import pandas as pd

df = pd.read\_csv('/data/ecommerce\_tratados.csv')

# Verifica a quantidade de dados únicos em cada coluna

unicos = df.nunique()

print('Análise de dados únicos:\n', unicos)

# .nunique() conta quantos valores distintos existem em cada coluna

# Calcula estatísticas descritivas das colunas numéricas

estatisticas = df.describe()

print('Estatísticas dos dados:\n', estatisticas)

# .describe() retorna estatísticas como média, desvio padrão, mínimo e máximo das colunas numéricas

# Criar o campo Qtd\_Vendidos\_Cod convertendo a coluna Qtd\_Vendidos para números

def transform\_qtd\_vendidos(qtd):

    """

    Converte os valores da coluna Qtd\_Vendidos para números inteiros, de acordo com a grandeza informada no dataset.

    """

    mapping = {

        'Nenhum': 0, '1': 1, '2': 2, '3': 3, '4': 4, '+5': 5, '+25': 25,

        '+50': 50, '+100': 100, '+1000': 1000, '+10mil': 10000, '+50mil': 50000

    }

    return mapping.get(qtd, None)  # Retorna o número correspondente ou None caso o valor não esteja no dicionário

df['Qtd\_Vendidos\_Cod'] = df['Qtd\_Vendidos'].apply(transform\_qtd\_vendidos)

# Aplica a função transform\_qtd\_vendidos à coluna Qtd\_Vendidos para converter os valores

# Criar o campo Marca\_Freq, que representa a frequência relativa de cada marca

marca\_freq = df['Marca'].value\_counts() / len(df)

# Conta quantas vezes cada marca aparece no dataset e divide pelo total de registros para obter a frequência relativa

df['Marca\_Freq'] = df['Marca'].map(marca\_freq)

# Mapeia cada marca para sua respectiva frequência e armazena na nova coluna Marca\_Freq

# Criar o campo Material\_Freq, que representa a frequência relativa de cada material

material\_freq = df['Material'].value\_counts() / len(df)

# Conta quantas vezes cada tipo de material aparece no dataset e divide pelo total de registros

df['Material\_Freq'] = df['Material'].map(material\_freq)

# Mapeia cada material para sua respectiva frequência e armazena na nova coluna Material\_Freq

# Exibe as primeiras linhas do DataFrame para verificar as alterações

print("DataFrame após as transformações:")

print(df.head())

criado com sucesso.")

else:

    print("Erro: O campo 'Temporada' não foi encontrado no DataFrame.")

import pandas as pd

from sklearn.preprocessing import MinMaxScaler, LabelEncoder

# Carregar os dados do arquivo CSV

df = pd.read\_csv('/data/ecommerce\_tratados\_ex2.csv')

# Inicializar MinMaxScaler e LabelEncoder

scaler = MinMaxScaler()

label\_encoder = LabelEncoder()

# Transformar o campo Nota para escala de 0 a 1

if 'Nota' in df.columns:

    df['Nota\_MinMax'] = scaler.fit\_transform(df[['Nota']])

    print("Campo 'Nota\_MinMax' criado com sucesso.")

else:

    print("Erro: O campo 'Nota' não foi encontrado no DataFrame.")

# Transformar o campo N\_Avaliacoes para escala de 0 a 1

if 'N\_Avaliacoes' in df.columns:  # Nota: corrigido o nome da coluna

    df['N\_Avaliacoes\_MinMax'] = scaler.fit\_transform(df[['N\_Avaliacoes']])

    print("Campo 'N\_Avaliacoes\_MinMax' criado com sucesso.")

else:

    print("Erro: O campo 'N\_Avaliacoes' não foi encontrado no DataFrame.")

# Transformar o campo Desconto para escala de 0 a 1

if 'Desconto' in df.columns:

    df['Desconto\_MinMax'] = scaler.fit\_transform(df[['Desconto']])

    print("Campo 'Desconto\_MinMax' criado com sucesso.")

else:

    print("Erro: O campo 'Desconto' não foi encontrado no DataFrame.")

# Transformar o campo Preco para escala de 0 a 1

if 'Preco' in df.columns:

    df['Preco\_MinMax'] = scaler.fit\_transform(df[['Preco']])

    print("Campo 'Preco\_MinMax' criado com sucesso.")

else:

    print("Erro: O campo 'Preco' não foi encontrado no DataFrame.")

# Codificar o campo Marca para numérico

if 'Marca' in df.columns:

    df['Marca\_Cod'] = label\_encoder.fit\_transform(df['Marca'])

    print("Campo 'Marca\_Cod' criado com sucesso.")

else:

    print("Erro: O campo 'Marca' não foi encontrado no DataFrame.")

# Codificar o campo Material para numérico

if 'Material' in df.columns:

    df['Material\_Cod'] = label\_encoder.fit\_transform(df['Material'])

    print("Campo 'Material\_Cod' criado com sucesso.")

else:

    print("Erro: O campo 'Material' não foi encontrado no DataFrame.")

# Codificar o campo Temporada para numérico

if 'Temporada' in df.columns:

    df['Temporada\_Cod'] = label\_encoder.fit\_transform(df['Temporada'])

    print("Campo 'Temporada\_Cod'

# Exibir as primeiras linhas do DataFrame para verificar as transformações

print("\nDataFrame com os campos transformados:")

print(df.head())

import pandas as pd

df = pd.read\_csv('/data/ecommerce\_tratados.csv')

# Verifica a quantidade de dados únicos em cada coluna

unicos = df.nunique()

print('Análise de dados únicos:\n', unicos)

# .nunique() conta quantos valores distintos existem em cada coluna

# Calcula estatísticas descritivas das colunas numéricas

estatisticas = df.describe()

print('Estatísticas dos dados:\n', estatisticas)

# .describe() retorna estatísticas como média, desvio padrão, mínimo e máximo das colunas numéricas

# Criar o campo Qtd\_Vendidos\_Cod convertendo a coluna Qtd\_Vendidos para números

def transform\_qtd\_vendidos(qtd):

    """

    Converte os valores da coluna Qtd\_Vendidos para números inteiros, de acordo com a grandeza informada no dataset.

    """

    mapping = {

        'Nenhum': 0, '1': 1, '2': 2, '3': 3, '4': 4, '+5': 5, '+25': 25,

        '+50': 50, '+100': 100, '+1000': 1000, '+10mil': 10000, '+50mil': 50000

    }

    return mapping.get(qtd, None)  # Retorna o número correspondente ou None caso o valor não esteja no dicionário

df['Qtd\_Vendidos\_Cod'] = df['Qtd\_Vendidos'].apply(transform\_qtd\_vendidos)

# Aplica a função transform\_qtd\_vendidos à coluna Qtd\_Vendidos para converter os valores

# Criar o campo Marca\_Freq, que representa a frequência relativa de cada marca

marca\_freq = df['Marca'].value\_counts() / len(df)

# Conta quantas vezes cada marca aparece no dataset e divide pelo total de registros para obter a frequência relativa

df['Marca\_Freq'] = df['Marca'].map(marca\_freq)

# Mapeia cada marca para sua respectiva frequência e armazena na nova coluna Marca\_Freq

# Criar o campo Material\_Freq, que representa a frequência relativa de cada material

material\_freq = df['Material'].value\_counts() / len(df)

# Conta quantas vezes cada tipo de material aparece no dataset e divide pelo total de registros

df['Material\_Freq'] = df['Material'].map(material\_freq)

# Mapeia cada material para sua respectiva frequência e armazena na nova coluna Material\_Freq

# Exibe as primeiras linhas do DataFrame para verificar as alterações

print("DataFrame após as transformações:")

print(df.head())