Experiment no. 4

Title :- Linear Algebra Tensors Tensor arithmetic Implementing matrix multiplication.

Aim:- To Linear Algebra Tensors Tensor arithmetic Implementing matrix multiplication.

Linear algebra plays a fundamental role in machine learning, and tensors are essential data structures used for representing multi-dimensional arrays. In this section, I'll cover basic tensor operations, tensor arithmetic, and how to implement matrix multiplication using NumPy.

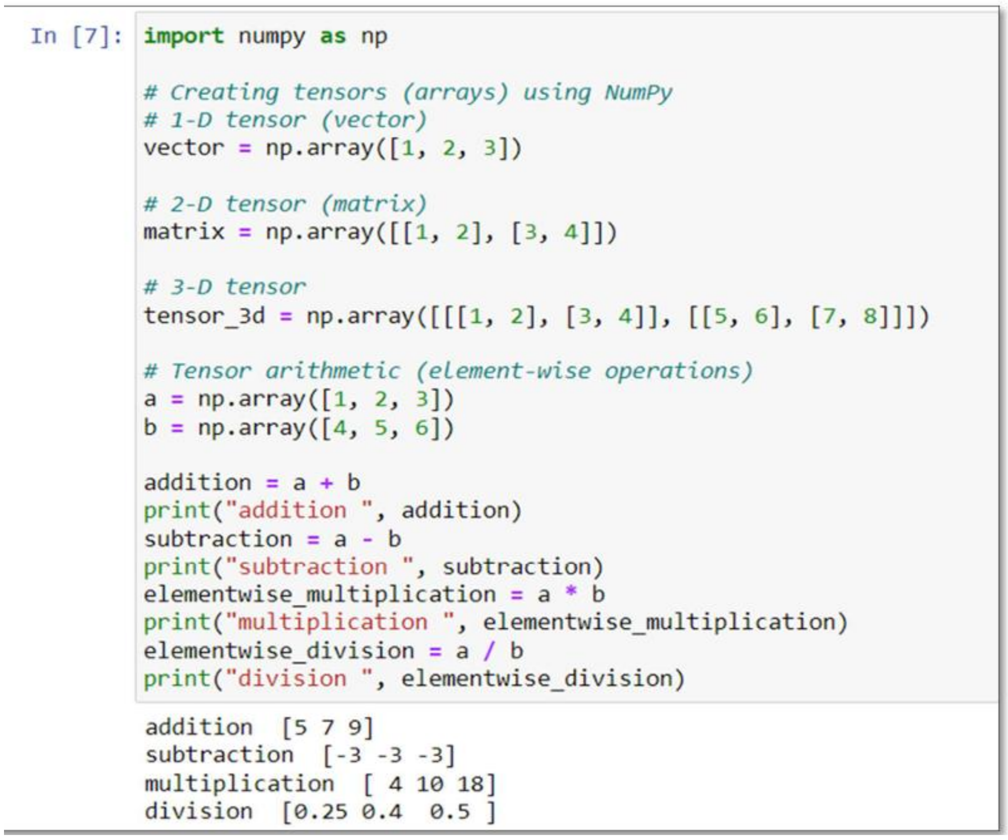
A tensor is a generalization of vectors and matrices and is easily understood as a multidimensional array.

A vector is a one-dimensional or first order tensor and a matrix is a two-dimensional or second order tensor.

Step 1) Tensors and Tensor Arithmetic:

Tensors are multi-dimensional arrays. In the context of linear algebra, vectors are 1-dimensional tensors,

matrices are 2-dimensional tensors, and so on.

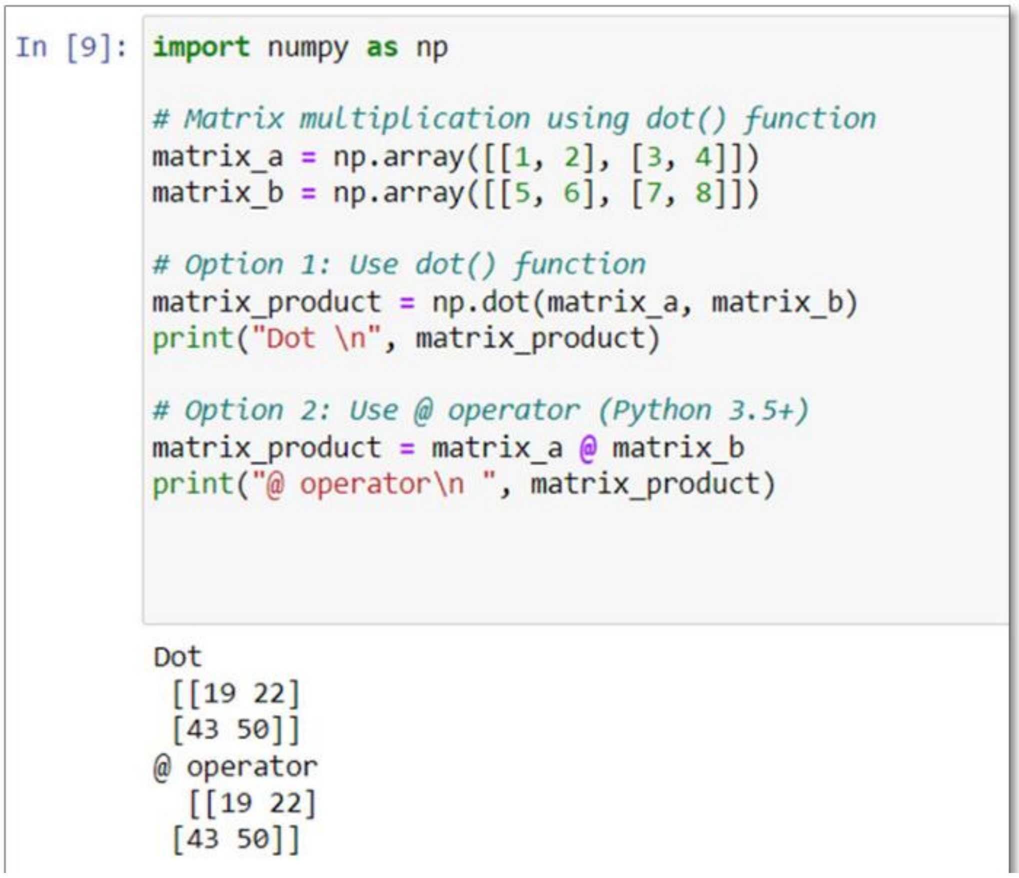


```
In [7]: import numpy as np

        # Creating tensors (arrays) using NumPy
        # 1-D tensor (vector)
        vector = np.array([1, 2, 3])

        # 2-D tensor (matrix)
        matrix = np.array([[1, 2], [3, 4]])

        # 3-D tensor
        tensor_3d = np.array([[[1, 2], [3, 4]], [[5, 6], [7, 8]]])

        # Tensor arithmetic (element-wise operations)
        a = np.array([1, 2, 3])
        b = np.array([4, 5, 6])

        addition = a + b
        print("addition ", addition)
        subtraction = a - b
        print("subtraction ", subtraction)
        elementwise_multiplication = a * b
        print("multiplication ", elementwise_multiplication)
        elementwise_division = a / b
        print("division ", elementwise_division)
```

```
addition  [5 7 9]
subtraction  [-3 -3 -3]
multiplication  [ 4 10 18]
division  [0.25 0.4  0.5 ]
```

Step 2) Implementing Matrix Multiplication:

Matrix multiplication is a fundamental operation in linear algebra. In NumPy, you can use the `dot()` function or `@` operator to perform matrix multiplication.



In [9]:
```
import numpy as np

# Matrix multiplication using dot() function
matrix_a = np.array([[1, 2], [3, 4]])
matrix_b = np.array([[5, 6], [7, 8]])

# Option 1: Use dot() function
matrix_product = np.dot(matrix_a, matrix_b)
print("Dot \n", matrix_product)

# Option 2: Use @ operator (Python 3.5+)
matrix_product = matrix_a @ matrix_b
print("@ operator\n ", matrix_product)
```

```
Dot
 [[19 22]
 [43 50]]
@ operator
   [[19 22]
 [43 50]]
```

Conclusion

Implemented Matrix Multiplication and Linear Algebra Tensors using Tensorflow.