**DESENVOLVIMENTO DE UMA APLICAÇÃO WEB AUTOMATIZADA PARA ANÁLISE DE DADOS COM MACHINE LEARNING E PROCESSAMENTO DE LINGUAGEM NATURAL**

Rafael Souza Osadzuk

Acadêmico do Curso de Engenharia de Software do Centro Universitário Cesumar - UNICESUMAR, Curitiba - PR.

**RESUMO:** Este trabalho propõe o desenvolvimento de uma aplicação interativa e automatizada de aprendizado de máquina, utilizando as bibliotecas PyCaret e Streamlit, com o objetivo de facilitar o processo de análise exploratória, modelagem preditiva e análise de sentimentos. O sistema foi projetado para ser flexível, permitindo o carregamento de qualquer base de dados tabular, a escolha do tipo de tarefa (regressão, classificação, clusterização) e a execução de modelos de análise de sentimentos com dados textuais. O backend utiliza PyCaret para automação dos experimentos, enquanto o frontend em Streamlit permite a interação intuitiva com o usuário. O projeto também integra um banco de dados relacional (SQL) contendo múltiplas bases padronizadas para cada tarefa de ML. O estudo demonstra que é possível criar uma solução completa, escalável e de fácil uso para projetos de ciência de dados educacionais ou profissionais.

**PALAVRAS-CHAVE:** Machine Learning; Streamlit; PyCaret; Análise de Dados; Modelagem Preditiva.

## INTRODUÇÃO

## Nos últimos anos, o uso de dados como base para a tomada de decisões tem crescido de forma significativa, impulsionando a necessidade por ferramentas que tornem o aprendizado de máquina (Machine Learning – ML) mais acessível e eficaz. Tradicionalmente, desenvolver modelos de ML envolve diversas fases técnicas – desde a preparação dos dados até a criação, validação e aplicação dos modelos –, o que demanda conhecimento especializado em programação, estatística e ciência de dados. Essa exigência acaba restringindo o uso dessas tecnologias a um público altamente técnico.

## Com o avanço das ferramentas de desenvolvimento low-code e no-code, surgem alternativas que simplificam esse processo, tornando-o mais amigável para profissionais de outras áreas. Dentre essas soluções, destaca-se o PyCaret, que automatiza grande parte do ciclo de vida de projetos de ML, e o Streamlit, um framework que facilita a construção de interfaces gráficas interativas com Python, permitindo a criação rápida de dashboards e aplicações web voltadas à análise de dados.

## Neste contexto, este projeto propõe o desenvolvimento de uma aplicação interativa voltada à experimentação e construção de modelos de Machine Learning de forma prática e acessível. Integrando o PyCaret com o Streamlit, a aplicação permite que usuários realizem desde a análise exploratória de dados até a escolha do tipo de problema (seja ele de regressão, classificação ou agrupamento), o treinamento e a comparação de modelos, e até mesmo a geração de previsões a partir de novos dados. A proposta visa democratizar o uso do ML, oferecendo uma solução flexível para diversos contextos, mesmo para aqueles que não dominam linguagens de programação ou técnicas avançadas de modelagem.

* 1. **Estrutura Tecnológica da Aplicação**

A solução proposta foi desenvolvida com base em uma combinação de tecnologias que oferecem simplicidade, performance e escalabilidade. Os principais elementos dessa arquitetura são descritos a seguir:

* **Interface Interativa com Streamlit**: Atuando como camada de apresentação, o Streamlit possibilita a construção de uma interface web intuitiva, por meio da qual os usuários podem carregar arquivos, configurar parâmetros e visualizar os resultados de forma dinâmica e direta no navegador.
* **Automação do Ciclo de Machine Learning com PyCaret**: No núcleo da aplicação está o PyCaret, uma biblioteca que automatiza tarefas comuns em projetos de aprendizado de máquina, como preparação dos dados, escolha de modelos, avaliação e comparação de desempenho. Isso reduz consideravelmente o esforço necessário para conduzir experimentos preditivos.
* **Manipulação de Dados com Pandas**: A biblioteca Pandas é responsável pelo carregamento e transformação dos dados em memória. Sua função é facilitar a análise preliminar e permitir a transição fluida entre a interface e os módulos de modelagem.
* **Empacotamento com Docker (uso opcional)**: Para garantir que a aplicação possa ser executada de maneira consistente em diferentes sistemas operacionais, considera-se o uso do Docker. A containerização permite que todo o ambiente da aplicação — incluindo dependências e configurações — seja distribuído em um pacote único e reprodutível.
* **Integração com Banco de Dados Relacional (em desenvolvimento)**: Visando evoluir para uma solução mais robusta, está prevista a conexão com sistemas de banco de dados SQL. Essa integração permitirá armazenar bases de dados, modelos treinados e previsões realizadas, promovendo escalabilidade e rastreabilidade para a aplicação.
  1. **Recursos Implementados na Plataforma**

A aplicação foi construída para contemplar, de maneira prática e acessível, todas as fases essenciais de um projeto de aprendizado de máquina. A seguir, são descritas as principais funcionalidades disponíveis ao usuário:

* **Importação de Conjuntos de Dados**: O sistema permite o carregamento de arquivos nos formatos .csv ou .xlsx, possibilitando que o usuário traga seus próprios dados. Após o upload, o conteúdo é automaticamente convertido para um formato compatível com as etapas seguintes.
* **Visualização Inicial e Diagnóstico dos Dados**: Com base nos dados carregados, a aplicação realiza uma análise descritiva inicial, exibindo informações como tipos de variáveis, estatísticas básicas (média, mediana, desvio padrão etc.) e a quantidade de valores ausentes. Essa etapa fornece uma visão geral da estrutura e qualidade dos dados.
* **Escolha da Variável Alvo**: O usuário define a variável que será prevista (no caso de tarefas supervisionadas), orientando o processo de modelagem. Isso é essencial para os tipos de problema de regressão e classificação.
* **Definição do Tipo de Tarefa Preditiva**: A plataforma oferece suporte a três categorias principais de problemas: classificação, regressão e clusterização. A escolha dessa opção ajusta automaticamente os algoritmos utilizados e as métricas de avaliação apresentadas.
* **Execução Automatizada de Modelos com PyCaret**: A aplicação utiliza o PyCaret para preparar os dados, selecionar os algoritmos adequados, treinar múltiplos modelos e comparar seus desempenhos. Essa operação é realizada de forma automatizada e os resultados são exibidos em tabelas ordenadas por performance.
* **Visualização de Métricas e Gráficos de Avaliação**: Para o modelo com melhor desempenho, o usuário pode gerar visualizações específicas para facilitar a interpretação dos resultados. Exemplos incluem: curva ROC e matriz de confusão (classificação), gráfico de resíduos (regressão), e análise de agrupamentos como gráfico de Elbow (clusterização).
* **Previsões com Novos Conjuntos de Dados**: Após o treinamento, é possível carregar um novo arquivo contendo dados inéditos. O modelo salvo será utilizado para realizar previsões automaticamente, replicando um cenário de uso real em produção.

Essas funcionalidades foram projetadas para fornecer ao usuário uma experiência fluida, didática e tecnicamente sólida, mesmo sem exigir conhecimento avançado em programação ou ciência de dados.

**1.3 Etapas de Navegação e Uso da Aplicação**

A jornada do usuário dentro da plataforma foi desenhada para ser lógica, sequencial e de fácil entendimento, mesmo para pessoas com pouca familiaridade com aprendizado de máquina. O processo se desenrola da seguinte maneira:

1. **Acesso à Plataforma**: A aplicação é executada em um navegador web, por meio do Streamlit, permitindo uso imediato sem necessidade de instalação local de interface gráfica.
2. **Importação do Arquivo de Dados**: O usuário carrega um arquivo de entrada nos formatos .csv ou .xlsx. A leitura e estruturação dos dados são realizadas com o auxílio da biblioteca Pandas, convertendo o conteúdo para um DataFrame manipulável.
3. **Visualização e Diagnóstico Preliminar**: Após o carregamento, são exibidas estatísticas resumidas dos dados, como contagem de valores nulos, tipos de variáveis e medidas descritivas. Essa etapa oferece uma visão geral da base para orientar as decisões seguintes.
4. **Configuração do Problema Preditivo**: O usuário escolhe qual coluna representa a variável a ser prevista (alvo) e informa se deseja trabalhar com uma tarefa de **classificação**, **regressão** ou **agrupamento (clusterização)**. Essa seleção adapta automaticamente o comportamento do PyCaret.
5. **Treinamento Automatizado de Modelos**: Com um clique, o PyCaret é acionado. Ele realiza a preparação dos dados, treina diversos modelos disponíveis e apresenta os resultados comparativos com base em métricas apropriadas para o tipo de problema selecionado.
6. **Exibição de Resultados e Gráficos de Avaliação**: A interface apresenta os desempenhos dos modelos testados em forma de tabela e gráficos interativos, permitindo uma análise mais aprofundada do modelo com melhor performance.
7. **Salvamento do Modelo e Previsões Futuras**: Após o treinamento, o modelo mais eficiente é salvo em disco. A plataforma oferece uma opção para carregar novos dados e aplicar o modelo armazenado, simulando um ambiente real de produção.

Esse fluxo foi desenvolvido para tornar o processo de aprendizado de máquina mais acessível, prático e automatizado, permitindo que usuários com diferentes níveis de conhecimento consigam conduzir projetos completos sem escrever código manualmente.

**Conclusão**

A solução interativa de Machine Learning criada, que integra as ferramentas Streamlit e PyCaret, representa um avanço importante na ampliação do acesso a recursos avançados de análise e modelagem de dados. Ao simplificar etapas complexas do ciclo de desenvolvimento de modelos preditivos, a plataforma possibilita que usuários com diferentes níveis de conhecimento possam explorar dados, construir, avaliar e comparar modelos de forma intuitiva e eficiente. Essa integração demonstra que é possível acelerar significativamente o desenvolvimento de projetos de ML, oferecendo resultados consistentes para tarefas de classificação, regressão e clusterização.

Um dos maiores benefícios identificados durante o desenvolvimento foi a efetividade da abordagem low-code para aplicações de Machine Learning. A união do Streamlit, que facilita a criação rápida de interfaces interativas, com o PyCaret, que automatiza o pipeline completo de modelagem, resultou em uma solução ágil, acessível e robusta. Além disso, a estrutura modular adotada favorece a escalabilidade do sistema, permitindo futuras expansões, atualizações e manutenção simplificada, essenciais para a aplicação prática em ambientes reais.

Esse trabalho reforça a viabilidade de utilizar plataformas low-code como alternativas poderosas para democratizar o uso do Machine Learning, abrindo novas possibilidades para profissionais de diversas áreas aplicarem técnicas preditivas sem a necessidade de profundo conhecimento técnico.