**Para saber mais: vantagens dos arrays**

As listas em Python são estruturas de dados básicas que podem conter elementos de diferentes tipos (inteiros, strings, outras listas, etc.). Já a Numpy (Numerical Python) é uma biblioteca do Python que fornece suporte a arrays multidimensionais, estruturas de dados mais avançadas e eficientes para cálculos numéricos.

Aqui está um exemplo de como transformar uma lista em um array Numpy:

**import** numpy **as** np

# cria uma lista

lista = [1, 2, 3, 4, 5]

# transforma a lista em um array Numpy

array = np.array(lista)

print("Lista: ", lista)

print("Array: ", array)

Saída:

Lista: [1, 2, 3, 4, 5]

Array: [1 2 3 4 5]

Existem várias vantagens em usar arrays Numpy ao invés de listas regulares do Python e aqui estão algumas:

* **Eficiência de processamento**

As operações matemáticas em arrays Numpy são muito mais rápidas do que em listas regulares, pois a NumPy foi otimizada para trabalhar com conjuntos de dados homogêneos e libera memória do computador rapidamente.

* **Facilidade de uso**

As operações matemáticas em arrays Numpy são expressas de forma muito mais clara e concisa do que em listas regulares, tornando o código mais fácil de ler e manter.

* **Integração com outras bibliotecas**

A Numpy é uma das bibliotecas mais utilizadas em ciência de dados e aprendizado de máquina. Muitas outras bibliotecas, como a Pandas e a Matplotlib, são projetadas para trabalhar diretamente com arrays Numpy.

**Comparativo de performance: listas vs arrays**

Focando na eficiência, podemos comparar o tempo levado para efetuar um cálculo utilizando listas e arrays.

**import** numpy **as** np

**import** time

# cria uma lista com 1000000 elementos

lista = list(range(1000000))

# transforma a lista em um array Numpy

array = np.array(lista)

# começa a cronometrar o tempo para a operação com a lista

start\_time = time.time()

# realiza a operação de elevar ao quadrado cada elemento da lista

lista\_quadrado = [i\*\*2 **for** i **in** lista]

# para o cronômetro

list\_time = time.time() - start\_time

# começa a cronometrar o tempo para a operação com o array

start\_time = time.time()

# realiza a operação de elevar ao quadrado cada elemento do array

array\_quadrado = array\*\*2

# para o cronômetro

array\_time = time.time() - start\_time

print("Tempo da operação com a lista: ", list\_time)

print("Tempo da operação com o array: ", array\_time)

Saída:

Tempo da operação com a lista: 0.2745847702026367

Tempo da operação com o array: 0.004081010818481445

Como pode ser visto, a operação realizada com o array Numpy foi muito mais rápida do que com a lista regular, o que demonstra a eficiência no processamento com o array.