Trabalho Prático – DGT2823

Aluno: Eli Sidney Bicalho Santos
Disciplina: Tecnologias para desenvolvimento de soluções de Big Data
Data: 29/09/2025

Objetivos

- Ler arquivos CSV usando Pandas (Python)
- Criar subconjuntos de dados a partir de conjuntos existentes
- Configurar número máximo de linhas exibidas
- Exibir primeiras e últimas “N” linhas de um DataFrame
- Obter informações gerais sobre colunas, linhas e tipos de dados
- Limpeza e tratamento de dados para análise

Microatividade 1 – Ler o arquivo CSV

Código principal:

```python
import pandas as pd
import numpy as np

df = pd.read_csv(
    'dataset.csv',
    header=None,
    names=['all_data'],
    engine='python',
    encoding='utf-8'
)
df = df['all_data'].str.split(';', expand=True)
df.columns = ['ID', 'Duration', 'Date', 'Pulse', 'Maxpulse', 'Calories']
df['Date'] = df['Date'].str.replace("'", "")
numeric_cols = ['ID', 'Duration', 'Pulse', 'Maxpulse', 'Calories']
for col in numeric_cols:
    df[col] = pd.to_numeric(df[col], errors='coerce')
df = df[df['ID'].notna()]
print(df.head())
```

Saída esperada (primeiras linhas):

ID	Duration	Date	Pulse	Maxpulse	Calories
0	60	2020/12/01	110	130	4091
1	60	2020/12/02	117	145	4790
2	60	2020/12/03	103	135	3400
3	45	2020/12/04	109	175	2824
4	45	2020/12/05	117	148	4060

Microatividade 2 – Criar um subconjunto de dados

Código:

```python
sub_df = df[['ID', 'Date', 'Calories']]
print(sub_df.head())
```

Saída esperada (primeiras linhas):

ID	Date	Calories
0	2020/12/01	4091
1	2020/12/02	4790
2	2020/12/03	3400
3	2020/12/04	2824
4	2020/12/05	4060

Microatividade 3 – Configurar número máximo de linhas exibidas

Código:

```python
pd.set_option('display.max_rows', 9999)
print(df.to_string())
```

Descrição: O DataFrame completo é exibido na tela, sem truncamento de linhas.

Microatividade 4 – Exibir primeiras e últimas 10 linhas

Código:

```python
print(df.head(10))  # primeiras 10 linhas
print(df.tail(10))  # últimas 10 linhas
```

Descrição: Permite visualizar rapidamente o início e o fim do DataFrame.

Microatividade 5 – Informações gerais

Código:

```python
print(df.info())
```

Saída esperada:

- Total de linhas e colunas
- Quantidade de valores nulos por coluna
- Tipos de dados (int64, float64, object)
- Uso de memória do DataFrame

Tratamento de dados

Código:

```python
clean_df = df.copy()
clean_df['Calories'].fillna(0, inplace=True)
clean_df['Date'].fillna('1900/01/01', inplace=True)
clean_df['Date'].replace('1900/01/01', np.nan, inplace=True)
clean_df['Date'] = clean_df['Date'].astype(str).replace('20201226', '2020/12/26')
clean_df['Date'] = pd.to_datetime(clean_df['Date'], errors='coerce')
clean_df.dropna(subset=['Date'], inplace=True)
clean_df.to_csv('dataset_limpo.csv', sep=';', index=False)
```

Mudanças aplicadas:

- Valores nulos de Calories substituídos por 0
- Datas inválidas substituídas e convertidas para datetime
- Removidos registros com Date nulo

DataFrame final (exemplo de algumas linhas)

ID	Duration	Date	Pulse	Maxpulse	Calories
0	60	01/12/2020	110	130	4091
1	60	02/12/2020	117	145	4790
2	60	03/12/2020	103	135	3400
3	45	04/12/2020	109	175	2824
18	45	18/12/2020	90	112	0
28	60	28/12/2020	103	132	0

Conclusão

- Todos os valores nulos foram tratados adequadamente
- Datas foram ajustadas para o formato datetime correto
- O dataset está limpo e pronto para análise futura
- O trabalho demonstrou domínio de leitura de CSV, criação de subconjuntos, configuração de visualização, extração de informações gerais e limpeza de dados usando Pandas