Twitter Thread by Santiago



Santiago
@svpino

An introduction to one of the the most basic structures used in machine learning: a tensor.

■■

Tensors are the data structure used by machine learning systems, and getting to know them is an essential skill you should build early on.

A tensor is a container for numerical data. It is the way we store the information that we'll use within our system.

(2 / 16)

Three primary attributes define a tensor:

■■ Its rank
■■ Its shape
■■ Its data type

(3 / 16)

The rank of a tensor refers to the tensor's number of axes.

Examples:

■■ The rank of a matrix is 2 because it has two axes.
■■ The rank of a vector is 1 because it has a single axis.

(4 / 16)

The shape of a tensor describes the number of dimensions along each axis.

Example:

■■ A square matrix may have (3, 3) dimensions.
■■ A tensor of rank 3 may have (2, 5, 7) dimensions.

(5 / 16)

The data type of a tensor refers to the type of data contained in it.

For example, when thinking about Python ■'s numpy library, here are some of the supported data types:

■■ float32
■■ float64
■■ uint8
■■ int32
■■ int64

(6 / 16)

In the previous tweets I used the terms "vector" and "matrix," to referr to tensors with a specific rank (1 and 2 respectively.)

We can also use these mathematical concepts when describing tensors.

(7 / 16)

A scalar —or a 0D tensor— has rank 0 and contains a single number. These are also called "0-dimensional tensors."

The attached image shows how to construct a 0D tensor using numpy. Notice its shape and its rank (.ndim attribute.)

(8 / 16)



A vector —or a 1D tensor— has rank 1 and represents an array of numbers.

The attached image shows a vector with shape (4,). Notice how its rank (.ndim attribute) is 1.

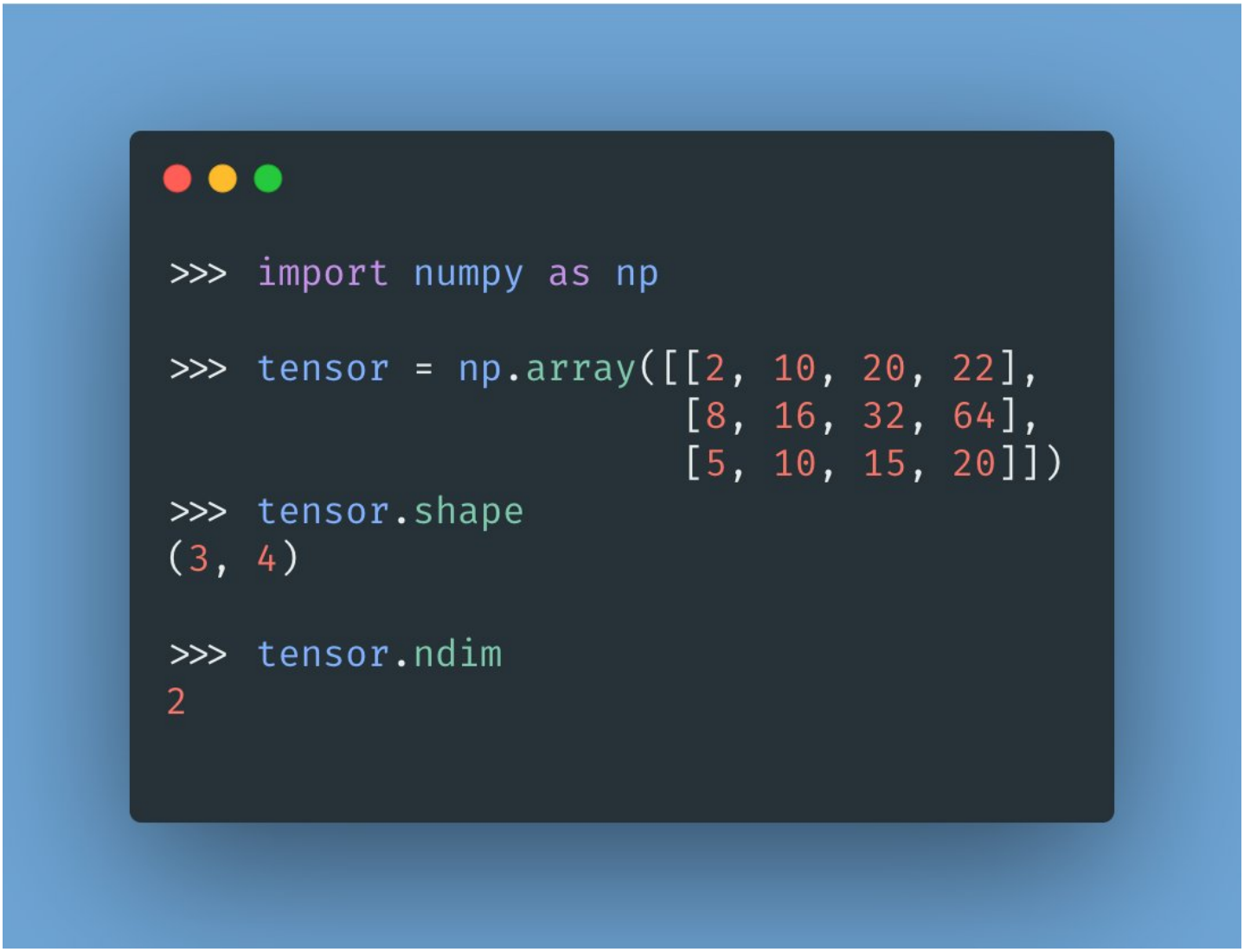
(9 / 16)



A matrix —or a 2D tensor— has rank 2 and represents an array of vectors. The two axes of a matrix are usually referred to as "rows" and "columns."

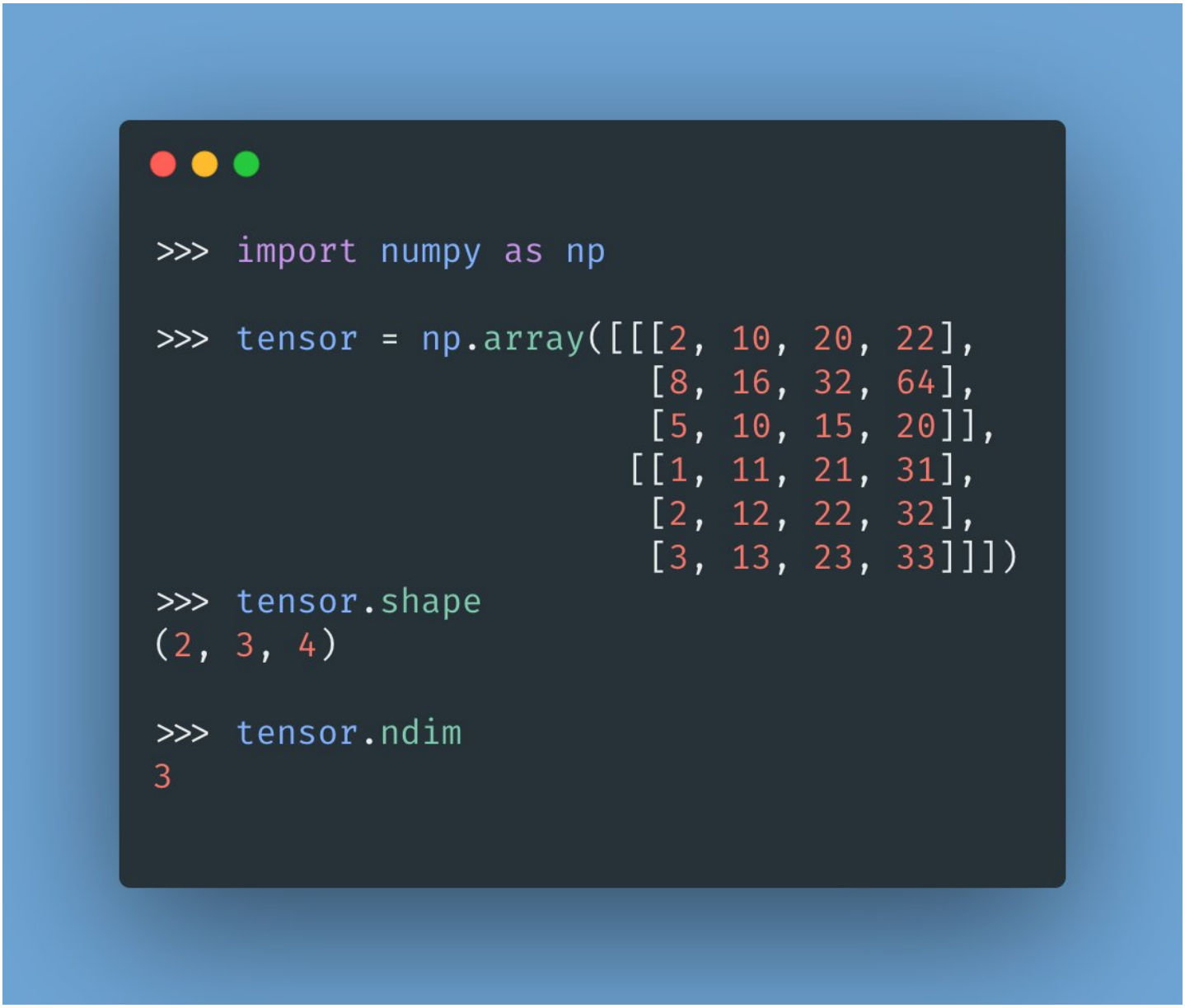
The attached image shows a matrix with shape (3, 4).

(10 / 16)



You can obtain higher-dimensional tensors (3D, 4D, etc.) by packing lower-dimensional tensors in an array.

For example, packing a 2D tensor in an array gives us a 3D tensor. Packing this one in another array gives us a 4D tensor, and so on.



Here are some common tensor representations:

■■ Vectors: 1D - (features)
■■ Sequences: 2D - (timesteps, features)
■■ Images: 3D - (height, width, channels)
■■ Videos: 4D - (frames, height, width, channels)

(12 / 16)

Commonly, machine learning algorithms deal with a subset of data at a time (called "batches.")

When using a batch of data, the tensor's first axis is reserved for the size of the batch (number of samples.)

(13 / 16)

For example, if your handling 2D tensors (matrices), a batch of them will have a total of 3 dimensions:

■■ (samples, rows, columns)

Notice how the first axis is the number of matrices that we have in our batch.

(14 / 16)

Following the same logic, a batch of images can be represented as a 4D tensor:

■■ (samples, height, width, channels)

And a batch of videos as a 5D tensor:

■■ (samples, frames, height, width, channels)

(15 / 16)

If all of this makes sense, you are on your way! If something doesn't click, reply with your question, and I'll try to answer.

Either way, make sure to follow me for more machine learning content! 2021 is going to be great!

(16 / 16)