**Para saber mais: erro de encoding - Como resolver?**

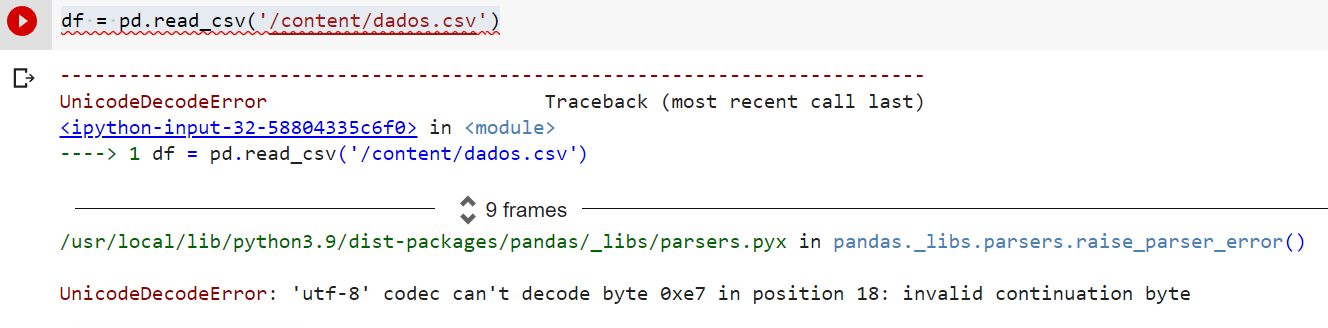
Giovanna é uma cientista de dados que trabalha em uma empresa de comércio eletrônico. Ela é responsável por analisar os dados de vendas para identificar padrões e tendências que possam ajudar a empresa a tomar decisões informadas.

Recentemente, ela recebeu uma nova tarefa: analisar um arquivo CSV contendo informações sobre as vendas da empresa em um determinado período. Para isso, Giovanna tentou fazer a leitura do um arquivo CSV no Google Colab, utilizando a biblioteca Pandas com o seguinte comando:

**import** pandas as **pd**

df = pd.read\_csv('/content/dados.csv')Copiar código

Porém, surgiu o erro abaixo:



Em português a mensagem quer dizer **UnicodeDecodeError: o codec 'utf-8' não pode decodificar o byte 0xe7 na posição 18: byte de continuação inválido**

**Mas o que isso quer dizer? O que é esse tal de UTF-8?**

O erro de *encoding* ocorre quando a biblioteca Pandas não consegue interpretar corretamente os caracteres de um arquivo CSV. Isso pode acontecer quando ele contém caracteres especiais que não são reconhecidos pela biblioteca Pandas ou quando foi salvo em um formato de codificação diferente do esperado.

Para resolver esse erro, é necessário identificar a codificação correta do arquivo CSV e especificá-la ao carregar o arquivo com a biblioteca Pandas. Essa codificação padrão é o **UTF-8**, mas em alguns casos, o arquivo pode ter sido salvo com uma codificação diferente, como ISO-8859-1.

O UTF-8 é uma codificação de caracteres universal usada para representar caracteres de diferentes idiomas de forma compatível com a internet e com sistemas de computador em geral. A sigla UTF significa *Unicode Transformation Format* (Formato de Transformação Unicode) e o número 8 indica que essa codificação associa uma sequência de 1 a 4 bytes (8 a 32 bits) com cada caractere.

A codificação UTF-8 é amplamente utilizada na internet e em sistemas de computador em todo o mundo, pois permite a representação de caracteres de diferentes idiomas em um único conjunto de caracteres. Além disso, essa codificação é capaz de preservar a compatibilidade com outras mais antigas, como ASCII, o que a torna uma escolha popular para a criação e compartilhamento de arquivos de texto.

Pode ser que você esteja pensando neste momento: como será que a Giovanna conseguiu descobrir qual é a codificação do arquivo que ela está tentando ler?

Existem algumas formas para descobrir isso, no entanto, teremos a oportunidade de experimentar uma maneira prática de fazer isso no próprio Google Colab. Vamos lá?

Nós podemos usar uma biblioteca chamada [chardet](https://pypi.org/project/chardet/" \t "_blank) para detectar o *encoding* de um arquivo CSV. Para utilizar essa biblioteca no Google Colab, basta realizar sua importação:

**import** chardet

Em seguida, digitamos o seguinte bloco de código:

**with** open('/content/dados.csv', 'rb') **as** file:

print(chardet.detect(file.read()))Copiar código

Note que a primeira linha deste código abre o arquivo CSV em modo de leitura binária rb e atribui o objeto de arquivo retornado a uma variável chamada file.

Na segunda linha do código, o conteúdo do arquivo CSV é lido usando o método read() e o resultado é passado para a função chardet.detect(), que retorna um dicionário contendo informações sobre a codificação mais provável do arquivo. O resultado é impresso com a função print() que mostra a codificação e a confiança associada a essa codificação.

Ao executar o código Giovanna obteve o seguinte resultado:

{'encoding': 'ISO-8859-1', 'confidence': 0.73, 'language': ''}

Agora ela sabe que o arquivo CSV provavelmente possui a codificação **ISO-8859-1** com uma confiança de 0.73. Para especificar a codificação correta ao carregar o arquivo CSV com a biblioteca Pandas, é possível usar um parâmetro chamado encoding:

df = pd.read\_csv('/content/dados.csv', encoding='ISO-8859-1')

Desta forma, a biblioteca Pandas irá carregar o arquivo CSV usando a codificação ISO-8859-1, o que vai resolver o erro de *encoding*. Então, se algum dia você tentar ler um arquivo e surgir esse mesmo erro, lembre-se dessa dica!

Caso queira aprofundar o assunto, deixamos alguns links para materiais que foram utilizados como referência:

* [Instituto de Matemática e Estatística - IME/USP: Unicode e UTF-8](https://www.ime.usp.br/~pf/algoritmos/apend/unicode.html#detection)
* [IBM: O que é Unicode](https://www.ibm.com/docs/pt-br/workload-automation/9.3.0?topic=support-what-is)
* [Kaggle:Character Encodings](https://www.kaggle.com/code/alexisbcook/character-encodings/tutorial)

# **Para saber mais: explorando a documentação**

A biblioteca Pandas é uma das ferramentas mais poderosas disponíveis para a análise de dados em Python. Ela oferece uma ampla variedade de funcionalidades que tornam o processo de manipulação e análise de dados muito mais fácil e eficiente.

Para aproveitar ao máximo todas as funcionalidades da biblioteca, é essencial explorar a sua documentação, que é extensa e bem organizada, e pode ser facilmente acessada no modo on-line. Confira as indicações:

* [Função read\_csv](https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html)
* [Função to\_csv](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.to_csv.html)

Explorar a documentação da biblioteca é importante por diversas razões. Ela pode ajudar a entender como as funções e métodos da biblioteca funcionam e como elas podem ser usadas para analisar e manipular os dados.

Além disso, a documentação fornece detalhes sobre os parâmetros de cada função e exemplos de como eles podem ser usados na prática. Então, busque aprofundar os seus estudos nesses parâmetros das funções read\_csv e to\_csv e aprenda mais sobre cada um deles.