In [121…

```python
import numpy as np
from scipy.io import loadmat

def kMeans(X, K, maxIters = 20):

    centroids = X[np.random.choice(len(X), K)]
    for i in range(maxIters):
        # Cluster Assignment step
        C = np.array([np.argmin([(x_i-y_k)@(x_i-y_k) for y_k in centroids])
        # Update centroids step
        centroids = []
        for k in range(K):
            if (C == k).any():
                centroids.append( X[C == k].mean(axis = 0) )
            else: # if there are no data points assigned to this certain cen
                centroids.append( X[np.random.choice(len(X))] )
    return np.array(centroids) , C

# Load data for activity
in_data = loadmat('Period11Activity.mat')
X = in_data['X']

rows, cols = np.shape(X)
```

Question 1-a

In [122…

```python
# k-means with 2 clusters
centroids, C = kMeans(X.transpose(), K = 2)

print('X = ', X, sep="\n", end='\n\n')
print('centroid assigned = ',C, sep="\n", end='\n\n')
print('centroids =', centroids.T.round(3), sep="\n", end='\n\n')
```

```
X =
[[ 4  7  2  8  7  4  2]
 [ 9  3  5  6 10  5  5]
 [ 4  8  3  7  6  4  1]
 [ 9  2  6  5  9  5  4]
 [ 4  9  2  8  7  4  1]]

centroid assigned =
[1 0 1 0 1 1 1]

centroids =
[[7.5 3.8]
 [4.5 6.8]
 [7.5 3.6]
 [3.5 6.6]
 [8.5 3.6]]
```

In [123…

```python
# Construct rank-2 approximation using clusters
```

```
Xhat_2 = np.zeros((rows,cols),float)
for i in range(cols):
    Xhat_2[:,i]=centroids.transpose()[:,C[i]]

print('Rank-2 Approximation = ', Xhat_2.round(3), sep="\n", end='\n\n')
```

```
Rank-2 Approximation =
[[3.8 7.5 3.8 7.5 3.8 3.8 3.8]
 [6.8 4.5 6.8 4.5 6.8 6.8 6.8]
 [3.6 7.5 3.6 7.5 3.6 3.6 3.6]
 [6.6 3.5 6.6 3.5 6.6 6.6 6.6]
 [3.6 8.5 3.6 8.5 3.6 3.6 3.6]]
```

The 2 rank matrix is a close rounding to the original

Question 1-b

In [124…

```
# k-means with 3 clusters
centroids, C = kMeans(X.transpose(), K = 3)

# add code here
print('X = ', X, sep="\n", end='\n\n')
print('centroid assigned = ',C, sep="\n", end='\n\n')
print('centroids =', centroids.T.round(3), sep="\n", end='\n\n')
```

```
X =
[[ 4  7  2  8  7  4  2]
 [ 9  3  5  6 10  5  5]
 [ 4  8  3  7  6  4  1]
 [ 9  2  6  5  9  5  4]
 [ 4  9  2  8  7  4  1]]

centroid assigned =
[1 2 1 0 0 1 1]

centroids =
[[7.5  3.   7.  ]
 [8.   6.   3.  ]
 [6.5  3.   8.  ]
 [7.   6.   2.  ]
 [7.5  2.75 9.  ]]
```

In [125…

```
# Construct rank-3 approximation using clusters
# add code here

Xhat_3 = np.zeros((rows,cols),float)
for i in range(cols):
    Xhat_3[:,i]=centroids.transpose()[:,C[i]]

print('Rank-3 Approximation = ', Xhat_3.round(3), sep="\n", end='\n\n')
```

```
Rank-3 Approximation =
[[3.   7.   3.   7.5  7.5  3.   3.  ]
 [6.   3.   6.   8.   8.   6.   6.  ]
 [3.   8.   3.   6.5  6.5  3.   3.  ]
 [6.   2.   6.   7.   7.   6.   6.  ]
 [2.75 9.   2.75 7.5  7.5  2.75 2.75]]
```

The 3 rank approx is a closer approx than the 2 rank but still rounds ratings a bit

Question 1-c

In [126…

```python
U,s,VT = np.linalg.svd(X,full_matrices=False)

print('U = ',U.round(3), sep="\n", end='\n\n')
print('Singular Values = ',s.round(3), sep="\n", end='\n\n')
print('V^T = ',VT.round(3), sep="\n", end='\n\n')
```

```
U =
[[ 0.419  0.319  0.565 -0.634  0.043]
 [ 0.506 -0.469  0.402  0.428 -0.424]
 [ 0.402  0.372 -0.582 -0.106 -0.592]
 [ 0.466 -0.552 -0.424 -0.318  0.444]
 [ 0.436  0.485 -0.019  0.55   0.521]]

Singular Values =
[32.952 10.165  1.788  0.699  0.407]

V^T =
[[ 0.418  0.38   0.25   0.456  0.536  0.3    0.184]
 [-0.441  0.695 -0.289  0.34  -0.177 -0.04  -0.301]
 [-0.191 -0.286 -0.664  0.329  0.3   -0.142  0.472]
 [ 0.329  0.445 -0.363 -0.626  0.276 -0.301  0.062]
 [ 0.174 -0.306 -0.252  0.121  0.385 -0.023 -0.806]]
```

In [127…

```python
# svd rank-1 approximation
T = U[:,0]
print('Taste for rank-1 = ', T.round(3), sep="\n", end='\n\n')

W = s[0]*VT[0,:]
print('Weights for rank-1 = ',W.round(3), sep="\n", end='\n\n')

X_1 = s[0]*U[:,[0]]@VT[[0],:]
print("Rank-1 Approximation = ",X_1.round(3), sep="\n", end='\n\n')
```

```
Taste for rank-1 =
[0.419 0.506 0.402 0.466 0.436]

Weights for rank-1 =
[13.773 12.521  8.24  15.017 17.647  9.886  6.068]

Rank-1 Approximation =
[[5.766 5.242 3.45  6.286 7.387 4.139 2.54 ]
 [6.964 6.331 4.167 7.593 8.923 4.999 3.068]
 [5.538 5.035 3.313 6.038 7.095 3.975 2.44 ]
 [6.419 5.835 3.84  6.998 8.224 4.607 2.828]
 [6.006 5.461 3.594 6.549 7.696 4.311 2.646]]
```

In [128…

```python
# svd rank-2 approximation
T = U[:,:2] # add code here

print('Taste for rank-2 = ', T.round(3), sep="\n", end='\n\n')

W = np.vstack((s[0]*VT[0,:],s[1]*VT[1,:]))
print('Weights for rank-2 = ', W.round(3), sep="\n", end='\n\n')

X_2 = s[:2]*U[:,[0,1]]@VT[[0,1],:]

print('Rank-2 Approximation = ',X_2.round(3), sep="\n", end='\n\n')
```

```
Taste for rank-2 =
[[ 0.419  0.319]
 [ 0.506 -0.469]
 [ 0.402  0.372]
 [ 0.466 -0.552]
 [ 0.436  0.485]]

Weights for rank-2 =
[[13.773 12.521  8.24  15.017 17.647  9.886  6.068]
 [-4.488  7.06  -2.935  3.458 -1.802 -0.403 -3.059]]

Rank-2 Approximation =
[[4.336 7.491 2.515 7.388 6.813 4.01  1.565]
 [9.069 3.02  5.543 5.971 9.768 5.188 4.503]
 [3.868 7.662 2.221 7.325 6.425 3.825 1.301]
 [8.897 1.937 5.461 5.089 9.219 4.83  4.517]
 [3.83  8.884 2.171 8.226 6.822 4.116 1.163]]
```

In [129…

```python
# Use svd to predict Jon's ratings

# first two tastes
T = U[:,:2]

# tastes for which we have ratings
G = T[:2,:2]

# ratings for first two movies
```

```python
y = np.array([[6], [4]])

# use first two movies to find weights for tastes
a = np.linalg.inv(G.transpose()@G)@G.transpose()@y

# now use weights and tastes to predict all ratings
Jon_ratings = T@a

print('Jon ratings =',Jon_ratings.round(3), sep="\n", end='\n\n')
```

```
Jon ratings =
[[6.   ]
 [4.   ]
 [6.014]
 [3.231]
 [6.832]]
```

In [130…

```python
U = (1/2) * np.array([
    [1,  1],
    [-1,  1],
    [-1, -1],
    [1, -1]
])
γ = 1 # CHANGE
S = np.array([
    [1, 0],
    [0, γ]
])
V = (1/np.sqrt(2)) * np.array([
    [1, 1],
    [1, -1]
])
```

Questoin 2-a

$$\min_w ||Xw - y||_2^2$$

$$w = (X^T X)^{-1} X^T y$$

$$X = USV^T$$

$$X^T X = (USV^T)^T (USV^T) = VS^T U^T USV^T = VS^T SV^T$$

$$(X^T X)^{-1} = (VS^T SV^T)^{-1} = V(S^T S)^{-1} V^T$$

$$w = V(S^T S)^{-1} V^T VS^T U^T y = VS^{-1} U^T y$$

$$w = VS^{-1} U^T y$$

Question 2-b

In [131…
```
y = np.array([
    [1],
    [0],
    [0],
    [1]
])
```

$$w = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{\gamma} \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$w = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{\gamma} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$w = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ \frac{1}{\gamma} \end{bmatrix}$$

$$w = \frac{1}{2\sqrt{2}} \begin{bmatrix} \frac{\gamma+1}{\gamma} \\ \frac{\gamma-1}{\gamma} \end{bmatrix}$$

$$||y - Xw||_2^2$$

$$(y - Xw)^T(y - Xw) = (y^T - w^T X^T)(y - Xw) = y^T y - y^T Xw - w^T X^T y + w^T X^T X$$

$$||y - Xw||_2^2 = y^T y - 2y^T Xw + w^T X^T Xw = y^T y - 2y^T (USV^T)w + w^T (USV^T)^T (U$$

$$y^T y = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = 1^2 + 0^2 + 0^2 + 1^2 = 2$$

$$2y^T USV^T w = 2 \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix} \frac{1}{2\sqrt{2}} \begin{bmatrix} 1+\gamma & 1-\gamma \\ -1+\gamma & -1-\gamma \\ -1-\gamma & -1+\gamma \\ 1-\gamma & 1+\gamma \end{bmatrix} \frac{1}{2\sqrt{2}} \begin{bmatrix} \frac{\gamma+1}{\gamma} \\ \frac{\gamma-1}{\gamma} \end{bmatrix} = 2\sqrt{2}$$

$$w^T V S^T S V^T w = 2(\gamma + 1)$$

$$2 - 2\sqrt{2} + 2(\gamma + 1)$$

$$||w||_2^2 = w^T w = \left(\frac{1}{2\sqrt{2}}\right)^2 \begin{bmatrix} \frac{\gamma+1}{\gamma} & \frac{\gamma-1}{\gamma} \end{bmatrix} \begin{bmatrix} \frac{\gamma+1}{\gamma} \\ \frac{\gamma-1}{\gamma} \end{bmatrix} = \frac{\gamma^2+1}{4\gamma^2}$$

As γ approaches 0 $||w||_2^2$ goes to infinity

Question 2-c

$$w = \frac{1}{\sigma_1} v_1 u_1^T y = \frac{1}{1} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$||y - Xw||_2^2 = y^T y - 2y^T USV^T w + w^T VS^T SV^T w = \sqrt{2}$$

$$||w||_2^2 = w^T w = 1$$

The values are constant with the rank 1 approx