

UNIVERSIDADE FEDERAL DO PARANÁ

LUIZ HENRIQUE DOS SANTOS CRUZ

MEMORIAL DE PROJETOS: PROPOSTA DE SOLUÇÃO DE *RETRIEVAL-
AUGMENTED GENERATION* (RAG) PARA CONSULTA DE ACERVOS
JORNALÍSTICOS DIGITAIS

CURITIBA

2025

LUIZ HENRIQUE DOS SANTOS CRUZ

MEMORIAL DE PROJETOS: PROPOSTA DE SOLUÇÃO DE *RETRIEVAL-
AUGMENTED GENERATION* (RAG) PARA CONSULTA DE ACERVOS
JORNALÍSTICOS DIGITAIS

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



MINISTÉRIO DA EDUCAÇÃO
SETOR DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PÓS-GRADUAÇÃO
CURSO DE PÓS-GRADUAÇÃO INTELIGÊNCIA ARTIFICIAL
APLICADA - 40001016399E1

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 **LUIZ HENRIQUE DOS SANTOS CRUZ**, intitulada: **MEMORIAL DE PROJETOS: PROPOSTA DE SOLUÇÃO DE RETRIEVAL-AUGMENTED GENERATION (RAG) PARA CONSULTA DE ACERVOS JORNALÍSTICOS DIGITAIS**, 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

JAIME WOJCIECHOWSKI
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

RESUMO

Este memorial apresenta uma proposta de *Retrieval-Augmented Generation* (RAG) para otimizar a consulta a acervos jornalísticos digitais, enfrentando o desafio de filtrar conteúdo preciso e contextualmente relevante em grandes volumes de texto. O trabalho detalha o desenvolvimento de um protótipo funcional, construído com tecnologias de código aberto em etapas que incluem raspagem de dados, limpeza, vetorização semântica e geração de respostas com um Grande Modelo de Linguagem (LLM) local. O resultado é uma arquitetura acessível e replicável que aprimora a apuração jornalística. A discussão final aborda as limitações do modelo, especialmente em escalabilidade e na fidelidade do raciocínio causal, um requisito essencial para a integridade da informação no jornalismo.

Palavras-chave: Retrieval-Augmented Generation (RAG); Inteligência Artificial Aplicada; Grandes Modelos de Linguagem (LLMs); Embeddings; Busca Vetorial.

ABSTRACT

This abstract presents a proposal for Retrieval-Augmented Generation (RAG) to optimize querying in digital journalistic archives, addressing the challenge of filtering precise and contextually relevant content from large volumes of text. The work details the development of a functional prototype, built with open-source technologies in stages that include data scraping, cleaning, semantic vectorization, and response generation with a local Large Language Model (LLM). The result is an accessible and replicable architecture that enhances journalistic investigation. The final discussion addresses the model's limitations, especially in scalability and the fidelity of causal reasoning, an essential requirement for the integrity of information in journalism.

Keywords: Retrieval-Augmented Generation (RAG); Applied Artificial Intelligence; Large Language Models (LLMs); Embeddings; Vector Search.

LISTA DE FIGURAS

FIGURA 1 – Visão geral do processo da solução	15
FIGURA 2 - Exercício de Heurística	28
FIGURA 3 - Exercício de Lógica	29
FIGURA 4 - Exercício de Redes Neurais	29
FIGURA 5 - Estrutura dos dados de carros	35
FIGURA 6 - Descrição variáveis numéricas	35
FIGURA 7 - Descrição variáveis categóricas	36
FIGURA 8 - Contagem de carros por marca	36
FIGURA 9 - Preço médio de carros por ano.....	38
FIGURA 10 - Transformação mês para numérico	38
FIGURA 11 - Preço médio por engrenagem	39
FIGURA 12 - Preço médio por tipo de combustível	40
FIGURA 13 - Ajuste das variáveis engrenagem e combustível	42
FIGURA 14 - Remoção da variável alvo.....	43
FIGURA 15 - Dados de treinamento	43
FIGURA 16 - Dados de teste	43
FIGURA 17 - Treinamento Random Forest	44
FIGURA 18 - Importância das variáveis	44
FIGURA 19 - Treinamento Random Forest com parâmetros ajustados	45
FIGURA 20 - Importância das variáveis	45
FIGURA 21 - Treinamento XGBoost	46
FIGURA 22 - Importância das variáveis	46
FIGURA 23 - Matriz de confusão para Random Forest	50
FIGURA 24 - Matriz de confusão para SVM.....	51
FIGURA 25 - Matriz de confusão para RNA.....	52
FIGURA 26 - Avaliação dos modelos RF, SVM, RNA e Alométrico de Spurr	53
FIGURA 27 - Atribuição de características	70
FIGURA 28 - F1 Score e Matriz de confusão para SVM	71
FIGURA 29 - Estrutura da base de saúde materna	72
FIGURA 30 – Avaliação de risco.....	72
FIGURA 31 - Risco percentual.....	72
FIGURA 32 - Checagem de nulos por coluna	72

FIGURA 33 - Estatísticas das variáveis.....	73
FIGURA 34 - Separação do alvo para treinamento	73
FIGURA 35 - Separação entre teste e treinamento	74
FIGURA 36 - Matriz de confusão do SVM.....	74
FIGURA 37 - Matriz de confusão do novo treinamento	75
FIGURA 38 - Matriz de confusão com hiper parâmetros	76
FIGURA 39 - Matriz de confusão para SVC	76
FIGURA 40 - Predição de novos casos.....	78
FIGURA 41 - Lista de comandos.....	78
FIGURA 42 - Aplicação de algoritmos de classificação em dados de Diabetes	79
FIGURA 43 - Predição de novos casos.....	79
FIGURA 44 - Lista de comandos algoritmo escolhido	80
FIGURA 45 - Aplicação de algoritmos de predição de regressão	81
FIGURA 46 - Predição de novos casos.....	81
FIGURA 47 - Lista de comandos do algoritmo escolhido	82
FIGURA 48 - Lista de comandos do algoritmo escolhido (continuação).....	83
FIGURA 49 - Aplicação de algoritmos de regressão em Biomassa.....	84
FIGURA 50 - Predição de novos casos.....	84
FIGURA 51 - Lista de comandos do algoritmo escolhido	85
FIGURA 52 - Lista de comandos do algoritmo escolhido (continuação).....	86
FIGURA 53 - Visão dos dados de agrupamento	87
FIGURA 54 - Lista de comandos do algoritmo escolhido	87
FIGURA 55 - Suporte e confiança dos dados de Musculação.....	87
FIGURA 56 - Lista de comandos do algoritmo selecionado	88
FIGURA 57 - Download da base CIFAR10	89
FIGURA 58 - Separação em treino e teste.....	89
FIGURA 59 - Camadas da CNN.....	90
FIGURA 60 - Execução de treinamento em 15 épocas	90
FIGURA 61 - Exemplo de predição errada.....	92
FIGURA 62 - Download de base de SPAM	92
FIGURA 63 - Separação de treinamento e teste SPAM	92
FIGURA 64 - Modelo RNN com LSTM.....	93
FIGURA 65 - Treinamento RNN com 5 épocas.....	93
FIGURA 66 - Predição de SPAM com RNN	94

FIGURA 67 - Download base MNIST	94
FIGURA 68 - Geração de imagem sem treino	95
FIGURA 69 - Dados discriminador do GAN	95
FIGURA 70 - Resultado do GAN para 100 épocas de treinamento	95
FIGURA 71 - GIF combinado	96
FIGURA 72 - Download transcrição palestras TED	96
FIGURA 73 - Amostra da base TED	96
FIGURA 74 - Variação de acordo com dimensão de <i>embedding</i>	97
FIGURA 75 - Modelo de <i>Transformer</i>	97
FIGURA 76 - Treinamento do Transformer com 20 épocas	98
FIGURA 77 - Tradução com o Transformer treinado	98
FIGURA 78 - Importação bibliotecas	106
FIGURA 79 - Base de dados Warwick e separação em treino e teste	107
FIGURA 80 - Carregando imagens com OpenCV	107
FIGURA 81 – Amostra da base Warrick	108
FIGURA 82 - Separação em teste e treinamento	109
FIGURA 83 - Características da base separada	109
FIGURA 84 - Extração de características das imagens	109
FIGURA 85 - Treinamento com modelos RF, SVM e RNA	111
FIGURA 86 - Aplicação das funções de extração	111
FIGURA 87 - Avaliação dos modelos	111
FIGURA 88 - Resultados dos modelos	111
FIGURA 89 - Seleção do melhor modelo	112
FIGURA 90 - Importação de bibliotecas para redes neurais	112
FIGURA 91 - Redefinição das imagens	113
FIGURA 92 - Forma das imagens de treino, teste e validação	113
FIGURA 93- Código do treinamento da rede neural VGG16 com Resnet50	113
FIGURA 94 - Treinamento em 10 épocas	116
FIGURA 95 - Avaliação dos modelos com matriz de confusão	117
FIGURA 96 - Resultado da avaliação	117
FIGURA 97 - Importação das bibliotecas	137
FIGURA 98 - Download base Fashion MNIST e separação de treinamento e teste	138
FIGURA 99 - Características do conjunto de dados	138

FIGURA 100 - Normalização dos pixels	139
FIGURA 101 - Criação do modelo sequencial.....	139
FIGURA 102 - Compilação do modelo com otimizado com Adam.....	139
FIGURA 103 - Treinamento do modelo em 10 épocas	140
FIGURA 104 - Avaliação do resultado	140
FIGURA 105 - Acuaría e perda do modelo	141
FIGURA 106 - Teste de predição	141
FIGURA 107 - Avaliação do modelo com matriz de confusão	142
FIGURA 108 - Demonstração da classificação	143
FIGURA 109 - Rótulos de identificação.....	143
FIGURA 110 - Importação das bibliotecas	144
FIGURA 111 - Download base de dados de Vinhos.....	144
FIGURA 112 - Tradução dos títulos das colunas	144
FIGURA 113 - Descrição dos dados	145
FIGURA 114 - Remoção da primeira coluna	146
FIGURA 115 - Separação em teste e treinamento	146
FIGURA 116 - Definição do modelo	146
FIGURA 117 - Funções de avaliação do modelo	146
FIGURA 118 - Otimização do modelo com Adam	147
FIGURA 119 - Treinamento do modelo.....	147
FIGURA 120 - Treinamento do modelo.....	147
FIGURA 121 - Avaliação do resultado do treinamento	148
FIGURA 122 - Erro quadrático médio e coeficiente de determinação	148
FIGURA 123 - Predição do modelo.....	150
FIGURA 124 - Avaliação do modelo	151
FIGURA 125 - Importando as bibliotecas	152
FIGURA 126 - Upload da base via Google Colab	152
FIGURA 127 - Base de dados carregada.....	152
FIGURA 128 - <i>Embeddings</i> do modelo	153
FIGURA 129 - Definição do modelo	153
FIGURA 130 - Compilando o modelo com otimizador SGD	153
FIGURA 131 - Separação em treino e teste.....	154
FIGURA 132 - Treinamento modelo.....	154
FIGURA 133 - Treinamento em 25 épocas	155

FIGURA 134 - Gerando gráfico de perda do treinamento	155
FIGURA 135 - Teste de recomendação do modelo treinado	156
FIGURA 136 - Importação de bibliotecas para daydream	157
FIGURA 137 – Definindo imagem base	157
FIGURA 138 - Ajuste da imagem para RNN	157
FIGURA 139 - Download da imagem base	157
FIGURA 140 - Baixando modelo RNN treinado.....	158
FIGURA 141 - Criação do modelo e ativando camadas	158
FIGURA 142 - Calculo da perda.....	158
FIGURA 143 - Classe Deepdream	159
FIGURA 144 - Execução do Deep Dream	159
FIGURA 145 - Resultado da execução do Deep Dream	160
FIGURA 146 - Aplicação de escala de uma oitava.....	161
FIGURA 147 - Importando bibliotecas.....	168
FIGURA 148 - Parâmetros problema caixeiro viajante	168
FIGURA 149 – Funções de apoio para algoritmo genético	169
FIGURA 150 - Execução do algoritmo genético	169
FIGURA 151 - Primeira melhor solução da população inicial	170
FIGURA 152 - Melhor solução após 1000 gerações	170
FIGURA 153 - Importando bibliotecas.....	171
FIGURA 154 - Textos base de comparação vetorial	171
FIGURA 155- Frequência de palavras usando PCA	171
FIGURA 156 - Calculo matriz de similaridade por cosseno	172
FIGURA 157 - Avaliando similaridades	173
FIGURA 158 - Criando o modelo por <i>embeddings</i> e gerando gráfico.....	173
FIGURA 159 - Matriz de similaridade por cosseno.....	174
FIGURA 160 - Avaliação dos resultados similares	174
FIGURA 161 – Chat do RAG AI	178

LISTA DE GRÁFICOS

GRÁFICO 1 - Distribuição de carros por marca	37
GRÁFICO 2 - Distribuição de carros por engrenagem	37
GRÁFICO 3 - Representação evolução preços dos carros	39
GRÁFICO 4 - Distribuição por marca e engrenagem	40
GRÁFICO 5 - Distribuição média de preços por marca e combustível	41
GRÁFICO 6 - Análise da variável alvo	41
GRÁFICO 7 - Gráfico correlação das variáveis numéricas.....	42
GRÁFICO 8 - Idade média das esposas	55
GRÁFICO 9 - Histograma salários das esposas	55
GRÁFICO 10 - Idade média dos maridos	56
GRÁFICO 11 - Histograma dos salários dos maridos	56
GRÁFICO 12 - Teste de hipótese das medianas	60
GRÁFICO 13 - Gráfico de resíduos do modelo RNA.....	82
GRÁFICO 14 - Resíduos do modelo Random Forest.....	85
GRÁFICO 15 - Perda e acurácia do treinamento CNN.....	91
GRÁFICO 16 - Matriz de confusão pós treinamento	91
GRÁFICO 17 - Gráfico de perda e acuraria durante treinamento RNN	94
GRÁFICO 18 – Erro quadrático médio e perda do modelo	149
GRÁFICO 19 - Coeficiente de determinação do modelo	149
GRÁFICO 20 - Gráfico de perda durante o treinamento.....	156
GRÁFICO 21 - Mapa de calor interativo de empresas de IA	163
GRÁFICO 22 - Distribuição de empresas de IA por localidade	163
GRÁFICO 23 - Distribuição de salário por senioridade	164
GRÁFICO 24 - Histograma de salários em dólar.....	164
GRÁFICO 25 - Representação vetorial por frequência de palavras (PCA).....	172
GRÁFICO 26 - Representação vetorial por <i>embedding</i>	173

LISTA DE TABELAS

TABELA 1 - Erro quadrático médio e R^2	63
TABELA 2 - Erro quadrático médio e R^2	64
TABELA 3 - Alfa e lambda para Lasso e Ridge	65
TABELA 4 - Erro quadrático médio e R^2 para Elastic Net.....	68
TABELA 5 - Comparativo entre modelos.....	68
TABELA 6 - Aplicação dos algoritmos de classificação em veículos	77

SUMÁRIO

1 PARECER TÉCNICO	14
APÊNDICE 1 – INTRODUÇÃO À INTELIGÊNCIA ARTIFICIAL	23
APÊNDICE 2 – LINGUAGEM DE PROGRAMAÇÃO APLICADA	31
APÊNDICE 3 – LINGUAGEM R	47
APÊNDICE 4 – ESTATÍSTICA APLICADA I	54
APÊNDICE 5 – ESTATÍSTICA APLICADA II	61
APÊNDICE 6 – ARQUITETURA DE DADOS	69
APÊNDICE 7 – APRENDIZADO DE MÁQUINA.....	77
APÊNDICE 8 – DEEP LEARNING	89
APÊNDICE 9 – BIG DATA	99
APÊNDICE 10 – VISÃO COMPUTACIONAL	104
APÊNDICE 11 – ASPECTOS FILOSÓFICOS E ÉTICOS DA IA	119
APÊNDICE 12 – GESTÃO DE PROJETOS DE IA	131
APÊNDICE 13 – FRAMEWORKS DE INTELIGÊNCIA ARTIFICIAL	135
APÊNDICE 14 – VISUALIZAÇÃO DE DADOS E STORYTELLING.....	162
APÊNDICE 15 – TÓPICOS EM INTELIGÊNCIA ARTIFICIAL	166
APÊNDICE 16 – CÓDIGO RAG	175
ANEXO 1 – CÓDIGO STORYTELLING.....	179

1 PARECER TÉCNICO

Este estudo descreve uma proposta de sistema para acervos jornalísticos digitais que, por meio da arquitetura de *Retrieval-Augmented Generation* (RAG), integra a busca vetorial em um vasto corpus de conteúdo a um grande modelo de linguagem (LLM), objetivando a geração de respostas contextuais, factuais e referenciadas para aprimorar a eficiência e celeridade do processo de apuração.

1.1 INTRODUÇÃO

No jornalismo, apuração é um processo em que o jornalista coleta informações que precedem a produção de uma nova notícia. Envolve a busca por fatos, entrevistas, análise de documentos e contextualização de um tema, com o objetivo de garantir a precisão e a credibilidade da informação. Em um veículo de mídia tradicional, o acervo pode ser grande, desordenado, produzido sob diferentes padrões e muitas vezes duplicado.

Métodos de busca tradicionais, como os *B-Trees* utilizados em buscas de *e-commerces* (Sowell; Golab; Shah, 2012), ou índices invertidos, utilizados em buscadores como o Google (Dean; Ghemawat, 2004), podem trazer muitos resultados similares, mas nestas condições são eventualmente fora de contexto.

O conceito RAG se baseia na integração de dois componentes do Processamento de Linguagem Natural (PLN): a Recuperação de Informação (RI) e a Geração de Linguagem Natural (GLN). A estrutura RAG, introduzida pela primeira vez por Lewis (Lewis et al., 2020), combina métodos de recuperação densa com modelos generativos de larga escala para produzir respostas que são tanto contextualmente relevantes quanto factualmente precisas.

A proposta de solução aqui apresentada consiste em utilizar um mecanismo de busca fácil de implementar, associada a um grande modelo de linguagem (LLM) aberto, para que o contexto seja entendido e gere respostas melhores neste processo de apuração jornalística.

Vale ressaltar que a o uso de RAG emergiu como uma solução promissora ao incorporar conhecimento de bancos de dados externos e, principalmente, permite atualizações contínuas de conhecimento e a integração de informações de domínio específico (Gao et al., 2023).

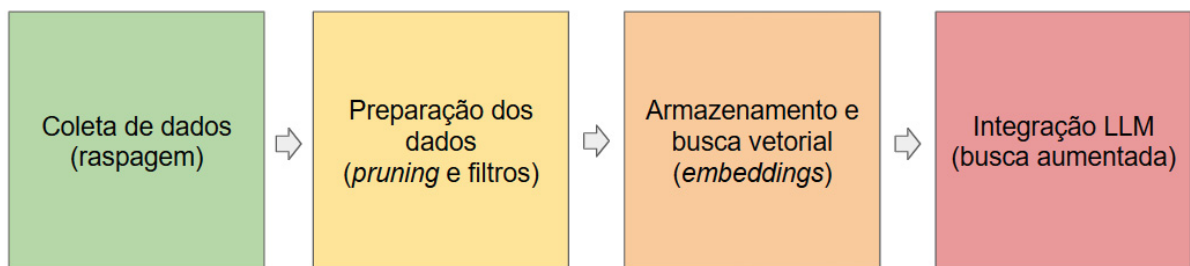
1.2 PROCESSO DE CRIAÇÃO

Nesta proposta de solução, documenta-se a experiência de desenvolvimento como um guia passo a passo para construir sistemas RAG, conforme mostra a FIGURA 1, capaz de buscar e indexar conteúdo de um site como a principal base de conhecimento. Discute-se as escolhas de design, o desenvolvimento do sistema e a avaliação do guia, oferecendo percepções sobre os desafios técnicos encontrados e as soluções práticas aplicadas.

O código completo desta implementação etapa por etapa está no Apêndice 16 deste trabalho.

Durante o desenvolvimento aplicou-se princípios básicos conhecidos e utilizou-se a linguagem Python como base, tal qual implementações similares (Khan et al., 2024).

FIGURA 1 – Visão geral do processo da solução



FONTE: Luiz Henrique dos Santos Cruz (2025)

1.2.1 Etapa de raspagem

Para obter o conteúdo, mesmo sem acesso a base de dados, pode-se consumir URLs de qualquer site através do arquivo de *sitemap* (são arquivos XML geralmente disponibilizados para buscadores tradicionais indexar o conteúdo). O processo recomendado utiliza a biblioteca Python XML ElementTree, para extrair todas as URLs deste arquivo.

Para cada URL, pode-se usar uma segunda biblioteca preparada para tal, chamada Crawl4AI (Unclecode, 2024). Com esta biblioteca, recomenda-se salvar arquivos JSON com os dados de título, conteúdo, o endereço *Uniform Resource*

Location (URL) para referência e *slug* (termo usado para título transformado em URL amigável). Esta recomendação facilita a rastreabilidade posterior.

Vale ressaltar que é possível otimizar a execução com *multithread*, para acessar quatro a seis URLs simultaneamente durante a raspagem.

1.2.2 Etapa de preparação

Durante o processo de extração, deve-se executar limpezas no conteúdo. A primeira limpeza é conhecida como *pruning* (ou poda, em tradução literal). Este procedimento reduz a densidade do texto sem comprometer seu significado. Recomenda-se também limpar links do conteúdo que apontem para sites externos ou irrelevantes, ou até links de navegação interna do site.

1.2.3 Etapa de vetorização e armazenamento

Com os dados tratados, a próxima técnica é a de vetorização, que, em termos simples, cria uma representação vetorial do texto. Cada palavra é representada no espaço, e a distância entre pontos indica o grau de semelhança entre elas.

Para facilitar, adotou-se o modelo pré-treinado all-MiniLM-L6-v2 (Sentence-Transformers, 2022) para usar a técnica de vetorização conhecida como *sentence transformer*. Esse modelo gera *embeddings* (vetores) coesos para textos. Para ilustrar a ideia, as frases "O céu está azul" e "O dia está lindo hoje" são lexicalmente diferentes, mas semanticamente próximas e, conseqüentemente, geram vetores próximos.

Para armazenar tanto a base limpa, quanto sua representação vetorial, utilizou-se o banco vetorial de código aberto ChromaDB (Chroma, 2023). A sugestão é devido a sua simplicidade de implementação. Com os arquivos de cada conteúdo salvo localmente, basta agrupá-los antes de salvar. O banco já salva os *embeddings* e permite acesso rápido em uma busca.

1.2.4 Etapa de recuperação e geração

Para ter um LLM sendo executado localmente há diversas alternativas. Uma das mais práticas é utilizar a ferramenta Ollama (Ollama, 2024) e baixar modelos

abertos, como o LLM DeepSeek R1 de oito bilhões de parâmetros. Essa configuração é suficiente para experimentação.

Para avançar além de implementar um modelo de chat, é possível utilizar a biblioteca Streamlit (Streamlit, 2023), que permite criar interfaces web rapidamente. Com pouco esforço é possível capturar o texto da pergunta do usuário, realiza uma busca vetorial no ChromaDB e passar o resultado da busca para o LLM, num *prompt* mais organizado. A resposta é refinada e apresentada ao usuário. No *prompt* é recomendado também adicionar informações para listar os links das matérias encontradas, o que ajuda a criar referências.

1.3 DISCUSSÃO

A estratégia modular, que engloba etapas de raspagem, preparação, vetorização e geração, possibilita um desenvolvimento controlado e progressivo.

As tecnologias propostas são de código aberto, o que as torna acessíveis para execução local, sem a necessidade de incorrer em custos adicionais provenientes de APIs de modelos mais avançados. Este é um ponto crucial para as empresas de mídia, que geralmente operam dentro de orçamentos restritos.

Contudo, há uma preocupação com a escalabilidade. O modelo apresentado é eficiente para empresas que possuem acervos de até médio porte. Todavia, para acervos com milhões de conteúdos, é imprescindível escalar para múltiplas instâncias de banco, em regime de *cluster*. Ou buscar limpezas de conteúdo mais eficientes para minimizar o tamanho armazenado.

O modelo RAG já se consolidou como um paradigma robusto na comunidade de Processamento de Linguagem Natural (PLN). A aplicação de *sentence transformers* para a produção de *embeddings*, que capturam a essência semântica do texto, representa no momento o que há de mais avançado para a fase de recuperação de informações.

É preciso que se faça uma avaliação quanto aspectos cruciais em análises jornalísticas: a preocupação com o encadeamento cronológico dos fatos. E também se a solução consegue entender a relação de causa e efeito dos artigos (narrativa).

Finalmente, uma outra preocupação é que já existem estudos (Yamin et al., 2024) que demonstram que os LLMs se confundem com a ordem em que os eventos são mencionados e sua relação de causalidade. Se uma história diz "O evento B

aconteceu. Antes disso, o evento A ocorreu", o modelo tende a pensar que B causou A. Gerando uma correlação inexistente.

1.4 CONSIDERAÇÕES FINAIS

Este parecer técnico demonstrou a viabilidade da implementação de um sistema de Geração Aumentada por Recuperação (RAG) para a consulta a acervos jornalísticos digitais, detalhando uma arquitetura modular e de baixo custo que utiliza exclusivamente tecnologias de código aberto. Esta abordagem não apenas democratiza o acesso a ferramentas avançadas de inteligência artificial, como também oferece uma solução prática e replicável para veículos de mídia superarem os desafios de busca em arquivos desordenados, aprimorando o processo de apuração com respostas contextualmente ricas e devidamente referenciadas.

Apesar dos resultados promissores, a solução enfrenta limitações, principalmente em escalabilidade para acervos de grande porte e, de forma mais crucial, na compreensão do encadeamento cronológico e causal dos fatos. A tendência dos LLMs em se confundir com a ordem narrativa, como aponta a literatura, representa um risco significativo para a precisão jornalística.

Portanto, trabalhos futuros devem se concentrar não apenas em otimizar a performance para grandes volumes de dados, mas, fundamentalmente, em desenvolver mecanismos que garantam a correta interpretação da linha do tempo e das relações de causa e efeito, seja por etapas com prompts específicos ou modelos pré-treinados. De toda forma, o resultado deste aprimoramento é um requisito indispensável para a integridade da apuração.

REFERÊNCIAS

ALGORITHMIC JUSTICE LEAGUE. Disponível em: <https://www.ajl.org/>. Acesso em: Acesso em 29 de Junho de 2024.

BUCHHOLZ, KATHARINA. **Threads Shoots Past One Million User Mark at Lightning, Speed**, Statista, 2023. Disponível em: <https://www.statista.com/chart/29174/time-to-one-million-users/>. Acesso em 21 Junho de 2024.

BRASIL. Senado Federal. Projeto de Lei nº 2338, de 2023. **Dispõe sobre o uso da Inteligência Artificial**. Brasília, DF: Senado Federal, 2023. Disponível em: <https://www25.senado.leg.br/web/atividade/materias/-/materia/157233>. Acesso em 21 de Junho de 2024.

BRASIL. Senado Federal. **Marco Legal da Inteligência Artificial**. Disponível em: <https://www12.senado.leg.br/radio/1/projeto-em-destaque/2023/06/13/marco-legal-da-inteligencia-artificial>. Acesso em 21 de Junho de 2024.

BROWN, T. B.; MANN, B.; RYDER, N.; SUBBIAH, M. et al. **Language Models are Few-Shot Learners**. 2020. Disponível em: <https://arxiv.org/pdf/2005.14165>. Acesso em: 29 de Junho de 2024.

BUOLAMWINI, Joy. **How I'm fighting bias in algorithms**. TED, 2016. Disponível em: https://www.ted.com/talks/joy_buolamwini_how_i_m_fighting_bias_in_algorithms?language=en. Acesso em 29 de Junho de 2024.

CHROMA. **The AI-native open-source embedding database**. San Francisco, CA: Chroma, 2023. Disponível em: <https://www.trychroma.com/>. Acesso em: 20 Julho de 2025.

DEAN, Jeffrey; GHEMAWAT, Sanjay. **MapReduce: simplified data processing on large clusters**. In: SYMPOSIUM ON OPERATING SYSTEM DESIGN AND

IMPLEMENTATION (OSDI), 6., 2004, San Francisco. *Anais [...]*. Berkeley, CA: USENIX Association, 2004. p. 137-150. Disponível em: <https://static.googleusercontent.com/media/research.google.com/pt-BR/pubs/archive/37043.pdf>. Acesso em: 19 Julho de 2025.

DW. **ChatGPT: Italy blocks AI chatbot over privacy concerns**. 2023. Disponível em: <https://www.dw.com/en/chatgpt-italy-blocks-ai-chatbot-over-privacy-concerns/a-65200137>. Acesso em 29 de Junho de 2024.

EMSLEY, ROBIN. **ChatGPT: não são alucinações – são invenções e falsificações**. *Nature Esquizofr*, v. 9, p. 52, 2023. Disponível em: <https://doi.org/10.1038/s41537-023-00379-4>. Acesso em 21 de Junho de 2024.

GOMES, Paulo Emílio Salles. **Uma situação Colonial?**. São Paulo: Companhia das Letras, 2016.

GAO, Yunfan et al. **Retrieval-augmented generation for large language models: a survey**. *Arxiv*, 2023. DOI: 10.48550/arXiv.2312.10997. Disponível em: <https://arxiv.org/abs/2312.10997>. Acesso em: 19 Julho de 2025.

HU, KRYSTAL. **ChatGPT sets record for fastest-growing user base - analyst note**, Reuters, 2023. Disponível em: <https://www.reuters.com/technology/chatgpt-sets-record-fastest-growing-user-base-analyst-note-2023-02-01/>. Acesso em 21 de Junho de 2024.

HER. Direção: Spike Jonze. [Filme]. Produção: *Warner Bros. Pictures*, 2013

LEWIS, Patrick et al. **Retrieval-augmented generation for knowledge-intensive NLP tasks**. *Arxiv*, 2020. DOI: 10.48550/arXiv.2005.11401. Disponível em: <https://arxiv.org/pdf/2005.11401>. Acesso em: 20 Julho de 2025.

LIU, Yi; DENG, Gelei; XU, Zhengzi; LI, Yuekang; ZHENG, Yaowen; ZHANG, Ying; ZHAO, Lida; WANG, Kailong; ZHANG, Tianwei; LIU, Yang. **Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study**. 2023. Disponível em: <https://arxiv.org/pdf/2305.13860>. Acesso em 30 de Junho de 2024.

MAZZONI, L. et al. ***Language Models and the Future of Information Retrieval***. 2023. Disponível em: <https://arxiv.org/abs/2303.08774>. Acesso em 29 de Junho de 2024.

NUNES, Devair Sebastião. **A inteligência artificial pode apresentar viés racista, preconceituoso?** Senado Federal. Comitê Permanente pela Promoção da Igualdade de Gênero e Raça. Brasília, 04 ago. 2022. Disponível em: <https://www2.senado.gov.br/bdsf/handle/id/599393>. Acesso em 26 de Junho de 2024.

OLLAMA. ***Get up and running with large language models, locally***. [S. l.]: Ollama, 2024. Disponível em: <https://ollama.com/>. Acesso em: 20 Julho de 2025.

OPENAI. ***Supported Countries***. Disponível em: <https://platform.openai.com/docs/supported-countries>. Acesso em 26 de Junho de 2024.

OPENAI. ***Data Partnerships***. Disponível em: <https://openai.com/index/data-partnerships/>. Acesso em 29 de Junho de 2024.

ORTHOPEDIC AND SPORTS SPECIALISTS. ***De Quervain's Tendons***. Disponível em: <https://www.orthopedicandsportsspecialists.com/de-quervains-tendonitis-andrew-louis-degruccio-md.html>. Acesso em 21 de Junho de 2024.

PATEL, Dylan; AHMAD, Afzal. ***The Inference Cost of Search Disruption***. SemiAnalysis, 2023. Disponível em: <https://www.semianalysis.com/p/the-inference-cost-of-search-disruption>. Acesso em 26 de Junho de 2024.

RIGHT TO WARN. Carta aberta. Disponível em: <https://righttowarn.ai/>. Acesso em: 29 de Junho de 2024.

SENTENCE-TRANSFORMERS. **all-MiniLM-L6-v2**. [S. l.]: Hugging Face, 2022.
Disponível em: <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>.
Acesso em: 20 jul. 2025.

SOWELL, B; GOLAB, W; SHAH, M. A. **Minuet: A Scalable Distributed Multiversion B-Tree**, Arxiv. 2012. Disponível em: <https://arxiv.org/abs/1205.6699>. Acesso em 10 Junho de 2025.

STREAMLIT. **A faster way to build and share data apps**. San Francisco, CA: Streamlit, 2024. Disponível em: <https://streamlit.io/>. Acesso em: 20 Julho de 2025.

STROM, P. S.; STROM, R. D.; SINDEL-ARRINGTON, T.; RUDE, R. V. **Student Attention and Distraction in Community College**. 2023. Disponível em: https://www.researchgate.net/publication/368301459_Student_Attention_and_Distraction_in_Community_College. Acesso em 21 de Junho de 2024.

UNCLECODE. **Crawl4AI**: Open-source LLM friendly web crawler & scraper. [S. l.]: GitHub, 2024. Disponível em: <https://github.com/unclecode/crawl4ai>. Acesso em: 20 Julho de 2025.

YAMIN, Khurram et al. **Failure modes of LLMs for causal reasoning on narratives**. Arxiv, 2024. Disponível em: <https://arxiv.org/pdf/2410.23884>. Acesso em: 20 Julho de 2025.

YOUNGOV. **Could you live without your smartphone?** Disponível em: <https://yougov.co.uk/technology/articles/22577-could-you-live-without-your-smartphone>. Acesso em 21 de Junho de 2024

ZIEGLER, Daniel M.; STIENNON, Nisan; WU, Jeffrey; BROWN, Tom B.; RADFORD, Alec; AMODEI, Dario; CHRISTIANO, Paul; IRVING, Geoffrey. **Fine-Tuning Language Models from Human Preferences**. 2019. Disponível em: <https://arxiv.org/pdf/1909.08593>. Acesso em 29 de Junho de 2024.

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.

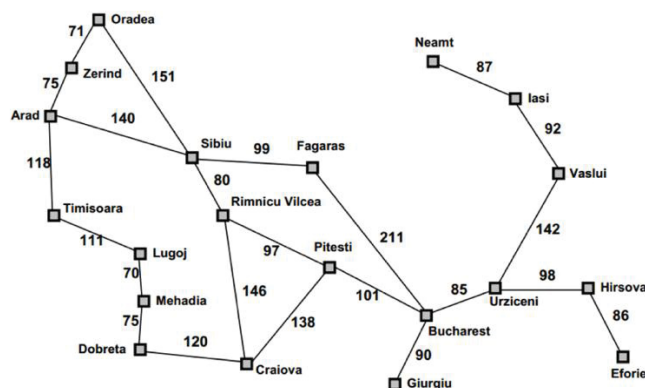
2 Busca Heurística

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

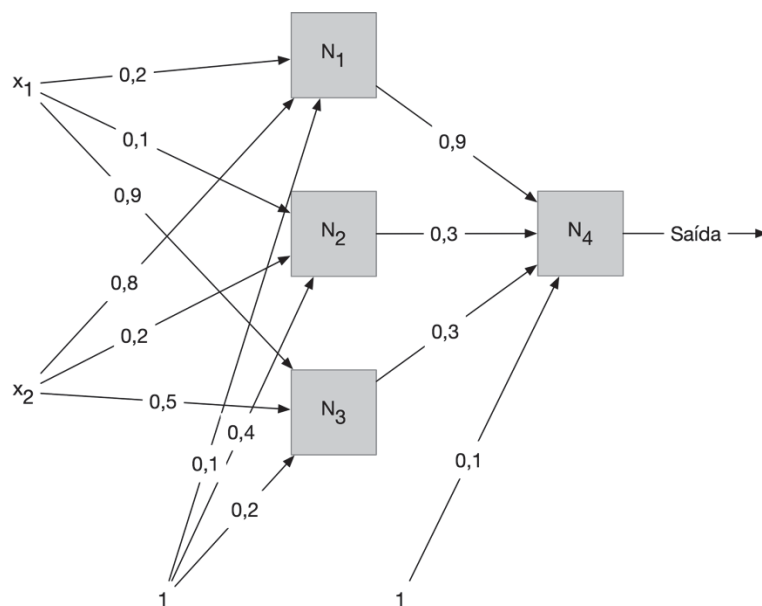
Conjunção: $\alpha, \beta \vdash \alpha \wedge \beta$

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

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 habilidades como aprendizado, raciocínio, resolução de problemas, compreensão de linguagem natural, reconhecimento de padrões, tomada de decisões e muito mais.

Os sistemas de IA são projetados para analisar grandes quantidades de dados, identificar padrões e tendências, aprender com exemplos passados e tomar decisões ou fornecer insights baseados nesses dados. Existem várias abordagens e técnicas dentro da IA incluindo aprendizado de máquina, redes neurais, lógica *fuzzy*, algoritmos genéticos, processamento de linguagem natural, visão computacional, entre outras.

A IA tem uma ampla gama de aplicações em diversos setores, incluindo medicina, finanças, manufatura, automotivo, jogos, assistentes virtuais, entretenimento e muito mais. Seu objetivo final é criar sistemas que possam imitar a inteligência humana de maneira eficaz em várias tarefas e contextos.

1 - A citação da frase "desenvolvimento de sistemas e algoritmos capazes de realizar tarefas que normalmente exigiriam inteligência humana", por si só já assinala o uso da abordagem de "Pensar como os humanos" indicando a busca por desenvolver algoritmos que se aproximem deste estado.

2 – A resposta cita também que a IA é projetada para analisar grandes quantidades de dados o que indica a abordagem "Pensar racionalmente".

3 – A observação que a IA possui uma ampla gama de aplicações em diversos setores nos indica a abordagem "Agir como humanos"

4 – O texto fala sobre que as IA são projetadas para aprender por exemplos e tomar decisões baseadas nesses dados, essa abordagem está relacionada a "Agir racionalmente"

Os conteúdos relacionados as repostas citam que o ChatGPT pode ser definido como um modelo de linguagem que tem como base IA e utiliza redes neurais, machine learning e avançados algoritmos de processamento de linguagem natural.

Com foco em diálogos virtuais, sendo a evolução dos antigos *chatbots*, tendo sido treinado com grandes volumes de dados ou qual consegue manter uma conversa humana em linguagem natural, respondendo perguntas feitas pelos usuários, muito útil para facilitar o acesso a informações e códigos a partir de seus dados. Porém criticado por possivelmente utilizar informações autorais ao qual indicaria plagio e por não indicar as fontes utilizadas para montar o racional da resposta, deixando assim lacunas e questionamento sobre as origens dos dados.

Outro ponto é relação aos dados por nós imputados nas perguntas ao ChatGPT, elas são usadas para armazenamento e utilização posterior? São processadas e descartadas? As fontes nos indicam que temos que evitar de compartilhar informações confidenciais.

O ChatGPT possui limitações: Bias: pois devido ao treinamento com grandes volumes de dados desatualizados ou enviesados, Contexto limitado: ele pode ter ainda dificuldades em responder nuances complexas ou informações implícitas.

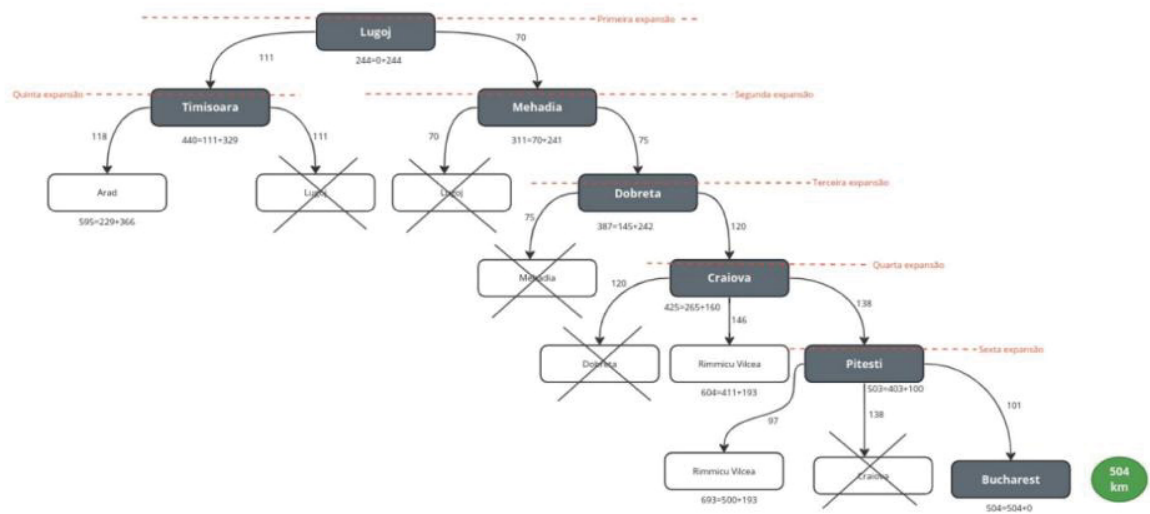
Eu classificaria o ChatGPT como um avanço tecnológico, devendo ser considerado tal qual a revolução industrial, uma fonte que não irá fazer o trabalho para o ser humano, mas facilitará e auxiliará dando agilidade em trabalhos “mecânicos” por assim dizer, permitindo que o tempo seja aproveitado para desenvolvimento de novas ideias e criações.

Baseado nas 4 abordagens estudadas podemos indicar que o ChatGPT as simula, tentando ao buscar as respostas ele age como “Pensar como os humanos”, e dentro das grandes quantidades de material ele usa essa abordagem para separar os conteúdos como se pensasse racionalmente, claro que devido as suas limitações ainda não tem como ele agir como humanos, mas devido ao seu conhecimento os humanos podem aplicar seu conhecimento, o agir racionalmente pode ser atingido com maior facilidade uma vez que sua análise é baseada em dados.

2. BUSCA HEURÍSTICA

Resolução do exercício de busca heurística, em que o algoritmo é a aplicação de uma função heurística para distância, usada para decidir qual ramo seguir. A resposta chega a Bucareste, com 504 km de distância percorrida.

FIGURA 2 - Exercício de Heurística



FONTE: Equipe de Luiz Henrique dos Santos Cruz (2024).

3. LÓGICA

FIGURA 3 - Exercício de Lógica

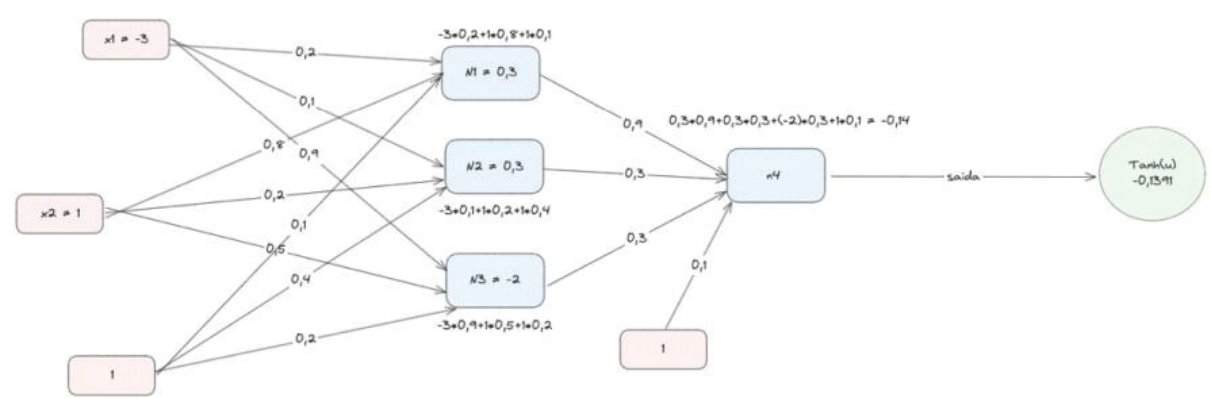
R1: $p \rightarrow q$	
R2: $q \rightarrow r$	
R3: $\neg r \vee p$	
R4: $r \rightarrow p$	Equivalência Implicação R3
R5: $q \rightarrow p$	Silogismo Hipotético R2, R4
R6: $(q \rightarrow p) \wedge (p \rightarrow q)$	Conjunção R5, R1
R7: $q \leftrightarrow p$	Equivalência Bicondicional R6

FONTE: Equipe de Luiz Henrique dos Santos Cruz (2024).

4. REDES NEURAIS ARTIFICIAIS

Resolução dos valores dos neurônios de acordo com os pesos e saída esperada da função de avaliação, da rede neural apresentada.

FIGURA 4 - Exercício de Redes Neurais



FONTE: Equipe de Luiz Henrique dos Santos Cruz (2024).

De acordo com entradas e pesos, calculamos:

- Valor do neurônio N1 = 0,3
- Valor do neurônio N2 = 0,3
- Valor do neurônio N3 = -2
- Valor do neurônio de Saída = -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](#)). A mesma foi adaptada para ser utilizada no presente exercício.

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

Metadados:

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

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

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

1 Análise Exploratória dos dados

A partir da base de dados **precos_carros_brasil.csv**, execute as seguintes tarefas:

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

2 Visualização dos dados

A partir da base de dados **precos_carros_brasil.csv**, execute as seguintes tarefas:

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

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

A partir da base de dados **precos_carros_brasil.csv**, execute as seguintes tarefas:

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

- f. Dê uma breve explicação (máximo de quatro linhas) sobre os resultados encontrados na análise de importância de variáveis
- g. Escolha o melhor modelo com base nas métricas de avaliação MSE, MAE e R^2
- h. Dê uma breve explicação (máximo de quatro linhas) sobre qual modelo gerou o melhor resultado e a métrica de avaliação utilizada

B - RESOLUÇÃO

1. ANÁLISE EXPLORATÓRIA DOS DADOS

Para a análise exploratória, listamos os tipos de dados de cada coluna da base de dados fornecida.

```

year_of_reference    float64
month_of_reference   object
fipecode             object
authentication       object
brand                object
model               object
fuel                 object
gear                 object
engine_size          object
year_model           float64
avg_price_brl        float64
dtype: object

```

Em seguida, verificamos a existência de dados faltantes.

```

year_of_reference    True
month_of_reference   True
fipecode             True
authentication       True
brand                True
model               True
fuel                 True
gear                 True
engine_size          True
year_model           True
avg_price_brl        True

```

```
dtype: bool
```

E também os valores faltantes por coluna:

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

Fizemos a limpeza dos dados e consultamos novamente os dados faltantes:

```
year_of_reference      0
month_of_reference     0
fipecode               0
authentication         0
brand                  0
model                  0
fuel                   0
gear                   0
engine_size            0
year_model             0
avg_price_brl          0
dtype: int64
```

Convertemos algumas colunas de dados que estavam no formato ponto flutuante para inteiro e exibimos seus tipos novamente.

```
year_of_reference      int32
month_of_reference     object
fipecode               object
authentication         object
brand                  object
model                  object
fuel                   object
gear                   object
engine_size            float64
year_model             int32
avg_price_brl          float64
dtype: object
```

Visualizamos o topo das linhas dos dados (*head*) para conferir os dados.

FIGURA 5 - Estrutura dos dados de carros

	year_of_reference	month_of_reference	fipe_code	authentication	brand	model	fuel	gear	engine_size	year_model	avg_price_brl
0	2021	January	004001-0	cfzltctzfwrpc	GM - Chevrolet	Corsa Wind 1.0 MPFI / EFI 2p	Gasoline	manual	1.0	2002	9162.0
1	2021	January	004001-0	cdqwxwpw3y2p	GM - Chevrolet	Corsa Wind 1.0 MPFI / EFI 2p	Gasoline	manual	1.0	2001	8832.0
2	2021	January	004001-0	cb1t3xwwj1xp	GM - Chevrolet	Corsa Wind 1.0 MPFI / EFI 2p	Gasoline	manual	1.0	2000	8388.0
3	2021	January	004001-0	cb9gct6j65r0	GM - Chevrolet	Corsa Wind 1.0 MPFI / EFI 2p	Alcohol	manual	1.0	2000	8453.0
4	2021	January	004003-7	g15wg0gbz1fx	GM - Chevrolet	Corsa Pick-Up GL/ Champ 1.6 MPFI / EFI	Gasoline	manual	1.6	2001	12525.0

Limpamos dados duplicados e validamos com a soma, para ver se era igual a zero, ou seja, se valores duplicados foram removidos. Depois da limpeza de duplicados, consultamos o número de linhas e colunas dos dados (*shape*).

```
(202295, 11)
```

Entendendo as variáveis numéricas, visualizamos a descrição dos dados e suas tendências centrais (*describe*).

FIGURA 6 - Descrição variáveis numéricas

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

E para variáveis categóricas, listamos.

FIGURA 7 - Descrição variáveis categóricas

	month_of_reference	fipe_code	authentication	brand	model	fuel	gear
count	202295	202295	202295	202295	202295	202295	202295
unique	12	2091	202295	6	2112	3	2
top	January	003281-6	cfzltzfwrcp	Fiat	Palio Week. Adv/Adv TRYON 1.8 mpi Flex	Gasoline	manual
freq	24260	425	1	44962	425	168684	161883

Para dupla conferência, uma contagem de valores faltantes por colunas.

```

year_of_reference      0
month_of_reference     0
fipe_code              0
authentication         0
brand                  0
model                  0
fuel                   0
gear                   0
engine_size            0
year_model             0
avg_price_brl          0
dtype: int64

```

2. VISUALIZAÇÃO DOS DADOS

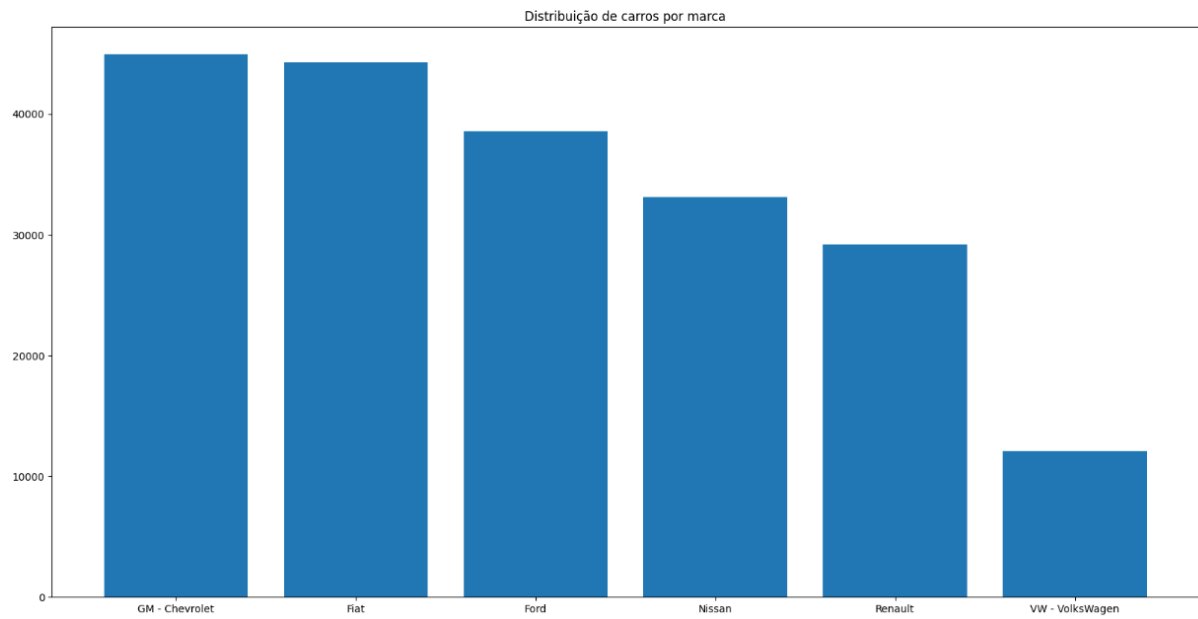
Iniciamos pela contagem de carros por marca.

FIGURA 8 - Contagem de carros por marca

	brand	count
0	Fiat	44962
1	VW - VolksWagen	44312
2	GM - Chevrolet	38590
3	Ford	33150
4	Renault	29191
5	Nissan	12090

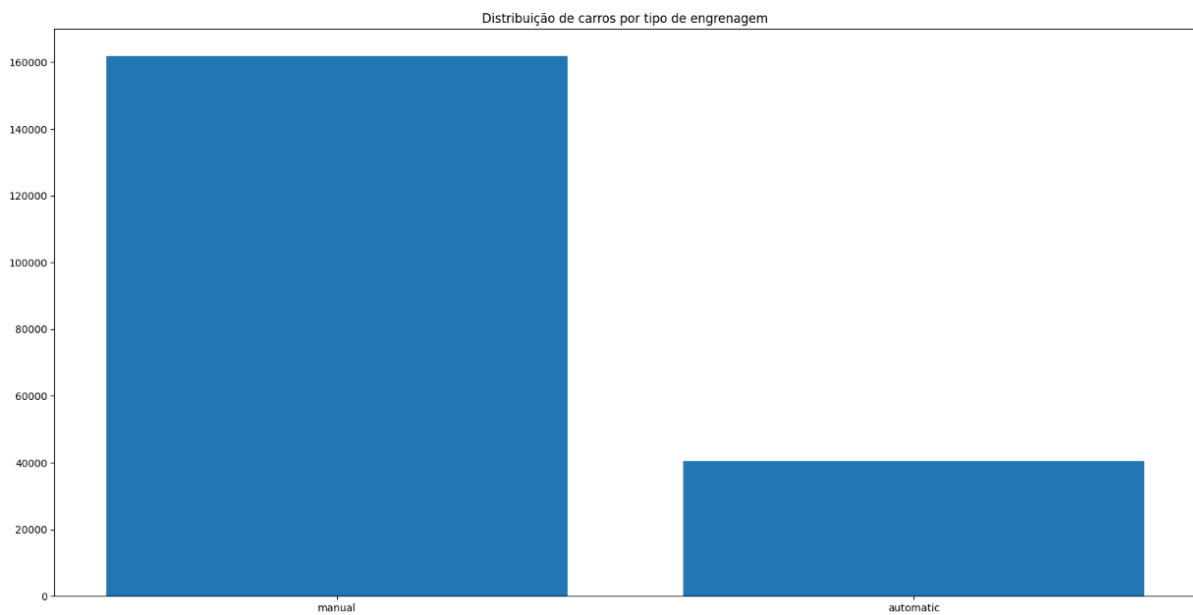
Distribuição de carros por marca.

GRÁFICO 1 - Distribuição de carros por marca



Distribuição de carros por engrenagem (*gear*).

GRÁFICO 2 - Distribuição de carros por engrenagem



Preço médio por ano.

FIGURA 9 - Preço médio de carros por ano

	Mês de referencia	year_of_reference	Preço médio
0	April	2021	44451.0
1	April	2022	57150.0
2	August	2021	49363.0
3	August	2022	57924.0
4	December	2021	53674.0

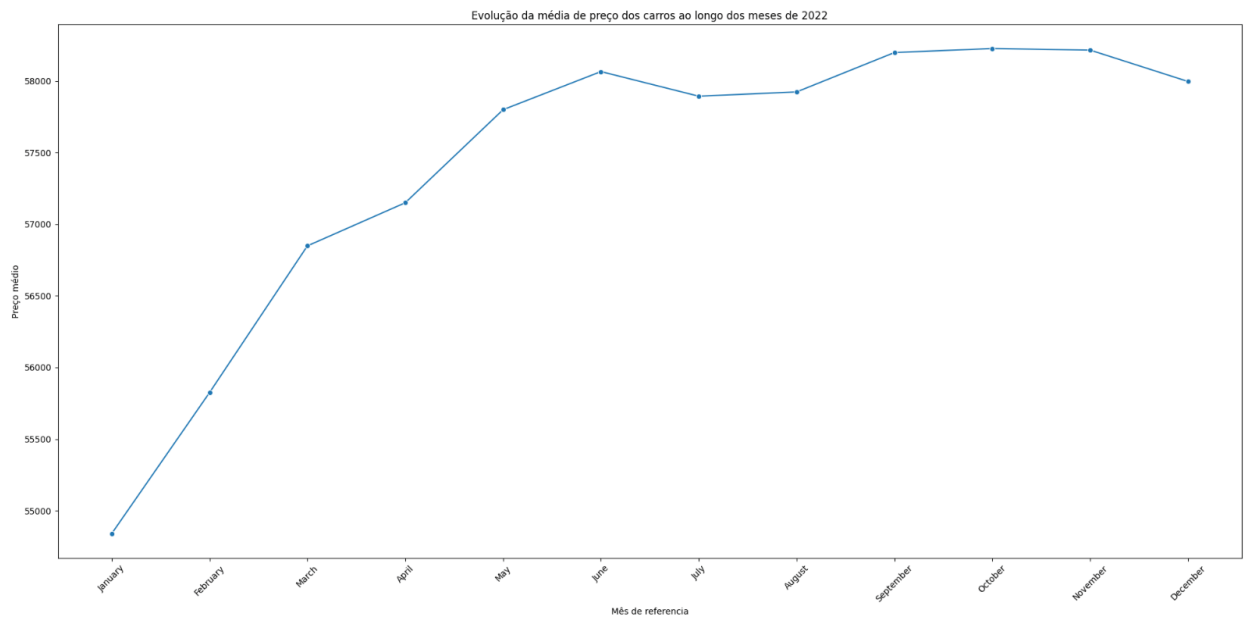
Para fazer uma análise de evolução de preço, filtrando o ano de 2022, transformamos o mês para numérico.

FIGURA 10 - Transformação mês para numérico

	Mês de referencia	year_of_reference	Preço médio	Mês de referencia - numerico
9	January	2022	54840.0	1
7	February	2022	55825.0	2
16	March	2022	56849.0	3
1	April	2022	57150.0	4
18	May	2022	57800.0	5
14	June	2022	58066.0	6
12	July	2022	57894.0	7
3	August	2022	57924.0	8
24	September	2022	58199.0	9
22	October	2022	58227.0	10
20	November	2022	58216.0	11
5	December	2022	57997.0	12

Representação linear da evolução dos preços para este ano.

GRÁFICO 3 - Representação evolução preços dos carros



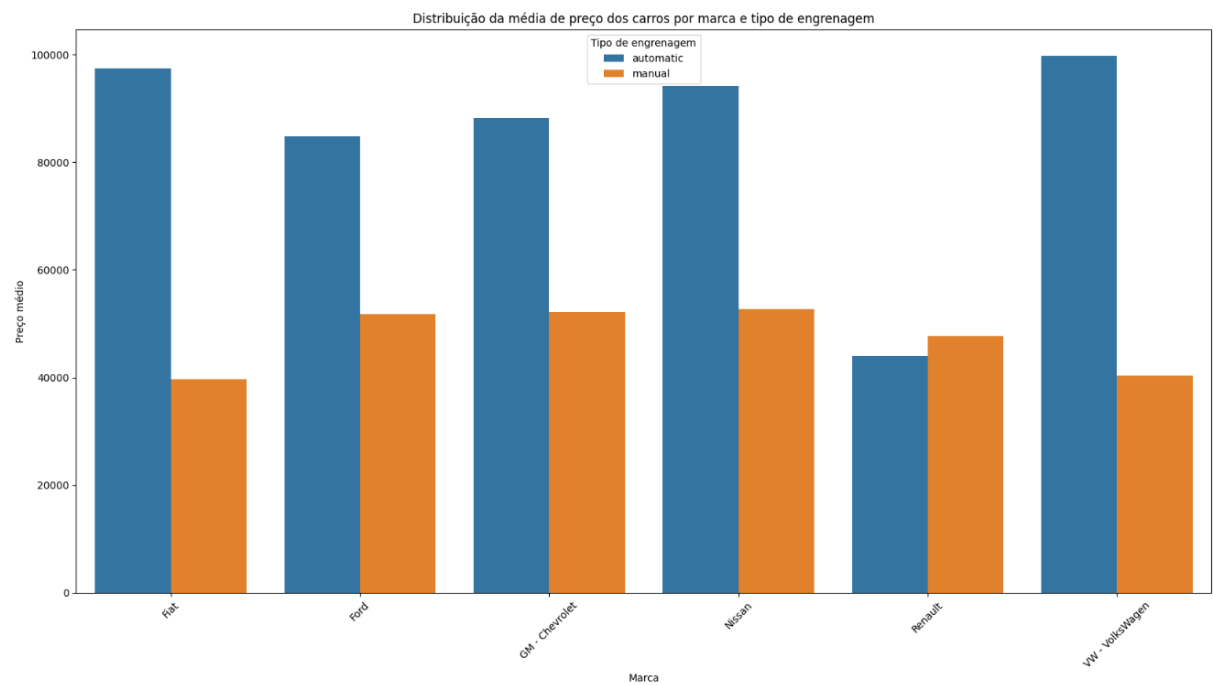
Preço médio por tipo de engrenagem.

FIGURA 11 - Preço médio por engrenagem

	Marca	Tipo de engrenagem	Preço médio
0	Fiat	automatic	97397.0
1	Fiat	manual	39694.0
2	Ford	automatic	84769.0
3	Ford	manual	51784.0
4	GM - Chevrolet	automatic	88157.0

Distribuição por marca e por engrenagem.

GRÁFICO 4 - Distribuição por marca e engrenagem

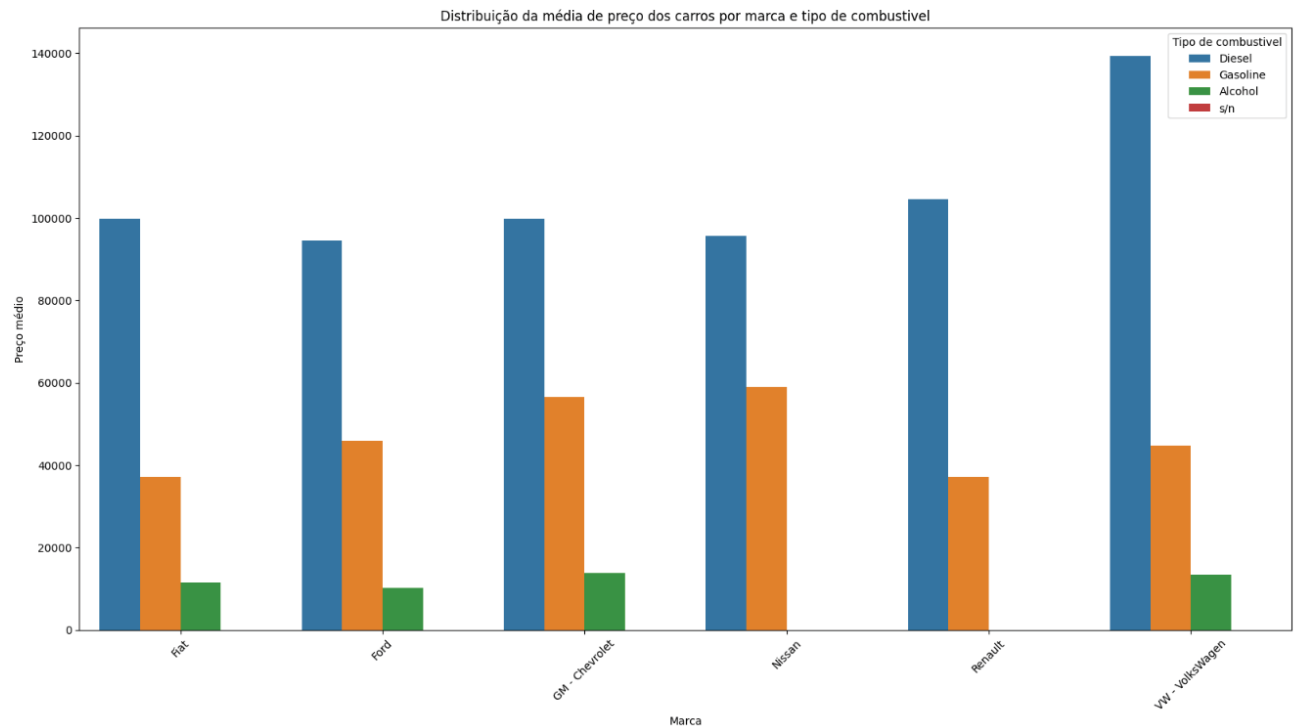


Preço médio por tipo de combustível

FIGURA 12 - Preço médio por tipo de combustível

	Marca	Tipo de combustível	Preço médio
0	Fiat	Alcohol	11510.0
1	Fiat	Diesel	99814.0
2	Fiat	Gasoline	37197.0
3	Ford	Alcohol	10149.0
4	Ford	Diesel	94526.0

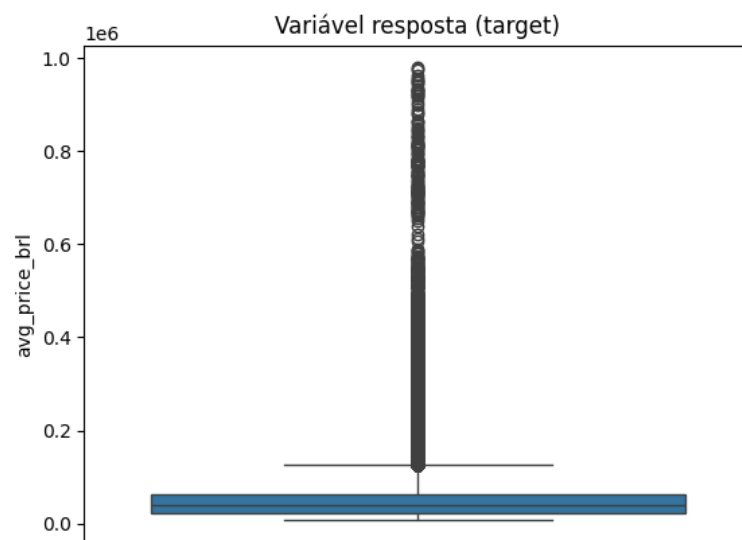
GRÁFICO 5 - Distribuição média de preços por marca e combustível



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

Análise da variável resposta ou alvo.

GRÁFICO 6 - Análise da variável alvo



Ajuste das variáveis *gear* e *fuel* para inclusão no modelo como numéricas.

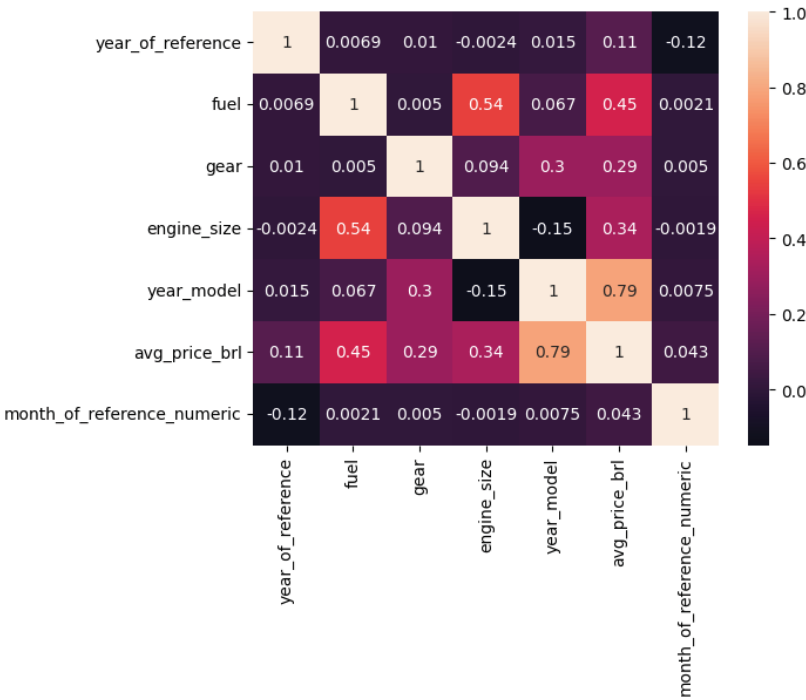
FIGURA 13 - Ajuste das variáveis engrenagem e combustível

	year_of_reference	fuel	gear	engine_size	year_model	avg_price_brl	month_of_reference_numeric
0	2021	1.0	0.0	1.0	2002	9162.0	1
1	2021	1.0	0.0	1.0	2001	8832.0	1
2	2021	1.0	0.0	1.0	2000	8388.0	1
3	2021	0.0	0.0	1.0	2000	8453.0	1
4	2021	1.0	0.0	1.6	2001	12525.0	1

Mapa de correlação das variáveis, para checar a validade para o treinamento.

GRÁFICO 7 - Gráfico correlação das variáveis numéricas

Mapa de Correlação das Variáveis Numéricas



Removemos a variável alvo, para a variável X.

FIGURA 14 - Remoção da variável alvo

	year_of_reference	fuel	gear	engine_size	year_model	month_of_reference_numeric
0	2021	1.0	0.0	1.0	2002	1
1	2021	1.0	0.0	1.0	2001	1
2	2021	1.0	0.0	1.0	2000	1
3	2021	0.0	0.0	1.0	2000	1
4	2021	1.0	0.0	1.6	2001	1

Variável Y fica apenas com a variável de target.

```
0    9162.0
1    8832.0
2    8388.0
3    8453.0
4   12525.0
Name: avg_price_brl, dtype: float64
```

Separamos a variável X em treinamento e teste respectivamente.

FIGURA 15 - Dados de treinamento

(151721, 6)

	year_of_reference	fuel	gear	engine_size	year_model	month_of_reference_numeric
156364	2022	2.0	0.0	2.3	2020	8

FIGURA 16 - Dados de teste

(50574, 6)

	year_of_reference	fuel	gear	engine_size	year_model	month_of_reference_numeric
180633	2022	1.0	0.0	1.6	2015	11

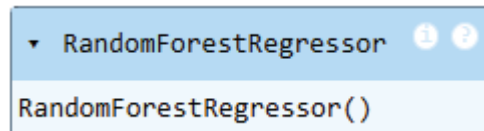
Observamos também a variável de target.

```
180633    42595.0
13130     10989.0
163315     9087.0
```

```
121464      26965.0
14044      57102.0
Name: avg_price_brl, dtype: float64
```

Treinamos o modelo de regressão Random Forest com os dados. E exibimos os valores preditos, para confirmar que o modelo foi treinado corretamente.

FIGURA 17 - Treinamento Random Forest



```
array([ 47256.76764717, 12693.01938437, 15322.83340587, ...,
       114026.86849782, 16275.65159838, 23545.50128735])
```

Listamos a importância das variáveis.

FIGURA 18 - Importância das variáveis

	importance
engine_size	0.436672
year_model	0.400412
fuel	0.090222
gear	0.044247
year_of_reference	0.014751
month_of_reference_numeric	0.013696

E avaliamos as métricas de acurácia do modelo: MSE, MAE e R^2 respectivamente.

```
203761114.492017
7216.59663624075
0.9242877621762614
```

Ajustamos novamente o modelo de regressão com parâmetros de profundidade da árvore, mínimo de folhas, divisão mínima e número de estimadores. Avaliamos novamente as métricas de importância e acuraria.

FIGURA 19 - Treinamento Random Forest com parâmetros ajustados

```
RandomForestRegressor
RandomForestRegressor(max_depth=29, min_samples_leaf=32, min_samples_split=28,
n_estimators=208, random_state=43)
```

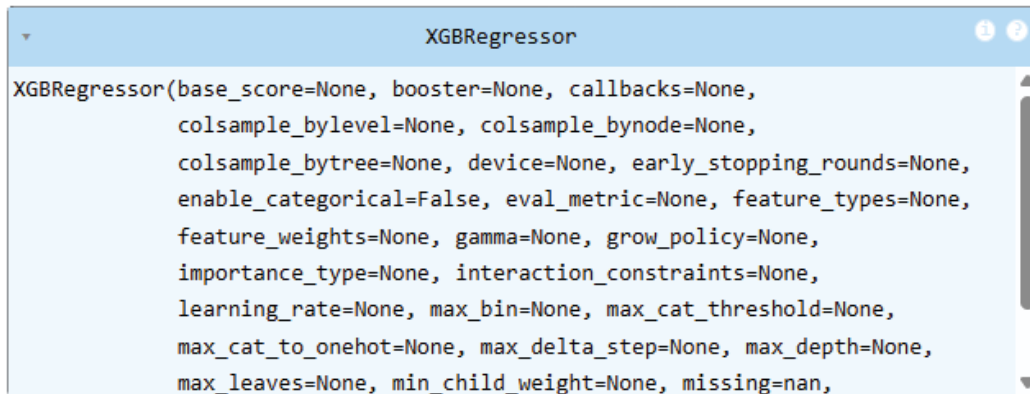
FIGURA 20 - Importância das variáveis

	importance
engine_size	0.444924
year_model	0.418263
fuel	0.094257
gear	0.026391
year_of_reference	0.012250
month_of_reference_numeric	0.003915

237397879.6039181
7386.7974547308
0.9117892304219445

Por fim, treinamos o modelo XGBoost. Mostrando os valores preditos e a importância.

FIGURA 21 - Treinamento XGBoost



```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             feature_weights=None, gamma=None, grow_policy=None,
             importance_type=None, interaction_constraints=None,
             learning_rate=None, max_bin=None, max_cat_threshold=None,
             max_cat_to_onehot=None, max_delta_step=None, max_depth=None,
             max_leaves=None, min_child_weight=None, missing=nan,
```

```
array([ 46954.406, 12775.354, 15823.265, ..., 116043.96 , 14408.948,
       23491.248], dtype=float32)
```

FIGURA 22 - Importância das variáveis

	importance
fuel	0.299901
engine_size	0.271182
year_model	0.249633
gear	0.156661
year_of_reference	0.017793
month_of_reference_numeric	0.004831

Para este modelo avaliamos a acurácia:

```
163253434.04382885
6774.170373854247
0.9393393442380654
```

E finalmente calculamos o mínimo de MSE dos modelos e o máximo do R^2

```
163253434.04382885
0.9393393442380654
```

Indicando o XGBoost como modelo mais adequado na predição dos preços dos veículos.

APÊNDICE 3 – LINGUAGEM R

A – ENUNCIADO

1 Pesquisa com Dados de Satélite (Satellite)

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

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

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

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

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

Tarefas:

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

2 Estimativa de Volumes de Árvores

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

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

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

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

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

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

Tarefas

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

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

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

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

- Coeficiente de determinação: R^2

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

onde y_i é o valor observado, $\hat{y}_i$ é o valor predito e $\bar{y}$ é a média dos valores y_i observados. Quanto mais perto de 1 melhor é o modelo;

- Erro padrão da estimativa: S_{yx}

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

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

- $S_{yx}\%$

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

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

7. Escolha o melhor modelo.

B – RESOLUÇÃO

1. PESQUISA COM DADOS DE SATÉLITE (SATELLITE)

Aplicamos os modelos Random Forest, SVM e RNA nos dados de satélites, particionados em 80% para treino e 20% para teste. Avaliamos os modelos com matriz de confusão.

FIGURA 23 - Matriz de confusão para Random Forest

```
confusionMatrix(predicoes.rf, teste$classes)
Confusion Matrix and Statistics
```

Prediction \ Reference	red soil	cotton crop	grey soil	damp grey soil	vegetation stubble	very damp grey soil
red soil	301	0	3	1		
cotton crop	0	138	0	1		
grey soil	1	3	263	26		
damp grey soil	4	0	3	77		
vegetation stubble	14	2	0	0	0	
very damp grey soil	3	0	2	2	20	
Total	323	140	266	101	20	20

```

overall statistics

      Accuracy : 0.9213
      95% CI   : (0.9052, 0.9355)
    No Information Rate : 0.2383
    P-Value [Acc > NIR] : < 2.2e-16

      Kappa : 0.9025

  Mcnemar's Test P-value : NA

Statistics by class:

              Class: red soil Class: cotton crop Class: grey soil
Sensitivity    0.61600      0.9837      0.9857      0.97
Specificity    0.98533      0.9888      0.9974      0.96
Pos Pred value 0.81915      0.9647      0.9787      0.88
Neg Pred value 0.8971      0.96154     0.96154     0.96154

```

FIGURA 24 - Matriz de confusão para SVM

```

Neg Pred Value      0.95966      0.9949      0.9983      0.99
19
0.9774
Prevalence          0.09735      0.2383      0.1090      0.21
11
0.2344
Detection Rate      0.05997      0.2344      0.1075      0.20
48
0.2173
Detection Prevalence 0.07321      0.2430      0.1098      0.23
05
0.2422
Balanced Accuracy    0.80067      0.9862      0.9915      0.96
90
0.9472
> confusionMatrix(predicoes_svm, teste$classes)
Confusion Matrix and Statistics

Prediction          Reference
getation stubble very damp grey soil cotton crop grey soil damp grey soil ve
red soil          303
5 cotton crop      0          0          138          2          2
2 grey soil        2          2          0          261          27
0 damp grey soil   7          0          1          5          74
1 vegetation stubble 21         1          0          0          1
126 very damp grey soil 3          0          1          1          21
7          268

Overall Statistics

      Accuracy : 0.9112
      95% CI : (0.8943, 0.9262)
      No Information Rate : 0.2383
      P-Value [Acc > NIR] : < 2.2e-16

      Kappa : 0.8901

      McNemar's Test P-Value : NA

Statistics by Class:

      Class: red soil Class: cotton crop Class: grey so
il Class: damp grey soil Class: vegetation stubble Class: very damp gr
ey soil
Sensitivity      0.59200      0.9902      0.9857      0.96
31
0.8904
Specificity      0.97584      0.9928      0.9930      0.96
45
0.9695
Pos Pred Value    0.72549      0.9774      0.9452      0.87
88
0.8993
Neg Pred Value    0.95685      0.9969      0.9982      0.98
99
0.9665
Prevalence        0.09735      0.2383      0.1090      0.21
11
0.2344

```

FIGURA 25 - Matriz de confusão para RNA

```

Detection Rate      0.05763      0.2360      0.1075      0.20
33
0.2087
Detection Prevalence 0.07944      0.2414      0.1137      0.23
13
0.2321
Balanced Accuracy    0.78392      0.9915      0.9894      0.96
38
0.9299
> confusionMatrix(predicoes.rna, teste$classes)
Confusion Matrix and Statistics

              Reference
Prediction    red soil cotton crop grey soil damp grey soil ve
getation stubble very damp grey soil
red soil      272          3          1          0
2
cotton crop    0         13        136          7
95
grey soil      2          0          40          5
1
damp grey soil 3          0          0          0
0
vegetation stubble 0          0          0          0
1
very damp grey soil 0          1          229        113
42
296

Overall Statistics

          Accuracy : 0.5802
          95% CI : (0.5527, 0.6074)
    No Information Rate : 0.2383
    P-Value [Acc > NIR] : < 2.2e-16

          Kappa : 0.4692

  McNemar's Test P-Value : NA

Statistics by Class:

              Class: red soil Class: cotton crop Class: grey so
il Class: damp grey soil Class: vegetation stubble Class: very damp gr
ey soil
Sensitivity      0.00000      0.8889      0.9714      0.147
60
0.9834
Specificity      1.00000      0.9939      0.8969      0.971
37
0.6073
Pos Pred Value    NaN          0.9784      0.5354      0.579
71
0.4340
Neg Pred Value    0.90265      0.9662      0.9961      0.809
88
0.9917
Prevalence        0.09735      0.2383      0.1090      0.211
06
0.2344
Detection Rate    0.00000      0.2118      0.1059      0.031
15
0.2305
Detection Prevalence 0.00000      0.2165      0.1978      0.053
74
0.5312

```

Pelos resultados obtidos através da matriz de confusão, olhando na tabela o valor predito e o valor de referência, o algoritmo preditivo Random Forest apresentou menor quantidade de erros na identificação da classe de solo.

Também teve a maior das acurácias, com 92,13% de precisão.

2. ESTIMATIVA DE VOLUMES DE ÁRVORES

Aplicamos os algoritmos de Random Forest, SVM, RNA e Alométrico de Spurr nos dados de medição de árvores para estimar o volume. Avaliamos os modelos com R^2 , Syx e Syxp.

FIGURA 26 - Avaliação dos modelos RF, SVM, RNA e Alométrico de Spurr

R2.alom	0.835689472578945
R2.rf	0.853564671341675
R2.rna	0.881198253121334
R2.svm	0.848465216045881
Syx.alom	0.153121430017649
Syx.rf	0.144552702228724
Syx.rna	0.130201112432106
Syx.svm	0.147048110566199
Syxp.alom	11.7331760315766
Syxp.rf	11.0765834729613
Syxp.rna	9.97686980520566
Syxp.svm	11.2677981533029

Pelo coeficiente de determinação, erro padrão e erro padrão percentual observados dos quatro modelos avaliados, o RNA foi o que apresentou melhores resultados. Mas vale a menção que os valores obtidos estão relativamente próximos. São alternativas viáveis.

APÊNDICE 4 – ESTATÍSTICA APLICADA I

A – ENUNCIADO

1) Gráficos e tabelas

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

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

2) Medidas de posição e dispersão

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

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

3) Testes paramétricos ou não paramétricos

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

Obs:

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

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

B – RESOLUÇÃO

1. GRÁFICOS E TABELAS

GRÁFICO 8 - Idade média das esposas

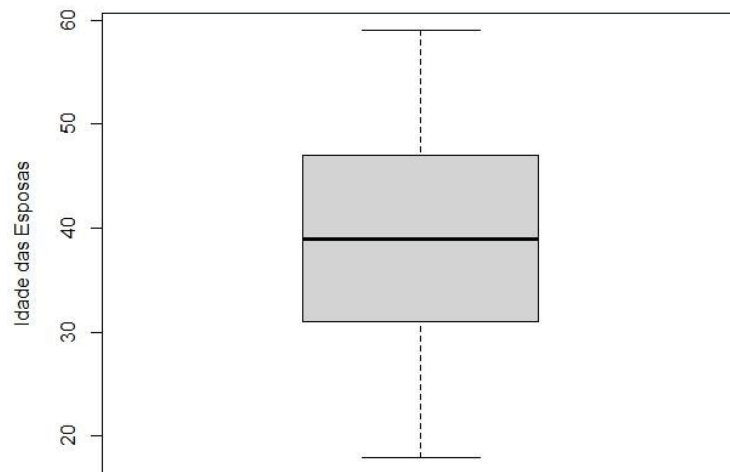


GRÁFICO 9 - Histograma salários das esposas

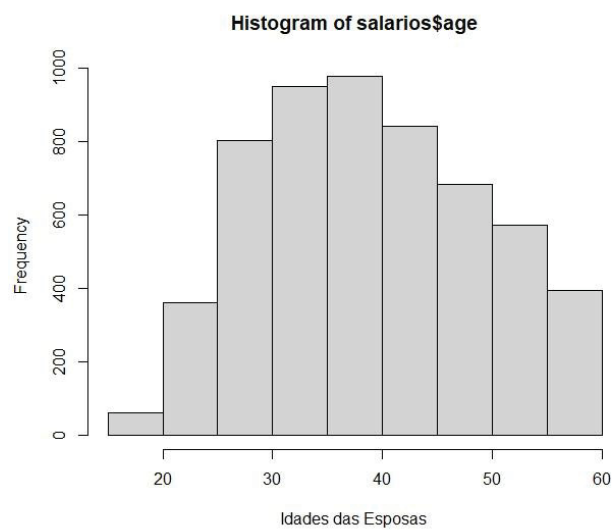


GRÁFICO 10 - Idade média dos maridos

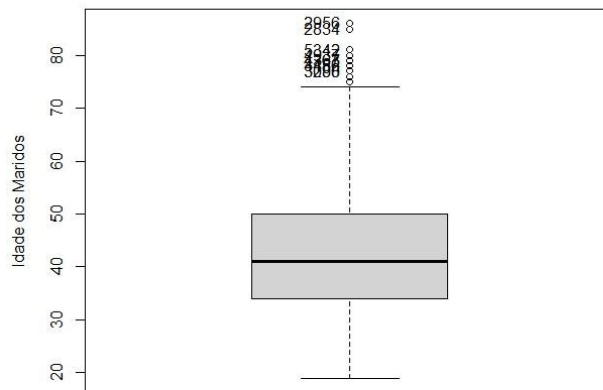
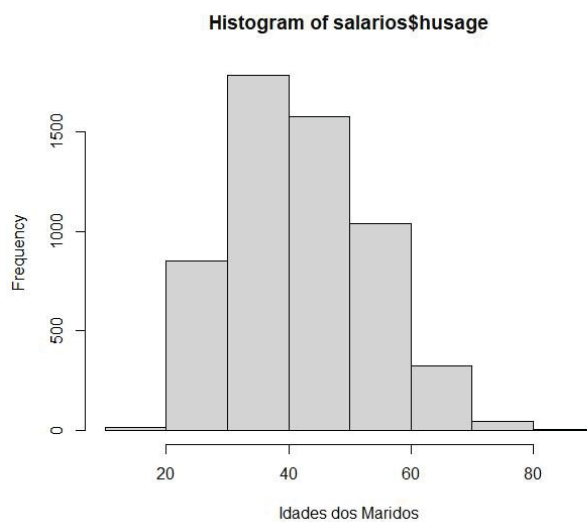


GRÁFICO 11 - Histograma dos salários dos maridos



Pelos gráficos podemos entender que os dados das Esposas estão mais distribuídos, mas não avançam mais que 60 anos (distância interquartílica está maior nas Esposas pelo gráfico). Os Maridos, por sua vez estão mais concentrados perto da média e tem outliers do gráfico. Chegando a idades de 70 a 80 anos. A distribuição tem curtose à direita.

Tabela de frequência das idades das Esposas.

<i>Class limits</i>	<i>f</i>	<i>rf</i>	<i>rf(%)</i>	<i>cf</i>	<i>cf(%)</i>
[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

Tabela de frequência das idades dos Maridos:

<i>Class limits</i>	<i>f</i>	<i>rf</i>	<i>rf(%)</i>	<i>cf</i>	<i>cf(%)</i>
[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

Pela tabela de frequência vemos a idade das Esposas estarem concentradas no intervalo de maior ou igual a 35,7 e menor que 38,7 anos. Os Maridos se concentram no intervalo maior ou igual a 38,2 e menor que 43,1 anos.

2. MEDIAS DE POSIÇÃO E DISPERSÃO

Para Esposas:

Média: 39,4 anos

Mediana: 39 anos

Moda: 37 anos

Para Maridos:

Media: 39 anos

Mediana: 41 anos

Moda: 44 anos

Os dados indicam que a média de idade das esposas é de 39,4 anos, com a mediana sendo 39 anos e a moda 37 anos. Isso sugere que a distribuição de idade das esposas é relativamente simétrica em torno da média, com uma leve concentração de valores em torno dos 37 anos. Corrobora com os gráficos do exercício anterior.

Para os maridos, a média de idade é de 39 anos, mas a mediana é mais alta, 41 anos, e a moda é ainda maior, 44 anos. Essa distribuição sugere uma assimetria, onde há uma concentração de maridos mais velhos, fazendo com que a mediana e a moda se desloquem para idades mais altas em comparação à média.

Fazendo comparação entre as médias 44 /37, temos que a idade dos maridos é 18,92% maior.

Para Esposas:

Variância: 99,8 anos

Desvio Padrão: 9,98 anos

Coefficiente Variância: 25,33%

Para Maridos:

Variância: 126,0 anos

Desvio Padrão: 11,22 anos

Coefficiente Variância: 26,44%

Para as esposas, os dados mostram uma variância de 99,8 anos e um desvio padrão de 9,98 anos, indicando que as idades estão dispersas em torno da média com essa variação padrão. O coeficiente de variação é de 25,33%, o que sugere uma dispersão moderada das idades em relação à média.

Para os maridos, a variância é de 126,0 anos e o desvio padrão é de 11,22 anos, o que aponta para uma dispersão maior das idades em comparação com as esposas. O coeficiente de variação é de 26,44%, indicando também uma dispersão moderada, mas ligeiramente maior em relação à média do que observado para as esposas. Esses dados ressaltam uma maior heterogeneidade nas idades dos maridos comparado às esposas.

26,38% mais variação de idade nos maridos e também há mais variação. O coeficiente de variação é 1 ponto percentual e alguns décimos maior nos maridos.

3. TESTES PARAMÉTRICOS OU NÃO PARAMÉTRICOS

Teste de normalidade Esposas:

```
Lilliefors (Kolmogorov-Smirnov) normality test
data:  salarios$husage
D = 0.059662, p-value < 0.000000000000000022
```

Teste de normalidade Maridos:

```
Lilliefors (Kolmogorov-Smirnov) normality test
data:  salarios$age
D = 0.058909, p-value < 0.000000000000000022
```

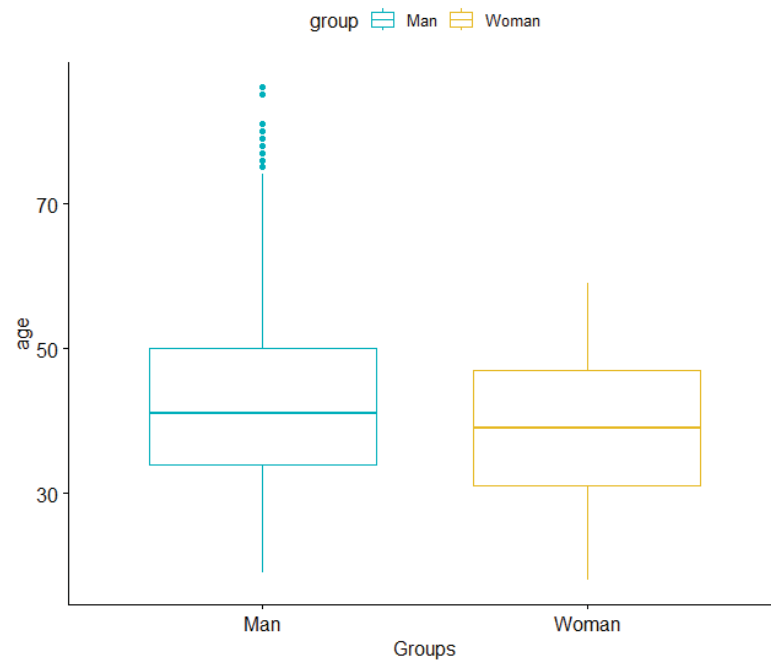
Regra de bolso: Como o p-value não é superior a 0.05, nenhuma das variáveis tem distribuição normal. Por conseguinte, seguiremos para um teste não paramétrico.

Relembrando o gráfico boxplot de ambos comparativamente testamos a hipóteses:

H0: As medianas de idade são iguais

Há: As medianas de idade não são iguais

GRÁFICO 12 - Teste de hipótese das medianas



Aplicando o teste.

Wilcoxon rank sum test with continuity correction

```
data: age by group
W = 18122044, p-value < 0.00000000000000022
alternative hypothesis: true location shift is not equal to 0
95 percent confidence interval:
 2.000033 3.000024
sample estimates:
difference in location
 2.999966
```

O teste de soma de postos de Wilcoxon com correção de continuidade mostra que existe uma diferença estatisticamente significativa entre as idades dos dois grupos analisados. O valor de W é 18122044 e o valor de p é extremamente baixo (< 0.00000000000000022), indicando que a diferença observada entre os grupos é muito improvável de ocorrer por acaso.

A hipótese alternativa de que as medianas são diferentes é suportada. O intervalo de confiança de 95% para a diferença de localização entre os grupos é de aproximadamente 2,00 a 3,00 anos, e a estimativa pontual dessa diferença é de aproximadamente 2,999966 anos. Isso sugere que um dos grupos tende a ter uma idade média significativamente maior que o outro por cerca de 3 anos.

APÊNDICE 5 – ESTATÍSTICA APLICADA II

A – ENUNCIADO

Regressões Ridge, Lasso e ElasticNet

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

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

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

B – RESOLUÇÃO

1. REGRESSÃO RIDGE, LASSO E ELASTICNET

Visão geral dos dados carregados:

```

Rows: 2,574
Columns: 17
$ husage <dbl> 56, 31, 33, 34, 42, 45, 33, 31, 31, 44, 45, 22, 66, 43, 26,
37, 51, 38, 37, 54, 29, 35, 38, 44, 31, 40, 33, 34, 53, 25, 26, 53, 3...
$ husunion <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, ...
$ husearns <dbl> 1500, 800, 950, 1000, 730, 1154, 1350, 769, 340, 750, 1200,
249, 500, 400, 520, 500, 500, 1500, 465, 300, 600, 600, 240, 1450, 67...
$ huseduc <dbl> 14, 17, 13, 14, 14, 16, 16, 18, 12, 12, 12, 12, 12, 16, 12, 14,
12, 12, 16, 12, 18, 16, 16, 18, 18, 16, 18, 12, 18, 12, 11, 15, 17, 1...
$ husblack <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, ...
$ hushisp <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ hushrs <dbl> 40, 40, 60, 50, 40, 38, 40, 55, 40, 40, 50, 40, 40, 50, 36,
50, 40, 40, 50, 60, 50, 42, 48, 40, 40, 48, 35, 60, 0, 40, 40, 91, 50...
$ kidge6 <dbl> 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, ...
$ earns <dbl> 100, 480, 455, 102, 300, 425, 770, 125, 245, 539, 300, 299,
500, 260, 345, 213, 320, 550, 687, 400, 460, 600, 400, 1100, 434, 100...
$ age <dbl> 49, 29, 30, 31, 41, 45, 32, 27, 30, 42, 42, 23, 55, 31, 27,
30, 49, 38, 32, 52, 27, 31, 36, 45, 32, 38, 31, 33, 45, 33, 24, 52, 2...
$ black <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ educ <dbl> 12, 14, 12, 12, 12, 18, 12, 14, 15, 12, 12, 13, 12, 12, 14,
12, 12, 16, 17, 18, 14, 12, 16, 18, 14, 18, 12, 18, 12, 10, 12, 18, 1...
$ hispanic <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ union <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ exper <dbl> 31, 9, 12, 13, 23, 21, 14, 7, 9, 24, 24, 4, 37, 13, 7, 12,
31, 16, 9, 28, 7, 13, 14, 21, 12, 14, 13, 9, 27, 17, 6, 28, 10, 4, 15,...
$ kidlt6 <dbl> 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0,
1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
$ lwage <dbl> 1.897120, 2.484907, 2.431418, 1.629241, 2.302585, 2.496741,
2.957511, 1.783791, 1.945910, 2.600836, 2.014903, 2.011564, 2.525729,...
```

Tamanho dos grupos na separação de 80/20:

Treino

[1] 2059 17

Teste

[1] 515 17

Avaliação pelo modelo de penalidade Ridge

Penalidade: Adiciona uma penalidade do tipo L2 (soma dos quadrados dos coeficientes) ao termo de erro.

Lambda ótimo do modelo Ridge:

0.03162278

Betas das variáveis pelo modelo Ridge

```
15 x 1 sparse Matrix of class "dgCMatrix"
      s0
husage -0.004619353
husearns 0.230244252
huseduc 0.051374229
hushrs -0.069514275
age 0.069223380
educ 0.327075655
husblck 0.217248348
hushisp 0.086154433
kidge6 -0.181933507
black -0.277286478
hispanic -0.011410377
union 0.365674748
kidlt6 -0.042468361
husunion 0.001290081
exper -0.016345064
```

Betas baixos indicam variáveis menos significativas na explicação da variável independente lwage.

Avaliando o erro quadrático médio e o coeficiente de determinação (R^2):

TABELA 1 - Erro quadrático médio e R^2

RMSE	Rsquare
0.8411446	0.2921319

Resultado: O valor do salário-hora é de 9.422981 dólares, pelo modelo de regressão com penalidade de Ridge. Em relação ao intervalo de confiança temos inferior em 9.401122 dólares e superior em 9.444839 dólares.

Avaliação pelo modelo de penalidade Lasso

Penalidade: Adiciona uma penalidade do tipo L1 (soma dos valores absolutos dos coeficientes) ao termo de erro.

Lambda ótimo do modelo Lasso:

0.01258925

Betas das variáveis pelo modelo Lasso

```
15 x 1 sparse Matrix of class "dgCMatrix"
s0
husage      .
husearns    0.23003841
huseduc     0.03008230
hushrs     -0.06025180
age         0.04689039
educ        0.33916643
husblack    .
hushisp     .
kidge6     -0.14252071
black       -0.03169357
hispanic    .
union       0.34330221
kidlt6      .
husunion    .
exper       .
```

Variáveis com. são não significativas no modelo preditivo.

Avaliando o erro quadrático médio e o coeficiente de determinação (R^2).

TABELA 2 - Erro quadrático médio e R^2

RMSE	R^2
0.8423523	0.2900977

Resultado: O valor do salário-hora é de 8.410696 dólares, pelo modelo de regressão com penalidade de Lasso (Least Absolute Shrinkage and Selection Operator). Em relação ao intervalo de confiança temos inferior em 8.388838 dólares e superior em 8.432554 dólares.

Avaliação pelo modelo de penalidade Elastic Net

Penalidade: Combina as penalidades L1 (Lasso) e L2 (Ridge)

Melhor estimativa do modelo para alfa e lambda.

TABELA 3 - Alfa e lambda para Lasso e Ridge

alpha	lambda
0.7429763	0.01233549

Betas do modelo:

```
15 x 77 sparse Matrix of class "dgCMatrix"
[[ suppressing 67 column names 's0', 's1', 's2' ... ]]
[[ suppressing 67 column names 's0', 's1', 's2' ... ]]

husage      . .      .      .      .      .      .      .
.           .      .      .      .      .      .      .
husearns    . .      .      .      .      .      .      .
0.05963892  0.0750015  0.08913756  0.1021342  0.1140741  0.1250595  0.1351556
huseduc     . .      .      .      .      .      .      .
.           .      .      .      .      .      .      .
hushrs      . .      .      .      .      .      .      .
.           .      .      .      .      .      .      .
age         . .      .      .      .      .      .      .
.           .      .      .      .      .      .      .
educ        . 0.03507813 0.06778955 0.0982384 0.125145010 0.14617055 0.16558804
0.18350187  0.2000085  0.21520547 0.2291845 0.2420330 0.2531897 0.2625779
husblck     . .      .      .      .      .      .      .
.           .      .      .      .      .      .      .

husage      .      .      .      .      .      .      .
.           .      .      .      .      .      .      .
husearns    0.1444226 0.1529183 0.1607027 0.1678319 0.173833446 0.179156492
0.1840182380 0.188394337 0.192895375 0.19701293 0.20076870 0.20420169
huseduc     .      .      .      .      .      .      .
0.0070671711 0.009692001 0.012130047 0.01426931 0.01627562 0.01810500
hushrs      .      .      .      .      .      .      .
-0.003737944 -0.009837325 -0.01541803 -0.02051807 -0.02518015
age         .      .      .      .      .      .      .
0.0009931822 0.006234683 0.010683650 0.01475248 0.01846884 0.02186285
educ        0.2711740 0.2790431 0.2862438 0.2928303 0.297810945 0.302025726
0.3057216471 0.309131551 0.312204359 0.31505880 0.31763223 0.31997957
husblck     .      .      .      .      .      .      .
.           .      .      .      .      .      .      .
```

```

husage . . . . .
. . . . .
husearns 0.20733894 0.21020519 0.21282324 0.21521407 0.21739697
0.21938968 0.22120847 0.22286826 0.22425417 0.22549168 0.22661338
huseduc 0.01977252 0.02129243 0.02267775 0.02394035 0.02509108
0.02613982 0.02709559 0.02796661 0.02880179 0.02942634 0.03008557
hushrs -0.02944071 -0.03333332 -0.03688891 -0.04013596 -0.04310064 -
0.04580704 -0.04827723 -0.05053150 -0.05259683 -0.05448882 -0.05621173
age 0.02496184 0.02779091 0.03037314 0.03272971 0.03488003
0.03684191 0.03863164 0.04026417 0.04173584 0.04307650 0.04429662
educ 0.32212067 0.32407348 0.32585439 0.32747840 0.32895922
0.33030941 0.33154039 0.33266264 0.33364741 0.33462081 0.33545519
husblck . . . . .
. . . . .

husage . . . . .
. . . . .
husearns 0.22763557 0.22856772 0.22940155 0.23011730 0.23076951
0.23136351 0.23190493 0.23239844 0.23285117 0.23326236 0.23363590
huseduc 0.03068984 0.03124069 0.03186811 0.03263283 0.03339727
0.03409753 0.03473628 0.03531876 0.03575080 0.03623007 0.03667901
hushrs -0.05778305 -0.05921617 -0.06050120 -0.06158733 -0.06257580 -
0.06347681 -0.06429821 -0.06504699 -0.06573242 -0.06635490 -0.06692181
age 0.04540920 0.04642363 0.04727635 0.04754591 0.04777970
0.04799196 0.04818484 0.04836014 0.04854642 0.04869006 0.04881966
educ 0.33621383 0.33690521 0.33757363 0.33832341 0.33896436
0.33954668 0.34007726 0.34056075 0.34105880 0.34146336 0.34182488
husblck . . . . .
. . . . .

husage . . . . .
. . . . .
husearns 0.23397613 0.234302403 0.23457019 0.23481649 0.23503758
0.23524178 0.23542489 0.23559450 0.23574624 0.23588727 0.23601308
huseduc 0.03708983 0.037639033 0.03815880 0.03862704 0.03906080
0.03945096 0.03981272 0.04013708 0.04043848 0.04070774 0.04095866
hushrs -0.06743847 -0.067899109 -0.06831098 -0.06868668 -0.06902866 -
0.06934063 -0.06962457 -0.06988364 -0.07011939 -0.07033455 -0.07053028
age 0.04893746 0.049016780 0.04910094 0.04917805 0.04924724
0.04931099 0.04936814 0.04942096 0.04946823 0.04951205 0.04955117
educ 0.34215342 0.342352008 0.34251725 0.34267117 0.34280687
0.34293384 0.34304554 0.34315074 0.34324284 0.34333025 0.34340631
husblck . 0.007295743 0.03634675 0.06238293 0.08708525
0.10887744 0.12959437 0.14773539 0.16506953 0.18012347 0.19460069

husage . . . -0.0002716424 -0.002158684 -0.00358911 -
0.00471071 -0.005783262 -0.006661823 .....
husearns 0.23613051 0.23623490 0.2363815821 0.236542564 0.23667499
0.23679976 0.236911010 0.237016057 .....
huseduc 0.04118170 0.04139039 0.0415540236 0.041691684 0.04181624
0.04192557 0.042029550 0.042116924 .....
hushrs -0.07070898 -0.07087150 -0.0710544985 -0.071224076 -0.07137585 -
0.07151563 -0.071642336 -0.071758936 .....
age 0.04958759 0.04962000 0.0498831915 0.051553402 0.05282458
0.05382458 0.054779523 0.055564264 .....
educ 0.34347922 0.34354216 0.3436012153 0.343647277 0.34369778
0.34374575 0.343786969 0.343828708 .....
husblck 0.20704468 0.21910724 0.2293417339 0.239228152 0.24848356
0.25603883 0.263460313 0.269436473 .....

```



```

kidlt6      -0.01921953  -0.02145752  -0.02350105  -0.02536472  -0.02712410  -
0.02867070  -0.03007995  -0.03136525  -0.03253623  -0.03360417  -0.03457696
husunion    .          .          .          .          .          .
.           .          .          .          .          .
exper       .          .          .          .          .          .
.           .          .          .          .          .

kidge6      -0.17914455  -0.1799848  -0.18075183  -0.18144961  -0.18208675  -
0.18266039  -0.18322243  -0.18373148  -0.18418786  -0.18460658  -0.18498279  .....
black       -0.22475129  -0.2398732  -0.25438012  -0.26688641  -0.27897455  -
0.28925476  -0.29913493  -0.30837523  -0.31596059  -0.32338481  -0.32940138  .....
hispanic    .          .          .          .          .          .
.           .          .          .          .          .
union       0.36696743  0.3674212  0.36783037  0.36820759  0.36854727  0.36885728
0.36911340  0.36934646  0.36956744  0.36976486  0.36995081  .....
kidlt6      -0.03546417  -0.0362722  -0.03700918  -0.03768022  -0.03829234  -
0.03886517  -0.03958127  -0.04020237  -0.04074453  -0.04124557  -0.04168828  .....
husunion    .          .          .          .          .          .
.           .          .          .          .          .
exper       .          .          .          .          .          .
.           .          .          .          .          .

```

Avaliando o erro quadrático médio e o coeficiente de determinação (R^2).

TABELA 4 - Erro quadrático médio e R^2 para Elastic Net

RMSE	R^2
0.8419627	0.2907543

Resultado: O valor do salário-hora é de 8.003162 dólares, pelo modelo de regressão com penalidade de Elastic Net. Em relação ao intervalo de confiança temos inferior em é 7.981303 dólares e superior em 8.02502 dólares.

Conclusão

Avaliando os modelos pelo menor erro quadrático médio e/ou maior coeficiente, temos:

TABELA 5 - Comparativo entre modelos

Modelo	RMSE	R^2
Ridge	0.9893314	0.2590861
Lasso	0.9899393	0.2581753
Elastic Net	0.9893725	0.2590245

Logo o modelo de regressão linear com penalidade escolhido foi o Ridge.

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 IDIOMAS

Usando um array com frases em português, inglês e espanhol, aplicou-se um pré-padrão.

```
[[ 'Where is the nearest restaurant?', 'inglês'], [ 'Estoy estudiando para mis exámenes.', 'espanhol'], [ '¿Cuál es tu comida favorita?', 'espanhol'], [ 'Meu time de futebol favorito ganhou o jogo.', 'português'], [ '¿Puedes ayudarme con esto?', 'espanhol'], [ 'What is your favorite color?', 'inglês'], [ 'Vou visitar meus avós no domingo.', 'português'] ...
```

Para definir atributos que auxiliaram na correta identificação do idioma, foram escolhidos três tipos de características, a saber: 1) Tamanho médio das frases; 2) Palavras comuns e 3) Dígrafos. Com as três funções bem definidas, aplicamos as frases e criamos variáveis numéricas.

FIGURA 27 - Atribuição de características

	0	1	2	3
0	4.250000	0	0	inglês
1	4.375000	1	1	português
2	4.625000	1	1	português
3	6.000000	0	0	espanhol
4	4.200000	0	0	inglês
...	...	...	...	...
87	3.833333	3	0	espanhol
88	5.857143	2	0	inglês
89	7.800000	0	0	espanhol
90	4.500000	1	0	português
91	3.500000	2	2	inglês

92 rows × 4 columns

Definimos os dados para treinamento e teste (75/25) e treinamos com o algoritmo SVM. Avaliamos o modelo de predição com métricas de f1 score e matriz de confusão.

FIGURA 28 - F1 Score e Matriz de confusão para SVM

```

Acurácia nos dados de treinamento: 81.16%
[[14  0  8]
 [ 1 19  3]
 [ 1  0 23]]

```

	precision	recall	f1-score	support
espanhol	0.88	0.64	0.74	22
inglês	1.00	0.83	0.90	23
português	0.68	0.96	0.79	24
accuracy			0.81	69
macro avg	0.85	0.81	0.81	69
weighted avg	0.85	0.81	0.81	69

```

métricas mais confiáveis
[[6 0 2]
 [0 3 4]
 [0 0 8]]

```

	precision	recall	f1-score	support
espanhol	1.00	0.75	0.86	8
inglês	1.00	0.43	0.60	7
português	0.57	1.00	0.73	8
accuracy			0.74	23
macro avg	0.86	0.73	0.73	23
weighted avg	0.85	0.74	0.73	23

2. MELHORE UMA BASE DE DADOS RUIM

Carregamos uma base de dados de risco a saúde materna (Maternal Health Risk). Primeiro entendemos os dados.

Shape.
(1014, 7)

Estrutura.

FIGURA 29 - Estrutura da base de saúde materna

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1014 entries, 0 to 1013
Data columns (total 7 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Age              1014 non-null   int64
1   SystolicBP       1014 non-null   int64
2   DiastolicBP      1014 non-null   int64
3   BS               1014 non-null   float64
4   BodyTemp         1014 non-null   float64
5   HeartRate        1014 non-null   int64
6   RiskLevel        1014 non-null   object
dtypes: float64(2), int64(4), object(1)
memory usage: 55.6+ KB
```

Contagem de risco

FIGURA 30 – Avaliação de risco

```
RiskLevel
low risk    406
mid risk    336
high risk   272
Name: count, dtype: int64
```

Percentual de risco

FIGURA 31 - Risco percentual

```
RiskLevel
low risk    0.400394
mid risk    0.331361
high risk    0.268245
Name: count, dtype: float64
```

Somatória dos valores nulos por coluna

FIGURA 32 - Checagem de nulos por coluna

```
Age          0
SystolicBP   0
DiastolicBP  0
BS           0
BodyTemp     0
HeartRate    0
RiskLevel    0
dtype: int64
```

Estatísticas

FIGURA 33 - Estatísticas das variáveis

	Age	SystolicBP	DiastolicBP	BS	BodyTemp	HeartRate
count	1014.00	1014.0	1014.00	1014.00	1014.00	1014.00
mean	29.87	113.2	76.46	8.73	98.67	74.30
std	13.47	18.4	13.89	3.29	1.37	8.09
min	10.00	70.0	49.00	6.00	98.00	7.00
25%	19.00	100.0	65.00	6.90	98.00	70.00
50%	26.00	120.0	80.00	7.50	98.00	76.00
75%	39.00	120.0	90.00	8.00	98.00	80.00
max	70.00	160.0	100.00	19.00	103.00	90.00

Agora começamos os tratamentos. Separamos a classe alvo dos atributos. Normalizamos os dados.

FIGURA 34 - Separação do alvo para treinamento

	Age	SystolicBP	DiastolicBP	BS	BodyTemp	HeartRate
0	25	130	80	15.0	98.0	86
1	35	140	90	13.0	98.0	70
2	29	90	70	8.0	100.0	80
3	30	140	85	7.0	98.0	70
4	35	120	60	6.1	98.0	76
['high risk' 'low risk' 'mid risk']						
	Age	SystolicBP	DiastolicBP	BS	BodyTemp	HeartRate
0	25	130	80	15.0	98.0	86
1	35	140	90	13.0	98.0	70
2	29	90	70	8.0	100.0	80
3	30	140	85	7.0	98.0	70
4	35	120	60	6.1	98.0	76
	0	1	2	3	4	5
0	0.250000	0.666667	0.607843	0.692308	0.0	0.951807
1	0.416667	0.777778	0.803922	0.538462	0.0	0.759036
2	0.316667	0.222222	0.411765	0.153846	0.4	0.879518
3	0.333333	0.777778	0.705882	0.076923	0.0	0.759036
4	0.416667	0.555556	0.215686	0.007692	0.0	0.831325

Separamos dados de treinamento e teste. Estatísticas dos dados de treinamento.

FIGURA 35 - Separação entre teste e treinamento

	0	1	2	3	4	5
count	7.600000e+02	7.600000e+02	7.600000e+02	7.600000e+02	7.600000e+02	7.600000e+02
mean	-8.414322e-17	-1.869849e-17	3.038505e-16	-9.349247e-18	5.609548e-17	3.085251e-16
std	1.000659e+00	1.000659e+00	1.000659e+00	1.000659e+00	1.000659e+00	1.000659e+00
min	-1.478012e+00	-2.354208e+00	-1.967992e+00	-8.267631e-01	-4.796526e-01	-1.946725e+00
25%	-8.027779e-01	-7.293385e-01	-8.206727e-01	-5.568988e-01	-4.796526e-01	-6.151517e-01
50%	-3.526220e-01	3.539080e-01	2.549388e-01	-3.769893e-01	-4.796526e-01	1.837921e-01
75%	5.476896e-01	3.539080e-01	9.720131e-01	-2.270647e-01	-4.796526e-01	7.164213e-01
max	2.723443e+00	2.520401e+00	1.689087e+00	3.071276e+00	3.221972e+00	2.047994e+00

Fizemos então o treinamento usando o algoritmo SVM com os dados originais, sem normalização. A matriz de confusão avalia o modelo.

FIGURA 36 - Matriz de confusão do SVM

Matriz de confusão - com os dados ORIGINAIS usados no TREINAMENTO

```
[[128 53 23]
 [ 4 272 28]
 [ 12 161 79]]
```

	precision	recall	f1-score	support
high risk	0.89	0.63	0.74	204
low risk	0.56	0.89	0.69	304
mid risk	0.61	0.31	0.41	252
accuracy			0.63	760
macro avg	0.69	0.61	0.61	760
weighted avg	0.66	0.63	0.61	760

Matriz de confusão - com os dados ORIGINAIS usados para TESTES

```
[[37 22 9]
 [ 1 89 12]
 [ 9 58 17]]
```

	precision	recall	f1-score	support
high risk	0.79	0.54	0.64	68
low risk	0.53	0.87	0.66	102
mid risk	0.45	0.20	0.28	84
accuracy			0.56	254
macro avg	0.59	0.54	0.53	254
weighted avg	0.57	0.56	0.53	254

Com normalização, fizemos um novo treinamento. A matriz de confusão mostra uma evolução.

FIGURA 37 - Matriz de confusão do novo treinamento

Matriz de confusão - com os dados TRATADOS usados no TREINAMENTO

```
[[173 13 18]
 [ 10 265 29]
 [ 17 126 109]]
```

	precision	recall	f1-score	support
high risk	0.86	0.85	0.86	204
low risk	0.66	0.87	0.75	304
mid risk	0.70	0.43	0.53	252
accuracy			0.72	760
macro avg	0.74	0.72	0.71	760
weighted avg	0.73	0.72	0.71	760

Matriz de confusão - com os dados ORIGINAIS usados para TESTES

```
[[57 5 6]
 [ 5 87 10]
 [13 43 28]]
```

	precision	recall	f1-score	support
high risk	0.76	0.84	0.80	68
low risk	0.64	0.85	0.73	102
mid risk	0.64	0.33	0.44	84
accuracy			0.68	254
macro avg	0.68	0.67	0.66	254
weighted avg	0.67	0.68	0.65	254

Tratamos os dados com hiper parâmetros. Treinamos novamente. A matriz de confusão.

FIGURA 38 - Matriz de confusão com hiper parâmetros

Matriz de confusão - com os dados TRATADOS usados no TREINAMENTO

```
[[188  6 10]
 [  8 264 32]
 [ 14  61 177]]
```

	precision	recall	f1-score	support
high risk	0.90	0.92	0.91	204
low risk	0.80	0.87	0.83	304
mid risk	0.81	0.70	0.75	252
accuracy			0.83	760
macro avg	0.83	0.83	0.83	760
weighted avg	0.83	0.83	0.83	760

Matriz de confusão - com os dados ORIGINAIS usados para TESTES

```
[[59  4  5]
 [  5 80 17]
 [15 24 45]]
```

	precision	recall	f1-score	support
high risk	0.75	0.87	0.80	68
low risk	0.74	0.78	0.76	102
mid risk	0.67	0.54	0.60	84
accuracy			0.72	254
macro avg	0.72	0.73	0.72	254
weighted avg	0.72	0.72	0.72	254

Mudando o algoritmo para SVC melhoramos ainda mais os resultados, conforme matriz de confusão.

FIGURA 39 - Matriz de confusão para SVC

Matriz de confusão - com os dados TRATADOS usados no TREINAMENTO

```
[[188  3 13]
 [  5 264 35]
 [  8  40 204]]
```

	precision	recall	f1-score	support
high risk	0.94	0.92	0.93	204
low risk	0.86	0.87	0.86	304
mid risk	0.81	0.81	0.81	252
accuracy			0.86	760
macro avg	0.87	0.87	0.87	760
weighted avg	0.86	0.86	0.86	760

Matriz de confusão - com os dados ORIGINAIS usados para TESTES

```
[[58  5  5]
 [  5 76 21]
 [10 21 53]]
```

	precision	recall	f1-score	support
high risk	0.79	0.85	0.82	68
low risk	0.75	0.75	0.75	102
mid risk	0.67	0.63	0.65	84
accuracy			0.74	254
macro avg	0.74	0.74	0.74	254
weighted avg	0.73	0.74	0.73	254

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

Aplicação de algoritmos de predição usando classificação em Veículos:

TABELA 6 - Aplicação dos algoritmos de classificação em veículos

Técnica	Parâmetro	Acurácia	Matriz de Confusão
RNA – CV	size=21, decay=0.4	0.8443	<pre> Reference Prediction bus opel saab van bus 41 1 0 0 opel 0 39 12 0 saab 0 11 31 0 van 2 0 0 39 </pre>
RF – Hold-out	mtry=10	0.7725	<pre> Reference Prediction bus opel saab van bus 41 0 1 0 opel 0 27 17 0 saab 0 12 23 1 van 2 3 2 38 </pre>
RF – CV	mtry=10	0.7665	<pre> Reference Prediction bus opel saab van bus 41 0 0 0 opel 0 26 18 0 saab 0 13 23 1 van 2 3 2 38 </pre>
SVM – CV	C=1, Sigma= 0.07587065	0.7425	<pre> Reference Prediction bus opel saab van bus 40 0 1 0 opel 0 25 22 0 saab 0 14 20 0 van 3 3 0 39 </pre>
SVM – Hold-out	C=1, Sigma= 0.07587065	0.7425	<pre> Reference Prediction bus opel saab van bus 40 0 1 0 opel 0 25 22 0 saab 0 14 20 0 van 3 3 0 39 </pre>
RNA – Hold-out	size=5, decay=0.1	0.6347	<pre> Reference Prediction bus opel saab van bus 39 2 3 0 opel 2 31 36 5 saab 1 1 3 1 van 1 0 1 33 </pre>
KNN	k=1	0.6294	<pre> Reference Prediction bus opel saab van bus 38 4 5 2 opel 2 14 20 1 saab 1 21 16 0 van 3 4 0 39 </pre>

Predição de novos casos:

FIGURA 40 - Predição de novos casos

	Comp	Circ	DCirc	RadRa	PrAxisRa	MaxLRa	ScatRa	Elong	PrAxisRect	MaxLRect	ScVarMaxis	ScVarnaxis	RaGyr	SkewMaxis	Skewmaxis	Kurtmaxis	KurtMaxis	HollRa	predict.rna
1	98	53	87	181	78	11	165	43	23	165	177	381	189	75	7	18	189	198	van
2	89	37	72	156	45	7	146	42	16	139	165	325	151	67	8	12	181	197	saab
3	106	58	89	210	87	12	211	36	28	155	219	648	185	76	7	7	186	193	bus

Lista de comandos do algoritmo escolhido:

FIGURA 41 - Lista de comandos

```
##### RNA - CV
### Instalação dos pacotes necessários
#install.packages("e1071")
#install.packages("caret")
#install.packages("mlbench")
#install.packages("mice")
library(mlbench)
library("caret")

## Leitura dos dados
dados <- read.csv("6 - Veiculos - Dados.csv")

### Retira o atributo a
dados$a <- NULL
View(dados)

### Cria um arquivo com treino com 80% e teste com 20% das linhas de forma randomizada
set.seed(202475)
indices <- createDataPartition(dados$tipo, p=0.80,list=FALSE)
treino <- dados[indices,]
teste <- dados[-indices,]

### indica o método cv e numero de folders 10
ctrl <- trainControl(method = "cv", number = 10)

### executa a RNA com esse ctrl
set.seed(202475)
rna <- train(tipo~, data=treino, method="nnet",trace=FALSE, trControl=ctrl)
predict.rna <- predict(rna, teste)
confusionMatrix(predict.rna, as.factor(teste$tipo))

#parametrização da RNA
### size, decay
grid <- expand.grid(size = seq(from = 1, to = 35, by = 10), decay = seq(from = 0.1, to = 0.6, by = 0.3))
set.seed(202475)
rna <- train(
  form = tipo~,
  data = treino,
  method = "nnet",
  tuneGrid = grid,
  trControl = ctrl,
  maxit = 2000,trace=FALSE)

# Exibir o modelo treinado
rna

### Faz as predições e mostra matriz de confusão
predict.rna <- predict(rna, teste)
confusionMatrix(predict.rna, as.factor(teste$tipo))

### PREDIÇÕES DE NOVOS CASOS
dados_novos_casos <- read.csv("6 - Veiculos - Novos Dados.csv")
View(dados_novos_casos)

predict.rna <- predict(rna, dados_novos_casos)
dados_novos_casos$tipo <- NULL
resultado <- cbind(dados_novos_casos, predict.rna)
resultado
```

Aplicação de algoritmos de predição usando classificação em Diabetes:

FIGURA 42 - Aplicação de algoritmos de classificação em dados de Diabetes

Técnica	Parâmetro	Acurácia	Matriz de Confusão
RNA – CV	size=11, decay=0.1	0.7647	Reference Prediction neg pos neg 89 25 pos 11 28
RF – Hold-out	mtry=2	0.7582	Reference Prediction neg pos neg 85 22 pos 15 31
RF – CV	mtry=5	0.7451	Reference Prediction neg pos neg 84 23 pos 16 30
RNA – Hold-out	size=3, decay=0.1	0.7451	Reference Prediction neg pos neg 82 21 pos 18 32
SVM – CV	C=0.25, Sigma= 0.1483148	0.732	Reference Prediction neg pos neg 86 27 pos 14 26
SVM – Hold-out	C=0.25, Sigma= 0.1483148	0.732	Reference Prediction neg pos neg 86 27 pos 14 26
KNN	k=9	0.7143	Reference Prediction neg pos neg 78 32 pos 12 32

Predição de novos casos:

FIGURA 43 - Predição de novos casos

```

preg0nt glucose pressure triceps insulin mass pedigree age predict.rna
1      2    145      75     38      0    34     0.20 32      neg
2      3     90     62     33    168    27     2.29 21      pos
3      6    110     68      2     88    23     0.36 26      neg

```

Lista de comandos do algoritmo escolhido:

FIGURA 44 - Lista de comandos algoritmo escolhido

```

### Instalação dos pacotes necessários
#install.packages("e1071")
#install.packages("caret")
#install.packages("mlbench")
#install.packages("mice")
library(mlbench)
library("caret")

## Leitura dos dados
dados <- read.csv("10 - Diabetes - Dados.csv")

### Retira o atributo num
dados$num <- NULL
View(dados)

### Cria um arquivo com treino com 80% e teste com 20% das linhas de forma randomizada
set.seed(202475)
indices <- createDataPartition(dados$diabetes, p=0.80,list=FALSE)
treino <- dados[indices,]
teste <- dados[-indices,]

### indica o método cv e numero de folders 10
ctrl <- trainControl(method = "cv", number = 10)

### executa a RNA com esse ctrl
set.seed(202475)
rna <- train(diabetes~., data=treino, method="nnet",trace=FALSE, trControl=ctrl)
predict.rna <- predict(rna, teste)
confusionMatrix(predict.rna, as.factor(teste$diabetes))

#parametrização da RNA
### size, decay
grid <- expand.grid(size = seq(from = 1, to = 35, by = 10), decay = seq(from = 0.1, to = 0.6, by = 0.3))
set.seed(202475)
rna <- train(
  form = diabetes~.,
  data = treino ,
  method = "nnet" ,
  tuneGrid = grid ,
  trControl = ctrl ,
  maxit = 2000,trace=FALSE)

# Exibir o modelo treinado
rna
|
### Faz as predições e mostra matriz de confusão
predict.rna <- predict(rna, teste)
confusionMatrix(predict.rna, as.factor(teste$diabetes))

### PREDIÇÕES DE NOVOS CASOS
dados_novos_casos <- read.csv("10 - Diabetes - Novos Dados.csv")
View(dados_novos_casos)

predict.rna <- predict(rna, dados_novos_casos)
dados_novos_casos$diabetes <- NULL
resultado <- cbind(dados_novos_casos, predict.rna)
resultado

```

2. REGRESSÃO

Aplicação de algoritmos de predição usando regressão em problemas de admissão:

FIGURA 45 - Aplicação de algoritmos de predição de regressão

Admissão						
Técnica	Parâmetro	R2	Syx	Pearson	Rmse	MAE
RNA – CV	size=10, decay=0.1	0.8172428	0.061240715	0.90526694	0.060612589	0.045849687
RF – Hold-out	mtry=2	0.8114712	0.062200211	0.90167994	0.061562243	0.042709547
RF – CV	mtry=2	0.80952613	0.062520251	0.90078255	0.061879	0.042564919
SVM – Hold-out	C=0.5, Sigma=0.174971	0.80723905	0.062894481	0.90227164	0.062249393	0.043032732
SVM – CV	C=1, Sigma=0.174971	0.80647206	0.063019484	.90127519	0.062373113	0.042353704
RNA – Hold-out	size=5, decay=0.1	0.80180708	0.063774502	0.89829229	0.063120388	0.049289792
KNN	K=9	0.71356658	0.076668127	0.84645202	0.075881766	0.057946712

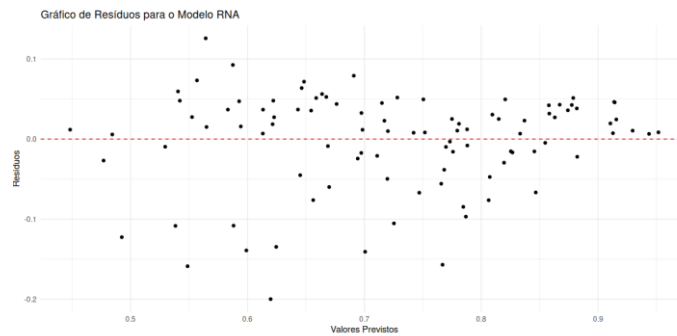
Predição de novos casos:

FIGURA 46 - Predição de novos casos

	GRE.Score	TOEFL.Score	University.Rating	SOP	LOR	CGPA	Research	predict.rna
1	300	105	5	4.5	3.0	7.9	1	0.69
2	331	99	3	4.0	2.0	8.4	0	0.61
3	305	107	4	3.0	3.5	9.0	1	0.79

Gráfico de resíduos para algoritmo escolhido:

GRÁFICO 13 - Gráfico de resíduos do modelo RNA



Lista de comandos do algoritmo escolhido:

FIGURA 47 - Lista de comandos do algoritmo escolhido

```
### Pacotes necessários:
#install.packages("caret")
#install.packages("e1071")
#install.packages("mlbench")
#install.packages("mlc")
#install.packages("Metrics")
library(Metrics)
library(mlbench)
library(caret)
library(mlc)

## Leitura dos dados
dados <- read.csv("9 - Admissao - Dados.csv")

### Retira o atributo num
dados$num <- NULL
View(dados)

### Cria arquivos de treino e teste
set.seed(202475)
ind <- createDataPartition(dados$ChanceOfAdmit, p=0.80, list = FALSE)
treino <- dados[ind,]
teste <- dados[-ind,]

### CV e parametrização da RNA
control <- trainControl(method = "cv", number = 10)
tuneGrid <- expand.grid(size = seq(from = 1, to = 10, by = 1), decay = seq(from = 0.1, to = 0.9, by = 0.3))

set.seed(202475)
rna <- train(ChanceOfAdmit~., data=treino, method="nnet", trainControl=control, tuneGrid=tuneGrid, llnout=T,
            MaxNWts=10000, maxit=2000, trace=F)

rna

### Predições e métricas
predicoes.rna <- predict(rna, teste)
rmse_valor <- rmse(teste$ChanceOfAdmit, predicoes.rna)
r2_valor <- r2(predicoes.rna, teste$ChanceOfAdmit)

# Cálculo do Erro Padrão da Estimativa (Syx)
syx <- function(observado, predito) {
  sqrt(sum((observado - predito)^2) / (length(observado) - 2))
}

syx_valor <- syx(teste$ChanceOfAdmit, predicoes.rna)

# Cálculo do Coeficiente de Correlação de Pearson
pearson_valor <- cor(teste$ChanceOfAdmit, predicoes.rna)

# Cálculo do MAE
mae <- function(observado, predito) {
  mean(abs(predito - observado))
}

mae_valor <- mae(teste$ChanceOfAdmit, predicoes.rna)

# Resultados
resultados <- list(
  R2 = format(r2_valor, digits = 8),
  Syx = format(syx_valor, digits = 8),
  Pearson = format(pearson_valor, digits = 8),
  RMSE = format(rmse_valor, digits = 8),
  MAE = format(mae_valor, digits = 8)
)

print(resultados)
```

FIGURA 48 - Lista de comandos do algoritmo escolhido (continuação)

```

# Calculo dos residuos
residuos <- teste$ChanceOfAdmit - predicoes.rna

# Cria um data frame para o ggplot
df_residuos <- data.frame(predicoes = predicoes.rna, residuos = residuos)

# Crie o gráfico de residuos
ggplot(df_residuos, aes(x = predicoes, y = residuos)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Gráfico de Resíduos para o Modelo RNA",
        x = "Valores Previstos",
        y = "Resíduos") +
  theme_minimal()

### PREDIÇÕES DE NOVOS CASOS
dados_novos_casos <- read.csv("9 - Admissao - Novos Dados.csv")
View(dados_novos_casos)

predict.rna <- predict(rna, dados_novos_casos)
dados_novos_casos$ChanceOfAdmit <- NULL
resultado <- cbind(dados_novos_casos, predict.rna)
resultado

```

Aplicação de algoritmos de predição usando regressão em problemas de biomassa:

FIGURA 49 - Aplicação de algoritmos de regressão em Biomassa

Biomassa

Técnica	Parâmetro	R2	Syx	Pearson	Rmse	MAE
RF – CV	mtry=2	0.91359863	396.03396	0.96719006	389.37745	102.82337
RF – Hold-out	mtry=2	0.89921349	427.73355	0.96240691	420.54423	106.57287
RNA – CV	size=8, decay=0.7	0.797106	606.88543	0.90592035	596.68495	154.11875
KNN	K=9	0.78619982	622.98291	0.91016038	612.51186	180.985
SVM – CV	C=0.25, Sigma=4.342076	0.43398421	1013.6462	0.72185385	996.60887	250.34411
SVM – Hold-out	C=0.25, Sigma=4.342076	0.43398421	1013.6462	0.72185385	996.60887	250.34411

RNA – Hold-out	size=5, decay=0.1	0.014193276	1337.7285	0.84993975	1315.2441	448.10302
----------------	-------------------	-------------	-----------	------------	-----------	-----------

Predição de novos casos:

FIGURA 50 - Predição de novos casos

```

dap  h  Me predict.rf
1 7.9 4.0 0.80      16
2 6.5 5.5 0.83      14
3 6.8 5.7 1.06      15

```

Gráfico de resíduos para algoritmo escolhido:

GRÁFICO 14 - Resíduos do modelo Random Forest



Lista de comandos para algoritmo escolhido:

FIGURA 51 - Lista de comandos do algoritmo escolhido

```
### Pacotes necessários:
#install.packages("caret")
#install.packages("e1071")
#install.packages("kernlab")
#install.packages("Metrics")
library(Metrics)
library("caret")
library(ggplot2)
|
## Leitura dos dados
dados <- read.csv("5 - Biomassa - Dados.csv")

### Visualizar dados
View(dados)

### Cria arquivos de treino e teste
set.seed(202475)
ind <- createDataPartition(dados$biomassa, p=0.80, list = FALSE)
treino <- dados[ind,]
teste <- dados[-ind,]

### Treinar Random Forest com a base de Treino
set.seed(202475)
rf <- train(biomassa~., data=treino, method="rf")
rf

### Aplicar modelos treinados na base de Teste
ctrl <- trainControl(method = "cv", number = 10)

set.seed(202475)
rf <- train(biomassa~., data=treino, method="rf", trControl=ctrl)
rf
predicoes.rf <- predict(rf, teste)
```

FIGURA 52 - Lista de comandos do algoritmo escolhido (continuação)

```

### Calcular as métricas
rmse_valor <- rmse(teste$biomassa, predicoes.rf)
r2 <- function(predito, observado) {
  return(1 - (sum((predito-observado)^2) / sum((observado-mean(observado))^2)))
}
r2_valor <- r2(predicoes.rf, teste$biomassa)

# Cálculo do Erro Padrão da Estimativa (Syx)
syx <- function(observado, predito) {
  sqrt(sum((observado - predito)^2) / (length(observado) - 2))
}

syx_valor <- syx(teste$biomassa, predicoes.rf)

# Cálculo do Coeficiente de Correlação de Pearson
pearson_valor <- cor(teste$biomassa, predicoes.rf)

# Cálculo do MAE
mae <- function(observado, predito) {
  mean(abs(predito - observado))
}

mae_valor <- mae(teste$biomassa, predicoes.rf)

# Resultados
resultados <- list(
  R2 = format(r2_valor, digits = 8),
  Syx = format(syx_valor, digits = 8),
  Pearson = format(pearson_valor, digits = 8),
  RMSE = format(rmse_valor, digits = 8),
  MAE = format(mae_valor, digits = 8)
)

resultados

# Cálculo dos resíduos
residuos <- teste$biomassa - predicoes.rf

# Cria um data frame para o ggplot
df_residuos <- data.frame(predicoes = predicoes.rf, residuos = residuos)

# Crie o gráfico de resíduos
ggplot(df_residuos, aes(x = predicoes, y = residuos)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Gráfico de Resíduos para o Modelo RF",
       x = "Valores Previstos",
       y = "Resíduos") +
  theme_minimal()

### PREDIÇÕES DE NOVOS CASOS
dados_novos_casos <- read.csv("5 - Biomassa - Novos Dados.csv")
View(dados_novos_casos)

predict.rf <- predict(rf, dados_novos_casos)
dados_novos_casos$biomassa <- NULL
resultado <- cbind(dados_novos_casos, predict.rf)
resultado

```


3. AGRUPAMENTO

Primeiras 10 linhas do agrupamento de veículos criado:

FIGURA 53 - Visão dos dados de agrupamento

Cluster modes:

	Comp	Circ	DCirc	RadRa	PrAxisRa	MaxLRa	ScatRa	Elong	PrAxisRect	MaxLRect	ScVarMaxis	ScVarmaxis	RaGyr	SkewMaxis	Skewmaxis	Kurtmaxis	KurtMaxis	HollRa	tipo
1	86	37	66	126	56	7	128	52	18	127	159	246	132	85	1	6	201	183	van
2	89	45	85	150	64	10	155	45	19	144	169	354	187	72	5	9	188	196	van
3	93	38	74	176	59	7	169	39	20	131	184	259	137	67	1	3	192	202	opel
4	101	55	101	193	62	10	212	31	24	159	219	682	214	74	5	11	187	197	opel
5	89	36	66	136	58	7	144	46	19	128	164	314	127	66	0	14	181	185	saab
6	104	54	103	209	57	11	213	31	24	162	223	706	218	72	0	19	188	198	opel
7	89	40	83	197	63	9	177	37	21	139	197	290	151	71	4	1	189	198	opel
8	82	43	70	141	64	7	151	44	19	143	175	341	172	75	4	0	183	187	bus
9	85	43	70	120	54	7	150	45	19	145	170	327	171	85	6	12	180	184	bus
10	89	46	85	162	64	11	157	43	20	148	170	363	186	68	1	11	189	199	saab

Lista de comandos do algoritmo:

FIGURA 54 - Lista de comandos do algoritmo escolhido

```
### Instalação dos pacotes necessários
install.packages("klaR")
library(klaR)

## Leitura dos dados
dados <- read.csv("6 - Veiculos - Dados.csv")

### Retira o atributo a
dados$a <- NULL
View(dados)

### Resultados
set.seed(202475)
cluster.results <- kmodes(dados, 10, iter.max = 10, weighted = FALSE )
cluster.results
```

4. REGRAS DE ASSOCIAÇÃO

Regras geradas com configuração de suporte e confiança, para os dados de musculação:

FIGURA 55 - Suporte e confiança dos dados de Musculação

summary of quality measures:

support	confidence	coverage	lift	count
Min. :0.03846	Min. :0.7000	Min. :0.03846	Min. :0.8667	Min. : 1.000
1st Qu.:0.07692	1st Qu.:0.8571	1st Qu.:0.07692	1st Qu.:1.5294	1st Qu.: 2.000
Median :0.07692	Median :1.0000	Median :0.07692	Median :1.7143	Median : 2.000
Mean :0.14665	Mean :0.9328	Mean :0.16799	Mean :1.7468	Mean : 3.813
3rd Qu.:0.23077	3rd Qu.:1.0000	3rd Qu.:0.26923	3rd Qu.:2.0000	3rd Qu.: 6.000
Max. :0.46154	Max. :1.0000	Max. :0.65385	Max. :3.2500	Max. :12.000

Lista de comandos:

FIGURA 56 - Lista de comandos do algoritmo selecionado

```
### Instalação dos pacotes necessários
install.packages('arules', dep=T)
library(arules)
library(datasets)

## Leitura dos dados
dados <- read.transactions(file="2 - Musculacao - Dados.csv",format="basket",sep=";")
inspect(dados[1:4])

### Podemos ver a frequência dos 10 primeiros itens:
itemFrequencyPlot(dados, topN=10, type='absolute')

### Suporte=0,001 e Confiança=0,7
set.seed(202475)
rules <- apriori(dados, parameter = list(supp = 0.001, conf = 0.7, minlen=2))
summary(rules)
```

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

Usando a base de dados CIFAR10 e o algoritmo CNN para solução. Iniciou-se pelo download dos dados e separação em treino e teste:

FIGURA 57 - Download da base CIFAR10

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 ————— 5s 0us/step
```

FIGURA 58 - Separação em treino e teste

```
x_train.shape: (50000, 32, 32, 3)
y_train.shape: (50000,)
x_test.shape: (10000, 32, 32, 3)
y_test.shape: (10000,)
```

Configurou-se o modelo CNN com as camadas:

FIGURA 59 - Camadas da CNN

Model: "functional"

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 32, 32, 3)	0
conv2d (Conv2D)	(None, 15, 15, 32)	896
conv2d_1 (Conv2D)	(None, 7, 7, 64)	18,496
conv2d_2 (Conv2D)	(None, 3, 3, 128)	73,856
flatten (Flatten)	(None, 1152)	0
dropout (Dropout)	(None, 1152)	0
dense (Dense)	(None, 1024)	1,180,672
dropout_1 (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 10)	10,250

Total params: 1,284,170 (4.90 MB)

Trainable params: 1,284,170 (4.90 MB)

Non-trainable params: 0 (0.00 B)

Treinamento com 15 épocas:

FIGURA 60 - Execução de treinamento em 15 épocas

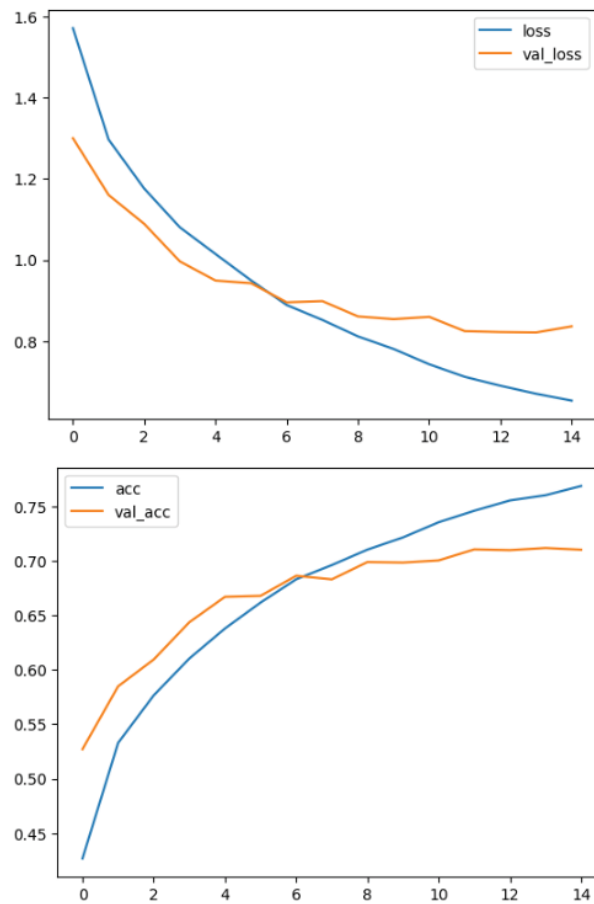
```

Epoch 1/15
1563/1563 ————— 114s 71ms/step - accuracy: 0.3589 - loss: 1.7515 - val_accuracy: 0.5271 - val_loss: 1.3005
Epoch 2/15
1563/1563 ————— 152s 77ms/step - accuracy: 0.5229 - loss: 1.3222 - val_accuracy: 0.5848 - val_loss: 1.1605
Epoch 3/15
1563/1563 ————— 111s 71ms/step - accuracy: 0.5718 - loss: 1.1901 - val_accuracy: 0.6094 - val_loss: 1.0893
Epoch 4/15
1563/1563 ————— 150s 76ms/step - accuracy: 0.6092 - loss: 1.0783 - val_accuracy: 0.6436 - val_loss: 0.9967
Epoch 5/15
1563/1563 ————— 140s 89ms/step - accuracy: 0.6396 - loss: 1.0163 - val_accuracy: 0.6669 - val_loss: 0.9495
Epoch 6/15
1563/1563 ————— 119s 75ms/step - accuracy: 0.6605 - loss: 0.9496 - val_accuracy: 0.6678 - val_loss: 0.9429
Epoch 7/15
1563/1563 ————— 141s 74ms/step - accuracy: 0.6809 - loss: 0.8918 - val_accuracy: 0.6862 - val_loss: 0.8958
Epoch 8/15
1563/1563 ————— 107s 68ms/step - accuracy: 0.7024 - loss: 0.8338 - val_accuracy: 0.6829 - val_loss: 0.8989
Epoch 9/15
1563/1563 ————— 154s 76ms/step - accuracy: 0.7143 - loss: 0.8003 - val_accuracy: 0.6988 - val_loss: 0.8611
Epoch 10/15
1563/1563 ————— 136s 73ms/step - accuracy: 0.7257 - loss: 0.7660 - val_accuracy: 0.6983 - val_loss: 0.8543
Epoch 11/15
1563/1563 ————— 142s 73ms/step - accuracy: 0.7389 - loss: 0.7336 - val_accuracy: 0.7003 - val_loss: 0.8600
Epoch 12/15
1563/1563 ————— 145s 75ms/step - accuracy: 0.7494 - loss: 0.7010 - val_accuracy: 0.7104 - val_loss: 0.8246
Epoch 13/15
1563/1563 ————— 117s 75ms/step - accuracy: 0.7595 - loss: 0.6769 - val_accuracy: 0.7097 - val_loss: 0.8225
Epoch 14/15
1563/1563 ————— 148s 79ms/step - accuracy: 0.7648 - loss: 0.6528 - val_accuracy: 0.7117 - val_loss: 0.8214
Epoch 15/15
1563/1563 ————— 139s 77ms/step - accuracy: 0.7722 - loss: 0.6402 - val_accuracy: 0.7101 - val_loss: 0.8366

```

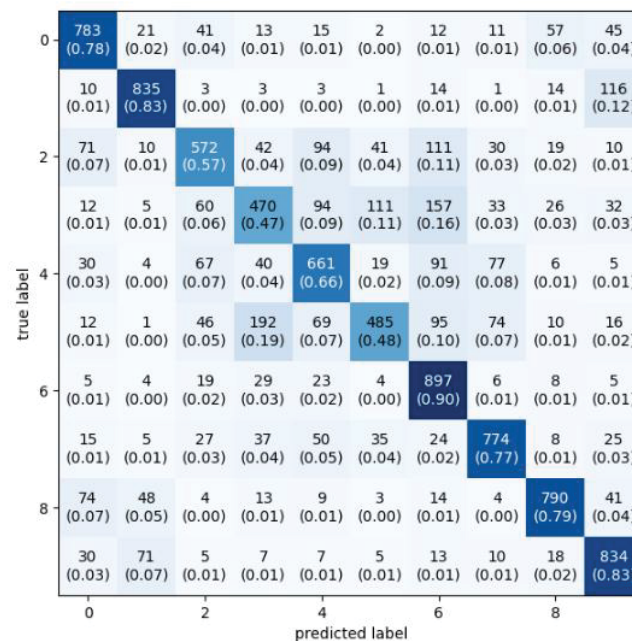
Perda e acurácia durante treinamento:

GRÁFICO 15 - Perda e acurácia do treinamento CNN



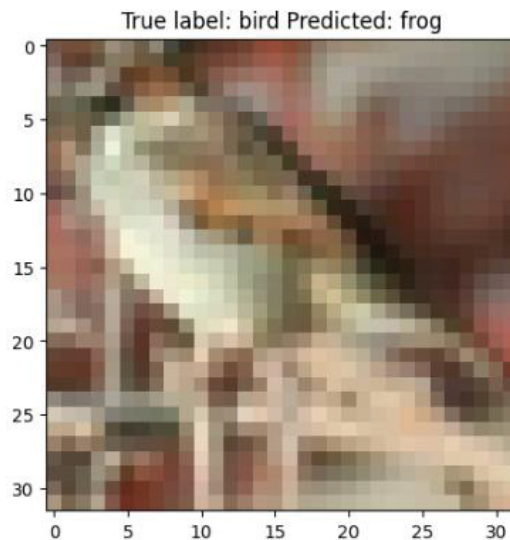
Matriz de confusão, pós treinamento:

GRÁFICO 16 - Matriz de confusão pós treinamento



Exemplo de errada:

FIGURA 61 - Exemplo de predição errada



2.DETECTOR DE SPAM

Usando base de mensagens SMS, implementamos um algoritmo RNN para detecção de SPAM. Iniciamos com o download da base, separação em treino e teste e tokenização com limite por frequência:

FIGURA 62 - Download de base de SPAM

```
--2024-08-18 02:27:17-- https://filebin.net/o7trbqexn1eg7706/spam.csv
Resolving filebin.net (filebin.net)... 88.99.137.18, 2a01:4f8:10a:2156::2
Connecting to filebin.net (filebin.net)|88.99.137.18|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://s3.filebin.net/filebin/4c8b12ad0793c5b0859130617115ec2fab8b929j
--2024-08-18 02:27:18-- https://s3.filebin.net/filebin/4c8b12ad0793c5b0859130617
Resolving s3.filebin.net (s3.filebin.net)... 88.99.137.18, 2a01:4f8:10a:2156::2
Connecting to s3.filebin.net (s3.filebin.net)|88.99.137.18|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 503663 (492K) [text/csv]
Saving to: 'spam.csv.1'

spam.csv.1      100%[=====>] 491.86K  867KB/s   in 0.6s

2024-08-18 02:27:19 (867 KB/s) - 'spam.csv.1' saved [503663/503663]
```

FIGURA 63 - Separação de treinamento e teste SPAM

```
data_train.shape: (3733, 189)
data_test.shape:  (1839, 189)
```

Criamos o modelo RNN com algoritmo LSTM:

FIGURA 64 - Modelo RNN com LSTM

Model: "functional"

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 189)	0
embedding (Embedding)	(None, 189, 20)	142,560
lstm (LSTM)	(None, 5)	520
dense (Dense)	(None, 1)	6

Total params: 143,086 (558.93 KB)

Trainable params: 143,086 (558.93 KB)

Non-trainable params: 0 (0.00 B)

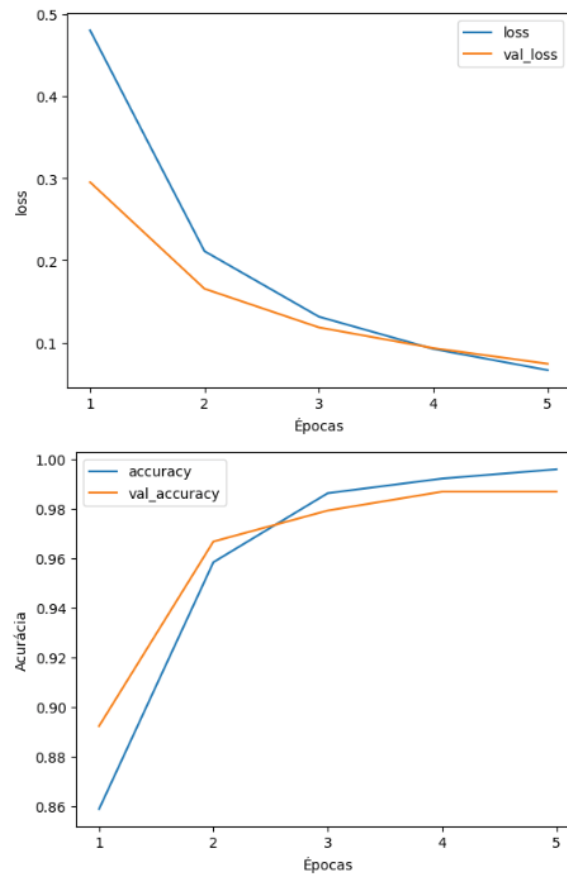
Treinamos com 5 épocas:

FIGURA 65 - Treinamento RNN com 5 épocas

```
Epoch 1/5
117/117 ————— 21s 146ms/step - accuracy: 0.8170 - loss: 0.5888 - val_accuracy: 0.8923 - val_loss: 0.2952
Epoch 2/5
117/117 ————— 20s 170ms/step - accuracy: 0.9404 - loss: 0.2392 - val_accuracy: 0.9668 - val_loss: 0.1656
Epoch 3/5
117/117 ————— 20s 167ms/step - accuracy: 0.9874 - loss: 0.1446 - val_accuracy: 0.9793 - val_loss: 0.1185
Epoch 4/5
117/117 ————— 17s 139ms/step - accuracy: 0.9929 - loss: 0.0995 - val_accuracy: 0.9869 - val_loss: 0.0933
Epoch 5/5
117/117 ————— 18s 154ms/step - accuracy: 0.9958 - loss: 0.0706 - val_accuracy: 0.9869 - val_loss: 0.0743
```

Gráfico de perda e acurácia durante treinamento:

GRÁFICO 17 - Gráfico de perda e acuraria durante treinamento RNN



Exemplo de detecção de SPAM com o modelo treinado, para a frase “*Is your car dirty? Discover our new product. Free for all. Click the link.*”

FIGURA 66 - Predição de SPAM com RNN

```
1/1 ————— 0s 51ms/step
[[0.65158933]]
SPAM
```

3. GERADOR DE DÍGITOS FAKE

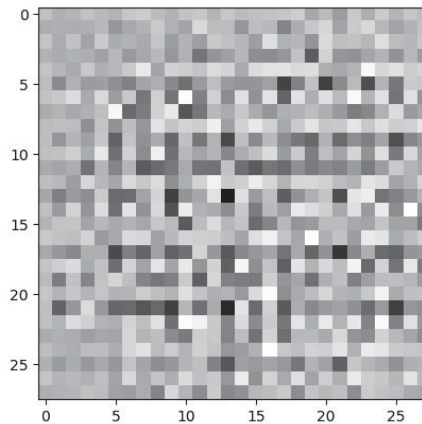
Com dados do MNIST, implementamos a arquitetura GAN para gerar dígitos. Usamos as libs imageio e tensor flow. Iniciamos com download da base:

FIGURA 67 - Download base MNIST

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ————— 0s 0us/step
```

Montamos o modelo de geração e fizemos um teste com ele sem estar treinado:

FIGURA 68 - Geração de imagem sem treino



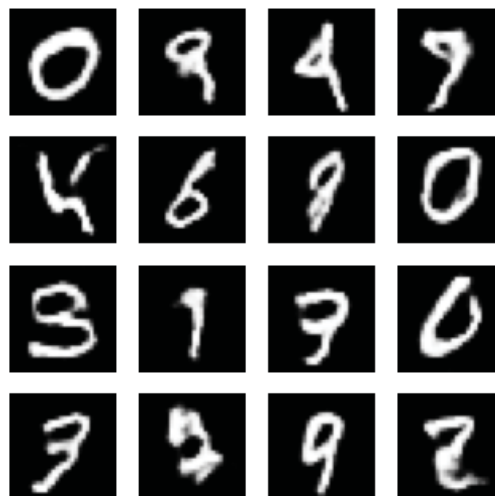
Da mesma forma com o discriminador:

FIGURA 69 - Dados discriminador do GAN

```
tf.Tensor([[0.00541993]], shape=(1, 1), dtype=float32)
```

Criamos funções para calcular perda do discriminador e gerador, uma função de treinamento que combina as duas técnicas, com otimizador adam e uma estrutura para salvar o treinamento (checkpoint) caso o treinamento seja interrompido. Realizamos o treinamento de 100 épocas, salvando cada imagem gerada:

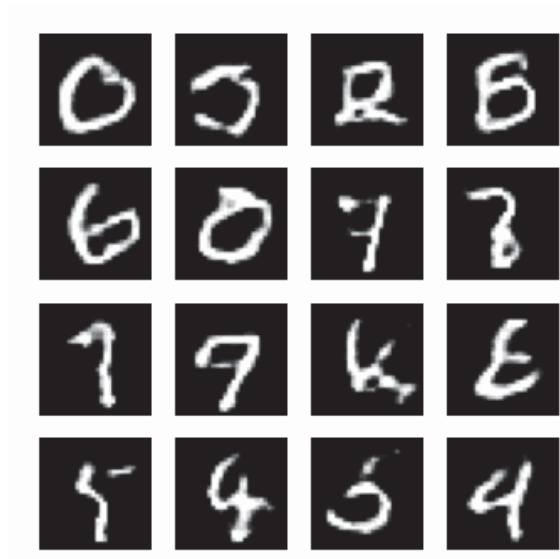
FIGURA 70 - Resultado do GAN para 100 épocas de treinamento



Time for epoch 60 is 12.108076572418213 sec

Após o treinamento, combinamos as imagens geradas formando o GIF abaixo:

FIGURA 71 - GIF combinado



4. TRADUTOR DE TEXTO

Para construir um tradutor de textos, usando a técnica de Transformer, usamos a biblioteca TensorFlow e o conjunto de dados derivados de transcrições de palestras do TED. Iniciamos com o download:

FIGURA 72 - Download transcrição palestras TED

```
WARNING:absl:Variant folder /root/tensorflow_datasets/ted_hr1r_translate/pt_to_en/1.0.0 has no dataset_info.json
Downloading and preparing dataset Unknown size (download: Unknown size, generated: Unknown size, total: Unknown size) to /root/tensorflow_datasets/ted_hr1r_translate/pt_to_en/1.0.0...
DI Completed...: 100% 1/1 [00:04<00:00, 2.12s/ url]
DI Size...: 100% 124/124 [00:04<00:00, 60.53 MiB/s]
Extraction completed...: 100% 112/112 [00:04<00:00, 4.81s/ file]
Dataset ted_hr1r_translate downloaded and prepared to /root/tensorflow_datasets/ted_hr1r_translate/pt_to_en/1.0.0. Subsequent calls will reuse this data.
```

Depois de carregar os dados, exibimos uma amostra para validação:

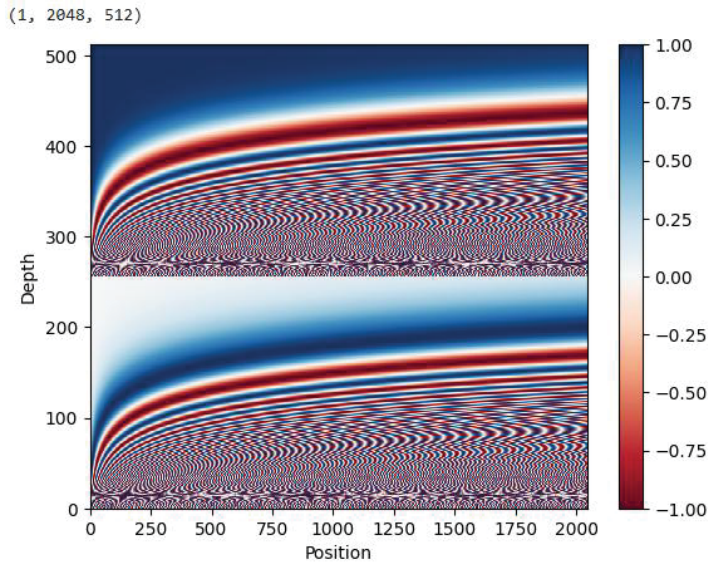
FIGURA 73 - Amostra da base TED

```
e quando melhoramos a procura , tiramos a única vantagem da impressão , que é a serendipidade .
mas e se estes fatores fossem ativos ?
mas eles não tinham a curiosidade de me testar .

and when you improve searchability , you actually take away the one advantage of print , which is serendipity .
but what if it were active ?
but they did n't test for curiosity .
```

Visualizamos como a codificação posicional varia com a posição e a dimensão do *embedding*.

FIGURA 74 - Variação de acordo com dimensão de *embedding*



Visualizamos o modelo depois de construído:

FIGURA 75 - Modelo de *Transformer*

Model: "transformer_1"

Layer (type)	Output Shape	Param #
encoder_1 (Encoder)	?	0 (unbuilt)
decoder_1 (Decoder)	?	0 (unbuilt)
dense_129 (Dense)	?	0 (unbuilt)

Total params: 0 (0.00 B)
Trainable params: 0 (0.00 B)
Non-trainable params: 0 (0.00 B)

Treinamento feito em 20 épocas salvando cada etapa (checkpoints):

FIGURA 76 - Treinamento do Transformer com 20 épocas

```

Epoch 19 Batch 0 Loss 1.3288 Accuracy 0.6962
Epoch 19 Batch 50 Loss 1.4567 Accuracy 0.6781
Epoch 19 Batch 100 Loss 1.4462 Accuracy 0.6799
Epoch 19 Batch 150 Loss 1.4516 Accuracy 0.6792
Epoch 19 Batch 200 Loss 1.4543 Accuracy 0.6783
Epoch 19 Batch 250 Loss 1.4580 Accuracy 0.6779
Epoch 19 Batch 300 Loss 1.4573 Accuracy 0.6779
Epoch 19 Batch 350 Loss 1.4582 Accuracy 0.6780
Epoch 19 Batch 400 Loss 1.4614 Accuracy 0.6776
Epoch 19 Batch 450 Loss 1.4669 Accuracy 0.6767
Epoch 19 Batch 500 Loss 1.4692 Accuracy 0.6765
Epoch 19 Batch 550 Loss 1.4708 Accuracy 0.6760
Epoch 19 Batch 600 Loss 1.4744 Accuracy 0.6755
Epoch 19 Batch 650 Loss 1.4775 Accuracy 0.6748
Epoch 19 Batch 700 Loss 1.4796 Accuracy 0.6747
Epoch 19 Batch 750 Loss 1.4817 Accuracy 0.6744
Epoch 19 Batch 800 Loss 1.4834 Accuracy 0.6743
Epoch 19 Loss 1.4836 Accuracy 0.6742
Time taken for 1 epoch: 98.27 secs

Epoch 20 Batch 0 Loss 1.2931 Accuracy 0.7111
Epoch 20 Batch 50 Loss 1.3878 Accuracy 0.6894
Epoch 20 Batch 100 Loss 1.4033 Accuracy 0.6874
Epoch 20 Batch 150 Loss 1.4096 Accuracy 0.6861
Epoch 20 Batch 200 Loss 1.4101 Accuracy 0.6866
Epoch 20 Batch 250 Loss 1.4153 Accuracy 0.6855
Epoch 20 Batch 300 Loss 1.4151 Accuracy 0.6856
Epoch 20 Batch 350 Loss 1.4191 Accuracy 0.6846
Epoch 20 Batch 400 Loss 1.4209 Accuracy 0.6842
Epoch 20 Batch 450 Loss 1.4253 Accuracy 0.6834
Epoch 20 Batch 500 Loss 1.4289 Accuracy 0.6828
Epoch 20 Batch 550 Loss 1.4319 Accuracy 0.6823
Epoch 20 Batch 600 Loss 1.4346 Accuracy 0.6816
Epoch 20 Batch 650 Loss 1.4362 Accuracy 0.6813
Epoch 20 Batch 700 Loss 1.4400 Accuracy 0.6808
Epoch 20 Batch 750 Loss 1.4431 Accuracy 0.6803
Epoch 20 Batch 800 Loss 1.4468 Accuracy 0.6798
Saving checkpoint for epoch 20 at ./checkpoints/train/ckpt-10
Epoch 20 Loss 1.4476 Accuracy 0.6798
Time taken for 1 epoch: 97.16 secs

```

Teste da predição, com o modelo treinado, usando a frase: “Eu li sobre triceratops na enciclopédia.”

FIGURA 77 - Tradução com o Transformer treinado

```
Prediction      : b'i read about thornal thornass in encycload .'
```

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

1. PROJETO BIG DATA

Estudo de caso da Aplicação de Big Data em E-commerce com Ferramentas NoSQL e NewSQL

1.2 INTRODUÇÃO

No cenário competitivo do comércio eletrônico, empresas líderes como Amazon, Netflix e Uber lidam diariamente com volumes massivos de dados provenientes de transações, interações de usuários, avaliações de produtos e atividades em redes sociais. Gerenciar e extrair valor desses dados é essencial para melhorar a experiência do cliente, otimizar operações e impulsionar as vendas. Este estudo de caso explora como essas empresas utilizam ferramentas de Big Data, especificamente bancos de dados NoSQL e NewSQL, para atender às suas demandas de processamento e análise de dados.

1.3 CARACTERIZAÇÃO DOS DADOS E DOS Vs

Essas empresas enfrentam desafios relacionados aos cinco Vs do Big Data:

Volume:

- Amazon: Processa petabytes de dados diariamente, incluindo histórico de compras, navegação no site e interações em tempo real;

- Netflix: Gerencia enormes volumes de dados de streaming e preferências de milhões de usuários globalmente;
- Uber: Lida com dados geoespaciais e transações em tempo real de milhões de corridas diárias;

Velocidade:

- Amazon: Necessita de atualizações em milissegundos para inventário e processamento de pedidos;
- Uber: Calcula rotas e tarifas em tempo real, exigindo processamento instantâneo de dados;
- Netflix: Oferece recomendações imediatas baseadas no comportamento atual do usuário;

Variedade:

- Amazon: Dados estruturados (transações), semiestruturados (logs de cliques) e não estruturados (comentários, avaliações);
- Netflix: Dados de streaming, logs de reprodução, interações do usuário e métricas de qualidade de serviço;
- Uber: Dados geoespaciais, transações financeiras, feedback de usuários e dados de tráfego;

Veracidade:

- Amazon: Investe em sistemas para garantir a autenticidade de avaliações e evitar fraudes;
- Uber: Necessita de dados precisos para segurança e confiabilidade do serviço;
- Netflix: Garante a qualidade dos dados para oferecer recomendações precisas;

Valor:

- Amazon: Utiliza análise de dados para personalizar ofertas e melhorar a satisfação do cliente;
- Uber: Otimiza rotas e tempos de espera, melhorando a eficiência operacional;

- Netflix: A análise de dados direciona a produção de conteúdo original e retenção de usuários;

1.4 FERRAMENTAS UTILIZADAS

Para lidar com esses desafios, as empresas implementaram uma combinação de bancos de dados NoSQL e NewSQL.

1.5 BANCO DE DADOS NOSQL

Amazon DynamoDB (Amazon)

- Uso: A Amazon utiliza o DynamoDB para gerenciar dados de produtos, carrinhos de compras e sessões de usuários;
- Benefícios: Oferece latência de milissegundos em qualquer escala, permitindo que a Amazon mantenha alta performance durante picos de acesso, como na Black Friday;
- Modelagem: Adota um modelo de chave-valor e documento, proporcionando flexibilidade na estrutura dos dados;

Apache Cassandra (Netflix)

- Uso: A Netflix emprega o Cassandra para armazenar dados de streaming, preferências do usuário e histórico de visualização;
- Benefícios: Fornece alta disponibilidade e escalabilidade horizontal, essencial para o serviço global da Netflix;
- Modelagem: Utiliza um modelo de colunas largas, ideal para grandes volumes de dados e acessos rápidos;

1.6 BANCO DE DADOS NEWSQL

Google Spanner (Google)

- Uso: Embora não seja uma empresa de e-commerce tradicional, o Google utiliza o Spanner para serviços que requerem consistência global, como transações financeiras no Google Ads;
- Benefícios: Combina a escalabilidade de sistemas NoSQL com transações ACID, garantindo consistência em escala global;
- Modelagem: Mantém um esquema relacional tradicional, facilitando consultas complexas com SQL padrão;

MemSQL (SingleStore) (Uber)

- Uso: A Uber utiliza o MemSQL para processar transações financeiras e dados de telemetria em tempo real;
- Benefícios: Oferece processamento rápido de transações e análises, suportando decisões em tempo real;
- Modelagem: Combina armazenamento em memória e disco, permitindo consultas rápidas sem sacrificar a persistência dos dados;

Modelagem Necessária

A escolha do modelo de dados adequado é fundamental para otimizar o desempenho e a eficiência dos sistemas.

Modelagem em NoSQL

- Modelo de Documento (DynamoDB na Amazon):
 - Estrutura: Armazena dados em formatos flexíveis como JSON.
 - Aplicação: Permite à Amazon adicionar novos atributos aos produtos sem migrações complexas.
 - Benefícios: Escalabilidade e flexibilidade para acomodar diferentes tipos de produtos e categorias.
- Modelo de Colunas Largas (Cassandra na Netflix):

- Estrutura: Organiza dados em tabelas com linhas e colunas, mas com a capacidade de armazenar milhões de colunas.
- Aplicação: Ideal para o histórico de visualização de usuários e logs de atividades.
- Benefícios: Otimiza operações de gravação e leitura em grandes volumes de dados distribuídos.

Modelagem em NewSQL

- Modelo Relacional Distribuído (Spanner no Google):
 - Estrutura: Mantém tabelas relacionais com suporte a transações ACID.
 - Aplicação: Gerencia dados que requerem consistência forte e transações distribuídas.
 - Benefícios: Combina a familiaridade do SQL com a escalabilidade necessária para operações globais.
- Modelo Híbrido (MemSQL na Uber):
 - Estrutura: Une aspectos de bancos de dados relacionais e NoSQL, suportando SQL e armazenamento em memória;
 - Aplicação: Processa dados de corridas, pagamentos e localização em tempo real;
 - Benefícios: Permite análises rápidas e decisões imediatas sem comprometer a integridade dos dados.

1.7 CONCLUSÃO

A adoção de bancos de dados NoSQL e NewSQL permitiu que empresas como Amazon, Netflix e Uber superassem os desafios associados ao Big Data, proporcionando:

- Melhor Desempenho: Processamento eficiente de grandes volumes de dados em tempo real, melhorando a responsividade dos serviços e a satisfação do cliente.

- Flexibilidade: Capacidade de adaptar-se rapidamente a novas fontes de dados e requisitos de negócios sem reestruturações significativas do banco de dados.
- Confiabilidade: Garantia de integridade e consistência dos dados críticos.

Os desafios enfrentados incluíram a complexidade na integração de diferentes sistemas de banco de dados e a necessidade de profissionais qualificados em tecnologias diversas. Essas empresas superaram esses obstáculos através de arquiteturas bem planejadas, investimentos em infraestrutura e foco no desenvolvimento de equipes especializadas.

Este estudo de caso destaca a importância de selecionar as ferramentas e modelos de dados apropriados para atender às necessidades específicas do negócio. Empresas líderes como Amazon, Netflix e Uber demonstram que o uso estratégico de tecnologias NoSQL e NewSQL é crucial para maximizar o valor extraído dos dados e manter a competitividade no mercado global de e-commerce.

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:

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

2) Redes Neurais

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

Tarefas:

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

B – RESOLUÇÃO

1. EXTRAÇÃO DE CARACTERÍSTICAS

Iniciamos com importação das bibliotecas necessárias

FIGURA 78 - Importação bibliotecas

```
import os
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, classification_report
from keras.applications.vgg16 import VGG16
from keras.preprocessing import image
from keras.applications.vgg16 import preprocess_input
from skimage.feature import local_binary_pattern
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import f1_score, recall_score, precision_score
from sklearn import metrics
import zipfile
from keras.preprocessing.image import img_to_array, load_img
from sklearn.model_selection import train_test_split
import cv2
from google.colab.patches import cv2_imshow
```

Preparamos a base de dados

FIGURA 79 - Base de dados Warwick e separação em treino e teste

```

train_zip_path = '/content/Train_Warwick.zip'
test_zip_path = '/content/Test_Warwick.zip'

train_extract_dir = '/content/Train_Warwick'
test_extract_dir = '/content/Test_Warwick'

def unzip_file(zip_path, extract_dir):
    with zipfile.ZipFile(zip_path, 'r') as zip_ref:
        zip_ref.extractall(extract_dir)

unzip_file(train_zip_path, train_extract_dir)
unzip_file(test_zip_path, test_extract_dir)

train_dir_contents = os.listdir(train_extract_dir)
test_dir_contents = os.listdir(test_extract_dir)

train_image_dir = os.path.join(train_extract_dir, 'Train_4cls_amostra')
test_image_dir = os.path.join(test_extract_dir, 'Test_4cl_amostra')

train_files = os.listdir(train_image_dir)
test_files = os.listdir(test_image_dir)

train_dir_contents, test_dir_contents

# Verificar a estrutura dentro dos diretórios de treino e teste
train_image_dir = os.path.join(train_extract_dir, 'Train_4cls_amostra')
test_image_dir = os.path.join(test_extract_dir, 'Test_4cl_amostra')

train_files = os.listdir(train_image_dir)
test_files = os.listdir(test_image_dir)

# Exibir os primeiros arquivos encontrados
train_files, test_files

```

Primeiros arquivos encontrados:

```
(['1', '0', '3', '2'], ['1', '2_teste.txt', '0', '3', '2'])
```

Carregamos as imagens de treinamento e teste com OpenCV

FIGURA 80 - Carregando imagens com OpenCV

```

# Função para carregar as imagens usando OpenCV
def load_images_opencv(base_dir):
    images = []
    labels = []
    for label_dir in os.listdir(base_dir):
        label_path = os.path.join(base_dir, label_dir)
        if os.path.isdir(label_path): # Verificar se é um diretório de classe
            for img_file in os.listdir(label_path):
                if img_file.endswith('.png'):
                    img_path = os.path.join(label_path, img_file)
                    img = cv2.imread(img_path)
                    img = cv2.resize(img, (250, 250))
                    images.append(img)
                    labels.append(label_dir) # O nome do diretório é o rótulo
    return np.array(images), np.array(labels)

# Carregar as imagens e rótulos do conjunto de treino e teste
train_images, train_labels = load_images_opencv(train_image_dir)
test_images, test_labels = load_images_opencv(test_image_dir)

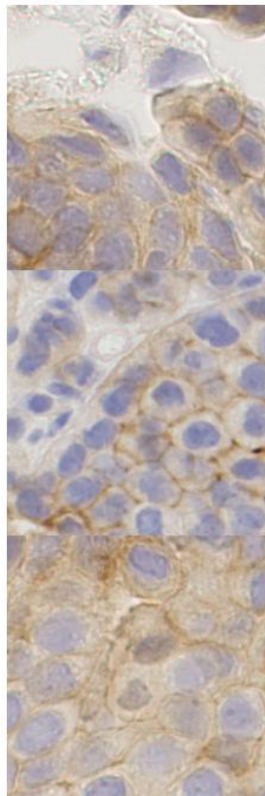
# Exibir o tamanho das bases carregadas
train_images.shape, len(train_labels), test_images.shape, len(test_labels)

#Exibir amostra
cv2.imshow(train_images[1])
cv2.imshow(train_images[2])
cv2.imshow(train_images[3])

```

Exibimos as primeiras imagens

FIGURA 81 – Amostra da base Warrick



Separamos as bases de teste e treinamento por paciente

FIGURA 82 - Separação em teste e treinamento

```
# Garantir que a separação seja feita por paciente
unique_patients = np.unique(train_labels)

# Separar 80% para treino e 20% para validação, garantindo a separação por paciente
train_patients, val_patients = train_test_split(unique_patients, test_size=0.2, random_state=42)

# Criar índices de treino e validação com base nos pacientes
train_idx = [i for i, label in enumerate(train_labels) if label in train_patients]
val_idx = [i for i, label in enumerate(train_labels) if label in val_patients]

# Criar os conjuntos de treino e validação
X_train, y_train = train_images[train_idx], train_labels[train_idx]
X_val, y_val = train_images[val_idx], train_labels[val_idx]

# Exibir o tamanho dos conjuntos de treino e validação
X_train.shape, len(y_train), X_val.shape, len(y_val)
```

Características do conjunto de treinamento e teste:

FIGURA 83 - Características da base separada

((446, 250, 250, 3), 446, (147, 250, 250, 3), 147)

Extração de características

FIGURA 84 - Extração de características das imagens

```

def extract_lbp_features(images):
    lbp_features = []
    for img in images:
        gray_img = cv2.cvtColor(img.astype('uint8'), cv2.COLOR_BGR2GRAY)
        lbp = local_binary_pattern(gray_img, 24, 3, method="uniform")
        hist, _ = np.histogram(lbp.ravel(), bins=np.arange(0, 27), range=(0, 26))
        hist = hist.astype('float')
        hist /= (hist.sum() + 1e-6) # Normalização
        lbp_features.append(hist)
    return np.array(lbp_features)

def extract_vgg16_features(images):
    vgg_model = VGG16(weights='imagenet', include_top=False)
    features = vgg_model.predict(images)
    features = features.reshape(features.shape[0], -1) # Flatten
    return features

# Extração para treino
lbp_train_features = extract_lbp_features(X_train)
vgg_train_features = extract_vgg16_features(X_train)

# Extração para validação
lbp_val_features = extract_lbp_features(X_val)
vgg_val_features = extract_vgg16_features(X_val)

# Salvar como CSV
pd.DataFrame(lbp_train_features).to_csv('lbp_train_features.csv')
pd.DataFrame(vgg_train_features).to_csv('vgg_train_features.csv')

```

Treinamento com os modelos Random Forest, SVM e RNA

FIGURA 85 - Treinamento com modelos RF, SVM e RNA

```
def avg_specificity(confusion_matrix):
    num_classes = confusion_matrix.shape[0]
    specificities = []

    for i in range(num_classes):
        # Verdadeiros Negativos (VN) são todos os valores fora da linha e coluna da classe i
        true_negatives = np.sum(confusion_matrix) - (np.sum(confusion_matrix[i, :]) + np.sum(confusion_matrix[:, i]) - confusion_matrix[i, i])
        # Falsos Positivos (FP) são a soma dos valores na coluna da classe i, exceto a diagonal (falsos positivos)
        false_positives = np.sum(confusion_matrix[:, i]) - confusion_matrix[i, i]

        # Evitar divisão por zero
        if (true_negatives + false_positives) > 0:
            specificity = true_negatives / (true_negatives + false_positives)
        else:
            specificity = 0 # Definir como 0 quando o denominador for 0
        specificities.append(specificity)

    # Especificidade média
    specificity_mean = np.mean(specificities)
    return specificity_mean

def train_and_evaluate(model, X_train, y_train, X_val, y_val):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_val)
    cm = confusion_matrix(y_val, y_pred)
    sensitivity = recall_score(y_val, y_pred, average='macro')
    specificity = avg_specificity(cm)
    f1 = f1_score(y_val, y_pred, average='macro')
    return sensitivity, specificity, f1

# Modelos
rf_model = RandomForestClassifier()
svm_model = SVC()
rna_model = MLPClassifier()

train_and_evaluate(rf_model, vgg_train_features, y_train, vgg_val_features, y_val)
train_and_evaluate(svm_model, vgg_train_features, y_train, vgg_val_features, y_val)
train_and_evaluate(rna_model, vgg_train_features, y_train, vgg_val_features, y_val)
```

Carrega a base de teste

FIGURA 86 - Aplicação das funções de extração

```
lbp_test_features = extract_lbp_features(test_images)
vgg_test_features = extract_vgg16_features(test_images)
```

Aplicação dos modelos treinados nos dados de teste e cálculo das métricas de Sensibilidade, Especificidade e F1-Score

FIGURA 87 - Avaliação dos modelos

```
rf_test_sens, rf_test_spec, rf_test_f1 = train_and_evaluate(rf_model, vgg_train_features, y_train, vgg_test_features, test_labels)
svm_test_sens, svm_test_spec, svm_test_f1 = train_and_evaluate(svm_model, vgg_train_features, y_train, vgg_test_features, test_labels)
rna_test_sens, rna_test_spec, rna_test_f1 = train_and_evaluate(rna_model, vgg_train_features, y_train, vgg_test_features, test_labels)

print(f"Random Forest - Teste Sensibilidade: {rf_test_sens}, Especificidade: {rf_test_spec}, F1-Score: {rf_test_f1}")
print(f"SVM - Teste Sensibilidade: {svm_test_sens}, Especificidade: {svm_test_spec}, F1-Score: {svm_test_f1}")
print(f"RNA - Teste Sensibilidade: {rna_test_sens}, Especificidade: {rna_test_spec}, F1-Score: {rna_test_f1}")
```

E o resultado

FIGURA 88 - Resultados dos modelos

Random Forest - Teste Sensibilidade: 0.7194444444444444, Especificidade: 0.9084025306445235, F1-Score: 0.6268723377755636
 SVM - Teste Sensibilidade: 0.6944444444444444, Especificidade: 0.9003229207855543, F1-Score: 0.6037909171630103
 RNA - Teste Sensibilidade: 0.7123212321232123, Especificidade: 0.9073283247660472, F1-Score: 0.6319611799889666

Exibe melhor modelo de acordo com F1

FIGURA 89 - Seleção do melhor modelo

```
models = {'Random Forest': rf_test_f1, 'SVM': svm_test_f1, 'RNA': rna_test_f1}
best_model = max(models, key=models.get)
print(f"O melhor modelo foi {best_model} com F1-Score de {models[best_model]}")
```

O melhor modelo foi RNA com F1-Score de 0.6319611799889666

Conclusão da extração de características

O Random Forest demonstrou ser o modelo mais eficaz, sobretudo por ter apresentado os melhores índices de sensibilidade e especificidade entre os três avaliados. Isso reflete sua capacidade superior tanto em identificar corretamente os casos positivos quanto em minimizar os falsos positivos. Apesar de a RNA ter alcançado o melhor F1-Score, o Random Forest se destacou por proporcionar um desempenho mais uniforme e consistente nas três métricas consideradas.

2. REDES NEURAIIS

Iniciamos com a importação as bibliotecas

FIGURA 90 - Importação de bibliotecas para redes neurais

```
from keras.applications.vgg16 import VGG16
from keras.applications.resnet import ResNet50
from keras.layers import Dense, Flatten, GlobalAveragePooling2D
from keras.models import Model
from keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator

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

Utilizamos a mesma base do exercício anterior, alterando algumas características

FIGURA 91 - Redefinição das imagens

```
def resize_images(images):
    resized_images = []
    for img in images:
        resized_img = cv2.resize(img, (224, 224)) # Redimensionar para 224x224
        if img.shape[-1] == 1: # Se a imagem tiver um único canal (em tons de cinza), converte para 3 canais
            resized_img = cv2.cvtColor(resized_img, cv2.COLOR_GRAY2RGB)
        resized_images.append(resized_img)
    return np.array(resized_images)

# Carregar as imagens e rótulos do conjunto de treino e validação
X_train_resized = resize_images(X_train) # De 250x250 para 224x224
X_val_resized = resize_images(X_val)
X_test_resized = resize_images(test_images)

# Verificar as formas das imagens redimensionadas
print("Forma das imagens de treino:", X_train_resized.shape)
print("Forma das imagens de validação:", X_val_resized.shape)
print("Forma das imagens de teste:", X_test_resized.shape)
```

FIGURA 92 - Forma das imagens de treino, teste e validação

```
Forma das imagens de treino: (446, 224, 224, 3)
Forma das imagens de validação: (147, 224, 224, 3)
Forma das imagens de teste: (371, 224, 224, 3)
```

Treina modelos VGG16 e Resnet50 adaptadas com e sem Data Augmentation

Redes Neurais

FIGURA 93- Código do treinamento da rede neural VGG16 com Resnet50

```

# Verificar os rótulos e garantir que há 4 classes
print(np.unique(y_train))

# One-hot encode dos rótulos com 4 classes (0, 1, 2, 3)
from sklearn.preprocessing import LabelBinarizer
lb = LabelBinarizer()
lb.fit([0, 1, 2, 3]) # Garantindo que 4 classes são configuradas corretamente
y_train_bin = lb.transform(y_train)
y_val_bin = lb.transform(y_val)
y_test_bin = lb.transform(test_labels)

# Verificar a forma dos rótulos binarizados
print("Forma dos rótulos de treino:", y_train_bin.shape)

# Função para criar o modelo baseado na VGG16 com camadas Fully Connected ajustadas
def create_vgg16_model():
    base_model = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
    x = base_model.output
    x = Flatten()(x) # Usando a camada Flatten para achatar as saídas
    x = Dense(128, activation='relu')(x)
    predictions = Dense(4, activation='softmax')(x) # Ajustar para 4 classes
    model = Model(inputs=base_model.input, outputs=predictions)

    # Congelar as camadas da base da VGG16
    for layer in base_model.layers:
        layer.trainable = False

    model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
    return model

# Função para criar o modelo baseado na ResNet50 com camadas Fully Connected ajustadas
def create_resnet50_model():
    base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
    x = base_model.output
    x = Flatten()(x)
    x = Dense(128, activation='relu')(x)
    predictions = Dense(4, activation='softmax')(x) # Ajustar para 4 classes
    model = Model(inputs=base_model.input, outputs=predictions)

    # Congelar as camadas da base da ResNet50
    for layer in base_model.layers:
        layer.trainable = False

    model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
    return model

```

```

# Função para treinar o modelo com ou sem Data Augmentation
def train_model(model, X_train, y_train, X_val, y_val, augment=False):
    if augment:
        datagen = ImageDataGenerator(
            rotation_range=20,
            width_shift_range=0.2,
            height_shift_range=0.2,
            horizontal_flip=True
        )
        datagen.fit(X_train)
        history = model.fit(datagen.flow(X_train, y_train, batch_size=32),
                            validation_data=(X_val, y_val),
                            epochs=10)
    else:
        history = model.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=10, batch_size=32)
    return history

# Criar os modelos
vgg16_model = create_vgg16_model()
resnet50_model = create_resnet50_model()

# Treinar o modelo VGG16 sem Data Augmentation
print("Treinando VGG16 sem Data Augmentation...")
train_model(vgg16_model, X_train_resized, y_train_bin, X_val_resized, y_val_bin, augment=False)

# Treinar o modelo VGG16 com Data Augmentation
print("Treinando VGG16 com Data Augmentation...")
train_model(vgg16_model, X_train_resized, y_train_bin, X_val_resized, y_val_bin, augment=True)

# Treinar o modelo ResNet50 sem Data Augmentation
print("Treinando ResNet50 sem Data Augmentation...")
train_model(resnet50_model, X_train_resized, y_train_bin, X_val_resized, y_val_bin, augment=False)

# Treinar o modelo ResNet50 com Data Augmentation
print("Treinando ResNet50 com Data Augmentation...")
train_model(resnet50_model, X_train_resized, y_train_bin, X_val_resized, y_val_bin, augment=True)

```

E o resultado

FIGURA 94 - Treinamento em 10 épocas

```
[ '0' '2' '3' ]
Forma dos rótulos de treino: (446, 4)
Treinando VGG16 sem Data Augmentation...
Epoch 1/10
14/14 ----- 6s 353ms/step - accuracy: 0.1593 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 2/10
14/14 ----- 3s 248ms/step - accuracy: 0.0112 - loss: 0.0000e+00 - val_accuracy: 0.8299 - val_loss: 0.0000e+00
Epoch 3/10
14/14 ----- 3s 203ms/step - accuracy: 0.1989 - loss: 0.0000e+00 - val_accuracy: 0.1293 - val_loss: 0.0000e+00
Epoch 4/10
14/14 ----- 3s 207ms/step - accuracy: 0.1055 - loss: 0.0000e+00 - val_accuracy: 0.8571 - val_loss: 0.0000e+00
Epoch 5/10
14/14 ----- 3s 252ms/step - accuracy: 0.3156 - loss: 0.0000e+00 - val_accuracy: 0.0476 - val_loss: 0.0000e+00
Epoch 6/10
14/14 ----- 3s 214ms/step - accuracy: 0.1735 - loss: 0.0000e+00 - val_accuracy: 0.0680 - val_loss: 0.0000e+00
Epoch 7/10
14/14 ----- 3s 214ms/step - accuracy: 0.1852 - loss: 0.0000e+00 - val_accuracy: 0.7755 - val_loss: 0.0000e+00
Epoch 8/10
14/14 ----- 3s 214ms/step - accuracy: 0.2753 - loss: 0.0000e+00 - val_accuracy: 0.0408 - val_loss: 0.0000e+00
Epoch 9/10
14/14 ----- 5s 206ms/step - accuracy: 0.2426 - loss: 0.0000e+00 - val_accuracy: 0.4762 - val_loss: 0.0000e+00
Epoch 10/10
14/14 ----- 3s 205ms/step - accuracy: 0.2767 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Treinando VGG16 com Data Augmentation...
Epoch 1/10
14/14 ----- 13s 466ms/step - accuracy: 0.2208 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 2/10
14/14 ----- 15s 287ms/step - accuracy: 0.2046 - loss: 0.0000e+00 - val_accuracy: 0.0204 - val_loss: 0.0000e+00
Epoch 3/10
14/14 ----- 10s 491ms/step - accuracy: 0.2924 - loss: 0.0000e+00 - val_accuracy: 0.5578 - val_loss: 0.0000e+00
Epoch 4/10
14/14 ----- 7s 292ms/step - accuracy: 0.3000 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 5/10
14/14 ----- 11s 293ms/step - accuracy: 0.1945 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00
Epoch 6/10
14/14 ----- 10s 546ms/step - accuracy: 0.4125 - loss: 0.0000e+00 - val_accuracy: 0.8163 - val_loss: 0.0000e+00
Epoch 7/10
14/14 ----- 10s 360ms/step - accuracy: 0.2654 - loss: 0.0000e+00 - val_accuracy: 0.9320 - val_loss: 0.0000e+00
Epoch 8/10
14/14 ----- 8s 351ms/step - accuracy: 0.2380 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 9/10
14/14 ----- 12s 323ms/step - accuracy: 0.2230 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 10/10
14/14 ----- 12s 323ms/step - accuracy: 0.2230 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Treinando ResNet50 sem Data Augmentation...
Epoch 1/10
14/14 ----- 20s 802ms/step - accuracy: 0.1393 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 2/10
14/14 ----- 9s 139ms/step - accuracy: 0.0198 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 3/10
14/14 ----- 2s 120ms/step - accuracy: 0.0098 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 4/10
14/14 ----- 3s 121ms/step - accuracy: 0.0164 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 5/10
14/14 ----- 2s 125ms/step - accuracy: 0.0068 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 6/10
14/14 ----- 3s 140ms/step - accuracy: 0.0113 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 7/10
14/14 ----- 2s 125ms/step - accuracy: 0.0215 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 8/10
14/14 ----- 3s 127ms/step - accuracy: 0.0292 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 9/10
14/14 ----- 2s 143ms/step - accuracy: 0.0164 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 10/10
14/14 ----- 2s 144ms/step - accuracy: 0.0219 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Treinando ResNet50 com Data Augmentation...
Epoch 1/10
14/14 ----- 19s 511ms/step - accuracy: 0.0693 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 2/10
14/14 ----- 7s 307ms/step - accuracy: 0.0841 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 3/10
14/14 ----- 5s 229ms/step - accuracy: 0.0940 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 4/10
14/14 ----- 15s 346ms/step - accuracy: 0.0984 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 5/10
14/14 ----- 10s 253ms/step - accuracy: 0.1469 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 6/10
14/14 ----- 9s 384ms/step - accuracy: 0.1098 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 7/10
14/14 ----- 8s 253ms/step - accuracy: 0.1078 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 8/10
14/14 ----- 10s 254ms/step - accuracy: 0.2165 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 9/10
14/14 ----- 11s 237ms/step - accuracy: 0.1030 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
Epoch 10/10
14/14 ----- 6s 251ms/step - accuracy: 0.1683 - loss: 0.0000e+00 - val_accuracy: 0.0000e+00 - val_loss: 0.0000e+00
<keras.src.callbacks.historyv.History at 0x7abcb25c250>
```

Avaliando de acordo com as métricas

FIGURA 95 - Avaliação dos modelos com matriz de confusão

```
def evaluate_model(model, X_test, y_test):
    y_pred = np.argmax(model.predict(X_test), axis=1)
    y_true = np.argmax(y_test, axis=1)

    # Identificar as classes presentes
    labels_present = np.unique(y_true)

    # Matriz de confusão
    cm = confusion_matrix(y_true, y_pred, labels=labels_present)
    print("Matriz de Confusão:")
    print(cm)

    # Relatório de classificação apenas para as classes presentes
    report = classification_report(y_true, y_pred, labels=labels_present, target_names=[str(l) for l in labels_present], zero_division=1)
    print(report)

    # Cálculo de Sensibilidade, Especificidade e F1-Score
    sensitivity = recall_score(y_true, y_pred, average='macro')
    specificity = np.mean(np.diag(cm) / np.sum(cm, axis=1))
    f1 = f1_score(y_true, y_pred, average='macro')

    return sensitivity, specificity, f1

# Avaliar os modelos na base de teste
print("Avaliando VGG16 sem Data Augmentation...")
vgg16_sens, vgg16_spec, vgg16_f1 = evaluate_model(vgg16_model, X_test_resized, y_test_bin)

print("Avaliando ResNet50 sem Data Augmentation...")
resnet50_sens, resnet50_spec, resnet50_f1 = evaluate_model(resnet50_model, X_test_resized, y_test_bin)

# Comparar os resultados
print(f"VGG16 - Sensibilidade: {vgg16_sens}, Especificidade: {vgg16_spec}, F1-Score: {vgg16_f1}")
print(f"ResNet50 - Sensibilidade: {resnet50_sens}, Especificidade: {resnet50_spec}, F1-Score: {resnet50_f1}")

# Comparar os modelos e exibir o melhor resultado
models = {'VGG16': vgg16_f1, 'ResNet50': resnet50_f1}
best_model = max(models, key=models.get)
print(f"O melhor modelo foi {best_model} com F1-Score de {models[best_model]}")
```

FIGURA 96 - Resultado da avaliação

```
Avaliando VGG16 sem Data Augmentation...
12/12 ————— 2s 183ms/step
Matriz de Confusão:
[[369]]
```

	precision	recall	f1-score	support
0	1.00	0.99	1.00	371
micro avg	1.00	0.99	1.00	371
macro avg	1.00	0.99	1.00	371
weighted avg	1.00	0.99	1.00	371

```

Avaliando ResNet50 sem Data Augmentation...
12/12 ————— 10s 348ms/step
Matriz de Confusão:
[[0]]
```

	precision	recall	f1-score	support
0	1.00	0.00	0.00	371.0
micro avg	1.00	0.00	0.00	371.0
macro avg	1.00	0.00	0.00	371.0
weighted avg	1.00	0.00	0.00	371.0

```

VGG16 - Sensibilidade: 0.4973045822102426, Especificidade: 1.0, F1-Score: 0.49864864864864866
ResNet50 - Sensibilidade: 0.0, Especificidade: nan, F1-Score: 0.0
O melhor modelo foi VGG16 com F1-Score de 0.49864864864864866
```

O modelo VGG16, treinado sem a utilização de técnicas de *Data Augmentation*, mostrou-se extremamente eficiente conforme demonstrado pela matriz de confusão. Esta matriz evidenciou um total de 369 previsões corretas de um universo de 371 possíveis.

A elevada especificidade do modelo indica uma eficácia notável em evitar falsos positivos. Por outro lado, a sensibilidade de 0.4973 sugere que o modelo foi capaz de reconhecer uma porção considerável dos exemplos verdadeiramente positivos.

Os resultados obtidos com o ResNet50 apontam para uma dificuldade do modelo em aprender a classificação. Diante disso, algumas estratégias poderiam ser adotadas para aprimorar o desempenho do modelo, tais como:

- Descongelar um número maior de camadas para permitir uma aprendizagem mais flexível;
- Incrementar o número de épocas para que o modelo tenha mais oportunidades de aprender com os dados;
- Realizar o balanceamento do conjunto de dados, a fim de evitar um viés do modelo em favor das classes mais frequentes;
- Garantir a qualidade dos dados, verificando se estão adequadamente formatados e rotulados, para assegurar a eficácia do treinamento do modelo.

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

1. ESTUDO DE CASO: IMPLICAÇÕES ÉTICAS DO USO DO CHATGPT

2. RESUMO

A ascensão do ChatGPT, um serviço de chat dotado de inteligência artificial da OpenAI, foi meteórica, atingindo 1 milhão de usuários em apenas 5 dias e 100 milhões em dois meses. Esse crescimento exponencial sugere a necessidade de um olhar cuidadoso sobre as questões éticas e de segurança, em especial no que tange à discriminação e desigualdades sociais. O Senado Federal brasileiro debate o PL 2.338/2023, um projeto que busca instituir o Marco Legal da Inteligência Artificial, mas o progresso tem sido lento. É imperioso que haja uma supervisão humana qualificada e uma conscientização sobre o uso da IA. Um ponto crucial é a neutralidade e representatividade, já que a IA é predominantemente treinada com dados em inglês, refletindo uma visão ocidental e perpetuando estereótipos. O presente trabalho também visa trazer uma visão sobre a transparência e a responsabilidade das empresas que usam IA, já que são elos fundamentais para assegurar a ética da sua aplicação.

Palavras-chave: Chat GPT, Inteligência Artificial, Ética, Tecnologia, Sociedade.

3. ABSTRACT

The rise of ChatGPT, a chat service powered by OpenAI's artificial intelligence, has been meteoric, reaching 1 million users in just 5 days and 100 million in two months. This exponential growth suggests the need for careful consideration of ethical and security issues, especially regarding discrimination and social inequalities. The

Brazilian Federal Senate is debating PL 2.338/2023, a bill that seeks to establish the Legal Framework for Artificial Intelligence, but progress has been slow. It is imperative that there be qualified human oversight and awareness of AI usage. A crucial point is neutrality and representativeness, as AI is predominantly trained with data in English, reflecting a Western perspective and perpetuating stereotypes. This work also aims to provide insight into the transparency and accountability of companies using AI, as they are fundamental links in ensuring the ethics of its application.

Keywords: Chat GPT, Artificial Intelligence, Ethics, Technology, Society.

4. INTRODUÇÃO

O serviço de chat, com respostas geradas por uma inteligência artificial, da Open AI, cresceu sua base de usuários de forma assustadoramente rápida. O Chat GPT (acrônimo de *Chat Generative Pre-trained Transformer*) foi liberado ao público em 2022 - sendo sua primeira versão privada criada em 2018 - e atingiu a marca de 1 milhão de usuários em apenas 5 dias (BUCHHOLZ, 2023). Em dois meses atingiu a marca de 100 milhões de usuários (HU, 2023). Tornando-se o serviço digital com a adesão mais rápida da história. Tamanha evolução causa riscos e preocupações éticas que não tínhamos até então nesta particular área do desenvolvimento de software. Urge a discussão dos efeitos causados na segurança, trabalho humano e atenção à discriminação e desigualdades sociais para que este desenvolvimento preserve os direitos fundamentais. E que seja assegurado desenvolvimento tecnológico seguro em benefício da pessoa humana.

No Brasil, tramita no Senado Federal a PL 2.338/2023 (BRASIL, 2023), texto inicial do Senador e presidente da casa Rodrigo Pacheco (PSD-MG), com a participação de uma comissão. Tal projeto de lei é considerado a primeira base para o Marco Legal da Inteligência Artificial no Brasil. Mas avança lentamente e as discussões parecem não ter eco nas empresas como fora o Marco Civil da Internet. Embora tenham realizado quatro audiências públicas, um seminário internacional, doze painéis temáticos e consulta a mais de 60 especialistas (BRASIL, 2023).

5. REVISÃO DE LITERATURA

5.1 RISCO DO USO EXCESSIVO

A motivação do uso massivo do Chat GPT, um LLM (*Large Language Model*) ou modelo de aprendizagem de máquina com bilhões de parâmetros, é bem clara. Conseguimos respostas rápidas para qualquer assunto, somos capazes de usá-lo para criar textos com estilos diversos, criar código de programação, traduzir documentos em idiomas diferentes, aprender sobre novos temas, obter insights de como resolver problemas, dos mais complexos aos mais frugais, como por resolver uma equação integral esférica para encontrar o volume de uma esfera imperfeita ou como trocar uma lâmpada de tungstênio.

Mas não se engane, como toda ferramenta, existem imperfeições, neste caso o termo usualmente utilizado para descrever estas imperfeições é alucinações. É quando o modelo de linguagem não consegue inferir uma resposta correta e acaba produzindo um texto falso, seja por vieses, poucos dados de treinamento ou *overfitting*, dos algoritmos. Emsley (2023) defende que não são alucinações e sim invenções e falsificações. Ele descreve a imperfeição como:

“A causa destas falsificações tem sido associada a uma perturbação na produção linguística, com resultados probabilísticos baseados em estimativas de similaridade semântica” - (EMSLEY,2023)

Emsley (2023) cita que em uma investigação de acurácia e autenticidade de artigos médicos, o ChatGPT citou 115 referências. Destas 47% foram fabricadas, 46% são autênticas porem imprecisas e apenas 7% foi identificada como autêntica e precisa.

Em que nível está esta alucinação? Só recentemente a OpenAI adicionou o alerta que passa despercebido pela maioria das pessoas: “ChatGPT pode cometer erros. Considere verificar informações importantes.”. Mas não há monitoramento de sua qualidade. Fica evidente a necessidade de supervisão humana qualificada. E mais, conscientização de seu uso, dado que a confiabilidade na assertividade está ainda longe de ser uma constante.

Como toda ferramenta nova, seu uso intenso pode causar uma série de efeitos colaterais não conhecidos ou não relacionados ainda. Por exemplo, no início da revolução da comunicação causada pelos dispositivos móveis cada vez mais cheio de recursos, aliados à expansão da própria internet, os teclados destes dispositivos causaram surtos de problemas nos tendões, inflamação que recebeu o apelido de “BlackBerry Thumb”. Embora tenha sido descoberta há anos com a alcunha de síndrome de Quervain (ORTHOPEDIC AND SPORTS SPECIALISTS, 2024), somente com a explosão massiva dos tecladinhos diminutos é que a síndrome foi experimentada massivamente.

A ansiedade humana da busca pelo conhecimento também viu na interconexão da internet, crises de ansiedade e doenças novas como a nomofobia (no *mobile phobia*), que é quando o indivíduo não consegue ficar longe do celular (YOUGOV, 2023). O uso massivo do assistente sugere que o caminho de novas doenças ou deficiências também seja trilhado. Por isso carece de estudo sério e de discussões a seu respeito, para que seja legislado sobre.

Vale ressaltar que a partir da versão 4-o do Chat GPT você pode conversar com a IA através do aplicativo móvel. O uso excessivo do assistente virtual sugere riscos nas relações humanas, uma vez que impreterivelmente podem ser substituídas em certo grau - o assistente lhe dá as respostas que você quer ouvir, reforçando suas crenças. Não estamos distantes do filme Her (HER, 2013), onde um escritor solitário chamado Theodore, se apaixona por um sistema operacional, cuja voz fora personificada como Samantha.

A modalidade do ensino remoto também já tem seus efeitos. Há estudos que indicam que os tempos de concentração são inferiores aos presenciais, tornando necessário intervenções regulares para garantir coesão (STROM et al., 2023). Agora mude para o cenário do Chat GPT. Entenda que neste estudo, você tenha à disposição algo que não só te indique o caminho como lhe responda de forma relativamente objetiva o que for perguntado. Ou que seja capaz de fazer a pesquisa por você. Mais do que um tutor, a IA se comporta como um parceiro de trabalho. Se o usuário não for consciente dessas implicações e das limitações da ferramenta, as consequências podem ser ainda mais desastrosas no campo do aprendizado.

5.2 NEUTRALIDADE E REPRESENTATIVIDADE

Para dar respostas mais precisas, o modelo de aprendizagem de máquina não supervisionado é treinado com um grande volume de dados. De acordo com (BROWN et al, 2020), o Chat GPT versão 3 (175 bilhões de parâmetros) usou para ser treinado 8 anos de indexação de sites, publicações do serviço, publicações do serviço de forum Reddit com mais votos, bibliotecas de livros digitalizados - inclusive acadêmicos, e a versão em inglês da Wikipedia. Sendo que destes 93% dos dados estão no idioma inglês.

Na versão 4 do ChatGPT (MAZZONI et al., 2023), além de usar dados licenciados de provedores terceiros, o algoritmo consegue consultar dados públicos na internet. E usa um mecanismo chamado de *Reinforcement Learning from Human Feedback* (RLHF) para que a resposta seja direcionada para a pergunta do usuário (diretrizes ou limitações impostas ou *user guardrails*).

Com esta formulação é difícil pensar em representatividade de toda a diversidade cultural existente no mundo ou isonomia de pensamento, já que majoritariamente usa só o idioma de uma nação. Com uma base notavelmente focada na interpretação do mundo pela ótica dos Estados Unidos. É o viés consumado. Quase a garantia de amplificação de preconceitos existentes nesta sociedade. Uma ferramenta de perpetuação de injustiças e discriminações.

Não precisamos ir longe para traçar paralelos com o que já vivenciamos. Veja o caso do cinema nacional, que teve sua criatividade e pujança duramente ofuscadas pela massividade de produções cinematográficas norte-americanas. Não só como efeito de dominação cultural, mas também legislativa, dado que impreterivelmente privilegiava o estrangeiro com financiamentos vultosos. Tal situação foi denunciada, discutida, escrutinada por Paulo Emílio Gomes, no livro “Uma situação Colonial?” (GOMES, 2016). Uma das frases do autor marca a situação:

“O cinema é incapaz de encontrar dentro de si próprio, energias que lhe permitam escapar à condenação do subdesenvolvimento” (GOMES, 2016)

Isto é, mesmo que o cinema nacional faça uma obra *sui generis*, o gosto do público já foi tomado pela outra cultura. E não há espaço ou valia no reconhecimento da própria identidade.

Por outro ângulo, a questão da neutralidade, devemos ter a ciência de que a ferramenta não está disponível em todos os países do mundo. Seja pela legislação do país, seja por motivos puramente estratégicos, bélicos ou financeiros. Pelo site oficial

é possível obter uma lista de países suportados (OPENAI, 2024). Nota-se a ausência de China, Rússia, Síria, Cuba, Coreia do Norte dentre outros. O ponto de análise é: se a IA aprende através de trocas de mensagens com os usuários e através de pesquisas em sites da internet, artificialmente (ou naturalmente, fosse um humano) aprenderá pouco da cultura Oriental - com a devida visão da origem. Suas inferências probabilísticas serão, portanto, apenas espelhos da realidade. Eventualmente reforçadores de estereótipos.

Não é difícil pensar também em diferentes manifestações de racismo. A geração de imagens, recurso do DALL-E, incorporado aos modelos do Chat GPT tem inúmeros casos de falta de representatividade. (NUNES, 2022) faz uma importante revelação ao testar a busca por termos como “homem foto” e “mulher foto” e contar as ocorrências. Os algoritmos de IA das buscas foram treinados para encontrar referências de homem e mulher brancos europeus. É o racismo estrutural sedimentado.

Nunes (2022) também cita (BUOLAMWINI, 2016), pesquisadora que criou a *Algorithmic Justice League* (ALGORITHMIC JUSTICE LEAGUE, 2024), para que algoritmos evitem vieses. Ela demonstrou em sua pesquisa no MIT Media Lab que os algoritmos foram treinados para identificar rostos de brancos. Ela percebeu que precisava usar uma máscara branca para ser identificada como humana por alguns softwares de identificação facial.

Extrapolando a ideia, se solicitar ao ChatGPT para gerar uma imagem de um casal, sem passar qualquer instrução que sugira diversidade, na maioria dos casos a imagem gerada será um casal branco. Não que ele tenha sido programado para isto, mas a questão do treinamento e dos dados utilizados acabam carregando os chamados vieses algorítmicos. Imagine que as empresas desenvolvam aplicações de classificação de pessoas para fornecer um empréstimo, por exemplo. O bem estar estaria ameaçado.

5.3 TRANSPARÊNCIA

É muito claro que vivenciamos uma corrida para melhores modelos de IA generativa. Empresas como Microsoft, Open AI, Claude AI, Meta e outras, investem pesado a caminho de uma IA multimodal mais completa com capacidade cognitiva semelhante a humana, conhecida como IA Geral ou AGI. Mas as questões de

transparência levantaram pressões internas nestas empresas. É tão crítico que funcionários e ex-funcionários da própria OpenAI e de outras empresas fronteiriças de IA, criaram uma carta aberta chamada *Right to Warn AI*. Na carta aberta (RIGHT TO WARN, 2024) deixam claro que as empresas tem fortes incentivos financeiros para evitar a supervisão eficaz.

Alegam ainda que as empresas possuem relatórios de falhas, de imprecisões, seus controles, riscos implícitos de seus processos e que estes não são públicos. As empresas tem pouca obrigação de partilhar estas informações com governos e sugere nas entrelinhas que as empresas não dão espaço para uma cultura aberta que fomenta a discussão.

Para se ter uma ideia de precificação, o chat GPT gasta diariamente cerca de 700 mil dólares para funcionar (PATEL; AHMAD, 2023). Convertendo em reais, sem impostos, a soma passaria 1,384 bilhões de reais. Fora os custos de treinamento, licenciamento de conteúdo e demais custos operacionais.

Embora seja razoavelmente público que os primeiros modelos usaram anos de coleta de informações da internet, pouco se sabe a respeito dos métodos que foram usados na filtragem destes dados. Somente recentemente a Open AI lançou um programa de parceria de dados, o Open AI Data Partnerships (OPENAI, 2024). E através dele fez parcerias com o governo da Islândia e com uma instituição sem fins lucrativos chamada *FreeLaw* para usar sua base de casos jurídicos no treinamento. São bases tratadas e revisadas.

5.4 PRIVACIDADE DOS DADOS

Desde o Chat GPT versão 2, o serviço de chat utiliza um mecanismo de aprendizagem por reforço com recompensas a partir das perguntas dos humanos (ZIEGLER et al., 2019). Para isso o histórico de uso é armazenado e o perfil do usuário vai sendo aos poucos traçados.

Em 2023, o serviço de proteção a dados da Itália bloqueou o serviço de chatbot pelo não cumprimento das normas estabelecidas pela GDPR (*General Data Protection Regulation*), dado que não havia naquele tempo tratamento adequado a adolescentes abaixo de 13 anos. E mais grave, detectaram que até 1,2% de dados pessoais de um indivíduo poderiam vazar para outro usuário do sistema.

Controles foram estabelecidos para que o histórico de conversas seja desabilitado e dados pessoais possam não só ser exportados, com limpos. Além de um parco controle de acesso a menores de idade. Mas ainda é questionável a qualidade destas proteções, dado que os controles são ativados por padrão – e a maioria dos usuários nem se dá conta disso.

Existem estudos como o de (LIU et al., 2023), que demonstram de forma empírica prompts que conseguiram liberar todas as restrições do modelo de linguagem. A preocupação excessiva com vazamentos não parece ser infundada portanto.

6. METODOLOGIA

Com todas estas questões analisadas, criamos um questionário qualitativo para avaliar como o uso nas empresas tem se dado. O questionário foi disparado para grupos de entusiastas de IA e profissionais da computação que trabalham em áreas diversas.

As questões do questionário foram:

1. Como que frequência você usa IA em seu contexto de trabalho?
2. Você acredita que a sua organização tem políticas suficientes para evitar o uso excessivo de IA?
3. Na sua opinião, a IA utilizada pela sua organização é neutra e imparcial?
4. Sua empresa toma medidas para garantir que os dados utilizados para treinar modelos de IA sejam representativos da diversidade da população?
5. A sua organização informa claramente aos usuários sobre o uso de IA em seus serviços/produtos?
6. Os algoritmos e processos de decisão da IA são acessíveis e compreensíveis para os funcionários?
7. Você considera que a sua empresa está em conformidade com as regulamentações de privacidade de dados ao usar IA?
8. A sua organização possui uma estrutura clara de *accountability* para o uso de IA?

9. Quem é responsável por monitorar e garantir a ética na utilização de IA em sua empresa?
10. Quais procedimentos são seguidos quando ocorre um erro significativo em um sistema de IA?

7. APRESENTAÇÃO DOS RESULTADOS

Embora os resultados não tenham sido em grande volume – apenas 17 respostas - para que a amostra seja considerada, ela fornece preciosos insights de como as empresas estão em sua maioria despreparadas para o uso equalitário e responsável.

47,1% das respostas afirmam que utilizam muito frequentemente o IA em suas empresas. Mas 35,3% deles não acreditam que a empresa tenha política para evitar o uso excessivo. Outros 35,3% acreditam que a empresa tem parcialmente política para evitar o uso excessivo. Apenas 23,5% acreditam que o algoritmo de IA é imparcial ou neutro. Apenas 29,5% afirmam que as empresas tomam sempre medidas para que os modelos sejam representativos da diversidade populacional.

47,1% das empresas informam claramente seus usuários sobre o uso de IA em seus produtos. Mas 11,8% das empresas nunca informam.

Apenas 29,5% acreditam que a IA das empresas são acessíveis e compreensíveis para todos funcionários.

O único ponto positivo da pesquisa é que 47,1% das pessoas acreditam que suas empresas estão em conformidade de privacidade de dados. Mas 35,3% não sabem se a empresa tem alguma estrutura clara de prestação de contas para o uso de IA. Em geral a área de TI ou de Compliance absorve a responsabilidade pelo uso ético.

8. CONSIDERAÇÕES FINAIS

Ao adotarmos o Chat GPT para as atividades cotidianas, possibilitamos a automação de tarefas e a substituição de funções, o que pode reduzir a necessidade de força de trabalho humana. É crucial examinarmos as implicações dessas mudanças. Para mitigar os impactos negativos, devemos equilibrar e, se necessário, restringir o uso da IA assegurando a proteção dos direitos individuais, a promoção da

transparência e da responsabilidade, a prevenção de discriminação e viés, e a preservação da autonomia e dignidade humanas.

Recomendamos que as empresas e a sociedade civil criem ambientes que fomentem discussões de forma constante, para que o uso de IA respeite os seguintes pontos:

Neutralidade Algorítmica: Criar algoritmos que evitem preconceitos e discriminações, sendo desenvolvidos e implementados com esse propósito;

Transparência: Prover explicações claras sobre o processo de coleta, armazenamento e utilização de dados, assim como os critérios e mecanismos de decisão dos algoritmos;

Prestação de Contas (*Accountability*): Implementar mecanismos efetivos de responsabilização, com auditorias e avaliações éticas;

Proteção da Privacidade: Adotar políticas rigorosas para a proteção de dados pessoais, assegurando que a coleta e uso de dados sejam éticos e seguros, com respeito à privacidade dos usuários, e fomentar o debate público sobre essas práticas;

Equidade e Justiça: Garantir que as decisões automatizadas sejam justas e equitativas, evitando discriminações de qualquer natureza, como racial, de gênero ou por origem, através da implementação de critérios transparentes e auditáveis, com a devida informação ao usuário;

Inovação Responsável: Fomentar o desenvolvimento de IA que respeite os direitos e valores democráticos, promovendo uma inovação tecnológica responsável e sustentável;

Conformidade com Regulamentações: Garantir que o desenvolvimento e uso da IA estejam em conformidade com as regulamentações vigentes, incluindo leis de proteção de dados e direitos individuais, o marco civil da internet, a lei geral de proteção aos dados e as cláusulas pétreas da constituição;

Conscientização: Criar e distribuir materiais que esclareçam os riscos e ofereçam recomendações para o uso racional das ferramentas de IA.

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

1. RESUMO DO ARTIGO ESCOLHIDO

Artigo escolhido: 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. New York: ACM, 2021

1.1 OBJETIVO DO ESTUDO

O Auto ML é uma ferramenta para criação de fluxos de aprendizado de máquina (ML) de forma automática, através de dados disponíveis. Em suma, encurtaria a Sprint 2 (Modelagem e Avaliação), no processo de um projeto de ML. Podendo também atender pequenos projetos em que, supostamente, o responsável não tenha conhecimento de modelagem de ML. Tendo esta informação, o estudo visa, de forma qualitativa, avaliar ferramentas de AutoML disponíveis, seus benefícios, malefícios e como os usuários podem utilizar, colaborar e interagir com as ferramentas para obter bons resultados, ou seja, qual o papel do homem em relação ao uso do Auto-ML.

1.2 PROBLEMA, OPORTUNIDADE OU SITUAÇÃO QUE LEVOU AO ESTUDO

Com a popularização de ferramentas de Auto-ML o estudo visa entender a situação atual em relação a democratização de Aprendizagem de Máquina. Ou o grau

de maturidade destas ferramentas em diversos domínios e situações diferentes. É um panorama geral.

1.3 METODOLOGIA USADA PARA OBTER E ANALISAR INFORMAÇÕES DO ESTUDO

Metodologia qualitativa que entrevistou 16 usuários de ferramentas de Auto-ML de diferentes níveis de experiência, desde amadores até pesquisadores experientes.

Os pesquisadores organizaram os dados das entrevistas pelo software Deodoose em categorias, como vantagens e desvantagens do Auto-ML, estratégias de trabalho, necessidades dos usuários e fatores de adoção. Deixando claramente a interpretação

1.4 RESULTADOS OBTIDOS PELO ESTUDO

Os resultados obtidos podem ser divididos em benefícios, malefícios e o papel do humano. O primeiro:

- a) O Auto-ML facilita o acesso ao aprendizado de máquina (ML) para usuários sem grande experiência técnica, permitindo que profissionais de várias áreas integrem modelos aos seus fluxos de trabalho.
- b) Reduz o tempo necessário para desenvolver modelos, permitindo que os usuários explorem mais opções de modelagem em menos tempo.
- c) As ferramentas ajudam a popularizar o uso de ML, oferecendo padrões para workflows e aumentando a reprodutibilidade dos experimentos.

Em relação aos malefícios (ou limitações):

- a) As ferramentas atuais de Auto-ML oferecem automação principalmente na etapa de modelagem, deixando grande parte do pré e pós-processamento manual, o que pode atrasar ao invés de ajudar (alguém sem experiência).
- b) Muitas ferramentas funcionam como "caixas-pretas" e limitam o controle e a compreensão dos usuários sobre o que ocorre dentro do modelo, prejudicando a

confiança, especialmente em aplicações de alto risco. Ou seja, os modelos aplicados não podem muitas vezes serem explicados (se pensarmos modelo de processo proposto na aula, isto vai contra este princípio).

c) Usuários mais experientes sentem que as ferramentas de Auto-ML não permitem controle suficiente, enquanto os menos experientes podem se sentir sobrecarregados pela quantidade de opções.

A conclusão final aborda o papel do humano no processo:

O estudo sugere que, ao invés de excluir o desenvolvedor humano, as ferramentas de Auto-ML devem promover uma colaboração mútua, onde o usuário traz conhecimento contextual e toma decisões estratégicas, enquanto a automação acelera o processo. Essa abordagem "humano na interação" garante maior segurança, controle e eficácia, principalmente em situações onde o conhecimento de domínio é essencial.

APÊNDICE 13 – FRAMEWORKS DE INTELIGÊNCIA ARTIFICIAL

A – ENUNCIADO

1 Classificação (RNA)

Implementar o exemplo de Classificação usando a base de dados Fashion MNIST e a arquitetura RNA vista na aula **FRA - Aula 10 - 2.4 Resolução de exercício de RNA - Classificação**.

Além disso, fazer uma breve explicação dos seguintes resultados:

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

Informações:

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

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

2 Regressão (RNA)

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

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

Informações:

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

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


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

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

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

3 Sistemas de Recomendação

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

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

Informações:

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

4 Deepdream

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

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

Informações:

- **Base de dados:** https://commons.wikimedia.org/wiki/File:Felis_catus-cat_on_snow.jpg
- **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. RNA COM CLASSIFICAÇÃO

Para aplicação do RNA na base de dados “*Fashion*”, iniciamos com o carregamento das bibliotecas necessárias

FIGURA 97 - Importação das bibliotecas

```
#Bibliotecas

import tensorflow as tf
import matplotlib.pyplot as plt
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from mlxtend.plotting import plot_confusion_matrix
from sklearn.metrics import confusion_matrix
```

Carregamos a base de dados e dividimos em treino e teste

FIGURA 98 - Download base Fashion MNIST e separação de treinamento e teste

```
# Carregando base https://www.tensorflow.org/datasets/catalog/fashion\_mnist?hl=pt-br

data = tf.keras.datasets.fashion_mnist.load_data()

(x_train, y_train), (x_test, y_test) = data

x_data = np.concatenate([x_train, x_test])
y_data = np.concatenate([y_train, y_test])

# Split 75% treinamento e 25% teste
x_train, x_test, y_train, y_test = train_test_split(
    x_data, y_data, test_size=0.3, random_state=42
)
```

Averiguamos as características do conjunto

FIGURA 99 - Características do conjunto de dados

```
x_train shape: (49000, 28, 28)
y_train shape: (49000,)
x_test shape: (21000, 28, 28)
y_test shape: (21000,)
[[[0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  ...
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]]

[[[0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  ...
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]]

[[[0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  ...
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]]

...

[[[0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  ...
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]]

[[[0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  ...
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]
  [0 0 0 ... 0 0 0]]
...]
```

Normalizamos os pixels

FIGURA 100 - Normalização dos pixels

```
#Normalização dos pixels  
  
x_train = x_train / 255.0  
x_test = x_test / 255.0
```

Criamos o modelo de avaliação

FIGURA 101 - Criação do modelo sequencial

```
#Criação do modelo  
  
model = tf.keras.models.Sequential([  
    tf.keras.layers.Flatten(input_shape=(28, 28)),  
    tf.keras.layers.Dense(128, activation='relu'),  
    tf.keras.layers.Dense(10, activation='softmax')  
])
```

Escolhemos a otimização do modelo

FIGURA 102 - Compilação do modelo com otimizado com Adam

```
#Otimização do modelo com gradiente descendente, avaliando com acurácia  
#OBS: A função de perda escolhida é mais adequada a problemas de classificação, quando o índice é inteiro e representa a classe  
  
model.compile(optimizer='adam',  
              loss='sparse_categorical_crossentropy',  
              metrics=['accuracy'])
```

Treinamos o modelo com 10 épocas

FIGURA 103 - Treinamento do modelo em 10 épocas

```
# Treinamento com 10 épocas
history = model.fit(x_train, y_train, epochs=10, validation_data=(x_test, y_test))
```

Epoch 1/10
1532/1532 ————— 10s 5ms/step - accuracy: 0.7703 - loss: 0.6625 - val_accuracy: 0.8480 - val_loss: 0.4238
Epoch 2/10
1532/1532 ————— 7s 3ms/step - accuracy: 0.8562 - loss: 0.4060 - val_accuracy: 0.8621 - val_loss: 0.3768
Epoch 3/10
1532/1532 ————— 7s 4ms/step - accuracy: 0.8704 - loss: 0.3592 - val_accuracy: 0.8693 - val_loss: 0.3607
Epoch 4/10
1532/1532 ————— 11s 5ms/step - accuracy: 0.8818 - loss: 0.3254 - val_accuracy: 0.8709 - val_loss: 0.3496
Epoch 5/10
1532/1532 ————— 7s 4ms/step - accuracy: 0.8878 - loss: 0.3057 - val_accuracy: 0.8709 - val_loss: 0.3460
Epoch 6/10
1532/1532 ————— 9s 4ms/step - accuracy: 0.8906 - loss: 0.2961 - val_accuracy: 0.8790 - val_loss: 0.3352
Epoch 7/10
1532/1532 ————— 12s 5ms/step - accuracy: 0.8987 - loss: 0.2701 - val_accuracy: 0.8669 - val_loss: 0.3686
Epoch 8/10
1532/1532 ————— 6s 4ms/step - accuracy: 0.8995 - loss: 0.2670 - val_accuracy: 0.8768 - val_loss: 0.3354
Epoch 9/10
1532/1532 ————— 13s 6ms/step - accuracy: 0.9058 - loss: 0.2552 - val_accuracy: 0.8851 - val_loss: 0.3178
Epoch 10/10
1532/1532 ————— 11s 6ms/step - accuracy: 0.9077 - loss: 0.2413 - val_accuracy: 0.8921 - val_loss: 0.2994

Avaliamos o resultado.

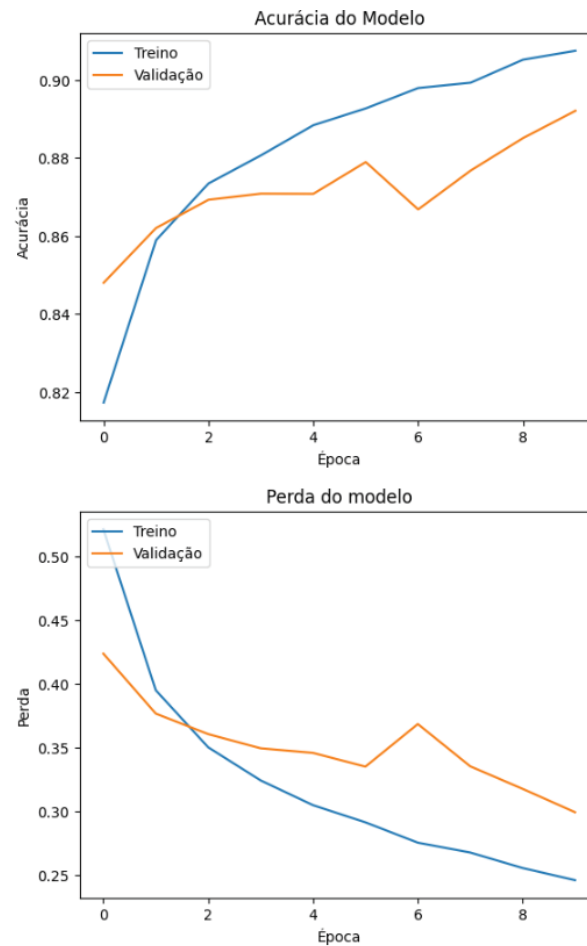
FIGURA 104 - Avaliação do resultado

```
#Checando resultados
print("Pontuação de treinamento: ", model.evaluate(x_train,y_train))
print("Pontuação de teste: ", model.evaluate(x_test, y_test))
```

1532/1532 ————— 3s 2ms/step - accuracy: 0.9210 - loss: 0.2159
Pontuação de treinamento: [0.21703098714351654, 0.9197347164154053]
657/657 ————— 1s 1ms/step - accuracy: 0.8954 - loss: 0.2966
Pontuação de teste: [0.2994283437728882, 0.8921428322792053]

Geramos gráficos de acurácia e perda do processo de treinamento.

FIGURA 105 - Acurácia e perda do modelo



Aplicamos um exemplo de predição.

FIGURA 106 - Teste de predição

```
#Exemplo de predição do modelo
y_pred = model.predict(x_test).argmax(axis=1)
print(y_pred)
```

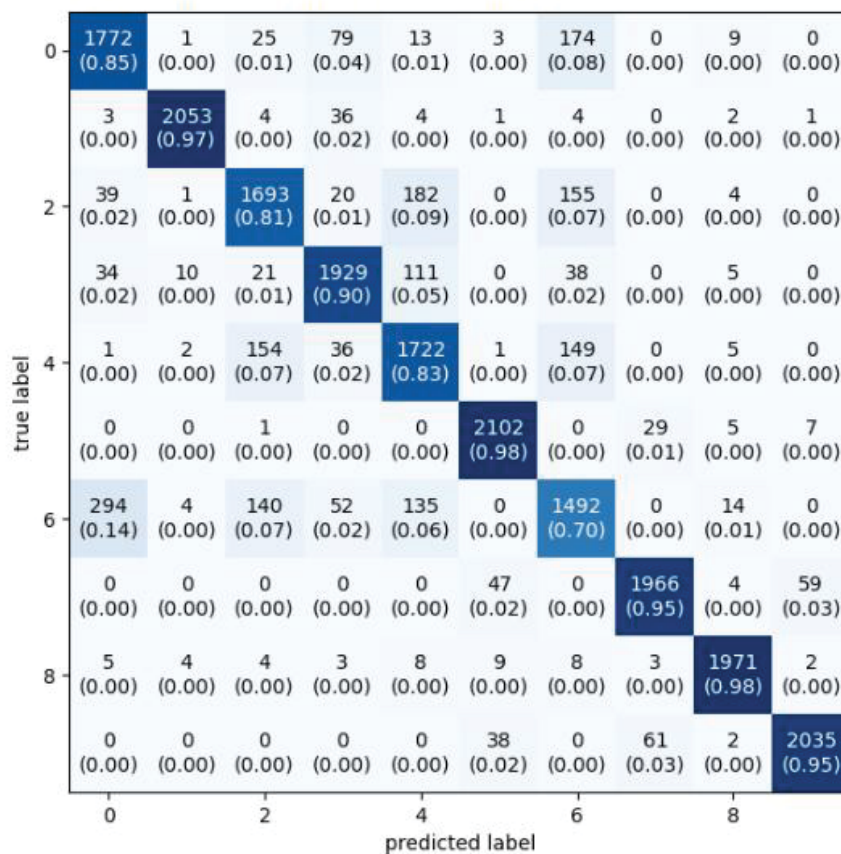
657/657 ————— 1s 2ms/step
[2 7 0 ... 4 7 7]

Criamos a matriz de confusão para avaliar o modelo.

FIGURA 107 - Avaliação do modelo com matriz de confusão

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

(<Figure size 700x700 with 1 Axes>,
<Axes: xlabel='predicted label', ylabel='true label'>)



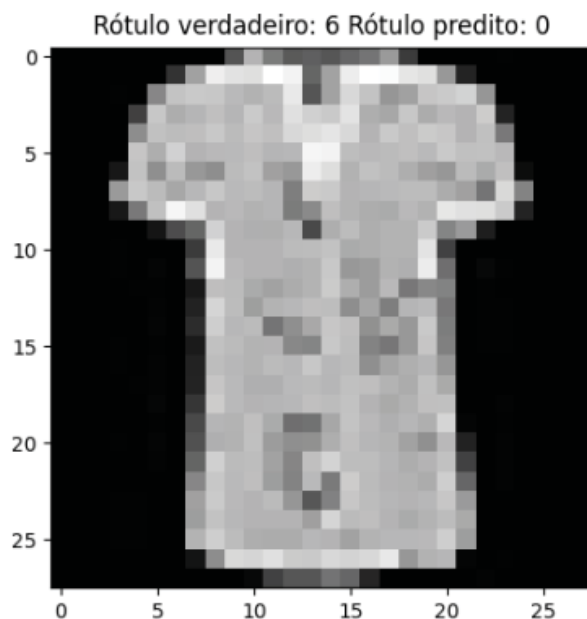
Demonstramos uma predição errada.

FIGURA 108 - Demonstração da classificação

```
# Demonstrando uma predição errada
misclassified = np.where(y_pred != y_test)[0]
i = np.random.choice(misclassified)

plt.imshow(x_test[i].reshape(28, 28), cmap="gray")
plt.title("Rótulo verdadeiro: %s Rótulo predito: %s" % (y_test[i], y_pred[i]))

Text(0.5, 1.0, 'Rótulo verdadeiro: 6 Rótulo predito: 0')
```



Seguindo os rótulos de referência identificamos melhor o que foi predito errado.

FIGURA 109 - Rótulos de identificação

Rótulos da biblioteca (para referência):

<https://github.com/zalandoresearch/fashion-mnist>

Código	Rótulo
0	Camiseta/top
1	Calça
2	Suéteres
3	Vestido
4	Casacos
5	Sandálias
6	Camisa
7	Tênis
8	Bolsa
9	Suéteres
2	Bota de tornozelo

2. RNA COM REGRESSÃO

Para aplicação do modelo de RNA com regressão na base “Wine”, iniciamos carregando as bibliotecas necessárias.

FIGURA 110 - Importação das bibliotecas

```
# Bibliotecas
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from tensorflow.python.keras import backend
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_squared_error
from math import sqrt
```

Importamos a base de dados

FIGURA 111 - Download base de dados de Vinhos

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

Traduzimos as colunas para facilitar o entendimento.

FIGURA 112 - Tradução dos títulos das colunas

```
#Traduzindo colunas para entender melhor os dados
df = pd.DataFrame(data)

df.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
]

print(df)
```

O resultado dos dados.

FIGURA 113 - Descrição dos dados

	acidez_fixa	acidez_volatil	acido_citrico	acucar_residual	cloretos	\
0	7.4	0.700	0.00	1.9	0.076	
1	7.8	0.880	0.00	2.6	0.098	
2	7.8	0.760	0.04	2.3	0.092	
3	11.2	0.280	0.56	1.9	0.075	
4	7.4	0.700	0.00	1.9	0.076	
...	...	...	...	...	...	
1594	6.2	0.600	0.08	2.0	0.090	
1595	5.9	0.550	0.10	2.2	0.062	
1596	6.3	0.510	0.13	2.3	0.076	
1597	5.9	0.645	0.12	2.0	0.075	
1598	6.0	0.310	0.47	3.6	0.067	

	dioxido_de_enxofre_livre	dioxido_de_enxofre_total	densidade	pH	\
0	11.0	34.0	0.99780	3.51	
1	25.0	67.0	0.99680	3.20	
2	15.0	54.0	0.99700	3.26	
3	17.0	60.0	0.99800	3.16	
4	11.0	34.0	0.99780	3.51	
...	...	...	...	...	
1594	32.0	44.0	0.99490	3.45	
1595	39.0	51.0	0.99512	3.52	
1596	29.0	40.0	0.99574	3.42	
1597	32.0	44.0	0.99547	3.57	
1598	18.0	42.0	0.99549	3.39	

	sulfatos	alcool	score_qualidade_vinho
0	0.56	9.4	5
1	0.68	9.8	5
2	0.65	9.8	5
3	0.58	9.8	6
4	0.56	9.4	5
...	...	...	...
1594	0.58	10.5	5
1595	0.76	11.2	6
1596	0.75	11.0	6
1597	0.71	10.2	5
1598	0.66	11.0	6

[1599 rows x 12 columns]

Separamos os dados e removemos primeira coluna.

FIGURA 114 - Remoção da primeira coluna

```
# Separação dos dados, removendo primeira coluna
X = df.iloc[1:, :-1].values.astype(float)
Y = df.iloc[1:, -1].values.astype(float)

print(X)

[[ 7.8    0.88    0.    ...   3.2    0.68    9.8   ]
 [ 7.8    0.76    0.04   ...   3.26   0.65    9.8   ]
 [11.2    0.28    0.56   ...   3.16   0.58    9.8   ]
 ...
 [ 6.3    0.51    0.13   ...   3.42    0.75   11.    ]
 [ 5.9    0.645   0.12   ...   3.57    0.71   10.2   ]
 [ 6.     0.31    0.47   ...   3.39    0.66   11.    ]]
```

Separamos os dados em treinamento e teste.

FIGURA 115 - Separação em teste e treinamento

```
# Separação para treinamento e teste do modelo
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.25)
```

Construímos o modelo.

FIGURA 116 - Definição do modelo

```
# Definição do modelo

i = tf.keras.layers.Input(shape=(11,))
x = tf.keras.layers.Dense(50, activation="relu")(i)
x = tf.keras.layers.Dense(1)(x)

model = tf.keras.models.Model(i, x)
```

Criamos funções de avaliação.

FIGURA 117 - Funções de avaliação do modelo

```
# Funções de avaliação da regressão

def rmse(y_true, y_pred):
    return backend.sqrt(backend.mean(backend.square(y_pred - y_true), axis=-1))

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

Definimos a otimização do modelo.

FIGURA 118 - Otimização do modelo com Adam

```
#Definição do modelo

optimizer=tf.keras.optimizers.Adam(learning_rate=0.05)
# optimizer=tf.keras.optimizers.SGD(learning_rate=0.2, momentum=0.5)
# optimizer=tf.keras.optimizers.RMSprop(0.01)

model.compile(optimizer=optimizer,
              loss="mse",
              metrics=[rmse, r2])
```

Funções para treinamento.

FIGURA 119 - Treinamento do modelo

```
# Callback para parar treinamento
early_stop = tf.keras.callbacks.EarlyStopping(
    monitor='val_loss',
    patience=20,
    restore_best_weights=True)

# Épocas de treinamento com callback
history = model.fit(x_train, y_train,
                   epochs=1500,
                   validation_data=(x_test, y_test),
                   callbacks=[early_stop])
```

Rodamos o treinamento.

FIGURA 120 - Treinamento do modelo

```
Epoch 1/1500 2s 46ms/step - loss: 82.6954 - r2: -5861.6753 - rmse: 6.3619 - val_loss: 1.7435 - val_r2: -100.9685 - val_rmse: 1.2592
Epoch 2/1500 38/38 1s 15ms/step - loss: 1.8075 - r2: -121.0796 - rmse: 1.3028 - val_loss: 0.9463 - val_r2: -67.4078 - val_rmse: 1.0563
Epoch 3/1500 38/38 2s 5ms/step - loss: 0.7833 - r2: -54.5655 - rmse: 0.9920 - val_loss: 0.7471 - val_r2: -47.9945 - val_rmse: 0.9218
Epoch 4/1500 38/38 0s 2ms/step - loss: 0.7683 - r2: -50.4173 - rmse: 0.9508 - val_loss: 0.5810 - val_r2: -43.8615 - val_rmse: 0.9086
Epoch 5/1500 38/38 0s 2ms/step - loss: 0.6272 - r2: -43.2521 - rmse: 0.9453 - val_loss: 0.5017 - val_r2: -40.8692 - val_rmse: 0.8728
Epoch 6/1500 38/38 0s 2ms/step - loss: 0.6083 - r2: -43.8322 - rmse: 0.9345 - val_loss: 0.4689 - val_r2: -39.4939 - val_rmse: 0.8605
Epoch 7/1500 38/38 0s 2ms/step - loss: 0.5496 - r2: -42.5668 - rmse: 0.8981 - val_loss: 0.5372 - val_r2: -46.5834 - val_rmse: 0.9152
Epoch 8/1500 38/38 0s 6ms/step - loss: 0.5795 - r2: -42.9166 - rmse: 0.9410 - val_loss: 0.4453 - val_r2: -38.4823 - val_rmse: 0.8520
Epoch 9/1500 38/38 0s 2ms/step - loss: 0.5105 - r2: -40.5997 - rmse: 0.9179 - val_loss: 0.5246 - val_r2: -38.8942 - val_rmse: 0.8559
Epoch 10/1500 38/38 0s 3ms/step - loss: 0.5056 - r2: -41.5497 - rmse: 0.9063 - val_loss: 0.4446 - val_r2: -39.4295 - val_rmse: 0.8614
Epoch 11/1500 38/38 0s 2ms/step - loss: 0.5030 - r2: -42.6395 - rmse: 0.9095 - val_loss: 0.4362 - val_r2: -36.6058 - val_rmse: 0.8346
Epoch 12/1500 38/38 0s 2ms/step - loss: 0.5046 - r2: -41.6447 - rmse: 0.9125 - val_loss: 0.5794 - val_r2: -43.7684 - val_rmse: 0.9008
Epoch 13/1500 38/38 0s 8ms/step - loss: 0.5218 - r2: -43.4222 - rmse: 0.9134 - val_loss: 0.4211 - val_r2: -39.9609 - val_rmse: 0.8667
Epoch 14/1500 38/38 0s 2ms/step - loss: 0.4874 - r2: -42.4220 - rmse: 0.9115 - val_loss: 0.4585 - val_r2: -45.2604 - val_rmse: 0.9086
Epoch 15/1500 38/38 0s 2ms/step - loss: 0.5451 - r2: -45.3555 - rmse: 0.9532 - val_loss: 0.4151 - val_r2: -40.9818 - val_rmse: 0.8759
Epoch 16/1500 38/38 0s 2ms/step - loss: 0.4804 - r2: -42.5756 - rmse: 0.9343 - val_loss: 0.4336 - val_r2: -43.1451 - val_rmse: 0.8942
Epoch 17/1500 38/38 0s 2ms/step - loss: 0.4665 - r2: -44.9799 - rmse: 0.9198 - val_loss: 0.5208 - val_r2: -50.4987 - val_rmse: 0.9480
Epoch 18/1500 38/38 0s 2ms/step - loss: 0.5117 - r2: -45.9018 - rmse: 0.9733 - val_loss: 0.4162 - val_r2: -40.8887 - val_rmse: 0.8744
Epoch 19/1500 38/38 0s 2ms/step - loss: 0.5156 - r2: -45.8186 - rmse: 0.9728 - val_loss: 0.4263 - val_r2: -41.6414 - val_rmse: 0.8767
Epoch 20/1500 38/38 0s 2ms/step - loss: 0.4739 - r2: -46.0221 - rmse: 0.9295 - val_loss: 0.5355 - val_r2: -53.1375 - val_rmse: 0.9659
Epoch 21/1500 38/38 0s 2ms/step - loss: 0.5659 - r2: -52.7100 - rmse: 0.9786 - val_loss: 0.4740 - val_r2: -41.4287 - val_rmse: 0.8768
Epoch 22/1500 38/38 0s 2ms/step - loss: 0.4531 - r2: -47.2599 - rmse: 0.9269 - val_loss: 0.4451 - val_r2: -40.9830 - val_rmse: 0.8735
Epoch 23/1500 38/38 0s 2ms/step - loss: 0.4500 - r2: -44.7042 - rmse: 0.9208 - val_loss: 0.5199 - val_r2: -42.8665 - val_rmse: 0.8886
Epoch 24/1500 38/38 0s 10ms/step - loss: 0.4342 - r2: -45.4082 - rmse: 0.9143 - val_loss: 0.4403 - val_r2: -40.0274 - val_rmse: 0.8653
Epoch 25/1500 38/38 0s 3ms/step - loss: 0.4382 - r2: -43.8237 - rmse: 0.9117 - val_loss: 0.4461 - val_r2: -40.9070 - val_rmse: 0.8726
Epoch 26/1500 38/38 0s 3ms/step - loss: 0.4996 - r2: -48.9225 - rmse: 0.9567 - val_loss: 0.4725 - val_r2: -41.4147 - val_rmse: 0.8779
Epoch 27/1500 38/38 0s 3ms/step - loss: 0.4841 - r2: -46.1856 - rmse: 0.9288 - val_loss: 0.4082 - val_r2: -40.3861 - val_rmse: 0.8701
Epoch 28/1500 38/38 0s 3ms/step - loss: 0.4310 - r2: -44.3685 - rmse: 0.9366 - val_loss: 0.4053 - val_r2: -40.6265 - val_rmse: 0.8729
Epoch 29/1500 38/38 1s 20ms/step - loss: 0.4824 - r2: -44.8805 - rmse: 0.9729 - val_loss: 0.5177 - val_r2: -52.9040 - val_rmse: 0.9655
```

Resultado do treinamento.

FIGURA 121 - Avaliação do resultado do treinamento

```
# Resultados do treinamento e teste
print("Resultado treinamento: ", model.evaluate(x_train, y_train))
print("Resultado teste: ", model.evaluate(x_test, y_test))

38/38 ————— 1s 15ms/step - loss: 0.4643 - r2: -42.2929 - rmse: 0.9339
Resultado treinamento: [0.42607665061950684, 0.9177801609039307, -41.79179382324219]
13/13 ————— 0s 16ms/step - loss: 0.3757 - r2: -42.1896 - rmse: 0.8604
Resultado teste: [0.3936346471309662, 0.8682646155357361, -40.01666259765625]
```

Criamos os gráficos de erro quadrático médio e perda e coeficiente de determinação do modelo durante o treinamento

FIGURA 122 - Erro quadrático médio e coeficiente de determinação

```
# Gráficos durante o treinamento e teste

plt.plot(history.history['rmse'])
plt.plot(history.history['val_rmse'])
plt.title('Erro quadrático médio do modelo')
plt.ylabel('RMSE')
plt.xlabel('Época')
plt.legend(['Treino', 'Validação'], loc='upper right')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Perda do modelo')
plt.ylabel('Perda')
plt.xlabel('Época')
plt.legend(['Treino', 'Validação'], loc='upper right')
plt.show()

plt.plot(history.history['r2'])
plt.plot(history.history['val_r2'])
plt.title('Coeficiente de determinação do modelo')
plt.ylabel('Coeficiente')
plt.xlabel('Época')
plt.legend(['Treino', 'Validação'], loc='lower right')
plt.show()
```

GRÁFICO 18 – Erro quadrático médio e perda do modelo

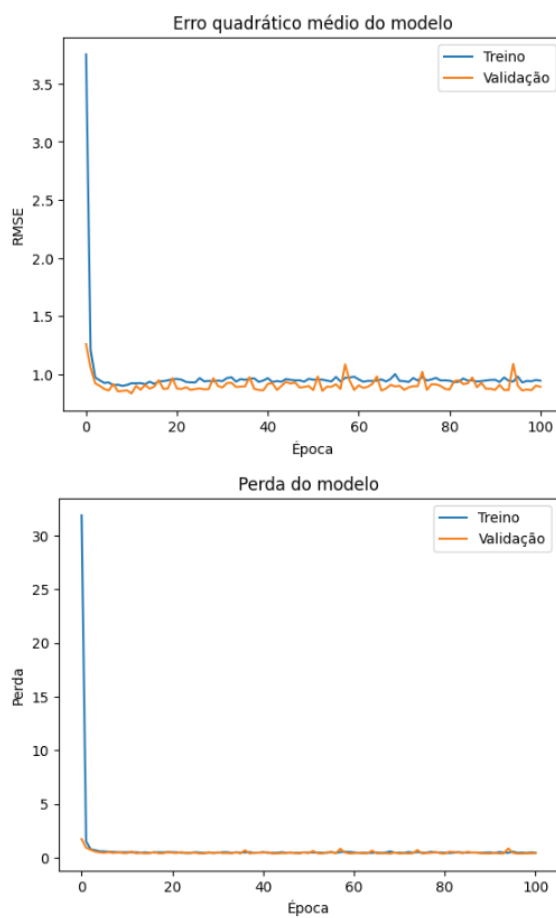
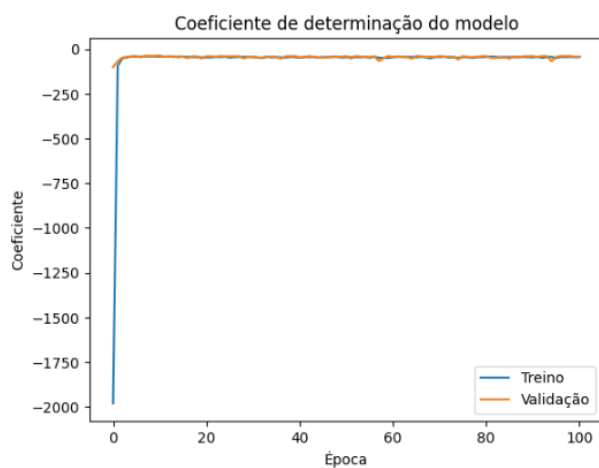


GRÁFICO 19 - Coeficiente de determinação do modelo



Executamos a predição do modelo.

FIGURA 123 - Predição do modelo

```
# Predição com o modelo

y_pred = model.predict(x_test).flatten()
print(y_pred)
```

4.8672523	5.977433	5.298209	5.883625	5.7786818	5.991962	5.340973
6.1581817	5.6632643	5.701598	6.037753	6.592324	5.328522	5.088885
5.543335	6.060793	6.2033653	5.910296	5.0974007	4.8263865	6.5213995
5.7125206	5.5288973	5.6978626	5.654664	5.797232	5.001116	5.2429886
5.8926153	5.7402964	5.1138244	5.9868255	5.8160458	6.5218925	5.1692305
5.2375326	5.643093	5.9601054	6.4342165	5.8852177	5.191886	5.660449
5.473385	5.9333515	5.0538883	5.9096622	5.847824	5.245156	6.3149676
5.2928	5.423332	5.654597	6.343215	5.261221	4.9001484	6.203624
5.865238	6.317827	5.222643	5.40904	5.9434376	5.060213	5.5843487
5.6030917	5.170514	5.4933786	5.333521	5.8941746	5.440053	6.153892
5.731764	5.196306	6.1162415	5.481457	5.4430976	5.4335194	5.830949
5.413283	5.753912	6.200671	5.595646	5.1229544	5.767761	5.2502975
5.648843	5.223076	5.984506	4.9218884	5.1493707	5.8118324	5.543631
6.409523	5.0480733	5.2112985	5.34209	5.149371	5.876257	6.0162506
5.367725	5.5663157	5.7703567	5.1698637	5.30606	5.296498	5.295879
5.3604403	6.317827	5.0549192	5.2457175	5.0301123	5.968399	6.2390585
5.855746	4.936945	5.6481886	5.3887854	5.5196886	4.9775815	5.6676555
5.3519154	5.7045937	5.454116	5.2688456	6.3520193	4.998802	6.3672237
5.0026817	6.2659483	5.7076006	5.046815	5.583876	6.4524803	5.556161
6.233242	6.0616426	5.2689157	5.5693808	4.944804	5.3336825	5.96939
5.452164	5.754266	5.8691015	5.0118914	6.1105065	6.0929976	6.175446
5.075632	5.0685153	5.1603785	5.4498243	5.3957634	5.3466015	5.814055
6.4380355	6.281309	5.883625	5.3541822	5.7204933	6.0582166	5.340782
5.3296924	4.947322	6.0659356	5.730406	5.3336825	5.22814	5.3279305
5.533479	5.3085775	5.580185	5.3591986	5.792576	5.36333	4.7873297
4.95238	6.207378	5.155093	6.254943	6.201291	6.316066	5.0457497
4.864609	5.337472	5.5929966	6.201198	5.0517683	5.4790206	5.493619
5.048253	6.343215	6.1178217	5.814055	5.9873915	5.3007703	6.0271287
5.4833956	5.0045357	4.5467687	5.0127096	6.190041	5.445063	5.512149
5.8328466	6.346902	5.6817575	5.3390765	5.9023194	5.4635344	6.0668955
5.943298	4.8528056	5.784584	6.0461316	5.3068495	6.453708	5.1070232
5.723259	5.614018	5.670738	6.150578	5.8983192	5.31737	5.5162416
5.3380485	5.333521	5.1571136	5.3541822	5.485459	6.0704794	4.9264107
6.3636117	5.8004894	5.951803	5.1119504	5.648406	5.0942917	5.640893
5.5299044	5.701598	5.3185205	5.244891	5.1690855	6.0582166	5.9499345
5.07246	6.35772	5.617006	5.0615225	5.6442018	6.233242	5.7233057
5.4488173	5.2284894	6.0005994	5.798398	5.5205193	4.8556585	6.1015077
6.15855	5.617347	5.4541197	5.315198	5.420233	5.090685	5.194495
5.8603225	5.9437723	5.38017	5.566826	6.072126	5.1666613	6.124735
6.0715165	4.932083	6.3842263	5.1977224	4.971037	5.373157	6.0237617
6.017794	5.658801	5.994917	5.8749847	5.016589	5.4938116	6.193963
5.0301123	5.7984776	4.919219	5.2490177	5.07262	4.847314	4.791117
5.2573223	6.05501	5.261333	5.527498	6.16512	4.920385	5.935751
5.3737955	6.1614747	5.6987104	5.953808	5.1309214	6.1461368	5.506214
4.9722576	4.938671	5.2931404	5.5626698	5.1540985	5.096057	5.1900177
5.4302206	5.5929966	5.772382	4.9823294	5.763646	5.432001	5.461852
5.0849237	6.113697	4.8648567	6.0817227	6.0206275	5.4188623	5.2881875
5.0003247	6.1525307	5.1955338	6.110401	5.4713354	5.3796597	5.4797373
5.474763	5.8983755	5.333611	5.378911	5.400511	5.515419	5.2590427
5.343156	5.9645696	6.2085557	5.350751	5.1430297	5.5651608	5.1571836
6.0953245	5.833683	5.192988	5.5082564	5.694469	5.7404046	6.142609
6.2553716	5.747788	5.0373564	5.504937	4.9667745	5.027594	5.2595205
4.931018	5.3307133	5.633752	5.135465	6.022397	4.9148765	5.4558115
5.5287046	5.0111904	5.684088	5.6066785	5.6676555	5.7558026	5.3420734
5.264943	5.593643	5.6259933	5.9581947	5.2596064	6.2995005	5.7849035
5.2590427	6.307364	5.833934	5.5520678	5.0480237	5.477087	5.2683115
6.1745977	5.8075333	5.153427	5.6164255	6.5782733	6.002743	6.1680903
5.1307864]						

Avaliamos as métricas de sucesso do modelo.

FIGURA 124 - Avaliação do modelo

```
# Avaliando modelo treinado
mse = mean_squared_error(y_test, y_pred)
rmse = sqrt(mse)
r2 = r2_score(y_test, y_pred)

print("mse      = ", mse)
print("rmse     = ", rmse)
print("r2       = ", r2)

mse      = 0.3936346721908853
rmse     = 0.6274031177726848
r2       = 0.3422361379981237
```

O modelo de previsão da qualidade do vinho obteve os seguintes resultados:

MSE de 0.3936: Isso significa que, em média, a diferença entre as previsões do modelo e os valores reais de qualidade, quando elevada ao quadrado, é de 0.4038. Quanto menor esse valor, melhor, então 0.4038 é um erro razoável, mas há espaço para melhorias.

RMSE de 0.6274: Este é o erro médio do modelo, mas em unidades diretas de qualidade do vinho. Ou seja, em média, o modelo erra por cerca de 0.64 unidades na escala de qualidade. Novamente, não é um erro gigantesco, mas também não é extremamente preciso.

R^2 de 0.3422: Isso quer dizer que o modelo consegue explicar cerca de 40% da variação na qualidade do vinho. Ou seja, ele tem uma boa capacidade explicativa, mas ainda há uma grande parte da variação que não é explicada pelas características do vinho usadas no modelo.

Em resumo, o modelo está razoavelmente bom, mas há margem para melhorar. Ele consegue capturar uma parte significativa da variação da qualidade do vinho, mas não de forma perfeita, e ainda com um erro médio considerável.

3. SISTEMA DE RECOMENDAÇÃO

Para o sistema de recomendação de livros “Books”, iniciamos com a importação das bibliotecas necessárias.

FIGURA 125 - Importando as bibliotecas

```
#Bibliotecas

import tensorflow as tf
from tensorflow.keras.layers import Input, Dense, Embedding, Flatten, Concatenate
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import SGD, Adam

from sklearn.utils import shuffle

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Fizemos a importação da base via upload do Google Colab.

FIGURA 126 - Upload da base via Google Colab

```
# Importação da base.csv (via ação do usuario)
from google.colab import files
import io
uploaded = files.upload()
```

Carregamos a base de dados.

FIGURA 127 - Base de dados carregada

```
#Carregando base importada (usamos o on_bad_lines para evitar falhas)
import io
filename = next(iter(uploaded))
df = pd.read_csv(filename, delimiter=",", on_bad_lines="skip")
df
```

	ISBN	Titulo	Autor	Ano	Editora	ID_usuario	Notas
0	2005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	276725	0
1	60973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial	276726	2
2	374157065	Flu: The Story of the Great Influenza Pandemic...	Gina Bari Kolata	1999	Farrar Straus Giroux	276727	6
3	393045218	The Mummies of Urumchi	E. J. W. Barber	1999	W. W. Norton & Company	276729	1
4	399135782	The Kitchen God's Wife	Amy Tan	1991	Putnam Pub Group	276729	9
...	...	...	...	...	...	...	...
108966	743436865	Funny Money	James Swain	2002	Atria	29888	10
108967	451410394	Sun Valley	Gena Hale	2002	Signet Book	29888	10
108968	345391101	The Adventures of Lando Calrissian: Lando Calr...	L. Neil Smith	1994	Del Rey Books	29888	6
108969	1591050480	Beloved Stranger	Patricia Crossley	2002	Novelbooks	29888	9
108970	051732170X	Who's Who in the Bible	RONALD BROWNRIGG	1993	Testament	29888	4

108971 rows x 7 columns

Definimos os embeddings do modelo.

FIGURA 128 - *Embeddings* do modelo

```
#Embeddings (ISBN e ID_usuario precisam ser categóricas)

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

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

# Dimensões
N = len(set(df.ISBN))
M = len(set(df.ID_usuario))

# dimensão do embedding (tentar outros)
K = 10
```

Definimos o modelo.

FIGURA 129 - Definição do modelo

```
# Livro
u = Input(shape=(1,))
u_emb = Embedding(N, K)(u) # saída : num_samples, 1, K
u_emb = Flatten()(u_emb)   # saída : num_samples, K

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

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

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

model = Model(inputs=[u, m], outputs=x)
```

Definimos a otimização.

FIGURA 130 - Compilando o modelo com otimizador SGD

```
# Criação do modelo
model.compile(
    loss="mse",
    optimizer=SGD(learning_rate=0.08, momentum=0.9)
)
```

Separamos a base em treino e teste.

FIGURA 131 - Separação em treino e teste

```
# Separando os dados para treino e teste (considerando apenas 3 itens da tabela)

ISBN, ID_usuario, Notas = shuffle(df.ISBN, df.ID_usuario, df.Notas)

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

train_ISBN = ISBN[:Ntrain]
train_ID_usuario = ID_usuario[:Ntrain]
train_Notas = Notas[:Ntrain]

test_ISBN = ISBN[Ntrain:]
test_ID_usuario = ID_usuario[Ntrain:]
test_Notas = Notas[Ntrain:]

avg_Notas = train_Notas.mean()
train_Notas = train_Notas - avg_Notas
test_Notas = test_Notas - avg_Notas
```

Treinamos o modelo.

FIGURA 132 - Treinamento modelo

```
# Treinamento do modelo em épocas
epochs = 25

history = model.fit(
    x=[train_ISBN, train_ID_usuario],
    y=train_Notas,
    epochs=epochs,
    batch_size=1024,
    verbose=2, # não imprime o progresso
    validation_data=([test_ISBN, test_ID_usuario], test_Notas)
)
```

E o resultado de 25 épocas.

FIGURA 133 - Treinamento em 25 épocas

```

Epoch 1/25
86/86 - 2s - 21ms/step - loss: 9.9849 - val_loss: 9.9525
Epoch 2/25
86/86 - 2s - 29ms/step - loss: 9.9861 - val_loss: 9.9799
Epoch 3/25
86/86 - 0s - 2ms/step - loss: 9.9822 - val_loss: 9.9550
Epoch 4/25
86/86 - 0s - 3ms/step - loss: 9.9842 - val_loss: 9.9620
Epoch 5/25
86/86 - 0s - 4ms/step - loss: 9.9144 - val_loss: 10.0839
Epoch 6/25
86/86 - 0s - 4ms/step - loss: 9.3058 - val_loss: 18.7664
Epoch 7/25
86/86 - 1s - 11ms/step - loss: 3.9738 - val_loss: 11.4292
Epoch 8/25
86/86 - 1s - 13ms/step - loss: 1.2417 - val_loss: 10.7514
Epoch 9/25
86/86 - 1s - 6ms/step - loss: 0.3277 - val_loss: 10.5730
Epoch 10/25
86/86 - 0s - 4ms/step - loss: 0.1753 - val_loss: 10.5511
Epoch 11/25
86/86 - 0s - 3ms/step - loss: 0.1349 - val_loss: 10.5340
Epoch 12/25
86/86 - 0s - 2ms/step - loss: 0.1148 - val_loss: 10.5525
Epoch 13/25
86/86 - 0s - 2ms/step - loss: 0.1076 - val_loss: 10.5347
Epoch 14/25
86/86 - 0s - 2ms/step - loss: 0.1027 - val_loss: 10.5434
Epoch 15/25
86/86 - 0s - 3ms/step - loss: 0.0975 - val_loss: 10.5739
Epoch 16/25
86/86 - 0s - 3ms/step - loss: 0.0942 - val_loss: 10.5269
Epoch 17/25
86/86 - 0s - 3ms/step - loss: 0.0895 - val_loss: 10.5678
Epoch 18/25
86/86 - 0s - 3ms/step - loss: 0.0797 - val_loss: 10.5565
Epoch 19/25
86/86 - 0s - 2ms/step - loss: 0.0660 - val_loss: 10.5521
Epoch 20/25
86/86 - 0s - 3ms/step - loss: 0.0532 - val_loss: 10.5563
Epoch 21/25
86/86 - 0s - 3ms/step - loss: 0.0402 - val_loss: 10.5592
Epoch 22/25
86/86 - 0s - 3ms/step - loss: 0.0296 - val_loss: 10.5569
Epoch 23/25
86/86 - 0s - 2ms/step - loss: 0.0210 - val_loss: 10.5609
Epoch 24/25
86/86 - 0s - 3ms/step - loss: 0.0147 - val_loss: 10.5610
Epoch 25/25
86/86 - 0s - 2ms/step - loss: 0.0100 - val_loss: 10.5587

```

FIGURA 134 - Gerando gráfico de perda do treinamento

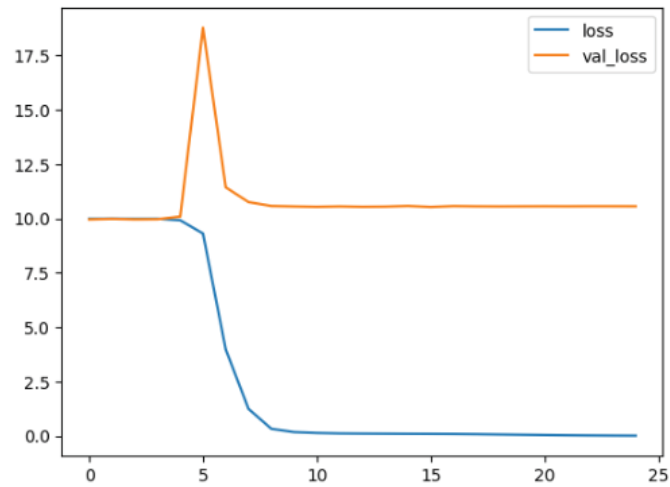
```

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

```

Gráfico de perda do modelo durante treinamento.

GRÁFICO 20 - Gráfico de perda durante o treinamento



Executamos um teste de recomendação para um usuário gerado.

FIGURA 135 - Teste de recomendação do modelo treinado

```
# Gerar o array com o usuário único
# repete a quantidade de livros
input_usuario = np.repeat(a=29888, repeats=N)
books = np.array(list(set(ISBN)))

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

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

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

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

```
3406/3406 ————— 12s 4ms/step
Recomendação: Livro - 2412 / 1.1802464 *
```

```
# Identificação do título sugerido para usuário
print( df[df['ISBN'] == books[idx]].values[0])

[2412 'The Perfect Storm: A True Story of Men Against the Sea'
'Sebastian Junger' 1997 'Little Brown and Company' 10202 10]
```

4 DEEPPDREAM

Para criação através do modelo deepdream, iniciamos com a carga das bibliotecas necessárias

FIGURA 136 - Importação de bibliotecas para daydream

```
#Bibliotecas

import tensorflow as tf
import numpy as np
import matplotlib as mpl
import IPython.display as display
import PIL.Image
```

Selecionamos uma imagem de referência.

FIGURA 137 – Definindo imagem base

```
# Imagem de referência
url = "https://commons.wikimedia.org/wiki/Special:FilePath/Felis_catus-cat_on_snow.jpg"
```

Carregamos a imagem e diminuimos o tamanho para facilitar a RNN

FIGURA 138 - Ajuste da imagem para RNN

```
# Download da imagem e gravação em array Numpy
def download(url, max_dim=None):
    name = url.split('/')[-1]
    image_path = tf.keras.utils.get_file(name, origin=url)
    img = PIL.Image.open(image_path)
    if max_dim:
        img.thumbnail((max_dim, max_dim))
    return np.array(img)

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

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

# Redução do tamanho da imagem para facilitar o trabalho da RNN
original_img = download(url, max_dim=500)
show(original_img)
display.display(display.HTML('Image cc-by: <a href=https://commons.wikimedia.org/wiki/File:Felis_catus-cat_on_snow.jpg>Von.grzanka</a>'))
```

FIGURA 139 - Download da imagem base

Downloading data from https://commons.wikimedia.org/wiki/Special:FilePath/Felis_catus-cat_on_snow.jpg
2125399/2125399 — 1s 1us/step



Image cc-by: [Von.grzanka](#)

Baixamos um modelo já treinado.

FIGURA 140 - Baixando modelo RNN treinado

```
#Modelo de base já treinado
base_model = tf.keras.applications.InceptionV3(include_top=False, weights='imagenet')

Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception\_v3/inception\_v3\_weights\_tf\_dim\_ordering\_tf\_kernels\_notop.h5
87910968/87910968 ————— 8s 0us/step
```

Habilitamos o modelo.

FIGURA 141 - Criação do modelo e ativando camadas

```
# Maximizando as ativações das camadas
names = ['mixed3', 'mixed5']
layers = [base_model.get_layer(name).output for name in names]

# Criação do modelo
dream_model = tf.keras.Model(inputs=base_model.input, outputs=layers)
```

Definimos função de perda

FIGURA 142 - Calculo da perda

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

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

    return tf.reduce_sum(losses)
```

Criamos a classe base do modelo Depdream.

FIGURA 143 - Classe Deepdream

```

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

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

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

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

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

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

        return loss, img

```

Instanciamos o modelo e aplicamos a função de perda.

FIGURA 144 - Execução do Deep Dream

```

deepdream = DeepDream(dream_model)

def run_deep_dream_simple(img, steps=100, step_size=0.01):

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

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

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

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

    return result

```


Executamos o modelo.

FIGURA 145 - Resultado da execução do Deep Dream

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



Aplicamos agora a escala de uma oitava para refinar o resultado

FIGURA 146 - Aplicação de escala de uma oitava

```

import time
start = time.time()

OCTAVE_SCALE = 1.30

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

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

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

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

display.clear_output(wait=True)
img = tf.image.resize(img, base_shape)
img = tf.image.convert_image_dtype(img/255.0, dtype=tf.uint8)
show(img)

end = time.time()
end-start

```



16.928105115890503

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

1. APURAÇÃO E VISUALIZAÇÃO

Com o objetivo de ter uma visão de quais países investem ou investiram mais em Inteligência Artificial até 2024, busquei bases de dados de pesquisas prévias no Kaggle. Encontrei três bases de dados, a saber:

1. AI Companies Dataset (<https://www.kaggle.com/datasets/vineethakkinapalli/ai-companies>) . A base lista 3096 startups de IA ao redor do mundo;
2. AI/ML Salaries (<https://www.kaggle.com/datasets/cedricaubin/ai-ml-salaries>). A base de dados tem pouco mais de 73 mil vagas que envolvam IA de alguma forma;
3. 100 AI Companies of 2024 (<https://www.kaggle.com/datasets/raniritu/ai-companies>) – Lista as 100 empresas mais importantes de IA no mundo.

Consultei também algumas fontes de notícias, para entender o que além dos gráficos poderia conseguir. Listo-os no final deste trabalho.

Com os dados, desenvolvi scripts em Python para agrupar as vagas e empresas por localização, exibindo um mapa de calor interativo (com funções de *zoom* e *pan*). Um gráfico de barras para entender melhor a concentração por país e dois gráficos para avaliar os salários das vagas, um por especialização e outro histograma de distribuição, como segue.

Com esta análise dos gráficos abaixo, desenvolvi o roteiro informativo (*storytelling*), a seguir.

GRÁFICO 21 - Mapa de calor interativo de empresas de IA

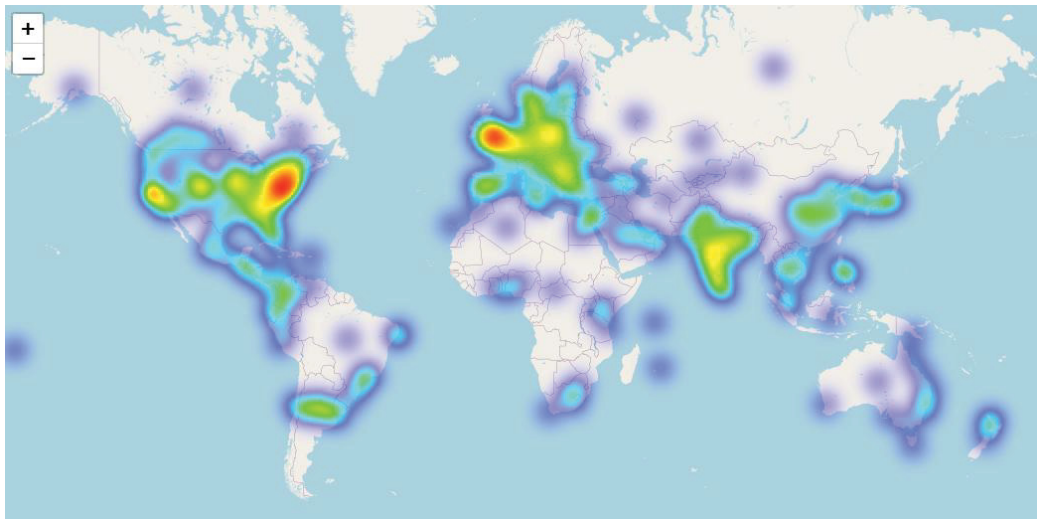


GRÁFICO 22 - Distribuição de empresas de IA por localidade

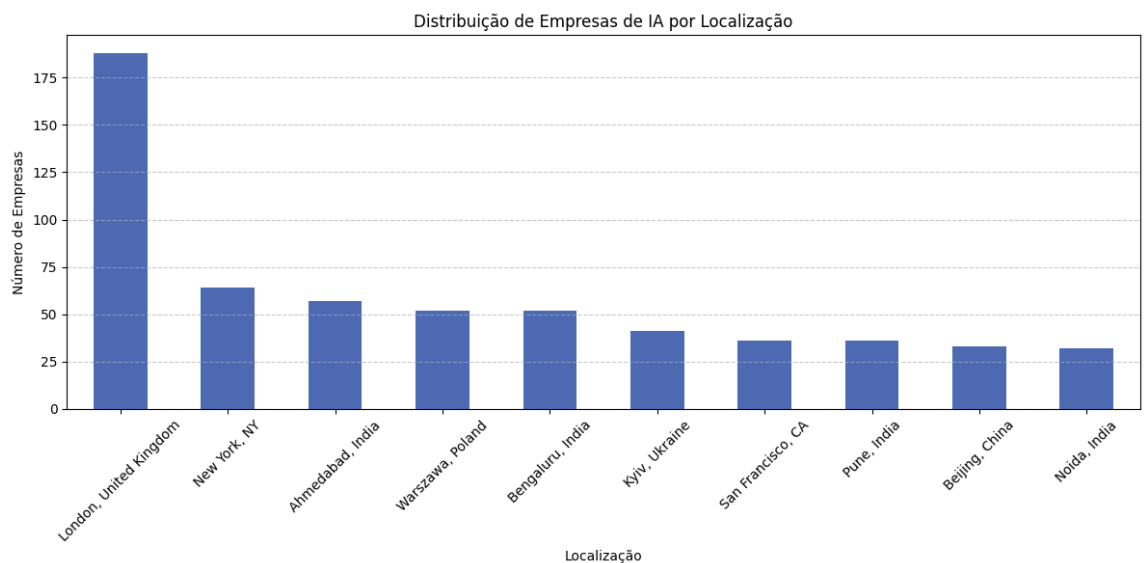


GRÁFICO 23 - Distribuição de salário por senioridade

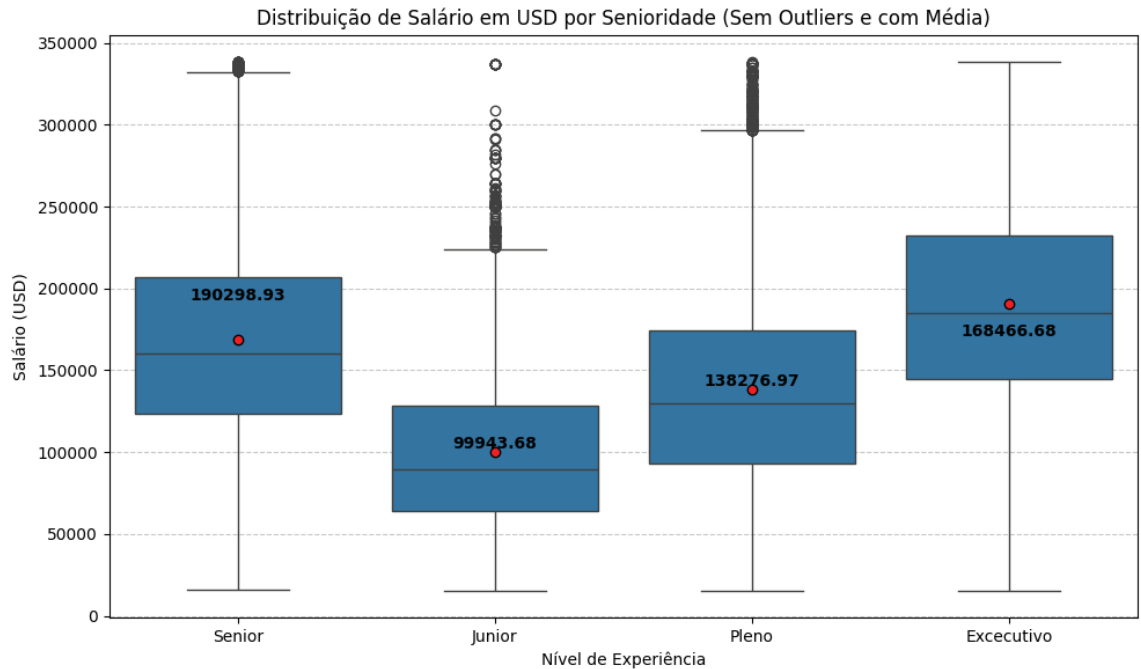
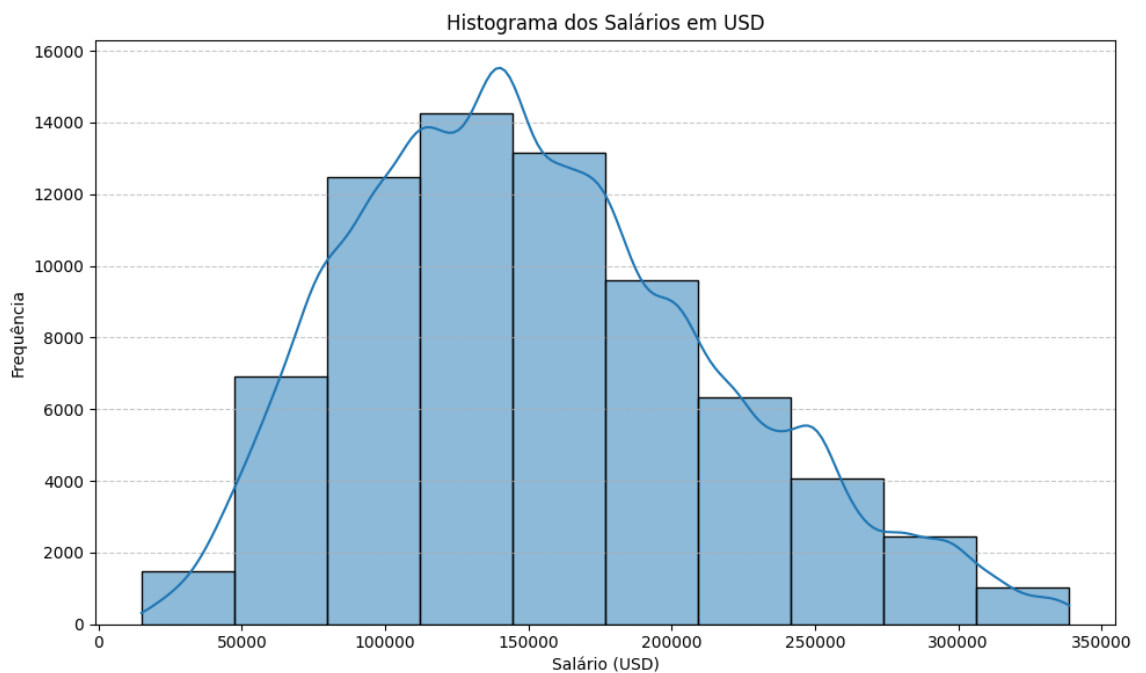


GRÁFICO 24 - Histograma de salários em dólar



2. CONTEXTO E OBJETIVO

A análise busca compreender quais países estão na fronteira de Inteligência Artificial. Seja pela pesquisa e desenvolvimento de novos produtos, seja pela utilização dos modelos existentes em produtos e serviços – que se multiplicam aos montes.

3. PRINCIPAIS DESCOBERTAS

- **Estados Unidos e Inglaterra lideram maior concentração**

Como era de se esperar (por causa da OpenAI, Microsoft, X, Meta e outros), os Estados Unidos tem o maior contingente de empresas de IA ou empresas que já empregam IA em seus produtos. Nova York e São Francisco em lados opostos do país são os pontos focais, mas o país tem outras localidades que compõem a área mais densa. Com concentração significativa, está também o Reino Unido, especialmente Londres. A londrina Synthesia é um bom exemplo, foi recém declarada unicórnio e recebe fortes investimentos da gigante do setor Nvidia;

- **Distribuição por localidade**

Interessante notar que no agrupamento por localidade, os locais dos Estados Unidos estão mais pulverizados do que em Londres, com a maior quantidade de empresas. Embora Nova York seja polo indiscutível;

- **Distribuição Salarial**

Os salários são altíssimos, mesmo para cargos que exigem menos experiência. Se convertido para reais, passam de 40 mil reais por mês. Outro detalhe que vale menção é que existem muitos outliers. A tendência é que estes valores comecem a se assentar a medida que a barreira tecnológica é apaziguada.

4. AÇÕES POSSÍVEIS

- **Políticas de incentivos em outros países**

Compreender a extensão das empresas atuais e incentivar programas de pesquisa em outros países. A tecnologia empregada em IA parece estar contribuindo para disparidades entre países dominantes e dominados;

- **Aprofundar a pesquisa na China**

Apesar da dificuldade em se obter informação do que acontece internamente na China, me parece que modelos recentes muito bons DeepSeek, WAN, QwQ surpreenderam o mundo. E não estão mapeados nos dados que encontrei disponíveis;

- **Identificação das próximas inovações**

Entendendo a concentração em alguns países, é interessante acompanhar o mercado. Índia é um exemplo. Podemos ver inovações significativas vindas de lá;

- **Tendências salariais**

Para as vagas recentes no país, a média salarial encontrada pode ser uma baliza para competitividade global. Dado que o trabalho remoto é alternativa viável.

5. CONCLUSÃO

A concentração da inteligência artificial em poucos países descortina a necessidade de uma abordagem estratégica para equilibrar o avanço tecnológico global. Incentivar a pesquisa, monitorar tendências e buscar inovação em mercados emergentes ou não, são ações essenciais para evitar que a IA se torne um monopólio. Afora isso, a transparência na distribuição salarial pode ser um fator determinante para a competitividade. O futuro da IA dependerá da capacidade de diferentes regiões se adaptarem, investirem e inovarem de forma autônoma, reduzindo desigualdades e promovendo novos caminhos.

APÊNDICE 15 – TÓPICOS EM INTELIGÊNCIA ARTIFICIAL

A – ENUNCIADO

1) Algoritmo Genético

Problema do Caixeiro Viajante

A Solução poderá ser apresentada em: Python (preferencialmente), ou em R, ou em Matlab, ou em C ou em Java.

Considere o seguinte problema de otimização (a escolha do número de 100 cidades foi feita simplesmente para tornar o problema intratável. A solução ótima para este problema não é conhecida).

Suponha que um caixeiro deva partir de sua cidade, visitar clientes em outras 99 cidades diferentes, e então retornar à sua cidade. Dadas as coordenadas das 100 cidades, descubra o percurso de menor distância que passe uma única vez por todas as cidades e retorne à cidade de origem.

Para tornar a coisa mais interessante, as coordenadas das cidades deverão ser sorteadas (aleatórias), considere que cada cidade possui um par de coordenadas (x e y) em um espaço limitado de 100 por 100 pixels.

O relatório deverá conter no mínimo a primeira melhor solução (obtida aleatoriamente na geração da população inicial) e a melhor solução obtida após um número mínimo de 1000 gerações. Gere as imagens em 2d dos pontos (cidades) e do caminho.

Sugestão:

- (1) considere o cromossomo formado pelas cidades, onde a cidade de início (escolhida aleatoriamente) deverá estar na posição 0 e 100 e a ordem das cidades visitadas nas posições de 1 a 99 deverão ser definidas pelo algoritmo genético.
- (2) A função de avaliação deverá minimizar a distância euclidiana entre as cidades (os pontos).
- (3) Utilize no mínimo uma população com 100 indivíduos;
- (4) Utilize no mínimo 1% de novos indivíduos obtidos pelo operador de mutação;
- (5) Utilize no mínimo de 90% de novos indivíduos obtidos pelo método de cruzamento (crossover-ox);
- (6) Preserve sempre a melhor solução de uma geração para outra.

Importante: A solução deverá implementar os operadores de “cruzamento” e “mutação”.

2) Compare a representação de dois modelos vetoriais

Pegue um texto relativamente pequeno, o objetivo será visualizar a representação vetorial, que poderá ser um vetor por palavra ou por sentença. Seja qual for a situação, considere a quantidade de palavras ou sentenças onde tenha no mínimo duas similares e no mínimo 6 textos, que deverão produzir no mínimo 6 vetores. Também limite o número máximo, para que a visualização fique clara e objetiva.

O trabalho consiste em pegar os fragmentos de texto e codificá-las na forma vetorial. Após obter os vetores, imprima-os em figuras (*plot*) que demonstrem a projeção desses vetores usando a PCA.

O PDF deverá conter o código-fonte e as imagens obtidas.

B – RESOLUÇÃO

1. ALGORITMO GENÉTICO

Para resolver o problema do caixeiro viajante com algoritmo genético, iniciamos com a importação das bibliotecas necessárias.

FIGURA 147 - Importando bibliotecas

```
import numpy as np
import matplotlib.pyplot as plt
import random
from itertools import permutations
```

Criamos parâmetros iniciais.

FIGURA 148 - Parâmetros problema caixeiro viajante

```
# Parâmetros Iniciais
NUM_CIDADES = 100
POPULACAO_SIZE = 100
GERACOES = 1000
MUTACAO_RATE = 0.01
CROSSOVER_RATE = 0.90

cidades = np.random.rand(NUM_CIDADES, 2) * 100
```

Criamos funções de apoio para a base do algoritmo genético

FIGURA 149 – Funções de apoio para algoritmo genético

```

# Calcula a distância euclidiana (comprimento do vetor) total percorrida no caminho
def calcular_distancia(caminho):
    return sum(np.linalg.norm(cidades[caminho[i]] - cidades[caminho[i+1]]) for i in range(len(caminho)-1))

def criar_populacao():
    return [np.random.permutation(range(1, NUM_CIDADES)).tolist() for _ in range(POPULACAO_SIZE)]

# Avalia a população com base na distância total do percurso
def avaliar_populacao(populacao):
    return sorted(populacao, key=lambda ind: calcular_distancia([0] + ind + [0]))

# Seleciona um indivíduo por torneio
def selecao_torneio(populacao):
    torneio = random.sample(populacao, 5)
    return min(torneio, key=lambda ind: calcular_distancia([0] + ind + [0]))

# Aplica crossover genético nos pais para gerar dois filhos
def crossover(pai1, pai2):
    tamanho = len(pai1)
    ponto1, ponto2 = sorted(random.sample(range(tamanho), 2))
    meio = pai1[ponto1:ponto2]
    filho1 = meio + [gene for gene in pai2 if gene not in meio]
    filho2 = meio + [gene for gene in pai1 if gene not in meio]
    return filho1, filho2

# Aplica mutação trocando duas cidades de posição
def mutacao(individuo):
    if random.random() < MUTACAO_RATE:
        i, j = random.sample(range(len(individuo)), 2)
        individuo[i], individuo[j] = individuo[j], individuo[i]
    return individuo

# Imprime o percurso das cidades
def mostra_solucao(melhor_solucao, titulo):
    caminho = [0] + melhor_solucao + [0]
    plt.figure(figsize=(8, 8))
    plt.scatter(cidades[:, 0], cidades[:, 1], c='blue')
    plt.plot(cidades[caminho, 0], cidades[caminho, 1], 'r-', lw=1.5)
    plt.title(titulo)
    plt.show()

```

Seguimos para execução do algoritmo.

FIGURA 150 - Execução do algoritmo genético

```

def main():

    populacao = criar_populacao()
    melhor_solucao = avaliar_populacao(populacao)[0]
    mostra_solucao(melhor_solucao, "Primeira melhor solução (população inicial)")

    for _ in range(GERACOES):
        nova_populacao = []
        while len(nova_populacao) < POPULACAO_SIZE * CROSSOVER_RATE:
            pai1, pai2 = selecao_torneio(populacao), selecao_torneio(populacao)
            filho1, filho2 = crossover(pai1, pai2)
            nova_populacao.extend([filho1, filho2])

        while len(nova_populacao) < POPULACAO_SIZE:
            nova_populacao.append(mutacao(selecao_torneio(populacao)))

        populacao = avaliar_populacao(nova_populacao)[:POPULACAO_SIZE]
        melhor_solucao = populacao[0]

    mostra_solucao(melhor_solucao, "Melhor solução após 1000 gerações")

main()

```

E o resultado após execução.

FIGURA 151 - Primeira melhor solução da população inicial

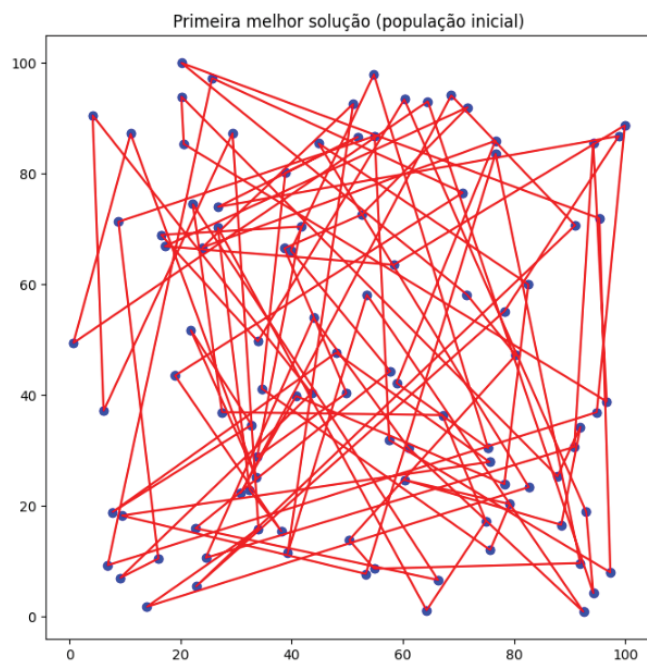
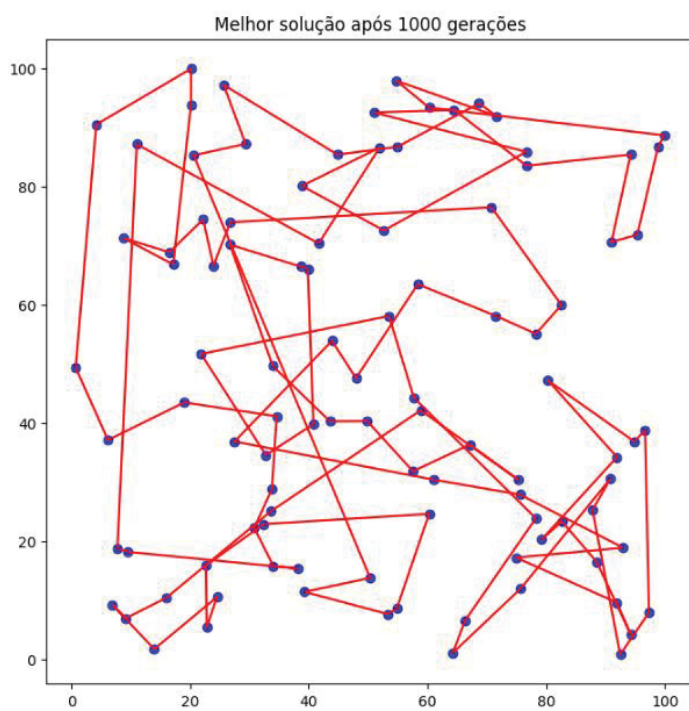


FIGURA 152 - Melhor solução após 1000 gerações



Para o a comparação de dois modelos vetoriais, iniciamos importando as bibliotecas necessárias

FIGURA 153 - Importando bibliotecas

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.feature_extraction.text import CountVectorizer
from sentence_transformers import SentenceTransformer
```

Criamos textos bases com similaridades, para o teste.

FIGURA 154 - Textos base de comparação vetorial

```
textos = [
    "O cachorro correu pelo parque atrás da bola",
    "O cão brincou no parque com sua bola favorita",
    "O gato dormia tranquilamente no sofá da sala",
    "O felino estava descansando calmamente no sofá",
    "Um pássaro voou rapidamente para a árvore mais alta",
    "O drone pousou cuidadosamente no topo da árvore"
]
```

Aplicamos o modelo por frequência de palavras, com visualização usando PCA.

FIGURA 155- Frequência de palavras usando PCA

```
vectorizer = CountVectorizer()
X_palavras = vectorizer.fit_transform(textos).toarray()

# PCA para visualização (reduzindo a 2 dimensões)
pca_palavras = PCA(n_components=2)
X_pca_palavras = pca_palavras.fit_transform(X_palavras)

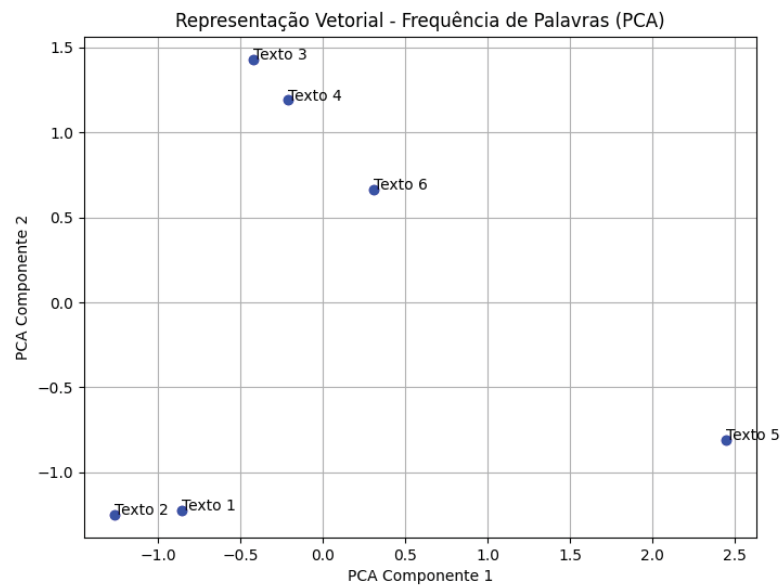
# Plot da representação baseada em frequência de palavras
plt.figure(figsize=(8, 6))
plt.scatter(X_pca_palavras[:,0], X_pca_palavras[:,1], color='blue')

for i, texto in enumerate(textos):
    plt.annotate(f'Texto {i+1}', (X_pca_palavras[i,0], X_pca_palavras[i,1]))

plt.title('Representação Vetorial - Frequência de Palavras (PCA)')
plt.xlabel('PCA Componente 1')
plt.ylabel('PCA Componente 2')
plt.grid(True)
plt.show()
```

E o resultado da comparação.

GRÁFICO 25 - Representação vetorial por frequência de palavras (PCA)



Calculando a matriz de similaridade usando cosseno.

FIGURA 156 - Calculo matriz de similaridade por cosseno

```
# Similaridade por cosseno
sim_matrix = cosine_similarity(X_pca_palavras)

print("Matriz de similaridade por cosseno entre textos:\n")
print(sim_matrix)
```

Matriz de similaridade por cosseno entre textos:

```
[[ 1.          0.98364566 -0.6237046 -0.70578672 -0.28672528 -0.98482673]
 [ 0.98364566  1.          -0.47271605 -0.56664659 -0.45458803 -0.93746339]
 [-0.6237046  -0.47271605  1.          0.99394952 -0.57000857  0.74989067]
 [-0.70578672 -0.56664659  0.99394952  1.          -0.47631276  0.81801797]
 [-0.28672528 -0.45458803 -0.57000857 -0.47631276  1.          0.11612067]
 [-0.98482673 -0.93746339  0.74989067  0.81801797  0.11612067  1.          ]]
```

Desta forma, avaliamos as similaridades.

FIGURA 157 - Avaliando similaridades

```
pares_similares = np.argsort(-sim_matrix, axis=1)[: , 1]

print("\nTextos mais similares entre si:")
for idx, par in enumerate(pares_similares):
    print(f"Texto {idx+1} é mais próximo do Texto {par+1}")
```

```
Textos mais similares entre si:
Texto 1 é mais próximo do Texto 2
Texto 2 é mais próximo do Texto 1
Texto 3 é mais próximo do Texto 4
Texto 4 é mais próximo do Texto 3
Texto 5 é mais próximo do Texto 6
Texto 6 é mais próximo do Texto 4
```

Agora criamos o modelo por *embedding* e visualização usando PCA

FIGURA 158 - Criando o modelo por *embeddings* e gerando gráfico

```
# Usando modelo de sentence transformer https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
modelo_embed = SentenceTransformer('all-MiniLM-L6-v2')
X_embeddings = modelo_embed.encode(textos)

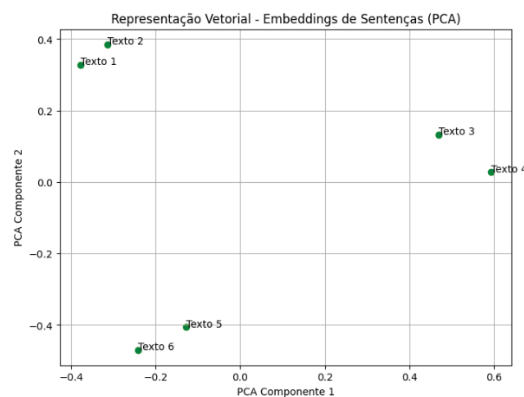
# PCA para visualização (reduzindo a 2 dimensões)
pca_embed = PCA(n_components=2)
X_pca_embed = pca_embed.fit_transform(X_embeddings)

# Plot da representação baseada em embeddings de sentenças
plt.figure(figsize=(8, 6))
plt.scatter(X_pca_embed[:,0], X_pca_embed[:,1], color='green')

for i, texto in enumerate(textos):
    plt.annotate(f'Texto {i+1}', (X_pca_embed[i,0], X_pca_embed[i,1]))

plt.title('Representação Vetorial - Embeddings de Sentenças (PCA)')
plt.xlabel('PCA Componente 1')
plt.ylabel('PCA Componente 2')
plt.grid(True)
plt.show()
```

E o resultado gráfico.

GRÁFICO 26 - Representação vetorial por *embedding*

Calculamos a matriz de similaridade usando cosseno.

FIGURA 159 - Matriz de similaridade por cosseno

```
# Similaridade por cosseno
sim_matrix = cosine_similarity(X_pca_embed)

print("Matriz de similaridade por cosseno entre textos:\n")
print(sim_matrix)
```

Matriz de similaridade por cosseno entre textos:

```
[[ 1.0000001  0.9852994 -0.5465784 -0.7219985 -0.39772758 -0.23916142]
 [ 0.9852994  1.0000001 -0.39548331 -0.59318376 -0.548624  -0.40152457]
 [-0.5465784 -0.39548331  0.99999994  0.97402674 -0.5509355  -0.6823858 ]
 [-0.7219985 -0.59318376  0.97402674  0.99999994 -0.347657  -0.49914145]
 [-0.39772758 -0.548624  -0.5509355  -0.347657  1.          0.9859986 ]
 [-0.23916142 -0.40152457 -0.6823858  -0.49914145  0.9859986  1.          ]]
```

E finalmente avaliamos resultados similares.

FIGURA 160 - Avaliação dos resultados similares

```
pares_similares = np.argsort(-sim_matrix, axis=1)[: , 1]

print("\nTextos mais similares entre si:")
for idx, par in enumerate(pares_similares):
    print(f"Texto {idx+1} é mais próximo do Texto {par+1}")
```

Textos mais similares entre si:
 Texto 1 é mais próximo do Texto 2
 Texto 2 é mais próximo do Texto 1
 Texto 3 é mais próximo do Texto 4
 Texto 4 é mais próximo do Texto 3
 Texto 5 é mais próximo do Texto 6
 Texto 6 é mais próximo do Texto 5

APENDICE 16 – CÓDIGO RAG

```

import os
import json
import asyncio
import requests
import streamlit as st
import xml.etree.ElementTree as ET
import chromadb
import ollama
import re
from urllib.parse import urlparse
from crawl4ai import AsyncWebCrawler, CrawlerRunConfig, CacheMode
from crawl4ai.content_filter_strategy import PruningContentFilter
from crawl4ai.markdown_generation_strategy import DefaultMarkdownGenerator
from sentence_transformers import SentenceTransformer

# Configuração global
PASTA_JSON = "./data"
CHROMA_DB_PATH = "./chroma_db"
SITEMAP_URL = "https://www.lance.com.br/sitemap/news/today.xml"
MODEL_NAME = "deepseek-r1:8b"
MODEL_EMBEDDING = "all-MiniLM-L6-v2"
MAX_CRAWLERS = 5

# Conectar ao banco ChromaDB
db = chromadb.PersistentClient(path=CHROMA_DB_PATH)
collection = db.get_or_create_collection(name="noticias")
modelo = SentenceTransformer(MODEL_EMBEDDING)

# Função para extrair slug da URL
def extrair_slug(url):
    path = urlparse(url).path
    slug = path.rstrip('/').split('/')[-1]
    return os.path.splitext(slug)[0]

# Função para extrair links do sitemap
def extrair_links_sitemap(url_sitemap):
    resposta = requests.get(url_sitemap)
    if resposta.status_code != 200:
        st.error(f"Erro ao acessar {url_sitemap}")
        return []

    root = ET.fromstring(resposta.text)
    urls = [elem.text for elem in root.findall("./{http://www.sitemaps.org/schemas/sitemap/0.9}loc")]
    return urls

# Função para rodar um crawler em paralelo
async def rodar_crawler_para_link(link):
    prune_filter = PruningContentFilter(threshold=0.45,
    threshold_type="dynamic", min_word_threshold=5)
    md_generator = DefaultMarkdownGenerator(content_filter=prune_filter)
    config = CrawlerRunConfig(markdown_generator=md_generator,
    cache_mode=CacheMode.BYPASS, excluded_tags=["nav", "footer", "header"],
    exclude_external_links=True)

    async with AsyncWebCrawler() as crawler:
        result = await crawler.arun(url=link, config=config)
        if result.success:

```



```

mkResult = result.markdown_v2.fit_markdown
match = re.search(r'^# (.+)', mkResult, re.MULTILINE)
title = match.group(1) if match else "Sem título"

json_data = {"title": title, "link": link, "content": mkResult}
filename = os.path.join(PASTA_JSON,
f"{extrair_slug(link)}.json")
with open(filename, "w", encoding="utf-8") as f:
    json.dump(json_data, f, ensure_ascii=False, indent=4)

    return f" Notícia salva: {title}"
else:
    return f" Erro ao extrair {link}: {result.error_message}"

async def rodar_crawler():

    os.makedirs(PASTA_JSON, exist_ok=True)
    links = extrair_links_sitemap(SITEMAP_URL)

    # Executar crawlers em paralelo com limite máximo de processos simultâneos
    resultados = []
    for i in range(0, len(links), MAX_CRAWLERS):
        batch = links[i:i+MAX_CRAWLERS] # Divide os links em lotes de
MAX_CRAWLERS
        resultados += await asyncio.gather(*(rodar_crawler_para_link(link)
for link in batch))
    for resultado in resultados:
        st.write(resultado)

# Função para carregar arquivos JSON e inserir no ChromaDB
def processar_arquivos_json():
    documentos = []
    for arquivo in os.listdir(PASTA_JSON):
        if arquivo.endswith(".json"):
            caminho_arquivo = os.path.join(PASTA_JSON, arquivo)
            with open(caminho_arquivo, "r", encoding="utf-8") as f:
                dados = json.load(f)

            slug = os.path.splitext(arquivo)[0]
            documentos.append({"id": slug, "title": dados.get("title",
slug), "link": dados.get("link", ""), "content": dados.get("content", "")})

    for doc in documentos:
        embedding = modelo.encode(doc["content"]).tolist()
        collection.add(ids=[doc["id"]], embeddings=[embedding],
metadatas=[{"title": doc["title"], "link": doc["link"], "content":
doc["content"]}])

    st.success(f"{len(documentos)} documentos inseridos no ChromaDB!")

def apagar_arquivos_json():
    """Apaga todos os arquivos JSON dentro do diretório de dados."""
    if not os.path.exists(PASTA_JSON):
        st.warning("A pasta de dados não existe.")
        return

    arquivos = [f for f in os.listdir(PASTA_JSON) if f.endswith(".json")]

    if not arquivos:
        st.warning("Nenhum arquivo JSON encontrado para deletar.")
        return

```

```

for arquivo in arquivos:
    caminho_arquivo = os.path.join(PASTA_JSON, arquivo)
    try:
        os.remove(caminho_arquivo)
    except Exception as e:
        st.error(f"Erro ao deletar {arquivo}: {e}")

    st.success(f"    {len(arquivos)} arquivos JSON foram deletados com sucesso!")

# Função para buscar contexto no ChromaDB
def buscar_contexto(consulta):
    consulta_embedding = modelo.encode(consulta).tolist()
    resultados = collection.query(query_embeddings=[consulta_embedding],
n_results=3)

    contexto = ""
    for r in resultados["metadatas"][0]:
        contexto += f"\n    {r['title']}\n    {r['content'][:500]}...\n
{r['link']}\n\n"

    return contexto.strip() if contexto else "Nenhuma informação relevante encontrada."

# Função para consultar o LLM com contexto
def perguntar_ao_llm(consulta):
    contexto = buscar_contexto(consulta)
    prompt = f"""Você é um assistente especializado em notícias.
Use as informações abaixo para responder à pergunta do usuário, incluindo
os links das fontes:
{contexto}
Pergunta: {consulta}
Resposta: """

    resposta = ollama.chat(model=MODEL_NAME, messages=[{"role": "user",
"content": prompt}])
    return resposta['message']['content']

# Interface do Streamlit
st.set_page_config(page_title="RAG AI", page_icon="🤖")

# Criar abas no Streamlit
aba = st.sidebar.radio("Navegação", [" Crawler", " Vetorização", " Apagar Local", " Chat"])

if aba == " Crawler":
    st.title(" Crawler - Extração de Notícias")
    st.write("Clique no botão abaixo para iniciar a coleta de notícias do site!")

    if st.button("Iniciar Crawler"):
        st.warning("O processo pode levar alguns minutos...")
        asyncio.run(rodar_crawler())

elif aba == " Vetorização":
    st.title(" Vetorização - Inserir Notícias no ChromaDB")
    st.write("Processar os arquivos JSON e armazenar as notícias no banco vetorial.")

    if st.button("Processar Arquivos"):

```

```

        processar_arquivos_json()

elif aba == " Apagar Local":
    st.title(" Apagar Arquivos JSON")
    st.write("Clique no botão abaixo para apagar todos os arquivos JSON extraídos.")

    if st.button("Apagar Arquivos JSON"):
        apagar_arquivos_json()

elif aba == " Chat":
    st.title(" Sports AI")
    st.write("Digite uma pergunta e obtenha respostas baseadas em notícias armazenadas no banco vetorial.")

    consulta_usuario = st.text_input("Pergunte:", placeholder="Quem joga hoje?")

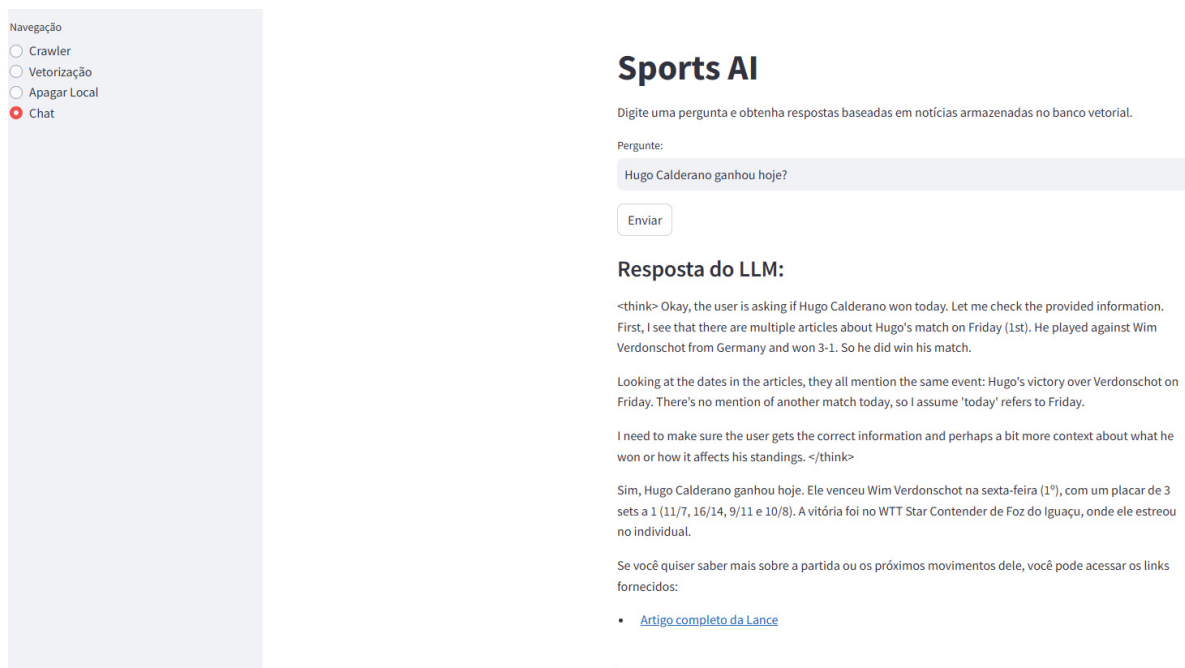
    if st.button("Enviar"):
        if consulta_usuario:
            with st.spinner(" Buscando resposta..."):
                resposta = perguntar_ao_llm(consulta_usuario)

            st.subheader(" Resposta do LLM:")
            st.write(resposta)
        else:
            st.warning("Por favor, digite uma pergunta antes de continuar.")

# Rodapé
st.markdown("---")
st.markdown(" Desenvolvido com **Ollama + Crawl4AI + ChromaDB + Streamlit + DeepSeek-R1:8B**")

```

FIGURA 161 – Chat do RAG AI



FONTE: Luiz Henrique dos Santos Cruz (2025).

ANEXO 1 – CÓDIGO STORYTELLING

```
# -*- coding: utf-8 -*-
"""Storytelling.ipynb

https://colab.research.google.com/drive/1L2aygHxUpXvEFweNeUw8PaqDaC9RdzQO

# INSTALAÇÃO E IMPORTAÇÃO DE BIBLIOTECAS
"""
!pip install pandas geopy folium matplotlib kaggle pycountry

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import difflib
import pycountry
import os
import folium
from folium.plugins import HeatMap
import time
import requests

# Definir credenciais do Kaggle (necessário configurar 'kaggle.json'
localmente)
!mkdir -p ~/.kaggle
!mkdir -p ./data/1/
!mkdir -p ./data/2/
!mkdir -p ./data/3/
!echo '{"username":"cruzzz","key":"chave_secreta"}' > ~/.kaggle/kaggle.json
!chmod 600 ~/.kaggle/kaggle.json

# Baixar datasets e descompactar arquivos
!kaggle datasets download -d vineethakkinapalli/ai-companies
!mv ai-companies.zip 1_ai_companies.zip && unzip -o 1_ai_companies.zip -d
./data/1/
!kaggle datasets download -d cedricaubin/ai-ml-salaries
!mv ai-ml-salaries.zip 2.zip && unzip -o 2.zip -d ./data/2/
!kaggle datasets download -d raniritu/ai-companies
!mv ai-companies.zip 3_ai_companies.zip && unzip -o 3_ai_companies.zip -d
./data/3/

# Carregar datasets
df_100 = pd.read_csv("./data/1/AI_Companies.csv")
df_salary = pd.read_csv("./data/2/salaries.csv")
df_companies = pd.read_csv("./data/3/Ai_companies.csv")

"""# DOWNLOAD E CARREGAMENTO DE DADOS DO KAGGLE"""
# Definir credenciais do Kaggle (necessário configurar 'kaggle.json'
localmente)
!mkdir -p ~/.kaggle
!mkdir -p ./data/1/
!mkdir -p ./data/2/
!mkdir -p ./data/3/
!echo '{"username":"cruzzz","key":"<key>"}' > ~/.kaggle/kaggle.json
!chmod 600 ~/.kaggle/kaggle.json
```

```

# Baixar datasets e descompactar arquivos
!kaggle datasets download -d vineethakkinapalli/ai-companies
!mv ai-companies.zip 1_ai_companies.zip && unzip -o 1_ai_companies.zip -d
./data/1/
!kaggle datasets download -d cedricaubin/ai-ml-salaries
!mv ai-ml-salaries.zip 2.zip && unzip -o 2.zip -d ./data/2/
!kaggle datasets download -d raniritu/ai-companies
!mv ai-companies.zip 3_ai_companies.zip && unzip -o 3_ai_companies.zip -d
./data/3/

# Carregar datasets
df_100 = pd.read_csv("./data/1/AI_Companies.csv")
df_salary = pd.read_csv("./data/2/salaries.csv")
df_companies = pd.read_csv("./data/3/Ai_companies.csv")

""""# PRÉ-PROCESSAMENTO DOS DADOS

##Limpeza e padronização
""""

# Ajustando nomes das colunas
df_100.rename(columns={"Company Name": "Company_Name", "Headquarters":
"Location"}, inplace=True)
df_companies.rename(columns={"Company Name": "Company_Name", "Headquarters":
"Location"}, inplace=True)
df_salary.rename(columns={"company_location": "Location"}, inplace=True)

df_companies.drop(columns=["Unnamed: 7"], inplace=True, errors='ignore')
df_100.drop(columns=["Unnamed: 7"], inplace=True, errors='ignore')

# Removendo valores nulos nas colunas importantes
df_100.dropna(subset=["Location"], inplace=True)

df_salary.dropna(subset=["Location"], inplace=True)

# Extraíndo países
def extract_country(location):
    if isinstance(location, str) and "," in location:
        return location.split(",")[-1].strip()
    return location

df_salary["Country"] =
df_salary["Location"].apply(extract_country).fillna("Unknown")

# Converter códigos de país para nomes completos
def get_country_name(country_code):
    try:
        return pycountry.countries.lookup(country_code).name
    except LookupError:
        return None

df_salary["Country"] =
df_salary["Country"].apply(get_country_name).fillna("Unknown")

# Padronizando nomes dos países
df_salary.dropna(subset=["Country"], inplace=True)
df_salary["Country"] = df_salary["Country"].str.title()

# Removendo caracteres não numéricos das colunas de salário
df_salary["salary_in_usd"] = pd.to_numeric(df_salary["salary_in_usd"],
errors='coerce').fillna(0)

```

```

# Removendo valores NaN após conversão
df_salary.dropna(subset=["salary_in_usd"], inplace=True)

df_100.head(10)
df_companies.head(10)
df_salary.head(10)

"""## Geolocalizando os endereços e agrupando"""

API_KEY = "<key>"

# Função para converter códigos de países em nomes completos
def get_full_country_name(code):
    try:
        return pycountry.countries.lookup(code).name
    except LookupError:
        return code # Retorna o código original se não for um país válido

# Função para obter coordenadas via API geocode.maps.co
def get_coordinates(location):
    location = get_full_country_name(location) # Converter siglas para nomes completos
    base_url =
    f"https://geocode.maps.co/search?q={location}&api_key={API_KEY}"

    try:
        response = requests.get(base_url, timeout=10)
        if response.status_code != 200:
            print(f"Erro {response.status_code} ao buscar {location}")
            return None, None

        data = response.json()
        time.sleep(1.1)

        if isinstance(data, list) and len(data) > 0:
            lat, lon = float(data[0]["lat"]), float(data[0]["lon"]) #
            Pegando o primeiro item
            return lat, lon

        else:
            print(f"Nenhum dado encontrado para {location}")
            return None, None

    except Exception as e:
        print(f"Erro ao buscar coordenadas para {location}: {e}")
        return None, None

    # Evita bloqueios da API
    return None, None

# Concatenar todas as localizações únicas
all_locations = pd.concat([df_salary["Location"], df_companies["Location"],
df_100["Location"]]).dropna().unique()

# Criar um DataFrame para armazenar as coordenadas
locations_df = pd.DataFrame(all_locations, columns=["Location"])
locations_df["Latitude"], locations_df["Longitude"] =
zip(*locations_df["Location"].apply(get_coordinates))

# Remover valores sem coordenadas

```

```

locations_df.dropna(subset=["Latitude", "Longitude"], inplace=True)

"""# ANÁLISES E VISUALIZAÇÕES

## 1) Considerando empresas e vagas como itens, geramos um mapa para entender
em quais regiões elas se concentram.
"""

# Criar um mapa base
mapa = folium.Map(location=[20, 0], zoom_start=2)

# Criar um heatmap com as localizações geocodificadas
heat_data = list(zip(locations_df["Latitude"], locations_df["Longitude"]))
HeatMap(heat_data, min_opacity=0.4, max_zoom=10, radius=15).add_to(mapa)

# Exibir o mapa no Colab
mapa

"""## 2) Usando somente as empresas, sem as vagas, contabilize para facilitar
o entendimento"""

# Concatenando e limpando os dados
df = pd.concat([df_100, df_companies],
ignore_index=True).dropna(subset=["Location"])

# Agrupar por país e selecionar os 10 maiores
location_counts = df["Location"].value_counts().nlargest(10)

# Gráfico otimizado
plt.figure(figsize=(12, 6))
location_counts.plot(kind="bar", color="royalblue")
plt.xlabel("Localização")
plt.ylabel("Número de Empresas")
plt.title("Distribuição de Empresas de IA por Localização")
plt.xticks(rotation=45)
plt.grid(axis="y", linestyle="--", alpha=0.7)

plt.tight_layout()
plt.show()

"""## 3) Distribuição salarial por senioridade"""

df_salaries = df_salary.copy()

# Mapeando siglas para explicações em português
exp_level_mapping = {
    "EN": "Junior",
    "MI": "Pleno",
    "SE": "Senior",
    "EX": "Excecutivo"
}

# Substituir siglas pelas explicações no DataFrame
df_sem_outliers["experience_level_pt"] =
df_sem_outliers["experience_level"].map(exp_level_mapping)

# Plot com explicações na legenda
plt.figure(figsize=(10, 6))

```

```

ax = sns.boxplot(x="experience_level_pt", y="salary_in_usd",
data=df_sem_outliers, showmeans=True,
                meanprops={"marker":"o", "markerfacecolor":"red",
"markeredgecolor":"black"})

# Calculando médias com as novas legendas
medias_pt = df_sem_outliers.groupby("experience_level_pt")["salary_in_usd"].mean().reset_index()

# Adicionando valores das médias ao gráfico
for idx, row in medias_pt.iterrows():
    ax.text(idx, row["salary_in_usd"], f'{row["salary_in_usd"]:.2f}',
            color='black', ha="center", va="bottom", fontweight='bold')

plt.xlabel("Nível de Experiência")
plt.ylabel("Salário (USD)")
plt.title("Distribuição de Salário em USD por Senioridade (Sem Outliers e com Média)")
plt.grid(axis="y", linestyle="--", alpha=0.7)
plt.tight_layout()

plt.show()

# Histograma dos salários em USD
plt.figure(figsize=(10, 6))
sns.histplot(df_sem_outliers["salary_in_usd"], bins=10, kde=True)

plt.xlabel("Salário (USD)")
plt.ylabel("Frequência")
plt.title("Histograma dos Salários em USD")
plt.grid(axis="y", linestyle="--", alpha=0.7)
plt.tight_layout()

plt.show()

```