

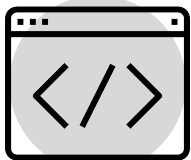
Profissão: Cientista de Dados



BOAS PRÁTICAS



Elementos básicos de Python



● Pacotes e o numpy

● NDArray

● Gere dados

● Construa funções



Pacotes e o numpy

- Ao importar bibliotecas, use apelidos curtos e comuns para facilitar a referência posterior. Por exemplo, 'numpy' é geralmente importado como 'np'.
- Evite importar todas as funções de uma biblioteca de uma vez usando o comando 'from library import *'. Isso pode sobrescrever funções padrão e causar conflitos.



Pacotes e o numpy

- Ao usar bibliotecas não padrão, certifique-se de instalá-las corretamente usando o comando 'pip' no CMD. Lembre-se de que o 'pip' busca a versão mais recente de uma biblioteca quando você a instala.
- Ao trabalhar com grandes volumes de dados, considere usar arrays numpy em vez de listas padrão do Python. Arrays numpy são mais eficientes para operações matemáticas em grandes volumes de dados.



Pacotes e o numpy

- Ao criar arrays numpy a partir de listas, certifique-se de que todos os elementos da lista são do mesmo tipo. Arrays numpy só podem armazenar dados do mesmo tipo.





NDArray

- Ao criar um NDArray a partir de uma lista com diferentes tipos de dados, esteja ciente de que o tipo de dado "mais forte" prevalecerá. Portanto, se você deseja um NDArray com um tipo de dado específico, certifique-se de que todos os elementos da lista sejam desse tipo ou especifique o tipo de dado desejado na construção do NDArray.
- Utilize as rotinas do Numpy para facilitar a criação de NDArrays. Por exemplo, você pode criar uma lista de números pares até 200 ou NDArrays de zeros com apenas uma linha de código.



NDArray

- 
 Aproveite a função Random para criar matrizes com valores aleatórios entre 0 e 1. Isso pode ser útil para inicializar pesos em algoritmos de aprendizado de máquina, por exemplo.
- 
 Lembre-se de que o Numpy permite realizar operações matemáticas elemento a elemento com NDArrays. Isso inclui adição, multiplicação, divisão e exponenciação



NDArray

- Quando realizar operações matemáticas com escalares e vetores, saiba que o Numpy automaticamente "vetoriza" os escalares. Isso significa que você pode realizar operações entre um escalar e um vetor sem a necessidade de criar um vetor do mesmo tamanho que o escalar.



Gere dados

- Use a indexação booleana: A indexação booleana é uma maneira poderosa de selecionar elementos de um array com base em condições. Isso pode ser útil para filtrar dados ou realizar análises condicionais.
- Realize operações entre colunas: Ao trabalhar com matrizes, é possível realizar operações entre suas colunas para criar novas variáveis. Isso pode ser útil para a criação de recursos em modelagem de dados.



Construa funções

- Organize seu código com funções: Funções são uma maneira eficiente de organizar seu código, tornando-o mais legível e reutilizável. Sempre que você perceber que está repetindo o mesmo código várias vezes, considere criar uma função.
- Use funções nativas do Python: Python tem muitas funções nativas úteis, como `print`, `len`, `type` e `int`. Familiarize-se com essas funções e use-as para tornar seu código mais eficiente.



Construa funções

- Considere o uso de funções lambda para tarefas simples: Funções lambda são uma maneira mais simples e rápida de criar funções no Python. Elas são particularmente úteis para tarefas simples que podem ser realizadas em uma única linha de código.
- Aproveite os métodos nativos para diferentes tipos de dados: Python tem muitos métodos nativos para diferentes tipos de dados, como strings, listas e dicionários. Esses métodos podem tornar seu código mais eficiente e fácil de ler.



Bons estudos!

