**06 - Seleções com arrays**

Anteriormente aprendemos a fazer seleções em listas do Python, inclusive quando elas possuem sublistas. Se você absorveu esse conteúdo, as seleções com arrays Numpy não serão um problema, já que são basicamente a mesma coisa. Como referência, continuaremos utilizando o desenho do array dados que criamos no vídeo anterior.



Esse array pode ser encarado com uma matriz/tabela em que temos duas linhas e cinco colunas, lembrando que a indexação se inicia no 0.



Começaremos nossos estudos criando uma variável contador que receberá a chamada de np.arange(10), gerando um array Numpy de tamanho 10.

contador = np.arange(10)

contadorCOPIAR CÓDIGO

array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

Sabemos que contador[0] nos retornará o primeiro elemento desse array, que é o 0. Agora imagine que desejamos criar um programa que nos retorna o sexto item de um array Numpy. Nesse caso, criaríamos uma variável item recebendo o valor 6, e outra, chamada index, que receberia item - 1 - ou seja, para acessarmos o índice correto basta subtrairmos em 1 o item desejado. Se fizermos contador[index], conseguiremos retornar corretamente o valor desse sexto item.

item = 6

index = item - 1

contador[index]COPIAR CÓDIGO

5

Dessa forma, se desejarmos acessar outros itens desse array, bastará alterarmos o valor de item. Se quisermos acessar o último elemento do array, também podemos utilizar o índice negativo -1

contador[-1]COPIAR CÓDIGO

9

Em relação ao nosso array dados, como vimos no vídeo anterior, a primeira linha pode ser acessada com dados[0], e a segunda com dados[1]. Mas e se quisermos selecionar parte desses arrays menores, como o valor 1990? Para isso, precisaremos passar os valores da linha e da coluna. Nesse caso, queremos a linha com índice 1 e a coluna com índice 2.

Existem duas formas de fazer isso. A primeira dela é usando dois pares de colchetes,seguindo a sintaxe nomedoarray[linha][coluna]. Outra opção é usando somente um par de colchetes e separando os dois índices com uma vírgula, ou seja, nomedoarray[linha,coluna]. Em ambos os casos o nosso retorno será o mesmo, como verificamos nos exemplos abaixo:

dados[1][2]COPIAR CÓDIGO

1990

dados[1,2]COPIAR CÓDIGO

1990

Agora falaremos sobre os **fatiamentos**. No notebook do curso deixamos uma breve explicação sobre a sintaxe dos fatiamentos, incluindo um conteúdo novo, que é o "passo".

A sintaxe para realizar fatiamento em um array Numpy é i:j:k onde i é o índice inicial, j é o índice de parada, e k é o indicador de passo ( k≠0 )

Lembrando que o i é incluído e o j não é incluído. Ou seja, se quisermos fatiar o segundo, terceiro e quarto item do nosso array, precisaremos iniciar no índice 1 e varrer o array até o índice 4, que não será incluído.

No contador que acabamos de criar, queremos selecionar os valores 1, 2 e 3. Como esse array se inicia com 0, o primeiro valor estará exatamente no índice 1. Para recuperarmos até o valor 3, usaremos o índice 4.

contador[1:4]COPIAR CÓDIGO

array([1, 2, 3])

Mas e o "passo", como funciona? Para testarmos, usaremos contador[1:8:2] - ou seja, vamos varrer o array contador do índice 1 até o 8, que não será incluído, contando dois passos a cada elemento encontrado.

contador[1:8:2]COPIAR CÓDIGO

array([1, 3, 5, 7])

O tamanho do passo determinado será o tamanho dos "pulos" dados na varredura do array. No próximo exemplo, queremos pegar todos os números pares do conjunto contador. Para isso, não passaremos nenhum índice inicial ou final, somente o passo 2 - ou seja, contador[::2].

contador[::2]COPIAR CÓDIGO

Como retorno, teremos somente os números pares:

array([0, 2, 4, 6, 8])

Se quisermos repetir o processo para os ímpares, basta começarmos no 1:

contador[1::2]COPIAR CÓDIGO

array([1, 3, 5, 7, 9])

Agora voltaremos ao nosso conjunto dados:

array([[44410., 5712., 37123., 0., 25757.], [ 2003., 1991., 1990., 2019., 2006.]])

Queremos, por exemplo, calcular a quilometragem média somente para o segundo, o terceiro e o quarto veículo desse conjunto. Para isso, primeiro recuperaremos todas as linhas do conjunto, ou seja, dados[:,]. Em seguida, queremos apenas os índices 1, 2 e 3. Conseguiremos isso com a instrução 1:4 - ou seja, os índices de 1 a 3, já que o quatro não é incluído.

dados[:, 1:4]COPIAR CÓDIGO

array([[ 5712., 37123., 0.], [ 1991., 1990., 2019.]])

Repare que é criado um array somente com as informações dos veículos que separamos. Se quisermos somente os veículos 1 e 2, podemos usar dados[:, 1:3].

array([[ 5712., 37123.], [ 1991., 1990.]])

É possível calcularmos a quilometragem média desses dois veículos, bastando seguirmos a fórmula abaixo:

dados[:, 1:3][0] / (2019 - dados[:, 1:3][1])COPIAR CÓDIGO

Parece confuso? Pois vamos explicar! Aqui estamos fatiamento o conjunto dados de modo a selecionarmos somente os veículos com os índices 1 e 2 (dados[:, 1:3]). Em seguida, pegamos somente a primeira linha - ou seja, o índice 0 desse conjunto. De posse desses valores, passamos a dividi-los por 2019 subtraído de dados[:, 1:3][1], a segunda linha do conjunto também fatiada somente com os veículos de índices 1 e 2. Com isso, estamos dividindo a quilometragem total dos veículos pelo tempo desde a sua fabricação, o que nos resulta na quilometragem média:

array([ 204. , 1280.10344828])

Outra forma de fazermos seleções é por meio dos arrays booleanos. A ideia é informarmos quais posições queremos que sejam mostradas com True, e quais não queremos com false. Nesse exemplo voltaremos a usar nosso contador:

contador = np.arange(10)

contadorCOPIAR CÓDIGO

Imagine que queremos saber quais são os elementos maiores do que 5 nesse array. Para isso, usaremos contador > 5. O retorno é um array Numpy de booleanos:

array([False, False, False, False, False, False, True, True, True, True])

Repare que todos os números menores ou iguais a 5 recebem False, mas 6 em diante recebem True. Isso significa que a expressão contador > 5 nos permitirá selecionar somente os elementos maiores do que 5 no array, da seguinte forma:

contador[contador >5]COPIAR CÓDIGO

array([6, 7, 8, 9])

Nesse retorno, só recebemos os valores que obtiverem True no "validador" que criamos. Tanto isso é verdade que podemos, inclusive, passar diretamente os valores False e True conforme desejarmos para o array. Nesse caso, manteremos como True todos os valores maiores do que 5.

contador[[False, False, False, False, False, False, True, True, True,

True]]COPIAR CÓDIGO

array([6, 7, 8, 9])

Vamos supor que queremos calcular a quilometragem média dos veículos fabricados apos o ano 2000. Para selecionarmos esses carros, podemos usar a expressão dados[1] > 2000, já que os anos se encontraram no índice 1 desse array.

dados[1] > 2000COPIAR CÓDIGO

array([ True, False, False, True, True])

Para selecionarmos somente esses dados, basta passarmos todo o conjunto como instrução das "linhas" e essa verificação como instrução das "colunas".

dados[:, dados[1] > 2000]COPIAR CÓDIGO

O retorno serão somente as colunas cujos veículos foram fabricados após o ano 2000.

ray([[44410., 0., 25757.], [ 2003., 2019., 2006.]])

Agora que fizemos a seleção, você pode realizar o cálculo como exercício desse tópico! No próximo vídeo aprenderemos um pouco sobre atributos e métodos de arrays Numpy.