)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()
```

2)calculate the 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)
```

3)create an orthogonal matrix and check Q transpose*Q=Q*Q transpose=1

```
def isOrthogonal(a, m, n):
   if (m != n):
      return False


   trans = [[0 for x in range(n)]
           for y in range(n)]
   for i in range(0, n):
      for j in range(0, n):
         trans[i][j] = a[j][i]


   prod = [[0 for x in range(n)]
         for y in range(n)]
   for i in range(0, n):
      for j in range(0, n):


         sum = 0
         for k in range(0, n):
            sum = sum + (a[i][k] *
                  a[j][k])
```

```
            prod[i][j] = sum
    for i in range(0, n):
        for j in range(0, n):

            if (i != j and prod[i][j] != 0):
                return False
            if (i == j and prod[i][j] != 1):
                return False

    return True

Rows = int(input("Give the number of rows:"))
Columns = int(input("Give the number of columns:"))
a = [[int(input()) for c in range(Columns)] for r in range(Rows)]
print(a)

if (isOrthogonal(a, 3, 3)):
    print("the given matrix is orthogonal")
else:
    print("the given matrix is not orthogonal")
```

4)Define a 5*5 matrix data set ,splitit into x and y components and plot dataset and scatterplot

5)eigen values and eigen vectors:

```
import numpy as np
a = np.array([[3,1],[2,2]])
w,v = np.linalg.eig(a)
```

```
print("eigen values: ",w)
print("eigen vector:",v)
```

6)calculate eigenvalues and eigenvectors of a matrix and reconstruct the matrix: