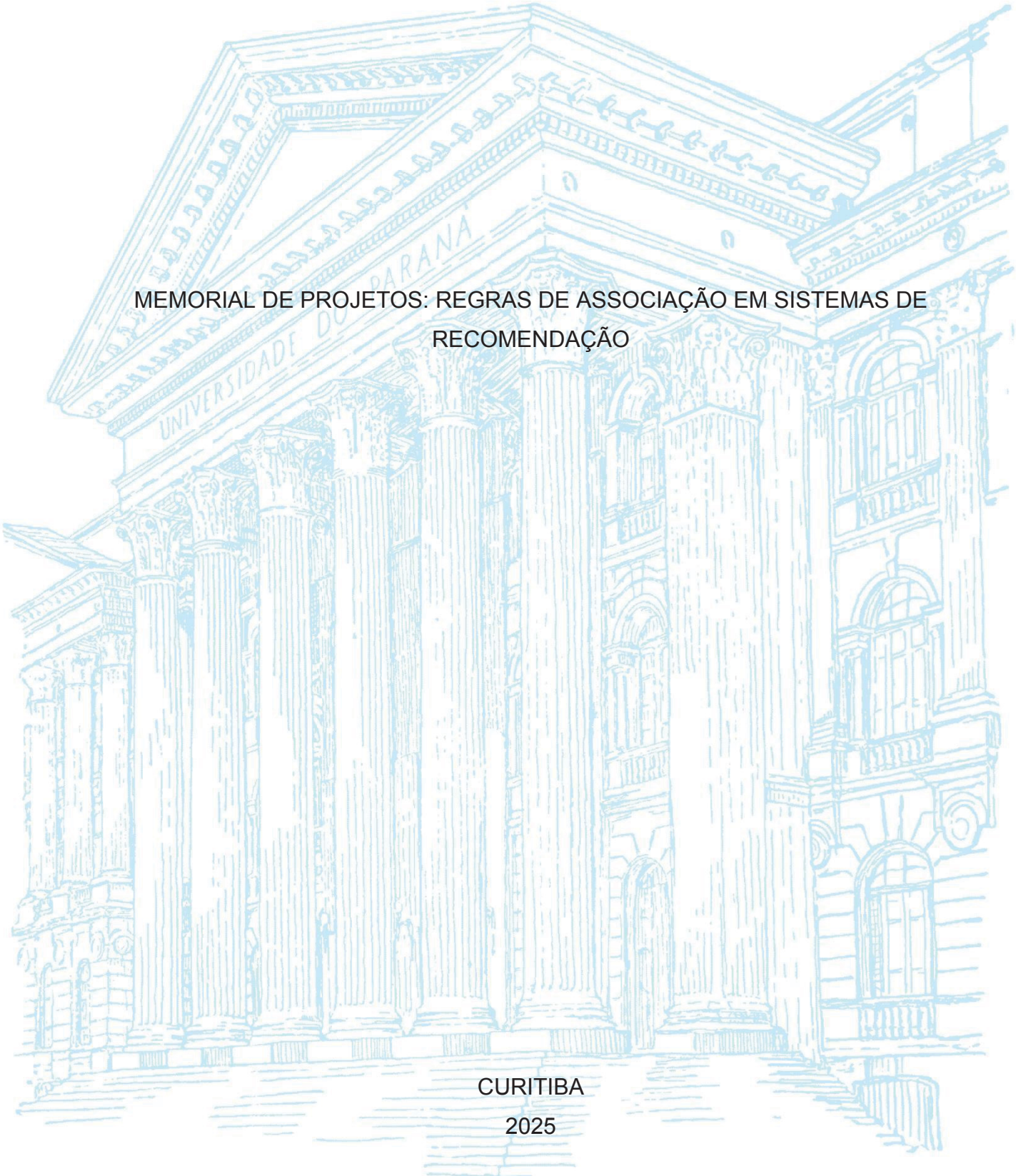
UNIVERSIDADE FEDERAL DO PARANÁ

RENAN ROGER FERREIRA DA SILVA



MEMORIAL DE PROJETOS: REGRAS DE ASSOCIAÇÃO EM SISTEMAS DE RECOMENDAÇÃO

CURITIBA

2025

RENAN ROGER FERREIRA DA SILVA

MEMORIAL DE PROJETOS: REGRAS DE ASSOCIAÇÃO EM SISTEMAS DE RECOMENDAÇÃO

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.

Orientador: Profa. Dra. Rafaela Mantovani Fontana

CURITIBA

2025

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação Inteligência Artificial Aplicada da Universidade Federal do Paraná foram convocados para realizar a arguição da Monografia de Especialização de **RENAN ROGER FERREIRA DA SILVA**, intitulada: **MEMORIAL DE PROJETOS: REGRAS DE ASSOCIAÇÃO EM SISTEMAS DE RECOMENDAÇÃO**, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua aprovação no rito de defesa.

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

Curitiba, 08 de Agosto de 2025.

RAFAELA MANTOVANI FONTANA
Presidente da Banca Examinadora

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

RESUMO

No campo do aprendizado de máquina, o aprendizado não supervisionado destaca-se por sua capacidade de identificar padrões e estruturas ocultas em conjuntos de dados sem a necessidade de rótulos pré-definidos. Em vez de prever resultados conhecidos, seu objetivo é explorar os dados e extrair insights valiosos. Nesse contexto, os algoritmos de Regras de Associação, como o Apriori, surgem como ferramentas de alto impacto comercial, possibilitando a análise de grandes volumes de dados transacionais para descobrir relações frequentes entre itens, como produtos frequentemente adquiridos em conjunto. Esses algoritmos são amplamente utilizados como motores de sistemas de recomendação, contribuindo para o aumento das vendas no varejo. O presente parecer técnico avalia a aplicação das Regras de Associação com foco na geração de recomendações interpretáveis, utilizando métricas como suporte, confiança e *lift* para seleção das regras mais relevantes, ressaltando a importância da escolha adequada dos parâmetros para garantir eficiência e evitar a explosão combinatória. Conclui-se que a abordagem é eficaz na identificação de padrões de consumo e na construção de sistemas de recomendação automatizados.

Palavras-chave: Aprendizado de máquina; aprendizado não supervisionado; regras de associação;

ABSTRACT

In the field of machine learning, unsupervised learning stands out for its ability to identify hidden patterns and structures in datasets without the need for predefined labels. Rather than predicting known outcomes, its goal is to explore data and extract valuable insights. In this context, Association Rule algorithms, such as Apriori, emerge as tools with high commercial impact, enabling the analysis of large volumes of transactional data to discover frequent relationships between items, such as products commonly purchased together. These algorithms are widely used as engines behind recommendation systems, contributing to increased retail sales. This technical report evaluates the application of Association Rules with a focus on generating interpretable recommendations, using metrics such as support, confidence, and lift to select the most relevant rules. It emphasizes the importance of appropriately selecting parameters to ensure efficiency and avoid combinatorial explosion. It is concluded that this approach is effective in identifying consumption patterns and building automated recommendation systems.

Keywords: Machine learning; unsupervised learning; association rules.

SUMÁRIO

1 PARECER TÉCNICO

Este parecer técnico tem como objetivo apresentar e analisar a aplicabilidade de algoritmos de Regras de Associação (Agrawal, Imielinski e Swami, 1993) no contexto de sistemas de recomendação, com foco na geração automática de sugestões personalizadas a partir de padrões de venda dos itens. Tais algoritmos se destacam por sua capacidade de identificar relações frequentes entre itens em bases de dados transacionais, possibilitando a construção de um sistema de recomendação interpretável e de fácil implementação.

Regras de associação consistem em encontrar conjuntos de itens que ocorram simultaneamente e de forma frequente em um banco de dados (Goldschmidt e Passos, 2005). O exemplo clássico para regras de associação é o da cesta de super mercado e foi introduzido por Agrawal, Imielinski e Swami (1993), onde um cliente compra uma série de itens e cada compra é referida como uma transação. O objetivo do algoritmo é determinar a associação entre grupos de itens comprados pelos clientes, esse exemplo pode ser transportado para os dias atuais levando em consideração as compras em e-commerces.

Regras de associação são geradas na forma de X → Y onde X e Y são conjuntos de itens, então dado a regra de associação {ovo, leite} → {iogurte} entende-se que quem compra ovos e leite tende a comprar iogurte (Aggarwal, 2015).

São criadas regras associativas para todos os itens contidos nas transações, mas nem toda regra encontrada é útil. Precisa-se de métricas quantitativas para filtrar o ruído e encontrar as associações verdadeiramente significativas, as três métricas essenciais são suporte, confiança e *lift*:

1. **Suporte**: É a métrica utilizada para verificar se uma associação é frequente. É a proporção de transações que contém um conjunto de itens, a fórmula é dada por:

$$Suporte(x) = (nº\ de\ transações\ contendo\ x) / (total\ de\ transações)$$

2. **Confiança:** Mede a força da regra. É a probabilidade condicional de encontrar o consequente (Y) em uma transação, dado que ela já contém o antecedente (X).

$$Confiança(X -> Y) = Suporte(X \cup Y) / Suporte(x)$$

3. **Elevação (*Lift*)**: Mede o quão provável é que Y seja comprado quando X é comprado, em comparação com a probabilidade de Y ser comprado de forma geral. É a métrica mais importante para avaliar se a associação é realmente interessante e não apenas resultado da alta popularidade individual dos itens.

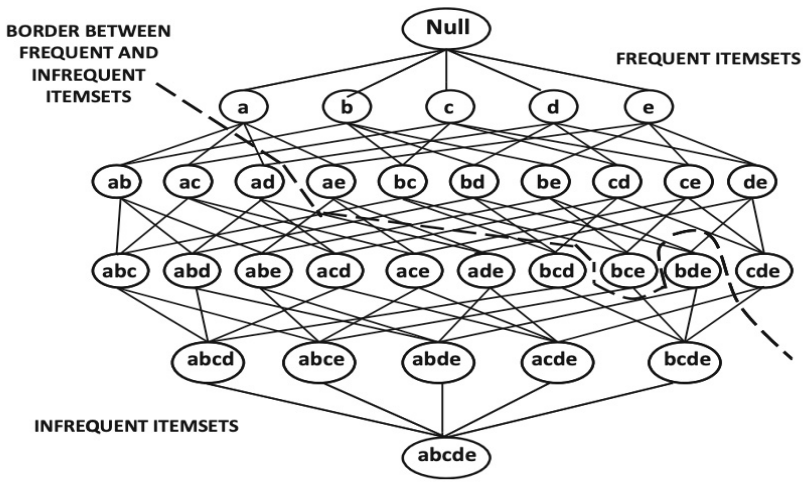
$$Lift(x \ -> \ y) \ = \ Confiança(X \ -> \ Y)/Suporte(y)$$

No caso do retorno do *lift* o resultado é interpretado da seguinte forma:

1. Lift > 1: Indica uma associação positiva. A presença de X *aumenta* a probabilidade da presença de Y.
2. Lift = 1: Indica que X e Y são independentes. Não há associação entre eles.
3. Lift < 1: Indica uma associação negativa. A presença de X *diminui* a probabilidade da presença de Y (ex: clientes que compram um produto X raramente compram produto Y).

Como pode ser visualizado na FIGURA 1, é exibido um grafo onde demostra como os conjuntos de itens(*itemsets)* são criados a partir de 5 itens iniciais (a, b, c, d, e), é possível observar que acada nível do grafo são criados novos *itemsets* que são resultante das combinações dos *itemsets* do mesmo nível até não existir mais combinações possíveis. A linha tracejada indica a separação de quais são os conjuntos frequentes e infrequentes, essa separação é feita através das métricas especificadas anteriormente.

FIGURA 1 – Grafo demonstrando as possíveis associações que podem ser criadas.



FONTE : Aggarwal (2015, p.97).

Existem vários algoritmos que implementam a teoria acima, o mais conhecido e visto ao longo do curso foi o algoritmo Apriori. Sua implementação, em Python, pode ser encontrada dentro da biblioteca `mlxtend` (Raschka, 2025), e é com ele que pode-se criar um motor de um sistema recomendação de produtos.

O algoritmo Apriori funciona da seguinte maneira: deve ser definido os valores mínimos de corte das métricas (suporte, confiança e *lift*), para esse exemplo foi utilizado apenas o valor de suporte mínimo de 0,3, como podemos verificar na TABELA 1, é feito o cálculo de suporte para os *itemset* iniciais.

TABELA 1 – 1-itemset

itemset	**suporte**
leite	0,2
café	0,3
cerveja	0,2
Pão	0,5
Manteiga	0,5
Arroz	0,2
Feijão	0,2

FONTE : Goldschmidt (2005 p.107).

Após remover os itemsets que não satisfaça o valor mínimo de suporte estabelecido, são combinados os *1-itemsets* restante para gerar os *2-itemsets* conforme podemos observar na TABELA 2.

TABELA 2 – 2-itemset

itemset	**suporte**
Café, Pão	0,3
Café, Manteiga	0,3
Pão, Manteiga	0,4

FONTE : Goldschmidt (2005 p.107).

O algoritmo encerra a criação dos *itemset* quando não existem mais combinações possíveis para serem criadas que respeitem os valores mínimos do suporte (GOLDSCHMIDT e PASSOS, 2005), observamos na TABELA 3 o último *itemset* gerado, a partir dele não é possível gerar mais nenhuma combinação e o algoritmo se encerra.

TABELA 3 – 3-itemset

itemset	suporte
Café, Pão, Manteiga	0,3

FONTE : Goldschmidt (2005 p.107).

Um ponto para atenção é na tarefa de encontrar todos os *k-itemsets*, pois ela exige um maior custo computacional dependendo da quantidade de transações e da granularidade dos itens contidos nessas transações, pois os conjuntos podem acabar sendo criados de forma exponencial caso os parâmetros de suporte e confiança não consigam filtrar muitos itens.

Uma forma de aplicar esse algoritmo em um sistema de recomendação seria:

1. Obter todas as transações (vendas) de e-commerce em um período histórico, ex: últimos 6 meses.
2. Gerar as regras de associação definidas por um valor de suporte e confiança mínimo;
3. Sempre que um produto for adicionado em um carrinho, filtrar esse produto ou conjunto de produtos dentro das regras de associação geradas anteriormente, ordenar pelas regras com o melhor *lift*, com base nos resultados, identificam-se os produtos mais adequados a serem recomendados ao cliente.

A aplicação de Regras de Associação demonstrou-se eficaz na identificação de padrões relevantes de consumo (Prokeinová, Paluchová, 2014), com potencial de aplicação prática em estratégias de vendas cruzadas. Recomenda-se a adaptação progressiva dos valores de suporte e confiança até a obtenção de regras de associação consistentes, utilizando-se a métrica de *lift* para a seleção das regras mais relevantes.

REFERÊNCIAS

AGGARWAL, Charu C. Data Mining: The Textbook. EUA: Springer, 2015.

AGRAWAL, Rakesh; IMIELIŃSKI, Tomasz; SWAMI, Arun. "Mining Association Rules between Sets of Items in Large Databases", San José, CA., 1993. Disponível em https://dl.acm.org/doi/10.1145/170036.170072 Acesso em: 23 jun. 2025.

GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel. Data mining: um guia prático. Rio de Janeiro, Elsevier, 2005 - 4º Reimpressão.

PROKEINOVÁ, R. Benda; PALUCHOVÁ, Johana. Identification of the Patterns Behavior Consumptions by Using Chosen Tools of Data Mining – Association Rules. Agris on-line Papers in Economics and Informatics, 2014. Disponível em https://online.agris.cz/files/2014/agris_on-line_2014_3_benda-prokeinova_paluchova.pdf. Acesso em: 23 jun. 2025.

RASCHKA, Sebastian. Mlxtend. Disponível em https://rasbt.github.io/mlxtend/ Acesso em: 23 jun. 2025.

APÊNDICE 1 – INTRODUÇÃO À INTELIGÊNCIA ARTIFICIAL

A – ENUNCIADO

1 ChatGPT

a) **(6,25 pontos)** Pergunte ao ChatGPT o que é Inteligência Artificial e cole aqui o resultado.
b) **(6,25 pontos)** Dada essa resposta do ChatGPT, classifique usando as 4 abordagens vistas em sala. Explique o porquê.
c) **(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.
d) **(6,25 pontos)** Entendendo o que é o ChatGPT, classifique o próprio ChatGPT usando as 4 abordagens vistas em sala. Explique o porquê.

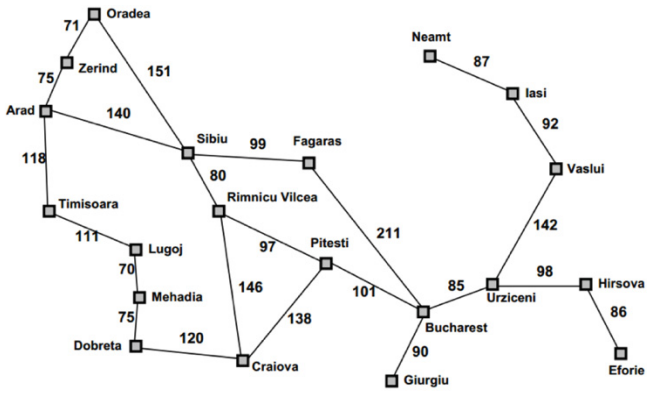
2 Busca Heurística

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.

a) **(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.

2

3 Lógica

Verificar se o argumento lógico é válido.

Se as uvas caem, então a raposa as come
Se a raposa as come, então estão maduras
As uvas estão verdes ou caem

Logo

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

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

Dicas:

1. Transformar as afirmações para lógica:

p: as uvas caem
q: a raposa come as uvas
r: as uvas estão maduras

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

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

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

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

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

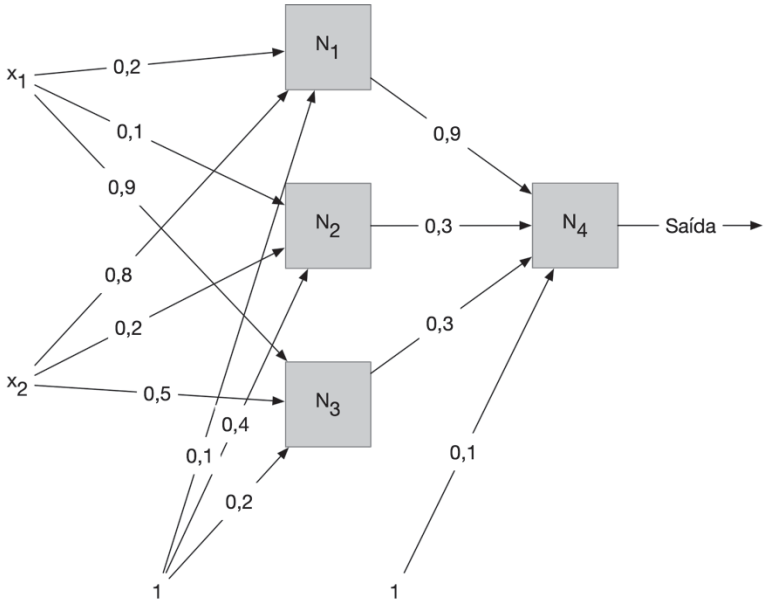
Conjunção: $\alpha, \beta \vdash \alpha \wedge \beta$

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

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

4 Redes Neurais Artificiais

Seja a RNA da figura abaixo.



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

Dada a entrada x_1=-3, x_2=1, dê:

a) **(6,25 pontos)** Valor de saída do neurônio N1
b) **(6,25 pontos)** Valor de saída do neurônio N2
c) **(6,25 pontos)** Valor de saída do neurônio N3
d) **(6,25 pontos)** Valor de saída da rede como um todo

B – RESOLUÇÃO

1 ChatGPT

a) Existem dois tipos principais de Inteligência Artificial: a IA fraca (ou estreita) e a IA forte (ou geral).

IA Fraca (IA Estreita): Este é o tipo mais comum de IA na atualidade. Refere-se a sistemas que são projetados e treinados para realizar tarefas específicas sem possuir uma compreensão ampla ou consciência. Exemplos incluem assistentes virtuais, reconhecimento de voz, carros autônomos e sistemas de recomendação.

2. IA Forte (IA Geral): Este é um conceito mais avançado, envolvendo sistemas que possuem uma inteligência comparável à humana, sendo capazes de entender, aprender e realizar qualquer tarefa intelectual que um ser humano possa fazer. A IA forte ainda está em um estágio teórico e não foi plenamente alcançada.

As técnicas utilizadas na implementação da IA incluem machine learning (aprendizado de máquina), deep learning (aprendizado profundo), processamento de linguagem natural (PLN), visão computacional e algoritmos de otimização, entre outros. A IA tem aplicações em diversas áreas, como saúde, finanças, educação, manufatura, automação, jogos e muitas outras.

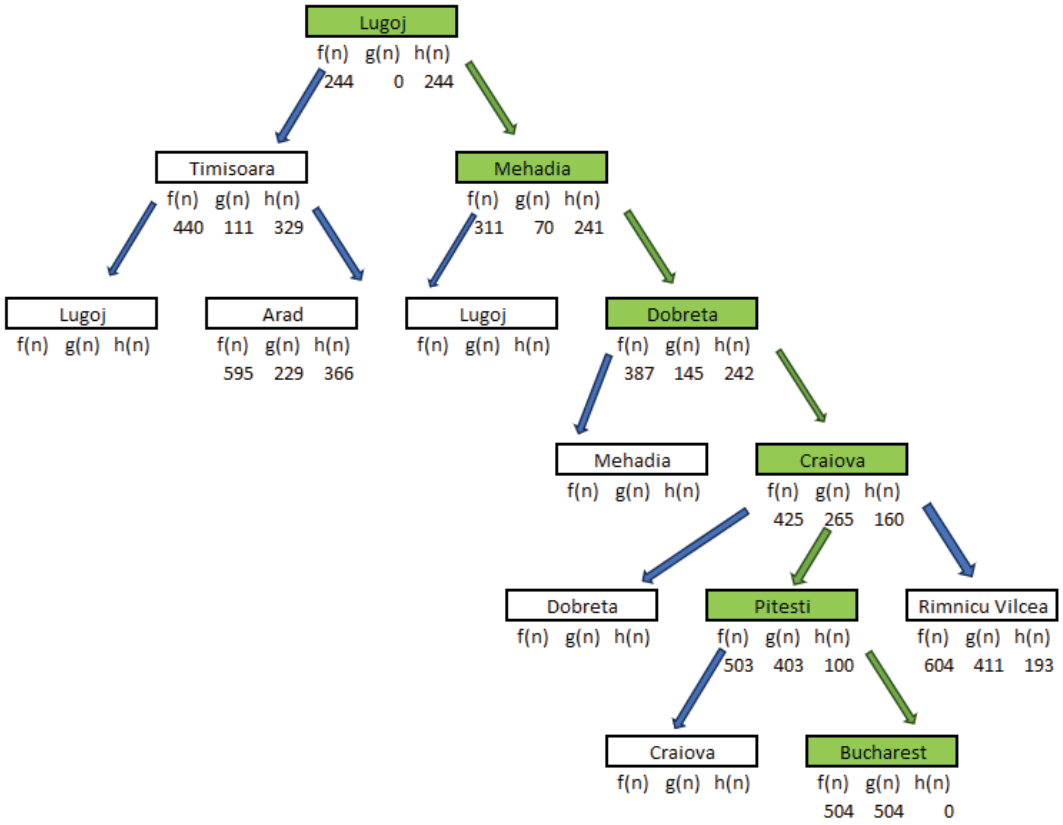
b) Agir como humano, Pois ele conseguiu interpretar a linguagem natural e retornar uma resposta satisfatória, demonstrando uma boa represetação do conhecimento.

c) Um dos principais recursos do chat GPT é a capacidade de entender e gerar respostas como um humano em tempo real. Isso é possivel, pois o mesmo foi treinado utilizando como base de conhecimento textos da internet, isso incluiu dados obtidos de livros, Wikipedia, artigos e outros textos na internet.

O ChatGPT é baseado na arquitetura GPT (Generative Pretrained Transformer), através dos Transformers Models é possível dar um peso maior de influência para algumas palavras em uma frase e isso faz com que o GPT consiga entregar respostas mais coerentes e contextualmente precisas. Essas respostas são geradas na probabilidade capaz de adivinhar qual deve ser a próxima palavra em uma frase e para chegar a um estágio em que pudesse fazer isso, o modelo passou por fases de testes supervisionados e ajustes. https://www.sciencefocus.com/future-technology/gpt-3 https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf https://www.datacamp.com/blog/a-chat-with-chatgpt-on-the-method-behind-the-bot

d) Agir racionalmente, o mesmo consegue criar inferências baseadas em seus contextos, tem as habilidades necessárias para passar no teste de turing, consegue representar conhecimento e mesmo quando ele não tem certeza, retorna o melhor resultado possível.

2 Busca Heurística



3

3 Lógica

R1: $p \rightarrow q$

R2: $q \rightarrow r$

R3: $\neg r \vee p$

R4: $p \rightarrow r$ Silogismo Hipotético R1 e R2

R5: $r \rightarrow p$ Equivalência Implicação R3

R6: $q \rightarrow p$ Silogismo Hipotético R2 e R5

R7: $p \rightarrow q \ \wedge q \rightarrow p$ Conjunção R6 e R1

R7: $q \leftrightarrow p$ Bicondicional R7

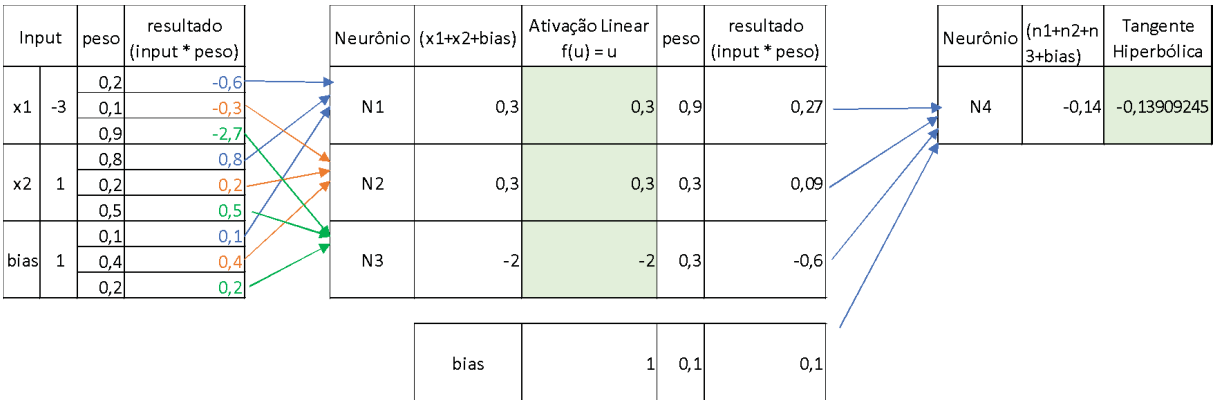
4 Redes Neurais Artificiais

a) 0,3

b) 0,3

c) -2

d) -0,13909245



Input		peso	resultado (input * peso)
x1	-3	0,2	-0,6
		0,1	-0,3
		0,9	-2,7
x2	1	0,8	0,8
		0,2	0,2
		0,5	0,5
bias	1	0,1	0,1
		0,4	0,4
		0,2	0,2

Neurônio	(x1+x2+bias)	Ativação Linear f(u) = u	peso	resultado (input * peso)
N1	0,3	0,3	0,9	0,27
N2	0,3	0,3	0,3	0,09
N3	-2	-2	0,3	-0,6
bias		1	0,1	0,1

Neurônio	(n1+n2+n3+bias)	Tangente Hiperbólica
N4	-0,14	-0,13909245

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:

a. Carregue a base de dados media_precos_carros_brasil.csv
b. Verifique se há valores faltantes nos dados. Caso haja, escolha uma tratativa para resolver o problema de valores faltantes
c. Verifique se há dados duplicados nos dados
d. Crie duas categorias, para separar colunas numéricas e categóricas. Imprima o resumo de informações das variáveis numéricas e categóricas (estatística descritiva dos dados)
e. Imprima a contagem de valores por modelo (model) e marca do carro (brand)
f. 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:

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

a. 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
b. Crie partições contendo 75% dos dados para treino e 25% para teste
c. 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

d. Grave os valores preditos em variáveis criadas
e. Realize a análise de importância das variáveis para estimar a variável target, **para cada modelo treinado**
f. Dê uma breve explicação (máximo de quatro linhas) sobre os resultados encontrados na análise de importância de variáveis
g. Escolha o melhor modelo com base nas métricas de avaliação MSE, MAE e R^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

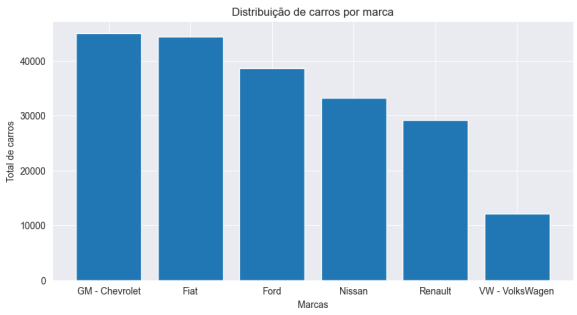
1 Análise Exploratória dos dados

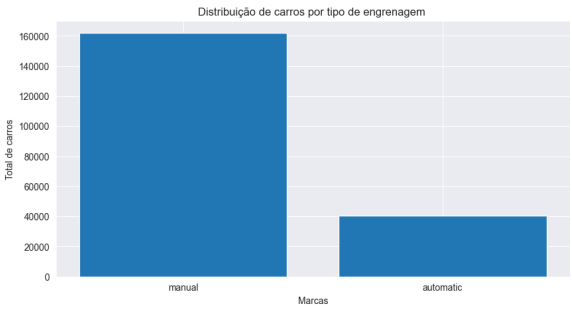
Arquivo original com 267.542 registros, sendo 65.245 vazios (NAs) e 2 duplicados. Há 6 marcas que produziram 2112 diferentes modelos de carros, com variações quanto ao tipo de combustível (3 cat.), quanto ao tipo de câmbio (2 cat.) e tamanho do motor (29 cat.). A média do ano dos modelos é 2011 +- 6 anos (dp). Quanto ao preço, a média é de aproximadamente 52.756, com grande dp (+- 51.628) sendo o mais barato 6647 e o mais caro 979358.

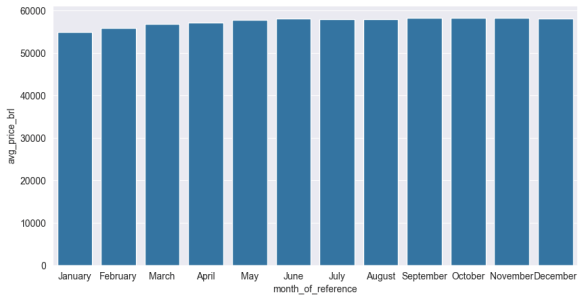
 # Recortes do Código

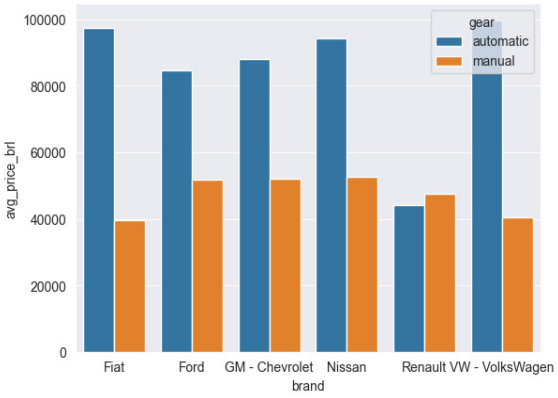
```
dados = pd.read_csv('precos_carros_brasil.csv')
dados = dados.dropna()
dados = dados.drop_duplicates()
numericas_cols = [col for col in dados.columns if
dados[col].dtype != 'object']
categoricas_cols = [col for col in dados.columns if
dados[col].dtype == 'object']
dados['model'].value_counts()
dados['brand'].value_counts()
```

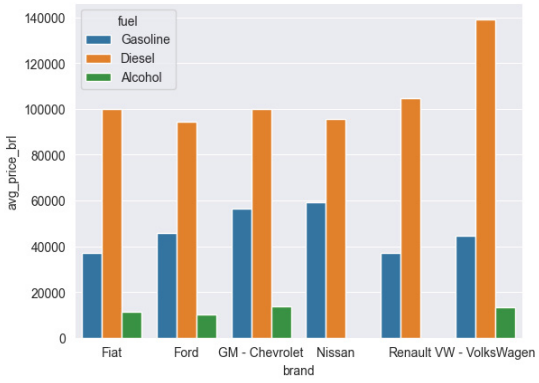
2 Visualização dos dados











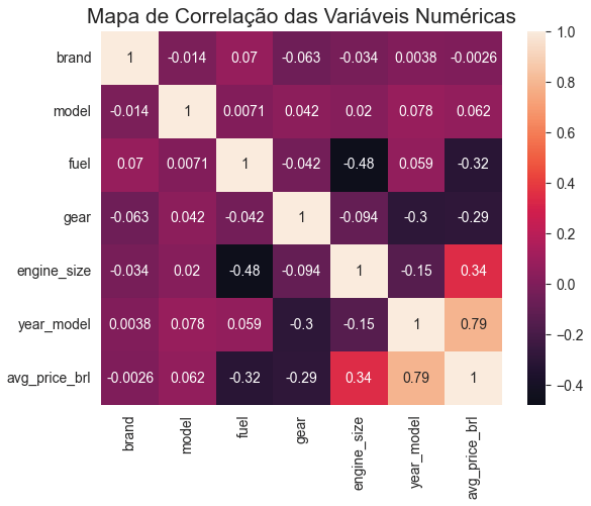
Podemos verificar que existem uma constante em todas as marcas onde os maiores valores são dos carros a diesel, seguidos por gasolina e álcool. Os carros a diesel da Volkswagem são bem mais caros que os das outras marcas. A Renaut e Nissan não possuem carros a álcool. A GM e Nissan possuem os carros mais caros a gasolina.

 # Recortes do Código

```
plt.figure(figsize=(10,5))
plt.bar(dados['brand'].unique(),
dados['brand'].value_counts())
plt.title('Distribuição de carros por marca')
plt.ylabel('Total de carros');
```

```
plt.figure(figsize=(10,5))
plt.bar(dados['gear'].unique(), dados['gear'].value_counts())
plt.title('Distribuição de carros por tipo de engrenagem ')
plt.ylabel('Total de carros');
df = dados.groupby(['brand','gear'])['avg_price_brl'].mean().round(2)
df = df.reset_index(name='avg_price_brl')
sns.barplot(x='brand', y='avg_price_brl', hue='gear', data=df, hue_order=['automatic','manual'])
df = dados.groupby(['brand','fuel'])['avg_price_brl'].mean().round(2)
df = df.reset_index(name='avg_price_brl')
sns.barplot(x='brand', y='avg_price_brl', hue='fuel', data=df, hue_order=['Gasoline','Diesel','Alcohol'])
```

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



importance	
engine_size	0.479710
fuel	0.180833
year_model	0.165360
gear	0.104630
brand	0.050185
model	0.01928

Engine_size tem uma importância elevada nos dois modelos, embora tenha correlação baixa com preço (0,34). A importância das variáveis diferencia-se bastante nos dois modelos Year_model tem correlação alta com preço (0,79) mas maior importância somente no Random Forest, já no gxboost a importância é baixa Model, Fuel, Gear e Brand tem pouca importância no Random Forest Brand e Model tem pouca importância no xgboost.

RandomForest
MSE 54248180.20965601
MAE 4222.612390664574
R2 0.9796442228984908

gxboost
MSE 69506536.98775701
MAE 4917.806998993991
R2 0.9739187643059638

Levando em consideração o MSE, o modelo RF é melhor pois tem o valor menor (54249235.2) comparado ao GX (69506536.9). Olhando o MAE o RF (4221.5) e o gx com (4917.8), saindo o RF como melhor modelo pois sua média de erro é menor. Já no R2 ambos ficaram muito próximos o RF (0.979) e o gx (0.973) como ambos são maiores que 0,7 indica que são modelos fortes.

 # Recortes do Código

```
sns.heatmap(df.corr("spearman"), annot = True)
```

```
plt.title("Mapa de Correlação das Variáveis Numéricas\n",
fontsize = 15)
plt.show()
X = df.drop(['avg_price_brl'],axis = 1) # variáveis
independentes
Y = df['avg_price_brl'] # TARGET
percTest = 0.25 # 25% para teste
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size = percTest, random_state = 33)
model_rf = RandomForestRegressor()
model_rf.fit(X_train, Y_train)
model_xgboost = XGBRegressor()
model_xgboost.fit(X_train, Y_train)
valores_preditos_rf = model_rf.predict(X_test)
valores_preditos_xgboost = model_xgboost.predict(X_test)
model_rf.feature_importances_
feature_importances =
pd.DataFrame(model_rf.feature_importances_, index =
X_train.columns,
columns=['importance']).sort_values('importance', ascending =
False)
feature_importances
model_xgboost.feature_importances_
feature_importances =
pd.DataFrame(model_xgboost.feature_importances_, index =
X_train.columns,
columns=['importance']).sort_values('importance', ascending =
False)
feature_importances
```

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:

$$\textbf{Volume} = b0 + b1 * dap^2 * Ht$$

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: Volume = b0 + b1 * dap^2 * Ht

      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 $\underline{y}$ é a média dos valores y_i observados. Quanto mais perto de 1 melhor é o modelo;

- Erro padrão da estimativa: S_{yx}

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

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

- Syx%

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

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)

```
# Efetua repartição
indices <- createDataPartition(Satellite$classes, p=0.8, list=FALSE)
treino <- Satellite[indices, ]
teste <- Satellite[-indices, ]

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

```
predict.rf <- predict(rf, teste)
predict.svm <- predict(svm, teste)
predict.rna <- predict(rna, teste)

# matrizes confusão
confusionMatrix(predict.rf, teste$classes)
confusionMatrix(predict.svm, teste$classes)
confusionMatrix(predict.rna, teste$classes)
```

Comparando o resultado das matrizes confusão dos três modelos: RandomForest, SVM e RNA.

1. **Acurácia**:
   RandomForest: 92.13%
   SVM: 91.12%
   RNA: 46.5%
2. **Sensibilidade** (por classe):
   RandomForest: Varia entre 0.616 e 0.990
   SVM: Varia entre 0.592 e 0.990
   RNA: Varia entre 0.000 e 0.996
3. **Especificidade** (por classe):
   RandomForest: Varia entre 0.736 e 0.996
   SVM: Varia entre 0.737 e 0.994
   RNA: Varia entre 0.568 e 1.000
4. **Valor Preditivo Positivo** (por classe):
   RandomForest: Varia entre 0.725 e 0.978
   SVM: Varia entre 0.500 e 0.961
   RNA: Varia entre 0.414 e 0.977
5. **Valor Preditivo Negativo** (por classe):
   RandomForest: Varia entre 0.891 e 0.998
   SVM: Varia entre 0.888 e 0.999
   RNA: Varia entre 0.788 e 0.998
6. **Acurácia Balanceada** (por classe):
   RandomForest: Média de 0.9862.
   SVM: Média de 0.9915.
   RNA: Média de 0.5031.

Conclusões: Com base nos resultados, podemos observar que RandomForest e SVM têm desempenho significativamente melhor em comparação com RNA, com RandomForest ligeiramente à frente em termos de acurácia e sensibilidade.

2 Estimativa de Volumes de Árvores

```
# Particiona
indices <- createDataPartition(data$VOL, p=0.8, list=FALSE)
treino <- data[indices, ]
teste <- data[-indices, ]

# treina os modelos
rf <- train(VOL~., data=treino, method="rf")
svm <- train(VOL~., data=treino, method="svmRadial")
rna <- train(VOL~., data=treino, method="nnet", linout=1)
modelo_alometrico <- nls(VOL ~ b0 + b1*DAP*DAP*HT, treino, start=list(b0=0.5,
b1=0.5))

# predição
predict.rf <- predict(rf, teste)
predict.svm <- predict(svm, teste)
predict.rna <- predict(rna, teste)
predict.ma <- predict(modelo_alometrico, teste)

# Funções
r_quadrado <- function(y_real, y_pred) {
  return(1 - sum((y_real - y_pred)^2) / sum((y_real - mean(y_real))^2))
}
erro_padrao_estimativa <- function(y_real, y_pred) {
  n <- length(y_real)
  residual <- y_real - y_pred
  return (sqrt(sum(residual^2) / (n - 2)))
}
erro_padrao_estimativa_porcentagem <- function(y_real, y_pred){
  media <- mean(y_real)
  return(erro_padrao_estimativa(y_real, y_pred) / media) * 100
}
```

A comparação das métricas R quadrado e Erro Padrão da Estimativa (Syx) para os modelos RandomForest (rf), SVM (svm), RNA (rna) e um modelo alométrico (modelo alométrico):

1. **R quadrado** (Quanto mais perto de 1, melhor é o modelo):

RandomForest: 0.731
SVM: 0.760
RNA: -1.172
Modelo alométrico: 0.825

2. **Erro padrão da estimativa** (Syx) (Quanto mais perto de 0, melhor é o modelo):
   RandomForest: 0.162
   SVM: 0.153
   RNA: 0.460
   Modelo alométrico: 0.130

3. **Syx%** (Quanto mais perto de 0, melhor é o modelo):
   RandomForest: 0.123
   SVM: 0.116
   RNA: 0.348
   Modelo alométrico: 0.099

Conclusões: Com base nessas métricas, o modelo alométrico tem o melhor desempenho geral, seguido pelos modelos RandomForest e SVM, e por último, RNA. Isso é indicado pelos valores mais altos de R quadrado e valores mais baixos de Erro Padrão da Estimativa (Syx) e Syx%.

1. **R quadrado:** O modelo alométrico teve o valor mais alto de R quadrado (0.825), indicando uma melhor capacidade de explicar a variação nos dados em comparação com os outros modelos.

2. **Erro padrão da estimativa (Syx):** O modelo alométrico teve o valor mais baixo de Syx (0.130), sugerindo uma menor dispersão dos dados em torno da linha de regressão, o que indica uma melhor precisão das previsões.

3. **Syx%**: O modelo alométrico também teve o valor mais baixo de Syx% (0.099), indicando uma menor variação percentual das previsões em relação aos valores reais.

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 frequencias 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:

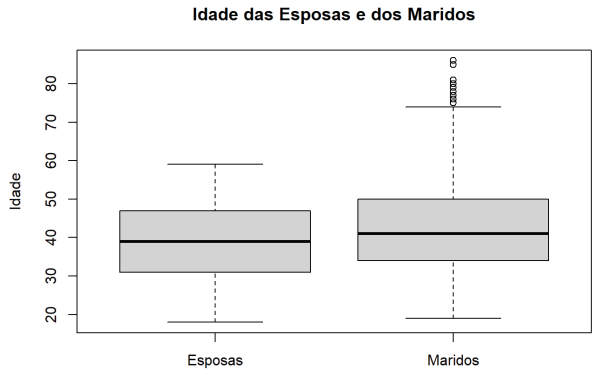
Você deve fazer os testes necessários (e mostra-los no documento pdf) para saber se você deve usar o unpaired test (paramétrico) ou o teste U de Mann-Whitney (não paramétrico), justifique sua resposta sobre a escolha.

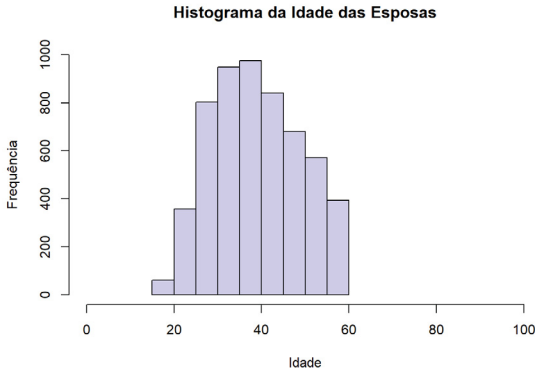
Lembre-se de que os intervalos de confiança já são mostrados nos resultados dos testes citados no item 1 acima.

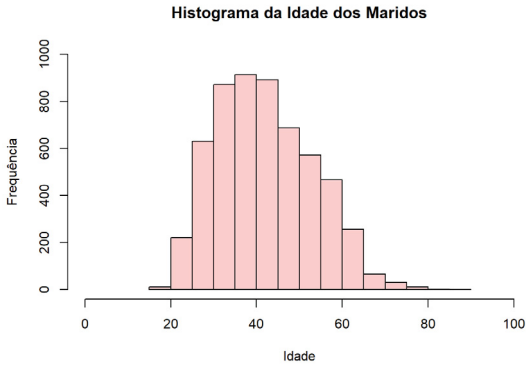
B – RESOLUÇÃO

1) Gráficos e tabelas

Distância interquartílica é a mesma (16). A principal diferença é que os Maridos possuem indivíduos com idades superiores a 60. Também é possível verificar a presença de outliers com idades acima de 74 anos entre os Maridos.







Observando as duas tabelas de frequências (Idade dos Maridos e Idade das Esposas) verifica-se que a principal diferença é que os Maridos possuem indivíduos com idades superiores a 60, enquanto as Esposas não.

2) Medidas de posição e dispersão

Média de Idade dos Maridos (42.45) um pouco maior (7,67%) que das Esposas (39.43). Mediana da Idade dos Maridos (41) também um pouco maior (5,13%) que das Esposas (39). Moda para Esposas é a idade de 37 anos, com 217 indivíduos, enquanto para os Maridos é de 44 anos com 201 indivíduos.

```
summary(df$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   31.00   39.00   39.43   47.00   59.00
```

```
summary(df$husage)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.00   34.00   41.00   42.45   50.00   86.00
```

 # Diferença das Medianas

```
mediana_dif = round((median(df$husage)/median(df$age)-1)*100,2)
```

```
## [1] "Maridos com mediana da idade 5.13 % maior que as Esposas"
```

 # Diferença das Médias

```
media_dif = round((mean(df$husage)/mean(df$age)-1)*100,2)
```

```
## [1] "Maridos com média de idade 7.67 % maior que as Esposas"
```

 # Moda para a idade das esposas

```
subset(table(df$age),
       table(df$age) == max(table(df$age)))
```

```
## 37
## 217
```

 # Moda para a Idade dos Maridos

```
subset(table(df$husage),
       table(df$husage) ==
         max(table(df$husage)))
```

```
## 44
## 201
```

Maridos com variância 26.38 % maior que as Esposas. Maridos com desvio padrão da idade 12.42 % maior que as Esposas. Coeficiente de variação muito próximo para ambos (25,33% e 26,45%), e indicam uma média dispersão dos resultados (entre 15% e 30%).

 # Variância

```
var_esposa = round(var(df$age),2)
var_marido = round(var(df$husage),2)
```

```
## [1] "Variância para a idade das Esposas= 99.75"
## [1] "Variância para a idade dos Maridos= 126.07"
```

 # Diferença das Variâncias

```
var_dif = round((var(df$husage)/var(df$age)-1)*100,2)
```

```
## [1] "Maridos com variância de idade 26.38 % maior que as Esposas"
```

 # Desvio Padrão

```
dp_esposa = round(sd(df$age),2)
dp_marido = round(sd(df$husage),2)
```

```
## [1] "Desvio Padrão para a idade das Esposas= 9.99"
## [1] "Desvio Padrão para a idade dos Maridos= 11.23"
```

 # Diferença dos Desvio Padrão

```
dp_dif = round((sd(df$husage)/sd(df$age)-1)*100,2)
```

[1] "Maridos com desvio padrão da idade 12.42 % maior que as Esposas"

 # Coeficiente de variação das idades (desvio padrão / média * 100)

```
cv_esposa <- round((sd(df$age)/mean(df$age))*100,2)
cv_marido <- round((sd(df$husage)/mean(df$husage))*100,2)
```

[1] "Coeficiente de variacao das idades das Esposas é de= 25.33 %"

[1] "Coeficiente de variacao das idades dos Maridos é de= 26.45 %"

3) Testes paramétricos ou não paramétricos

Tratam-se de amostras independentes; não normalmente distribuídas, conforme testes de normalidade de Kolmogorov-Smirnov e JarqueBera (em ambos os casos o valor de p é menor que 0,05, rejeitando a hipótese nula de que os dados são normalmente distribuídos); com variância não homogêneas, conforme teste F (valor de p é menor que 0,05 rejeitando a hipótese nula de que as variâncias são homogêneas). Em virtude desses parâmetros, foi escolhido o teste não-paramétrico U de Mann-Whitney, confirmando que a mediana da idade não são iguais para Maridos e Esposas (valor de p < 0,05, rejeitando a hipótese nula de que as medianas são estatisticamente iguais). Com 95% de intervalo de confiança, a diferença da mediana está entre -3.000024 e -2.000033.

```
# Kolmogorov-Smirnov test
ks.test(df$age,"pnorm",mean(df$age),sd(df$age))
      ##  One-sample Kolmogorov-Smirnov test
      ## data:  df$age
      ## D = 0.058909, p-value < 0.00000000000000022
      ## alternative hypothesis: two-sided
ks.test(df$husage,"pnorm",mean(df$husage),sd(df$husage))
      ##  One-sample Kolmogorov-Smirnov test
      ## data:  df$husage
      ## D = 0.059662, p-value < 0.00000000000000022
      ## alternative hypothesis: two-sided

# JarqueBera test
JarqueBeraTest(df$age, robust=TRUE)
      ##  Robust Jarque Bera Test
      ## data:  df$age
      ## X-squared = 158.49, df = 2, p-value < 0.00000000000000022
JarqueBeraTest(df$husage, robust=TRUE)
      ##  Robust Jarque Bera Test
      ## data:  df$husage
```

```
      ## X-squared = 153.12, df = 2, p-value < 0.00000000000000022
```

 # F test

```
var.test(x=df$age,y=df$husage, data = df)
      ##  F test to compare two variances
      ## data:  df$age and df$husage
      ## F = 0.79123, num df = 5633, denom df = 5633, p-value <
      ## 0.00000000000000022
      ## alternative hypothesis: true ratio of variances is not equal to 1
      ## 95 percent confidence interval:
      ##  0.7509664 0.8336625
      ## sample estimates:
      ## ratio of variances
      ##           0.7912349
max = round(qf(0.95,5633,5633),2)
min = round(1/max,2)
      ## [1] "Teste de F tem valor critico entre 0.96 e 1.04"
      # F = 0.79 portanto, região de rejeição de H0 => não são homogêneas

# U de Mann-Whitney
wilcox.test(Age ~ Sexo, data = df_emp, exact = FALSE, conf.int=TRUE)
      ##  Wilcoxon rank sum test with continuity correction
      ## data:  Age by Sexo
      ## W = 13619912, p-value < 0.00000000000000022
      ## alternative hypothesis: true location shift is not equal to 0
      ## 95 percent confidence interval:
      ##  -3.000024 -2.000033
      ## sample estimates:
      ## difference in location
      ##              -2.999966
```

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)
husblck = 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 logarítmo, portanto voçê não precisa aplicar o logaritmo nela para fazer as regressões, mas é necessário aplicar o antilog para obter o resultado da predição.

B – RESOLUÇÃO

Regressões Ridge, Lasso e ElasticNet

 # Recortes do Código

```
# TREINANDO Rige e Lasso
modelo_ridge = glmnet(x_train, y_train, nlambda = 25, alpha = 0,
                      family = 'gaussian',
                      lambda = best_lambda_ridge)
modelo_lasso = glmnet(x_train, y_train, alpha = 1,
                         lambda = best_lambda_lasso,
                         standardize = TRUE)

# Controle para ELASTIC NET
train_cont = trainControl(method = "repeatedcv",
                              number = 10,
                              repeats = 5,
                              search = "random",
                              verboseIter = FALSE)

# TREINAMENTO Elastic NET
modelo_elasticnet = train(lwage~husage+husunion+husearns+huseduc+husblck+
                            hushisp+hushrs+kidge6+earns+age+
                            black+educ+hispanic+union+exper+kidlt6,
                          data = train,
                          method = "glmnet",
                          tuneLength = 10,
                          trControl = train_cont)

best_lambda_elasticnet = modelo_elasticnet$bestTune[[2]]
```

Modelo	ALPHA	LAMDA
RIDGE	0	0.015848932
LASSO	1	0.010000000
ELASTIC_NET	0.739429239649326	0.005297002

PREDIÇÃO E AVALIAÇÃO

```
# Dados de TREINAMENTO:
prediction = predict(modelo_ridge,
                               s = best_lambda_ridge,
                               newx = x_train)
df = avaliacao_resultados(y_train, prediction, train)

prediction = predict(modelo_lasso,
```

```
                              s = best_lambda_lasso,
                              newx = x_train)
df = rbind(df,avaliacao_resultados(y_train, prediction, train))

prediction = predict(modelo_elasticnet, x_train)
df = rbind(df,avaliacao_resultados(y_train, prediction, train))

# Dados de TESTE:
prediction = predict(modelo_ridge,
                             s = best_lambda_ridge,
                             newx = x_test)
df = rbind(df,avaliacao_resultados(y_test, prediction, test))

prediction = predict(modelo_lasso,
                             s = best_lambda_lasso,
                             newx = x_test)
df = rbind(df,avaliacao_resultados(y_test, prediction, test))

prediction = predict(modelo_elasticnet, x_test)
df = rbind(df,avaliacao_resultados(y_test, prediction, test))
```

Modelo	RMSE	Rsquare
RIDGE	0.2967970	0.6838540
LASSO	0.2984090	0.6804104
ELASTIC NET	0.2969706	0.6834841
RIDGE	0.2630827	0.7187452
LASSO	0.2608951	0.7234033
ELASTIC NET	0.2616748	0.7217475

PREDIÇÕES E INTERVALOS DE CONFIANÇA

```
prediction = predict(modelo_ridge,
                              s = best_lambda_ridge,
                              newx = our_pred)
df = data.frame(prediction)

prediction = predict(modelo_lasso,
                              s = best_lambda_lasso,
                              newx = our_pred)
df = rbind(df,prediction)
```

```
prediction = predict(modelo_elasticnet, our_pred)
df = rbind(df,prediction)

# O resultado da predicao é salario por hora (US$)
df = exp(df)
lwage_antilog = exp(trabalhosalarios$lwage)

# O intervalo de confianca
n = nrow(train) # tamanho da amostra
s = sd(lwage_antilog) # desvio padrao
dam = s/sqrt(n) # distribuicao da amostragem da media

df$interval_inf = df[,1] + (qnorm(0.025))*dam # intervalo inferior
df$interval_sup = df[,1] - (qnorm(0.025))*dam # intervalo superior
```

Modelo	salario_previsto	interval_inf	interval_sup
RIDGE	8.082569	7.822426	8.342712
LASSO	8.640474	8.380331	8.900618
ELASTIC_NET	8.591191	8.331048	8.851334

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 disponibilidade e crie as funções para calcular as diferentes características para o problema da identificação da língua do texto de entrada.

Nessa atividade é para "construir características".

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

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

2 Melhore uma base de dados ruim

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

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

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

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

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

B – RESOLUÇÃO

1- Construção de Características: Identificador automático de idioma

 # Recortes do Código

Amostras de texto em diferentes línguas

```
ingles = [
"Hello, how are you?",
"I love to read books.",
… ]
```

```
espanhol = [
"Hola, ¿cómo estás?",
…]

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

def extraiCaracteristicas(frase: str) -> int:
  texto = frase[0]
  pattern_regex = re.compile('[^\w+]', re.UNICODE)
  texto = re.sub(pattern_regex,' ',texto)

  caracteristica1 = tamanhoMedioFrases(texto)
  caracteristica2 = caracteristicas_do_ingles(texto)
  caracteristica3 = caracteristicas_do_espanhol(texto)
  caracteristica4 = caracteristicas_do_portugues(texto)
  caracteristica5 = quantidade_de_palavras_na_frase(texto)
  caracteristica6 = caracteristicas_latinas(texto)
  padrao = [caracteristica1, caracteristica2, caracteristica3,
            caracteristica4, caracteristica5, caracteristica6, frase[1]]
  return padrao


Executa o treinamento
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import numpy as np

vet = np.array(padroes)
classes = vet[:,-1]
padroes_sem_classe = vet[:,0:-1]

scaler = StandardScaler(with_mean=False, with_std=True)
padroes_sem_classe = scaler.fit_transform(padroes_sem_classe)
```

```
X_train, X_test, y_train, y_test = train_test_split(padroes_sem_classe,
classes, test_size=0.30, stratify=classes)
acuracia = modelo.score(X_train, y_train)
acuracia_x = modelo.score(X_test, y_test)
```

Resultados:

Acurácia nos dados de treinamento: 76.56%

Acurácia nos dados de teste: 78.57%

```
[[11  3  7]
 [ 1 18  2]
 [ 2  0 20]]
```

	precision	recall	f1-score	support
espanhol	0.79	0.52	0.63	21
inglês	0.86	0.86	0.86	21
português	0.69	0.91	0.78	22
accuracy			0.77	64
macro avg	0.78	0.76	0.76	64
weighted avg	0.78	0.77	0.76	64

2- Melhore uma base de dados ruim

Evolução da solução com a aplicação das técnicas de pré-processamento:

Base de dados utilizada foi: https://archive.ics.uci.edu/dataset/545/rice+cammeo+and+osmancik

desempenho sem fazer nada: 88%

Aplicando min Max nas colunas numericas:92%

Encodando a classe: 92+

Remove colunas com pouca variância: 92 ~93%

 # Recortes do Código

```
colunas = ['area', 'perimeter', 'major_axis_length', 'minor_axis_length',
'eccentricity', 'convex_area', 'extent', 'Class']
url =
'https://raw.githubusercontent.com/eynan/IAA_UFPR/main/Rice_Cammeo_Osmancik.c
sv'
arroz_df = pd.read_csv(url, names=colunas)
valores_unicos = arroz_df['Class'].value_counts()
arroz_df = arroz_df.drop_duplicates()
classe = arroz_df[['Class']]
```

```
sem_classe = arroz_df.iloc[:, :-1]
encoder = LabelEncoder()
encoder.fit(classe['Class'])
classe['Class'] = encoder.transform(classe['Class'])
selector = VarianceThreshold(threshold=0.25)
selector.fit(sem_classe)
selected_class = sem_classe.columns[selector.get_support()]
# Selecionando as features com variância acima do limiar
sem_classe = arroz_df[selected_class]
# Aplicar normalização Min-Max
scaler = MinMaxScaler()
sem_classe          =          pd.DataFrame(scaler.fit_transform(sem_classe),
columns=sem_classe.columns)

treinador = svm.SVC()  #algoritmo escolhido
modelo = treinador.fit(X_train, y_train)
acuracia = modelo.score(X_train, y_train)
y_pred = modelo.predict(X_train)
cm = confusion_matrix(y_train, y_pred)
y_pred2 = modelo.predict(X_test)
cm = confusion_matrix(y_test, y_pred2)
#Adicionando a validação k-fold
n_splits = 5
kf = KFold(n_splits=n_splits, shuffle=True, random_state=42)
scores    =    cross_val_score(treinador,    X_train,    y_train,    cv=kf,
scoring='accuracy')
```

Acurácia nos dados de treinamento: 92.72%

	precision	recall	f1-score	support
0	0.92	0.90	0.91	1222
1	0.93	0.94	0.94	1635
accuracy			0.93	2857
macro avg	0.93	0.92	0.93	2857
weighted avg	0.93	0.93	0.93	2857

Acurácia (Cross-Validation): 92.58% Desvio Padrão da Acurácia: 0.99%

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

Classificação:

Veículo

 # Recortes do Código

```
########## Veiculos
setwd("/Users/renansilva/documents/pos-Iaa/aprendizado de maquina/praticas/06
- Veículos")
dados_veiculos <- read.csv("6 - Veiculos - Dados.csv", header=T)
dados_veiculos$a <- NULL
set.seed(202479)
indices <- createDataPartition(dados_veiculos$tipo, p=0.80, list=FALSE)
treino <- dados_veiculos[indices,]
teste <- dados_veiculos[-indices,]

### Gerar um novo modelo usando SVM, predições e matriz de confusão
rf <- train(tipo~., data=treino, method="rf")
rf
### Faz a predição e mostra a matriz de confusão
predict.rf <- predict(rf, teste)
confusionMatrix(predict.rf, as.factor(teste$tipo))

### PREDIÇÕES DE NOVOS CASOS
dados_novos_casos <- read.csv("novos_casos.csv")
dados_novos_casos$a <- NULL
View(dados_novos_casos)
predict.rf <- predict(rf, dados_novos_casos)
dados_novos_casos$Class <- NULL
resultado <- cbind(dados_novos_casos, predict.rf)
View(resultado)
```

Técnica	Parâmetro	Acurácia	Matriz de Confusão
RF – Hold-out	mtry=2	0.7485	Prediction bus opel saab van bus 43 1 2 0 opel 0 21 17 0 saab 0 17 22 0 van 0 3 2 39
RF – CV	mtry=10	0.7485	Prediction bus opel saab van bus 43 0 2 0 opel 0 20 17 0 saab 0 20 23 0 van 0 2 1 39
SVM – Hold-out	C=1 Sigma=0.05691857	0.7425	Prediction bus opel saab van bus 42 0 1 0 opel 0 17 13 0 saab 0 23 26 0 van 1 2 3 39
SVM – CV	C=1 Sigma=0.0532951	0.7305	Prediction bus opel saab van bus 42 0 1 0 opel 0 17 14 0 saab 0 23 24 0 van 1 2 4 39
RNA – Hold-out	size=5 decay=0.1	0.5509	Prediction bus opel saab van bus 16 3 0 0 opel 0 4 6 0 saab 1 35 34 1 van 26 0 3 38
RNA – CV	size=3 decay=0.1	0.4491	Prediction bus opel saab van bus 38 40 42 2 opel 0 0 0 0 saab 0 0 0 0 van 5 2 1 37
KNN	k=9	0.4412	Prediction bus opel saab van bus 35 3 7 14 opel 9 9 10 0 saab 8 18 15 0 van 4 11 11 16

Diabetes

 # Recortes do Código

```
############ Diabetes
setwd("/Users/renansilva/documents/pos-Iaa/aprendizado de maquina/praticas/10
- Diabetes")
dados_diabetes <- read.csv("10 - Diabetes - Dados.csv", header=T)
set.seed(202479)
dados_diabetes$num <- NULL
indices <- createDataPartition(dados_diabetes$diabetes, p=0.80, list=FALSE)
treino <- dados_diabetes[indices,]
teste <- dados_diabetes[-indices,]

### Gerar um novo modelo usando SVM, predições e matriz de confusão
svm <- train(diabetes~., data=treino, method="svmRadial")
svm
### Faz a predição e mostra a matriz de confusão
predict.svm <- predict(svm, teste)
confusionMatrix(predict.svm, as.factor(teste$diabetes))

### PREDIÇÕES DE NOVOS CASOS
dados_novos_casos <- read.csv("novos_casos.csv")
dados_novos_casos$num <- NULL
View(dados_novos_casos)
predict.svm <- predict(svm, dados_novos_casos)
dados_novos_casos$Class <- NULL
resultado <- cbind(dados_novos_casos, predict.svm)
View(resultado)
```

Técnica	Parâmetro	Acurácia	Matriz de Confusão
SVM – Hold-out	C=0.25 Sigma=0.1056637	0.7974	Prediction neg pos neg 92 23 pos 8 30
RF – CV	mtry=5	0.7843	Prediction neg pos neg 87 20 pos 13 33
SVM – CV	C=0.5 Sigma=0.1031122	0.7778	Prediction neg pos neg 89 23 pos 11 30
RF – Hold-out	mtry=9	0.7647	Prediction neg pos neg 86 22 pos 14 31

RNA – Hold-out	size=3 decay=0.1	0.7059	Prediction neg pos neg 86 31 pos 14 22
RNA – CV	size=5 decay=0.1	0.6993	Prediction neg pos neg 83 29 pos 17 24
KNN	k=9	0.6753	Prediction neg pos neg 85 38 pos 12 19

Regressão

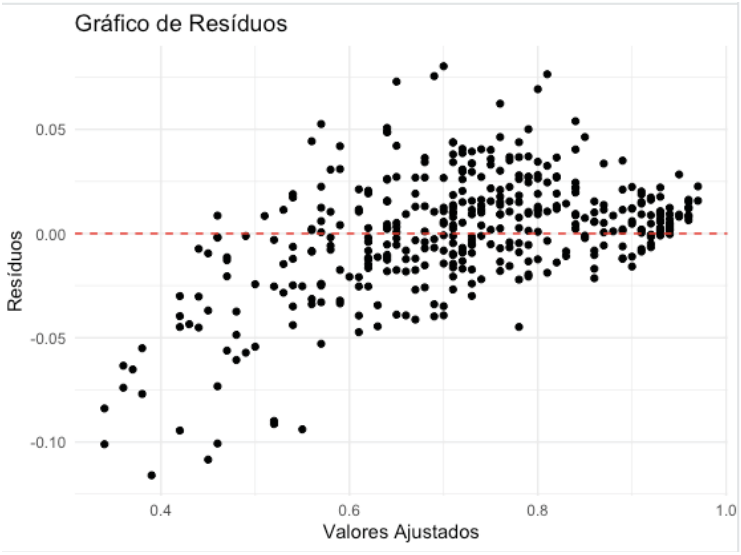
Admissão

 # Recortes do Código

```
### Leitura dos dados
setwd("/Users/renansilva/documents/pos-Iaa/aprendizado de maquina/praticas/09
- Admissa˜o")
dados <- read.csv("9 - Admissao - Dados.csv", header=T)
View(dados)
### Cria arquivos de treino e teste
set.seed(202479)
ind <- createDataPartition(dados$ChanceOfAdmit, p=0.80, list=FALSE)
treino <- dados[ind,]
teste <- dados[-ind,]
tuneGrid = expand.grid(mtry=c(2, 5, 7, 9))
rf <- train(ChanceOfAdmit~., data=treino, method="rf", tuneGrid=tuneGrid)
predicoes.rf <- predict(rf, teste)
r2 <- function(predito, observado) {
                return(1        -        (sum((predito-observado)^2)        /
sum((observado-mean(observado))^2)))
}
rmse(teste$ChanceOfAdmit, predicoes.rf)
r2(predicoes.rf,teste$ChanceOfAdmit)
mae(teste$ChanceOfAdmit, predicoes.rf)
cor(predicoes.rf, teste$ChanceOfAdmit)
residuos <- teste$ChanceOfAdmit - predicoes.rf
ssr <- sum(residuos^2)

## grafico de residuos
```

```
residuos <- residuals(rf)
dados_residuais <- data.frame(x=treino$ChanceOfAdmit, residuos=residuos)

# Plotar os resíduos
ggplot(dados_residuais, aes(x = x, y = residuos)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Gráfico de Resíduos",
       x = "Valores Ajustados",
       y = "Resíduos") +
  theme_minimal()
```

Técnica	Parâmetro	R2	Syx	Pearson	Rmse	MAE
RF – Hold-out	mtry=2	0.8718803	0.2093901	0.9365293	0.04622373	0.0342956
RF – CV	mtry=2	0.8669496	0.2485169	0.92432	0.0471048	0.03533905
SVM – CV	C=50 Sigma=0.015	0.8654763	0.3379901	0.9158425	0.04736489	0.0353502
SVM – Hold-out	C=1 Sigma=0.129469	0.8595345	0.2295673	0.9288191	0.04839962	0.03625344
RNA – Hold-out	size=5 decay=1e-04	0.8509534	0.2435915	0.924567	0.04985607	0.03769128
RNA – CV	size=10 decay=0.1	0.8482703	0.2435915	0.9230228	0.05030282	0.03855418
KNN	K=5	0.6887266	0.508724	0.833428	0.08131621	0.0567551



Biomassa

 # Recortes do Código

```
setwd("/Users/renansilva/documents/pos-Iaa/aprendizado de maquina/praticas/05
- Biomassa")
dados <- read.csv("5 - Biomassa - Dados.csv", header=T)
set.seed(202479)
ind <- createDataPartition(dados$biomassa, p=0.80, list=FALSE)
treino <- dados[ind,]
teste <- dados[-ind,]
### Prepara um grid com os valores de
tuneGrid = expand.grid(C=c(1, 2, 10, 50, 100), sigma=c(.01, .015, 0.2))
### Executa o smv com esse grid
svm <- train(biomassa~., data=treino, method="svmRadial", tuneGrid=tuneGrid)
### Aplica o modelo no arquivo de teste
predict.svm <- predict(svm, teste)
### Mostra as métricas
rmse(teste$biomassa, predict.svm)
r2 <- function(predito, observado) {
                return(1        -        (sum((predito-observado)^2)        /
sum((observado-mean(observado))^2)))
}
r2(predict.svm,teste$biomassa)
mae(teste$biomassa, predict.svm)
cor(predicoes.svm, teste$biomassa)
residuos <- teste$biomassa - predicoes.svm
ssr <- sum(residuos^2)
## grafico de residuos
residuos <- teste$biomassa - predict.svm
dados_residuais <- data.frame(x=treino$biomassa, residuos=residuos)

# Plotar os resíduos
ggplot(dados_residuais, aes(x = x, y = residuos)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Gráfico de Resíduos",
       x = "Valores Ajustados",
       y = "Resíduos") +
  theme_minimal()
```

Técnica	Parâmetro	R2	Syx	Pearson	Rmse	MAE

SVM – Hold-out	C=50 Sigma=0.01	0.9213451	28262945	0.9879051	686.3302	154.6674
RNA – CV	size=6 decay=0.7	0.7780533	79751813	0.9436218	1152.908	254.6021
KNN	K=1	0.6983408	108394808	0.8559113	1344.091	272.5272
RF – Hold-out	mtry=2	0.683581	113698406	0.9310649	1376.58	221.8353
SVM – CV	C=10 Sigma=0.01	0.677084	116032983	0.9535489	1390.641	256.1934
RF – CV	mtry=2	0.6755027	116601189	0.933685	1394.042	220.6593
RNA – Hold-out	size=6 decay=0.7	0.648118	126441294	0.8418479	1451.673	292.9453

Agrupamento

veiculo

Listagem dos cluster gerados

Cluster modes:

	Comp	Circ	DCirc	RadRa	PrAxisRa	MaxLRa	ScatRa	Elong	PrAxisRect	MaxLRect	ScVarMaxis	ScVarmaxis	RaGyr	SkewMaxis	Skewmaxis	Kurtmaxis	KurtMaxis	HollRa	Class
1	85	43	68	123	54	7	150	46	19	144	169	317	171	85	4	14	179	182	bus
2	90	37	66	136	58	6	135	50	18	127	165	262	144	69	3	21	180	185	saab
3	85	45	70	155	64	7	151	45	19	145	170	331	186	72	0	0	180	183	bus
4	97	44	96	197	63	9	185	35	22	161	202	342	198	66	4	11	193	199	opel
5	100	55	101	209	62	10	201	33	23	147	214	602	204	73	5	11	186	197	opel
6	93	36	66	117	56	8	128	52	18	130	159	246	127	62	1	3	201	183	van
7	86	37	53	110	58	6	122	57	17	128	135	209	137	64	7	7	199	183	van
8	104	53	103	197	64	11	218	31	24	162	226	709	214	71	0	4	189	198	saab
9	86	38	66	125	59	8	130	51	18	131	162	259	137	67	0	1	192	201	opel
10	89	46	85	160	64	11	157	43	20	148	173	354	186	75	1	9	183	193	van

10 primeiras linhas do arquivo com o cluster correspondente.

	Comp	Circ	DCirc	RadRa	PrAxisRa	MaxLRa	ScatRa	Elong	PrAxisRect	MaxLRect	ScVarMaxis	ScVarmaxis	RaGyr	SkewMaxis	Skewmaxis	Kurtmaxis	KurtMaxis	HollRa	Class	cluster.results$cluster
1	95	48	83	178	72	10	162	42	20	159	176	379	184	70	6	16	187	197	van	10
2	91	41	84	141	57	9	149	45	19	143	170	330	158	72	9	14	189	199	van	3
3	104	50	106	209	66	10	207	32	23	158	223	635	220	73	14	9	188	196	saab	5
4	93	41	82	159	63	9	144	46	19	143	160	309	127	63	6	10	199	207	van	7
5	85	44	70	205	103	52	149	45	19	144	241	325	188	127	9	11	180	183	bus	3
6	107	57	106	172	50	6	255	26	28	169	280	957	264	85	5	9	181	183	bus	1
7	97	43	73	173	65	6	153	42	19	143	176	361	172	66	13	1	200	204	bus	1
8	90	43	66	157	65	9	137	48	18	146	162	281	164	67	3	3	193	202	van	9
9	86	34	62	140	61	7	122	54	17	127	141	223	112	64	2	14	200	208	van	7
10	93	44	98	197	62	11	183	36	22	146	202	505	152	64	4	14	195	204	saab	4

 # Recortes do Código

```
setwd("/Users/renansilva/documents/pos-Iaa/aprendizado de maquina/praticas/11
- Agrupamento/")
dados <- read.csv("4 - Veiculos - Dados.csv")
View(dados)
set.seed(202479)
dados$a <- NULL

## Executa o cluster
cluster.results <- kmodes(dados, 10, iter.max = 10, weighted = FALSE )
cluster.results

resultado <- cbind(dados, cluster.results$cluster)
```

Regras de associação

musculação

Regras geradas com uma configuração de Suporte e Confiança.

Colocar a lista de comandos emitidos no RStudio para conseguir os resultados obtidos

lhs	rhs	support	confidence	coverage	lift	count
[1] {Crucifixo}	=> {Afundo}	0.07692308	1.0000000	0.07692308	2.8888889	2
[2] {Crucifixo}	=> {Gemeos}	0.07692308	1.0000000	0.07692308	1.5294118	2
[3] {Crucifixo}	=> {LegPress}	0.07692308	1.0000000	0.07692308	1.2380952	2
[4] {Adutor}	=> {Agachamento}	0.11538462	1.0000000	0.11538462	3.25000	3
[5] {Adutor}	=> {LegPress}	0.11538462	1.0000000	0.11538462	1.2380952	3

 # Recortes do Código

```
setwd("/Users/renansilva/documents/pos-Iaa/aprendizado de maquina/praticas/")
dados <- read.csv("2 - Musculacao - Dados.csv")
dados <- read.transactions(file="2 - Musculacao - Dados.csv",format="basket",sep=";")
View(dados)
set.seed(202479)
rules <- apriori(dados, parameter=list(supp = 0.001, conf = 0.7, minlen=2))
summary(rules)
inspect(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.

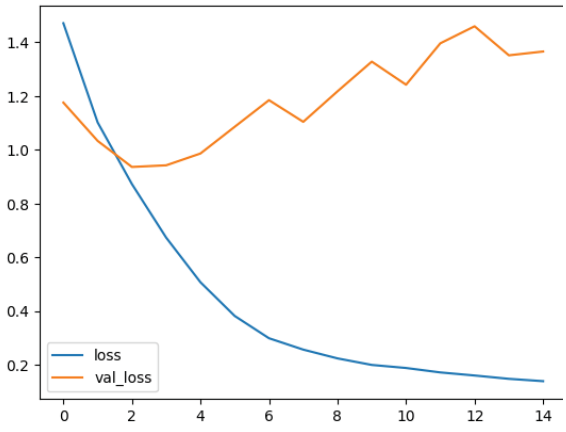
B – RESOLUÇÃO

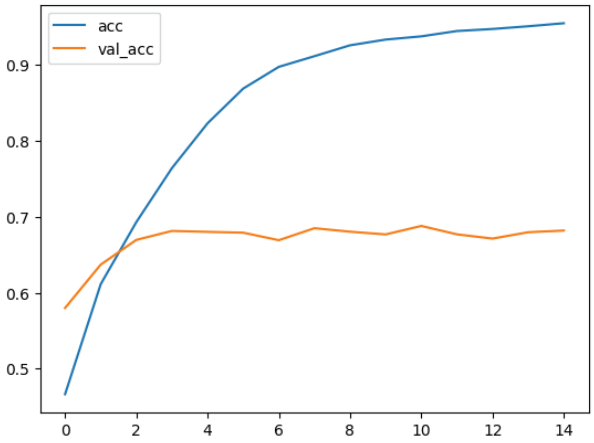
1 Classificação de Imagens (CNN)

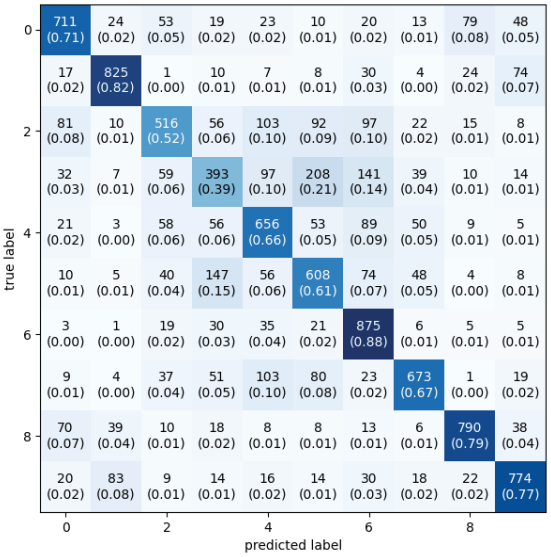
 # Recortes do Código

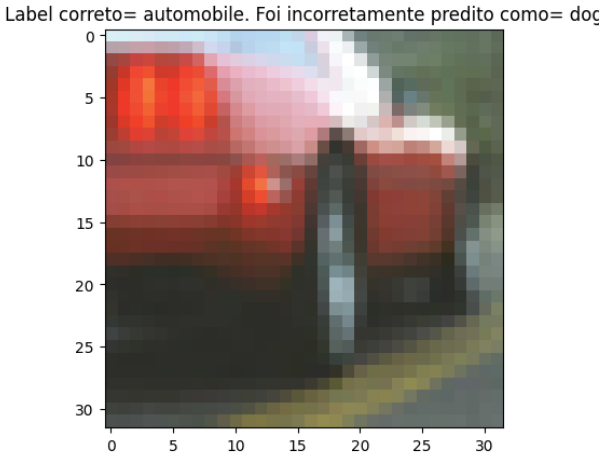
```
K = len(set(y_train))
print("número de classes:", K)
i = Input(shape=x_train[0].shape)
x = Conv2D(32, (3, 3), strides=1, activation='relu')(i)
x = Conv2D(32, (3, 3), strides=1, activation='relu')(x)
x = Conv2D(32, (3, 3), strides=1, activation='relu')(x)
x = Flatten()(x)
x = Dropout(0.5)(x)
x = Dense(1024, activation='relu')(x)
```

```
x = Dropout(0.25)(x)
x = Dense(K, activation='softmax')(x)
model = Model(i, x)
model.summary()
model.compile(optimizer='adam',        loss='sparse_categorical_crossentropy',
metrics=['accuracy'])
r = model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=15)
```







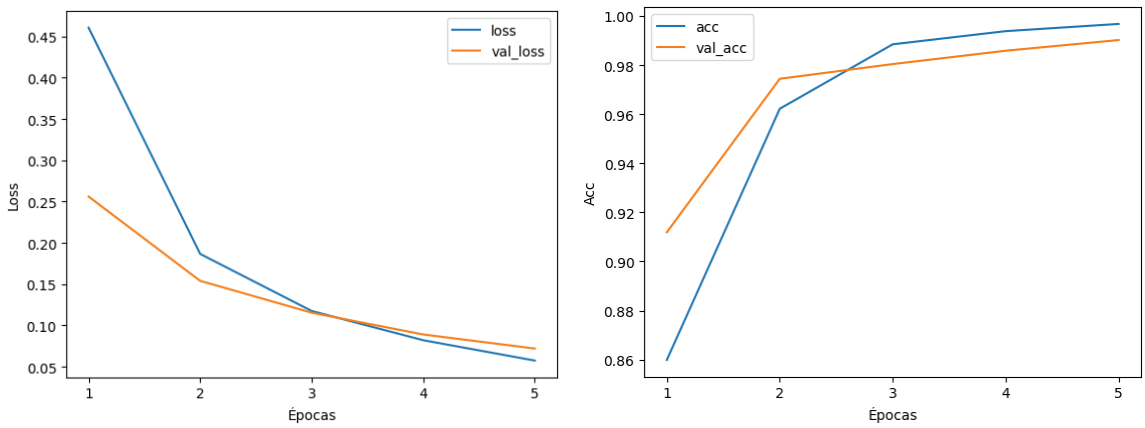


2 Detector de SPAM (RNN)

 # Recortes do Código

```
Dimension_embed = 20
Hidden_state = 5
i = Input(shape=(max_len,))
```

```
# Layer embedding cria um vetor numérico para cada palavra (word2vec),
calculado a partir das palavras e suas palavras adjacentes (janelas ou
windows)
x = Embedding(num_words+1,Dimension_embed)(i)
# Layer LSTM com 5 unidades, com a função de relembrar (long memory) os
estados anteriores
x = LSTM(Hidden_state)(x)
# Camada final para indicar se é ou não SPAM
x = Dense(1,activation='sigmoid')(x)
model = Model(i,x)
model.summary()

# treina o modelo
model.compile(loss                                                    =
"binary_crossentropy",optimizer="adam",metrics=["accuracy"])
epochs = 5
r                                                                     =
model.fit(data_train,y_train,epochs=epochs,validation_data=(data_test,y_test)
)
```



3 Gerador de Dígitos Fake (GAN)

 # Recortes do Código

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

```
  model.add(layers.Reshape((7, 7, 256)))
  assert model.output_shape == (None, 7, 7, 256)

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

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

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

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

# Cria o DISCRIMINADOR
def make_discriminator_model():
  model = tf.keras.Sequential()
  model.add(layers.Conv2D(64, (5, 5), strides=(2, 2), padding='same',
  input_shape=[28, 28, 1]))
  model.add(layers.LeakyReLU())
  model.add(layers.Dropout(0.3))
  model.add(layers.Conv2D(128, (5, 5), strides=(2, 2), padding='same'))
  model.add(layers.LeakyReLU())
  model.add(layers.Dropout(0.3))
  model.add(layers.Flatten())
  model.add(layers.Dense(1))
  return model
```

Imagem gerada inicialmente, ainda sem o treinamento.

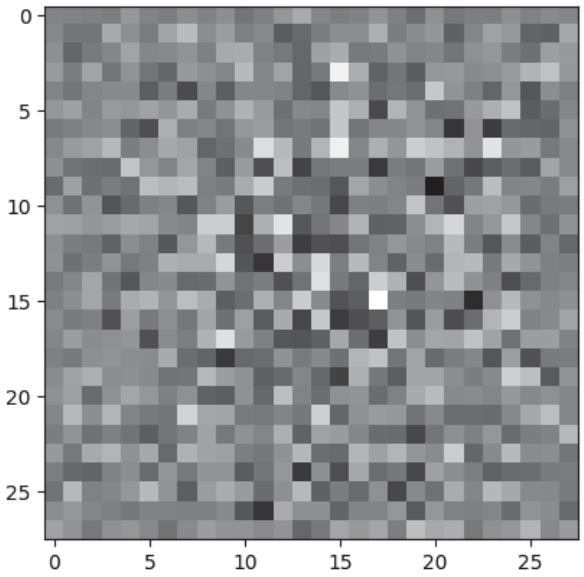
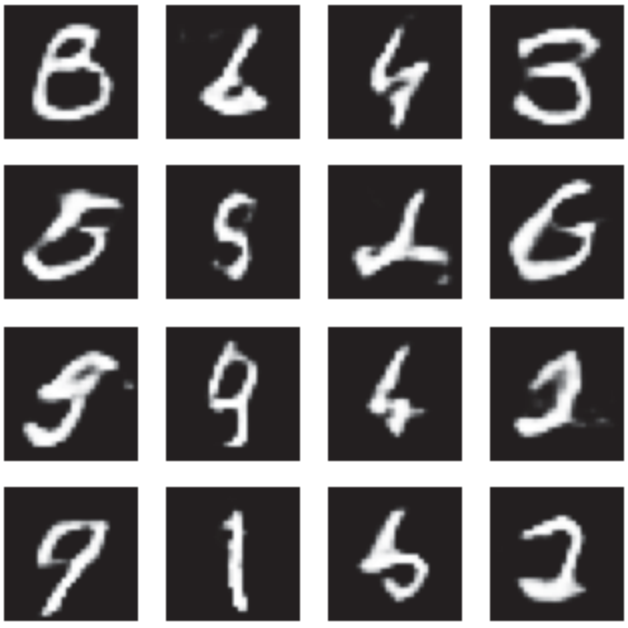
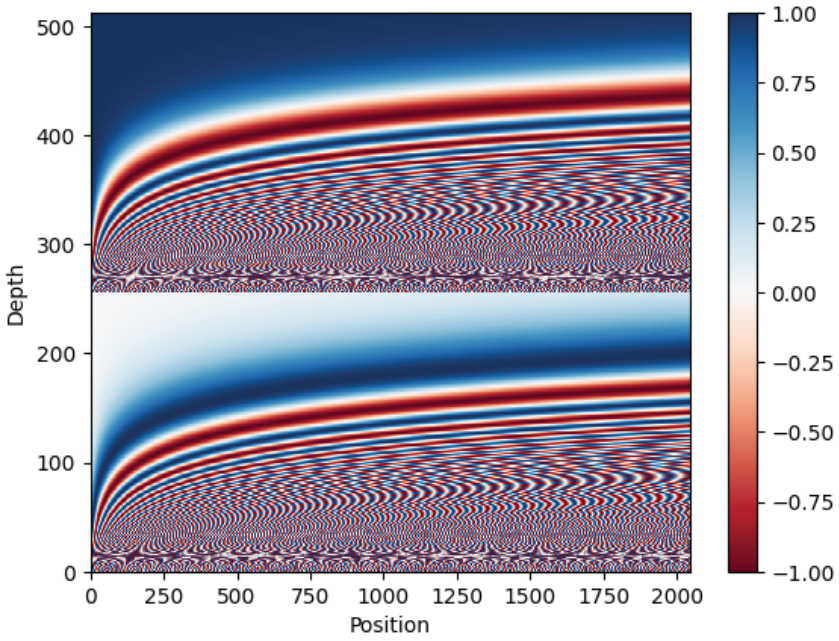


Imagem de dígitos gerada após o treinamento



4 Tradutor de Textos (Transformer)



 # Recortes do Código

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

```
    out2 = self.layernorm2(out1 + ffn_output) # (batch_size, input_seq_len,
d_model)
return out2

class Decoder(tf.keras.layers.Layer):
  def __init__(self, num_layers, d_model, num_heads, dff, target_vocab_size,
maximum_position_encoding, rate=0.1):
    super(Decoder, self).__init__()
    self.d_model = d_model
    self.num_layers = num_layers
    self.embedding = tf.keras.layers.Embedding(target_vocab_size, d_model)
    self.pos_encoding = positional_encoding(maximum_position_encoding,
d_model)
    self.dec_layers = [DecoderLayer(d_model, num_heads, dff, rate) for _ in
range(num_layers)]
    self.dropout = tf.keras.layers.Dropout(rate)

  def call(self, x, enc_output, training, look_ahead_mask, padding_mask):
    seq_len = tf.shape(x)[1]
    attention_weights = {}
    x = self.embedding(x) # (batch_size, target_seq_len, d_model)
    x *= tf.math.sqrt(tf.cast(self.d_model, tf.float32))
    x += self.pos_encoding[:, :seq_len, :]
    x = self.dropout(x, training=training)
    for i in range(self.num_layers):
      x, block1, block2 = self.dec_layers[i](x, enc_output, training,
look_ahead_mask, padding_mask)
      attention_weights[f'decoder_layer{i+1}_block1'] = block1
      attention_weights[f'decoder_layer{i+1}_block2'] = block2
    # x.shape == (batch_size, target_seq_len, d_model)
    return x, attention_weights
```

Teste de tradução

```
translator = Translator(tokenizers, transformer)
sentence = "A inteligência artificial é uma ferramenta poderosa para a
tradução de textos."
translated_text, translated_tokens, attention_weights =
translator(tf.constant(sentence))
print(f'{"Prediction":15s}: {translated_text}')
```

Prediction : 'artificial intelligence is a powerful tool for translation of text.'

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

Apresentar a resolução (somente o resultado) das questões do trabalho.

Estudo de Caso

Monitoramento e Mitigação de Ataques Cibernéticos no contexto do Big Data

1. Contexto

O crescente aumento da produção de dados pelas empresas com o uso das tecnologias disruptivas pelas empresas, como o *Big Data*, Internet das Coisas, Computação em Nuvem e a Inteligência Artificial, tem tornado os dados um ativo de valor. Nesse contexto as organizações enfrentam desafios relacionados com a Segurança Cibernética para proteção dos dados, tanto pessoais como os corporativos. Os ataques cibernéticos fazem uso da dependência das tecnologias da informação (TI) no ambiente corporativo. O estudo de caso foca em uma empresa de pequeno porte que compartilha dados de sensores de monitoramento de máquinas. Considerando as tentativas de invasão, a empresa decidiu implementar uma solução de *Big Data* para monitoramento em tempo real e mitigação de ameaças cibernéticas.

2. Objetivo

O objetivo deste estudo de caso é demonstrar como o uso de *Big Data* pode ajudar na identificação, análise e resposta a incidentes de segurança cibernética. A solução proposta visa monitorar continuamente as atividades da rede, detectar padrões anômalos e tomar medidas preventivas para mitigar os ataques.

3. Ferramentas e Tecnologias Utilizadas

- **Sistemas de Log Management e SIEM (*Security Information and Event Management*):**

a) *Splunk* e *Elastic Stack*: para obter, agregar e analisar de grandes volumes de logs gerados por diferentes componentes de TI.

b) *IBM QRadar ou ArcSight: para correlacionar eventos de segurança em tempo real.*

- **Plataformas de *Big Data*:**

a) *Apache Hadoop*: para armazenamento distribuído e processamento em larga escala dos dados de *logs* e eventos de segurança.

b) *Apache Spark*: para processamento em tempo real de fluxos de dados, integrando-se a sistemas SIEM para análises rápidas.

- **Bancos de Dados *NoSQL*:**

a) *MongoDB* ou *Cassandra*: para armazenar logs de eventos, registros de atividades de rede, e dados de ameaças de fontes externas.

- ***Machine Learning* e Análise Comportamental:**

a) *TensorFlow* ou *Scikit-Learn*: para desenvolver modelos preditivos de detecção de anomalias com base em comportamento histórico da rede.

b) *Jupyter Notebooks*: para desenvolvimento de modelos e visualização de dados.

- **Ferramentas de Automação e Resposta:**

a) *Ansible* ou *Puppet*: para orquestração de respostas automatizadas a incidentes.

b) *SOAR (Security Orchestration, Automation, and Response)*: para automação de fluxos de trabalho de segurança, como integração de *Phantom* com o SIEM.

- **Ferramentas de Visualização:**

a) *Kibana* (parte do Elastic Stack) ou *Grafana*: para criar dashboards em tempo real, mostrando padrões de tráfego suspeitos, alertas de segurança e outras métricas críticas.

4. Caracterização dos Dados

Os dados coletados e analisados incluem:

a) Logs de Segurança: *Logs* de *firewalls*, IDS/IPS, *proxies*, e *endpoints*.

b) Dados de Rede: padrões de tráfego de rede, fluxos NetFlow, e registros DNS.

c) Ameaças Conhecidas: informações de fontes de inteligência de ameaças para correlação com eventos de segurança.

d) Dados Temporais: evolução dos padrões de tráfego e comportamento ao longo do tempo.

5. Os Vs do *Big Data*

a) Volume: grande quantidade de logs e eventos gerados em tempo real por dispositivos de segurança.

b) Velocidade: necessidade de análise em tempo real para detectar e mitigar ameaças antes que causem danos significativos.

c) Variedade: diversos tipos de dados, desde logs estruturados até dados não estruturados de inteligência de ameaças.

6. Modelagem de Dados

a) **Modelo de Tempo Real:** utilizando *Apache Kafka* para ingestão de dados em tempo real e *Spark Streaming* para processamento contínuo.

7. Sistemas para Séries Temporais

Para trabalhar com dados temporais e séries temporais, que são essenciais para monitoramento e análise em tempo real, podem ser consideradas as ferramentas e sistemas:

a) *InfluxDB*: Um banco de dados de séries temporais altamente escalável e otimizado para armazenar e consultar dados temporais. É ideal para monitoramento de métricas e eventos ao longo do tempo.

b) *TimescaleDB*: Uma extensão para PostgreSQL que adiciona suporte a dados de séries temporais, combinando a robustez do PostgreSQL com a eficiência para dados temporais.

c) *OpenTSDB*: Um sistema de banco de dados distribuído projetado para armazenar e consultar grandes volumes de dados temporais, com suporte a escalabilidade horizontal.

d) *Prometheus*: Uma ferramenta de monitoramento e alerta que coleta e armazena métricas como séries temporais e é eficaz para aplicações em tempo real.

8. *Kafka* para indicadores de comportamentos

Apache Kafka pode ser usado para processar e analisar padrões de comportamento em tempo real através de suas funcionalidades de processamento de fluxo e integração com sistemas de análise. Aqui estão algumas considerações sobre como Kafka pode ser utilizado para indicadores de comportamentos:

a) *Apache Kafka Streams*: é uma biblioteca para processamento de fluxo em tempo real que permite a análise e transformação de dados enquanto eles são consumidos. É útil para identificar padrões e comportamentos anômalos nos dados.

b) *Kafka Connect*: pode ser usado para integrar Kafka com outras fontes e sistemas de dados, facilitando a ingestão de dados de séries temporais e eventos de segurança.

c) Janelas de Tempo: no contexto do *Kafka*, pode ser utilizado janelas de tempo para agrupar eventos em períodos específicos (como minutos ou horas). Isso permite analisar padrões e comportamentos ao longo do tempo. As janelas podem ser definidas como janelas deslizantes, *tumbling* ou sessões, dependendo das necessidades da análise.

d) *KSQL*: é uma interface SQL para *Kafka* que permite criar consultas sobre streams de dados em tempo real. Isso pode ser usado para detectar comportamentos e padrões anômalos diretamente nos dados que estão sendo processados.

9. Modelagem de Dados e Análise Comportamental

Para modelar dados em tempo real e analisar padrões de comportamento, considere as seguintes etapas:

a) Ingestão de dados com o *Apache Kafka* para obter dados em tempo real de diversas fontes, como logs de segurança e tráfego de rede.

b) Processamento e análise com o *Apache Spark* para processar e analisar dados em fluxo, integrando-se com *Kafka* para análise contínua e detecção de anomalias.

c) Armazenamento de séries temporais com os sistemas especializados em séries temporais, como InfluxDB ou TimescaleDB, para armazenar e consultar dados de eventos e métricas ao longo do tempo.

d) Visualização e monitoramento com ferramentas, como Kibana e Grafana, podem ser usadas para criar *dashboards* e visualizar os dados temporais e eventos em tempo real.

As ferramentas e práticas mencionadas podem ajudar no contexto de uma solução para monitoramento e mitigação de ameaças cibernéticas, considerando as capacidades de *Big Data* e processamento em tempo real.

APÊNDICE 10 – VISÃO COMPUTACIONAL

A – ENUNCIADO

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

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

 # Recortes do Código

```
   # Extrair características
     features = model.predict(img_array)
     # Achatar as características (flatten) de (7, 7, 512) para (7*7*512,)
     resultado = features.flatten()
     return resultado

def calcula_caracteristicas(arq, tipo):
  img_path = open(arq, 'r')
  features_vgg16 = []
  features_lbp = []
  labels_list = []
  for line in img_path:
    #line = img_path.readline()
    image_block = line.rstrip('\n')
    nums = image_block.split('.')
    nums = nums[0].split('_')
    label = nums[3]
    #Le imagem original
    img = cv2.imread(image_block,0)

    labels_list.append(label)
    #Chama as funcoes de caracteristicas
    caracteristicas_lbp = lbp(img, label, arq, tipo)
    features_lbp.append(caracteristicas_lbp)
    caracteristicas_vgg16 = extrair_caracteristicas_vgg16(image_block)
    features_vgg16.append(caracteristicas_vgg16)
  #cria csv das caracteristicas
  df_lbp = pd.DataFrame(features_lbp)
  cols = list(df_lbp.columns)
```

```
  cols[0] = 'label'
  df_lbp.columns = cols
  df_lbp.to_csv(f'/content/Test_4cl_amostra/caracteristicas_lbp_{tipo}.csv',
index=False)

  df_vgg16 = pd.DataFrame(features_vgg16)
  df_vgg16['label'] = labels_list

df_vgg16.to_csv(f'/content/Test_4cl_amostra/caracteristicas_vgg16_{tipo}.csv'
, index=False)

calcula_caracteristicas("/content/Test_4cl_amostra/train/treino.txt",
'treino')
calcula_caracteristicas("/content/Test_4cl_amostra/test/teste.txt", 'teste')
# svm lbp
# x sempre e a matriz de caracteristicas
# y sempre e o vetor de classes
print ("Loading data...")
X, y = load_svmlight_file("/content/Test_4cl_amostra/lbp_riu_8_1_treino.txt")

X_test, y_test =
load_svmlight_file("/content/Test_4cl_amostra/lbp_riu_8_1_teste.txt")
#print("----------------\n", X_test)

# Criando uma instância do SVM
clf = svm.SVC(kernel='linear')

# Treinando o SVM
clf.fit(X, y)

# Obtendo o kernel utilizado
kernel_used = clf.kernel
print("Kernel utilizado: ", kernel_used)

# Fazendo previsões nos dados de teste
y_pred = clf.predict(X_test)

# Calculando a precisão do modelo
accuracy = accuracy_score(y_test, y_pred)
print("Precisão do modelo: %.4f" % accuracy)
```

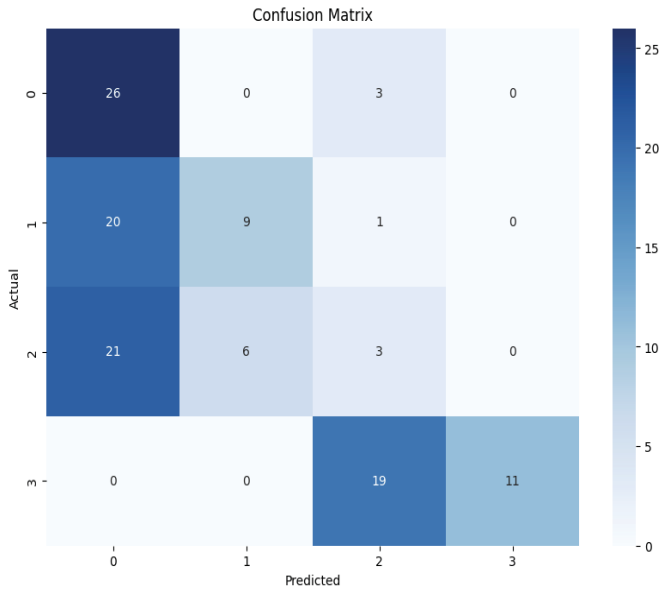
```
# Assume y_true and y_pred are defined here
matrix = confusion_matrix(y_test, y_pred)

labels = ['0', '1', '2', '3']
classification_report(y_test, y_pred, target_names=labels)
print(classification_report(y_test, y_pred, target_names=labels))
pl.figure(figsize=(10,7))
sns.heatmap(matrix, annot=True, fmt="d", cmap='Blues', xticklabels=labels,
yticklabels=labels)
pl.xlabel('Predicted')
pl.ylabel('Actual')
pl.title('Confusion Matrix')
pl.show()
```

Resultados:

Kernel utilizado: linear

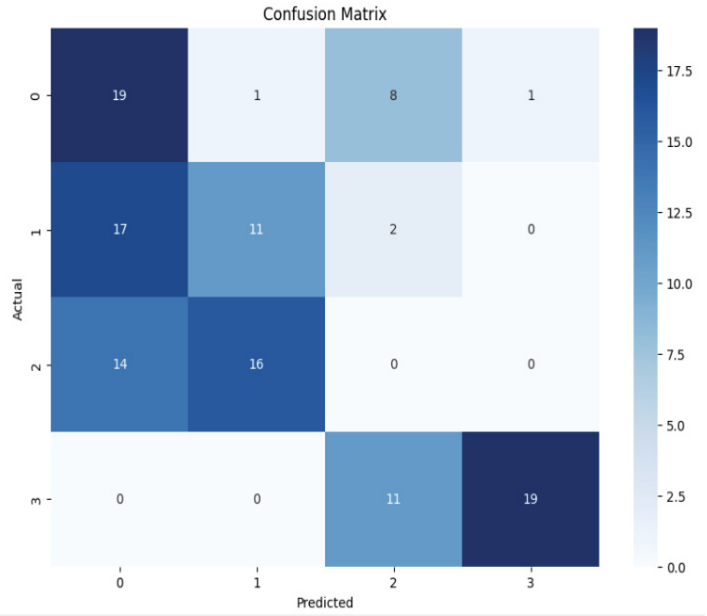
Precisão do modelo: 0.4118

	precision	recall	f1-score	support
0	0.39	0.90	0.54	29
1	0.60	0.30	0.40	30
2	0.12	0.10	0.11	30
3	1.00	0.37	0.54	30
accuracy			0.41	119
macro avg	0.53	0.42	0.40	119
weighted avg	0.53	0.41	0.40	119



Random Forest Accuracy: 0.4117647058823529

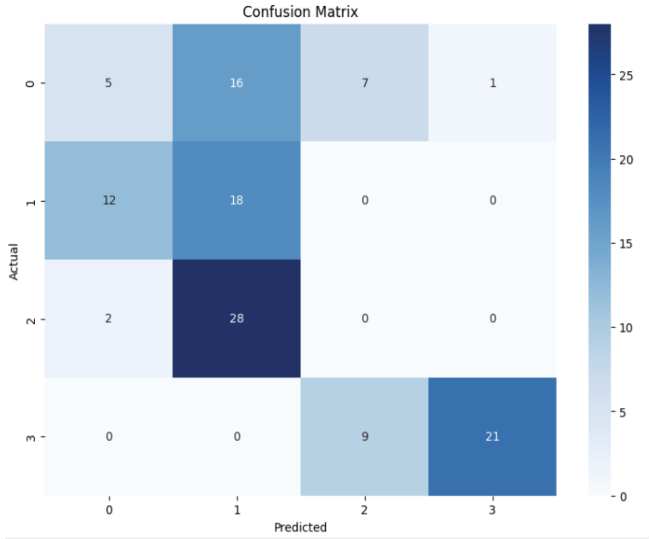
	precision	recall	f1-score	support
0	0.38	0.66	0.48	29
1	0.39	0.37	0.38	30
2	0.00	0.00	0.00	30
3	0.95	0.63	0.76	30
accuracy			0.41	119
macro avg	0.43	0.41	0.41	119
weighted avg	0.43	0.41	0.40	119



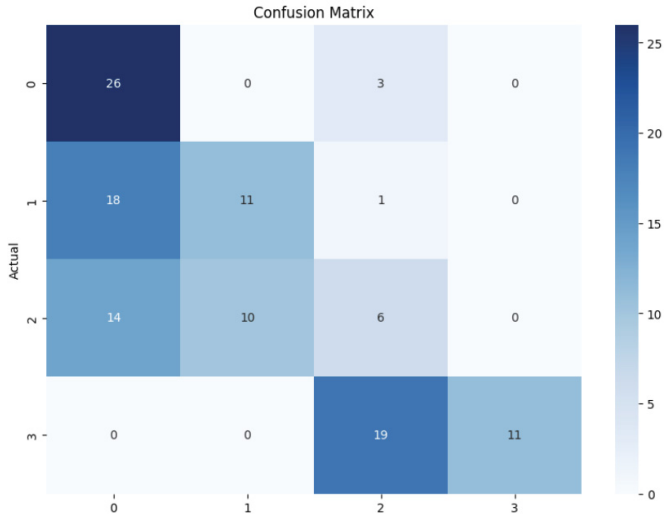
MLP Accuracy: 0.3697478991596639

	precision	recall	f1-score	support
0	0.26	0.17	0.21	29
1	0.29	0.60	0.39	30

2 0.00 0.00 0.00 30
3 0.95 0.70 0.81 30
accuracy 0.37 119
macro avg 0.38 0.37 0.35 119
weighted avg 0.38 0.37 0.35 119



Precisão do modelo: 0.4538
precision recall f1-score support
0 0.45 0.90 0.60 29
1 0.52 0.37 0.43 30
2 0.21 0.20 0.20 30
3 1.00 0.37 0.54 30
accuracy 0.45 119
macro avg 0.54 0.46 0.44 119
weighted avg 0.55 0.45 0.44 119



Indique qual modelo dá o melhor o resultado e a métrica utilizada:

A metrica escolhida foi a Recall, por se tratar de cancer acreditamos que é melhor termos mais FALSOS POSITIVOS e EVITAR ao máximo os FALSOS NEGATIVOS porque se a pessoa não for diagnosticada e de fato tiver câncer, seu tratamento pode ser prejudicado e a recuperação ser muito mais difícil.

Então considerando o **recall** treinado com as caracteristicas extraidas via **LBP** o melhor modelo seria o **SVM** com 0.42 e para as caracteristicas extraidas com **vgg16** o melhor modelo também seria o **SVM** com 0.46.

2 Redes Neurais

 # Recortes do Código

```
# Treinando com Data Augmentation
steps_per_epoch = traingen.samples // BATCH_SIZE
val_steps = validgen.samples // BATCH_SIZE
n_epochs = 10
optimizer = RMSprop(learning_rate=0.0001)
model_resnet_tl.compile(loss='categorical_crossentropy', optimizer=optimizer,
metrics=['accuracy'])
# Salva o modelo Keras após cada época, porém só o de melhor resultado
checkpointer =
ModelCheckpoint(filepath='img_model_resnet_tl.weights.best.keras',
                                verbose=1,
                                save_best_only=True)
# Para o treinamento para prevenir o overfitting
# Não utilizei aqui, pois queria que rodasse todas as 10 épocas
early_stop = EarlyStopping(monitor='val_loss',
                           patience=10,
                           restore_best_weights=True,
                           mode='min')
# Treinamento do Modelo
history_tl = model_resnet_tl.fit(traingen,
                    epochs=n_epochs,
                    steps_per_epoch=steps_per_epoch,
                    validation_data=validgen,
                    validation_steps=val_steps,
                    callbacks=[checkpointer, PlotLossesKeras()],
                    #callbacks=[early_stop, checkpointer, PlotLossesKeras()],
                    verbose=False)
# Generate predictions
```

```
model_resnet_tl.load_weights('img_model_resnet_tl.weights.best.keras') #
initialize the best trained weight
true_classes = testgen.classes
class_indices = traingen.class_indices
class_indices = dict((v,k) for k,v in class_indices.items())
preds_res_tl = model_resnet_tl.predict(testgen)
pred_classes_res_tl = np.argmax(preds_res_tl, axis=1)
acc_0 = accuracy_score(true_classes, pred_classes_res_tl)
print("Acurácia Modelo ResNet50 com Data augmentation: {:.2f}%".format(acc_0
* 100))
```

resultados:

accuracy ResNet50 sem Data augmentation

training (min: 0.000, max: 1.000, cur: 0.000)

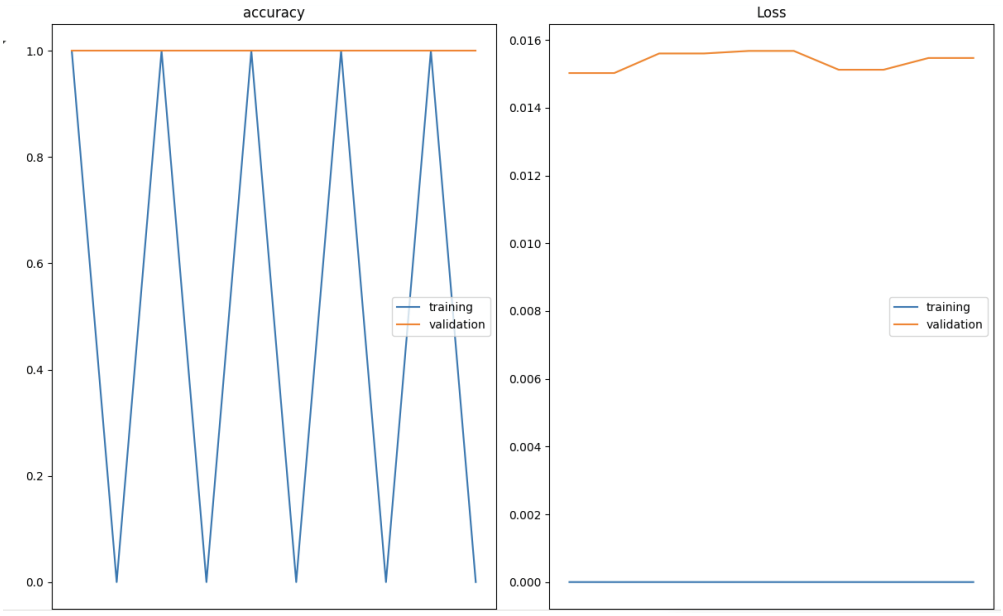
validation (min: 1.000, max: 1.000, cur: 1.000)

Loss

training (min: 0.000, max: 0.000, cur: 0.000)

validation (min: 0.015, max: 0.016, cur: 0.015)

Acurácia Modelo ResNet50 sem Data augmentation: 98.32%



accuracy ResNet50 com Data augmentation

training (min: 0.268, max: 0.800, cur: 0.800)

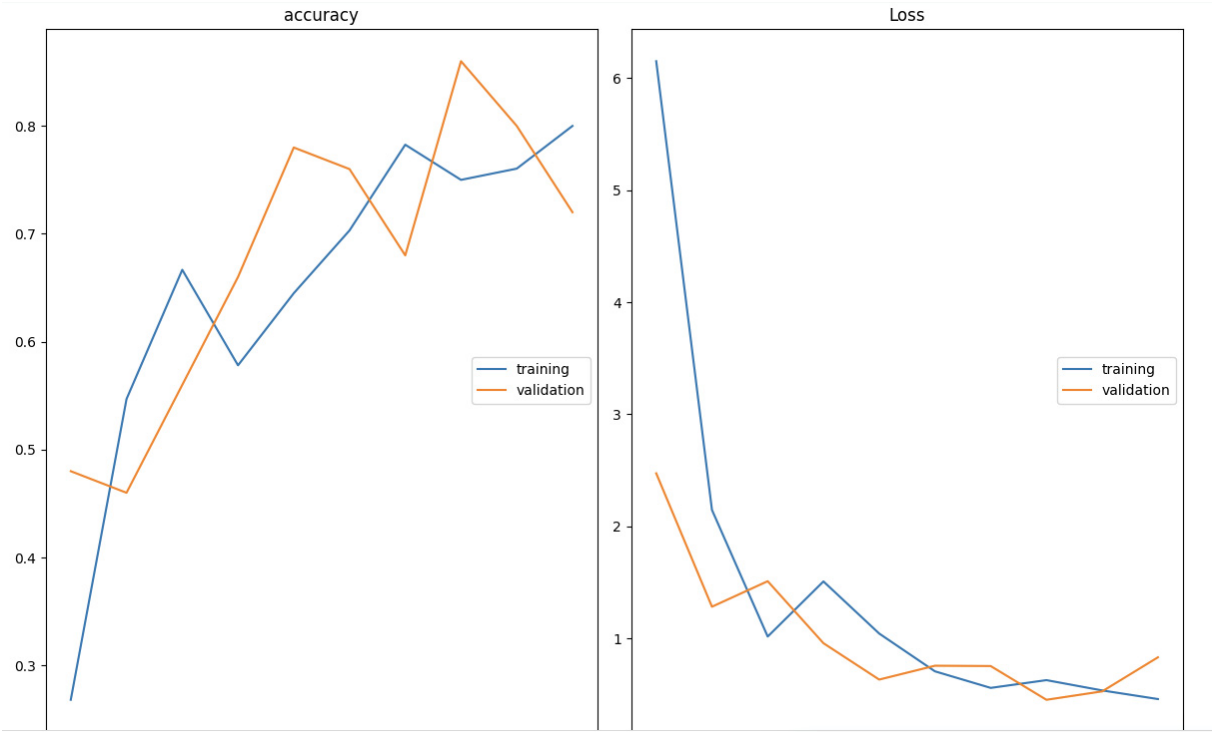
validation (min: 0.460, max: 0.860, cur: 0.720)

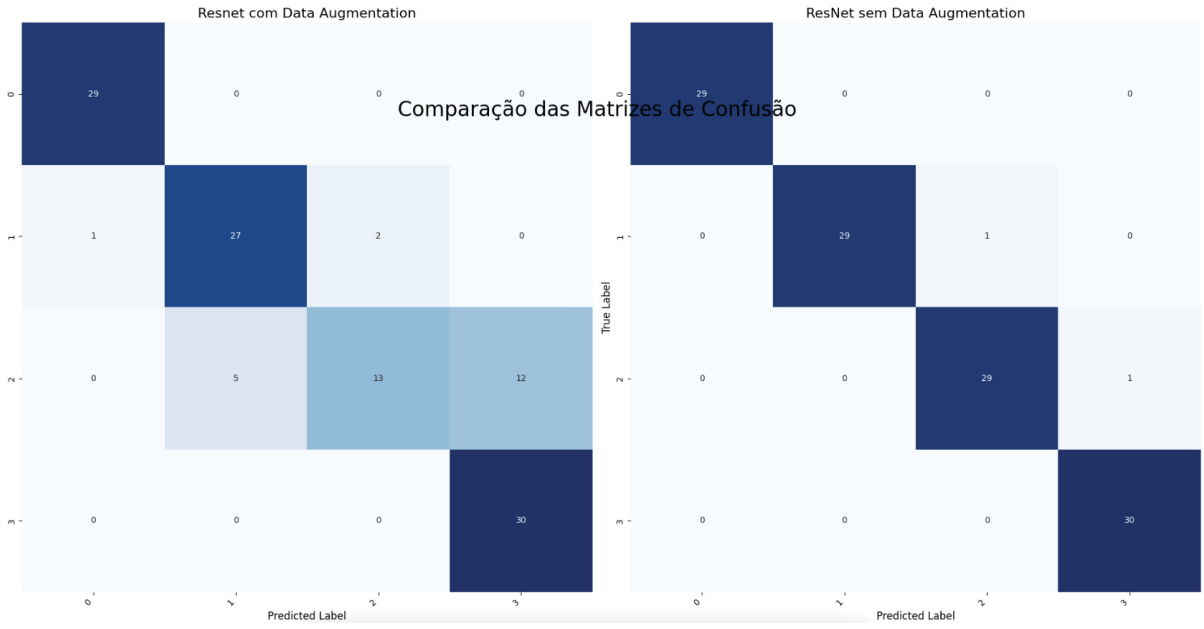
Loss

training (min: 0.459, max: 6.151, cur: 0.459)

validation (min: 0.452, max: 2.473, cur: 0.831)

Acurácia Modelo ResNet50 com Data augmentation: 83.19%





Metricas com Data Augmentation

	precision	recall	f1-score	support
0	0.97	1.00	0.98	29
1	0.84	0.90	0.87	30
2	0.87	0.43	0.58	30
3	0.71	1.00	0.83	30
accuracy			0.83	119
macro avg	0.85	0.83	0.82	119
weighted avg	0.85	0.83	0.81	119

Metricas sem Data Augmentation

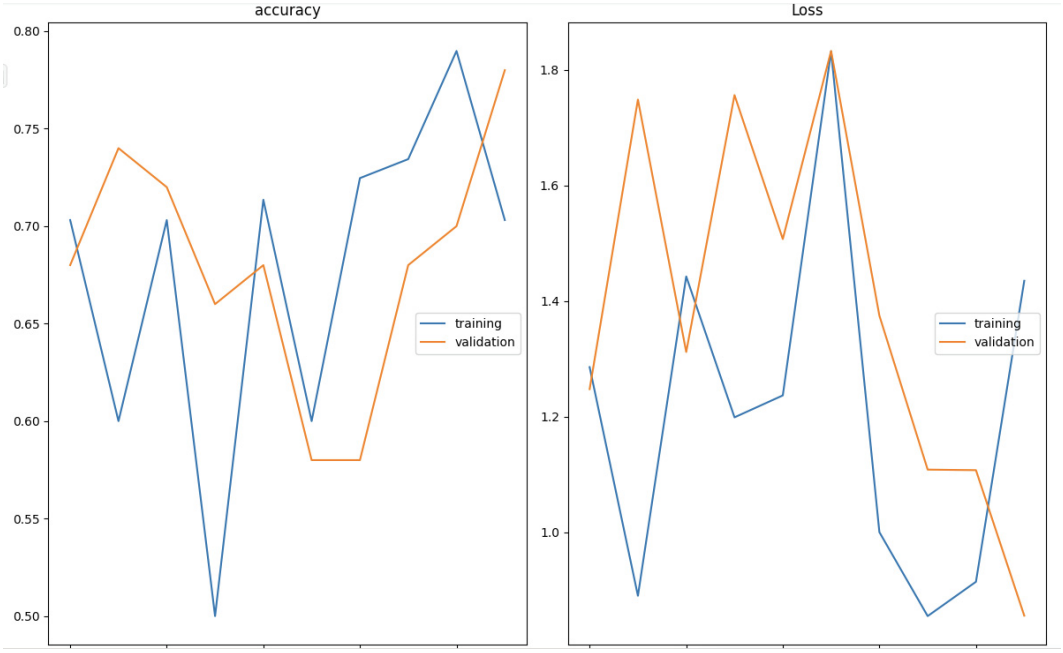
	precision	recall	f1-score	support
0	1.00	1.00	1.00	29
1	1.00	0.97	0.98	30
2	0.97	0.97	0.97	30
3	0.97	1.00	0.98	30
accuracy			0.98	119
macro avg	0.98	0.98	0.98	119
weighted avg	0.98	0.98	0.98	119

Vgg16 Com transfer Learn

accuracy
- training (min: 0.500, max: 0.790, cur: 0.703)
- validation (min: 0.580, max: 0.780, cur: 0.780)

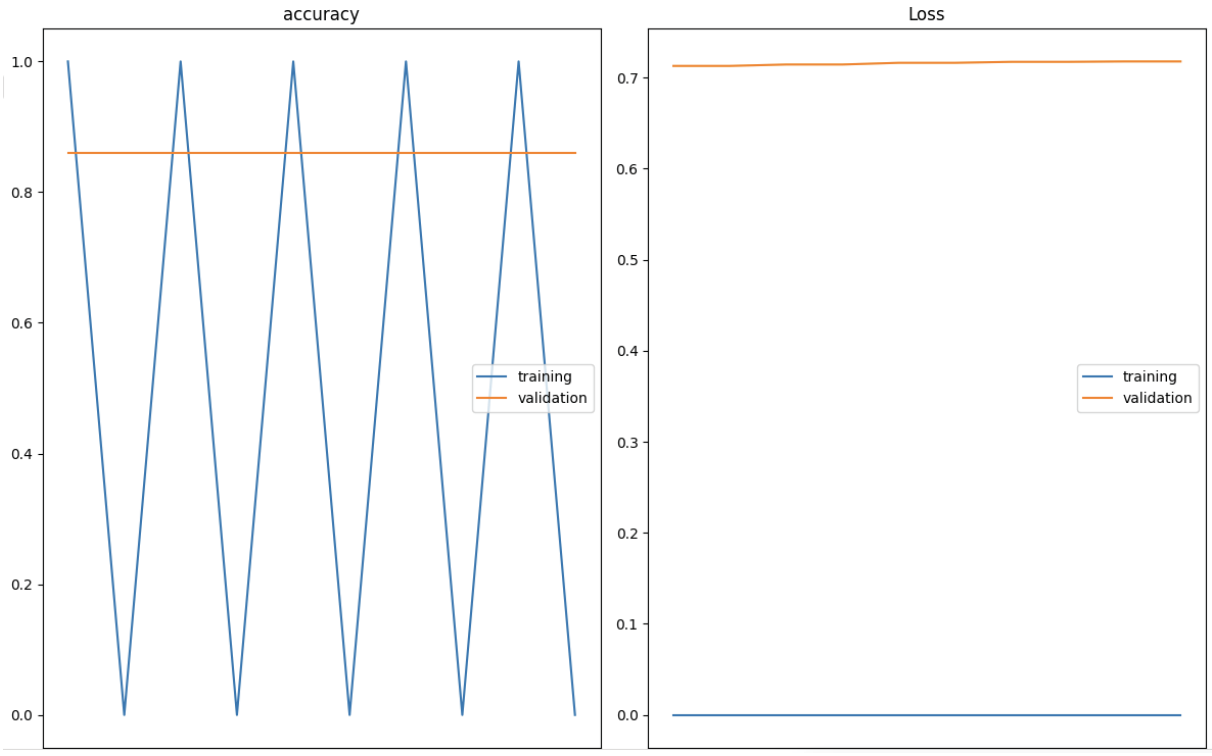
Loss
- training (min: 0.855, max: 1.832, cur: 1.435)
- validation (min: 0.856, max: 1.833, cur: 0.856)

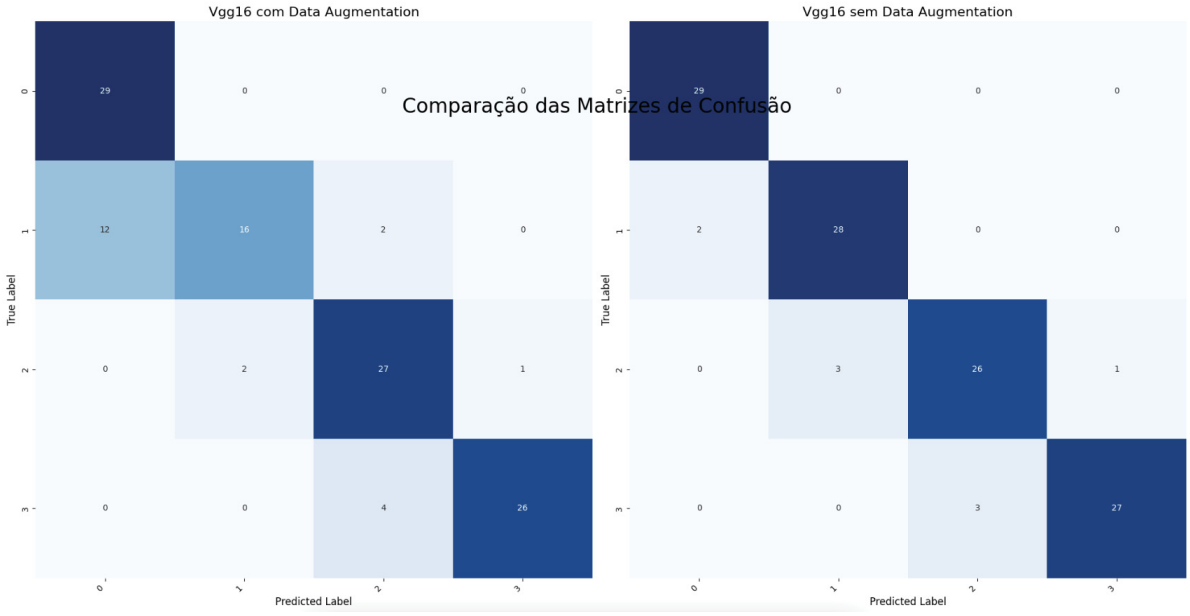
Acurácia Modelo VGG16 Com Data augmentation: 83.19%



accuracy
- training (min: 0.000, max: 1.000, cur: 0.000)

validation (min: 0.860, max: 0.860, cur: 0.860)
Loss
training (min: 0.000, max: 0.000, cur: 0.000)
validation (min: 0.713, max: 0.718, cur: 0.718)
Acurácia Modelo ResNet50 sem Data augmentation: 92.44%





Metricas com Data Augmentation

	precision	recall	f1-score	support
0	0.97	1.00	0.98	29

1 0.84 0.90 0.87 30
2 0.87 0.43 0.58 30
3 0.71 1.00 0.83 30
accuracy 0.83 119
macro avg 0.85 0.83 0.82 119
weighted avg 0.85 0.83 0.81 119
Metricas sem Data Augmentation
precision recall f1-score support
0 1.00 1.00 1.00 29
1 1.00 0.97 0.98 30
2 0.97 0.97 0.97 30
3 0.97 1.00 0.98 30
accuracy 0.98 119
macro avg 0.98 0.98 0.98 119
weighted avg 0.98 0.98 0.98 119

Indique qual modelo dá o melhor o resultado e a métrica utilizada:

A métrica escolhida foi a Recall, por se tratar de câncer acreditamos que é melhor termos mais FALSOS POSITIVOS e EVITAR ao máximo os FALSOS NEGATIVOS porque se a pessoa não for diagnosticada e de fato tiver câncer, seu tratamento pode ser prejudicado e a recuperação ser muito mais difícil.

Então considerando o recall o treinamento usando o modelo vgg16 e resnet sem data augmentation tiveram o melhor resultado, ambos os resultados foram iguais, acredito que pela pouca quantidade de imagens.

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: hfps://bibliotecas.ufpr.br/wp-content/uploads/2022/03/template-artigo-de-periodico.docx

B – RESOLUÇÃO

Estudo de Caso: Implicações Éticas do Uso do ChatGPT

RESUMO: A inteligência artificial (IA) tem transformado diversos setores da sociedade, como a educação, a política, a tecnologia, a saúde e as relações pessoais. Uma das aplicações de IA com impactos recentes na sociedade é o ChatGPT, um modelo de IA avançado desenvolvido pela OpenAI. O ChatGPT é capaz de gerar textos coerentes e contextualmente relevantes, facilitando a comunicação e a automação de tarefas. No entanto, o uso dessa tecnologia levanta uma série de questões éticas que precisam ser cuidadosamente consideradas. Este trabalho aborda questões éticas relacionadas com o uso da IA, especialmente no contexto ChatGPT, tendo como objetivo geral identificar dilemas éticos significativos, como a disseminação de informações falsas, a privacidade dos usuários, a tomada de decisões automatizadas, vieses algorítmicos e discriminação. A metodologia utilizada inclui uma análise crítica desses dilemas, avaliando como eles afetam usuários, desenvolvedores e a sociedade em geral. Considerando a abordagem dos desafios éticos este estudo propõe soluções responsáveis e éticas para lidar com esses dilemas. As propostas incluem sugestões para políticas, regulamentações e práticas de *design* que promovam transparência, justiça e privacidade. Os resultados indicam que, embora o ChatGPT ofereça benefícios significativos, é crucial abordar os desafios éticos associados ao seu uso para garantir que a tecnologia seja desenvolvida e aplicada de maneira responsável e benéfica para todos.
Palavras-chave: Inteligência Artificial, ChatGPT, Ética

ABSTRACT: Artificial intelligence (AI) has been transforming various sectors of society, such as education, politics, technology, healthcare, and personal relationships. One of the AI applications with recent impacts on society is ChatGPT, an advanced AI model developed by OpenAI. ChatGPT can generate coherent and contextually relevant texts, facilitating communication and task automation. However, the use of this technology raises a series of ethical issues that need to be carefully considered. This work addresses ethical issues related to the use of AI, especially in the context of ChatGPT, with the general objective of identifying significant ethical dilemmas, such as the dissemination of false information, user privacy, automated decision-making, algorithmic biases, and discrimination. The methodology used includes a critical analysis of these dilemmas, assessing how

they affect users, developers, and society in general. Considering the approach to ethical challenges, this study proposes responsible and ethical solutions to address these dilemmas. The proposals include suggestions for policies, regulations, and design practices that promote transparency, fairness, and privacy. The results indicate that, although ChatGPT offers significant benefits, it is crucial to address the ethical challenges associated with its use to ensure that the technology is developed and applied in a responsible and beneficial manner for everyone.
Keywords: Artificial Intelligence, ChatGPT, Ethics

1 INTRODUÇÃO

A inteligência artificial (IA) tem suas raízes na metade do século XX, quando cientistas começaram a explorar a ideia de máquinas capazes de simular processos de pensamento humano. Alan Turing, com seu famoso "Teste de Turing", e John McCarthy, que cunhou o termo "inteligência artificial" em 1956, são figuras fundamentais na história da IA (HUANG, 2010). Desde então, a IA evoluiu significativamente, passando de programas simples de xadrez e resolução de problemas para sistemas avançados capazes de aprender, adaptar-se e interagir com seres humanos de maneiras complexas.

Recentemente os avanços em aprendizado de máquina e processamento de linguagem natural levaram ao desenvolvimento de modelos como o ChatGPT, criado pela OpenAI. O ChatGPT é uma aplicação sofisticada que pode gerar textos coerentes e contextualmente relevantes, auxiliando na comunicação e automação de tarefas em diversos setores, como educação, tecnologia, saúde e até relações pessoais (BROWN et al., 2020). No entanto, com essas capacidades avançadas, surgem também questões éticas significativas.

A utilização de IA, especialmente modelos como o ChatGPT, levanta preocupações éticas em várias frentes. A disseminação de informações falsas, a privacidade dos usuários, a tomada de decisões automatizadas, os vieses algorítmicos e a discriminação são alguns dos principais dilemas que precisam ser abordados. Esses desafios éticos não apenas afetam os usuários finais, mas também os desenvolvedores e a sociedade em geral. Por exemplo, a capacidade do ChatGPT de gerar textos pode ser usada tanto para educar quanto para desinformar, dependendo de como é implementada e controlada (JOBIN; IENCA; VAYENA, 2019).

O objetivo deste trabalho é abordar as implicações éticas do uso do ChatGPT em diferentes contextos, identificando dilemas éticos significativos e propondo soluções responsáveis para lidar com esses desafios. A metodologia utilizada inclui uma análise crítica dos dilemas éticos no contexto do uso da IA. O estudo conta com uma reflexão que permite uma compreensão dos impactos e possíveis soluções para os dilemas identificados.

O trabalho está organizado da seguinte maneira: inicialmente, serão discutidas as questões éticas significativas e como elas afetam diferentes partes interessadas; em seguida, será realizada uma análise crítica das implicações éticas utilizando estudos de caso; posteriormente, serão propostas soluções responsáveis e éticas para os dilemas identificados; e, por fim, o trabalho será concluído com uma síntese dos achados e recomendações para futuras pesquisas.

Com essa estrutura, espera-se fornecer uma visão abrangente das questões éticas associadas ao uso do ChatGPT, destacando a importância de abordagens responsáveis para garantir que os benefícios dessa tecnologia sejam maximizados, enquanto os riscos são minimizados.

2 DESENVOLVIMENTO

2.1 Questões Éticas que Impactam as Partes Interessadas

O advento do ChatGPT representa um marco significativo na evolução da IA oferecendo capacidades avançadas de geração de texto que têm sido aplicadas em uma variedade de contextos, desde assistentes virtuais até ferramentas de automação de processos. No entanto, junto com os benefícios proporcionados por essa tecnologia, surgem preocupações éticas substanciais que afetam diretamente várias partes interessadas.

Uma das questões éticas mais prementes é a disseminação de informações falsas. O ChatGPT, por sua habilidade de gerar texto de maneira autônoma, pode ser explorado para disseminar desinformação deliberada. Este problema não é apenas uma questão de precisão da informação, mas também de confiança pública e integridade democrática. A disseminação de informações falsas pode influenciar negativamente decisões individuais e coletivas, impactando desde o comportamento do consumidor até os processos eleitorais. Esse desafio coloca um ônus significativo sobre os desenvolvedores, que precisam implementar estratégias eficazes para identificar e mitigar a propagação de conteúdo enganoso (CHORAŚ et al., 2021).

A privacidade dos usuários também é uma área de preocupação crítica. Ao interagir com o ChatGPT, os usuários podem inadvertidamente compartilhar informações pessoais sensíveis. A coleta e o armazenamento desses dados levantam questões sobre transparência, consentimento informado e proteção contra o uso indevido. A falta de clareza nessas áreas pode levar a violações de privacidade que minam a confiança dos usuários na tecnologia de IA em geral. Para os desenvolvedores, isso significa a necessidade de adotar políticas de privacidade rigorosas e implementar medidas de segurança robustas que garantam a proteção dos dados do usuário em todas as fases da interação com o sistema.

O ChatGPT, como qualquer sistema de IA baseado em aprendizado de máquina, é treinado em grandes conjuntos de dados que podem refletir e amplificar preconceitos existentes na sociedade. Sendo assim, os vieses algorítmicos representam um desafio ético significativo. Isso pode resultar em respostas discriminatórias que perpetuam desigualdades sociais, como viés de gênero, racial ou econômico. Os desenvolvedores enfrentam o desafio de identificar e mitigar esses vieses por meio de práticas de design inclusivas, revisões rigorosas dos conjuntos de dados de treinamento e testes extensivos para garantir que o sistema seja justo e equitativo em suas interações com usuários de diferentes origens e contextos (LI et al., 2023).

2.2 Propostas de Soluções Responsáveis e Éticas

Para abordar esses dilemas éticos de maneira eficaz, é crucial desenvolver e implementar soluções responsáveis e éticas que promovam o uso seguro e benéfico do ChatGPT e outras tecnologias de IA.

Uma abordagem fundamental é o estabelecimento de políticas e regulamentações claras que governem o desenvolvimento e a implementação de sistemas de IA. Isso inclui a criação de diretrizes robustas para a obtenção, o uso e o compartilhamento de dados pessoais e corporativos, garantindo transparência e consentimento informado por parte dos usuários. Regulamentações governamentais também são essenciais para garantir que as empresas sejam responsabilizadas por práticas inadequadas, incentivando práticas éticas e promovendo a confiança pública na tecnologia de IA.

As práticas de *design* ético desempenham um papel crucial na mitigação de vieses algorítmicos e na promoção da equidade nos sistemas de IA. Isso envolve desde a inclusão de diversidade nos conjuntos de dados de treinamento até a implementação de mecanismos de auditoria e monitoramento contínuo para identificar e corrigir vieses potenciais. Integrar feedback dos usuários de forma sistemática também é fundamental, permitindo ajustes iterativos que melhorem a precisão e a equidade do sistema ao longo do tempo (HOLSTEIN et al., 2019).

Ao equilibrar os interesses das partes interessadas, como usuários, desenvolvedores e sociedade em geral, é possível promover um uso mais responsável e ético do ChatGPT e outras tecnologias de IA. Essas soluções não apenas mitigam os riscos associados ao uso da tecnologia, mas também fortalecem a confiança pública, facilitando uma adoção mais ampla e sustentável de inovações que têm o potencial de transformar positivamente diversos aspectos da vida cotidiana e organizacional (LI et al., 2023).

2.2.1 Abordagem das questões éticas da Política ACT da União Europeia sobre IA

A política de Cooperação Europeia em Inteligência Artificial (AI *Coordinated Plan for Trustworthy* AI - ACT) adotada pela União Europeia aborda uma série de questões éticas cruciais para promover o uso responsável e ético da inteligência artificial na região. Este plano abrange diversos aspectos que visam garantir que a IA desenvolvida e implementada na UE respeite princípios éticos fundamentais e promova o bem-estar dos cidadãos europeus (HACKER; CORDES; ROCHON, 2024). Algumas das questões éticas abordadas incluem:

2.2.1.1 Transparência e Explicabilidade

A política ACT enfatiza a importância da transparência no desenvolvimento e na implementação de sistemas de IA. Isso inclui a exigência de que sistemas de IA sejam compreensíveis e explicáveis, garantindo que os usuários entendam como as decisões são tomadas e quais critérios são utilizados pelos algoritmos. A transparência é essencial para promover a confiança pública na IA e para permitir a responsabilização adequada em caso de decisões adversas ou impactos negativos (FLORIDI et al., 2018).

2.2.1.2 Privacidade e Proteção de Dados

A política da UE coloca um forte foco na proteção da privacidade e dos dados pessoais dos cidadãos europeus. Ela exige que os sistemas de IA respeitem totalmente os direitos à privacidade e ao controle sobre os dados pessoais, conforme estabelecido no Regulamento Geral de Proteção de Dados (GDPR). Isso inclui a implementação de medidas técnicas e organizacionais robustas para garantir que os dados sejam coletados e processados de maneira legal, ética e segura (CHIN; ZHAO, 2022).

2.2.1.3 Equidade e Não Discriminação

A política ACT da UE proíbe explicitamente a criação ou utilização de sistemas de IA que perpetuem ou ampliem preconceitos ou discriminações existentes. Ela busca garantir que os sistemas de IA sejam desenvolvidos de forma a promover a equidade e a inclusão, minimizando vieses algorítmicos e garantindo tratamento justo para todos os indivíduos, independentemente de características como raça, gênero, orientação sexual ou origem étnica (FLORIDI et al., 2018).

2.2.1.4 Segurança e Responsabilidade

A segurança dos sistemas de IA é uma preocupação central na política da UE. Ela estabelece requisitos rigorosos para garantir que os sistemas de IA sejam robustos, seguros e resilientes contra-ataques cibernéticos e manipulações maliciosas. A política enfatiza a necessidade de atribuição clara de responsabilidades, garantindo que os desenvolvedores e operadores de sistemas de IA sejam responsáveis por eventuais danos causados por suas tecnologias (CHRISTIDIS; DEVETSIKIOTIS, 2016).

2.2.1.5 Supervisão e Governança

A política ACT promove a criação de estruturas de supervisão e governança eficazes para monitorar o uso de IA na UE. Isso inclui a criação de autoridades regulatórias independentes e a implementação de mecanismos de monitoramento contínuo para avaliar a conformidade com os princípios éticos e regulamentações estabelecidas. A supervisão adequada é essencial para garantir que os sistemas de IA operem dentro de limites éticos e legais, promovendo a confiança pública e a aceitação da tecnologia (CATH et al., 2018).

3 CONCLUSÃO

A evolução da IA tendo como foco o ChatGPT da OpenAI, tem demonstrado o potencial transformador em diversas esferas da sociedade, oferecendo benefícios significativos em comunicação, automação e tomada de decisões. No entanto, junto com esses avanços surgem desafios éticos complexos que precisam ser abordados de maneira responsável e proativa.

Este estudo explorou diversas questões éticas críticas relacionadas ao uso do ChatGPT, destacando preocupações como a disseminação de informações falsas, a privacidade dos usuários, vieses algorítmicos e a necessidade de uma supervisão regulatória eficaz. Cada uma dessas áreas

apresenta desafios únicos que impactam não apenas os usuários finais, mas também os desenvolvedores e a integridade democrática.

Considerando essas questões o trabalho aborda diretrizes para a obter, usar e compartilhar os dados pessoais e corporativos, garantindo transparência e consentimento informado por parte dos usuários, com soluções éticas e responsáveis para enfrentar os dilemas éticos, baseados em princípios de transparência, equidade e segurança. A implementação de políticas claras e regulamentações robustas, como as delineadas na política ACT da União Europeia, desempenha um papel crucial na garantia de que a IA seja desenvolvida e utilizada de maneira ética. A ACT enfatiza a importância da transparência, privacidade, equidade e responsabilidade, estabelecendo um padrão para práticas éticas que respeitem os direitos fundamentais dos indivíduos.

Ao equilibrar os interesses das partes interessadas e promover uma governança eficaz, podemos mitigar os riscos associados ao uso da tecnologia enquanto maximizamos seus benefícios potenciais. Isso não apenas fortalece a confiança pública na inteligência artificial, mas também facilita uma adoção mais ampla e sustentável de inovações que possam melhorar significativamente diversos aspectos da vida e da sociedade. Para tanto, o caminho para um futuro ético e responsável em IA exige uma colaboração contínua entre governos, empresas, acadêmicos e sociedade civil para garantir que os avanços tecnológicos sejam orientados pelos mais altos padrões éticos, beneficiando a todos de maneira equitativa e justa.

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

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

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

Qual o objetivo do estudo descrito pelo artigo?

O estudo visa refletir sobre arquitetura de software especificamente voltada para sistemas de aprendizado de máquina, a partir do ponto de vista de pesquisadores e profissionais do ramo.

Três foram as principais questões norteadoras do estudo:

1) Quais são os desafios mais comuns de design de arquitetura de sistemas de aprendizado de máquina?

2) Quais são as melhores práticas na arquitetura de sistemas de aprendizado de máquina?

3) Quais são as principais decisões de design de arquitetura de sistemas de aprendizado de máquina?

Qual o problema/oportunidade/situação que levou a necessidade de realização deste estudo?

Soluções baseadas no aprendizado de máquina tem sido usadas em muitos campos, incluindo sistemas autônomos, biologia computacional, sistemas de recomendação, robótica, Internet das Coisas, etc.

Com essa crescente demanda, pesquisadores e especialistas veem investigando as melhores práticas de design (práticas recomendadas) no projeto de arquitetura de software para esses sistemas.

Contudo, segundo avaliação dos próprios autores do estudo, ainda faltam estudos que analisem e discutam como profissionais do ramo percebem e usam decisões de design na arquitetura de sistemas de aprendizado de máquina.

A premissa é que identificar e reconhecer decisões, decisões e desafios de design, pode auxiliar no aprimoramento do complexo processo de desenvolvimento de softwares baseados em aprendizado de máquina.

Qual a metodologia que os autores usaram para obter e analisar as informações do estudo?

O estudo empregou primeiramente uma revisão sistemática da literatura, seguido de entrevistas com especialistas.

Na revisão sistemática foi realizada uma busca automática de literatura em quatro bases de dados científicas (IEEE, ACM, SCOPUS e WOS). A partir dessa busca inicial, que identificou 3.038 potenciais estudos, foram aplicados critérios de inclusão e exclusão, bem como o processo "backwards and forward snowballing" para encontrar novos estudos adicionais. Restaram, assim, finalmente, 41 estudos que serem submetidos aos processo de extração, tratamento e análise.

Já na entrevista, foram entrevistados 12 participantes especialistas em aprendizagem de máquina de 9 diferentes países, de diferentes instituições e com tempo variado de experiência (com menos de 1 ano até mais de 10 anos) convidados a partir do LinkedIn e outras redes de contatos dos próprios autores. A entrevista foi conduzida a partir de 15 questões abertas préestabelecidas.

Foram realizadas análises quantitativas e qualitativas combinando análise de conteúdo e síntese narrativa.

Quais os principais resultados obtidos pelo estudo?

Como principais resultados, no que se refere a arquitetura de sistemas de aprendizado de máquina, o estudo mapeou e identificou 35 desafios de design, 42 práticas recomendadas e 27 decisões de design.

Os resultados foram agrupados em 5 principais categorias: Arquitetura, Dados e Modelos.

No que se refere a arquitetura, o uso de padrões ou estilos de arquitetura é geralmente reconhecido como uma prática recomendada, podendo proporcionar vantagens no projeto de sistemas, mas também podem representar desafios únicos.

Sobre os dados, segundo os autores, continuam a existir lacunas críticas (principalmente referentes a dependência dos dados e sua precisão) que precisam ser identificadas e enfrentadas no processo de desenvolvimento.

Já a modelagem enfrenta desafios significativos, especialmente em relação a seleção de modelos e ao controle de versão. A integração de tecnologias de nuvem e do TensorFlow foram consideradas como uma abordagem benéfica para melhorar o desempenho e a escalabilidade dos sistemas de aprendizado de máquina

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_livos.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:

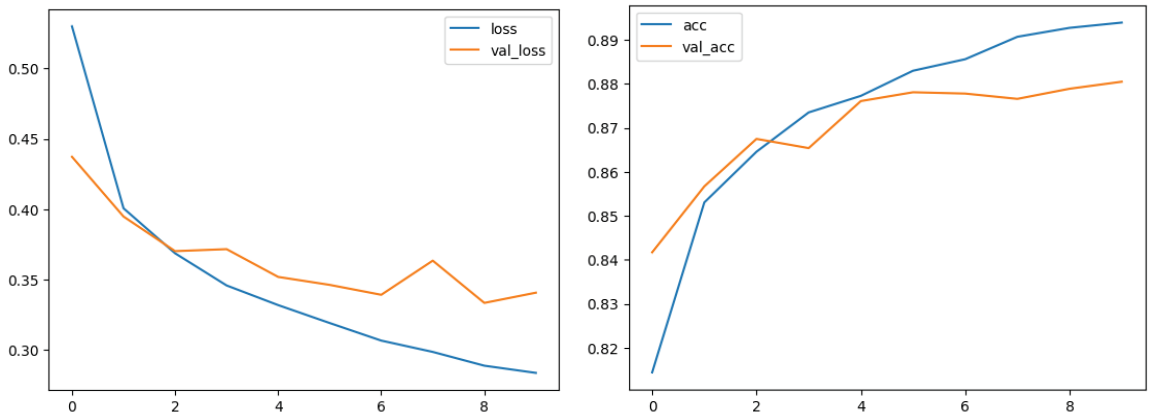
- **Base de dados:** https://commons.wikimedia.org/wiki/File:Felis_catus-cat_on_snow.jpg
- **Importação da imagem**: Copiar código abaixo.

```
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 Classificação (RNA)



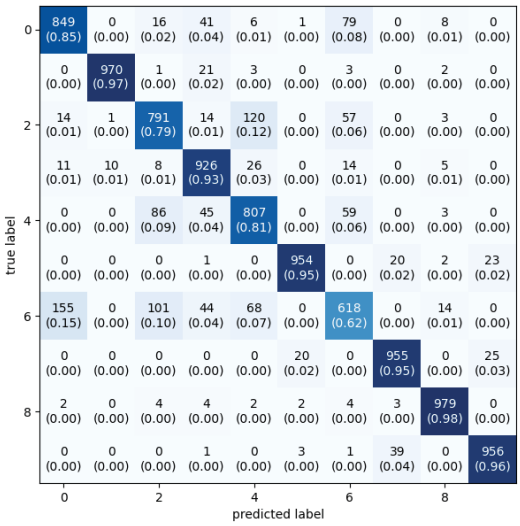
 # Explicação dos Gráficos de Perda

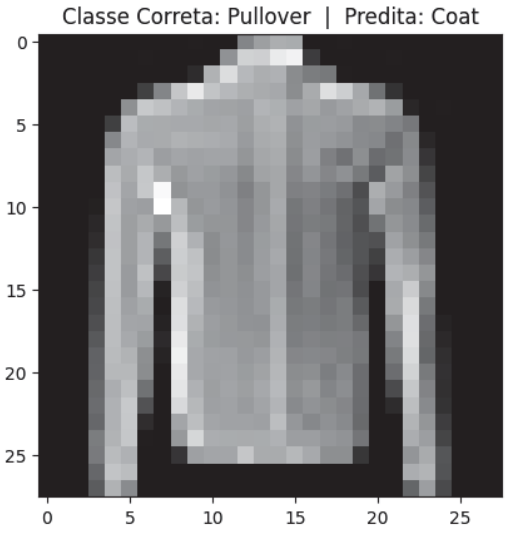
No primeiro gráfico (Gráfico de Perda), é possível ver uma diminuição rápida da perda (loss) nas primeiras épocas de treinamento passando gradualmente a uma diminuição menos intensas nas épocas seguintes, especialmente nos dados de treinamento. Nos dados de validação (val_loss)

também há uma diminuição da perda, porém de maneira menos acentuada. Ao final das 10 épocas, a perda é maior no grupo de validação.

 # Explicação dos Gráficos de Acurácia

No segunda gráfico (Gráfico de Acurácia), é possível verificar um aumento progressivo na acurácia (acc) dos dados de treinamento, com crescimento maior nas primeiras épocas. Também é possível constatar uma melhoria na acurácia dos dados de validação (val_acc), embora com intensidade inferior.

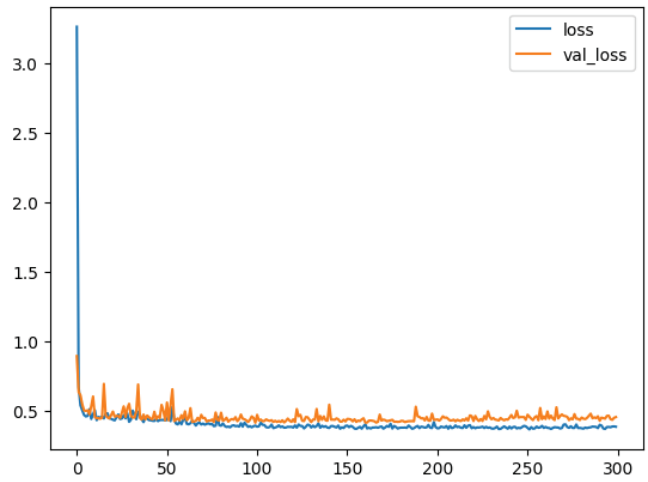




 # Explicação da imagem gerada

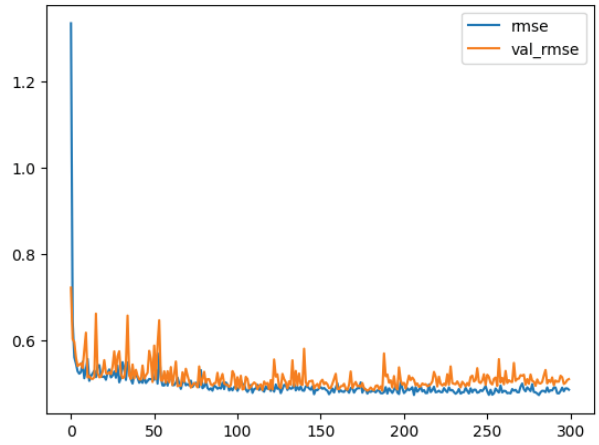
Como é possível verificar acima, o modelo previu "Coat" (Casaco) quando o correto deveria ser "Pullover". Conforme verificado na matriz de confunsão, trata-se de uma das quatorze (1/14) predições erradas envolvendo essas categorias (Pullover erroneamente classificado como Coat)

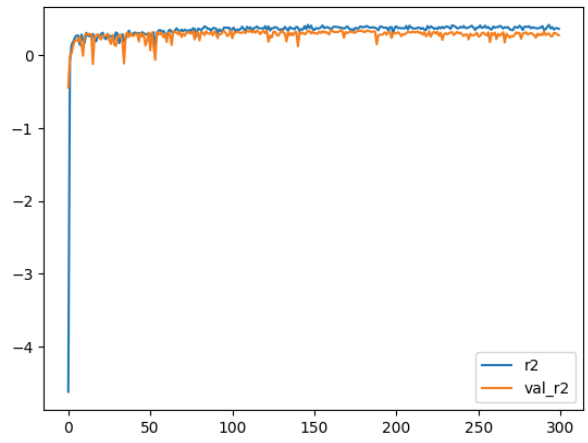
2 Regressão (RNA)



 # Avaliação do Gráfico

Houve uma queda inicial bastante significativa de "loss". Depois esse valor diminui muito pouco (aproximadamente de .5 para .4), mostrando que o modelo não se mostrou adequado.



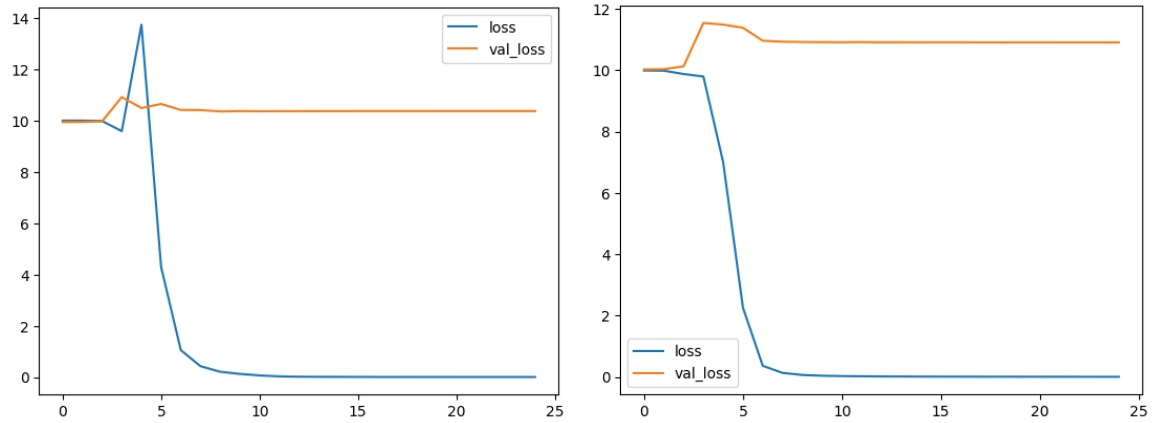


 # Avaliação de métrica do modelo

Conforme foi possível verificar no gráfico anterior, o valor de R2 não apresentou melhora significativa, terminando com míseros 0.28. Isso significa que o modelo utilizado não consegue predizer (por meio da regressão) um valor adequado para a qualidade do vinho.

3 Sistemas de Recomendação

Modelo 1 e Modelo 2:



 # Avaliação e comentários

Foi testado dois modelos diferentes para o sistema de recomendação. O primeiro deles utilizava somente duas variáveis (ISBN e Usuario), enquanto que o segundo utilizava todas as seis variáveis disponíveis (ISBN, Titulo, Autor, Ano, Editor, Usuario).

Como pode ser possível observar nos gráficos de perda, ambos os modelos apresentaram "overfitting", caracterizado pela diminuição da perda (loss) nos dados de treinamento, mas com uma não diminuição dessa perda nos dados de validação.

Ao final do Modelo 1 foi efetuada a recomendação de um livro para um usuário (cat.cod=11121) o resultado foi, conforme verificado, o livro com a cat.cod = "89491", pois este foi o livro com maior score (9.60) para este usuário.

4 Deepdream

Imagem original:

Imagens com “deepdream”:

 # Explicação dos resultados

Foram geradas duas imagens.

Comparando as duas imagens, é verificada que na primeira imagem os padrões aparentemente ocorrem na mesma granularidade, enquanto na segunda imagem esse padrão sofre um pouco de alteração. No caso, a segunda imagem foi redimensionada em 20% (OCTAVE_SCALE= 1.20) a cada interação. Com isso, temos uma imagem com características mais psicodélicas e abastratas, remetendo a ideia de imagem sonhada, onírica.

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 publico-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) O conjunto de dados brutos;

Venda de jogos de video-game, utilizarei um conjunto de dados disponível no Kaggle (https://www.kaggle.com/datasets/gregorut/videogamesales), a base contem os jogos de video-game que venderam mais de 100.000 copias. A base é extensa contendo mais de 16.000 registros. segue abaixo um preview da base:
Abaixo segue um pequeno recorte dos dados brutos.

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
1	Wii Sports	Wii	2006	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
2	Super Mario Bros.	NES	1985	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24

3	Mario Kart Wii	Wii	2008	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
4	Wii Sports Resort	Wii	2009	Sports	Nintendo	15.75	11.01	3.28	2.96	33
5	Pokemon Red/Pokemon Blue	GB	1996	Role-Playing	Nintendo	11.27	8.89	10.22	1	31.37
6	Tetris	GB	1989	Puzzle	Nintendo	23.2	2.26	4.22	0.58	30.26

2) Explicação do contexto e o publico-alvo da visualização de dados e do storytelling que será desenvolvido;

O objetivo principal das visualizações é apresentar de forma clara e concisa, quais são os gêneros de videogames que geraram mais receita em vendas globais ao longo do tempo. Entender essa distribuição ajuda a identificar tendências de mercado, oportunidades e possíveis nichos a serem explorados. Os públicos alvo são:

- Desenvolvedores de Jogos: Buscando entender as tendências de mercado e onde concentrar seus esforços de desenvolvimento.
- Investidores: Interessados em identificar oportunidades lucrativas no mercado de videogames.
- Analistas de Mercado: Estudando o panorama da indústria de jogos..

3) A visualização desses dados explicando a escolha do tipo de visualização e da ferramenta usada;

As ferramentas escolhidas para a criação dos gráficos foram: Python com as bibliotecas Pandas, Matplotlib e Seaborn.

Gráfico de barras: Como os jogos foram agrupados por seus gêneros, o gráfico de barras é eficaz para a comparação dos valores de cada gênero, ordenando os gêneros por quantidade de vendas só de olhar já é possível identificar quais gêneros fazem mais ou menos vendas. Utilizei uma paleta de cores do seaborn chamada colorblind onde a mesma possui cores acessivas para daltonismo.

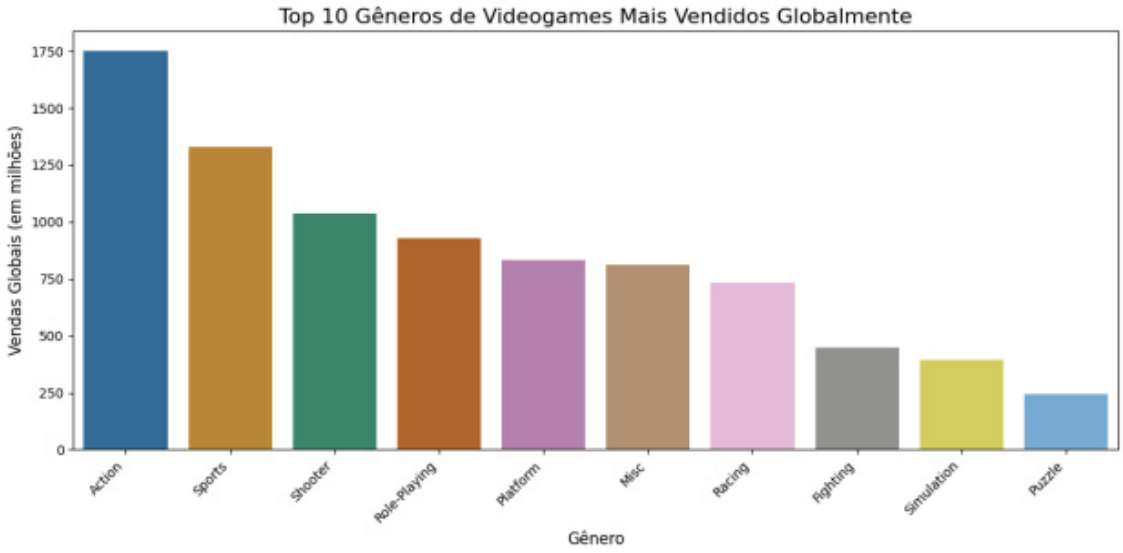
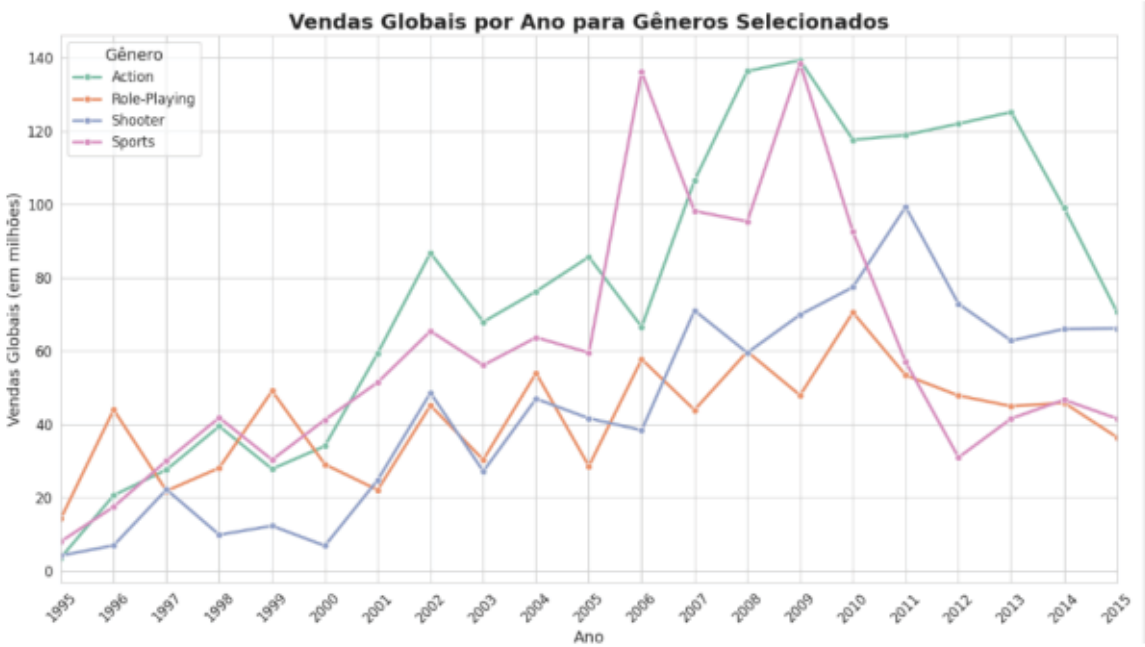


Gráfico de linhas: utilizei o gráfico de linha por ser ótimo para mostrar mudanças ao longo do tempo e facilita a interpretação, pois permite ver o comportamento contínuo dos dados de forma clara. Ele permite identificar se as vendas dos quatro gêneros que mais venderam, aumentaram ou diminuíram, se existiram picos específicos entre 1995 e 2015. Com ele também é possível comprar as vendas dos 4 gêneros, verificar picos de vendas.



4) A descrição da narrativa/storytelling dessa visualização de dados.

O mercado de videogames é vasto e diversificado, com uma variedade de gêneros competindo pela atenção dos jogadores e, consequentemente, pelas suas carteiras. Para desenvolvedores e

investidores, navegar por essa complexidade e identificar oportunidades lucrativas pode ser desafiador. Esta visualização foi criada para simplificar essa jornada, revelando os gêneros que consistentemente dominam as vendas globais.

Análise da Visualização:

O gráfico de barras demonstra, de forma inequívoca, que alguns gêneros se destacam significativamente em termos de vendas. Ao analisarmos a altura das barras, podemos observar que ação é o gênero com maior volume de vendas globais, seguido por esporte e assim por diante. Essa hierarquia nos oferece insights valiosos sobre as preferências dos jogadores em escala global.

O gráfico de linha nos ajuda a verificar como foi a desempenho de venda dos 4 gêneros mais vendidos, no período entre 1995 a 2015, nos trazendo segurança se esses gêneros se mantém com vendas solidas ao passar dos anos ou se existiu algum evento anormal em algum dos anos. Podemos destacar que o gênero de ação consegue se manter como o gênero mais vendido em quase todos os anos e o gênero de shooter vem tendo vendas melhores com o pasar do ano e pode ser uma tendencia de mercado.

O Que os Dados Significam:

Preferências Globais: Os gêneros no topo do ranking representam os gostos predominantes dos jogadores em todo o mundo. Isso não significa que outros gêneros não sejam lucrativos, mas indica que há um público muito maior para esses tipos de jogos. Tendências de Mercado: Observar quais gêneros se mantêm consistentemente no topo ao longo dos anos (gráfico de linhas) pode indicar tendências de longo prazo, onde pode existir um retorno mais seguro do investimento.

Conclusão:

Esta visualização fornecem uma visão geral valiosa do mercado de videogames, destacando os gêneros que dominam as vendas globais. Ao entender essas tendências, desenvolvedores, investidores e analistas de mercado podem tomar decisões mais informadas e estratégicas, maximizando suas chances de sucesso neste mercado dinâmico e em constante evolução.

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

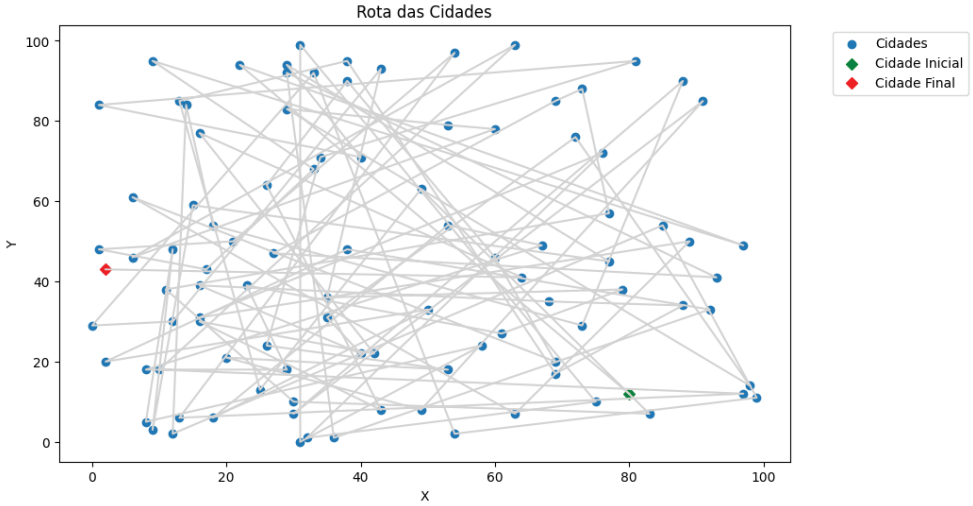
 # Trechos do Código: funções

```
# Gera cromossomos randomicamente para compor a população inicial.
def gerar_cromossomos(qtde_individuos, cidade_inicial, cidade_final):
  populacao = []
  for _ in range(qtde_individuos):
    vetor = list(range(1, 99))  # Gera números de 1 a 98
    random.shuffle(vetor)  # Embaralha os números
    vetor.insert(0, cidade_inicial)  # Insere a cidade inicial na posição 0
    vetor.append(cidade_final)  # Insere a cidade final na posição 99
    populacao.append(vetor)
  return populacao

# calcula a aptidao para cada individuo da população
def calcular_aptidao(populacao):
  custos = []
  for cromossomo in populacao:
    df = reorganiza_df(df_base, cromossomo)
    custo = calcular_distancia(df, cromossomo)
    custos.append(custo)
  return custos

def selecao(populacao,qtde_selecao):
```

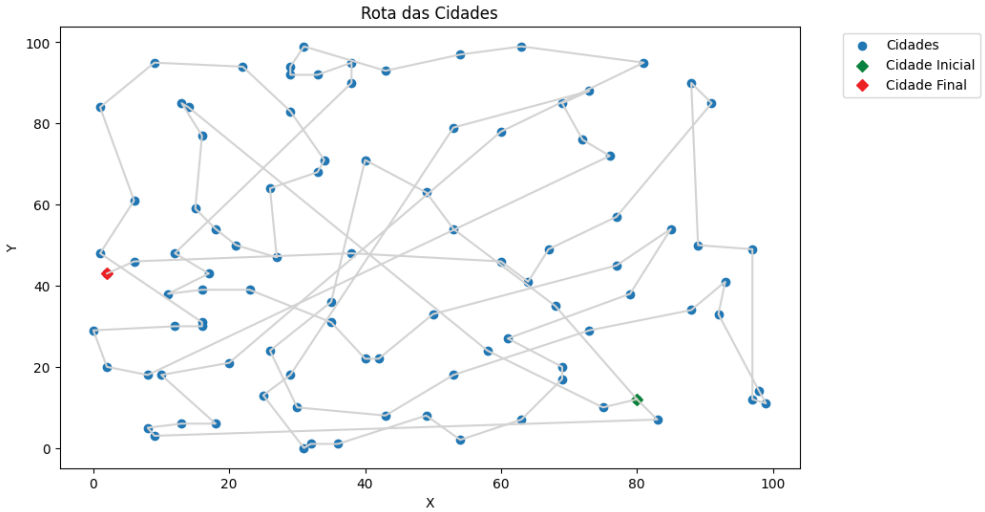
```
  #calcula aptidao de toda a populacao
  custos = calcular_aptidao(populacao)
  # Ordena os custos e mantém os índices originais usando a função zip
  custos_ordenados = sorted(range(len(custos)), key=lambda k: custos[k])
  indice_menor_custo = custos_ordenados[0]
  #mantém histórico de custos
  historico_custos.append(custos[indice_menor_custo])
  if  geracao % ciclos_custo == 0:
    print("Geração=",geracao ," Menor Custo=",custos[indice_menor_custo])
  populacao = [populacao[i] for i in custos_ordenados]
  populacao = populacao[:qtde_selecao]
  return populacao

def mutacao(cromossomo, chance_mutacao):
  for i in range(1, 50):
    if random.random() < chance_mutacao:
      gene1 = random.randint(1, 99)
      gene2 = random.randint(1, 99)
      cromossomo[gene1], cromossomo[gene2] = cromossomo[gene2],
cromossomo[gene1]
    return cromossomo

def crossover(cromossomo1, cromossomo2, chance_mutacao):
  novo_cromossomo1 = cromossomo1.copy()
  novo_cromossomo2 = cromossomo2.copy()
  for i in range(1,50):
    novo_cromossomo1[i] = cromossomo2[i]
    novo_cromossomo2[i] = cromossomo1[i]
  cidades_faltantes = verifica_cidades_faltantes(novo_cromossomo1)
  inclui_cidades_faltantes(novo_cromossomo1,cidades_faltantes)
  cidades_faltantes = verifica_cidades_faltantes(novo_cromossomo2)
  inclui_cidades_faltantes(novo_cromossomo2,cidades_faltantes)
  novo_cromossomo1 = mutacao(novo_cromossomo1, chance_mutacao)
  novo_cromossomo2 = mutacao(novo_cromossomo2, chance_mutacao)
  return novo_cromossomo1, novo_cromossomo2

def nova_populacao(populacao,qtde_individuos_cruzamento, chance_mutacao):
  nova_populacao = []
  for i in range(qtde_individuos_cruzamento):
    cromossomo1 = random.choice(populacao)
```

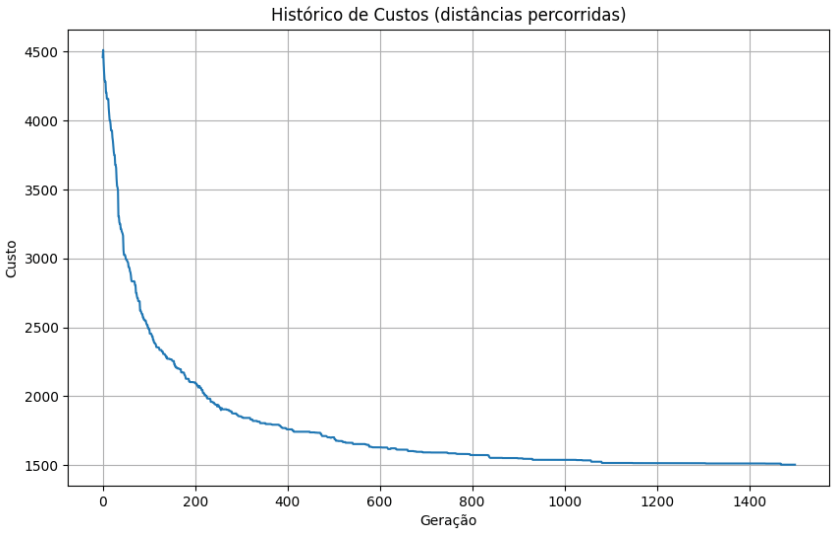
```
    cromossomo2 = random.choice(populacao)
    novo_cromossomo1, novo_cromossomo2 = crossover(cromossomo1, cromossomo2,
chance_mutacao)
    nova_populacao.append(novo_cromossomo1)
    nova_populacao.append(novo_cromossomo2)
  return nova_populacao
```

Rotas estabelecidas randomicamente com o primeiro cromossomo:



Rotas depois de 1000 gerações:





4

2) Compare a representação de dois modelos vetoriais

 # Recortes do código

```
# textos contendo similaridades
textos = [
    "A receita de bolo de chocolate é deliciosa e fácil de fazer.",
    "Eu adoro fazer bolo de chocolate, é sempre um sucesso!",
    "Prefiro a receita de torta de maçã, é mais leve.",
    "Torta de maçã com canela é uma combinação perfeita.",
    "O jantar de hoje terá lasanha e salada.",
    "Lasanha à bolonhesa é um prato reconfortante.",
    "Preparei um delicioso bolo de limão para o café.",
    "Bolo de limão com cobertura de cream cheese é divino."
]

# 2. Vectorização TF-IDF
vectorizer = TfidfVectorizer()
vetores = vectorizer.fit_transform(textos)
vetores = vetores.toarray()

# 3. PCA
pca = PCA(n_components=2)
componentes_principais = pca.fit_transform(vetores)

# 4. Visualização
plt.figure(figsize=(10, 8))  # Aumentei o tamanho da figura
plt.scatter(componentes_principais[:, 0], componentes_principais[:, 1])
```

```
plt.title("Projeção dos Textos usando PCA (Receitas)")
plt.xlabel("x")
plt.ylabel("y")

# Adicionar rótulos aos pontos com ajuste de posição
for i, texto in enumerate(textos):
                plt.annotate(texto,       (componentes_principais[i,       0],
componentes_principais[i, 1]),
                 textcoords="offset points", xytext=(5,5), ha='left')

plt.grid(True)
plt.tight_layout()
plt.show()
```