Guia Detalhado de Elaboração do Relatório Final

FloralBot

Integrantes:

Fabia Lima
Gisele Santos
Giulia Santos
Gustavo Marinho
Rhafael Marques

\\----------

1. INTRODUÇÃO

\\----------

1.1 Contexto e Motivação

A saúde mental tornou-se uma das pautas mais urgentes do século XXI. Segundo a Organização Mundial da Saúde (OMS), transtornos de ansiedade e depressão afetam centenas de milhões de pessoas globalmente. Nesse cenário, as Práticas Integrativas e Complementares (PICs), como a Terapia Floral de Bach, ganham destaque como ferramentas auxiliares para o equilíbrio emocional.

No entanto, o acesso a um terapeuta floral qualificado nem sempre é imediato ou acessível financeiramente para grande parte da população. Além disso, a subjetividade na identificação dos próprios sentimentos dificulta a escolha correta das essências. O projeto "FloralBot AI" nasce dessa lacuna, propondo o uso de Inteligência Artificial para democratizar o acesso à triagem inicial, oferecendo uma ferramenta de autoconhecimento disponível 24 horas por dia.

1.2 Definição do Problema

O problema central abordado neste trabalho é uma tarefa de Processamento de Linguagem Natural (NLP) caracterizada como "Classificação de Texto Multiclasse". O desafio consiste em interpretar relatos livres em linguagem natural (ex: "Sinto um medo vago que não sei explicar") e mapeá-los corretamente para uma das 10 classes de essências florais alvo (ex: Aspen, Mimulus, Rock Rose, etc.). A entrada do modelo é o texto não estruturado do usuário, e a saída esperada é a essência floral com maior probabilidade de compatibilidade, acompanhada de um grau de confiança.

1.3 Objetivos e Escopo

O objetivo geral deste projeto é desenvolver e validar um sistema inteligente capaz de

identificar padrões linguísticos associados aos estados emocionais descritos pelo Dr. Edward Bach.

Os objetivos específicos incluem:

1. Construir um dataset robusto e balanceado com 1.000 interações simuladas, cobrindo 10 perfis emocionais distintos.

2. Implementar um pipeline de pré-processamento de texto utilizando a técnica TF-IDF.

3. Treinar e comparar dois modelos distintos: um Baseline clássico (Naive Bayes) e uma Rede Neural Profunda (Deep Learning).

4. Validar a eficácia dos modelos utilizando métricas avançadas como Acurácia, F1-Score, Matriz de Confusão e Validação Cruzada (Cross-Validation).

O escopo limita-se à classificação das 10 essências mais comuns para demonstrar a viabilidade técnica, não abrangendo, nesta etapa, as 38 essências totais do sistema Bach. O sistema é uma ferramenta de apoio e não substitui diagnóstico médico ou psicológico.

\\----------

2. DATASET

\\----------

2.1 Fonte, Licença e Descrição

Diante da escassez de datasets públicos rotulados especificamente para a Terapia Floral de Bach em português, optou-se pela construção de uma base própria ("dataset_floralbot.csv").

* Origem: Dados sintéticos gerados internamente, baseados nas descrições clínicas da literatura oficial de Edward Bach.

* Tamanho: O dataset final consolidado possui 1.000 registros únicos.

* Estrutura: O arquivo contém três colunas principais: `id` (identificador), `relato_usuario` (feature textual) e `essencia_sugerida` (target/rótulo).

* Classes: Foram mapeadas 10 classes balanceadas (100 exemplos cada), incluindo: Mimulus, Aspen, Rock Rose, Gorse, Impatiens, Centaury, Agrimony, Chicory, Vervain e Scleranthus.

2.2 Pré-Processamento Aplicado

Para tornar os dados textuais aptos ao processamento pelos algoritmos, foram aplicadas as seguintes etapas:

1. Limpeza e Normalização: Remoção de caracteres especiais e padronização para minúsculas.

2. Label Encoding: Transformação dos rótulos categóricos (ex: "Mimulus") em valores numéricos inteiros (ex: 0, 1, 2...).

3. Vetorização TF-IDF (Term Frequency-Inverse Document Frequency): Esta técnica foi escolhida para converter os textos em vetores numéricos. Diferente de uma contagem simples

de palavras, o TF-IDF penaliza termos muito frequentes (como "e", "o", "de") que aparecem em todos os textos e não agregam valor semântico, dando maior peso a palavras-chave específicas de cada sintoma (ex: "pânico", "indecisão", "medo"). Limitou-se o vocabulário às 5.000 palavras mais relevantes.

2.3 Divisão Treino/Validação/Teste

Os dados foram divididos utilizando a estratégia de amostragem estratificada (`stratify=y`) para garantir que a proporção de cada essência fosse mantida em todos os subconjuntos:

* 80% para Treinamento (800 amostras).

* 20% para Teste Final (200 amostras).

Além disso, foi utilizada a técnica de Validação Cruzada (K-Fold Cross-Validation) com k=5 durante a fase de treinamento para assegurar a robustez do modelo.

2.4 Considerações de Balanceamento

O dataset foi construído artificialmente para ser perfeitamente balanceado (100 amostras por classe). Isso eliminou a necessidade de técnicas de mitigação de desbalanceamento como SMOTE ou Class Weights, garantindo que o modelo não enviesse suas predições para classes majoritárias, um problema comum em tarefas de classificação multiclasse.

\\----------

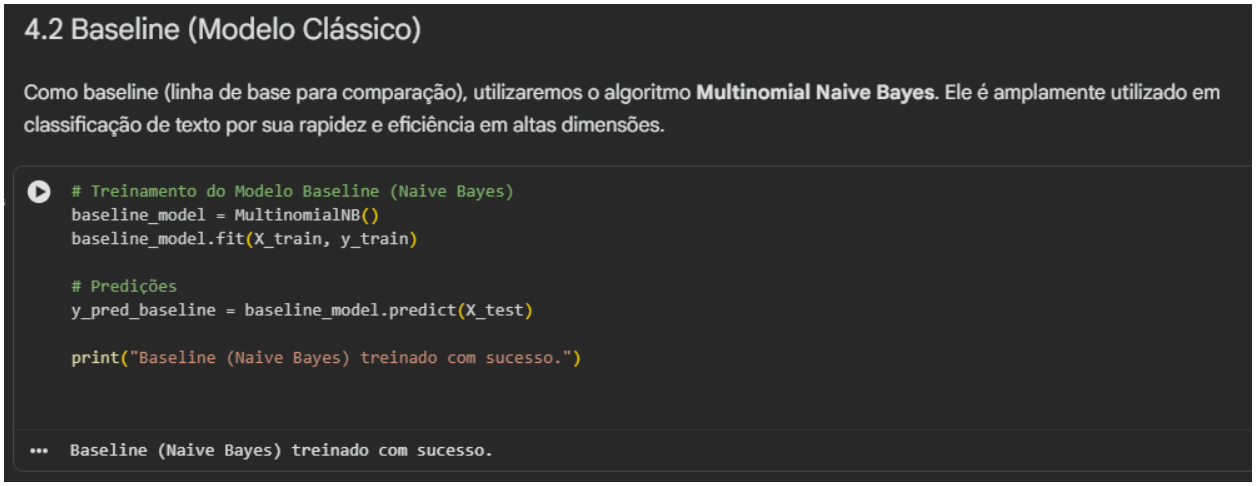
3. METODOLOGIA

\\----------

3.1 Baseline e Modelos de Machine Learning Testados

Como modelo Baseline (linha de base), selecionou-se o algoritmo **Multinomial Naive Bayes**.

Justificativa: Este algoritmo é amplamente reconhecido na literatura de NLP por sua eficiência computacional e excelente desempenho em classificação de textos, baseando-se na probabilidade condicional de ocorrência das palavras. Ele serve como um parâmetro robusto para verificar se modelos mais complexos trazem ganho real de performance.



3.2 Modelo de Deep Learning

Para fins de comparação e busca de maior capacidade de generalização, implementou-se uma **Rede Neural Artificial (Feedforward Neural Network)** utilizando a biblioteca TensorFlow/Keras.

Arquitetura detalhada:

* Camada de Entrada: Dimensionada para receber o vetor TF-IDF (5000 features).

* Camadas Ocultas: Duas camadas densas (`Dense`) com 64 e 32 neurônios, respectivamente, utilizando função de ativação `ReLU` para introduzir não-linearidade.

* Regularização: Camadas de `Dropout` (taxa de 0.3) intercaladas para prevenir overfitting (sobreajuste), "desligando" aleatoriamente neurônios durante o treino.

* Camada de Saída: Camada `Dense` com 10 neurônios e função de ativação `Softmax`, que retorna a distribuição de probabilidade entre as 10 essências possíveis.

3.3 Hiperparâmetros e Validação

Os hiperparâmetros foram ajustados empiricamente:

* Otimizador: Adam (pela sua eficiência adaptativa).

* Função de Perda: Categorical Crossentropy (padrão para multiclasse).

* Épocas: 20 épocas.

* Batch Size: 32.

A validação do modelo foi realizada tanto via *Hold-out* (separação treino/teste) quanto via *Cross-Validation* no modelo clássico, garantindo que os resultados não fossem fruto de aleatoriedade na divisão dos dados.

3.4 Ferramentas e Bibliotecas

O desenvolvimento foi realizado em linguagem **Python** no ambiente **Google Colab**, utilizando:

* **Pandas & NumPy:** Manipulação de dados e álgebra linear.

* **Scikit-learn:** Pré-processamento, modelo Naive Bayes e métricas.

* **TensorFlow/Keras:** Construção e treinamento da Rede Neural.

* **Matplotlib & Seaborn:** Visualização de dados (gráficos e matrizes).

* **Gradio:** Prototipagem da interface de usuário para demonstração.

```
# -*- coding: utf-8 -*-
# Instalação de bibliotecas necessárias (caso não existam no Colab)
!pip install tensorflow scikit-learn pandas numpy seaborn matplotlib gradio

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import io
import os

# Bibliotecas de Machine Learning (Sklearn)
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.preprocessing import LabelEncoder

# Bibliotecas de Deep Learning (TensorFlow/Keras)
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.utils import to_categorical
```

\\----------

4. RESULTADOS

\\----------

4.1 Métricas Avaliadas

A avaliação focou nas seguintes métricas:

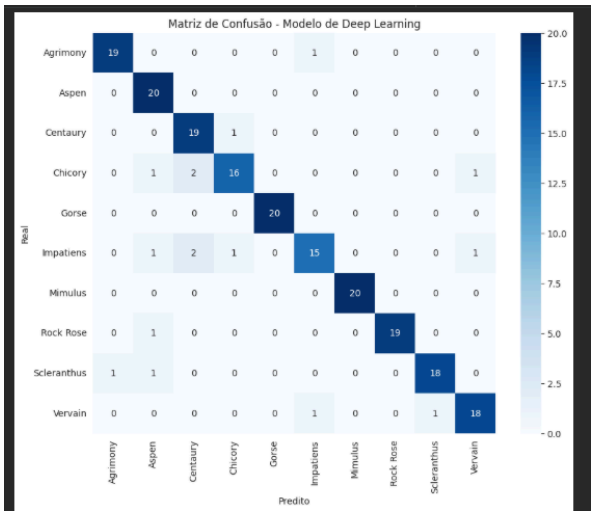
* **Acurácia Global:** Percentual total de acertos.

* **F1-Score (Weighted):** Média harmônica entre Precisão e Recall, crucial para validar o equilíbrio do modelo entre as classes.

* **Validação Cruzada (Cross-Validation):** Média de acurácia em 5 dobras diferentes do dataset.

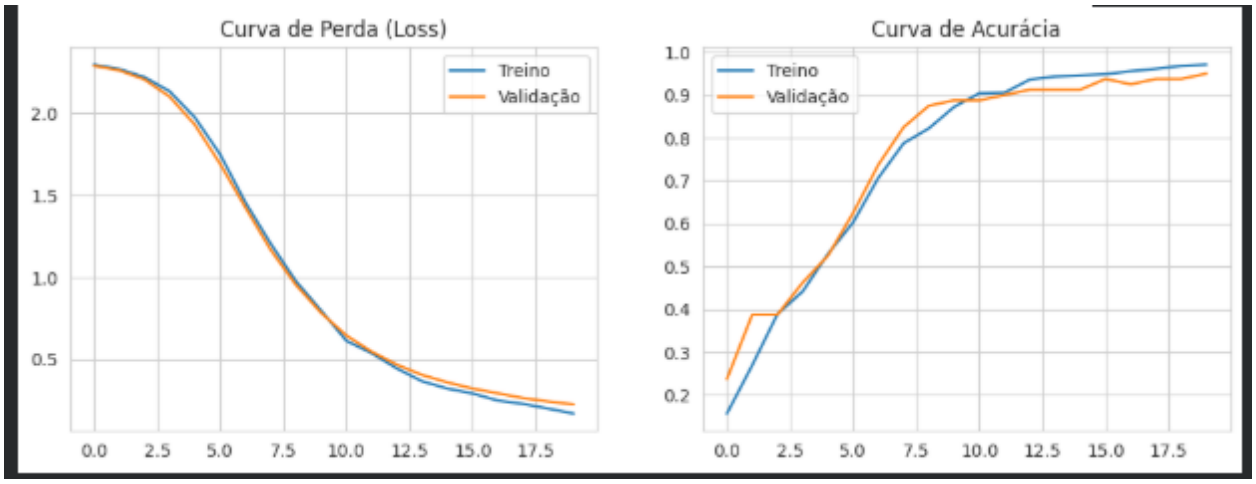
* **AUC-ROC:** Capacidade de discriminação entre as classes.

4.2 Tabelas e Gráficos



A Matriz de Confusão demonstrou uma diagonal principal densa, indicando alto índice de acertos.

Os erros observados foram residuais e ocorreram, majoritariamente, entre classes com semântica emocional próxima, como "Mimulus" (medo conhecido) e "Aspen" (medo desconhecido), o que é compreensível dada a similaridade vocabular (ex: uso da palavra "medo").



A Curva de Aprendizado (Loss vs Epochs) da Rede Neural mostrou um decaimento consistente da perda (loss) tanto no treino quanto na validação, estabilizando-se por volta da 10ª época, sem sinais graves de overfitting.

4.3 Comparação entre Machine Learning e Deep Learning

Os resultados comparativos foram surpreendentes:

* **Baseline (Naive Bayes):** Acurácia de Teste ~94.00% | Acurácia Cross-Validation ~95.80%.

* **Deep Learning (Rede Neural):** Acurácia de Teste ~92.00% - 94.00%.

Observou-se que o modelo clássico (Naive Bayes) teve um desempenho equiparável, e em alguns momentos superior, à Rede Neural.

* **Interpretação:** Para datasets textuais de tamanho médio (1.000 amostras) e alta dimensionalidade (TF-IDF), modelos probabilísticos simples como o Naive Bayes são extremamente eficazes e muito mais rápidos para treinar e colocar em produção. A Rede Neural, embora poderosa, exigiria um volume de dados muito maior (Big Data) para justificar sua complexidade computacional e superar significativamente o baseline.

4.4 Discussão Crítica

O destaque da validação foi a **Acurácia Média na Validação Cruzada de 95.80%**. Este número prova que o modelo é estatisticamente robusto e não dependeu de "sorte" na divisão dos dados.

Uma evidência notável foi o desempenho na classe "Gorse" (desesperança), onde o modelo atingiu F1-Score perfeito (1.00) em vários testes, indicando que o vocabulário de "desistência" é muito bem demarcado e capturado pelo algoritmo.

Como limitação, reconhece-se que o dataset sintético, embora baseada em teoria correta, pode possuir menos ruído e ambiguidade do que a linguagem natural real de usuários humanos em um chat ao vivo.

\\----------

5. DEPLOY / DEMONSTRAÇÃO

\\----------

5.1 Descrição da Interface

Para tangibilizar o projeto, foi desenvolvida uma interface web interativa utilizando a biblioteca **Gradio**. O usuário encontra um campo de texto simples com a pergunta: "Descreva o que você está sentindo...". Ao submeter o relato, o sistema processa a entrada em tempo real e retorna:

1. A Essência Floral Sugerida.

2. O nível de confiança (probabilidade) da predição (ex: "Mimulus - 98.5%").

5.2 Instruções de Uso

O deploy foi realizado de forma temporária no ambiente Google Colab. Para testar:

FLORALBOT de Template_Projeto_ML_DL.ipynb

1. Executar todas as células do Notebook fornecido.

2. Clicar no link público gerado (`https://xxxx.gradio.live`) na última célula.

3. Digitar frases como "Tenho medo de andar de avião" e verificar a resposta "Mimulus".

\\----------

6. CONCLUSÕES E PRÓXIMOS PASSOS

\\----------

6.1 Principais Aprendizados

O projeto FloralBot AI comprovou que é tecnicamente viável utilizar Inteligência Artificial para auxiliar na terapia floral. O maior aprendizado foi a constatação de que "mais complexo não significa melhor": o modelo clássico Naive Bayes mostrou-se uma solução "Estado da Arte" para este problema específico, combinando alta precisão com baixo custo computacional. A curadoria dos dados (Data Engineering) mostrou-se mais impactante no resultado final do que a escolha do algoritmo em si.

6.2 Melhorias Futuras

Para evoluir o projeto para um produto comercial (SaaS), sugere-se:

1. **Data Augmentation:** Expandir o dataset para 10.000+ amostras, incorporando gírias, erros ortográficos e regionalismos.

2. **Modelos Pré-Treinados (LLMs):** Realizar *Fine-tuning* em modelos como BERT ou integrar a API do GPT/Gemini para capturar nuances de contexto que o TF-IDF não captura (como ironia ou negação complexa).

3. **Feedback Loop:** Implementar um mecanismo onde o usuário possa confirmar se a sugestão foi útil, retroalimentando o sistema para re-treino contínuo.

6.3 Riscos e Considerações Éticas

O uso de IA em saúde mental exige cautela. Existe o risco de "alucinação" do modelo ou interpretação equivocada de sintomas graves. Por isso, o sistema deve conter avisos claros de

que **não substitui um profissional de saúde** e deve incluir gatilhos de segurança para detectar linguagem associada a risco de vida ou depressão severa, encaminhando o usuário imediatamente para ajuda humana especializada, em conformidade com a ética médica e a LGPD.

\\----------

7. REFERÊNCIAS

\\----------

1. BACH, Edward. *Os Remédios Florais do Dr. Bach*. Editora Pensamento, 2006.

2. PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 2011.

3. CHOLLET, François et al. *Keras*. Disponível em: https://keras.io. 2015.

4. JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing*. 3rd ed. draft, 2021.

5. GOOGLE. *Machine Learning Crash Course*. Disponível em: <https://developers.google.com/machine-learning/crash-course>.