

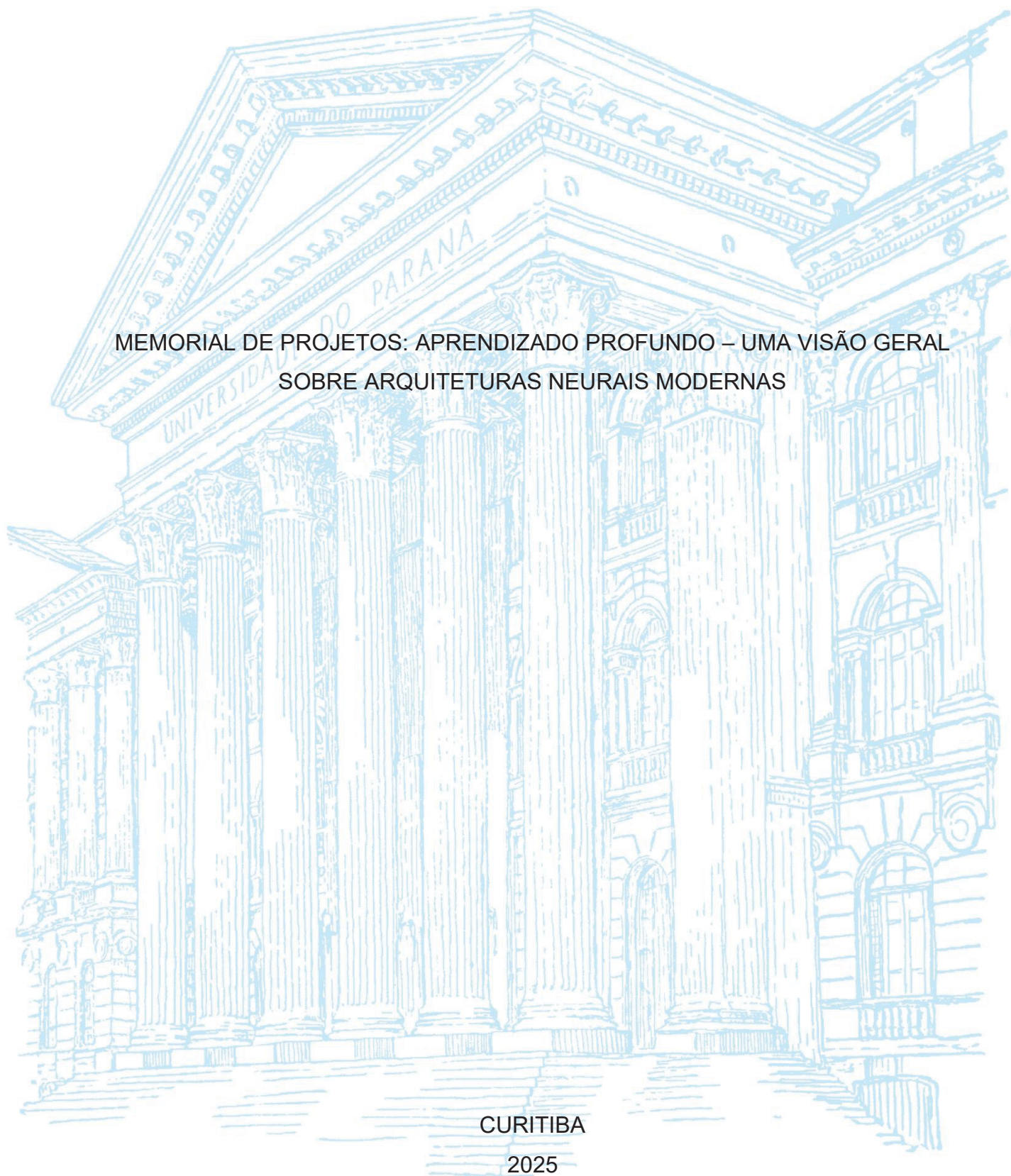
UNIVERSIDADE FEDERAL DO PARANÁ

LUÍS HENRIQUE PAIVA

MEMORIAL DE PROJETOS: APRENDIZADO PROFUNDO – UMA VISÃO GERAL
SOBRE ARQUITETURAS NEURAIIS MODERNAS

CURITIBA

2025



LUÍS HENRIQUE PAIVA

MEMORIAL DE PROJETOS: APRENDIZADO PROFUNDO – UMA VISÃO GERAL
SOBRE ARQUITETURAS NEURAIS MODERNAS

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 **LUÍS HENRIQUE PAIVA**, intitulada: **MEMORIAL DE PROJETOS: APRENDIZADO PROFUNDO UMA VISÃO GERAL SOBRE ARQUITETURAS NEURAIS MODERNAS**, que após terem inquirido o aluno 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

As redes neurais formam um dos pilares do Aprendizado Profundo. Esses algoritmos funcionam de forma similar ao cérebro humano e possibilitam que problemas reais possam ser resolvidos, como reconhecimento de imagens, processamento de linguagem natural, sistemas de recomendação, robótica, reconhecimento de padrões em texto e áudio, entre outros. Este trabalho está dividido em duas partes. Na primeira, o parecer técnico, é apresentada uma visão geral sobre as arquiteturas RNN (Rede Neural Recorrente), CNN (Rede Neural Convolucional), GAN (Rede Adversária Generativa) e Transformer, passando pela sua definição, principais características e principais caso de aplicação. O parecer conclui retomando os principais conceitos discutidos nele e finaliza indicando a importância e o impacto dessas estruturas na melhoria de processos e atividades humanas, além dos novos desafios que se apresentam quando essas arquiteturas são expostas a novos tipos de problemas. Na segunda, e última parte, são apresentados os apêndices com as principais resoluções desenvolvidas durante cada disciplina do curso, além dos resultados aos questionamentos propostos.

Palavras-chave: Aprendizado Profundo; Redes Neurais Recorrentes; Redes Neurais Convolucionais; Redes Adversariais Generativas; Transformers.

ABSTRACT

Neural networks are one of the pillars of Deep Learning. These algorithms work in a similar way to the human brain and enable real-world problems to be solved, such as image recognition, natural language processing, recommendation systems, robotics, pattern recognition in text and audio, among others. This paper is divided into two parts. The first, the technical review, presents an overview of the RNN (Recurrent Neural Network), CNN (Convolutional Neural Network), GAN (Generative Adversarial Network), and Transformer architectures, covering their definitions, main characteristics, and key application cases. The review concludes by reviewing the main concepts contained therein and concluding by highlighting the importance and impact of these structures in improving human processes and activities, as well as the new challenges that arise when these architectures are exposed to new types of problems. In the second and final part, appendices are presented with the main resolutions developed during each course discipline, in addition to the results of the proposed questions.

Keywords: Deep Learning; Recurrent Neural Networks; Convolutional Neural Networks; Generative Adversarial Networks; Transformers.

SUMÁRIO

1 PARECER TÉCNICO.....	6
REFERÊNCIAS.....	11
APÊNDICE 1 – INTRODUÇÃO À INTELIGÊNCIA ARTIFICIAL	13
APÊNDICE 2 – LINGUAGEM DE PROGRAMAÇÃO APLICADA	19
APÊNDICE 3 – LINGUAGEM R	26
APÊNDICE 4 – ESTATÍSTICA APLICADA I.....	30
APÊNDICE 5 – ESTATÍSTICA APLICADA II.....	35
APÊNDICE 6 – ARQUITETURA DE DADOS	42
APÊNDICE 7 – APRENDIZADO DE MÁQUINA.....	49
APÊNDICE 8 – DEEP LEARNING	55
APÊNDICE 9 – BIG DATA.....	60
APÊNDICE 10 – VISÃO COMPUTACIONAL	62
APÊNDICE 11 – ASPECTOS FILOSÓFICOS E ÉTICOS DA IA.....	66
APÊNDICE 12 – GESTÃO DE PROJETOS DE IA.....	69
APÊNDICE 13 – FRAMEWORKS DE INTELIGÊNCIA ARTIFICIAL	72
APÊNDICE 14 – VISUALIZAÇÃO DE DADOS E STORYTELLING	80
APÊNDICE 15 – TÓPICOS EM INTELIGÊNCIA ARTIFICIAL	83

1 PARECER TÉCNICO

Este parecer técnico tem como objetivo apresentar uma visão geral sobre as principais arquiteturas em Aprendizado Profundo. Na Seção 1.1, é feita uma introdução do assunto. Seções 1.2, 1.3, 1.4 e 1.5 apresentam as principais arquiteturas em Aprendizado Profundo. Por fim, na Seção 1.6, é feita a conclusão do trabalho, destacando a importância e impacto dessas tecnologias.

1.1 INTRODUÇÃO

Atualmente, muito se tem discutido sobre os avanços da Inteligência Artificial (IA) nas atividades humanas, especialmente no trabalho. Entender a IA e conhecer suas possibilidades se torna fator crucial, porém desafiador para os profissionais das próximas gerações. Nessa linha, é relevante os estudos de redes neurais artificiais, linguagens de programação direcionada, estatística, ética e gestão de projetos direcionados à área de IA, como as oferecidas no presente curso.

Conforme Russell e Norvig (2022), os primeiros estudos da IA são marcados pelos trabalhos de Warren McCulloch e Walter Pitts em 1943, com base em conhecimento de fisiologia e função dos neurônios no cérebro, lógica proposicional e a teoria da computação de Allan Turing. Um ano depois, Marvin Minsky e Dean Edmonds construíram o primeiro computador de rede neural, o SNARC - *Stochastic Neural Analog Reinforcement Calculator* (Russell e Norvig, 2022).

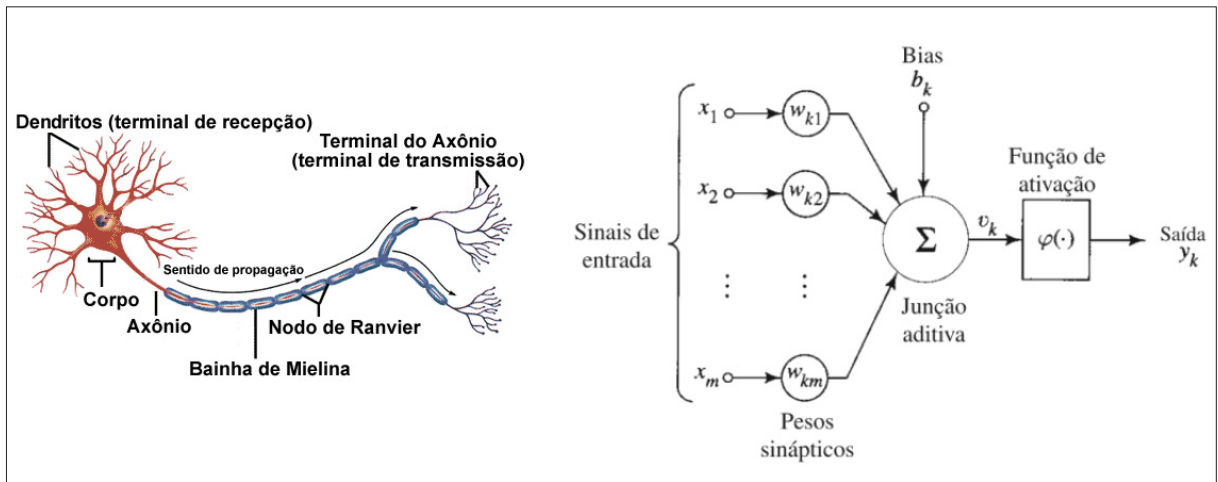
O termo inteligência artificial foi cunhado primeiramente em 1956, aproximadamente seis anos após *Computing machinery and intelligence*, artigo de Turing, quando John McCarthy, Marvin Minsky, Claude Shannon e Nathaniel Rochester demonstraram interesse na teoria dos autômatos, redes neurais e ciência cognitiva, durante o seminário de dois meses em Dartmouth (Russell e Norvig, 2022; Toosi *et al.*, 2022). Nesse evento McCarthy definiu IA como “a ciência e engenharia de fazer máquinas inteligentes”, enfatizando o crescimento simultâneo dos computadores e da IA (Russell e Norvig, 2022; Toosi *et al.*, 2022).

A IA pode ser dividida em diversas áreas, como, por exemplo, Robótica, Processamento de Linguagem Natural (PLN), Visão Computacional e Aprendizado de Máquina (*Machine Learning - ML*). Nesta, encontram-se, além de outros, os estudos das redes neurais, que tem como principal inspiração o funcionamento do cérebro

humano, as quais processam os dados sensoriais dentro da rede por meio de seus neurônios (Haykin, 2001).

Um neurônio biológico é uma célula nervosa, altamente complexa, com a capacidade de transformar estímulos em outros. Um neurônio artificial, é uma simplificação matemática do modelo biológico e, embora também processe entradas, se diferencia por ter uma única saída. Essa comparação é ilustrada pela FIGURA 1.

FIGURA 1 – COMPARAÇÃO NEURÔNIO BIOLÓGICO E NEURÔNIO ARTIFICIAL



FONTE: Adaptado de Haykin (2001).

O Aprendizado Profundo (*Deep Learning* - DL), ocupa-se do estudo de um tipo especial de redes e pode ser definido como uma técnica de computação para extrair e processar dados por meio de redes neurais de múltiplas camadas (Howard e Gugger, 2020). Essas redes se caracterizam por utilizar o resultado da camada anterior na próxima, criando, assim, um tipo de refinamento do modelo (Howard e Gugger, 2020). Algumas das arquiteturas em DL são: Redes Neurais Recorrentes, Redes Neurais Convolucionais, Redes Adversariais Generativas e Transformers.

1.2 REDES NEURAIS RECORRENTES

As Redes Neurais Recorrentes, ou *Recurrent Neural Network* (RNN), utilizam-se de redes neurais com múltiplas camadas projetadas para processamento de dados sequenciais, como texto, áudio, séries temporais, entre outros. Diferentemente das redes tradicionais, que assumem entradas independentes, as RNN possuem conexões recorrentes que permitem que as informações (dados processados) das

etapas anteriores sejam utilizadas como critério de avaliação das etapas atuais, criando uma espécie de memória temporal (Mienye, Swart e Obaido, 2006).

Entre as principais características das RNN, pode-se destacar a memória de curto longo prazo (LSTM), processamento sequencial e os ciclos (*loops*) internos (Subasi, 2020; Nisha, Sathik e Meeral, 2021). As RNN são aplicadas em Processamento de Linguagem Natural (NLP), reconhecimento de fala, geração de texto, análise de séries temporais, entre outros (Mienye, Swart e Obaido, 2006).

1.3 REDES NEURAIAS CONVOLUCIONAIS

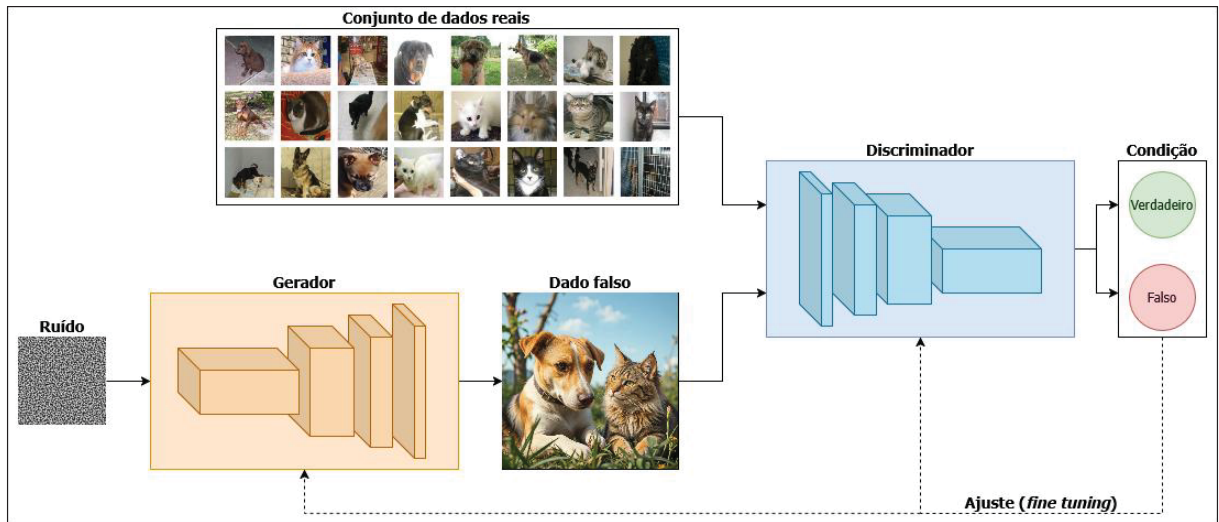
As Redes Neurais Convulacionais, ou *Convolutional Neural Network* (CNN), são um tipo de arquitetura utilizada especialmente em tarefas relacionadas a dados com estrutura espacial, com detecção de objetos (Ersavas, Smith e Mattick, 2024). Possuem uma arquitetura baseada no córtex visual dos animais e se diferenciam das redes neurais tradicionais por possuírem neurônios que se auto otimizam por meio do processo de aprendizado (O'Shea e Nash, 2015; Subasi, 2020).

As CNN possuem como principais características camadas convolucionais, responsáveis por extrair padrões espaciais como entradas, formas, bordas e texturas; camadas de *pooling*, que reduzem a dimensionalidade dos dados e as camadas totalmente conectadas, que são responsáveis pelas tarefas de regressão ou classificação (O'Shea e Nash, 2015; Ersavas, Smith e Mattick, 2024).

1.4 REDES ADVERSARIAIS GENERATIVAS

Redes Adversariais Generativas, ou *Generative Adversarial Networks* (GAN), foram introduzidas em 2014 por Goodfellow, Bengio e Courville (2016) e podem ser descritas, basicamente, como um sistema composto por duas redes neurais que competem entre si: o gerador e o discriminador (Cohen e Giryas, 2022; Saxena e Cao, 2020). O gerador procura criar dados sintéticos que sejam indistinguíveis de dados reais, tentando enganar o discriminador, que avalia se os dados recebidos são verdadeiros ou falsos (Cohen e Giryas, 2022; Saxena e Cao, 2020). Esse processo, de acordo com Gomes e Bruno (2023), faz com que ambas evoluam de forma adversária, competindo uma com a outra. A FIGURA 2 oferece uma visão geral do funcionamento de uma GAN.

FIGURA 2 – VISÃO GERAL DO FUNCIONAMENTO DE UMA GAN



FONTE: Adaptado de Sharma *et al.*, 2024.

As GAN são utilizadas em variadas aplicações, como geração e aprimoramento de imagens e vídeos. Também podem ser utilizadas na geração de dados sintéticos para minimizar ou sanar problemas quando há poucos dados disponíveis, conforme lembram Douzas e Bação (2017) e Brophy *et al.* (2025). Como desafios, citam-se as dificuldades no treinamento desse modelo e a possibilidade de gerar dados enviesados ou imprecisos (Gomes e Bruno, 2023).

1.5 TRANSFORMERS

Os Transformers foram introduzidos no artigo *Attention is All you Need*, por Vaswani *et al.* (2017). Essa arquitetura funciona com base em um mecanismo de autoatenção multicabeça, o qual permite ponderar as diferentes partes de uma entrada (Jurafsky e Martin, 2025; Khan *et al.*, 2022). Tem como característica o processamento de entradas em modo paralelo, o que o torna adequado para processamento de grandes volumes de dados (Khan *et al.*, 2022; Phuong e Hutter, 2022). Os Transformers têm aplicação nas mais diversas áreas, como processamento de linguagem natural, visão computacional e aplicações multimodais, como texto para imagem, por exemplo (Lin *et al.*, 2022). São modelos Transformer o EDT, o BERT e o GPT (Phuong e Hutter, 2022).

1.6 CONCLUSÃO

O estudo de Aprendizado Profundo envolve discussões acerca de arquitetura neurais, como RNN, CNN, GAN e Transformers. Elas são importantes, na medida em que possibilitam abordagens inovadoras na resolução de problemas, desde classificação de texto até análise de sentimentos, por exemplo. Nesse sentido, os estudos, principalmente, como apresentados na disciplina de *Deep Learning* do presente curso, demonstram como cada uma das arquiteturas podem ser usadas, de acordo com o problema a ser resolvido ou a atividade a ser executada.

Vale ressaltar que essas abordagens vêm sendo utilizadas nas mais diversas áreas do conhecimento, e propiciaram um aumento na efetividade de diversas tarefas, seja nas atividades profissionais ou acadêmicas. Desse modo, fica clara a importância e o impacto significativo dessas arquiteturas nas mais variadas áreas da tecnologia e da ciência. Espera-se, assim, que elas continuem a ajudar a resolver os desafios atuais e os próximos que se apresentarem.

REFERÊNCIAS

BROPHY, E. *et al.* Generative Adversarial Networks in Time Series: A Systematic Literature Review. **ACM Computing Surveys**, v. 55, p. 1-31. 2022. Disponível em: <https://doi.org/10.1145/3559540>. Acesso em: 10 ago. 2025.

COHEN, Gilad; GIRYES, Raja. Generative adversarial networks. In: **Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook**. Cham: Springer International Publishing, 2023. p. 375-400. Disponível em: <https://arxiv.org/abs/2203.00667>. Acesso em: 9 ago. 2025.

DOUZAS, G.; BAÇÃO, F. Effective data generation for imbalanced learning using conditional generative adversarial networks. **Expert Systems with Applications**, v. 91, p. 464-471. 2018. Disponível em: <https://doi.org/10.1016/J.ESWA.2017.09.030>. Acesso em: 10 ago. 2025.

ERSAVAS, T.; SMITH, M. A.; MATTICK, J. S. Novel Applications of Convolutional Neural Networks in the age of Transformers. **Scientific Reports**, Londres, v. 14, n. 10.000, p. 1-11, 2024. Disponível em: <https://doi.org/10.1038/s41598-024-60709-z>. Acesso em: 23 maio 2025.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. Cambridge: MIT Press. 2016.

GOMES, J. C.; BRUNO, D. R. GANs – REDES ADVERSÁRIAS GENERATIVAS: definições e aplicações. **Revista Interface Tecnológica**, Taquaritinga, SP, v. 20, n. 2, p. 182–194, 2023. DOI: 10.31510/infa.v20i2.1800. Disponível em: <https://revista.fatectq.edu.br/interfacetecnologica/article/view/1800>. Acesso em: 17 maio. 2025.

HAYKIN, Simon. **Redes Neurais: Princípios e prática**. São Paulo: Bookman, 2001. 906 p.

HOWARD, Jeremy; GUGGER, Sylvian. **Deep Learning for Coders with fastai and PyTorch: AI Applications Without a PhD**. Sebastopol: O'Reilly Media. 2020. 621 p.

JURAFSKY, Dan; MARTIN, James H. **Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models**. Colorado, 2025. 599 p. Disponível em: <https://web.stanford.edu/~jurafsky/slp3/>. Acesso em: 25 maio 2025.

KHAN, Salman *et al.* Transformers in vision: A survey. **ACM computing surveys (CSUR)**, v. 54, n. 10s, p. 1-41, 2022. Disponível em: <https://arxiv.org/abs/2101.01169v5>. Acesso em: 10 ago. 2025.

LIN, Tianyang *et al.* A survey of transformers. **AI Open**, v. 3, p. 111-132. Disponível em: <https://doi.org/10.1016/j.aiopen.2022.10.001>. Acesso em: 10 ago. 2025.

MIENYE I.D.; SWART, T.G.; OBAIDO G. Recurrent Neural Networks: A Comprehensive Review of Architectures, Variants, and Applications. **Information**, Basel, v. 15, p. 1-34, 2024. Disponível em: <https://doi.org/10.3390/info15090517>. Acesso em: 15 maio 2025.

NISHA, S. S.; SATHIK, M. M.; MEERAL, N. M. Application, algorithm, tools directly related to deep learning. In: BALAS, V. E.; MISHRA, B. K.; KUMAR, R. **Handbook of Deep Learning in Biomedical Engineering**. Tirunelveli: Elsevier. p. 61-84. 2020. Disponível em: <https://doi.org/10.1016/B978-0-12-823014-5.00007-7>. Acesso em: 22 maio 2025.

O'SHEA, K.; NASH, R. An introduction to Convolutional Neural Networks. **arXiv**. Ithaca. 2015. Disponível em: <https://arxiv.org/abs/1511.08458>. Acesso em: 16 maio 2025.

PHUONG, Mary; HUTTER, Marcus. Formal algorithms for transformers. **arXiv preprint arXiv:2207.09238**, 2022. Disponível em: <https://arxiv.org/abs/2207.09238v1>. Acesso em: 10 ago. 2025.

RUSSELL, Stuart J.; NORVIG, Peter. **Inteligência Artificial: Uma Abordagem Moderna**. 4. ed. Rio de Janeiro: Grupo Editora Nacional, LTC, 2022. 1136 p.

SAXENA, Divya; CAO, Jiannong. Generative Adversarial Networks (GANs). **ACM Computing Surveys (CSUR)**, v. 54, p. 1-42. Disponível em: <https://doi.org/10.1145/3446374>. Acesso em: 9 ago. 2025.

SHARMA, P.; KUMAR, M.; SHARMA, H. K.; MATHEW, S. Generative adversarial networks (GANs): Introduction, Taxonomy, Variants, Limitations, and Applications. **Multimed Tools Appl**, Nova Iorque, v. 83, p. 88811–88858. 2024. Disponível em: <https://doi.org/10.1007/>. Acesso em: 30 maio 2025.

SUBASI, Abdulhamit. **Practical Machine Learning for Data Analysis Using Python**. Gidá: Elsevier, 2020. 520 p.

TOOSI, Amirhosein *et al.* A brief history of AI: how to prevent another winter (a critical review). **PET clinics**, v. 16, n. 4, p. 449-469, 2021. Disponível em: <https://arxiv.org/abs/2109.01517v3>. Acesso em: 10 ago. 2025.

VASWANI, A. et al. Attention is all you need. In: 31ST CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS, 2017. **Proceedings**. Long Beach: NIPS, v.30, p. 1-15, June 2017. Disponível em: <https://doi.org/10.48550/arXiv.1706.03762>. Acesso em: 15 jul. 2025.

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.

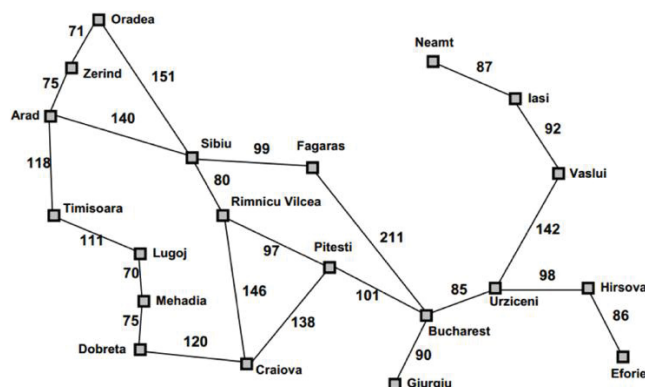
2 BUSCA HEURÍSTICA

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: $\alpha \rightarrow \beta, \beta \rightarrow \gamma \vdash \alpha \rightarrow \gamma$

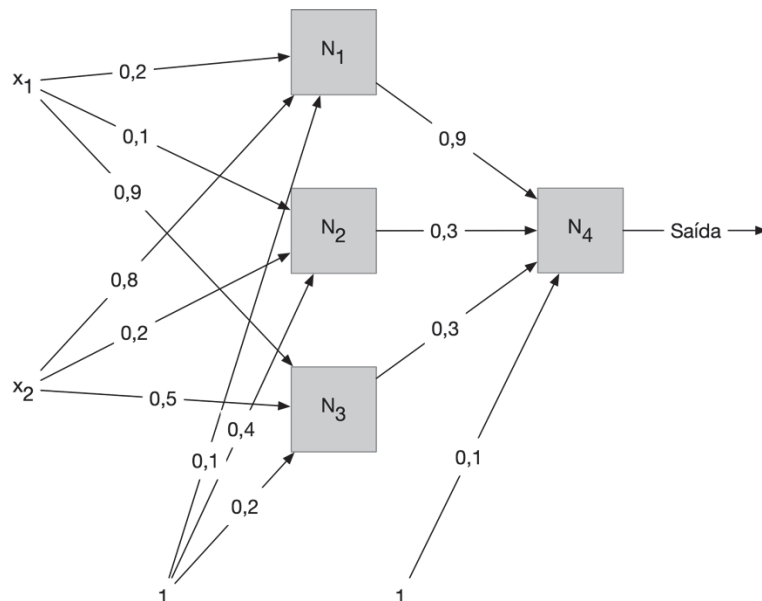
Conjunção: $\alpha, \beta \vdash \alpha \wedge \beta$

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 N1, N2 e N3 possuem função de ativação linear. Já N4 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 N1
- (6,25 pontos)** Valor de saída do neurônio N2
- (6,25 pontos)** Valor de saída do neurônio N3
- (6,25 pontos)** Valor de saída da rede como um todo

B – RESOLUÇÃO

1 CHATGPT

- a) **Prompt:** Olá ChatGPT. O que é inteligência artificial?

Resposta: Olá! Inteligência artificial (IA) é um campo da ciência da computação que se concentra no desenvolvimento de sistemas e algoritmos capazes de realizar tarefas que normalmente exigiriam inteligência humana. Isso inclui uma variedade de habilidades, como reconhecimento de padrões, tomada de decisões, resolução de problemas, compreensão de linguagem natural e aprendizado. O objetivo da inteligência artificial é criar sistemas que possam perceber o ambiente ao seu redor, aprender com essas percepções e tomar decisões autônomas para alcançar objetivos específicos. Esses sistemas são frequentemente baseados em algoritmos de aprendizado de máquina e redes neurais, que podem analisar grandes conjuntos de dados, identificar padrões e fazer previsões. A inteligência artificial tem aplicações em uma ampla variedade de campos, incluindo medicina, finanças, transporte, manufatura, jogos, assistência virtual e muito mais.

- b) Inteligência Artificial (IA) como descrita pelo ChatGPT se encaixa no modelo “Agir Racionalmente”, a abordagem mais abrangente, visto que, como mencionado pelo Chat, a IA tem como objetivo ter consciência do seu ambiente, aprender com suas experiências e buscar tomar sempre a melhor decisão. Isso excluiria as abordagens “Agir como humano” e “Pensar como humano”, visto que o objetivo da IA descrita pelo chat não é imitar o comportamento humano nem replicar fielmente o modo como os humanos pensam, mas tomar decisões autônomas para atingir seus objetivos. Pela descrição fornecida, também é possível perceber que os objetivos de IA são mais abrangentes que a abordagem “Pensar racionalmente”, visto que essa linha tem como objetivo modelar um processo de raciocínio perfeito, mas nem sempre adaptável.

- c) O ChatGPT gera resposta baseada no contexto e na intenção por trás da questão do usuário. Seu poder está na habilidade de analisar perguntas e produzir respostas completas, detalhadas, e resultados baseados nas informações em texto mais frequentes acessíveis digitalmente do mundo à época de seu treinamento. Em termos gerais, o ChatGPT funciona em duas fases principais. A fase de coleta de dados é chamada de pré-treinamento, enquanto a fase de responsividade do usuário é chamada de inferência. A “magia” por trás da IA generativa e o motivo pelo qual ela explodiu de repente é que a maneira como o pré-treinamento funciona e ele ser enormemente escalável. Essa escalabilidade foi possibilitada por inovações recentes em tecnologia de hardware acessíveis e computação em nuvem.

Os modelos de IA são treinados usando dois principais enfoques: supervisionado e não supervisionado. Para a maioria dos projetos de IA, antes do ChatGPT, o enfoque era supervisionado. Mas a abordagem supervisionada é difícil de ser escalonada, pois é impossível antecipar todas as perguntas que poderiam ser feitas à IA. Assim, o ChatGPT foi desenvolvido a partir de treinamento não supervisionado, ou seja, o modelo é treinado em dados nos quais nenhuma saída específica está associada a cada entrada. Na abordagem não supervisionada o modelo é treinado para aprender a estrutura subjacente e os padrões nos dados de entrada

sem nenhuma tarefa específica definida, como em agrupamentos, detecção de anomalias e redução de dimensionalidade. No contexto da modelagem de linguagem, o treinamento não supervisionado foi usado para treinar o modelo a entender a sintaxe e semântica da linguagem natural, para que o ChatGPT pudesse gerar texto coerente e significativo em um contexto de conversação.

Assim, o ChatGPT adquire seu aparente conhecimento ilimitado. Como os desenvolvedores não precisam saber as saídas que vêm das entradas, tudo o que eles têm que fazer é despejar mais e mais informações no mecanismo de treinamento do ChatGPT, que é chamado de modelagem de linguagem baseada em transformadores. A arquitetura de transformadores é um tipo de rede neural usada para processar dados de linguagem natural, simulando o funcionamento de um cérebro humano processando informações por meio de camadas de nós interconectados.

A arquitetura de transformadores processa sequências de palavras usando "auto atenção" para ponderar a importância de diferentes palavras em uma sequência ao fazer previsões. A "auto atenção" é semelhante à maneira como um leitor pode voltar a uma frase ou parágrafo anterior para o contexto necessário para entender uma nova palavra em um livro. O transformador examina todas as palavras em uma sequência para entender o contexto e as relações entre as palavras. O transformador é composto por várias camadas, cada uma com várias subcamadas. As duas principais subcamadas são a camada de "auto atenção" e a camada de *feedforward*. A camada de "auto atenção" calcula a importância de cada palavra na sequência, enquanto a camada de *feedforward* aplica transformações não lineares aos dados de entrada. Essas camadas ajudam o transformador a aprender e entender as relações entre as palavras em uma sequência.

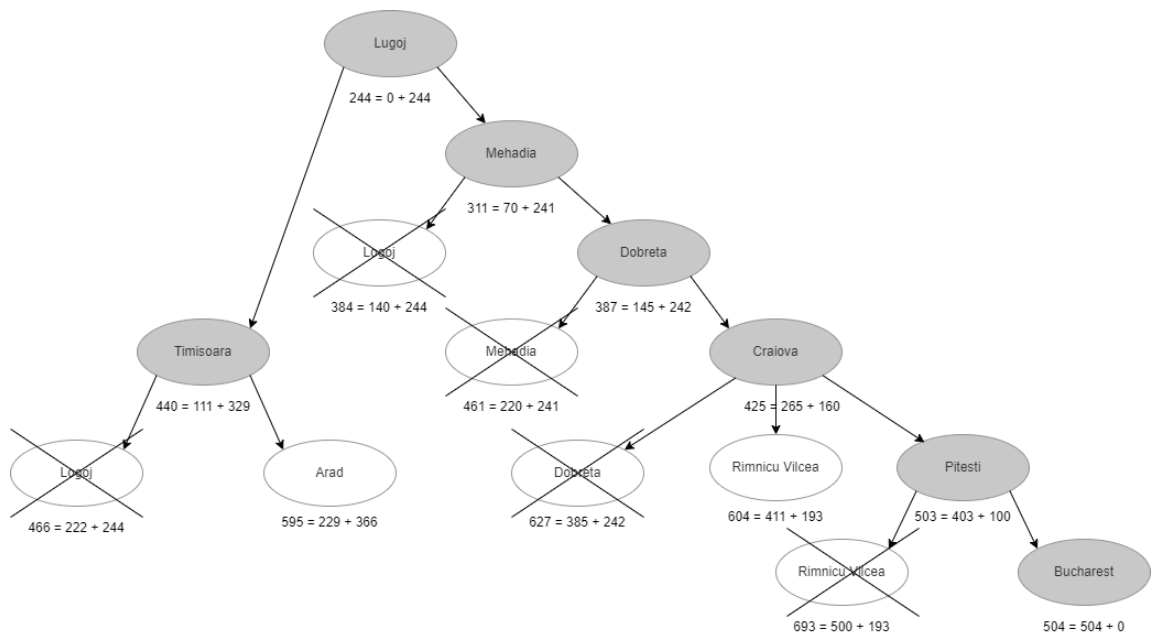
Durante o treinamento, o transformador recebe dados de entrada, como uma frase, e é solicitado a fazer uma previsão com base nessa entrada. O modelo é atualizado com base em quão bem sua previsão corresponde à saída real. Através desse processo, o transformador aprende a entender o contexto e as relações entre as palavras em uma sequência, tornando-se uma ferramenta poderosa para tarefas de processamento de linguagem natural, como tradução de idiomas e geração de texto. Há possibilidade da IA gerar conteúdo prejudicial ou tendencioso, pois podem aprender padrões e preconceitos presentes nos dados de treinamento. Por isso os desenvolvedores devem colocar limitações no conteúdo de treinamento e no aceite de resposta da IA, procurando prevenir e evitar questões sensíveis como racismo e extremismos. Também é preciso considerar que nem todo conteúdo produzido pelo ChatGPT é integralmente confiável, a IA está sendo gradualmente aprimorada e ainda comete erros, ou os chamados "delírios" ao ter a capacidade de inventar informações a partir de inferências feitas do conteúdo de seu treinamento.

Fontes:

(<https://www.zdnet.com/article/how-does-chatgpt-work/>);

(<https://help.openai.com/en/articles/7842364-how-chatgpt-and-our-language-models-are-developed>)

2 BUSCA HEURÍSTICA



3 LÓGICA

R4: $r \rightarrow p$	Equivalência Condicional em R3
R5: $q \rightarrow p$	Silogismo Hipotético em R2 e R4
R6: $q \rightarrow p \wedge p \rightarrow q$	Conjunção em R5 e R1
R7: $q \leftrightarrow p$	Equivalência Bicondicional em R6

4 REDES NEURAIIS ARTIFICIAIS

- a) 0,3
- b) 0,3
- c) -2
- d) -0,1391

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](#)) e 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 uma 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 imputados e indique o treinamento de cada modelo
- Grave os valores preditos em variáveis criadas
- Realize a análise de importância das variáveis para estimar a variável target, **para cada modelo treinado**
- Dê uma breve explicação (máximo de quatro linhas) sobre os resultados encontrados na análise de importância de variáveis
- Escolha o melhor modelo com base nas métricas de avaliação MSE, MAE e R^2
- Dê uma breve explicação (máximo de quatro linhas) sobre qual modelo gerou o melhor resultado e a métrica de avaliação utilizada

B - RESOLUÇÃO

1 ANÁLISE EXPLORATÓRIA DOS DADOS

```
a) 1. dados = pd.read_csv('precos_carros_brasil.csv', decimal=',')

b) 1. dados.isna().any()
   2. dados.isna().sum()
   3. dados = dados.dropna()

c) 1. dados.duplicated().sum()
   2. dados.drop_duplicates(inplace=True)

d) 1. colunas_numericas = [col for col in dados.columns if dados[col].dtype != 'object']
   2. colunas_categoricas = [col for col in dados.columns if dados[col].dtype == 'object']

e) 1. dados['model'].value_counts()
```

```
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
...
STEPWAY Zen Flex 1.0 12V Mec. 2
Saveiro Robust 1.6 Total Flex 16V CD 2
Saveiro Robust 1.6 Total Flex 16V 2
Gol Last Edition 1.0 Flex 12V 5p 2
Polo Track 1.0 Flex 12V 5p 2
Name: count, Length: 2112, dtype: int64
```

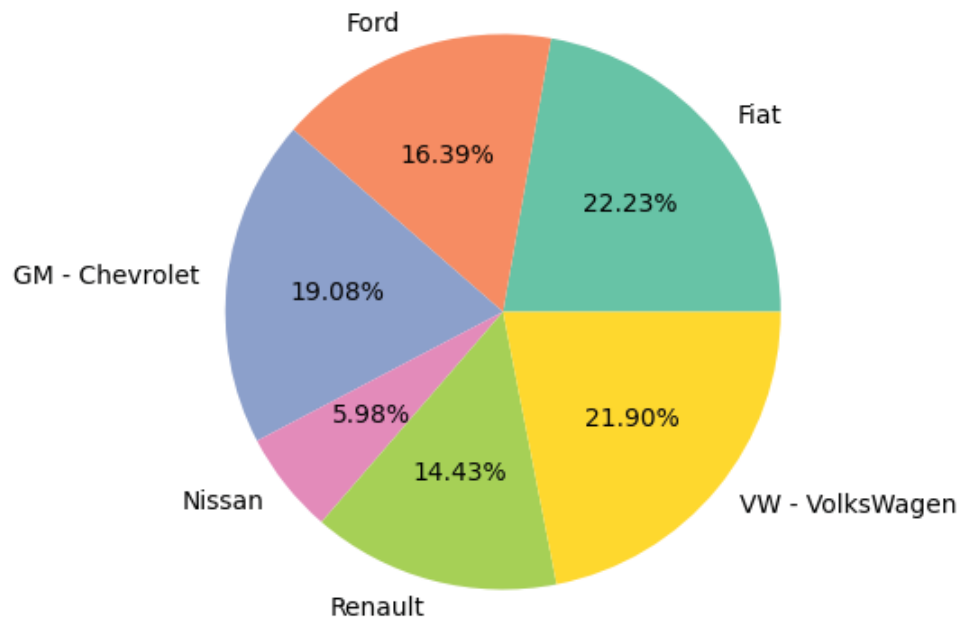
```
2. dados['brand'].value_counts()
```

```
brand
Fiat 44962
VW - Volkswagen 44312
GM - Chevrolet 38590
Ford 33150
Renault 29191
Nissan 12090
Name: count, dtype: int64
```

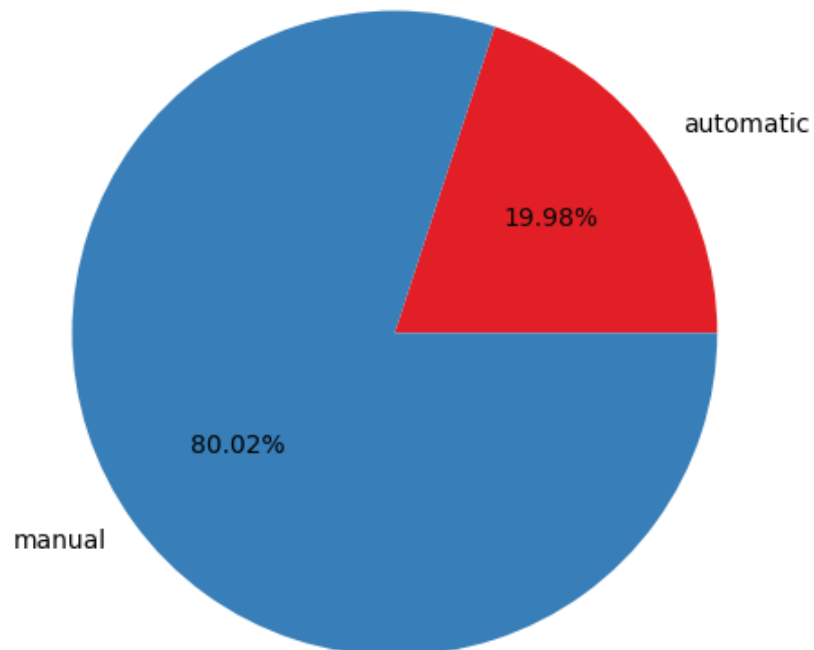
- f) Primeiramente, notou-se que as últimas linhas apresentam todas as colunas como NaN, ou seja, não disponíveis. Desse modo, optou-se por excluir as linhas com essa característica. No que se refere a dados duplicados, observou-se duas ocorrências, esses registros também foram apagados. Por fim, foram criados dois grupos, um para as colunas com dados numéricos e outra para dados categóricos.

2 VISUALIZAÇÃO DOS DADOS

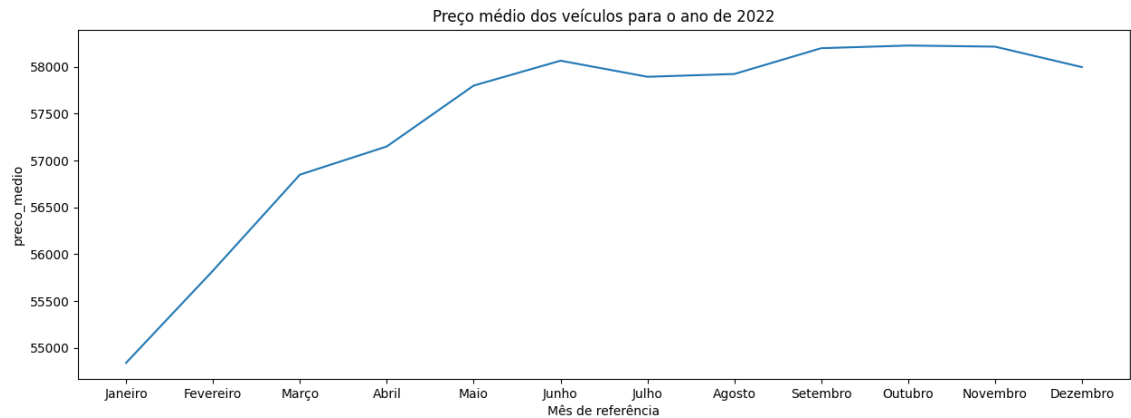
a) Distribuição da quantidade de carros por marca



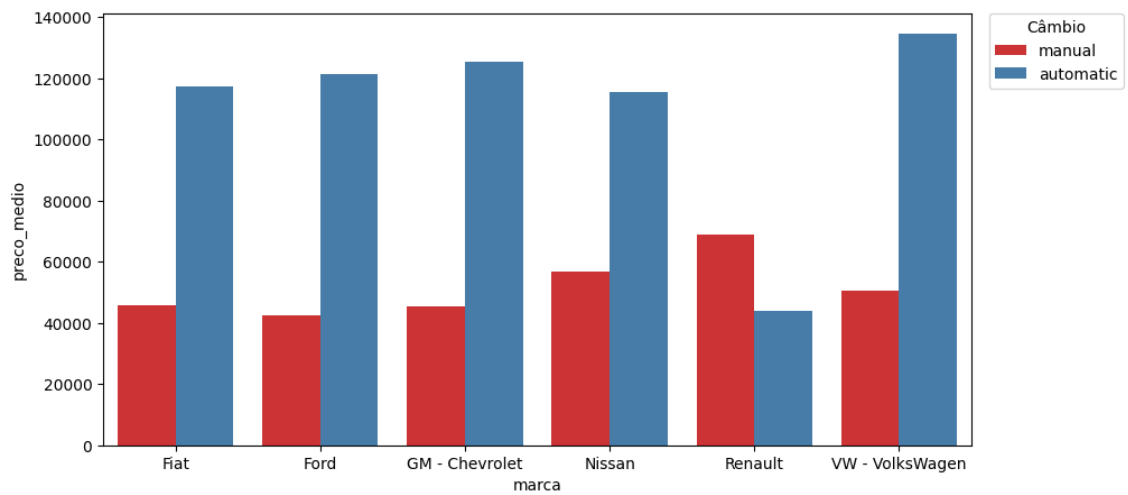
b) Distribuição da quantidade de carros por tipo de câmbio



c)

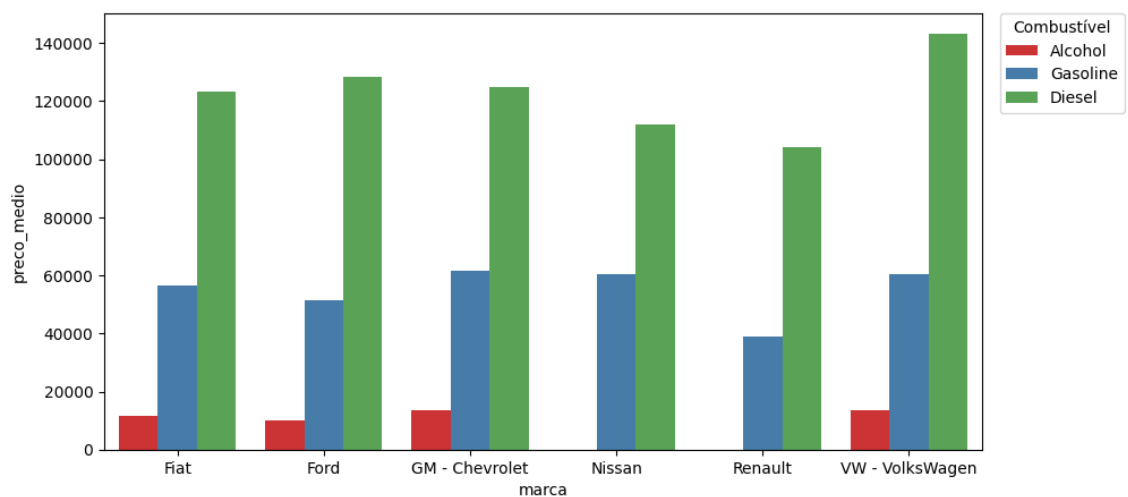


d)



e) Nota-se, pelo gráfico gerado, que, geralmente, carros com câmbio automático possuem um preço médio muito maior quando comparado com os carros com câmbio manual. A única exceção são os carros da marca Renault, que apresentam um valor médio maior para os modelos com câmbio manual.

f)



- g) Percebe-se que carros com combustível diesel possuem um preço médio bem maior que os modelos com outros tipos de combustíveis. Carros à gasolina possuem um preço médio em torno da metade dos modelos a diesel. Ainda, carros a álcool são os que possuem o menor valor, quando oferecidos pelas marcas. Nissan e Renault não oferecem modelos com esse tipo de combustível.

3 APLICAÇÃO DE MODELOS DE MACHINE LEARNING PARA PREVER O PREÇO MÉDIO DOS CARROS

```
a) 1. X = dados.drop(['preco_medio'],axis = 1)
    2. Y = dados['preco_medio']

b) 1. X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.25)

c) 1. model_rf = RandomForestRegressor()
    2. model_rf.fit(X_train, Y_train)

d) 1. valores_preditos_rf = model_rf.predict(X_test)
    2. valores_preditos_rf

e) 1. model_xgboost.feature_importances_
    2. feature_importances = pd.DataFrame(
        model_xgboost.feature_importances_,
        index = X_train.columns,
        columns=['importance']).sort_values('importance', ascending = False)
    3. feature_importances
```

	Importance
engine_size	0.426666
year_model	0.235787
combustivel	0.213182
cambio	0.071964
modelo	0.026653
ano	0.019103
mes_num	0.006645
abo_modelo	0.000000

```
4. mse = mean_squared_error(Y_test, valores_preditos_xgboost)
5. mse
```

2224917.727042086

```
6. mae = mean_absolute_error(Y_test, valores_preditos_xgboost)
7. mae
```

302.6570237540227

```
8. r2_score(Y_test, valores_preditos_xgboost)
9. r2_score
```

988026073370855

- f) Tanto no modelo de Random Forest como no modelo de XGBoost, a principal variável foi o *engine_size* (tamanho do motor). A variável *model_year* (ano do modelo) teve grande impacto, no entanto, no modelo de Random Forest teve quase o mesmo impacto que a variável tamanho do motor, enquanto no XGBoost ficou com importância menor, dividindo o posto com outra variável, *fuel*.
- g) Os melhores resultados foram obtidos com o modelo Random Forest, parâmetros default (mse próximo a 20180801, mae perto de 2340,6588 e r^2 score igual a 0,9925), o que se traduz em uma acurácia de mais de 99%. Para parâmetros personalizados do Random Forest, teve-se

mse 150349810, mae 4738,7235 e r^2 de 0,944133. Para o XGBoost, mse 42868382, mae 3811,0364 e r^2 pouco acima de 0,984071.

- h) Para se escolher o melhor resultado, toma-se como base a métrica R^2 , na qual, quanto maior o valor - de zero a um -, melhor o resultado. Para o teste de Random Forest com parâmetro default, o resultado foi de mais de 99%, valor considerado excelente. Os testes com parâmetros personalizado de Random Forest e XGBoost também tiveram resultados muito bons, acima dos 94%, mas não superior ao Random Forest default.

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;

- Syx%

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

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

7. Escolha o melhor modelo.

B – RESOLUÇÃO

1 PESQUISA COM DADOS DE SATÉLITE

```
a) 1. data("Satellite")
    2. dados<-Satellite[,c(17,18,19,20,37)]

b) 1. indices<-createDataPartition(dados$classes, p=0.80, list=FALSE)
    2. treino<-dados[indices,]
    3. teste<-dados[-indices,]

c) 1. rf<-caret::train(classes~., data=treino, method="rf")
    2. svm<-caret::train(classes~.,data=treino,method="svmRadial")
    3. rna<-caret::train(classes~.,data=treino,method="nnet", trace = FALSE)

d) 1. modelo <- c("Random Forest", "Support Vector Machine", "Neural Network")
    2. acuracia <- c(sum(diag(cm_rf))/sum(cm_rf),
                    sum(diag(cm_svm))/sum(cm_svm),
                    sum(diag(cm_rna))/sum(cm_rna))
    3. modelos<-c(rep("RF",6),rep("SVM",6),rep("RNA",6))
    4. grupo<-rep(c(1,2,3,4,5,6),6)
    5. descricao<-rep(c("solo vermelho", "colheita algodão", "solo cinza",
                        "solo cinza úmido", "restolho de vegetação",
                        "solo cinza muito úmido"),6)
    6. tpr<-round(c(tpr_rf, tpr_svm, tpr_rna),4)*100
    7. fpr<-round(c(fpr_rf, fpr_svm, fpr_rna),4)*100
    8. hr <-round(c(hr_rf, hr_svm, hr_rna),4)*100
    9. fscore<-round(c(fsc_rf,fsc_svm,fsc_rna),4)
    10. df2<-data.frame(modelos,grupo,descricao, tpr, fpr, hr, fscore)
```

- Com base nas matrizes de confusão, no geral, o melhor modelo foi o SVM com acurácia de 86%. Mas dependendo do interesse do pesquisador esse resultado pode ser alterado e avaliado conforme os seguintes indicadores:
 - Acurácia Geral do Modelo: Quantos por cento do total da população o modelo classificou corretamente;
 - True Positive Rate* (TPR) ou Sensibilidade: Do que originalmente era pertencente à categoria, quantos por cento o modelo classificou corretamente como a categoria;

- iii. *False Positive Rate* (FPR) ou Falso Positivo: Do que originalmente não era a categoria, quantos por cento o modelo classificou erroneamente como a categoria;
- iv. *Hit Rate* ou Precisão: De tudo o que o modelo apontou como a categoria, o que realmente acertou;
- v. Fscore: F1Score

2 ESTIMATIVA DE VOLUME DE ÁRVORES

```

a) 1. library(readr)
   2. url <- "http://www.razer.net.br/datasets/Volumes.csv"
   3. data <- read.csv2(url, sep = ";")

b) 1. df <- subset(data, select = -NR)

c) 1. indices<-createDataPartition(df$VOL, p=0.80, list=FALSE)
   2. treino<-df[indices,]
   3. teste<-df[-indices,]

d) 1. alom <- nls(VOL ~ b0 + b1*DAP*DAP*HT,df,start=list(b0=0.5, b1=0.5))
   2. rf<-caret::train(VOL~., data=treino, method="rf")
   3. svm<-caret::train(VOL~.,data=treino,method="svmRadial")
   4. rna<-caret::train(VOL~.,data=treino,method="nnet", trace = FALSE)

e) 1. predict.rf <-predict(rf,teste)
   2. predict.svm<-predict(svm,teste)
   3. predict.rna<-predict(rna,teste)
   4. predict_alom <- exp(predict(alom, newdata = teste, interval = "confidence"))

f) 1. coeficiente_determinacao <- function(y_obs, y_pred) {
   2.   # Calcular a média observada de y
   3.   y_medio <- mean(y_obs)
   4.
   5.   # Calcular a soma dos quadrados totais
   6.   SST <- sum((y_obs - y_medio)^2)
   7.
   8.   # Calcular a soma dos quadrados dos resíduos
   9.   SSR <- sum((y_obs - y_pred)^2)
  10.
  11.   # Calcular o coeficiente de determinação
  12.   R2 <- 1 - (SSR / SST)
  13.
  14.   return(R2)
  15. }
  16.
  17. erro_padrao_est<-function(y_obs,y_pred){
  18.   n<- length(y_obs)
  19.
  20.   # Calcular a soma dos quadrados dos resíduos
  21.   SSR <- sum((y_obs - y_pred)^2)
  22.
  23.   Sxy = sqrt(SSR/(n-2))
  24.
  25.   return(Sxy)
  26. }

```

- h) O melhor modelo é o Random Forest por apresentar melhor coeficiente de determinação, ou seja, melhor acurácia. Além disso apresenta menor erro padrão e menor percentual do erro.

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 mostrá-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.

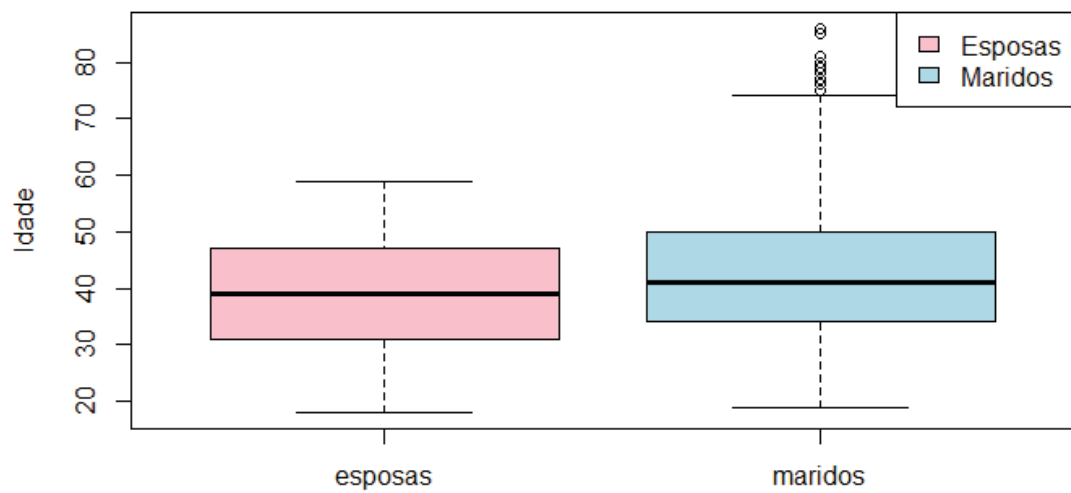
Lembre-se de que os intervalos de confiança já são mostrados nos resultados dos testes citados no item 1 acima.

B – RESOLUÇÃO

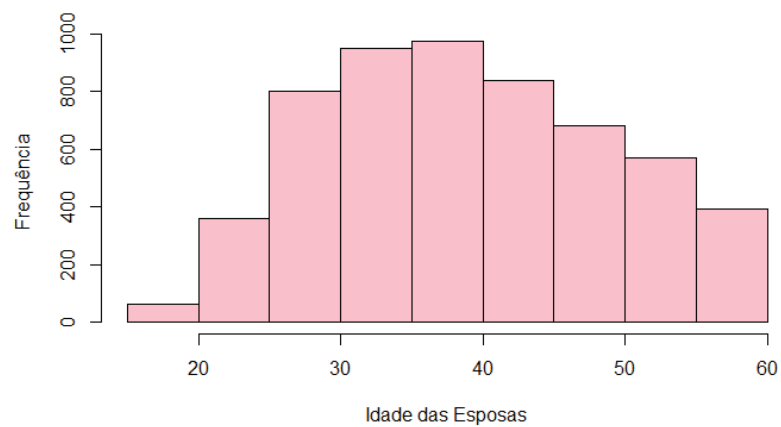
1 GRÁFICOS E TABELAS

a)

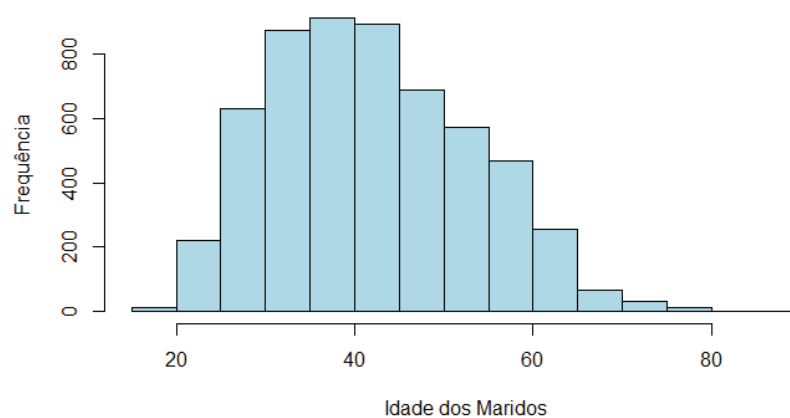
Boxplot Idades Esposas e Maridos



Histograma Idade das Esposas



Histograma Idade dos Maridos



b)

```
[1] "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

[1] "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
```

Na base salários, a idade dos maridos varia de 18,81 a 86,86 anos, a faixa etária dos homens com maior frequência é a de 38 a 43 anos com 917 registros. Nesse intervalo se encontra a mediana do grupo, pois até o limite superior dessa classe acumula-se 56,6% das observações relacionadas aos maridos. Há apenas 8 indivíduos homens com idade maior ou igual a 77,14 anos, demonstrando que seriam outliers. Quanto às esposas, menos de 10% delas tem idade inferior a 26,77 anos; a classe de maior frequência é a de 35,72 a 38,70 anos, a mediana está incluída na faixa etária de 38,71 a 41,68 anos.

2 MEDIDAS DE POSIÇÃO E DISPERSÃO

a)

```
1. media_grupo1 <- mean(salarios$age)
2. mediana_grupo1 <- median(salarios$age)
3. moda_grupo1 <- as.numeric(names(which.max(table(salarios$age))))
4.
5. media_grupo2 <- mean(salarios$husage)
6. mediana_grupo2 <- median(salarios$husage)
7. moda_grupo2 <- as.numeric(names(which.max(table(salarios$husage))))
8.
9. tabela <- data.frame(
10.   Grupo = c("Esposas", "Maridos"),
11.   Media = c(media_grupo1, media_grupo2),
12.   Mediana = c(mediana_grupo1, mediana_grupo2),
13.   Moda = c(modas_grupo1, modas_grupo2)
14. )
```

Description: df [2 × 4]

Grupo <chr>	Média <dbl>	Mediana <dbl>	Moda <dbl>
Esposas	39.42758	39	37
Maridos	42.45296	41	44

2 rows

Todos os valores das medidas resumo das esposas são inferiores aos dos esposos. Enquanto a média de idade da esposa é de 39,43 anos a do marido é de 42,45 anos. Metade do grupo de esposas tem idade até 39 anos, já para o grupo de maridos a idade que divide o grupo em 50% é 41 anos. Entre as esposas a idade com maior frequência é 37 anos, entre os maridos a idade mais frequente é 44 anos.

b)

```

1. var_grupo1 <- var(salarios$age)
2. dp_grupo1 <- sd(salarios$age)
3. cv_grupo1 <- 100 * (dp_grupo1 / mean(salarios$age))
4.
5. var_grupo2 <- var(salarios$husage)
6. dp_grupo2 <- sd(salarios$husage)
7. cv_grupo2 <- 100 * (dp_grupo2 / mean(salarios$husage))
8.
9. tabela2 <- data.frame(
10.   Grupo = c("Esposas", "Maridos"),
11.   Variância = c(var_grupo1, var_grupo2),
12.   Desvio_Padrao = c(dp_grupo1, dp_grupo2),
13.   Coeficiente_de_Variacao = c(cv_grupo1, cv_grupo2)
14. )

```

Description: df [2 × 4]

Grupo <chr>	Variância <dbl>	Desvio_Padrão <dbl>	Coeficiente_de_Varição <dbl>
Esposas	99.75234	9.98761	25.33153
Maridos	126.07173	11.22817	26.44849

2 rows

A variância da idade dos maridos é 1,26 vezes a variância da idade das esposas. Logo, a idade das esposas também apresenta um desvio padrão inferior ao da idade dos maridos. Enquanto as esposas apresentam um coeficiente de variação de 25,33% na idade, os maridos apresentam idade com coeficiente de variação de 26,45%.

3 TESTES PARAMÉTRICOS OU NÃO PARAMÉTRICOS

- a) Nos testes de normalidade Kolmogorov-Smirnov para a idade das esposas o p-valor obtido foi 0.00000000000000022204, ou seja, muito menor do que 0.05 indicando a não normalidade dos dados. Isso também ocorreu para as idades dos homens, ou seja, para ambas as variáveis não há normalidade, indicando necessidade de realizar teste não paramétrico U de Mann-Whitney para comparação de medianas.

Também foi testada a hipótese nula das medianas das idades dos maridos e esposas serem iguais, mas o teste U de Mann Whitney rejeitou a hipótese nula apresentando um p-valor idêntico ao calculado nos testes de normalidade Kolmogorov-Smirnov, muito inferior ao nível de significância 0,05. Assim, pode-se concluir que a idade mediana dos maridos é estatisticamente diferente da idade mediana da idade das mulheres. O intervalo de confiança da diferença entre as medianas está entre -3,00 e -2,00, com uma mediana de -2,99.

APÊNDICE 5 – ESTATÍSTICA APLICADA II

A – ENUNCIADO

1) 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)
husbck = 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 antilogaritmo (antilog) para obter o resultado da predição.

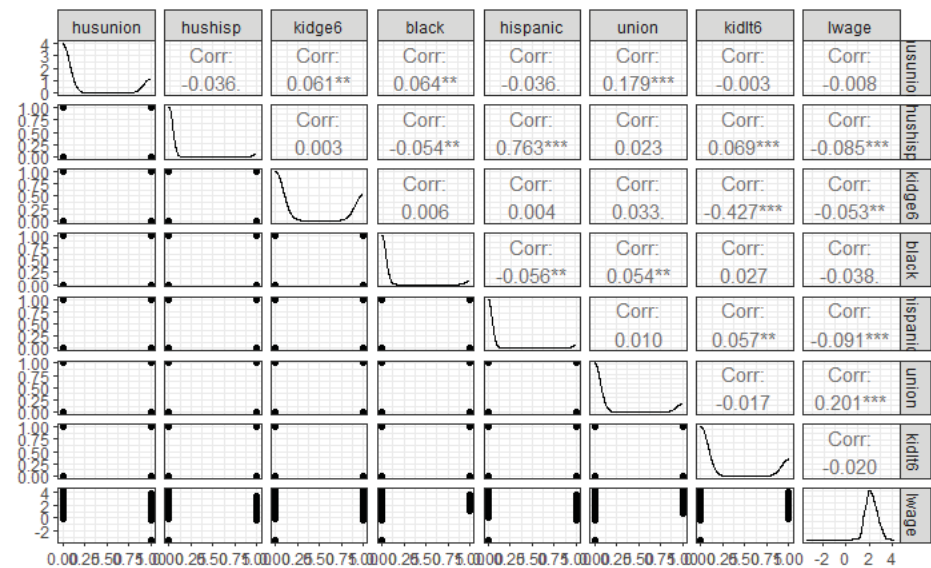
B – RESOLUÇÃO

1 REGRESSÕES RIDGE, LASSO E ELASTICNET

A rotina utilizada para análise do conjunto de dados começa com a leitura do arquivo trabalhosalarios.csv. Depois disso, são feitas algumas atividades de exploração e correção da base, a saber:

i. Correlações;

```
1. bin<-salarios[, -c(1,3,4,5,7,9,10,12,15)]
2. nonbin<-salarios[, c(1,3,4,5,7,9,10,12,15,17)]
```



ii. Regressão Linear Simples;

```
1. resultados <- lm(lwage~., data=salarios)
2. summary(resultados)
```

Call:

```
lm(formula = lwage ~ ., data = salarios)
```

Residuals:

Min	1Q	Median	3Q	Max
-4.9480	-0.1218	0.0033	0.1230	1.7260

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-0.75720680	1.74920919	-0.433	0.665135
husage	0.00105350	0.00130797	0.805	0.420637
husunion	0.00585021	0.01418495	0.412	0.680064
husearns	0.00011476	0.00002000	5.738	0.000000107 ***
huseduc	0.00399332	0.00279099	1.431	0.152612
husblk	0.00323764	0.07133041	0.045	0.963801
hushisp	-0.01033332	0.04076495	-0.253	0.799913
hushrs	-0.00099055	0.00045666	-2.169	0.030166 *
kidge6	0.01895905	0.01356875	1.397	0.162457
earns	0.00161127	0.00002781	57.948	< 0.000000000000002 ***
age	0.31459518	0.29146644	1.079	0.280532
black	-0.03657620	0.07229459	-0.506	0.612947
educ	-0.29189799	0.29150719	-1.001	0.316757
hispanic	-0.05939173	0.03888849	-1.527	0.126827
union	0.06012940	0.01702886	3.531	0.000421 ***
exper	-0.31402053	0.29149039	-1.077	0.281451
kidlt6	0.07147817	0.01637435	4.365	0.0000132021 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2909 on 2557 degrees of freedom

Multiple R-squared: 0.6914, Adjusted R-squared: 0.6894

F-statistic: 358 on 16 and 2557 DF, p-value: < 0.0000000000000022

iii. Outliers;

```
1. outlierTest(resultados)
```

```
2582      -18.13200
5517      -8.902132
1518      -7.835112
2225      -6.678241
316       -6.145771
4232       5.983227
859        5.706200
2745      -5.562087
3566      -5.205824
5052      -5.139500
```

```
1. row.names.salarios<-c(2582,5517,1518,2225,316,4232,859,2745,3566,5052)
2. salarios <- salarios[!(row.names(salarios) %in% row.names.salarios),]
3. summary(resultados)
```

Call:

```
lm(formula = lwage ~ ., data = salarios)
```

Residuals:

```
      Min       1Q   Median       3Q      Max
-1.34290 -0.12468 -0.00209  0.10773  1.45091
```

Coefficients:

```
              Estimate Std. Error t value      Pr(>|t|)
(Intercept) -0.64141062  1.50955996  -0.425    0.670946
usage       -0.00010907  0.00113117  -0.096    0.923194
union        0.02046460  0.01227495   1.667    0.095601 .
husearns     0.00011637  0.00001729   6.732    0.0000000000206 ***
huseduc      0.00327480  0.00241897   1.354    0.175921
husblack     0.11289971  0.06349405   1.778    0.075504 .
hushisp     -0.02633080  0.03540960  -0.744    0.457183
hushrs      -0.00077199  0.00039501  -1.954    0.050768 .
kidge6       0.02038797  0.01174040   1.737    0.082585 .
earnings     0.00168963  0.00002528  66.829 < 0.000000000000002 ***
age          0.29737798  0.25153191   1.182    0.237210
black       -0.14659955  0.06416005  -2.285    0.022400 *
educ        -0.27630340  0.25156597  -1.098    0.272163
hispanic    -0.01904529  0.03387662  -0.562    0.574032
union       0.04916728  0.01471898   3.340    0.000849 ***
exper       -0.29532607  0.25155266  -1.174    0.240500
kidlt6      0.07645003  0.01415708   5.400    0.0000000727520 ***
---
```

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2511 on 2547 degrees of freedom

Multiple R-squared: 0.7465, Adjusted R-squared: 0.7449

F-statistic: 468.7 on 16 and 2547 DF, p-value: < 0.0000000000000022

iv. Multicoliniaridade;

```
1. car::vif(resultados)
2. resultados <- lm(lwage~., data=salarios2)
3. car::vif(resultados)
4. summary(resultados)
```

```

  husage husunion husearns huseduc husblck hushisp hushrs
4.587359 1.055762 1.352849 1.435987 1.027656 2.433792 1.055999

kidge6   earns hispanic   union   exper   kidlt6
271893 1.250909 2.373772 1.089240 4.652911 1.543761

Call:
lm(formula = lwage ~ ., data = salarios2)

Residuals:
    Min       1Q   Median       3Q      Max
-1.31055 -0.12302  0.00164  0.10519  1.49291

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.26479598  0.04166325  30.358 < 0.0000000000000002 ***
husage       0.00253594  0.00107975   2.349    0.0189 *
husunion     0.01793918  0.01240155   1.447    0.1482
husearns     0.00011999  0.00001746   6.871    0.000000000008 ***
huseduc      0.01040304  0.00221242   4.702    0.000002712017 ***
husblck      -0.02381005  0.02053398  -1.160    0.2463
hushisp      -0.03519878  0.03577212  -0.984    0.3252
hushrs       -0.00071125  0.00039905  -1.782    0.0748 .
kidge6       0.02068915  0.01186785   1.743    0.0814 .
earns        0.00174094  0.00002459  70.811 < 0.0000000000000002 ***
hispanic     -0.02788505  0.03422014  -0.815    0.4152
union        0.05990478  0.01480371   4.047    0.000053511298 ***
exper        -0.00146564  0.00112453  -1.303    0.1926
kidlt6       0.07573194  0.01430834   5.293    0.000000130683 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2538 on 2550 degrees of freedom
Multiple R-squared:  0.7406,    Adjusted R-squared:  0.7393
F-statistic: 560.1 on 13 and 2550 DF,  p-value: < 0.00000000000000022

```

v. Teste de normalidade;

```
1. shapiro.test(resultados$residuals)
```

Shapiro-Wilk normality test

data: resultados\$residuals

W = 0.92683, p-value < 0.00000000000000022

```
1. lillie.test(resultados$residuals)
```

Lilliefors (Kolmogorov-Smirnov) normality test

data: resultados\$residuals

D = 0.10054, p-value < 0.00000000000000022

vi. Transformação de Box-Cox.

```

1. salarios3=as.matrix(salarios2)
2. outlm <- boxcoxlm(x = salarios3[,1:13], y = salarios3[,14])
3. outlm$p.value
4. shapiro.test(outlm$tf.residuals)

```


ii. Regressão Lasso

```

1. predictions_train <- predict(lasso_model,
2.                               s = best_lambda_lasso,
3.                               newx = x)
4.
5. eval_results <- function(true, predicted, df) {
6.   SSE <- sum((predicted - true)^2)
7.   SST <- sum((true - mean(true))^2)
8.   R_square <- 1 - SSE / SST
9.   RMSE = sqrt(SSE/nrow(df))
10.
11.   data.frame(
12.     RMSE = RMSE,
13.     Rsquare = R_square
14.   )
15. }
16.
17. lasso_train<-eval_results(y_train, predictions_train, train)
18. lasso_train
19.
20. predictions_test <- predict(lasso_model,
21.                              s = best_lambda_lasso,
22.                              newx = x_test)
23.
24. lasso_test<-eval_results(y_test, predictions_test, test)
25. lasso_test

```

Para a massa de treino a Regressão Lasso tem Raiz do Erro Quadrático Médio RMSE de 0.4916 e R2 de 0,7582.

Para a massa de teste a Regressão Lasso tem Raiz do Erro Quadrático Médio RMSE de 0.5309 e R2 de 0,7129.

iii. Regressão ElasticNet

```

1. train_cont <- trainControl(method = "repeatedcv",
2.                              number = 10,
3.                              repeats = 5,
4.                              search = "random",
5.                              verboseIter = TRUE)
6.
7. elastic_reg <- train(lwage ~ .,
8.                      data = train,
9.                      method = "glmnet",
10.                     tuneLength = 10,
11.                     trControl = train_cont)
12.
13. elastic_reg$bestTune
14.
15. elastic_reg[["finalModel"]][["beta"]]
16.
17. predictions_train <- predict(elastic_reg, x)
18.
19. eval_results <- function(true, predicted, df) {
20.   SSE <- sum((predicted - true)^2)
21.   SST <- sum((true - mean(true))^2)
22.   R_square <- 1 - SSE / SST
23.   RMSE = sqrt(SSE/nrow(df))

```

(continua)

(continuação)

```
24.  
25.   data.frame(  
26.     RMSE = RMSE,  
27.     Rsquare = R_square  
28.   )  
29. }  
30.  
31. el_train<-eval_results(y_train, predictions_train, train)  
32. el_train  
33.  
34. predictions_test <- predict(elastic_reg, x_test)  
35.  
36. el_test<-eval_results(y_test, predictions_test, test)  
37. el_test
```

Para a massa de treino a Regressão Elastic Net tem Raíz do Erro Quadrático Médio RMSE de 0.4906 e R2 de 0,7592.

Para a massa de teste a Regressão Elastic Net tem Raíz do Erro Quadrático Médio RMSE de 0.5306 e R2 de 0,7132.

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 DE CARACTERÍSTICAS: IDENTIFICADOR AUTOMÁTICO DE IDIOMA

```

1. #Característica 1
2. def tamanhoMedioFrases(texto):
3.     palavras = re.split("\s", texto)
4.
5.     tamanhos = [len(s) for s in palavras if len(s)>0]
6.
7.     soma = 0
8.     for t in tamanhos:
9.         soma=soma+t
10.    return soma / len(tamanhos)

```

```

1. #Característica 2
2. def encontra_upside(texto):
3.     if '¿' or '¡' in texto:
4.         return 1
5.     else:
6.         return 0

```

```

1. #Característica 3
2. def encontra_acentos_agudos(texto):
3.     acentos_agudos = 'áéíóúÁÊÎÖÜ'
4.     for caractere in texto:
5.         if caractere in acentos_agudos:
6.             return 1
7.     return 0

```

```

1. #Característica 4
2. def encontra_crase(texto):
3.     if 'à' in texto or 'Â' in texto:
4.         return 1
5.     return 0

```

```

1. #Característica 5
2. def encontra_letra_k(texto):
3.     if 'k' in texto or 'K' in texto:
4.         return 1
5.     return 0

```

```

1. #Característica 6
2. def encontra_til(texto):
3.     vogais_til = 'ãõÃÕ'
4.     consoantes_til = 'ñÑ'
5.     for caractere in texto:
6.         if caractere in vogais_til:
7.             return 1
8.         elif caractere in consoantes_til:
9.             return 2
10.    return 0

```

```

1. #Característica 7
2. def encontra_circunflexo(texto):
3.     caracteres_circunflexo = 'âêîôûÂÊÎÔÛ'
4.     for caractere in texto:
5.         if caractere in caracteres_circunflexo:
6.             return 1
7.     return 0

```

```

1. #Característica 8
2. def encontrar_pronomes_ingles(frase):
3.     # Lista de pronomes em inglês
4.     pronomes = ["I", "you", "he", "she", "it", "we", "they",
5.                 "me", "him", "her", "us", "them", "my",
6.                 "your", "his", "her", "its", "our", "their",
7.                 "mine", "yours", "hers", "ours", "theirs",
8.                 "myself", "yourself", "himself", "herself", "itself",
9.                 "ourselves", "yourselves", "themselves"]
10.
11.     palavras = frase.split()
12.
13.     for palavra in palavras:
14.         if palavra in pronomes:
15.             return 1
16.     return 0

```

```

1. #característica 9
2. def encontrar_pronomes_espanhol(frase):
3.     # Lista de pronomes em espanhol
4.     pronomes = ["yo", "tú", "él", "ella", "ello", "nosotros", "nosotras", "vosotros",
5.                 "vosotras", "ellos", "ellas", "me", "te", "lo", "la", "nos", "os",
6.                 "los", "las", "mi", "mis", "tu", "tus", "su", "sus", "nuestro",
7.                 "nuestra", "nuestros", "nuestras", "vuestro", "vuestra", "vuestros",
8.                 "vuestras", "mío", "mía", "míos", "mías", "tuyo", "tuya", "tuyos",
9.                 "tuyas", "suyo", "suya", "suyos", "suyas", "nuestro", "nuestra",
10.                "nuestros", "nuestras", "vuestro", "vuestra", "vuestros", "vuestras",
11.                "me", "te", "se", "nos", "os", "se", "mí", "ti", "sí", "conmigo",
12.                "contigo", "consigo", "esto", "eso", "aquello", "este", "ese",
13.                "aquel", "estos", "esos", "aquellos", "esta", "esa", "aquella",
14.                "estas", "esas", "aquellas"]
15.
16.     palavras = frase.split()
17.
18.     for palavra in palavras:
19.         if palavra in pronomes:
20.             return 1
21.     return 0

```

```

1. #Característica 10
2. def encontrar_pronomes_portugues(frase):
3.     # Lista de pronomes em português
4.     pronomes = ["eu", "tu", "ele", "ela", "nós", "vós", "eles", "elas", "me", "te",
5.                 "se", "nos", "vos", "lhe", "lhes", "o", "a", "os", "as", "meu",
6.                 "minha", "meus", "minhas", "teu", "tua", "teus", "tuas", "seu",
7.                 "sua", "seus", "suas", "nosso", "nossa", "nossos", "nossas", "vosso",
8.                 "vossa", "vossos", "vossas", "mim", "ti", "si", "conosco",
9.                 "convosco", "este", "esta", "estes", "estas", "esse", "essa",
10.                "esses", "essas", "aquele", "aquela", "aqueles", "aquelas", "isto",
11.                "isso", "aquilo", "mesmo", "mesma", "mesmos", "mesmas", "próprio",
12.                "própria", "próprios", "próprias", "alguém", "ninguém", "tudo",
13.                "nada", "algo", "cada", "qualquer", "todos", "todas", "outro",
14.                "outra", "outros", "outras", "certo", "certa", "certos", "certas",
15.                "muito", "muita", "muitos", "muitas", "pouco", "pouca", "poucos",
16.                "poucas", "todo", "toda", "todos", "todas", "qual", "quais",
17.                "quanto", "quanta", "quantos", "quantas", "que", "quem", "onde",
18.                "como", "quando"]
19.
20.     palavras = frase.split()
21.
22.     for palavra in palavras:
23.         if palavra.lower() in pronomes: # Verifica considerando letras minúsculas
24.             return 1
25.     return 0

```

```

1. # Função extrai características
2. def extraiCaracteristicas(frase):
3.     # frase é um vetor [ 'texto', 'lingua' ]
4.     texto = frase[0]
5.     pattern_regex = re.compile('[^\w+]', re.UNICODE)
6.     texto = re.sub(pattern_regex, ' ', texto)
7.
8.     caracteristica1=tamanhoMedioFrases(texto)
9.     caracteristica2=encontra_upside(texto)
10.    caracteristica3=encontra_acentos_agudos(texto)
11.    caracteristica4=encontra_crase(texto)
12.    caracteristica5=encontra_letra_k(texto)
13.    caracteristica6=encontra_til(texto)
14.    caracteristica7=encontra_circunflexo(texto)
15.    caracteristica8=encontrar_pronomes_ingles(texto)
16.    caracteristica9=encontrar_pronomes_espanhol(texto)
17.    caracteristica10=encontrar_pronomes_portugues(texto)
18.
19.    # acrescente as suas funcoes no vetor padrao
20.    padrao = [caracteristica1, caracteristica2, caracteristica3, caracteristica4,
21.              caracteristica5, caracteristica6, caracteristica7, caracteristica8,
22.              caracteristica9, caracteristica10, frase[1]]
23.    return padrao

```

```

1. # Função gera padrões
2. def geraPadroes(frases):
3.     padroes = []
4.     for frase in frases:
5.         padrao = extraiCaracteristicas(frase)
6.         padroes.append(padrao)
7.     return padroes

```

```

1. from sklearn import svm
2. from sklearn.metrics import confusion_matrix
3. from sklearn.metrics import classification_report
4.
5. treinador = svm.SVC() #algoritmo escolhido
6. modelo = treinador.fit(X_train, y_train)
7.
8. # score com os dados de treinamento
9. acuracia = modelo.score(X_train, y_train)
10. print("Acurácia nos dados de treinamento: {:.2f}%".format(acuracia * 100))
11.
12. # melhor avaliar com a matriz de confusão
13. y_pred = modelo.predict(X_train)
14. cm = confusion_matrix(y_train, y_pred)
15. print(cm)
16. print(classification_report(y_train, y_pred))
17.
18. # com dados de teste que não foram usados no treinamento
20. print('métricas mais confiáveis')
21. y_pred2 = modelo.predict(X_test)
22. cm = confusion_matrix(y_test, y_pred2)
23. print(cm)
24. print(classification_report(y_test, y_pred2))

```

```

Acurácia nos dados de treinamento: 73.91%
[[10  0 13]
 [ 1 18  3]
 [ 1  0 23]]

```

	precision	recall	f1-score	support
espanhol	0.83	0.43	0.57	23
inglês	1.00	0.82	0.90	22
português	0.59	0.96	0.73	24
accuracy			0.74	69
macro avg	0.81	0.74	0.73	69
weighted avg	0.80	0.74	0.73	69

```

métricas mais confiáveis
[[1 1 5]
 [0 7 1]
 [2 0 6]]

```

	precision	recall	f1-score	support
espanhol	0.33	0.14	0.20	7
inglês	0.88	0.88	0.88	8
português	0.50	0.75	0.60	8
accuracy			0.61	23
macro avg	0.57	0.59	0.56	23
weighted avg	0.58	0.61	0.57	23

2 MELHORE UMA BASE DE DADOS RUIM

```

1. treinador_orig = svm.SVC()
2. modelo_orig = treinador_orig.fit(X_orig_train, y_orig_train)

```

```

1. y_orig_pred = modelo_orig.predict(X_orig_train)
2. cm_orig_train = confusion_matrix(y_orig_train, y_orig_pred)
3. print('Matriz de confusão - com os dados SEM TRATAMENTO usados no TREINAMENTO')
4. print(cm_orig_train)
5. print(classification_report(y_orig_train, y_orig_pred))

```

```

Matriz de confusão - com os dados SEM TRATAMENTO usados no TREINAMENTO
[[ 0  0  1  7  0  0]
 [ 0  0 11 29  0  0]
 [ 0  0 238 273  0  0]
 [ 0  0 100 378  0  0]
 [ 0  0 13 134  2  0]
 [ 0  0  2 11  0  0]]

```

	precision	recall	f1-score	support
3	0.00	0.00	0.00	8
4	0.00	0.00	0.00	40
5	0.65	0.47	0.54	511
6	0.45	0.79	0.58	478
7	1.00	0.01	0.03	149
8	0.00	0.00	0.00	13
accuracy			0.52	1199
macro avg	0.35	0.21	0.19	1199
weighted avg	0.58	0.52	0.46	1199

```

1. y2_orig_pred = modelo_orig.predict(X_orig_test)
2. cm_orig_test = confusion_matrix(y_orig_test, y2_orig_pred)
3. print('Matriz de confusão - com os dados SEM TRATAMENTO usados para TESTES')
4. print(cm_orig_test)
5. print(classification_report(y_orig_test, y2_orig_pred))

```

Matriz de confusão - com os dados SEM TRATAMENTO usados para TESTES

```

[[ 0  0  0  2  0  0]
 [ 0  0  2 11  0  0]
 [ 0  0 66 104  0  0]
 [ 0  0 31 129  0  0]
 [ 0  0  6  44  0  0]
 [ 0  0  1  4  0  0]]

```

	precision	recall	f1-score	support
3	0.00	0.00	0.00	2
4	0.00	0.00	0.00	13
5	0.62	0.39	0.48	170
6	0.44	0.81	0.57	160
7	0.00	0.00	0.00	50
8	0.00	0.00	0.00	5
accuracy			0.49	400
macro avg	0.18	0.20	0.17	400
weighted avg	0.44	0.49	0.43	400

```

1. X_train, X_test, y_train, y_test = train_test_split(
2.     X, y, test_size=0.25, stratify=y, random_state=10)
3.
4. treinador = svm.SVC()
5.
6. modelo = treinador.fit(X_train, y_train)

```

```

1. y_pred = modelo.predict(X_train)
2. cm_train = confusion_matrix(y_train, y_pred)
3. print('Matriz de confusão - com os dados TRATADOS usados no TREINAMENTO')
4. print(cm_train)
5. print(classification_report(y_train, y_pred))

```

Matriz de confusão - com os dados TRATADOS usados no TREINAMENTO

```

[[ 0 12  8  0]
 [ 0 179 55  0]
 [ 0  89 162  0]
 [ 0  8  54  0]]

```

	precision	recall	f1-score	support
4	0.00	0.00	0.00	20
5	0.62	0.76	0.69	234
6	0.58	0.65	0.61	251
7	0.00	0.00	0.00	62
accuracy			0.60	567
macro avg	0.30	0.35	0.32	567
weighted avg	0.51	0.60	0.55	567

```

1. y2_pred = modelo.predict(X_test)
2. cm_test = confusion_matrix(y_test, y2_pred)
3. print('Matriz de confusão - com os dados TRATADOS usados para TESTES')
4. print(cm_test)
5. print(classification_report(y_test, y2_pred))

```

Matriz de confusão - com os dados TRATADOS usados para TESTES

```

[[ 0  6  1  0]
 [ 0 59 19  0]
 [ 0 40 44  0]
 [ 0  3 18  0]]

```

	precision	recall	f1-score	support
4	0.00	0.00	0.00	7
5	0.55	0.76	0.63	78
6	0.54	0.52	0.53	84
7	0.00	0.00	0.00	21
accuracy			0.54	190
macro avg	0.27	0.32	0.29	190
weighted avg	0.46	0.54	0.49	190

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

1 CLASSIFICAÇÃO

Base de Dados “Veículo”

Técnica	Parâmetro	Acurácia	Matriz de Confusão				
RF – Hold - out	mtry=2	0,7186	Prediction	bus	opel	saab	van
			bus	42	0	1	0
			opel	0	22	21	0
			saab	0	17	18	1
			van	1	3	3	38
SVM – CV	C=1 Sigma=0,06828384	0,7784	Prediction	bus	opel	saab	van
			bus	42	0	2	1
			opel	0	27	14	0
			saab	0	14	24	1
			van	1	1	3	37
SVM – Hold - out	C=1 Sigma=0,06828384	0,7784	Prediction	bus	opel	saab	van
			bus	42	0	2	1
			opel	0	27	14	0
			saab	0	14	24	1
			van	1	1	3	37
RF – CV	mtry=10	0,7246	Prediction	bus	opel	saab	van
			bus	42	1	1	0
			opel	0	21	17	1
			saab	1	19	21	1
			van	0	1	4	37
RNA – Hold-out	size=5 decay=0,1	0,6228	Prediction	bus	opel	saab	van
			bus	30	1	6	3
			opel	0	4	1	0
			saab	12	31	34	0
			van	1	6	2	36
KNN	k=1	0,6647	Prediction	bus	opel	saab	van
			bus	33	3	4	4
			opel	0	16	15	1
			saab	6	21	22	1
			van	2	0	0	42
RNA – CV	size=31 decay=0,7	0,8263	Prediction	bus	opel	saab	van
			bus	42	1	0	1
			opel	0	29	12	0
			saab	0	11	30	1
			van	1	1	1	37

Predição “Veículo”

Comp	85	115	105
Circ	37	61	51
DCirc	87	116	107
RadRa	158	230	219
PrAxisRa	71	70	67
MaxLRa	5	11	11
ScatRa	152	310	218
Elong	42	10	33
PrAxisRect	15	25	25
MaxLRect	140	159	157
ScVarMaxis	169	125	225
ScVarmaxis	300	730	633
RaGyr	185	348	221
SkewMaxis	60	70	74
Skewmaxis	7	15	15
Kurtmaxis	15	8	19
KurtMaxis	185	179	183
HollRa	180	201	197
tipo	bus	saab	saab

Base de Dados “Diabetes”

Técnica	Parâmetro	Acurácia	Matriz de Confusão		
RF – Hold - out	mtry=2	0,7255	Prediction	neg	pos
			neg	89	31
			pos	11	22
SVM – CV	C=0,05 Sigma=0,09755161	0,7451	Prediction	neg	pos
			neg	91	30
			pos	9	23
SVM – Hold - out	C=0,25 Sigma=0,09755161	0,7516	Prediction	neg	pos
			neg	93	31
			pos	7	22
RF – CV	mtry=2	0,7451	Prediction	neg	pos
			neg	90	29
			pos	10	24
RNA – Hold - out	size=1 decay=0,1	0,6471	Prediction	neg	pos
			neg	96	50
			pos	4	3
KNN	k=9	0,6883	Prediction	neg	pos
			neg	85	26
			pos	22	21
RNA – CV	size=11 decay=0,1	0,719	Prediction	neg	pos
			neg	87	30
			pos	13	23

Predição “Diabetes”

preg0nt	4	5	7
glucose	200	80	170
pressure	75	60	65
triceps	45	0	30
insulin	0	0	0
mass	35,7	28,1	22,1
pedigree	0,627	0,351	0,562
age	35	25	60
diabetes	pos	neg	pos

2 REGRESSÃO

Base de Dados “Admissão”

Técnica	Parâmetro	R2	Syx	Pearson	Rmse	MAE
RF – Hold - out	mtry=2	0,8234332	0,05641442	0,9103035	0,0558358	0,03989843
RF – CV	mtry=2	0,8277202	0,05572535	0,91234	0,05515379	0,03889543
SVM – Hold - out	C=0,5 Sigma=0,1730883	0,8251234	0,05614375	0,909074	0,05556791	0,03977015
SVM – CV	C=1 Sigma=0,1710585	0,8224601	0,05656965	0,9070763	0,05598944	0,03940272
RNA – Hold - out	size=5 decay=0,1	0,8242099	0,05629019	0,9110211	0,05571284	0,04164856
RNA – CV	size=10 decay=0,1	0,8370628	0,05419331	0,9179804	0,05363747	0,01011106
KNN	K=7	0,7776499	0,06330737	0,8823635	0,06265805	0,01883382

Predição “Diabetes”

GREScore	300	354	200
TEFLScore	108	119	120
University Rating	3	4	5
SOP	3,5	5,0	5,0
LOR	3,5	4,5	5,0
CGPA	7,77	7,50	8,00
Research	0	1	1
PredictedChanceOfAdmit	0,6033419	0,6536553	0,6758379

Base de Dados “Biomassa”

Técnica	Parâmetro	R2	Syx	Pearson	Rmse	MAE
RF – Hold-out	mtry=2	0,6590796	1455,6148	0,9586842	1431,1490	258,2948
RF – CV	mtry=2	0,6515169	1471,6714	0,9533632	1446,9357	260,0076
SVM – Hold-out	C=1 Sigma=0,744	0,1290280	2326,6020	0,4455273	2287,4966	406,4971
SVM – CV	C=50 Sigma=0,01	0,8822678	855,3966	0,9894328	841,0192	189,6646
RNA – Hold-out	size=5 decay=0,1	0,7580049	1226,3749	0,9898464	1205,7621	291,6457
RNA – CV	size=7 decay=0,4	0,7019301	1361,0656	0,9671596	1338,1889	256,3490
KNN	K=3	0,6902221	1387,5390	0,9560010	1364,7173	260,6396

Predição “Biomassa”

dap	6,6	5,3	17,5
h	7,5	7,5	17,5
Me	2,00	0,50	1,04
Predicted biomassa	546,69398	69,04666	310,29891

3 AGRUPAMENTO

1. Cluster modes:										
2.	Comp	Circ	DCirc	RadRa	PrAxisRa	MaxLRa	ScatRa	Elong	PrAxisRect	MaxLRect
3.	85	43	68	120	54	7	150	46	19	145
4.	89	47	85	162	64	11	157	43	20	160
5.	90	38	75	169	68	6	161	41	20	131
6.	107	54	103	197	62	11	220	31	24	162
7.	104	52	101	197	64	10	208	31	24	161
8.	89	42	66	155	64	7	149	45	19	144
9.	86	37	66	150	59	7	133	50	18	127
10.	86	39	72	140	56	7	151	44	19	134
11.	100	51	104	201	67	10	201	32	23	158
12.	91	46	76	176	63	9	157	40	20	148
1. Cluster modes:										
2.	ScVarMaxis	ScVarmaxis	RaGyr	SkewMaxis	Skewmaxis	Kurtmaxis	KurtMaxis	HollRa	tipo	
3.	169	341	171	85	4	14	180	184	bus	
4.	173	354	186	75	1	9	183	195	van	
5.	184	388	137	71	1	6	200	204	bus	
6.	229	635	214	70	0	11	189	199	saab	
7.	228	706	218	74	0	6	187	197	opel	
8.	167	327	158	72	1	13	188	201	saab	
9.	159	246	145	64	1	7	186	183	van	
10.	170	339	139	80	0	21	183	187	opel	
11.	214	307	186	72	5	11	192	196	bus	
12.	181	373	151	67	0	10	192	198	opel	

```

1. cluster.results$cluster
2. 1          5
3. 2          6
4. 3          9
5. 4         10
6. 5          6
7. 6          3
8. 7          3
9. 8          7
10. 9         7
11. 10         4
12. 11         7
13. 12         3
14. 13         3
15. 14         6
16. 15         3
17. 16         9
18. 17         2
19. 18         9
20. 19         5
21. 20         5
22. 21         2
23. 22         7
24. 23         1
25. 24         9
26. 25         9
27. 26         6
28. 27         7
29. 28         4
30. 29         3
31. 30         6
32. 31         2
33. 32         6
34. 33         7
35. 34         5
36. 35         10
37. 36         10
38. 37         1
39. 38         8
40. 39         5
41. 40         8
42. 41         9
43. 42         7
44. 43         2
45. 44         10
46. 45         4
47. 46         2
48. 47         7
49. 48         1
50. 49         2
51. 50         10

```

4 REGRAS DE ASSOCIAÇÃO

```
inspect(dados[1:5])
```

```

      items
[1] {Afundo, Crucifixo, Gemeos, LegPress}
[2] {Agachamento, Gemeos, LegPress}
[3] {Afundo, Agachamento, Gemeos, LegPress}
[4] {Adutor, Agachamento, LegPress}
[5] {Afundo, Bicicleta, Gemeos, LegPress}

```

```

1. rules <- apriori(dados, parameter = list(supp = 0.3, conf = 0.75, target = "rules"))
2. inspect(rules)

```

	lhs	rhs	support	confidence	coverage	lift	count
[1]	{}	=> {LegPress}	0.8076923	0.8076923	1.0000000	1.000000	21
[2]	{Agachamento}	=> {LegPress}	0.3076923	1.0000000	0.3076923	1.238095	8
[3]	{Afundo}	=> {Gemeos}	0.3461538	1.0000000	0.3461538	1.529412	9
[4]	{AgachamentoSmith}	=> {Esteira}	0.3076923	0.8000000	0.3846154	1.733333	8
[5]	{AgachamentoSmith}	=> {Extensor}	0.3461538	0.9000000	0.3846154	1.800000	9
[6]	{AgachamentoSmith}	=> {Bicicleta}	0.3076923	0.8000000	0.3846154	1.485714	8
[7]	{Esteira}	=> {Extensor}	0.4230769	0.9166667	0.4615385	1.833333	11
[8]	{Extensor}	=> {Esteira}	0.4230769	0.8461538	0.5000000	1.833333	11
[9]	{Esteira}	=> {Bicicleta}	0.3846154	0.8333333	0.4615385	1.547619	10
[10]	{Extensor}	=> {Bicicleta}	0.4615385	0.9230769	0.5000000	1.714286	12
[11]	{Bicicleta}	=> {Extensor}	0.4615385	0.8571429	0.5384615	1.714286	12
[12]	{AgachamentoSmith, Extensor}	=> {Bicicleta}	0.3076923	0.8888889	0.3461538	1.650794	8
[13]	{AgachamentoSmith, Bicicleta}	=> {Extensor}	0.3076923	1.0000000	0.3076923	2.000000	8
[14]	{Esteira, Extensor}	=> {Bicicleta}	0.3846154	0.9090909	0.4230769	1.688312	10
[15]	{Bicicleta, Esteira}	=> {Extensor}	0.3846154	1.0000000	0.3846154	2.000000	10
[16]	{Bicicleta, Extensor}	=> {Esteira}	0.3846154	0.8333333	0.4615385	1.805556	10

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 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 CLASSIFICAÇÃO DE IMAGENS (CNN)

```
1. cifar10 = tf.keras.datasets.cifar10
2. (x_train, y_train), (x_test, y_test) = cifar10.load_data()
```

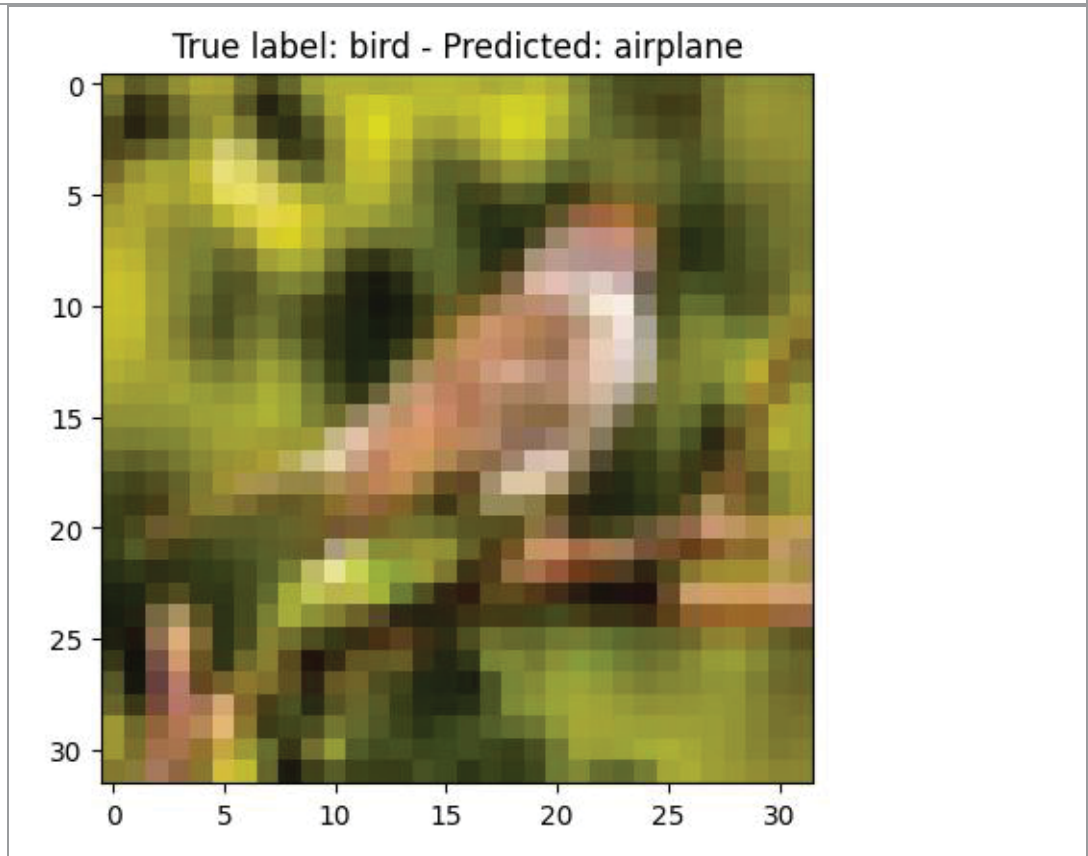
```
1. model.compile(optimizer="adam",
2.               loss="sparse_categorical_crossentropy", metrics=["accuracy"])
3. r = model.fit(x_train, y_train, validation_data= (x_test, y_test), epochs=15)
```

```
Epoch 1/15
1563/1563 ————— 16s 7ms/step - accuracy: 0.3566 - loss: 1.7506 -
val_accuracy: 0.5311 - val_loss: 1.3112
...
Epoch 15/15
1563/1563 ————— 4s 3ms/step - accuracy: 0.7697 - loss: 0.6443 -
val_accuracy: 0.7205 - val_loss: 0.8105
```

```

1. labels = ["airplane", "automobile", "bird", "cat", "deer", "dog", "frog", "horse",
"ship", "truck"]
2.
3. misclassified = np.where(y_pred!=y_test)[0]
4. i = np.random.choice(misclassified)
5.
6. plt.imshow(x_test[i], cmap="gray")
7. plt.title("True label: %s - Predicted: %s" % (labels[y_test[i]], labels[y_pred[i]]))

```



2 DETECTOR DE SPAM (RNN)

```

1. model.compile(loss="binary_crossentropy", optimizer="adam", metrics=["accuracy"])
2. epochs = 5
3. r = model.fit(data_train, y_train,
                 epochs= epochs, validation_data=(data_test, y_test))

```

```

Epoch 1/5
117/117 ————— 11s 71ms/step - accuracy: 0.8341 - loss: 0.5878 -
val_accuracy: 0.9233 - val_loss: 0.3001
Epoch 2/5
117/117 ————— 8s 69ms/step - accuracy: 0.9608 - loss: 0.2497 -
val_accuracy: 0.9668 - val_loss: 0.1772
Epoch 3/5
117/117 ————— 6s 55ms/step - accuracy: 0.9843 - loss: 0.1439 -
val_accuracy: 0.9821 - val_loss: 0.1188
Epoch 4/5
117/117 ————— 9s 76ms/step - accuracy: 0.9904 - loss: 0.0958 -
val_accuracy: 0.9859 - val_loss: 0.0924
Epoch 5/5
117/117 ————— 8s 56ms/step - accuracy: 0.9965 - loss: 0.0619 -
val_accuracy: 0.9859 - val_loss: 0.0773

```

```

1. texto = "Is your car dirty? Discover our new product. Free for all. Click the link."
2. seq_texto = tokenizer.texts_to_sequences([texto])
3. data_texto = pad_sequences(seq_texto, maxlen=T)
4.
5. pred = model.predict(data_texto)
6. print(pred)
7. print("SPAM" if pred >=0.5 else "OK")

```

```

1/1 ————— 0s 178ms/step
[[0.5046611]]
SPAM

```

3 GERADOR DE DÍGITOS FAKE (GAN)

```

1. model = tf.keras.Sequential()
2. model.add(layers.Dense(7*7*256, use_bias=False, input_shape=(100,)))
3. model.add(layers.BatchNormalization())
4. model.add(layers.LeakyReLU())
5.
6. model.add(layers.Reshape((7, 7, 256)))
7. assert model.output_shape == (None, 7, 7, 256)
8.
9. model.add(layers.Conv2DTranspose(128,(5,5), strides=1,padding='same',
    use_bias=False))
10. assert model.output_shape == (None, 7, 7, 128)
11.
12. model.add(layers.BatchNormalization())
13. model.add(layers.LeakyReLU())
14.
15. model.add(layers.Conv2DTranspose(64,(5,5), strides=(2,2),padding='same',
    use_bias=False))
16. assert model.output_shape == (None, 14, 14, 64)
17. model.add(layers.BatchNormalization())
18. model.add(layers.LeakyReLU())
19.
20. model.add(layers.Conv2DTranspose(1,(5,5), strides=(2,2),padding='same',
    use_bias=False, activation='tanh'))
21. assert model.output_shape == (None, 28, 28, 1)
22.
23. return model

```

```

1. def display_image(epoch_no):
2.     return PIL.Image.open('image_at_epoch_{:04d}.png'.format(epoch_no))
3.
4. display_image(EPOCHS)
5.
6. anim_file = 'dcgan.gif'
7.
8. with imageio.get_writer(anim_file, mode='I') as writer:
9.     filenames = glob.glob('image*.png')
10.    filenames = sorted(filenames)
11.
12.    for filename in filenames:
13.        image = imageio.imread(filename)
14.        writer.append_data(image)
15.    image = imageio.imread(filename)
16.    writer.append_data(image)
17.
18. import tensorflow_docs.vis.embed as embed
19. embed.embed_file(anim_file)

```



4 TRADUTOR DE TEXTOS (TRANSFORMER)

```

1. class Translator(tf.Module):
2.
3.     def __init__(self, tokenizers, transformer):
4.         self.tokenizers = tokenizers
5.         self.transformer = transformer
6.
7.     def __call__(self, sentence, max_length=20):
8.         assert isinstance(sentence, tf.Tensor)
9.
10.        if len(sentence.shape) == 0:
11.            sentence = sentence[tf.newaxis]
12.
13.        sentence = self.tokenizers.pt.tokenize(sentence).to_tensor()
14.        encoder_input = sentence
15.
16.        start_end = self.tokenizers.en.tokenize([''])[0]
17.        start = start_end[0][tf.newaxis]
18.        end = start_end[1][tf.newaxis]
19.
20.        output_array = tf.TensorArray(dtype=tf.int64, size=0, dynamic_size=True)
21.        output_array = output_array.write(0, start)
22.
23.        for i in tf.range(max_length):
24.            output = tf.transpose(output_array.stack())
25.            predictions, _ = self.transformer([encoder_input, output], training=False)
26.            predictions = predictions[:, -1:, :] # (batch_size, 1, vocab_size)
27.            predicted_id = tf.argmax(predictions, axis=-1)
28.            output_array = output_array.write(i+1, predicted_id[0])
29.
30.            if predicted_id == end:
31.                break
32.
33.        output = tf.transpose(output_array.stack())
34.        text = tokenizers.en.detokenize(output)[0]
35.        tokens = tokenizers.en.lookup(output)[0]
36.        _, attention_weights = self.transformer([encoder_input, output[:, :-1]],
37.                                                training=False)
38.        return text, tokens, attention_weights

```

```
1. translator = Translator(tokenizers, transformer)
2. sentence = "Eu li sobre triceratops na enciclopédia."
3. translated_text, translated_tokens, attention_weights =
translator(tf.constant(sentence))
4. print(f'{"Prediction":15s}: {translated_text}')
```

```
Prediction      : b'i read about tattoos in enclopedia .'
```

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

Resumo da implementação de Big Data no setor financeiro

Neste projeto, realizamos a migração de uma vasta quantidade de dados sigilosos (valor) de uma empresa do setor financeiro, anteriormente armazenados na AWS, para um ambiente on-premise, utilizando a plataforma Hortonworks com HBase e Hive, como fonte de dados. O objetivo principal era aumentar a segurança e reduzir custos, sem comprometer a performance e a integridade dos dados. O projeto envolveu a migração de dados estruturados, semiestruturados e não estruturados (Variedade), exigindo um planejamento cuidadoso para garantir a integridade, veracidade e disponibilidade dos dados durante e após a migração (Veracidade).

A empresa mantinha seus dados em um ambiente de nuvem na AWS, utilizando o S3 para armazenamento e o DynamoDB para gerenciar dados relacionais e não relacionais. No entanto, devido à natureza confidencial dos dados financeiros, foi necessário migrá-los para um ambiente on-premise, onde o controle e a segurança poderiam ser aprimorados. A escolha da plataforma Hortonworks, com HBase e Hive, foi feita devido à sua capacidade de lidar com grandes volumes de dados (Volume), variedade de formatos e a necessidade de manter a integridade dos dados.

A migração envolveu a transferência de dados de várias naturezas:

- i) **Dados Estruturados:** Informações organizadas em formatos tabulares, com relações bem definidas entre as entidades;
- ii) **Dados Semiestruturados:** Incluindo arquivos JSON e XML, que, embora possuam alguma estrutura, não são tão rigidamente organizados como os dados relacionais;
- iii) **Dados Não Estruturados:** Arquivos de texto, documentos e outras informações que não seguem uma estrutura pré-definida.

A complexidade da migração foi aumentada pela necessidade de adaptar modelos de dados relacionais para o HBase e Hive, que são mais orientados a colunas e não impõem restrições rígidas quanto à forma dos dados. Além disso, a volumetria foi um desafio significativo, uma vez que o banco de dados DynamoDB já continha uma vasta quantidade de registros, muitos dos quais eram dados sensíveis de clientes.

Os principais desafios enfrentados incluíram:

- i) **Segurança e Confidencialidade dos Dados:** A migração de dados sigilosos exigiu criptografia de ponta a ponta e verificações rigorosas de integridade para garantir que os dados não fossem corrompidos ou expostos durante o processo;
- ii) **Adaptação dos Modelos Relacionais:** A conversão de modelos relacionais para o HBase e Hive exigiu revisões de design e ajustes para garantir que a performance fosse mantida e que as consultas pudessem ser executadas eficientemente;
- iii) **Gerenciamento de Volume de Dados:** A volumetria foi um fator crítico, exigindo otimização dos processos de ingestão e indexação para garantir que o ambiente Hortonworks pudesse lidar com a carga sem comprometer a performance;
- iv) **Veracidade e Confiabilidade:** A precisão e a integridade dos dados eram fundamentais, especialmente em um ambiente financeiro. A implementação incluiu mecanismos de

verificação e validação para garantir que os dados migrados fossem consistentes e confiáveis;

- v) **Redução de Custos:** A migração para um ambiente on-premise resultou em economia de custos significativa, uma vez que o gerenciamento local de grandes volumes de dados era mais econômico a longo prazo do que manter esses dados na nuvem;
- vi) **Maior Segurança:** Com os dados mantidos em um ambiente controlado, a empresa pôde implementar medidas de segurança mais rigorosas, garantindo que os dados confidenciais dos clientes fossem protegidos contra acessos não autorizados;
- vii) **Manutenção da Performance:** Embora a velocidade de acesso aos dados não tenha sido drasticamente impactada pela migração, a implementação de um ambiente otimizado garantiu que a performance fosse mantida em níveis aceitáveis, mesmo com a complexidade e variedade dos dados (Velocidade);
- viii) **Flexibilidade no Gerenciamento de Dados:** A adoção do HBase e Hive permitiu à empresa lidar com diferentes tipos de dados de maneira eficiente, oferecendo flexibilidade para atender às necessidades do negócio.

A implementação de Big Data no ambiente on-premise da empresa do setor financeiro foi um sucesso, atendendo às exigências de segurança, redução de custos e integridade dos dados. A migração para a plataforma Hortonworks, com HBase e Hive, proporcionou à empresa uma solução robusta e escalável, capaz de gerenciar grandes volumes de dados variados, mantendo a performance e confiabilidade necessárias para suas operações.

Em um projeto de outro setor financeiro, os dados armazenados on-premise estão sendo migrados para a AWS, um processo denominado “democratização dos dados”. Essa democratização tem como objetivo criar um data lake para facilitar o compartilhamento de dados entre diversos setores da empresa.

O processo de democratização envolve 3 camadas:

- i) **SoR – Primeira Camada da Democratização dos Dados:** Nesta camada, as informações são idênticas aos dados da tabela on-premise, sem nenhuma adição, alteração ou remoção. Esta camada não pode ser compartilhada devido a possibilidade de conter dados sensíveis ou sigilosos.
- ii) **SoT – Segunda Camada da Democratização dos Dados:** Nesta camada, utilizam-se as informações da camada SoR, resultando em dados mais “polidos”. Alterações são feitas, se necessário, para tornar os dados mais “amigáveis”, facilitando no desenvolvimento da Spec e evitando retrabalho. Esta camada não pode ser compartilhada devido a possibilidade de conter dados sensíveis ou sigilosos.
- iii) **Spec – Terceira Camada da Democratização dos Dados:** Nesta camada, os dados são preparados especificamente para atender às necessidades do usuário final. Na empresa em questão, a área responsável solicita ao dono do dado uma tabela contendo as informações necessárias.

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:

- 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

Resultados obtidos com a Base de dados “Test_4cl amostra”

Desempenho dos Modelos com Características LBP					
Classe	Modelo	Acurácia	Sensibilidade	Especificidade	F1-Score
0	Random Forest	0.66	0.50	0.80	0.56
1	Random Forest	0.66	0.63	0.81	0.56
2	Random Forest	0.66	0.56	0.87	0.55
3	Random Forest	0.66	0.97	0.98	0.97
0	SVM	0.51	0.02	0.75	0.04
1	SVM	0.51	0.72	0.98	0.51
2	SVM	0.51	0.38	0.84	0.37
3	SVM	0.51	0.98	0.98	0.90
0	RNA	0.50	0.00	0.65	0.00
1	RNA	0.50	0.77	0.68	0.51
2	RNA	0.50	0.31	0.83	0.34
3	RNA	0.50	1.00	0.97	0.87

Desempenho dos Modelos com Características VGG16					
Classe	Modelo	Acurácia	Sensibilidade	Especificidade	F1-Score
0	SVM	0.91	0.97	0.92	0.93
1	SVM	0.91	0.91	0.88	0.90
2	SVM	0.91	0.84	0.83	0.90
3	SVM	0.91	0.92	0.97	0.93
0	Random Forest	0.87	0.95	0.85	0.91
1	Random Forest	0.87	0.83	0.88	0.88
2	Random Forest	0.87	0.81	0.85	0.82
3	Random Forest	0.87	0.86	0.93	0.84
0	RNA	0.89	0.94	0.93	0.89
1	RNA	0.89	0.79	0.88	0.82
2	RNA	0.89	0.90	0.96	0.91
3	RNA	0.89	0.93	0.96	0.95

Melhor Modelo por Característica

- LBP: O Random Forest apresentou uma Acurácia de 66% com um F1-Score elevado na classe 3 (0.97);
- VGG16: O SVM teve a melhor Acurácia de 91% com um F1-Score alto em todas as classes.

Conclusão

O modelo SVM com características VGG16 teve o melhor desempenho geral, superando o Random Forest e o RNA em termos de Acurácia e F1-Score, evidenciando sua eficácia na classificação.

2 REDES NEURAIAS

Resultados obtidos com a Base de dados: Test_4cl_amostra

Desempenho dos Modelos RESNET50 e VGG16					
Classe	Modelo	Acurácia	Sensibilidade	Especificidade	F1-Score
0	Resnet50 COM Data Augmentation	24.26%	0.059406	0.985185	0.112055
1	Resnet50 COM Data Augmentation	24.26%	0.022222	0.971530	0.043451
2	Resnet50 COM Data Augmentation	24.26%	0.155556	0.697509	0.254380
3	Resnet50 COM Data Augmentation	24.26%	0.944444	0.405694	0.567579
0	Resnet50 SEM Data Augmentation	81.13%	0.930693	1.000000	0.964103
1	Resnet50 SEM Data Augmentation	81.13%	0.700000	0.889680	0.783524
2	Resnet50 SEM Data Augmentation	81.13%	0.688889	0.882562	0.773791
3	Resnet50 SEM Data Augmentation	81.13%	0.933333	0.985765	0.958833
0	VGG16 COM Data Augmentation	44.47%	0.029703	1.000000	0.057692
1	VGG16 COM Data Augmentation	44.47%	0.000000	1.000000	0.000000
2	VGG16 COM Data Augmentation	44.47%	0.011111	1.000000	0.021978
3	VGG16 COM Data Augmentation	44.47%	1.000000	0.014235	0.028070
0	VGG16 SEM Data Augmentation	63.07%	0.811881	0.951852	0.876312
1	VGG16 SEM Data Augmentation	63.07%	0.466667	0.882562	0.610515
2	VGG16 SEM Data Augmentation	63.07%	0.466667	0.797153	0.588699
3	VGG16 SEM Data Augmentation	63.07%	0.833333	0.903915	0.867190

Conclusão

Com base nas métricas encontradas em todos os modelos treinados, o modelo ResNet50 SEM *Data Augmentation* é a melhor escolha. Ele combina uma alta acurácia com boas métricas de sensibilidade e F1-Score, sugerindo que ele generaliza bem e é eficaz em classificar corretamente as diferentes classes.

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

RESUMO

O uso de tecnologias de Inteligência Artificial (IA) como o ChatGPT levanta dilemas éticos importantes, incluindo a proteção da privacidade e segurança dos dados pessoais, a necessidade de transparência e consentimento informado, e a mitigação de vieses que perpetuam desigualdades sociais. A automação pode levar ao desemprego e aumentar as desigualdades econômicas, enquanto a falta de regulamentação clara dificulta a responsabilização por erros da IA. Para mitigar esses riscos, é essencial adotar políticas de educação, regulamentação, segurança tecnológica e uma abordagem coletiva entre desenvolvedores, governos e usuários. Somente assim pode-se maximizar os benefícios da IA e promover uma sociedade mais justa e equitativa.

Palavras-chave: Ética. Dilemas. Inteligência Artificial. ChatGPT.

CONSIDERAÇÕES FINAIS

O ChatGPT foi e vem sendo um ponto de *pivotagem* da era digital. A partir de sua criação, divulgação e disseminação de seu uso a sociedade tem que se readaptar para trilhar um novo rumo no qual a IA é agente produtor de conteúdo e meio de propagação de ideias e conhecimento e, vale lembrar, nem sempre confiáveis e éticos.

O uso do ChatGPT e de outras tecnologias de IA suscita dilemas éticos importantes, particularmente em relação à privacidade e à segurança de dados. Como precisam ser treinados com grandes volumes de dados, as IAs generativas podem capturar informações confidenciais, dados pessoais e sensíveis. Tais dados correm o risco de vazamento, por isso é fundamental que os desenvolvedores do ChatGPT sejam transparentes com os usuários, informando como seus dados são coletados e tratados e que não somente solicite o consentimento do compartilhamento dos dados, mas eduque e explique os riscos associados ao compartilhamento.

Outro ponto sensível é que os desenvolvedores do ChatGPT trabalhem com filtros e restrições para garantir que a resposta do modelo não seja enviesada, no sentido de evitar que se perpetuem estereótipos, preconceitos, injúrias, conteúdos extremistas e antiéticos.

A autonomia da IA é outra questão que gera debate, quanto mais autônoma a IA, maior a chance de manipulação de conteúdo, enviesamento, risco de graves erros e tendências perigosas. Por

isso, é fundamental que a equipe do ChatGPT esteja sempre à frente de decisões importantes e que tenha o controle da IA.

Entre outras questões tem-se uma provável onda de desemprego gerada pela substituição de pessoas pela IA em tarefas repetitivas. É preciso que a sociedade tenha plano e verba para alocar na capacitação desses profissionais em áreas “ímmunes” ao avanço do ChatGPT, principalmente no setor da saúde e de cuidados. Uma estratégia para mitigar o aumento da desigualdade social.

Para evitar danos do mal uso do ChatGPT e também dos problemas naturalmente atrelados ao seu surgimento e consolidação, várias abordagens podem ser adotadas, como por exemplo educação e conscientização, informando os usuários sobre os potenciais riscos e usos éticos da IA. Também é necessário atualizar e criar políticas e regulamentações: estabelecer e aplicar leis e diretrizes claras sobre o uso da IA. Tecnologia e segurança devem andar e evoluir juntas, é necessário desenvolver e implementar tecnologias de segurança que detectem e previnam usos indevidos do ChatGPT. Encorajar uma abordagem de responsabilidade coletiva onde desenvolvedores, governos, e usuários trabalhem juntos para assegurar o uso responsável da IA.

Em resumo, enquanto o ChatGPT oferece inúmeras oportunidades e benefícios, seu uso irresponsável ou mal-intencionado pode ter sérias consequências. A responsabilidade de garantir um uso ético e seguro recai sobre todos, os usuários, mas principalmente os stakeholders e desenvolvedores envolvidos.

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: Software engineering practices for machine learning - Adoption, effects, and team assessment

Qual o objetivo do estudo descrito pelo artigo?	Qual o problema/oportunidade/situação que levou à necessidade de realização desse estudo?
O artigo selecionado tem como objetivo principal a investigação sobre a forma como equipes de especialistas concebem, implementam e mantêm softwares com componentes de Machine Learning (ML) com foco particular na robustez e confiabilidade desses sistemas. De forma mais específica, busca disseminar, validar, organizar e sistematizar orientações práticas de engenharia para desenvolvimento de software que possuam componentes de ML. Além disso, pretende coletar e divulgar evidências empíricas sobre a importância das práticas de engenharia em organizações e outros domínios, e publicar um conjunto de dados abrangente que pode ser usado para desenvolver instrumentos que ajudem equipes na melhoria da qualidade de seus processos e métodos de engenharia.	O estudo traz o contexto atual marcado pela crescente adoção de Machine Learning (ML) em diversas aplicações. Isso exige técnicas de engenharia maduras para assegurar robustez no desenvolvimento, na implementação e na manutenção de softwares, evitando, assim, impactos negativos de ML nas pessoas, sociedade e ambiente. Por outro lado, o rápido avanço no campo de ML, com novas ferramentas, novos métodos e novos relatórios de experiência atualizados diariamente desafiam profissionais no sentido de difundir e selecionar métodos que sejam úteis e melhorem seus projetos.

(continua)

(continuação)

Qual a metodologia que os autores usaram para obter e analisar as informações do estudo?	Quais os principais resultados obtidos pelo estudo?
Os autores adotaram uma metodologia mista, com enfoque sequencial exploratório. Essa abordagem é composta por três etapas: Revisão Sistemática da Literatura (SLR, na sigla em inglês – <i>Systematic Literature Review</i>), Pesquisa em Larga Escala e Entrevistas de Validação. A primeira fase consiste em buscar e analisar documentos em fontes de informação, considerando tanto fontes acadêmicas (branca) como não acadêmicas (cinza); incluir ou excluir registros conforme critérios de relevância, autoridade, objetividade, idioma e data de publicação; Extração e análise dos dados conforme demografia, práticas de engenharia, efeitos e análise temática. Na fase dois, é elaborado um questionário, que será validado e distribuído utilizando a estratégia <i>snowball</i>. Com os dados coletados, é feita a análise, com atenção à estatística descritiva, à correlação e à importância das práticas. Na terceira e última etapa, ocorrem as entrevistas semiestruturadas, familiarização com os objetivos, seções de entrevista, amostragem proposital, perfil dos participantes, análise temática e temas emergentes.	Possivelmente, a principal contribuição do artigo é a elaboração de um catálogo abrangente de práticas de engenharia para Machine Learning (ML), desde a coleta até o monitoramento do sistema implantado. Outro resultado importante foi a validação de que as práticas de engenharia de software direcionadas a ML ainda é baixa quando comparada com as práticas tradicionais, o estudo indica que, embora times mais experientes possuam tendência a adotar mais práticas de engenharia de software para ML, normalmente, práticas de confiabilidades tendem a ser negligenciadas. Ainda, o estudo analisou a relação entre a adoção de práticas e os seus efeitos percebidos pelos profissionais. Os resultados indicam que a adoção dessas práticas auxilia na obtenção de efeitos desejáveis, como agilidade, qualidade, eficácia, rastreabilidade, reprodutibilidade e responsabilidade. Finalmente, com os dados obtidos no estudo, entre eles a taxa de adoção das práticas e os efeitos percebidos, há material disponível para que sejam desenvolvidos modelos de avaliação de qualidade de software com componentes de ML com maior maturidade.

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 DEEPPDREAM

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
- **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

B – RESOLUÇÃO

1 CLASSIFICAÇÃO (RNA)

```
1. i = tf.keras.layers.Input(shape=(28, 28))
2. x = tf.keras.layers.Flatten()(i)
3. x = tf.keras.layers.Dense(128, activation="relu")(x)
4. x = tf.keras.layers.Dropout(0.2)(x)
5. x = tf.keras.layers.Dense(10, activation="softmax")(x)
6.
7. model = tf.keras.models.Model(i, x)
```

```
1. model.compile(optimizer='adam',
2.               loss='sparse_categorical_crossentropy',
3.               metrics=['accuracy'])
```

```
1. r = model.fit(x_train, y_train, validation_data = (x_test, y_test), epochs=10)
```

```
1. Epoch 1/10
2. 1875/1875 _____ 9s 4ms/step -
   accuracy: 0.5306 - loss: 1.5471 - val_accuracy: 0.7438 - val_loss: 0.7421
...
19. Epoch 10/10
20. 1875/1875 _____ 10s 5ms/step -
    accuracy: 0.8383 - loss: 0.4534 - val_accuracy: 0.8338 - val_loss: 0.4632
```

```

1. misclassified = np.where(y_pred != y_test)[0]
2.
3. i = np.random.choice(misclassified)

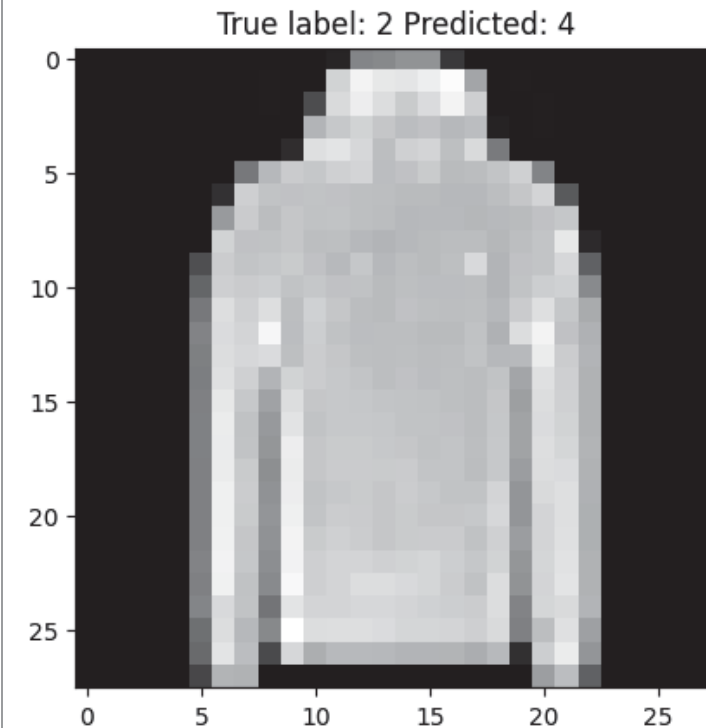
```

```

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

```

Text(0.5, 1.0, 'True label: 2 Predicted: 4')



2 REGRESSÃO (RNA).

```

1. scaler = StandardScaler()
2. X_train = scaler.fit_transform(X_train)
3. X_test = scaler.transform(X_test)

```

```

1. i = tf.keras.layers.Input(shape=(11,))
2. x = tf.keras.layers.Dense(16, activation="relu")(i)
3. x = tf.keras.layers.Dense(8)(x)
4. x = tf.keras.layers.Dense(6)(x)
5. x = tf.keras.layers.Dense(1)(x)
6.
7. model = tf.keras.models.Model(i, x)

```

```

1. r = model.fit(X_train, y_train,
2.               epochs=1500,
3.               batch_size=32,
4.               validation_data=(X_test, y_test),
5.               callbacks=[early_stop])

```

Epoch 1/1500

38/38 ————— 3s 13ms/step - loss: 13.0894 - r2: -21.6679 -
rmse: 2.9130 - val_loss: 1.5936 - val_r2: -1.4751 - val_rmse: 1.0191

...

Epoch 91/1500

38/38 ————— 0s 3ms/step - loss: 0.3302 - r2: 0.4584 -
rmse: 0.4518 - val_loss: 0.4411 - val_r2: 0.3193 - val_rmse: 0.5045

```

1. mse = mean_squared_error(y_test, y_pred)
2. rmse = sqrt(mse)
3. r2 = r2_score(y_test, y_pred)
4.
5. print("mse      = ", mse)
6. print("rmse     = ", rmse)
7. print("r2       = ", r2)

```

```

mse      = 0.4122875117521187
rmse     = 0.6420961857479911
r2       = 0.38747960329055786

```

Depois de executado o modelo, alcançou-se o valor de cerca de 0.41228 para o *Mean Squared Error* (MSE), que é a métrica utilizada para descrever o quão perto a reta de regressão está em relação ao conjunto de dados. Ainda, pode-se citar o Coeficiente de Determinação, que teve um valor aproximado de 0.38747. Esse valor indica que que, no conjunto de dados observado, há uma correlação leve em que as variáveis se movem-se na mesma direção crescente.

3 SISTEMAS DE RECOMENDAÇÃO

```

1. u = Input(shape=(1,))
2. u_emb = Embedding(N, K)(u)
3. u_emb = Flatten()(u_emb)
4.
5. m = Input(shape=(1,))
6. m_emb = Embedding(M, K)(m)
7. m_emb = Flatten()(m_emb)
8.
9. x = Concatenate()([u_emb, m_emb])
10.
11. x = Dense(1024, activation="relu")(x)
12. x = Dense(1)(x)
13.
14. model = Model(inputs=[u, m], outputs=x)
15.
16. model.compile(
17.     loss="mse",
18.     optimizer=SGD(learning_rate=0.08, momentum=0.9)
19. )
20.
21. user_ids, isbns, ratings = shuffle(df.new_ID_usuario, df.new_ISBN, df.new_notas)
22.
23. Ntrain = int(0.7 * len(ratings))
24.
25. train_users = user_ids[:Ntrain]
26. train_books = isbns[:Ntrain]
27. train_ratings = ratings[:Ntrain]
28. test_users = user_ids[Ntrain:]
29. test_books = isbns[Ntrain:]
30. test_ratings = ratings[Ntrain:]
31.
32. # centralizar as notas
33. avg_rating = train_ratings.mean()
34. train_ratings = train_ratings - avg_rating
35. test_ratings = test_ratings - avg_rating

```

```

1. epochs = 50
2. r = model.fit(
3.     x=[train_users, train_books],
4.     y=train_ratings,
5.     epochs=epochs,
6.     batch_size=1024,
7.     verbose=2,
8.     validation_data=([test_users, test_books], test_ratings)
9. )

```

```

Epoch 1/50
89/89 - 3s - 36ms/step - loss: 10.0144 - val_loss: 9.9412
...
Epoch 50/50
89/89 - 1s - 7ms/step - loss: 1.8809e-07 - val_loss: 10.5079

```

```

1. input_usuario = np.repeat(a=11089, repeats=M)
2. book = np.array(list(set(isbns)))
3.
4. preds = model.predict( [input_usuario, book] )
5.
6. rat = preds.flatten() + avg_rating
7.
8. idx = np.argmax(rat)
9.
10. print("Recomendação: Livro - ", book[idx], " / ", rat[idx] , "")

```

```

4028/4028 ----- 8s 2ms/step
Recomendação: Livro - 88700 / 10.694294 *

```

```

1. print(df.loc[[book[idx]]])

```

	ISBN	Titulo \
88700	451526201	Plunkitt of Tammany Hall: A Series of Very Pla...

	Autor	Ano	Editora	ID_usuario	Notas \
88700	William L. Riordon	1995	Signet Classics	19609	9

	new_ID_usuario	new_ISBN	new_notas
88700	7141	72537	9

```

1. input_usuario = np.repeat(a=50, repeats=M)
2. book = np.array(list(set(isbns)))
3.
4. preds = model.predict( [input_usuario, book] )
5.
6. rat = preds.flatten() + avg_rating
7.
8. idx = np.argmax(rat)
9.
10. print("Recomendação: Livro - ", book[idx], " / ", rat[idx] , "")

```

```

4028/4028 ----- 8s 2ms/step
Recomendação: Livro - 113079 / 10.522945 *

```

```

1. print(df.loc[[book[idx]]])

```

	ISBN	Titulo \
113079	1560441771	Where Dinosaurs Still Rule: A Guide to Dinosaur...

	Autor	Ano	Editora	ID_usuario	Notas	new_ID_usuario \
113079	Debbie Tewell	1993	Globe Pequot Pr	26216	3	9689

	new_ISBN	new_notas
113079	15779	3

4 DEEPDREAM

```
1. display.display(display.HTML('Image cc-by: <a
href=https://commons.wikimedia.org/wiki/File:Felis_catus-
cat_on_snow.jpg">Von.grzanka</a>'))
```



```
1. class DeepDream(tf.Module):
2.     def __init__(self, model):
3.         self.model = model
4.
5.     @tf.function(
6.         input_signature=(
7.             tf.TensorSpec(shape=[None, None, 3], dtype=tf.float32),
8.             tf.TensorSpec(shape=[], dtype=tf.int32),
9.             tf.TensorSpec(shape=[], dtype=tf.float32),)
10.    )
11.    def __call__(self, img, steps, step_size):
12.        print("Tracing")
13.        loss = tf.constant(0.0)
14.
15.        for n in tf.range(steps):
16.            with tf.GradientTape() as tape:
17.                # Gradientes relativos a img
18.                tape.watch(img)
19.                loss = calc_loss(img, self.model)
20.
21.                # Calculo do gradiente da perda em relação aos pixels da imagem de entrada.
22.                gradients = tape.gradient(loss, img)
23.
24.                # Normalizacao dos gradintes
25.                gradients /= tf.math.reduce_std(gradients) + 1e-8
26.
27.                # Na subida gradiente, a "perda" é maximizada.
28.                img = img + gradients*step_size
29.                img = tf.clip_by_value(img, -1, 1)
30.
31.        return loss, img
```

```
1. dream_img = run_deep_dream_simple(img=original_img, steps=100, step_size=0.01)
```



```
1. OCTAVE_SCALE = 1.60
2.
3. img = tf.constant(np.array(original_img))
4. base_shape = tf.shape(img)[: -1]
5. float_base_shape = tf.cast(base_shape, tf.float32)
6.
7. for n in range(-2, 3):
8.     new_shape = tf.cast(float_base_shape*(OCTAVE_SCALE**n), tf.int32)
9.     img = tf.image.resize(img, new_shape).numpy()
10.    img = run_deep_dream_simple(img=img, steps=50, step_size=0.01)
11.
12. display.clear_output(wait=True)
13. img = tf.image.resize(img, base_shape)
14. img = tf.image.convert_image_dtype(img/255.0, dtype=tf.uint8)
15. show(img)
```



APÊNDICE 14 – VISUALIZAÇÃO DE DADOS E STORYTELLING

A – ENUNCIADO

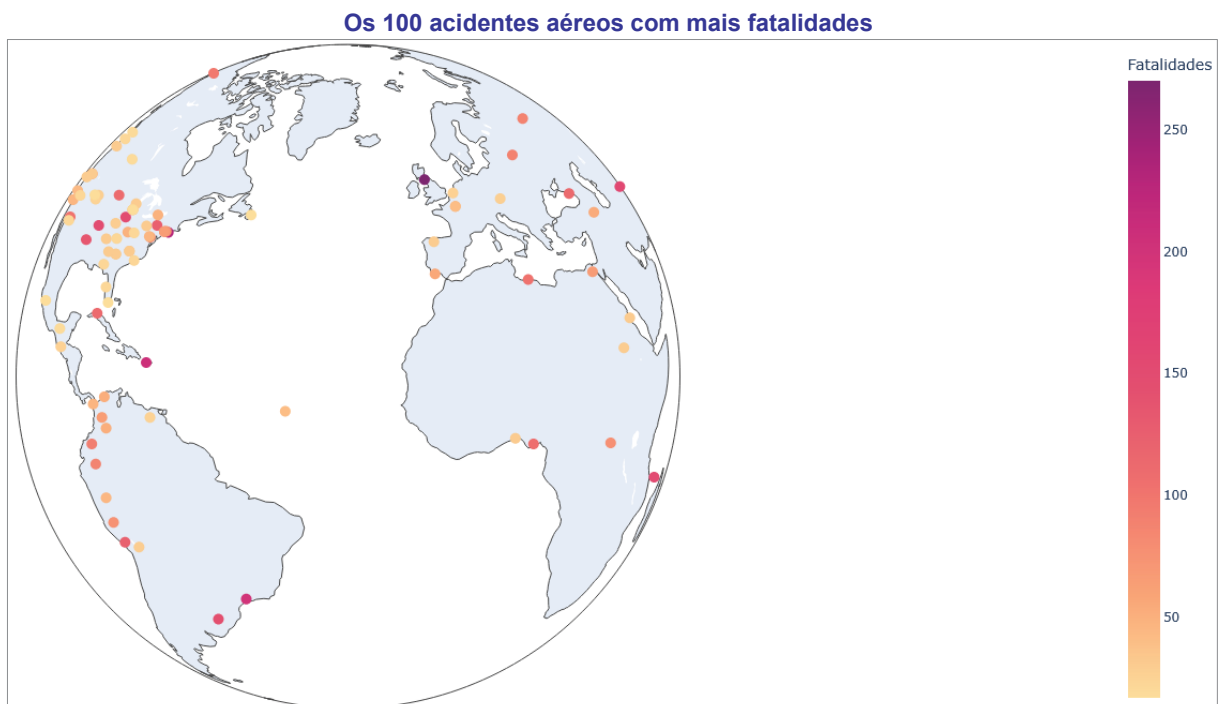
Escolha um conjunto de dados brutos (ou uma visualização de dados que você acredite que possa ser melhorada) e faça uma visualização desses dados (de acordo com os dados escolhidos e com a ferramenta de sua escolha)

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



Fonte: National Transportation Safety Board (Conselho Nacional de Segurança nos Transportes) - NTSB

Primeira Classe: Uma viagem pela história da segurança na aviação

Nossa viagem começa nos céus, onde diariamente há milhares de voos, conectando pessoas ao redor do mundo. No entanto, em meio aos voos habituais, às vezes, acontece um desastre. O conjunto de dados de acidentes de aviação coletados pela *National Transportation Safety Board* (Conselho Nacional de Segurança nos Transportes - NTSB) é o meio pelo qual podemos viajar ao passado e testemunhar parte da história da aviação e da sua segurança, bem como as lições aprendidas com os infortúnios do passado.

Estamos na década de 1960, nessa época, a aviação era, relativamente, jovem e crescia rapidamente. O mundo estava fascinado com as velocidades, as distâncias e a facilidade que os aviões proporcionavam. Mas, à medida que os céus se enchiam de mais e mais aviões, os acidentes se tornavam uma parte muito triste de todo esse desenvolvimento. A base capta bem esses primeiros anos - muitos dos acidentes resultaram de falhas mecânicas, condições climáticas ou pilotos inexperientes. Estes incidentes foram desoladores, porém cruciais para o desenvolvimento da segurança da aviação moderna.

No período entre as décadas de 1970 e 1990, houve um grande salto na tecnologia da aviação. Os motores a jato tornaram-se mais confiáveis e baratos, os materiais ficaram mais leves e resistentes e a tecnologia da cabine de comando avançou, agora, sendo povoada por equipamentos automatizados. Ainda assim, os fatores humanos e erros de comunicação continuaram sendo a causa da maioria dos acidentes. Especificamente para o conjunto de dados escolhido, uma análise aprofundada leva à descoberta de seus pontos fracos, o que, por sua vez, forçou uma revisão no modo como os treinamentos de pilotos eram feitos, novos protocolos de voo e até mesmo reformulação dos sistemas de controle de tráfego aéreo.

A década de 2000 marca uma era crucial, na qual a colaboração internacional floresceu. A NTSB faz uma investigação incansável de cada acidente, descobrindo as causas e oferecendo recomendações que moldariam o futuro da aviação. Nesse sentido, mais que saber as causas de um acidente, é importante que as causas sejam identificadas e corrigidas, evitando, assim, novos acidentes com a mesma causa. Também nessa época, os regulamentos tornaram-se mais rigorosos e inovações nas tecnologias de segurança da aviação avançaram muito em um pequeno período. A monitorização dos dados de voo, as caixas pretas e os sistemas de comunicação em tempo real entre os aviões e as torres de controle foram introduzidos - cada um desses avanços está registrado neste conjunto de dados.

Ao longo dos anos, os dados revelaram uma redução significativa no número de acidentes, que, por sua vez, é resultado do aprendizado e do avanço tecnológico nas últimas décadas. O valor do desenvolvimento da cultura de segurança, do aprimoramento do treinamento de pilotos, do uso da automação e da tomada de decisão com base em dados é evidente. Os registros revelam menos acidentes, mas também põe em evidência a natureza ainda em evolução da aviação.

O conjunto de dados permite que as autoridades do setor de aviação investiguem os projetos das aeronaves e a influência da humanidade no meio ambiente, bem como os limites da tecnologia pelo prisma do componente humano. Cada pequena diferença nos números, nas linhas e até mesmo nas colunas se soma a uma narrativa, como o grande esforço humano para tornar os céus mais seguros para passageiros e tripulantes.

Sob a perspectiva da análise dos dados, as informações abrigadas pelo conjunto podem ser usadas para explorar padrões como:

- i. As tendências ao longo do tempo: Como o número de acidentes diminuiu devido ao aprimoramento constante das medidas de segurança;
- ii. Os principais motivos: Elementos cruciais responsáveis pelas quedas como condições climáticas, falhas mecânicas ou erros humanos mais comuns ao longo dos anos. Nota-se que um acidente pode ter não apenas uma causa;
- iii. Análise do tipo de aeronave e do fabricante: Determinados tipos ou fabricantes de aeronaves causaram mais acidentes? Por vezes, aeronaves de um determinado modelo ou fabricante pode ter algum defeito silencioso. Isso pode ser um sinal de preocupações com a segurança de um modelo em específico;
- iv. Aumento da segurança: É possível identificar momentos marcantes no curso da história, quando os avanços tecnológicos ou as mudanças regulatórias causaram grandes saltos no setor da aviação;
- v. As distribuições dos registros em certas áreas geográficas: Em algumas regiões, pode haver mais incidentes desse tipo, devido ao controle inadequado do tráfego, tipo de terreno, clima ou à não conformidade com relação a normas de aviação. Este é o item central da visualização proposta.

Ao analisar os dados, você descobrirá não apenas o lado triste da história, mas, mais importante, as conquistas que tornaram o voo seguro como ele é hoje. Todos os registros são apenas peças no grande mosaico do progresso, que ainda está se desenrolando conforme a indústria da aviação busca incansavelmente tornar os céus mais seguros.

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);
- (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

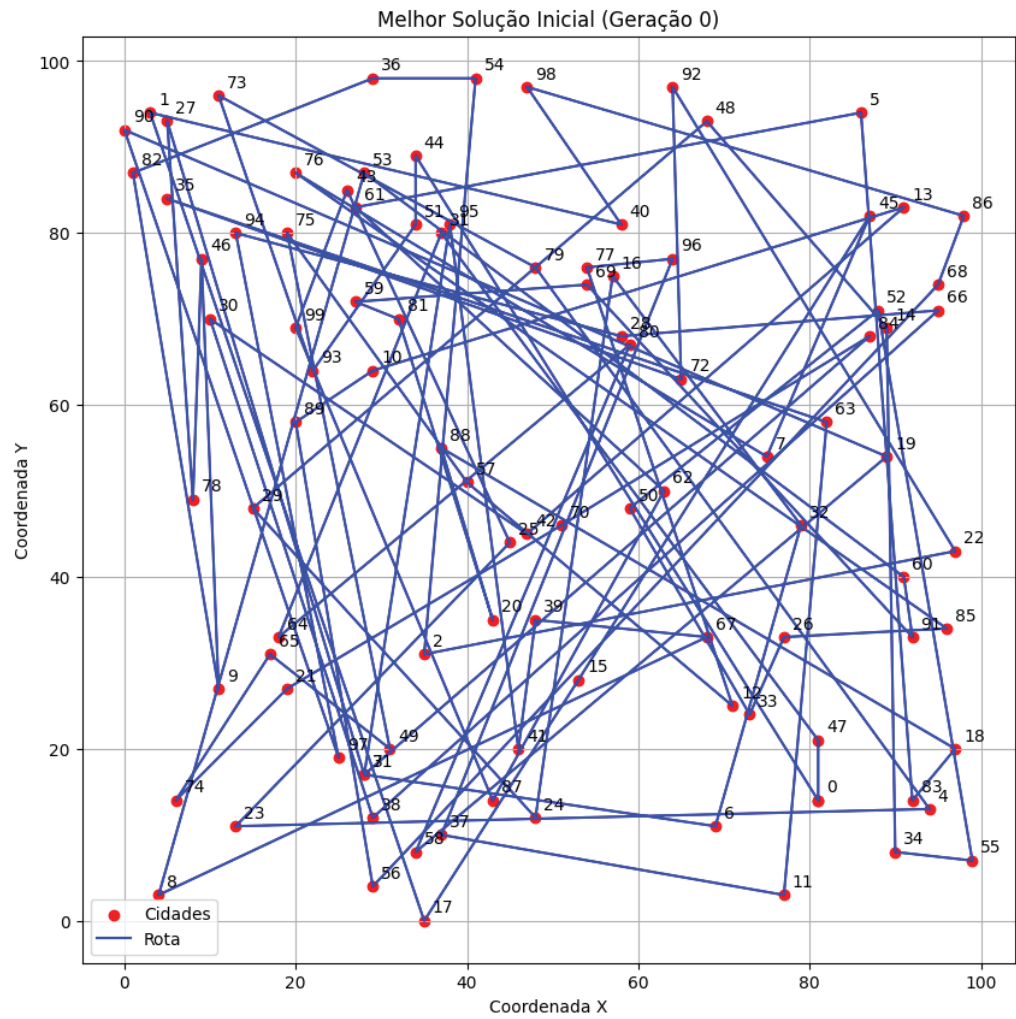
```

1. def executar_algoritmo_genetico(num_geracoes, tamanho_populacao,
                                   taxa_mutacao, taxa_crossover):
2.     """Executa o algoritmo genético para o TSP."""
3.     populacao = criar_populacao_inicial(tamanho_populacao)
4.     melhor_historico = []
5.
6.     populacao_avaliada = avaliar_populacao(populacao)
7.     melhor_inicial = min(populacao_avaliada, key=lambda x: x[0])
8.     melhor_historico.append(melhor_inicial)
9.     print(f"Melhor solução inicial (custo: {melhor_inicial[0]:.2f}):
           {melhor_inicial[1]}")
10.    plotar_rota(melhor_inicial[1], f"Melhor Solução Inicial (Geração 0)")
11.
12.    for geracao in range(num_geracoes):
13.        nova_populacao = []
14.        populacao_avaliada.sort(key=lambda x: x[0])
15.
16.        nova_populacao.append(melhor_historico[-1][1])
17.
18.        while len(nova_populacao) < tamanho_populacao:
19.            if random.random() < taxa_crossover:
20.                pai1, pai2 = selecionar_pais(populacao_avaliada)
21.                filho = crossover_ox(pai1, pai2)
22.                if random.random() < taxa_mutacao:
23.                    filho = mutacao(filho)
24.                nova_populacao.append(filho)
25.            else:
26.                pai1, _ = selecionar_pais(populacao_avaliada)
27.                individuo = pai1
28.                if random.random() < taxa_mutacao:
29.                    individuo = mutacao(individuo)
30.                nova_populacao.append(individuo)
31.
32.        populacao = nova_populacao
33.        populacao_avaliada = avaliar_populacao(populacao)
34.        melhor_geracao = min(populacao_avaliada, key=lambda x: x[0])
35.        melhor_historico.append(melhor_geracao)
36.
37.    melhor_final = min(melhor_historico, key=lambda x: x[0])
38.    print(f"\nMelhor solução final encontrada (custo: {melhor_final[0]:.2f}):
           {melhor_final[1]}") # Print cost and route
39.    plotar_rota(melhor_final[1], f"Melhor Solução Final (Geração {num_geracoes})")
40.
41.    return melhor_final

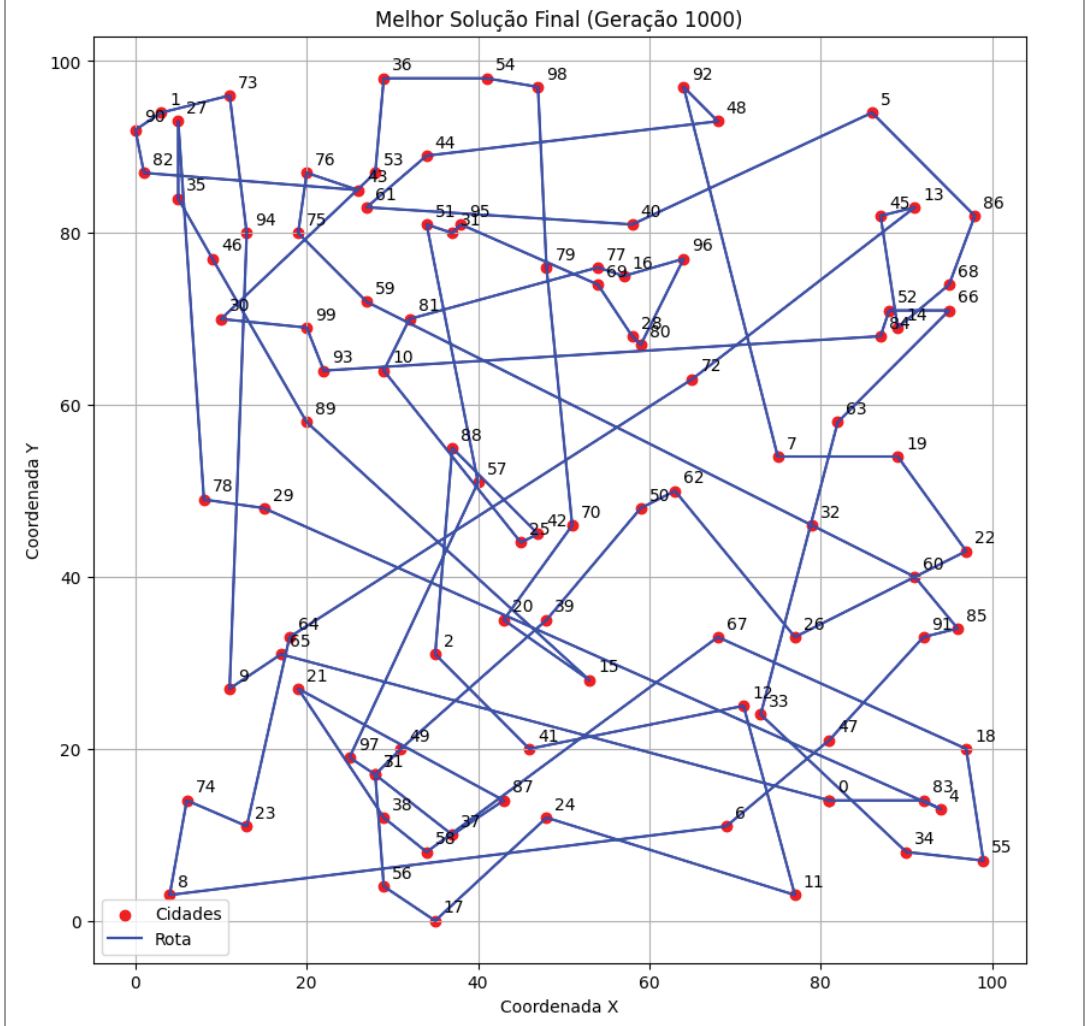
```

```
1. melhor_solucao = executar_algoritmo_genetico(num_geracoes, tamanho_populacao,
                                              taxa_mutacao, taxa_crossover)
```

Melhor solução inicial (custo: 4836.21): [0, 44, 51, 93, 73, 79, 4, 23, 25, 43, 99, 87, 62, 61, 5, 83, 18, 88, 20, 81, 59, 69, 91, 60, 53, 8, 67, 39, 41, 95, 3, 27, 78, 46, 9, 82, 36, 54, 2, 22, 92, 72, 35, 63, 11, 37, 96, 77, 12, 57, 75, 49, 65, 74, 21, 70, 84, 71, 6, 32, 16, 24, 29, 48, 14, 34, 55, 52, 50, 33, 26, 85, 76, 7, 45, 15, 17, 1, 40, 98, 86, 68, 56, 89, 10, 13, 42, 30, 38, 66, 28, 94, 97, 90, 19, 58, 80, 64, 31, 47]



Melhor solução final encontrada (custo: 1895.59): [0, 83, 4, 29, 78, 27, 35, 46, 89, 15, 20, 70, 79, 98, 54, 36, 53, 30, 99, 93, 84, 52, 66, 63, 33, 34, 55, 18, 67, 58, 38, 21, 87, 37, 71, 97, 57, 51, 31, 95, 69, 28, 80, 96, 16, 77, 81, 10, 25, 42, 88, 2, 41, 12, 11, 24, 17, 56, 3, 49, 39, 50, 62, 26, 22, 19, 7, 92, 48, 44, 61, 40, 5, 86, 68, 14, 45, 13, 72, 64, 23, 74, 8, 6, 47, 91, 85, 60, 32, 59, 75, 76, 43, 82, 90, 1, 73, 94, 9, 65]



2 COMPARAÇÃO DA REPRESENTAÇÃO DE DOIS MODELOS VETORIAIS

```
1. texto = [
2.     ["gato", "caça", "rato"],
3.     ["cachorro", "late", "gato"],
4.     ["rato", "foge", "gato"],
5.     ["gato", "esconde", "cachorro"],
6.     ["cachorro", "melhor", "amigo", "homem"],
7.     ["rato", "come", "queijo"],
8.     ["cachorro", "come", "ração"],
9.     ["ninguém", "sabe", "gato", "come"]
10. ]
```

```
1. modelo = Word2Vec(sentences=texto, vector_size=100, window=5, min_count=1, workers=4)
```

```
1. pca = PCA(n_components=2)
2. resultado = pca.fit_transform(vetores)
```

```

1. plt.figure(figsize=(10, 6))
2. plt.scatter(resultado[:, 0], resultado[:, 1])
3.
4. for i, x in enumerate(palavras):
5.     plt.annotate(x, xy=(resultado[i, 0], resultado[i, 1]))
6.
7. plt.title('Visualização de Vetores de Palavras com PCA')
8. plt.grid(True)
9. plt.show()

```

