Atividade 02 - Machine Learning

Elton Jorge

# Questão 1

Escreva uma função que receba uma lista de números e retorne outra lista com os números ímpares.

def filtrar\_impares(lista):  
 impares = [n for n in lista if n % 2 != 0]  
 if impares:  
 return impares  
 else:  
 return "Não há numeros impares"

# Questão 2

Escreva uma função que receba uma lista de números e retorne outra lista com os números primos presentes.

def eh\_primo(num):  
 if num < 2:  
 return False  
 for i in range(2, int(num \*\* 0.5) + 1):  
 if num % i == 0:  
 return False  
 return True  
  
def filtrar\_primos(lista):  
 return [n for n in lista if eh\_primo(n)]

# Questão 3

Escreva uma função que receba duas listas e retorne outra lista com os elementos que estão presentes em apenas uma das listas.

def diferenca\_listas(lista1, lista2):  
 if set(lista1) == set(lista2):  
 return "As duas listas são iguais"  
 else:  
 return [x for x in lista1 if x not in lista2] + [y for y in lista2 if y not in lista1]

# Questão 4

Dada uma lista de números inteiros, escreva uma função para encontrar o segundo maior valor na lista.

def segundo\_maior(lista):  
 lista.sort(reverse=True) # ordenando em ordem decrescente  
 return lista[1] if len(lista) > 1 else None

# Questão 5

Crie uma função que receba uma lista de tuplas, cada uma contendo o nome e a idade de uma pessoa, e retorne a lista ordenada pelo nome das pessoas em ordem alfabética.

def ordenar\_por\_nome(lista\_tuplas):  
 return sorted(lista\_tuplas, key=lambda x: x[0])  
  
ordenar\_por\_nome([("Maria", 25), ("Ana", 30), ("João", 22)])

[('Ana', 30), ('João', 22), ('Maria', 25)]

# Questão 6

Como identificar e tratar outliers em uma coluna numérica usando desvio padrão ou quartis?

Desvio padrão: calcula-se a média e o desvio padrão da coluna. Valores que estão muito acima ou abaixo da média, normalmente 2 ou 3 desvios padrões, são considerados outliers. Esses valores podem ser removidos ou substituídos pela média ou mediana. Quartis (IQR): calcula-se o 1º e 3º quartis da coluna e o intervalo interquartílico (IQR = Q3 − Q1). Valores abaixo de Q1 − 1,5×IQR ou acima de Q3 + 1,5×IQR são considerados outliers. Eles também podem ser removidos ou substituídos por valores próximos aos limites

import pandas as pd  
  
df = pd.DataFrame({"valores": [1, 2, 2, 3, 2, 100]})  
Q1 = df["valores"].quantile(0.25)  
Q3 = df["valores"].quantile(0.75)  
IQR = Q3 - Q1  
df\_filtrado = df[(df["valores"] >= Q1 - 1.5\*IQR) & (df["valores"] <= Q3 + 1.5\*IQR)]  
df\_filtrado

|  | valores |
| --- | --- |
| 0 | 1 |
| 1 | 2 |
| 2 | 2 |
| 3 | 3 |
| 4 | 2 |

# Questão 7

Como concatenar vários DataFrames (empilhando linhas ou colunas), mesmo que tenham colunas diferentes? Dica: Utiliza-se pd.concat() especificando axis=0 (linhas) ou axis=1 (colunas). Quando há colunas diferentes, os valores ausentes são preenchidos com NaN.

df1 = pd.DataFrame({"A": [1, 2], "B": [3, 4]})  
df2 = pd.DataFrame({"A": [5, 6], "C": [7, 8]})  
df\_concat = pd.concat([df1, df2], axis=0)  
df\_concat

|  | A | B | C |
| --- | --- | --- | --- |
| 0 | 1 | 3.0 | NaN |
| 1 | 2 | 4.0 | NaN |
| 0 | 5 | NaN | 7.0 |
| 1 | 6 | NaN | 8.0 |

# Questão 8

Utilizando pandas, como realizar a leitura de um arquivo CSV em um DataFrame e exibir as primeiras linhas?

try:  
 df = pd.read\_csv("arquivo.csv") # troque o arquivo.csv, pelo caminho do arquivo e o tipo  
 df.head()  
except Exception as e:  
 print(f"erro ao baixar o arquivo{e}")

erro ao baixar o arquivo[Errno 2] No such file or directory: 'arquivo.csv'

# Questão 9

Utilizando pandas, como selecionar uma coluna específica e filtrar linhas em um “DataFrame” com base em uma condição?

try:  
 coluna = df["coluna\_exemplo"]  
 filtro = df[df["coluna\_exemplo"] > 10]  
 filtro  
except Exception as e:  
 print(f"erro{e}")

erro'coluna\_exemplo'

# Questão 10

Utilizando pandas, como lidar com valores ausentes (NaN) em um DataFrame?

A diversas formas de tratar valores ausentes(NaN), se as colunas forem númerica e fazer sentido para os negocios, poderia ser substituido pela média ou mediana. a primeira parte e verificar com a parte solicitante. mas em muitos casos, teremos que excluir esses dados.

df\_sem\_nan = df.dropna() # remove linhas com NaN

df\_preenchido = df.fillna(df.mean(numeric\_only=True)) # preenche com média

df\_preenchido = df.fillna(df.median(numeric\_only=True)) # preenche com média