**1. Instalar o VSCode e Extensões Necessárias**

Se ainda não tiver o VSCode instalado, baixe e instale-o em: [VSCode](https://code.visualstudio.com/).

Além disso, é recomendado instalar as seguintes extensões para melhorar a experiência no desenvolvimento com Python e GitHub Copilot:

* **Python** (extensão oficial da Microsoft para Python).
* **GitHub Copilot** (para sugestões de código automáticas).

Para instalar o GitHub Copilot, siga as instruções neste [link](https://docs.github.com/en/copilot/getting-started-with-github-copilot).

**2. Criar a Estrutura de Pastas**

Abra o VSCode e crie a estrutura de pastas e arquivos do projeto:

1. Crie a pasta github\_copilot\_demo.
2. Dentro dessa pasta, crie as subpastas data/, scripts/, e models/.
3. Crie os arquivos train\_model.py, evaluate\_model.py, main.py, requirements.txt e README.md conforme os exemplos que vou fornecer.

**3. Instalar o Python e as Dependências**

No VSCode, abra o terminal integrado (Ctrl + `) e siga os seguintes passos:

1. **Instalar o Python**:
   * Certifique-se de ter o Python instalado no seu sistema. Se não, baixe e instale-o a partir do [site oficial do Python](https://www.python.org/downloads/).
2. **Criar um Ambiente Virtual**:
   * Abra o terminal no VSCode e navegue até a pasta do projeto (github\_copilot\_demo).
   * Crie um ambiente virtual:

bash

CopiarEditar

python -m venv venv

1. **Ativar o Ambiente Virtual**:
   * **Windows**:

bash

CopiarEditar

.\venv\Scripts\activate

* + **Mac/Linux**:

bash

CopiarEditar

source venv/bin/activate

1. **Instalar as Dependências**:
   * Crie um arquivo requirements.txt com as seguintes dependências:

txt

CopiarEditar

pandas

scikit-learn

joblib

* + No terminal do VSCode, execute o comando para instalar as dependências:

bash

CopiarEditar

pip install -r requirements.txt

**4. Criar os Arquivos do Projeto**

Com a estrutura pronta e as dependências instaladas, agora você pode criar os arquivos:

**Arquivo data/churn\_data.csv:**

Salve o seguinte CSV na pasta data/:

csv

CopiarEditar

CustomerID,Age,Income,Tenure,Churn

1,45,50000,5,0

2,34,60000,3,1

3,23,45000,1,0

4,50,80000,8,1

5,29,70000,4,0

6,36,40000,2,1

7,48,90000,10,0

8,32,55000,6,1

9,27,35000,3,0

10,41,62000,7,1

**Arquivo scripts/train\_model.py:**

python

CopiarEditar

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score

import joblib

def load\_data(filepath):

"""Carrega os dados do arquivo."""

return pd.read\_csv(filepath)

def preprocess\_data(df):

"""Pré-processa os dados."""

df = df.dropna()

X = df.drop("Churn", axis=1).drop("CustomerID", axis=1)

y = df["Churn"]

return X, y

def train\_model(X\_train, y\_train):

"""Treina o modelo RandomForest."""

model = RandomForestClassifier(random\_state=42)

model.fit(X\_train, y\_train)

return model

def save\_model(model, filepath):

"""Salva o modelo treinado."""

joblib.dump(model, filepath)

if \_\_name\_\_ == "\_\_main\_\_":

# Carrega os dados

data = load\_data("data/churn\_data.csv")

# Pré-processa os dados

X, y = preprocess\_data(data)

# Divide os dados em treino e teste

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Treina o modelo

model = train\_model(X\_train, y\_train)

# Avalia o modelo

predictions = model.predict(X\_test)

print(f"Accuracy: {accuracy\_score(y\_test, predictions)}")

# Salva o modelo

save\_model(model, "models/model.pkl")

**Arquivo scripts/evaluate\_model.py:**

python

CopiarEditar

import pandas as pd

import joblib

from sklearn.metrics import classification\_report

def load\_data(filepath):

"""Carrega os dados do arquivo."""

return pd.read\_csv(filepath)

def preprocess\_data(df):

"""Pré-processa os dados."""

df = df.dropna()

X = df.drop("Churn", axis=1).drop("CustomerID", axis=1)

y = df["Churn"]

return X, y

def evaluate\_model(model, X\_test, y\_test):

"""Avalia o modelo com base nos dados de teste."""

predictions = model.predict(X\_test)

print("Relatório de Classificação:")

print(classification\_report(y\_test, predictions))

if \_\_name\_\_ == "\_\_main\_\_":

# Carrega os dados e o modelo treinado

data = load\_data("data/churn\_data.csv")

model = joblib.load("models/model.pkl")

# Pré-processa os dados

X, y = preprocess\_data(data)

# Divide os dados em treino e teste

from sklearn.model\_selection import train\_test\_split

\_, X\_test, \_, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Avalia o modelo

evaluate\_model(model, X\_test, y\_test)

**Arquivo main.py:**

python

CopiarEditar

from scripts.train\_model import load\_data, preprocess\_data, train\_model, save\_model

def main():

# Caminho para os dados

data\_path = "data/churn\_data.csv"

# Carrega os dados

data = load\_data(data\_path)

# Pré-processa os dados

X, y = preprocess\_data(data)

# Treina o modelo

model = train\_model(X, y)

# Salva o modelo

save\_model(model, "models/final\_model.pkl")

print("Modelo treinado e salvo com sucesso!")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Arquivo requirements.txt:**

txt

CopiarEditar

pandas

scikit-learn

joblib

**5. Executar o Projeto no VSCode**

Agora, você pode rodar os scripts diretamente no terminal integrado do VSCode.

1. **Treinar o Modelo**: No terminal, digite:

bash

CopiarEditar

python scripts/train\_model.py

1. **Avaliar o Modelo**: No terminal, digite:

bash

CopiarEditar

python scripts/evaluate\_model.py

1. **Rodar o Script Principal**: No terminal, digite:

bash

CopiarEditar

python main.py

**6. Como Usar o GitHub Copilot no VSCode**

* **Autocompletar Código**: O GitHub Copilot irá sugerir automaticamente trechos de código enquanto você digita. Se uma sugestão for boa, você pode pressionar **Tab** para aceitá-la.
* **Comentários e Sugestões**: Escreva um comentário descrevendo a lógica do que você quer fazer e o Copilot irá sugerir o código correspondente. Por exemplo:

python

CopiarEditar

# Criação de uma função para treinar o modelo de classificação

def train\_classifier(X, y):

# Código sugerido automaticamente pelo Copilot

**7. Commit no GitHub (Opcional)**

Caso você queira versionar o código no GitHub, siga esses passos:

1. **Inicialize o repositório Git**:

bash

CopiarEditar

git init

1. **Adicione todos os arquivos ao repositório**:

bash

CopiarEditar

git add .

1. **Faça o commit**:

bash

CopiarEditar

git commit -m "Primeiro commit do projeto"

1. **Crie o repositório no GitHub** e faça o push:

bash

CopiarEditar

git remote add origin <url\_do\_repositorio>

git push -u origin master