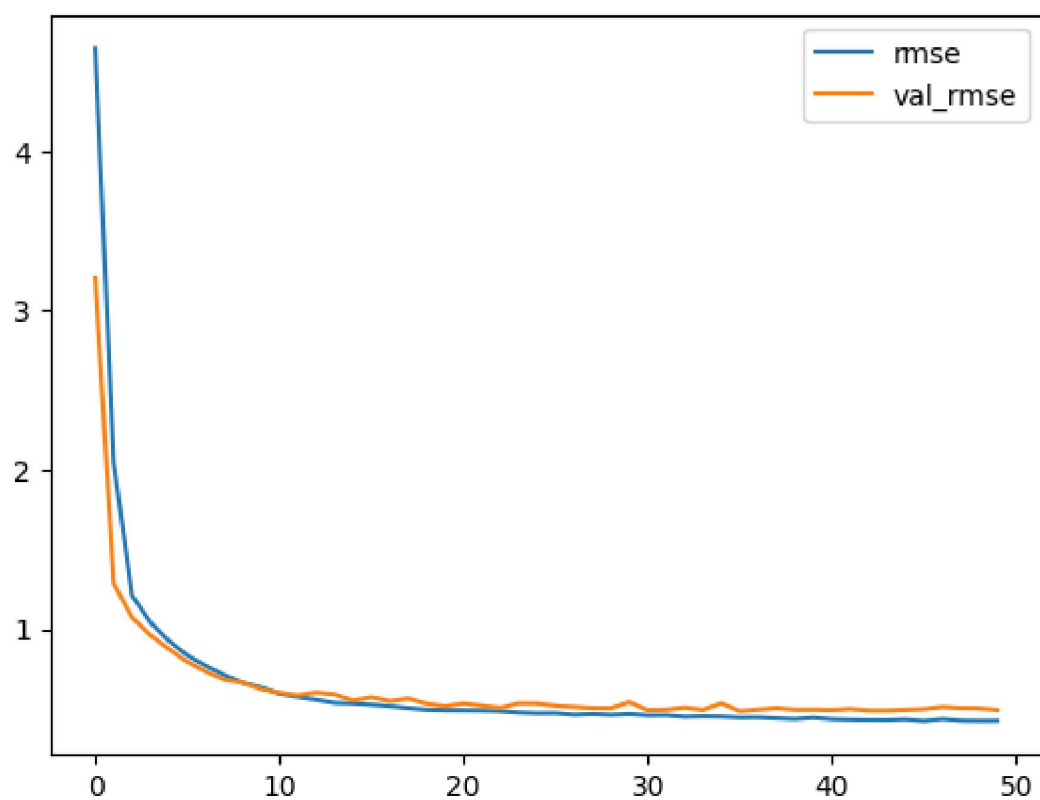
```

plt.plot( rf.history["loss"], label="loss" )
plt.plot( rf.history["val_loss"], label="val_loss" )
plt.legend()

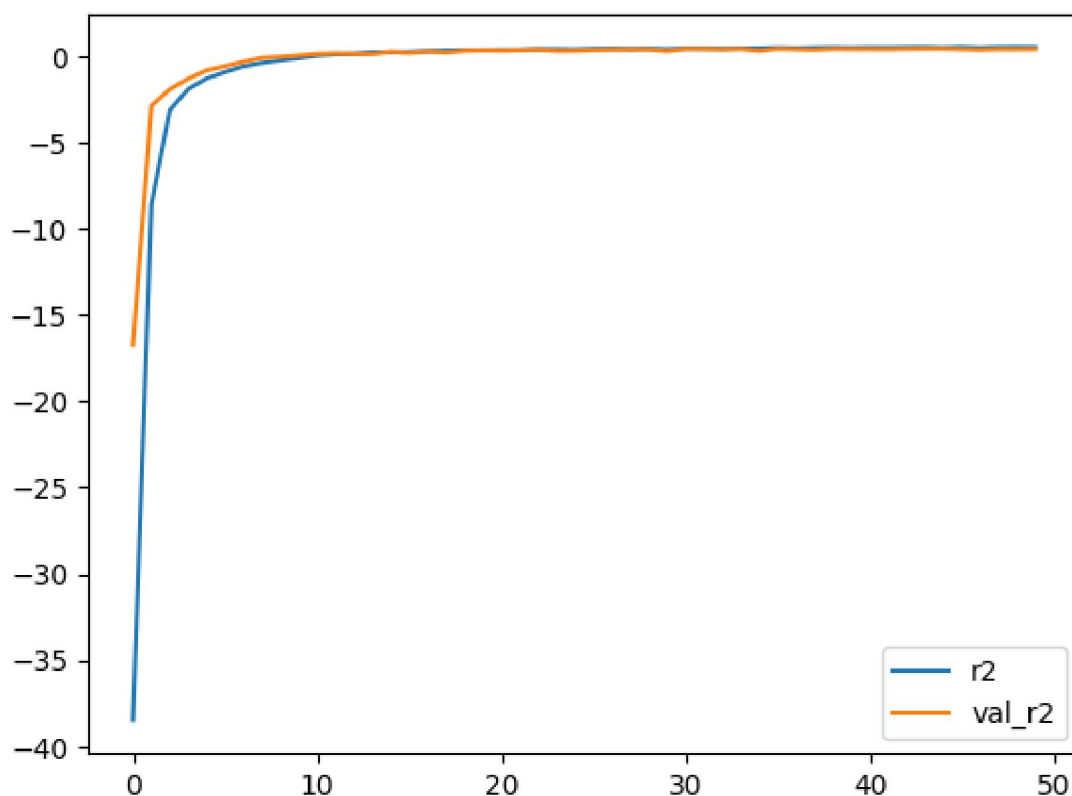
```



```
plt.plot( rf.history["rmse"], label="rmse" )  
plt.plot( rf.history["val_rmse"], label="val_rmse" )  
plt.legend()
```



```
plt.plot( rf.history["r2"], label="r2" )
plt.plot( rf.history["val_r2"], label="val_r2" )
plt.legend()
```



```
# Predição
y_pred = r.predict(X_test).flatten()
# Cálculo das métricas de acurácia: mse, r2 e rmse
mse = mean_squared_error(y_test, y_pred)
rmse = sqrt(mse)
r2 = r2_score(y_test, y_pred)
# Resultados das métricas de acurácia
print("mse      = ", mse)
print("rmse     = ", rmse)
print("r2       = ", r2)
```

```
mse      = 0.40027044277371315
rmse     = 0.6326692996927488
r2       = 0.41446149349212646
```

Gráficos de avaliação do modelo (loss):

Os valores da função de perda chegam próximo do mínimo a partir de aproximadamente 10 epochs.

Métricas de avaliação do modelo (pelo menos uma entre MAE, MSE, R<sup>2</sup>):

A métrica R<sup>2</sup>, coeficiente de determinação, ficou próxima de 41%, ou seja, o modelo não se ajusta muito bem à amostra e não explica adequadamente a variância dos dados.



### 3. Sistemas de Recomendação

```

df = pd.read_csv("/content/Base_livros(in).csv", on_bad_lines='skip')
df.head()
# Renomeia nome de coluna com sintaxe problematica
df.rename(columns={'Notas;;;;;;;;;': 'Notas'}, inplace = True)
df.head()
# Remove caracteres ";" da coluna de notas
df['Notas'] = df['Notas'].str.replace(';;;;;;;;;', '')
df.head()
# Transforma String em Int para obter a media depois
df.Notas = pd.to_numeric(df.Notas, errors='coerce').fillna(0).astype(np.int64)
df.head()
# Embeddings > devem ser categóricos

df.ISBN = pd.Categorical(df.ISBN)
df['new_isbn'] = df.ISBN.cat.codes

df.ID_usuario = pd.Categorical(df.ID_usuario)
df['new_id_usuario'] = df.ID_usuario.cat.codes
# Dimensões
N = len(set(df.new_isbn))
M = len(set(df.new_id_usuario))

# dimensão do embedding (tentar outros)
K = 10
# usuário
u = Input(shape=(1,))
u_emb = Embedding(N, K)(u) # saída : num_samples, 1, K
u_emb = Flatten()(u_emb) # saída : num_samples, K

# livro
m = Input(shape=(1,))
m_emb = Embedding(M, K)(m) # saída : num_samples, 1, K
m_emb = Flatten()(m_emb) # saída : num_samples, K

x = Concatenate()([u_emb, m_emb])

x = Dense(1024, activation="relu")(x)
x = Dense(1)(x)

model = Model(inputs=[u, m], outputs=x)
model.compile(
    loss="mse",
    optimizer=SGD(learning_rate=0.08, momentum=0.9)
)
ID_usuario, ISBN, Notas = shuffle(df.new_id_usuario, df.new_isbn,
df.Notas)

Ntrain = int(0.8 * len(Notas)) # separar os dados 80% x 20%

train_user = ID_usuario[:Ntrain]
train_livros = ISBN[:Ntrain]
train_notas = Notas[:Ntrain]
test_user = ID_usuario[Ntrain:]
test_livros = ISBN[Ntrain:]

```



```

test_notas = Notas[Ntrain:]

# centralizar as notas
avg_notas = train_notas.mean()
train_notas = train_notas - avg_notas
test_notas = test_notas - avg_notas
epochs = 25
r = model.fit(
    x=[train_user, train_livros],
    y=train_notas,
    epochs=epochs,
    batch_size=1024,
    verbose=2, # não imprime o progresso
    validation_data=([test_user, test_livros], test_notas)
)

```

```

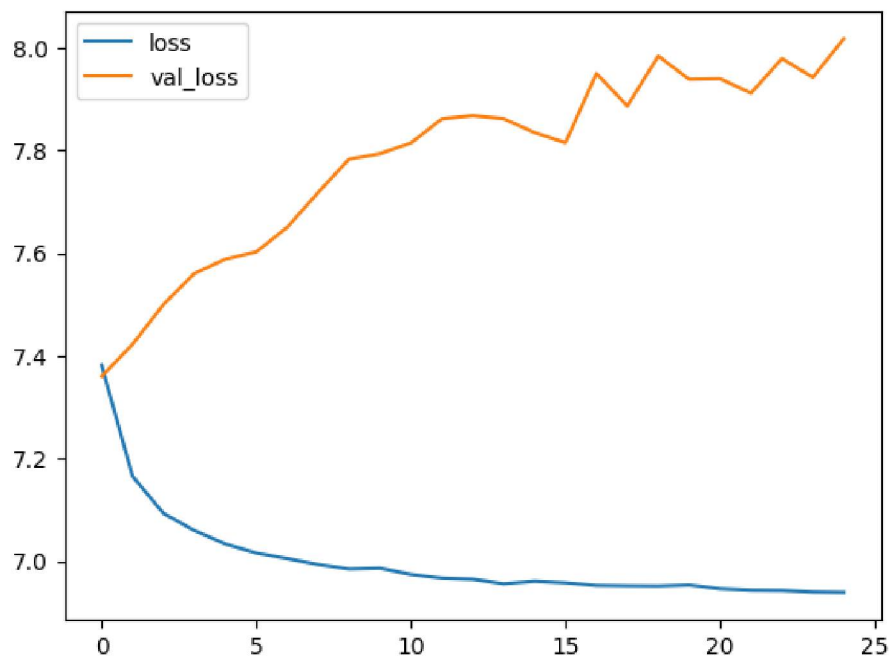
Epoch 1/25
101/101 - 2s - 24ms/step - loss: 8.8632 - val_loss: 8.5136
Epoch 2/25
101/101 - 1s - 8ms/step - loss: 8.4785 - val_loss: 8.5022
Epoch 3/25
101/101 - 0s - 3ms/step - loss: 8.4519 - val_loss: 8.5759
Epoch 4/25
101/101 - 0s - 3ms/step - loss: 8.4385 - val_loss: 8.5391
Epoch 5/25
101/101 - 0s - 2ms/step - loss: 8.3639 - val_loss: 8.5562
Epoch 6/25
101/101 - 0s - 2ms/step - loss: 8.2941 - val_loss: 8.6671
Epoch 7/25
101/101 - 0s - 3ms/step - loss: 8.1696 - val_loss: 8.7663
Epoch 8/25
101/101 - 0s - 3ms/step - loss: 8.0487 - val_loss: 8.6870
Epoch 9/25
101/101 - 0s - 3ms/step - loss: 7.9515 - val_loss: 8.8622
Epoch 10/25
101/101 - 0s - 3ms/step - loss: 7.8697 - val_loss: 8.8549
Epoch 11/25
101/101 - 0s - 2ms/step - loss: 7.8054 - val_loss: 8.7998
Epoch 12/25
101/101 - 0s - 3ms/step - loss: 7.7270 - val_loss: 8.7818
Epoch 13/25
...
Epoch 24/25
101/101 - 0s - 3ms/step - loss: 7.0006 - val_loss: 8.9692
Epoch 25/25
101/101 - 0s - 3ms/step - loss: 6.9814 - val_loss: 9.0859

```

```

plt.plot(r.history["loss"], label="loss")
plt.plot(r.history["val_loss"], label="val_loss")
plt.legend()
plt.show()

```



```
# Gerar o array com o usuário único
# repete a quantidade de filmes
input_usuario = np.repeat(a=276725, repeats=N)
livro = np.array(list(set(ISBN)))

preds = model.predict( [input_usuario, livro] )

# descentraliza as predições
rat = preds.flatten() + avg_notas

# índice da maior nota
idx = np.argmax(rat)

print("Recomendação: Livro - ", livro[idx], " / ", rat[idx] , "**")
```

Função de perda:

Aqui podemos ver que os dados de validação estão indo para uma tendência contrária aos dados de treinamento. E a diferença entre os dados de validação e os dados de treinamento apresenta grande divergência. Uma forma para mudarmos isso seria aumentarmos a amostra dos dados de teste. Também seria interessante para esse caso aumentar a quantidade de Epoch, mais do que 25, para que a perda siga a tendência de chegar a 0.

Recomendação específica para usuário:

Para esse usuário específico o livro 5853 foi indicado com uma nota de 9.6.

#### 4. Deepdream

```

url =
'https://commons.wikimedia.org/wiki/Special:FilePath/Felis_catus-cat_on_snow.jpg'
# Download da imagem e gravação em array Numpy
def download(url, max_dim=None):
    name = url.split('/')[-1]
    image_path = tf.keras.utils.get_file(name, origin=url)
    img = PIL.Image.open(image_path)
    if max_dim:
        img.thumbnail((max_dim, max_dim))
    return np.array(img)

# Normalização da imagem
def deprocess(img):
    img = 255*(img + 1.0)/2.0
    return tf.cast(img, tf.uint8)

# Mostra a imagem
def show(img):
    display.display(PIL.Image.fromarray(np.array(img)))

# Redução do tamanho da imagem para facilitar o trabalho da RNN
original_img = download(url, max_dim=500)
show(original_img)
display.display(display.HTML('Image cc-by: Von.grzanka'))

```



```

base_model = tf.keras.applications.InceptionV3(include_top=False,
weights='imagenet')
# Maximizando as ativações das camadas
names = ['mixed3', 'mixed5']
layers = [base_model.get_layer(name).output for name in names]

# Criação do modelo
dream_model = tf.keras.Model(inputs=base_model.input, outputs=layers)
def calc_loss(img, model):

```

```

# Passe a imagem pelo modelo para recuperar as ativações.
# Converte a imagem em um batch de tamanho 1.
img_batch = tf.expand_dims(img, axis=0)
layer_activations = model(img_batch)
if len(layer_activations) == 1:
    layer_activations = [layer_activations]

losses = []
for act in layer_activations:
    loss = tf.math.reduce_mean(act)
    losses.append(loss)

return tf.reduce_sum(losses)
class DeepDream(tf.Module):
    def __init__(self, model):
        self.model = model

    @tf.function(
        input_signature=(
            tf.TensorSpec(shape=[None, None, 3], dtype=tf.float32),
            tf.TensorSpec(shape=[], dtype=tf.int32),
            tf.TensorSpec(shape=[], dtype=tf.float32),
        )
    )
    def __call__(self, img, steps, step_size):
        print("Tracing")
        loss = tf.constant(0.0)

        for n in tf.range(steps):
            with tf.GradientTape() as tape:
                # Gradientes relativos a img
                tape.watch(img)
                loss = calc_loss(img, self.model)

            # Calculo do gradiente da perda em relação aos pixels da imagem
            # de entrada.
            gradients = tape.gradient(loss, img)

            # Normalizacao dos gradintnes
            gradients /= tf.math.reduce_std(gradients) + 1e-8

            # Na subida gradiente, a "perda" é maximizada.
            # Você pode atualizar a imagem adicionando diretamente os
            # gradientes (porque eles têm o mesmo formato!)
            img = img + gradients*step_size
            img = tf.clip_by_value(img, -1, 1)

        return loss, img
deepdream = DeepDream(dream_model)
def run_deep_dream_simple(img, steps=100, step_size=0.01):

    img = tf.keras.applications.inception_v3.preprocess_input(img)
    img = tf.convert_to_tensor(img)
    step_size = tf.convert_to_tensor(step_size)
    steps_remaining = steps
    step = 0
    while steps_remaining:
        if steps_remaining>100:
            run_steps = tf.constant(100)

```



```

else:
    run_steps = tf.constant(steps_remaining)
    steps_remaining -= run_steps
    step += run_steps

    loss, img = deepdream(img, run_steps, tf.constant(step_size))

    display.clear_output(wait=True)
    show(deprocess(img))
    print ("Step {}, loss {}".format(step, loss))

result = deprocess(img)
display.clear_output(wait=True)
show(result)

return result
dream_img = run_deep_dream_simple(img=original_img,
                                  steps=100, step_size=0.01)

```



```

import time
start = time.time()

OCTAVE_SCALE = 1.30

img = tf.constant(np.array(original_img))
base_shape = tf.shape(img)[::-1]
float_base_shape = tf.cast(base_shape, tf.float32)

for n in range(-2, 3):
    new_shape = tf.cast(float_base_shape*(OCTAVE_SCALE**n), tf.int32)

    img = tf.image.resize(img, new_shape).numpy()

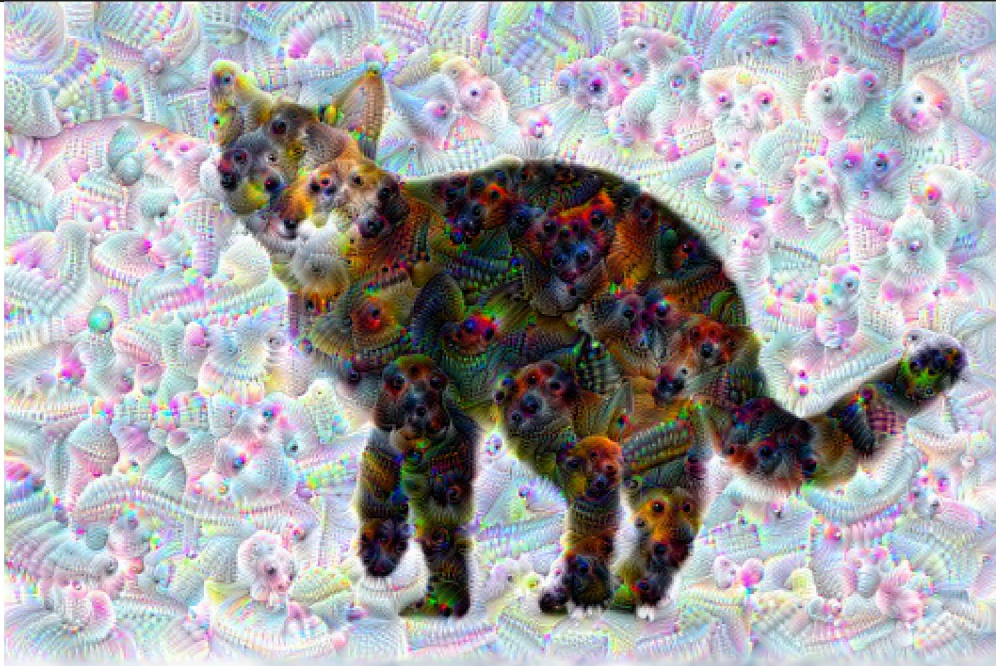
    img = run_deep_dream_simple(img=img, steps=50, step_size=0.01)

display.clear_output(wait=True)
img = tf.image.resize(img, base_shape)

```

```
img = tf.image.convert_image_dtype(img/255.0, dtype=tf.uint8)
show(img)

end = time.time()
end-start
```



Levando o modelo até uma oitava:

A saída é ruidosa e esse ruído pode ser resolvido com uma perda, a imagem também diminui a resolução e os padrões parecem estar acontecendo na mesma granularidade. O número de outras, a escala de oitavas e as camadas ativadas para alterar a aparência de uma imagem através do Deepdream.

## APÊNDICE 14 – VISUALIZAÇÃO DE DADOS E STORYTELLING

### A – ENUNCIADO

Escolha um conjunto de dados brutos (ou uma visualização de dados que você acredite que possa ser melhorada) e faça uma visualização desses dados (de acordo com os dados escolhidos e com a ferramenta de sua escolha)

Desenvolva uma narrativa/storytelling para essa visualização de dados considerando os conceitos e informações que foram discutidas nesta disciplina. Não esqueça de deixar claro para seu possível público alvo qual o **objetivo dessa visualização de dados, o que esses dados significam, quais possíveis ações podem ser feitas com base neles.**

**Entregue em um PDF:**

- O conjunto de dados brutos (ou uma visualização de dados que você acredite que possa ser **melhorada**);
- Explicação do **contexto e o público-alvo** da visualização de dados e do storytelling que será desenvolvido;
- A **visualização desses dados** (de acordo com os dados escolhidos e com a ferramenta de sua escolha) **explicando a escolha do tipo de visualização e da ferramenta usada; (50 pontos)**

### B – RESOLUÇÃO

**Conjunto de dados a ser usado:**

Base de dados pessoais de funcionários de uma empresa.

| ID R | Nome Completo                 | Estado C | Sexo | Idade | Tr | Data de Nascimento | Endereço                                                                       | Data de Contra |
|------|-------------------------------|----------|------|-------|----|--------------------|--------------------------------------------------------------------------------|----------------|
| 460  | Adelino Araújo                | C        | M    | 48    |    | 04/09/1976         | Shopping Barra - Av. Centenário, 2992 - Barra, Salvador - BA, 40155-150        | 11/7/2001      |
| 5    | Adelino Gomes                 | C        | M    | 50    |    | 30/03/1974         | R. Dr. Antônio Castro Prado, 422 - Vila Rossi e Borch, Campinas - SP, 13076-13 | 3/16/2001      |
| 195  | Adelino Werneck               | S        | M    | 27    |    | 27/04/1997         | Av. Borges de Medeiros, 3120 - Praia de Belas, Porto Alegre - RS, 90110-150    | 4/23/2001      |
| 102  | Adriano de Souza              | C        | M    | 31    |    | 09/12/1993         | R. Brasília Itiberê, 3279 - Água Verde, Curitiba - PR, 80250-160               | 12/5/2001      |
| 133  | Adrielle Sá Rodrigues         | S        | F    | 48    |    | 16/11/1976         | R. Dr. Antônio Castro Prado, 422 - Vila Rossi e Borch, Campinas - SP, 13076-13 | 4/25/2001      |
| 92   | Alberto Ruzza de Carvalho     | S        | M    | 44    |    | 06/06/1980         | R. Visc. de Pirajá, 136 - Ipanema, Rio de Janeiro - RJ, 22410-000              | 10/18/2001     |
| 167  | Alelio Villié Cerqueira Silva | C        | M    | 49    |    | 07/09/1975         | R. Visc. de Pirajá, 136 - Ipanema, Rio de Janeiro - RJ, 22410-000              | 6/9/2001       |

**Contexto e Público-alvo:**

O contexto para essa visualização dos dados será uma empresa que deseja mostrar os dados dos funcionários para os gerentes relevantes. Fica definido o público-alvo como sendo funcionários de nível gerencial que desejam ter insights sobre os funcionários aos quais ele lidera.



### Escolha da visualização e ferramenta utilizada

Para esse trabalho foi utilizada a visualização de gráficos como gráfico de pizza, histograma, gráfico de dispersão e gráfico de linha. Essa visualização foi escolhida pois mostra os dados brutos de forma clara para o público alvo, os gerentes. A escolha dos gráficos também foi tomada, pois o nível gerencial de uma empresa está normalmente acostumado com esse tipo de visualização e se sentiria confortável na análise de dados.

A ferramenta utilizada foi o Google Sheets. Os dados já estavam em formato de tabela e a forma mais natural de tratá-los seria no próprio ambiente em que eles estão. O Google Sheets é também uma ferramenta de fácil manuseio e entendimento, é gratuita e intuitiva.

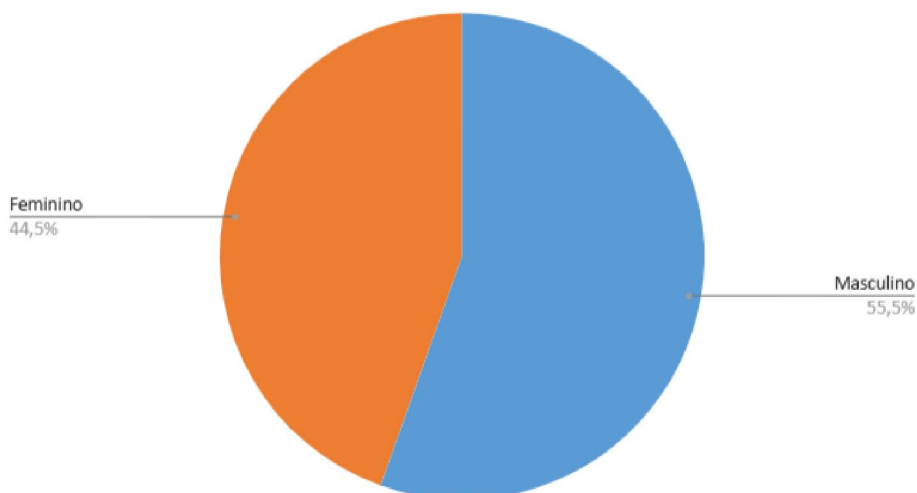
### Visualização dos dados e Storytelling:

O gerente de uma organização, tem uma função que vai muito além de gerenciar processos, é preciso entender a composição da equipe e o comportamento dos funcionários para garantir um ambiente de trabalho saudável e uma performance organizacional cada vez mais forte. Hoje, apresentamos alguns dados-chaves que ajudam a entender a estrutura da empresa e identificar pontos críticos para melhorar a gestão de talentos.

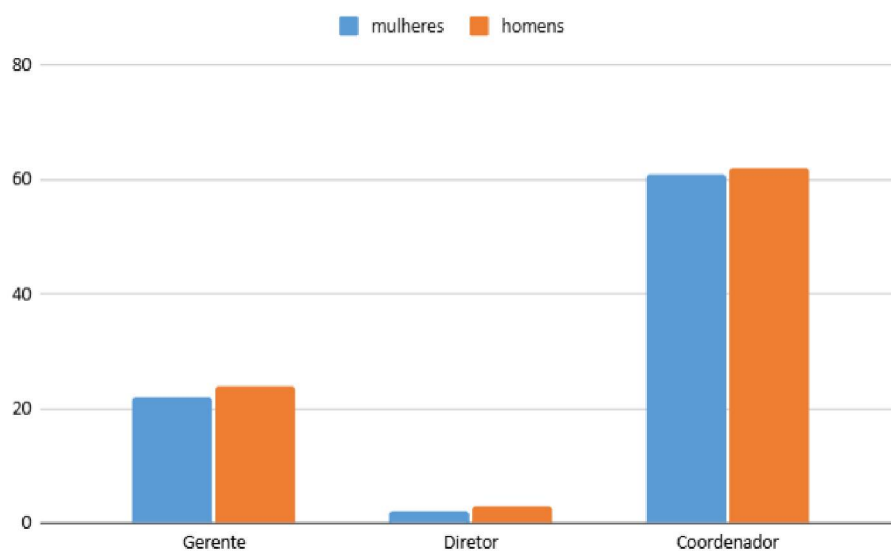
#### 1. Contagem de Gênero

A primeira análise feita foi a distribuição de gênero dentro da empresa. De acordo com os dados, temos uma proporção de 44,5% de mulheres e 55,5% de homens no quadro de funcionários. Embora a diferença não seja grande, podemos observar um leve desbalanceamento no gênero, mas que é amenizada levando em consideração os cargos de liderança.

Contagem de Gênero



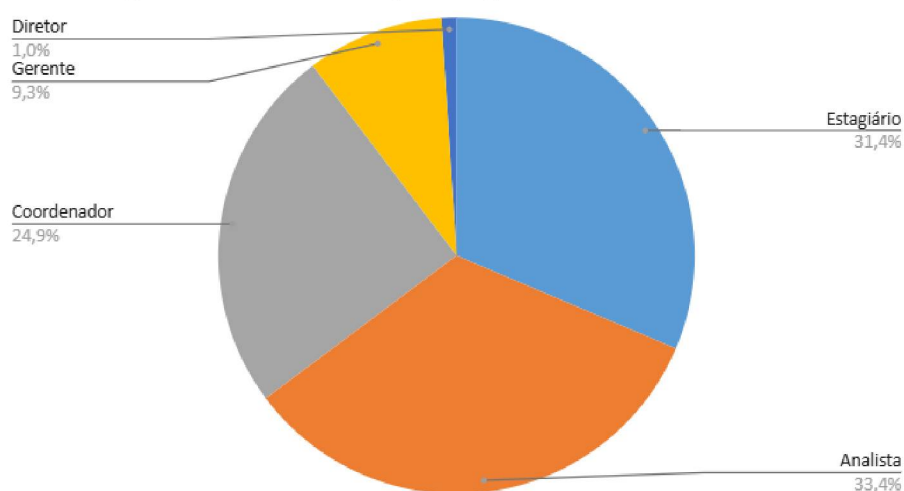
Podemos visualizar que homens e mulheres ocupam semelhantemente em quantidade as posições de Diretor, Coordenador e Gerente, com homens possuindo uma pequena vantagem em quantidade em relação às mulheres.



## 2. Porcentagem de Funcionários por Cargo

Seguindo adiante temos a porcentagem de funcionários divididos nos cargos da organização conforme abaixo.

Porcentagem de funcionários por cargo



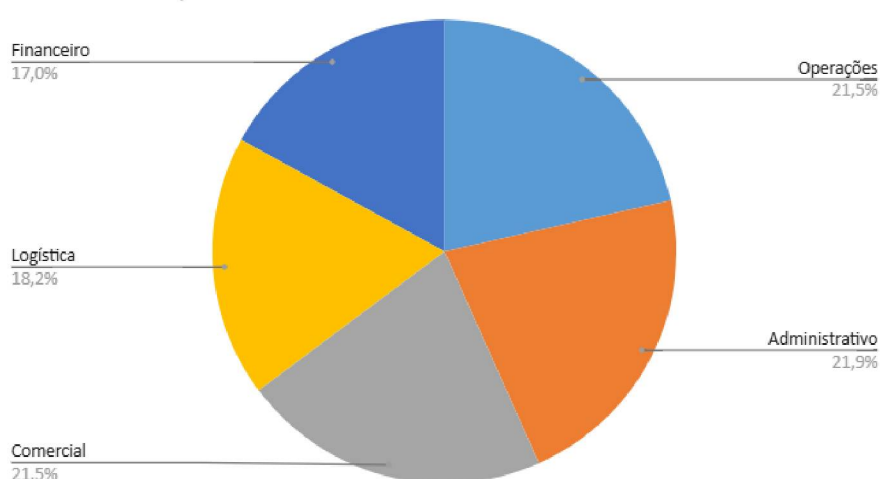
É possível visualizar que a maior força da organização está na área técnica representada pelos Analistas e Estagiários. Isso pode indicar que a organização tem o nível de especialização necessária para cumprir as metas organizacionais. A quantidade de coordenadores está em uma

proporção adequada para a área técnica. Pode ser revisado o crescimento interno, abrindo novas oportunidades para funcionários que queiram seguir carreira ascendente a cargos de gestão, principalmente para os cargos de Gerente e Diretores.

### 3. Porcentagem de Funcionários por Área

Analisando a distribuição por áreas, podemos observar uma divisão balanceada de funcionários entre as áreas. Pode-se pensar aqui nos objetivos da organização e analisar se existe a necessidade de crescimento de alguma área específica para que a empresa alcance suas metas. Como por exemplo incentivando e capacitando funcionários de áreas que não precisam de tanta mão de obra e transferindo-os para as áreas que necessitam de mais apoio.

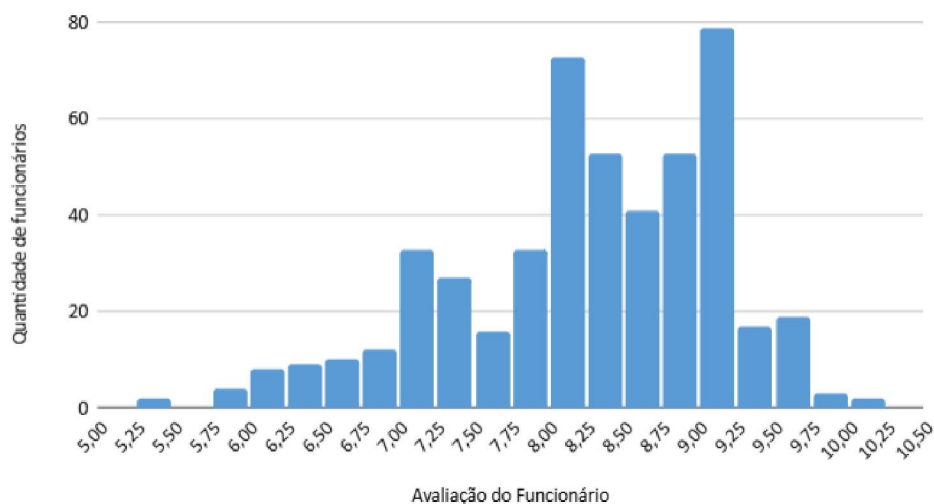
Funcionários por Área



### 4. Histograma de Avaliação dos Funcionários

Agora, olhando para o desempenho dos funcionários, analisa-se o histograma de avaliações de desempenho. Os dados indicam aproximadamente uma distribuição normal em que a maioria dos colaboradores encontram-se entre as notas 7,00 e 9,25. Apenas a minoria dos funcionários possuem uma avaliação maior que 9,5. Uma porcentagem também apresenta desempenho entre 6,00 e 7,00. Para que o desempenho seja cada vez melhor, pode-se incentivar treinamentos especializados e fazer um estudo sobre quais áreas podem estar afetando o desempenho dos colaboradores.

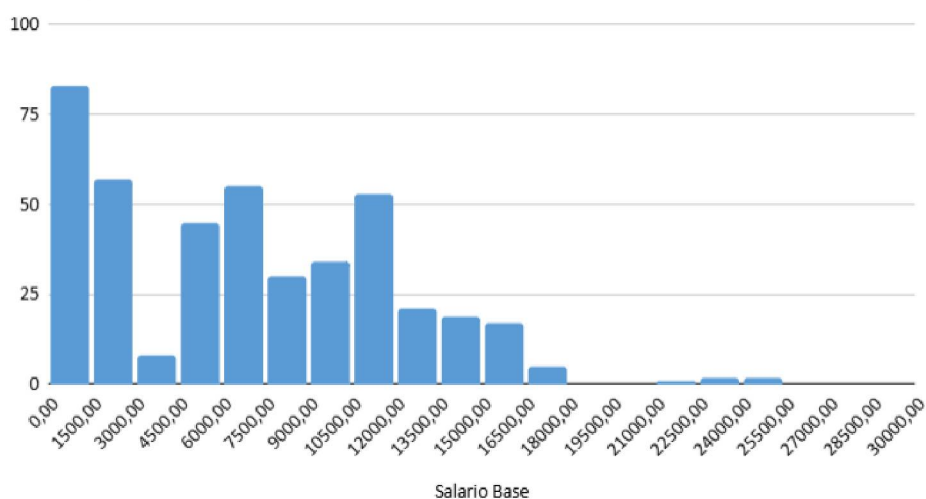
Histograma de Avaliação dos Funcionários



## 5. Histograma de Salário Base

O próximo ponto de análise é a faixa salarial. O histograma mostra que a maioria dos funcionários está concentrada nas faixas de R\$0,00 - R\$1500,00. Esses números corroboram a divisão de funcionários por cargo, pois entre a área técnica existe uma quantidade significativa de estagiários. Pode-se deduzir que a média de salários dos analistas esteja entre R\$3000,00 a R\$6000,00, pois também mostra uma significativa quantidade de funcionários nessa faixa de salário. É importante garantir que os salários e benefícios estejam bem ajustados para atrair e reter os melhores talentos.

Histograma de Salario Base

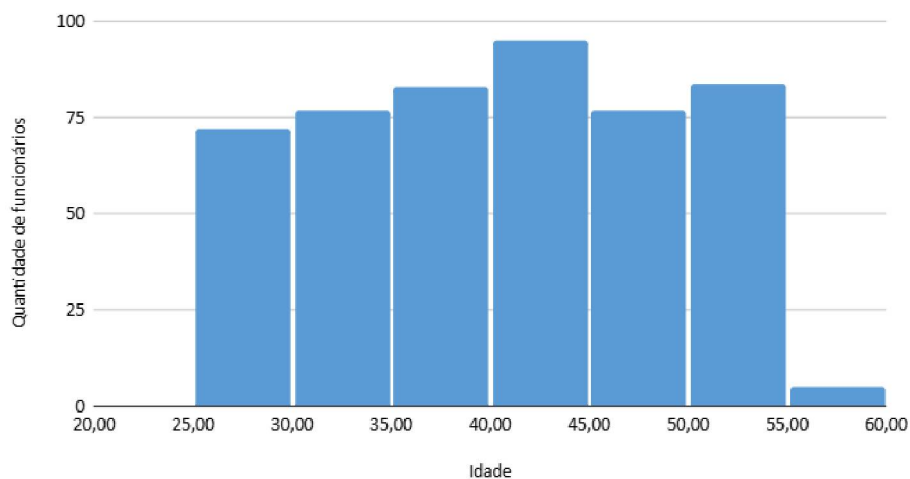


## 6. Histograma de Idade

A maioria dos colaboradores têm idade entre 30-45 anos, com uma quantidade menor de funcionários acima de 55 anos, e nenhum funcionário abaixo de 25 anos. Isso sugere que a

organização tem uma faixa etária elevada e que poderia investir mais em atrair funcionários jovens para trazer energia e inovação. A faixa etária mais elevada, no entanto, também sugere que a equipe é experiente e madura.

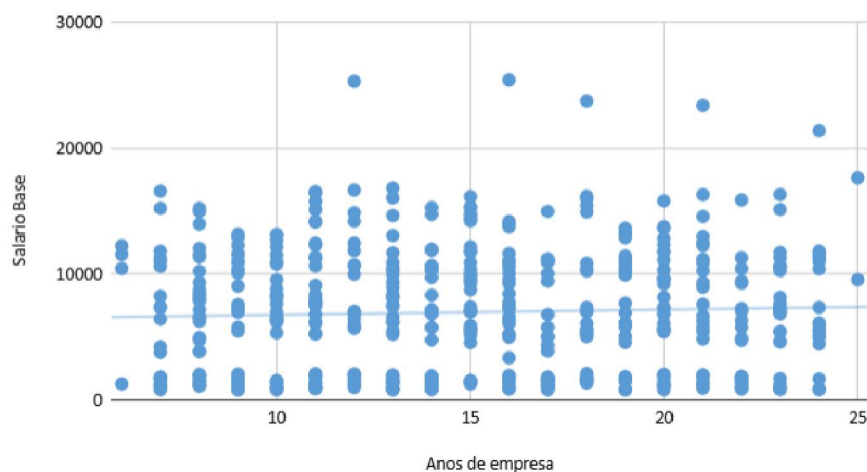
Histograma de Idade



## 7. Média salarial por tempo de empresa

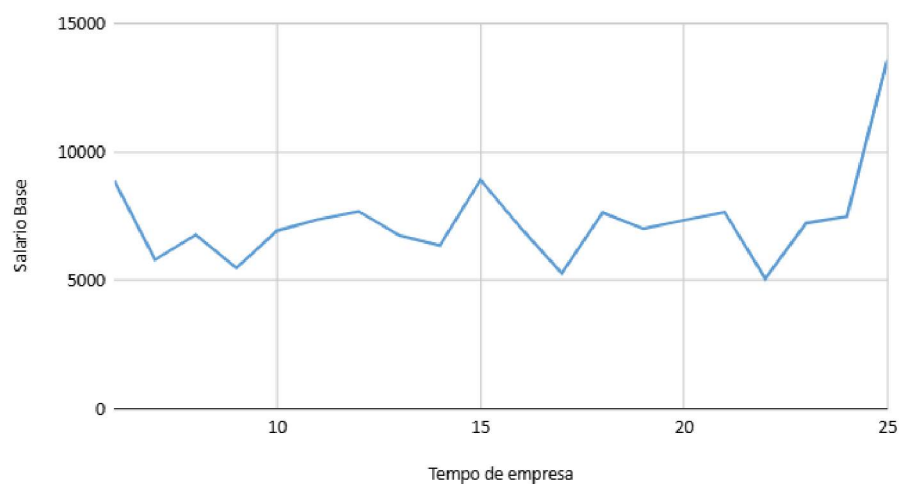
No gráfico de dispersão podemos observar que não há tendência crescente na média de salários conforme o tempo de empresa. Observa-se que a média de salários é praticamente constante e que o tempo de casa não é valorizado na organização. Alguns pontos dispersos indicam que o aumento de salário estaria mais ligado a promoções do que ao tempo de empresa.

Media salarial por tempo de empresa



O gráfico de linhas Tempo de Empresa x Salário base corrobora o gráfico de dispersão e mostra que não há uma forte correlação entre tempo de empresa e aumento de salário. Temos uma tendência ascendente apenas próximo aos 25 anos de tempo de empresa.

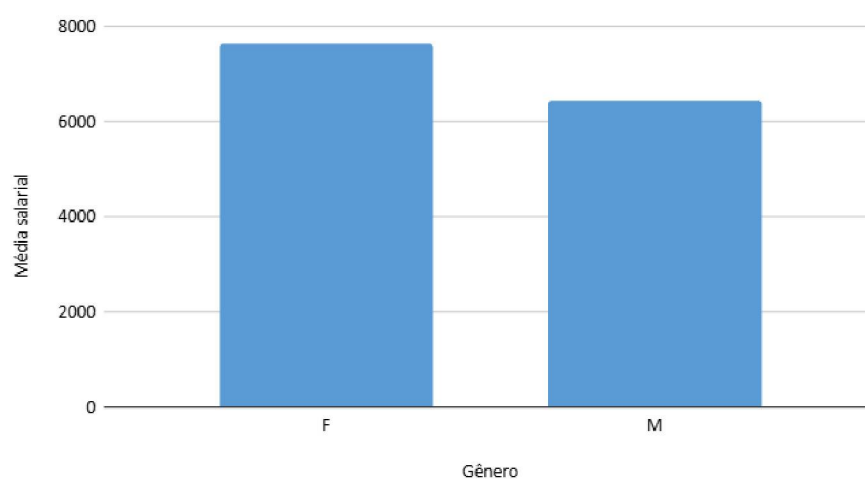
Tempo de Empresa x Salário Base



### 8. Média salarial entre homens e mulheres

Podemos concluir baseado no gráfico que a média salarial entre homens e mulheres é desproporcional, com mulheres ganhando em média 20% a menos que os homens. Levando em consideração que as mulheres ocupam a mesma proporção de cargos que os homens podemos ver que existe uma tendência discriminatória baseada no gênero e que essa correção deve ser tomada como prioridade, se tomarmos como base princípios de igualdade que são tão importantes para as organizações.

Média salarial de homens e mulheres



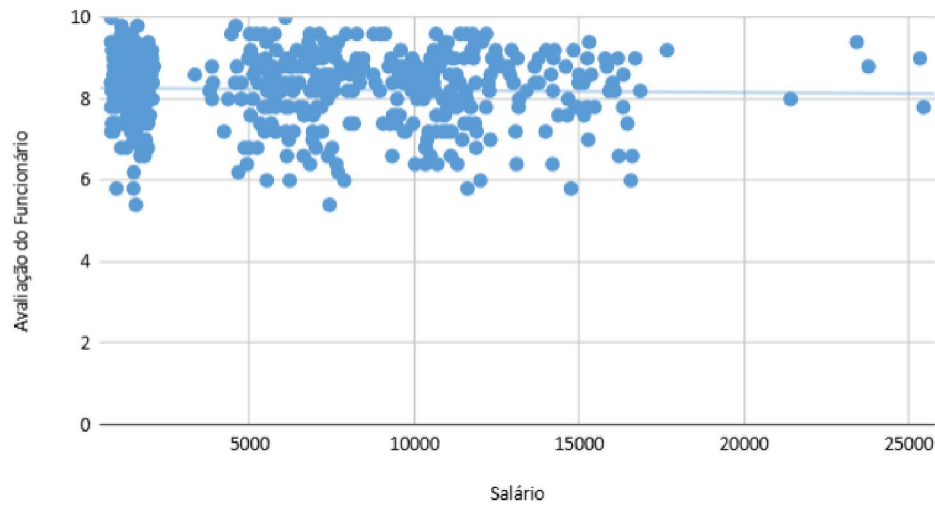
### 9. Base salarial x Desempenho

No gráfico de dispersão pode-se visualizar que não há uma tendência crescente em relação ao salário baseado no desempenho. Infelizmente isso pode indicar que o sistema de remuneração precisa ser ajustado para que seja justo e recompensador. Se houver um sistema de remuneração



baseado em performance os colaboradores podem se sentir mais propensos a cumprir as metas estabelecidas auxiliando a empresa a chegar nos seus objetivos.

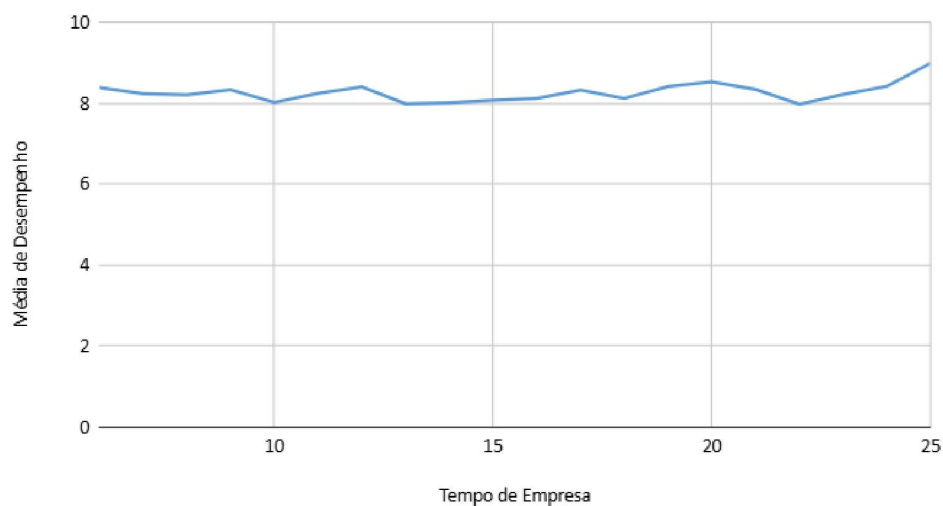
Salário x Desempenho



#### 10. Média de desempenho por tempo de empresa

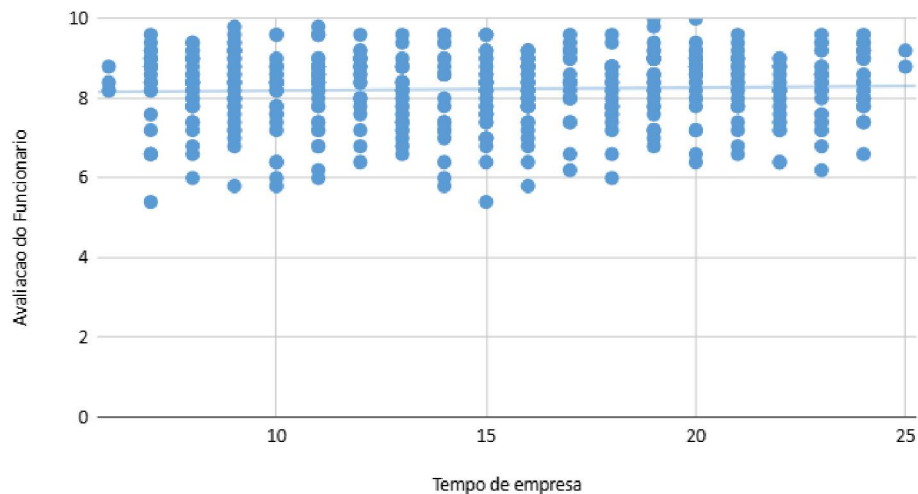
Para esse tópico, o gráfico mostra que não há uma forte relação entre tempo de empresa e desempenho, o que significa que profissionais mais recentes têm performance igual ou às vezes superior aos funcionários há mais tempo de casa. Podemos ver também que não há um pico de desempenho após alguns anos de empresa, pois o desempenho flutua acima e abaixo ao longo dos anos.

Média de Desempenho por Tempo de Empresa



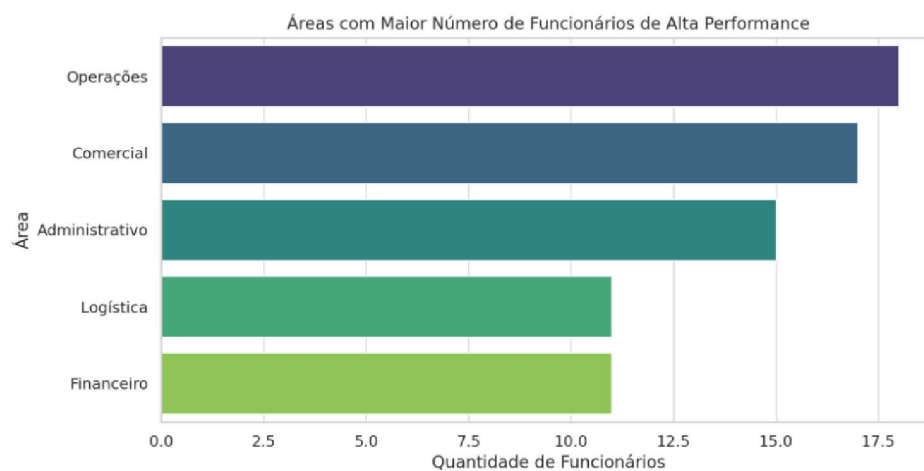
Os funcionários possuem no gráfico de dispersão abaixo avaliações similares independente do tempo de empresa, mostrando mais uma vez que colaboradores há menos tempo na função desempenham tão bem quanto funcionários mais experientes.

Tempo de empresa x Avaliacao do Funcionario



### 11. Áreas com maior número de funcionários de alta performance

Por último pode-se analisar quais áreas possuem a maior quantidade de funcionários de alta performance e procurar entender o que está sendo feito e trabalhando nessas áreas para que os funcionários desempenhem tão bem o seu papel.



Podemos identificar Operações e a área com a maior quantidade de funcionários de alta performance enquanto Logística e Financeiro ficam com as últimas posições. Um trabalho de pesquisa entre as áreas pode ajudar a identificar quais pontos estão sendo incentivados e quais pontos estão sendo negligenciados no crescimento dos colaboradores.

### Conclusão e Próximos Passos

Com base nesses dados, podemos concluir que a empresa tem uma estrutura robusta, mas há áreas que precisam de atenção para promover um ambiente de trabalho mais equilibrado e eficaz. Como por exemplo

- Igualdade salarial entre homens e mulheres: investir em iniciativas de igualdade salarial, para que não haja discriminação salarial entre gêneros.

- Estrutura de remuneração: Ajustar os fatores de remuneração com o tempo de serviço para que haja retenção de talentos.
- Atenção à diversidade etária: Buscar talentos jovens e inovativos no mercado para acrescentar mais diversidade no quadro de funcionários.

## APÊNDICE 15 – TÓPICOS EM INTELIGÊNCIA ARTIFICIAL

### A – ENUNCIADO

#### 1) Algoritmo Genético

Problema do Caixeiro Viajante

A Solução poderá ser apresentada em: Python (preferencialmente), ou em R, ou em Matlab, ou em C ou em Java.

Considere o seguinte problema de otimização (a escolha do número de 100 cidades foi feita simplesmente para tornar o problema intratável. A solução ótima para este problema não é conhecida).

Suponha que um caixeiro deva partir de sua cidade, visitar clientes em outras 99 cidades diferentes, e então retornar à sua cidade. Dadas as coordenadas das 100 cidades, descubra o percurso de menor distância que passe uma única vez por todas as cidades e retorne à cidade de origem.

Para tornar a coisa mais interessante, as coordenadas das cidades deverão ser sorteadas (aleatórias), considere que cada cidade possui um par de coordenadas (x e y) em um espaço limitado de 100 por 100 pixels.

O relatório deverá conter no mínimo a primeira melhor solução (obtida aleatoriamente na geração da população inicial) e a melhor solução obtida após um número mínimo de 1000 gerações. Gere as imagens em 2d dos pontos (cidades) e do caminho.

Sugestão:

- (1) considere o cromossomo formado pelas cidades, onde a cidade de início (escolhida aleatoriamente) deverá estar na posição 0 e 100 e a ordem das cidades visitadas nas posições de 1 a 99 deverão ser definidas pelo algoritmo genético.
- (2) A função de avaliação deverá minimizar a distância euclidiana entre as cidades (os pontos).
- (3) Utilize no mínimo uma população com 100 indivíduos;
- (4) Utilize no mínimo 1% de novos indivíduos obtidos pelo operador de mutação;
- (5) Utilize no mínimo de 90% de novos indivíduos obtidos pelo método de cruzamento (crossover-ox);
- (6) Preserve sempre a melhor solução de uma geração para outra.

**Importante:** A solução deverá implementar os operadores de “cruzamento” e “mutação”.

#### 2) Compare a representação de dois modelos vetoriais

Pegue um texto relativamente pequeno, o objetivo será visualizar a representação vetorial, que poderá ser um vetor por palavra ou por sentença. Seja qual for a situação, considere a quantidade de palavras ou sentenças onde tenha no mínimo duas similares e no mínimo 6 textos, que deverão produzir no mínimo 6 vetores. Também limite o número máximo, para que a visualização fique clara e objetiva.

O trabalho consiste em pegar os fragmentos de texto e codificá-las na forma vetorial. Após obter os vetores, imprima-os em figuras (plot) que demonstrem a projeção desses vetores usando a PCA.

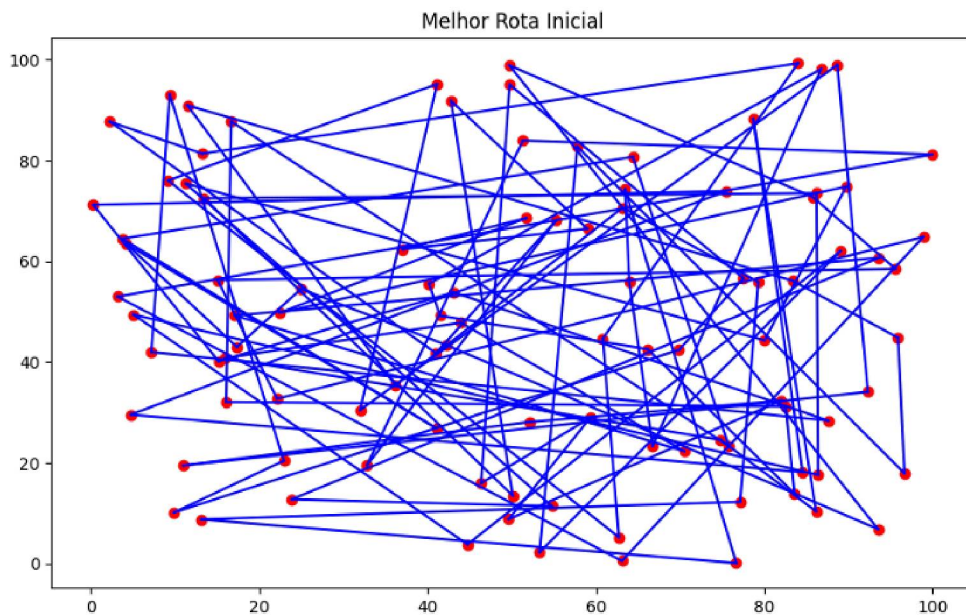
O PDF deverá conter o código-fonte e as imagens obtidas.

## B – RESOLUÇÃO

### 1. Algoritmo Genético

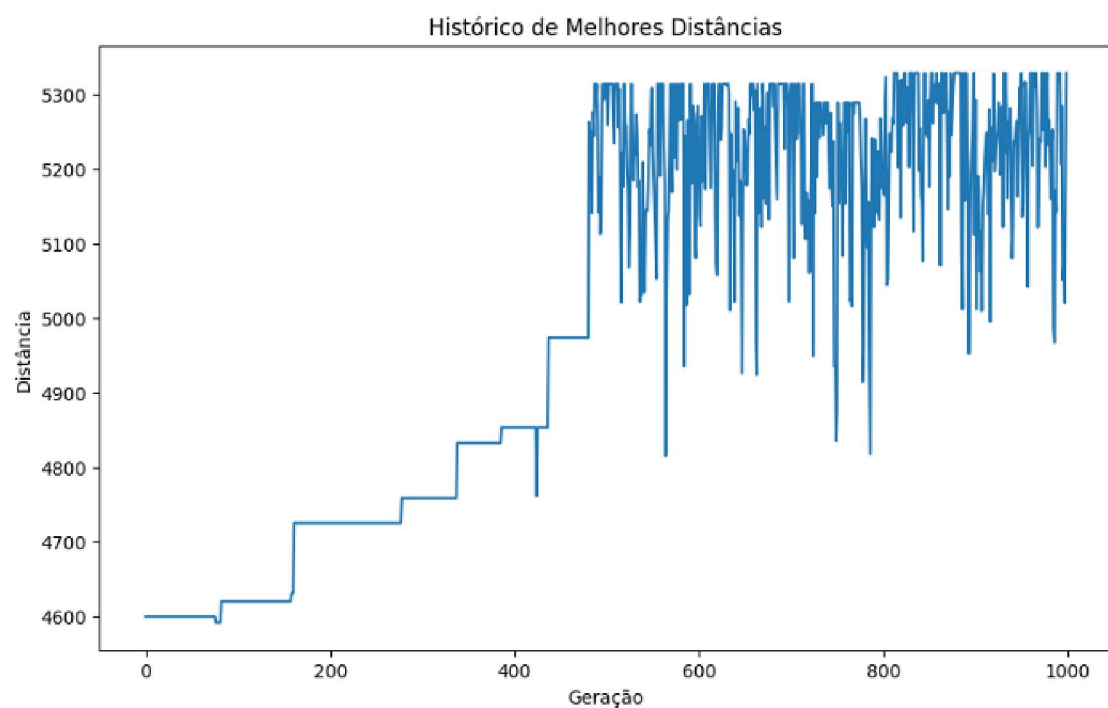
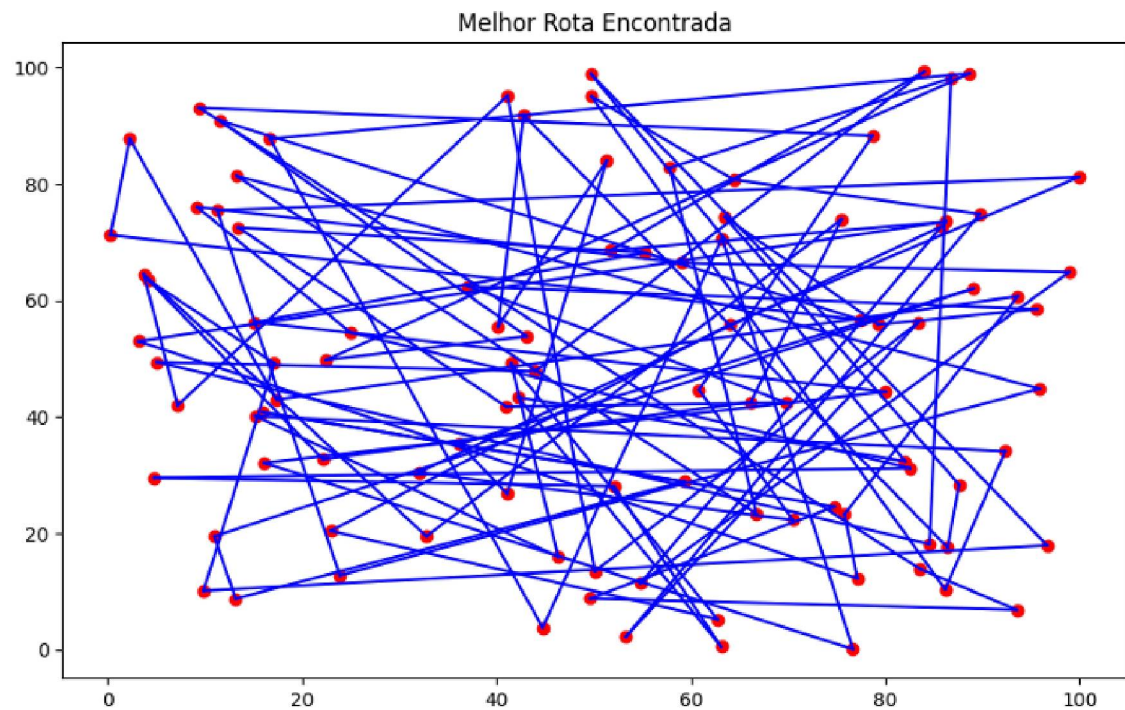
Melhor distância inicial: 5381.460998887509

Melhor rota inicial: [0, 54, 60, 15, 41, 81, 35, 12, 39, 11, 6, 36, 2, 93, 87, 74, 76, 73, 9, 37, 59, 18, 33, 71, 77, 38, 5, 75, 88, 20, 69, 48, 47, 95, 3, 4, 1, 68, 79, 80, 65, 62, 51, 78, 89, 14, 72, 92, 96, 45, 19, 7, 22, 23, 8, 53, 32, 57, 83, 63, 21, 43, 46, 28, 40, 10, 84, 17, 97, 42, 56, 13, 82, 24, 31, 85, 94, 16, 29, 91, 99, 49, 50, 86, 98, 25, 30, 66, 44, 55, 58, 70, 64, 90, 52, 67, 61, 34, 27, 26, 0]



Melhor distância encontrada: 4592.339863151977

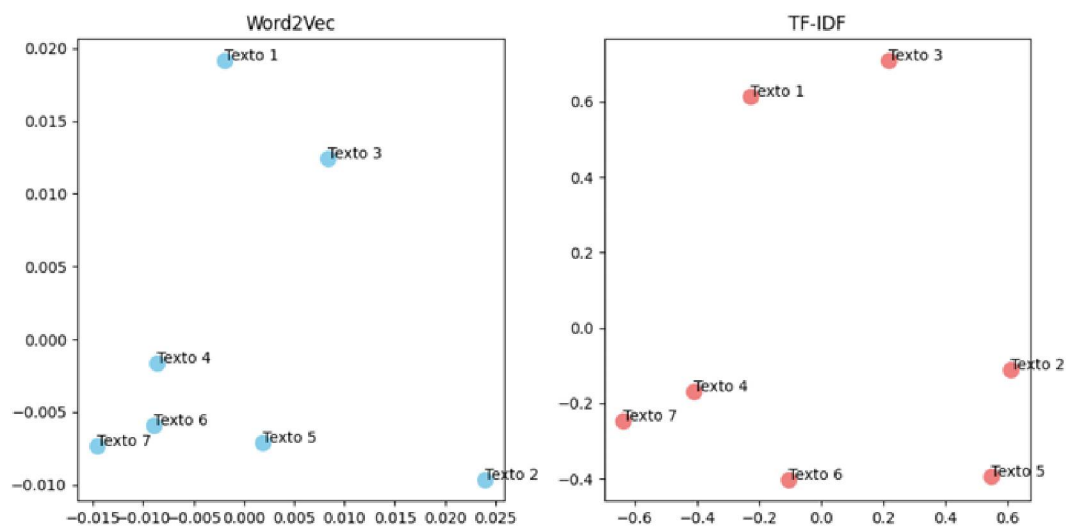
Melhor rota encontrada: [0, 45, 97, 61, 40, 70, 50, 4, 92, 31, 90, 52, 85, 5, 13, 74, 33, 51, 57, 47, 39, 2, 98, 88, 77, 19, 35, 91, 21, 6, 63, 7, 60, 68, 75, 71, 87, 79, 42, 41, 25, 18, 29, 66, 43, 89, 27, 80, 38, 53, 36, 81, 28, 67, 93, 58, 48, 73, 69, 46, 37, 26, 54, 32, 20, 49, 10, 95, 94, 30, 56, 64, 16, 65, 15, 76, 59, 11, 17, 86, 62, 83, 44, 3, 22, 78, 34, 99, 9, 96, 24, 72, 8, 82, 55, 1, 14, 84, 12, 23, 0]





## 2. Modelos Vetoriais

```
# Dados de texto
textos = [
    "O gato preto corre rápido.",
    "Um cachorro late alto.",
    "Um gato preto dorme no tapete.",
    "O cachorro corre atrás da bola.",
    "O pássaro voa quando o cachorro late. ",
    "As pessoas dormem quando está tarde.",
    "As pessoas correm rápido atrás da bola."
]
```



## TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação Inteligência Artificial Aplicada da Universidade Federal do Paraná foram convocados para realizar a arguição da Monografia de Especialização de **BARBARA SIMOES DE CARVALHO**, intitulada: **MEMORIAL DE PROJETOS: STORYTELLING EM INTELIGÊNCIA ARTIFICIAL - A NARRATIVA GUIADA POR DADOS**, que após terem inquirido a aluna e realizada a avaliação do trabalho, são de parecer pela sua aprovação no rito de defesa.

A outorga do título de especialista está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 08 de Agosto de 2025.



RAFAELA MANTOVANI FONTANA  
Presidente da Banca Examinadora



RAZER ANTHOM NIZER ROJAS MONTAÑO  
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)