Linear Algebra
Scalars

- x = np.array(3.0)
- = np.array(2.0)
- x + y, x * y, x / y, x ** y
- o/p
- (array(5.), array(6.), array(1.5), array(9.))

Linear Algebra …
Matrices

- We have already discussed about length dimensionality and shape.
- A = np.arange(20).
- reshape(5, 4)
- A
- o/p

```
array([[ 0.,  1.,  2.,  3.],
       [ 4.,  5.,  6.,  7.],
       [ 8.,  9., 10., 11.],
       [12., 13., 14., 15.],
       [16., 17., 18., 19.]])
```

Linear Algebra …
Matrices…Transpose of a Matrix

- A.T

```
array([[ 0., 4., 8., 12., 16.],
 [ 1., 5., 9., 13., 17.],
 [ 2., 6., 10., 14., 18.],
 [ 3., 7., 11., 15., 19.]])
```

Linear Algebra …
Transpose of a Matrix

- As a special type of the square matrix,a symmetric matrix A is equal to its transpose: A = AT.
- B = np.array([[1, 2, 3], [2, 0, 4], [3, 4, 5]])
- B

```
array([[1., 2., 3.],
       [2., 0., 4.],
       [3., 4., 5.]])
```

Linear Algebra …
Matrices…Transpose of a Matrix

- Now we compare B with its transpose.
- o/p

```
array([[ True,  True,  True],
       [ True,  True,  True],
       [ True,  True,  True]])
```

Linear Algebra
Basic Properties of Tensor Arithmetic

- Scalars, vectors, matrices, and tensors (“tensors” in this subsection refer to algebraic objects)of an arbitrary number of axes have some nice properties that often come in handy.
- For example
- Two tensors with the same shape, the result of any binary element wise operation will be a tensor of that same shape.

Linear Algebra
Basic Properties of Tensor Arithmetic….

- A = np.arange(20).reshape(5, 4)
- B = A.copy() # Assign a copy of `A` to `B` by allocating new memory
  A, A + B

```
(array([[ 0.,  1.,  2.,  3.],
        [ 4.,  5.,  6.,  7.],
        [ 8.,  9., 10., 11.],
        [12., 13., 14., 15.],
        [16., 17., 18., 19.]]),
 array([[ 0.,  2.,  4.,  6.],
        [ 8., 10., 12., 14.],
        [16., 18., 20., 22.],
        [24., 26., 28., 30.],
        [32., 34., 36., 38.]]))
```

Linear Algebra
Sum and Dot Product

- x = np.arange(4) x,
- x.sum()
- o/p
- (array([0., 1., 2., 3.]), array(6.))

- y = np.ones(4)
- x, y, np.dot(x, y)

o/p

- (array([0., 1., 2., 3.]), array([1., 1., 1., 1.]), array(6.))

Linear Algebra
Matrix-Vector Products

- A=[1,2,3,4]
- np.dot(A,2)
- **o/p**
- `array([2, 4, 6])`

Matrix-Matrix Multiplication

- A=[1 2 3]
- B=[2,3,4]
- Np.dot(A,B)
- o/p
- 20

$$\mathbf{C} = \mathbf{AB} = \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \vdots \\ \mathbf{a}_n^\top \end{bmatrix} \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_m \end{bmatrix}$$

Matrix-Matrix Multiplication…

- B = np.ones(shape=(4, 3))
- np.dot(A, B)