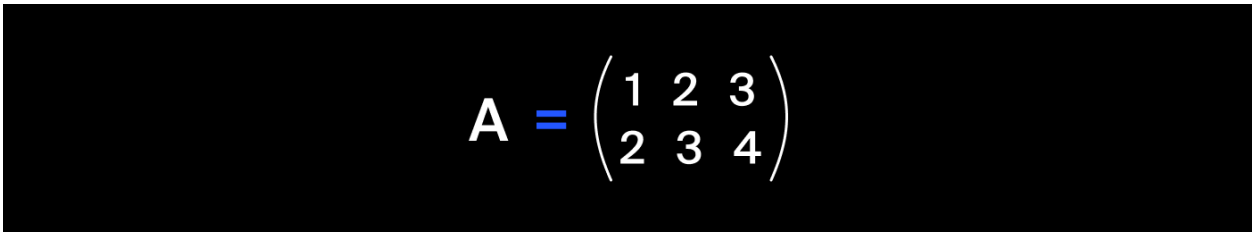
Resumen del capítulo: Matrices y operaciones matriciales

Creación de matrices

Una **matriz** (del latín "madre") es una tabla numérica rectangular o una matriz bidimensional. Consta de *m* filas y *n* columnas (el tamaño se escribe como $m \times n$). Las matrices generalmente se denotan con letras latinas mayúsculas, por ejemplo, *A*. Sus elementos están en minúsculas con doble índice a_{ij}, donde *i* es el número de fila y *j* es el número de columna.



Llama a `np.array()` para crear una matriz NumPy a partir de una lista de listas. Todas las listas anidadas tienen la misma longitud.

```
import numpy as np

matrix = np.array([
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]])
print(matrix)
```

Tomemos una lista de vectores en lugar de una lista de listas:

```
import numpy as np

string0 = np.array([1,2,3])
string1 = np.array([-1,-2,-3])
list_of_vectors = [string0, string1]
matrix_from_vectors = np.array(list_of_vectors)

print(matrix_from_vectors)
```

Crea una matriz a partir de la tabla pandas: su atributo `values` es una matriz.

```
import pandas as pd
import numpy as np

matrix = df.values
print(matrix)
```

El atributo `shape` define el tamaño de la matriz . Su a_{ij} elemento se establece en NumPy como A[I,j]: las filas y columnas se enumeran desde cero, al igual que los índices de matriz.

```
import numpy as np

A = np.array([
    [1, 2, 3],
    [2, 3, 4]])

print('Tamaño:', A.shape)
print('A[1, 2]:', A[1, 2])
```

Selecciona filas y columnas individuales de la matriz:

```
import numpy as np

matrix = np.array([
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9],
    [10,11,12]])

print('Row 0:', matrix[0, :])
print('Column 2:', matrix[:, 2])
```

Operaciones con elementos de matriz

Puedes realizar las mismas operaciones con elementos de matriz que con elementos vectoriales. Dos matrices se pueden sumar, restar, multiplicar o dividir. La parte más

importante es que las operaciones se realizan elemento por elemento y las matrices son del mismo tamaño. El resultado de una operación es una matriz del mismo tamaño.

```
import numpy as np

matrix1 = np.array([
    [1, 2],
    [3, 4]])

matrix2 = np.array([
    [5, 6],
    [7, 8]])

print(matrix1 + matrix2)
```

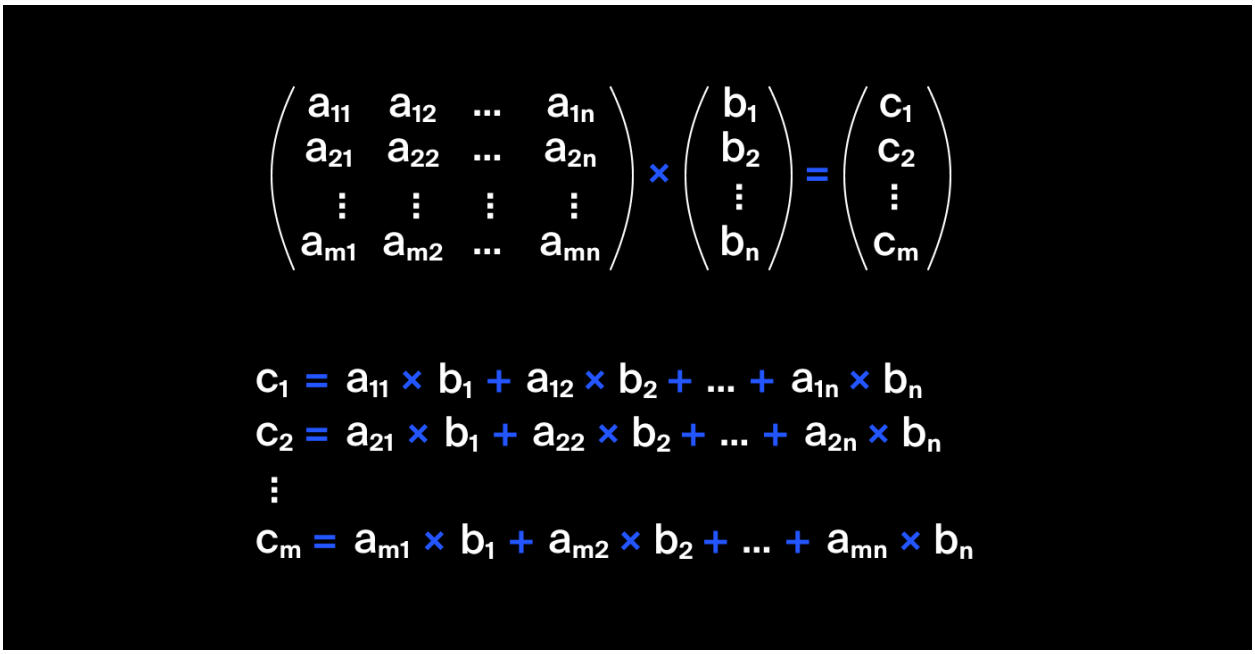
Puedes multiplicar una matriz por un número, sumar un número o restarlo: la operación se aplica a cada elemento.

```
import numpy as np

matrix = np.array([
    [1, 2],
    [3, 4]])

print(matrix * 2)
print(matrix - 2)
```

Multiplicar una matriz por un vector

Para entender cómo se multiplica una matriz por un vector, vamos a tomar una lista de filas. Cada fila de esta lista (matriz) es un vector multiplicado por un escalar y los números resultantes forman un nuevo vector.

Por ejemplo, una matriz A de tamaño $m \times n$ se multiplica por un vector $\vec{b}$ (n-dimensional). El producto será un nuevo vector $\vec{c} = A\vec{b}$. Este es un vector m dimensional cuya i coordenada es igual al producto escalar de la fila i de la matriz y el vector $\vec{b}$.



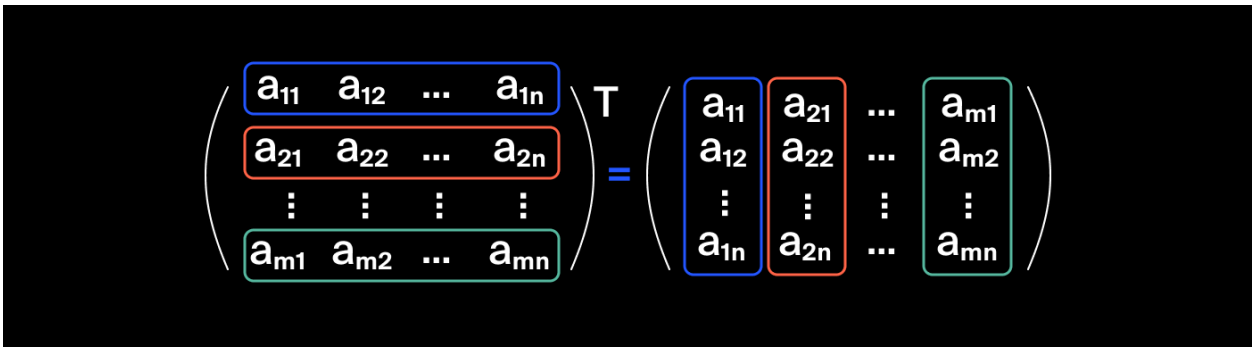
Vamos a realizar esta operación en NumPy y llamar a la función `np.dot()`, que ya conocemos.

```
import numpy as np

A = np.array([
    [1, 2, 3],
    [4, 5, 6]])

b = np.array([7, 8, 9])

print(np.dot(A, b))
print(A.dot(b))
```

Para que la multiplicación sea correcta, el tamaño del vector debe ser igual al ancho de la matriz.

Transposición de matriz

La transposición de una matriz es su "giro" sobre la diagonal principal de la matriz, que va desde la esquina superior izquierda hasta la esquina inferior derecha. Con esta inversión, la matriz A de tamaño $m \times n$ se transforma en una matriz de tamaño $n \times m$. En otras palabras, las filas de la matriz se convierten en sus columnas y las columnas se convierten en sus filas. La matriz transpuesta se indica con el índice superior **T**:



En NumPy, esta operación la realiza el atributo `T`. Si necesitas construir una matriz, comienza desde la lista de columnas para crear una matriz y aplica la transposición:

```
import numpy as np

matrix = np.array([
    [1, 2],
    [4, -4],
    [0, 17]])

print("Matriz transpuesta")
print(matrix.T)
```

Multiplica la matriz original por un vector de longitud igual a 3:

```
vector = [2, 1, -1]
print(np.dot(matrix, vector))
```

Tenemos un error: las dimensiones de la matriz (3, 2) y el vector (3,) no están alineados. La segunda dimensión de la matriz no es igual a la longitud del vector. Para hacer que la multiplicación sea correcta, podemos transponer la matriz:

```
print(np.dot(matrix.T, vector))
```

Multiplicación de matrices

Durante la **multiplicación de matrices**, se construye una tercera matriz usando dos matrices. Consiste en productos escalares de las filas de la primera matriz por las columnas de la segunda. El producto de la fila i de la matriz A (A_i) y la columna j de la matriz B (B_j) es igual a la matriz C_{ij}:

$$c_{ij} = (A_i, B_j)$$

La multiplicación de matrices es posible si el ancho de la primera matriz A ($m \times n$) es igual a la altura de la segunda matriz B (n×r). Entonces las dimensiones de su producto serán m×r. Así, decimos que la dimensión n "colapsa".

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \times \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1r} \\ b_{21} & b_{22} & \dots & b_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nr} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} & \dots & c_{1r} \\ c_{21} & c_{22} & \dots & c_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ c_{m1} & c_{m2} & \dots & c_{mr} \end{pmatrix}$$

$$\begin{aligned} c_{11} &= a_{11} \times b_{11} + a_{12} \times b_{21} + \dots + a_{1n} \times b_{n1} \\ c_{12} &= a_{11} \times b_{12} + a_{12} \times b_{22} + \dots + a_{1n} \times b_{n2} \\ &\vdots \\ c_{1r} &= a_{11} \times b_{1r} + a_{12} \times b_{2r} + \dots + a_{1n} \times b_{nr} \end{aligned}$$

En NumPy, las matrices A y B se multiplican llamando a las funciones `np.dot(A, B)` o `A.dot(B)`. También puedes reemplazar esta llamada con un signo de multiplicación de matriz @:

```
import numpy as np

print(A.dot(B))
print(np.dot(A,B))
print(A @ B)
```

El resultado de la multiplicación de matrices depende del orden de los multiplicadores.

```
matrix = np.array([
    [1, 2, 3],
    [-1, -2, -3]])

print(matrix @ matrix)
```

Ocurrió un error: las dimensiones de las matrices no coinciden.

Multiplicar la matriz por sí misma solo es posible si es cuadrada:

```
square_matrix = np.array([
    [1, 2, 3],
    [-1, -2, -3],
    [0, 0, 0]])

print(square_matrix @ square_matrix)
```