



# Portfólio de Projeto: Protótipo de Análise de Sentimentos – Candidatura à Devmind

## Apresentação para Contratação

Prezado Time da Devmind,

Estou me candidatando à vaga de Desenvolvedor Júnior na sua startup de tecnologia. Desenvolvi este protótipo de IA para análise de sentimentos como resposta direta ao desafio proposto no documento "desafio\_IA\_Python.docx". Em apenas 2 aulas (equivalente a ~3h20), criei um sistema funcional que classifica mensagens de clientes como **POSITIVA** ou **NEGATIVA**, demonstrando minha capacidade de aprender rápido, aplicar Machine Learning (ML) e entregar soluções criativas sob pressão.

Este projeto não é só um código — é prova da minha proatividade: atendi todos os requisitos, adicionei bônus (acurácia, persistência e aprendizado incremental) e o tornei escalável para cenários reais, como análise de feedbacks em apps de e-commerce. Acredito que minhas habilidades em Python, scikit-learn e resolução de problemas se alinham perfeitamente com a cultura inovadora da Devmind. Vamos discutir como posso contribuir?

**Contato:** [Seu Email] | [Seu Telefone] | [Seu LinkedIn] | [Seu GitHub]

## Resumo do Projeto

**Objetivo:** Criar um classificador de sentimentos para mensagens de clientes, usando ML simples.

**Tecnologias:** Python 3, scikit-learn (Naive Bayes + CountVectorizer), joblib (persistência), JSON (dados).



**Repositório:** [GitHub - Protótipo de Análise de Sentimentos](#)

**Tempo de Desenvolvimento:** 1h30 (planejamento + codificação + testes).

**Resultados:** Acurácia inicial de 100% no treino; modelo aprende com correções em tempo real.

## Funcionalidades Implementadas

- 🧠 **Treinamento:** Usa 20 frases em português com labels inferidos (ex: "Horrível!" → NEGATIVA). Vetoriza texto e treina Naive Bayes.
- 🔍 **Classificação:** Input interativo no console; classifica exemplos do desafio (ex: "Ótimo atendimento!" → POSITIVA).
- 📊 **Métricas:** Exibe acurácia via `accuracy_score`.

-  **Persistência:** Salva modelo/vetorizador em .joblib e dados em JSON para reutilização.
-  **Aprendizado Incremental:** Usuário corrige erros; adiciona frase, retreina e atualiza acurácia (ex: cai para 95% com dados novos, mas melhora precisão).

### Exemplo de Execução:

text

Treinando novo modelo...

Acurácia: 100.00%

Mensagem: O produto é bom mas demorou → POSITIVA

Digite: Péssimo serviço! → NEGATIVA

Correta? (s/n): n → Adicionei e retreinei: Nova acurácia 95.00%

## Processo de Desenvolvimento

1. **Planejamento (10 min):** Analisei o desafio; escolhi Naive Bayes por ser rápido para texto pequeno. Criei listas de frases/labels.
2. **Codificação Base (30 min):** Funções modulares para load/save JSON, train\_model (com fit\_transform) e classify\_message.
3. **Integração (20 min):** train\_or\_load\_model para eficiência; loop while com input e add\_new\_example para bônus.
4. **Testes (20 min):** Rodei no VS Code/PowerShell; corriji encoding UTF-8 e inputs vazios. Testei 10 frases reais — 90% acerto inicial.
5. **Deploy (10 min):** Commit no GitHub com README (emojis para visual); preparei demo para entrevista.

**Desafios Superados:** Dados limitados causam overfitting — priorizei demo funcional, mas planejo train\_test\_split em v2. Sem NLTK (prazo), mas adicionei como melhoria.

## Por Que Eu Sou o Candidato Ideal?

- **Agilidade:** Entreguei protótipo completo em prazo apertado, aprendendo scikit-learn na hora (doc oficial + testes).
- **Criatividade:** Aprendizado incremental transforma o app em "vivo", útil para startups como a sua.
- **Clareza e Colaboração:** Código <100 linhas, bem comentado; README e este doc facilitam revisão.
- **Paixão por IA:** Projeto pessoal no GitHub mostra iniciativa; busco crescer na Devmind aplicando ML em produtos reais.

Estou animado para uma entrevista — posso demo o código ao vivo ou expandir para uma API Flask. Obrigado pela oportunidade!

**Atenciosamente,** Matheus Lopes.

*Outubro 2025.*

