### **Documentação do Projeto: Análise e Previsão de Risco de Crédito (Banco Box)**

*Arthur Trindade*

#### **1. Objetivo do Projeto**

O objetivo principal é desenvolver uma solução de análise de dados para o Banco Box. Esta solução visa, em primeiro lugar, tratar e enriquecer os dados dos clientes e, em segundo lugar, construir um modelo de Machine Learning capaz de classificar novos clientes em três categorias de risco de crédito: **Baixo, Médio ou Alto**. O objetivo final é permitir que o banco tome decisões mais informadas e seguras, minimizando o risco de inadimplência.

#### 

#### **2. Processo de ETL (Extração, Transformação e Carga)**

O ETL é a fundação do projeto, onde preparamos os dados brutos para se tornarem informações úteis.

**2.1. Extração**

A primeira etapa é carregar os dados.

import pandas as pd

df = pd.read\_csv("banco\_box\_sintetico.csv")

* **Explicação:** Utilizamos a biblioteca pandas para ler o arquivo banco\_box\_sintetico.csv. Os dados são carregados para uma tabela em memória chamada DataFrame, que chamamos de df.

**2.2. Transformação**

Esta é a fase de limpeza e enriquecimento dos dados.

median\_renda = df['renda\_mensal'].median()

df['renda\_mensal'].fillna(median\_renda, inplace=True)

* **Explicação:** Identificamos que a coluna renda\_mensal tinha valores em falta. Para resolver isto, calculamos a **mediana** (o valor do meio) de todas as rendas e usamos para preencher os campos vazios. A mediana é preferível à média porque não é tão afetada por salários extremamente altos ou baixos.

df['relacao\_divida\_renda'] = df['dividas\_total'] / df['renda\_mensal'].replace(0, 1e-6)

* **Explicação:** Criamos uma nova coluna chamada relacao\_divida\_renda. Este é um indicador que calcula a percentagem da renda mensal de um cliente que já está comprometida com dívidas. Uma relação alta pode indicar um risco de crédito maior.

**2.3. Carga**

Após a transformação, guardamos o resultado.

df.to\_csv("banco\_tratado\_para\_analise.csv", index=False)

* **Explicação:** O DataFrame df, agora limpo e com a nova coluna, é guardado num novo arquivo chamado banco\_tratado\_para\_analise.csv. Este ficheiro está pronto para ser usado.

#### **3. Análise Exploratória**

#### Antes de construir o modelo, exploramos os dados para extrair informações.

**3.1. Análise com NumPy**

import numpy as np

media\_historico = np.mean(df['historico\_credito'])

* **Explicação:** Usamos a biblioteca NumPy, otimizada para cálculos numéricos, para calcular a **média** da coluna historico\_credito. Este valor dá-nos uma referência do tempo médio de histórico de crédito de um cliente típico do banco.

**3.2. Visualização com Matplotlib**

import matplotlib.pyplot as plt

import seaborn as sns

sns.countplot(x='risco\_credito', data=df, order=['Baixo', 'Médio', 'Alto'])

plt.title('Distribuição de Clientes por Nível de Risco')

plt.show()

* **Explicação:** Este código gera um **gráfico de barras** que mostra visualmente quantos clientes pertencem a cada categoria de risco (Baixo, Médio e Alto). Esta visualização é crucial para entendermos a distribuição dos dados e se uma categoria é muito mais frequente que as outras.

#### **4. Construção do Modelo de Classificação**

Esta é a fase onde ensinamos a prever o risco de crédito.

**4.1. Preparação e Pré-processamento**

X = df.drop('risco\_credito', axis=1)

y = df['risco\_credito']

numerical\_features = X.select\_dtypes(include=np.number).columns.tolist()

categorical\_features = X.select\_dtypes(exclude=np.number).columns.tolist()

preprocessor = ColumnTransformer(

transformers=[

('num', StandardScaler(), numerical\_features),

('cat', OneHotEncoder(handle\_unknown='ignore'), categorical\_features)

])

* **Explicação:**
  1. Separamos as colunas em X (as características que usamos para prever) e y (o que queremos prever: risco\_credito).
  2. O ColumnTransformer é uma ferramenta poderosa que aplica diferentes tratamentos a diferentes colunas.
  3. Para as colunas numéricas (idade, renda\_mensal, etc.), aplicamos o StandardScaler, que coloca todos os números na mesma escala de importância.
  4. Para as colunas de texto (cargo\_atual, possui\_imovel, etc.), aplicamos o OneHotEncoder, que converte o texto em colunas numéricas de 0s e 1s, um formato que o modelo consegue entender.

**4.2. Divisão, Treino e Avaliação**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y\_encoded, test\_size=0.2, random\_state=42)

accuracy = accuracy\_score(y\_test, predictions)

print(f"Acurácia do modelo: {accuracy\*100:.2f}%")

* **Explicação:**
  1. **train\_test\_split**: Dividimos os dados em 80% para treino e 20% para teste. O modelo aprende com os 80% e depois é avaliado com os 20% que nunca viu, garantindo um teste justo.
  2. **Treino (.fit)**: Treinamos vários modelos (como RandomForestClassifier) para que eles aprendam os padrões nos dados de treino.
  3. **Avaliação (accuracy\_score)**: Comparamos as previsões que o modelo fez para os dados de teste com as respostas corretas. A **acurácia** mede a percentagem de vezes que o modelo acertou.

#### **5. Conclusão e Resultados**

O código executa um fluxo de trabalho completo, transformando um ficheiro CSV bruto numa solução de análise inteligente. Ele não só limpa e enriquece os dados, como também testa vários algoritmos para encontrar o mais preciso na tarefa de prever o risco de crédito. O resultado final é um modelo treinado e validado, pronto para ser integrado nos sistemas do Banco Box para auxiliar na tomada de decisões.