Explicação do Processamento de Dados CSV

Arthur Trindade

Workshop Dados - Fábrica de Software

Este arquivo vai detalhar as etapas de tratamento e limpeza de dados realizadas em um arquivo CSV fornecido pelos instrutores Pedro e Igor, dados_nao_tratados.csv, e tendo como resultado a geração de um arquivo CSV processado e tratado, dados_tratados.csv.

1. Importação de Dados

Os dados foram inicialmente carregados do arquivo dados_nao_tratados.csv utilizando a biblioteca pandas para inspeção e manipulação subsequente.

```
# Fazendo a importação com pandas para mostrar o CSV
import pandas as pd

df = pd.read_csv('/content/dados_nao_tratados.csv')
display(df)
```

2. Tratamento da Coluna "idade"

A coluna "idade" foi convertida para um formato numérico, com valores não numéricos sendo coercidos para NaN (Not a Number).

```
# Transformando a coluna idade que tinha dados em texto para aceitar
apenas dados numéricos e os que não estão ficam nulos
df["idade"] = pd.to_numeric(df["idade"], errors='coerce')
df["idade"] = df["idade"].astype('Int64')
display(df)
```

3. Tratamento da Coluna "data_inscricao"

A coluna "data_inscricao" foi convertida para o formato de data e hora. Quaisquer valores que não puderam ser convertidos foram definidos como NaN.

```
# Deixando a coluna de data de inscrição no formato exato de data e os que não estão ficaram nulos
df['data_inscricao'] = pd.to_datetime(df['data_inscricao'], errors='coerce')
display(df)
```

4. Tratamento da Coluna "nota"

A coluna "nota" foi convertida para um formato de ponto flutuante (Float64), com valores não numéricos sendo convertidos para NaN.

```
# Deixando a coluna de notas aceitando apenas valores em float e os que não estão ficaram nulos
df["nota"] = pd.to_numeric(df["nota"], errors='coerce')
df["nota"] = df["nota"].astype('Float64')
display(df)
```

5. Tratamento da Coluna "ativo"

A coluna "ativo", que continha valores textuais como "sim" e "não", foi mapeada para o tipo booleano (True/False). Foi colocado um reconhecimento de diversas variações ('true', 'yes', 'nao') que estavam presentes no csv não tratado. Valores que não se encaixavam no mapeamento foram definidos como NaN.

```
# Ajeitando os sim e não para formato especificio booleano e os que não estão ficaram nulos
df["ativo"] = df["ativo"].fillna('').str.strip().str.lower().map({'sim': True, 'true': True, 'yes': True, 'não': False, 'false': False, 'nao': False})
display(df)
```

6. Tratamento da Coluna "idade"

O tipo de dado da coluna foi ajustado para inteiro (Int64). Valores nulos restantes foram preenchidos com o valor 1.

```
# Definindo o número 1 para preencher os valores nulos da coluna idade
df['idade'] = df['idade'].fillna(1)
display(df)
```

7. Tratamento da Coluna "nota"

Valores nulos restantes foram preenchidos com o valor 1.1.

```
# Definindo o número 1.1 para preencher os valores nulos da coluna nota
df['nota'] = df['nota'].fillna(1.1)
display(df)
```

8. Tratamento da Coluna "data_inscricao"

Os valores nulos restantes foram preenchidos com o valor 9.

```
# Definindo o valor 9 para preencher os valores nulos da coluna
data_inscrição
df['data_inscricao'] = df['data_inscricao'].fillna(9)
display(df)
```

9. Tratamento da Coluna "ativo"

Os valores nulos ou vazios restantes foram preenchidos com a string "Sem dados".

```
# Definindo 'Sem Dados' para valores vazios na coluna ativos
df['ativo'] = df['ativo'].fillna('Sem dados')
display(df)
```

10. Geração e Exportação do CSV Final

Após todas as etapas, o DataFrame processado foi exportado para um novo arquivo CSV, dados_tratados.csv.

```
# Gerando o novo arquivo de dados tratados em csv
df.to_csv("dados_tratados.csv", index=False)
```

11. Download do Arquivo Processado

O arquivo dados_tratados.csv foi feito o download, concluindo o processo de tratamento de dados.

```
# Baixando para o pc o novo csv de dados tratados
from google.colab import files
files.download("dados_tratados.csv")
```