set 1:

1)Define a vector:

import numpy as np

list = [10,2,3]

vector = np.array(list)

print("the vector is",vector)

2)add 2 vectors using numpy:

import numpy as np

list1 = [10, 20, 11, 45, 69]

list2 = [11, 12, 0, 6, 50]

vtr\_1 = np.array(list1)

vtr\_2 = np.array(list2)

print("we create vector from list1: ")

print("vtr1: ", vtr\_1)

print("we create vector from list2: ")

print("vtr2: ", vtr\_2)

vctr\_add = vtr\_1 + vtr\_2

print("addition of two vector: ", vctr\_add)

3)sup 2 vectors using numpy:

import numpy as np

list1 = [10, 20, 11, 45, 69]

list2 = [11, 12, 0, 6, 50]

vtr\_1 = np.array(list1)

vtr\_2 = np.array(list2)

print("we create vector from list1: ")

print("vtr1: ", vtr\_1)

print("we create vector from list2: ")

print("vtr2: ", vtr\_2)

vctr\_sub = vtr\_1 - vtr\_2

print("substraction of two vector: ", vctr\_sub)

4)multiply 2 vectors using numpy:

import numpy as np

list1 = [10, 2, 3, 5, 7]

list2 = [5, 17, 22, 60, 21]

vtr1 = np.array(list1)

vtr2 = np.array(list2)

print("we create vector from list1: ")

print(vtr1)

print("we create vector from list2: ")

print(vtr2)

vtr\_multi = vtr1 \* vtr2

print("substract of two vector is: ", vtr\_multi)

5)divide 2 vectors using numpy:

import numpy as np

list1 = [10, 15, 30, 50, 72]

list2 = [5, 5, 3, 5, 2]

vtr1 = np.array(list1)

vtr2 = np.array(list2)

print("we create vector from list1: ")

print(vtr1)

print("we create vector from list2: ")

print(vtr2)

vtr\_div = vtr1 / vtr2

print("division of two vector is: ", vtr\_div)

6)find dot product of two vectors:

import numpy as np

list1 = [10, 2, 3, 5, 7]

list2 = [5, 17, 22, 60, 21]

vtr1 = np.array(list1)

vtr2 = np.array(list2)

print("we create vector from list1: ")

print(vtr1)

print("we create vector from list2: ")

print(vtr2)

vtr\_product = vtr1 . dot(vtr2)

print("dot product of two vector is: ", vtr\_product)

7)perform vector scalar multiplication:

import numpy as np

list1 = [2, 4, 5, 6]

vtr1 = np.array(list1)

print("we create vector from list1: ")

scalar\_value = 10

print(vtr1)

print("scalar value: "+str(scalar\_value))

vtr\_scalar = vtr1 \* scalar\_value

print("multiplication of one scalar and vector is: ", vtr\_scalar

8)L1 norms:

from numpy import inf

from numpy import array

from numpy.linalg import norm

arr = array([23, 45, 67, 35])

print(arr)

maxnorm = norm(arr,inf)

print("maxnorm:", maxnorm)

norm\_l1 = norm(arr, 1)

print(norm\_l1)

print("array we have is:", arr)

l2\_norms = norm(arr)

print("l2\_norms", l2\_norms)

set 2:

1)define a matrix:

import numpy as np

r = int(input("enter number of rows: "))

c = int(input("enter number of columns: "))

print("enter the entries in a (single line seperated by space):")

entries = list(map(int, input().split()))

matrix = np.array(entries).reshape(r, c)

print(matrix)

2)add 2 matrix:

x = [[1, 2, 3],

[4, 5, 6],

[3, 4, 5]]

y = [[9, 8, 4],

[6, 5, 9],

[3, 4, 6]]

result =[[x[i][j]+y[i][j] for j in range(len(x[0]))]for i in range(len(x))]

for r in result:

print(r)

3)substract two matrix:

x = [[1, 2, 3],

[4, 5, 6],

[3, 4, 5]]

y = [[9, 8, 4],

[6, 5, 9],

[3, 4, 6]]

result =[[x[i][j]-y[i][j] for j in range(len(x[0]))]for i in range(len(x))]

for r in result:

print(r)

4)find hadamard product of two matrix:

import numpy as np

a = np.array([[1, 2, 3],

[3, 4, 5]])

b = np.array([[1, 2, 4],

[3, 4, 6]])

c = np.multiply(a, b)

print(c)

5)divide two matrix:

x = [[1, 2, 3],

[4, 5, 6],

[3, 4, 5]]

y = [[9, 8, 4],

[6, 5, 9],

[3, 4, 6]]

result =[[x[i][j]/y[i][j] for j in range(len(x[0]))]for i in range(len(x))]

for r in result:

print(r)

6)find the product of two matrix:

x = [[1, 2, 3],

[4, 5, 6],

[3, 4, 5]]

y = [[9, 8, 4],

[6, 5, 9],

[3, 4, 6]]

result =[[x[i][j]\*y[i][j] for j in range(len(x[0]))]for i in range(len(x))]

for r in result:

print(r)

7)perform vector matrix multiplication:

import numpy as np

A = np.array([[1, 6],

[2, 3]])

d = np.array([[3],

[2]])

result = np.matmul(A, d)

print(result)

result = np.matmul(A, d)

print(result)

8)perform scalar matrix multiplication:

def scalarProductMat(matrix, k):

for i in range(Rows):

for j in range(Columns):

matrix[i][j] = matrix[i][j] \* k

if \_\_name\_\_ == "\_\_main\_\_":

Rows = int(input("Give the number of rows:"))

Columns = int(input("Give the number of columns:"))

matrix = [[int(input()) for c in range(Columns)] for r in range(Rows)]

print(matrix)

k = int(input("Enter the scalar value:"))

scalarProductMat(matrix, k)

print("Scalar Product Matrix is : ")

for i in range(Rows):

for j in range(Columns):

print(matrix[i][j], end=" ")

print()

9)Define a 3\*3 square matrix extract the main diagonal as vector:

import numpy as np

row = int(input("enter number of rows:"))

col = int(input("enter number of cols:"))

print("enter the entries for matrix:")

num = list(map(int, input().split()))

matrix = np.array(num).reshape(row, col)

print(matrix)

geeks = matrix.diagonal()

print("diagonal vector:", geeks)

arr = []

def dia(matrix, row, col):

for i in range(0, row):

for j in range(0, col):

if(i==j):

m1=matrix[i][j]

arr.append(m1)

else:

m1=0

arr.append(m1)

return arr

res = dia(matrix, row,col)

print(np.array(res).reshape(row, col))

10)create an identity matrix of order 4:

def Identity(size):

for row in range(0, size):

for col in range(0, size):

if(row==col):

print("1 ",end=" ")

else:

print("0 ",end=" ")

print()

size = int(input("enter the order of the matrix\n"))

print("order of matrix is",size)

Identity(size)

11)find transpose of a matrix:

def transpose(A,B):

for i in range(n):

for j in range(m):

B[i][j] = A[j][i]

import numpy as np

n = int(input("enter number of rows: "))

m = int(input("enter number of columns: "))

print("enter the entries in a (single line seperated by space):")

entries = list(map(int, input().split()))

A = np.array(entries).reshape(n, m)

print("A:",A)

B = [[0 for x in range(m)]for y in range(n)]

transpose(A,B)

print("Result matrix is B: ")

for i in range(n):

for j in range(m):

print( B[i][j], "", end="")

print()

12)inverse of a matrix:

import numpy as np

n = int(input("enter number of rows:"))

m = int(input("enter number of columns:"))

print("enter the entries in a (single line seperate by space):")

entries = list(map(int, input().split()))

A = np.array(entries).reshape(n, m)

print("A:", A)

print(np.linalg.inv(A))

13)print determinant of matrix:

import numpy as np

n = int(input("Enter the order of matrix :"))

matrix=[]

print("Enter the elements of matrix :")

for i in range (n) :

for j in range (n) :

ele = int(input())

matrix.append(ele)

print("The matrix is")

matrix = np.array(matrix)

matrix= matrix.reshape(n,n)

determinant\_matrix = np.linalg.det(matrix)

print(matrix)

print("The determinant of matrix is")

print(determinant\_matrix)

set 3:

1)create an orthogonal matrix:

def is\_orthogonal(Q, m, n):

if m != n:

return False

else:

for i in range(0, n):

for j in range(0, n):

sum = 0

for k in range(0, n):

sum = sum + round(Q[i][k] \* Q[j][k])

if i != j and sum != 0:

return False

if i == j and sum != 1:

return False

else:

return True

Q = [[0.68567, 0.12975, -0.71626],

[0.14807, 0.93855, 0.31176],

[0.71269, -0.31982, 0.62433]]

if is\_orthogonal(Q, 3, 3):

print("The given matrix is orthogonal")

else:

print("The given matrix is not orthogonal")

2)print rank of matrix:

import numpy as np

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

[2, 4, 5, 6],

[3, 4, 5, 6],

[10, 33, 44, 55]])

print("rank of matrix:", np.linalg.matrix\_rank(matrix))

3)sparsity of matrix:

import numpy as np

from scipy.sparse import csr\_matrix

A=np.array([[1,0,0,0,0,0],[0,0,2,0,0,1],[0,0,0,2,0,0]])

print("Dense matrix representation:\n",A)

S=csr\_matrix(A)

print("sparse matrix:\n",S)

B=S.todense()

print("Dense matrix:\n",B)

4)print eigen values and eigen vectors of a matrix:

import numpy as np

from numpy.linalg import eig

def Eigen(matrix):

Evalue, Evector = eig(matrix)

value = np.array(Evalue)

vector = np.array(Evector)

print(f"\nEigen Value Of The Matrix :\n\n {value}\n\n\nEigen Vector Of The Matrix :\n\n{vector}\n\n")

matrix = np.array([

[1, 2, 3],

[0, 5, 6],

[7, 8, 9]

])

Eigen(matrix)

print("\nMatrix = :")

for row in matrix:

print(row)

5)calculate eigen values and eigen vectors of a matrix and reconstruct the matrix:

from numpy import diag

import numpy as np

from numpy import dot

from numpy.linalg import inv

from numpy.linalg import eig

matrix = np.array([

[1,2,3],

[4,5,6],

[7,8,9]

])

print("\nMatrix = :")

for row in matrix:

print(row)

Evalues, Evectors = eig(matrix)

invEig = inv(Evectors)

Diagfrmvect = diag(Evalues)

rematrix = Evectors.dot(Diagfrmvect).dot(invEig)

print("\nRe-Constructed Matrix = :")

for row in rematrix:

print(row)

6)define 5\*2 matrix dataset ,split it into x and y components and print data set as scatterplot

import matplotlib.pyplot as plt

import numpy as np

matrix = np.array([

[5,99],[6,85],[9,88],[4,100],[8,85]]

)

x,y=np.split(matrix,2,axis=1)

plt.scatter(x, y)

plt.show()