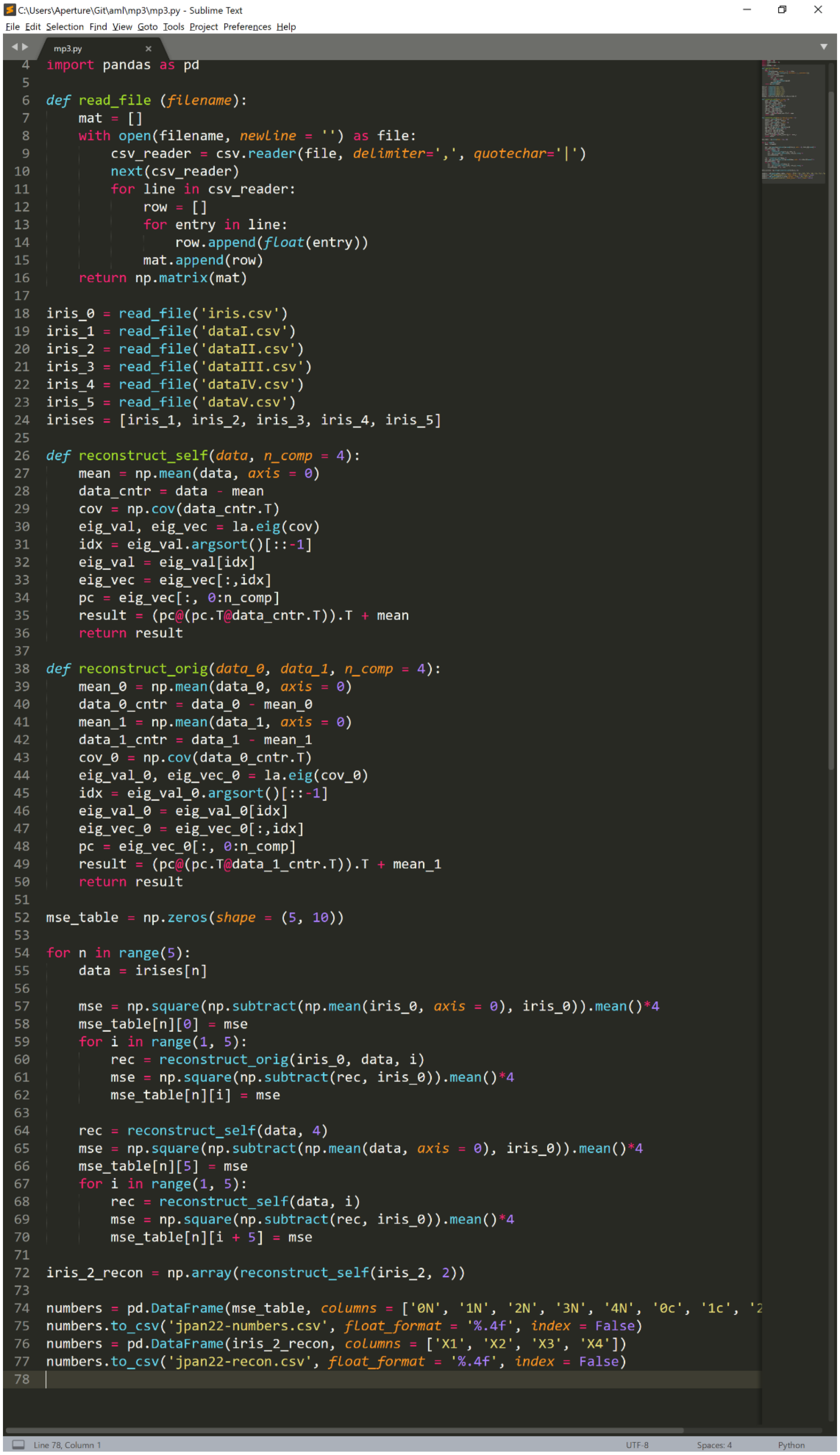
Number of PCs ->	**0N**	**1N**	**2N**	**3N**	**4N**	**0c**	**1c**	**2c**	**3c**	**4c**
Dataset I	4.5425	0.384	0.1759	0.1421	0.1608	4.5431	0.3846	0.1778	0.1444	0.1608
Dataset II	4.5425	0.6452	0.716	0.9085	1.1157	4.5495	0.6486	0.7506	0.942	1.1157
Dataset III	4.5425	1.3054	1.9798	2.651	3.6533	4.5575	1.3235	2.1197	3.0274	3.6533
Dataset IV	4.5425	0.8043	0.8299	0.9862	1.194	4.5662	0.8406	1.2071	1.2712	1.194
Dataset V	4.5425	2.1202	3.3572	4.5562	5.1393	4.9199	2.8357	4.6514	4.9712	5.1393



C:\Users\Aperture\Git\aml\mp3\mp3.py - Sublime Text
File Edit Selection Find View Goto Tools Project Preferences Help
mp3.py
Line 78, Column 1
UTF-8
Spaces: 4
Python

```python
import pandas as pd

def read_file (filename):
    mat = []
    with open(filename, newline = '') as file:
        csv_reader = csv.reader(file, delimiter=',', quotechar='|')
        next(csv_reader)
        for line in csv_reader:
            row = []
            for entry in line:
                row.append(float(entry))
            mat.append(row)
    return np.matrix(mat)

iris_0 = read_file('iris.csv')
iris_1 = read_file('dataI.csv')
iris_2 = read_file('dataII.csv')
iris_3 = read_file('dataIII.csv')
iris_4 = read_file('dataIV.csv')
iris_5 = read_file('dataV.csv')
irises = [iris_1, iris_2, iris_3, iris_4, iris_5]

def reconstruct_self(data, n_comp = 4):
    mean = np.mean(data, axis = 0)
    data_cntr = data - mean
    cov = np.cov(data_cntr.T)
    eig_val, eig_vec = la.eig(cov)
    idx = eig_val.argsort()[::-1]
    eig_val = eig_val[idx]
    eig_vec = eig_vec[:,idx]
    pc = eig_vec[:, 0:n_comp]
    result = (pc@(pc.T@data_cntr.T)).T + mean
    return result

def reconstruct_orig(data_0, data_1, n_comp = 4):
    mean_0 = np.mean(data_0, axis = 0)
    data_0_cntr = data_0 - mean_0
    mean_1 = np.mean(data_1, axis = 0)
    data_1_cntr = data_1 - mean_1
    cov_0 = np.cov(data_0_cntr.T)
    eig_val_0, eig_vec_0 = la.eig(cov_0)
    idx = eig_val_0.argsort()[::-1]
    eig_val_0 = eig_val_0[idx]
    eig_vec_0 = eig_vec_0[:,idx]
    pc = eig_vec_0[:, 0:n_comp]
    result = (pc@(pc.T@data_1_cntr.T)).T + mean_1
    return result

mse_table = np.zeros(shape = (5, 10))

for n in range(5):
    data = irises[n]

    mse = np.square(np.subtract(np.mean(iris_0, axis = 0), iris_0)).mean()*4
    mse_table[n][0] = mse
    for i in range(1, 5):
        rec = reconstruct_orig(iris_0, data, i)
        mse = np.square(np.subtract(rec, iris_0)).mean()*4
        mse_table[n][i] = mse

    rec = reconstruct_self(data, 4)
    mse = np.square(np.subtract(np.mean(data, axis = 0), iris_0)).mean()*4
    mse_table[n][5] = mse
    for i in range(1, 5):
        rec = reconstruct_self(data, i)
        mse = np.square(np.subtract(rec, iris_0)).mean()*4
        mse_table[n][i + 5] = mse

iris_2_recon = np.array(reconstruct_self(iris_2, 2))

numbers = pd.DataFrame(mse_table, columns = ['0N', '1N', '2N', '3N', '4N', '0c', '1c', '2
numbers.to_csv('jpan22-numbers.csv', float_format = '%.4f', index = False)
numbers = pd.DataFrame(iris_2_recon, columns = ['X1', 'X2', 'X3', 'X4'])
numbers.to_csv('jpan22-recon.csv', float_format = '%.4f', index = False)

```