Plano Detalhado: Previsão de Churn com Python (Dataset Externo)

Objetivo Inicial

Iniciar a estruturação da previsão de churn utilizando Python e um dataset externo (CSV) para desenvolvimento e testes do pipeline completo de Machine Learning. Após validação, adaptar para uso com dados do banco PostgreSQL.

Etapas Detalhadas do Projeto

1. Escolher dataset externo (ex: Telco Customer Churn da IBM, disponível no Kaggle).
2. Analisar estrutura do dataset e definir coluna alvo (`Churn`).
3. Limpar e preparar os dados para modelagem (tratamento de nulos, encoding, normalização).
4. Criar Jupyter Notebook com estrutura de pipeline:
   - Carregamento e pré-processamento
   - Análise exploratória (EDA)
   - Separação treino/teste
   - Modelagem com 3 algoritmos
   - Avaliação de desempenho
5. Exportar melhor modelo (.pkl)
6. Criar script Python que recebe JSON com dados e retorna predição
7. Planejar futura integração com backend PHP e PostgreSQL.

Stack Python e Bibliotecas

- pandas: manipulação de dados
- numpy: operações numéricas
- matplotlib / seaborn: gráficos e análise exploratória
- scikit-learn: modelagem (LogisticRegression, RandomForest)
- xgboost: Gradient Boosting
- joblib: salvar e carregar modelos
- json: entrada/saída estruturada
- Jupyter Notebook: desenvolvimento inicial

Passo a Passo com Código (resumo)

1. Importar bibliotecas:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
```

2. Carregar dataset:

```
df = pd.read_csv('churn_dataset.csv')
```

Plano Detalhado: Previsão de Churn com Python (Dataset Externo)

3. Pré-processar:
```
df.dropna(inplace=True)
df['Churn'] = df['Churn'].map({'Yes':1, 'No':0})
```

4. Separar features e target:
```
X = df.drop('Churn', axis=1)
y = df['Churn']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

5. Treinar modelo:
```
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

6. Avaliar:
```
y_pred = model.predict(X_test)
print(classification_report(y_test, y_pred))
```

7. Exportar modelo:
```
import joblib
joblib.dump(model, 'modelo_churn.pkl')
```

Organização Inicial do Projeto (estrutura de pastas)

/python/churn/
??? churn_dataset.csv
??? train_model.ipynb
??? modelo_churn.pkl
??? predict_churn.py
??? requirements.txt

Próximos Passos Pós-Validação

1. Validar modelo com dataset externo
2. Criar versão simplificada do script para integração com PHP
3. Adaptar estrutura de dados do PostgreSQL para ter colunas compatíveis com o modelo
4. Criar versão final com dados reais do sistema SaaS
5. Conectar previsões ao Power BI para visualização executiva