Homework 3B

All hand calculations attached to this document after the code.

All work done by Delaney Gomen.

1 PCA

1. Calculate the eigenvalues of $X^T X$ by code or calculators. (Do not subtract mean or normalize data)
2. Calculate the eigenvalues and eigenvectors of XX^T . Solve this by hand AND using code you implement to verify your solution. (Do not subtract mean or normalize data) (Note that all eigenvectors need to be normalized.)
3. Are the non-zero eigenvalues of $X^T X$ and those of XX^T the same? Why or why not?
4. For dataset X, there are 6 samples with 2 dimensions. Reduce X dimensions to 1. Print/write the reduced 1-dimension data. It should be a 1 ×6 vector. Solve this by hand AND using code you implement to verify your solution. (Please subtract mean and normalize data.)

In [128]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [102]:
```
X = np.array([[2,3,3,4,5,7],[2,4,5,5,6,8]], dtype=np.float64)

print(X)
```

```
[[2. 3. 3. 4. 5. 7.]
 [2. 4. 5. 5. 6. 8.]]
```

In [103]:
```python
# Question 1.1

print('Question 1.1\n')

XtX = X.T@X

print('Matrix:\n', XtX,  '\n')

eigvals1, eigvecs1 = np.linalg.eig(XtX)

print('Eigenvalues:\n', eigvals1, '\n\n', 'Eigenvectors:\n', eigvecs1)
```

```
Question 1.1

Matrix:
 [[  8.  14.  16.  18.  22.  30.]
 [ 14.  25.  29.  32.  39.  53.]
 [ 16.  29.  34.  37.  45.  61.]
 [ 18.  32.  37.  41.  50.  68.]
 [ 22.  39.  45.  50.  61.  83.]
 [ 30.  53.  61.  68.  83. 113.]]

Eigenvalues:
 [ 2.81035710e+02+0.00000000e+00j -5.81067532e-15+0.00000000e+00j
  9.64290269e-01+0.00000000e+00j -3.23092317e-15+0.00000000e+00j
  1.33527898e-18+2.05406773e-17j  1.33527898e-18-2.05406773e-17j]

 Eigenvectors:
 [[-1.67802227e-01+0.j          -9.39105329e-01+0.j
  -2.99872629e-01+0.j          -8.04437242e-01+0.j
  -6.54024328e-05+0.00570128j -6.54024328e-05-0.00570128j]
 [-2.98045267e-01+0.j          -7.85862200e-03+0.j
   1.91390370e-01+0.j           1.80165089e-01+0.j
  -1.89481554e-04-0.00275661j -1.89481554e-04+0.00275661j]
 [-3.44387193e-01+0.j          -2.04324172e-01+0.j
   8.32589683e-01+0.j          -2.78692195e-01+0.j
  -1.27501430e-01-0.02853984j -1.27501430e-01+0.02853984j]
 [-3.81946380e-01+0.j           5.50103540e-02+0.j
   4.14540554e-02+0.j           3.91153564e-01+0.j
   8.16822838e-01+0.j           8.16822838e-01-0.j        ]
 [-4.65847494e-01+0.j           1.17879330e-01+0.j
  -1.08482259e-01+0.j          -2.18690693e-01+0.j
  -5.23662829e-01+0.16818361j -5.23662829e-01-0.16818361j]
 [-6.33649721e-01+0.j           2.43617282e-01+0.j
  -4.08354888e-01+0.j           2.04756430e-01+0.j
  -3.79676665e-02-0.10834732j -3.79676665e-02+0.10834732j]]
```

In [104]:
```
# Question 1.2

print('Question 1.2\n')

XXt = X@X.T

print('Matrix:\n', XXt, '\n')

eigvals2, eigvecs2 = np.linalg.eig(XXt)

print('Eigenvalues:\n', eigvals2, '\n\n', 'Eigenvectors:\n', eigvecs2)
```

```
Question 1.2

Matrix:
 [[112. 137.]
 [137. 170.]]

Eigenvalues:
 [  0.96429027 281.03570973]

 Eigenvectors:
 [[-0.7768816  -0.62964671]
 [ 0.62964671 -0.7768816 ]]
```

In [105]:
```
# Question 1.3

print('Question 1.3\n')

print('Eigenvalues of first matrix (rounded):', list(np.around(eigvals1,decimals=8
)))
print('Eigenvalues of second matrix (rounded):', list(np.around(eigvals2,decimals=8
)))
```

```
Question 1.3

Eigenvalues of first matrix (rounded): [(281.03570973+0j), (-0+0j), (0.96429027+0
j), (-0+0j), 0j, -0j]
Eigenvalues of second matrix (rounded): [0.96429027, 281.03570973]
```

ANALYSIS We can see that the two matrices share non-zero eigenvalues. The construction of the matrix can be manipulated if we multiply each side by X.

$X^TXv = \lambda v$

$XX^TXv = X\lambda v$

$(XX^T)(Xv) = \lambda(Xv)$

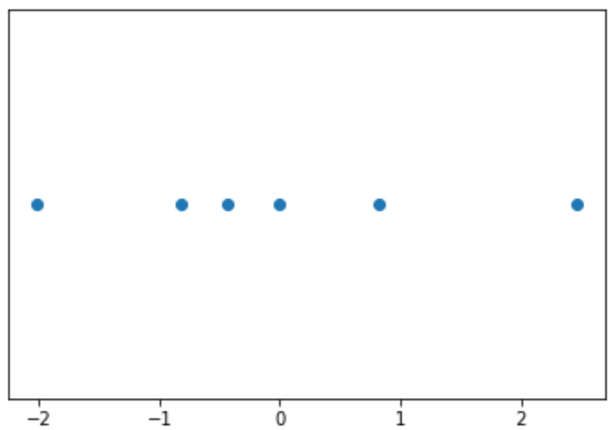
Since we have assumed that λ is non-zero, X is non-zero, and v is non-zero, we have shown that the non-zero eigenvalues of XX^T are also eigenvalues of X^TX.

In [149]:

```
# Question 1.4

from sklearn.preprocessing import StandardScaler
sc = StandardScaler() # Standardize features by removing the mean and scaling to un
it variance

# sc.fit_transform(X): X.shape = [n_samples, n_features]

X_std = sc.fit_transform(X.T)

print(X_std.shape, '\n')

# np.cov: Each row of m represents a variable, and each column a single observation
of all those variables

covmatrix = np.cov(X_std.T)

print(covmatrix, '\n')

eigvals3, eigvecs3 = np.linalg.eig(covmatrix)

print('Eigenvalues:\n', eigvals3, '\n\n', 'Eigenvectors:\n', eigvecs3, '\n')

#perform dimensionality reduction

eigenpairs = [(np.abs(eigvals3[i]), eigvecs3[:,i]) for i in range(len(eigvals3))]

eigenpairs.sort(reverse=True)

print(eigenpairs)

w = np.hstack((eigenpairs[0][1][:, np.newaxis]))

print(w)

y = X_std.dot(w)

print(y)

plt.scatter(y, y=[0]*6)
plt.yticks([])
```

```
(6, 2)

[[1.2        1.14039467]
 [1.14039467 1.2       ]]

Eigenvalues:
 [2.34039467 0.05960533]

 Eigenvectors:
 [[ 0.70710678 -0.70710678]
 [ 0.70710678  0.70710678]]

[(2.340394668524893, array([0.70710678, 0.70710678])), (0.05960533147510749, array
([-0.70710678,  0.70710678]))]
[0.70710678 0.70710678]
[-2.02792041 -0.82031104 -0.4330127   0.          0.82031104  2.46093311]
```

Out[149]: ([], <a list of 0 Text yticklabel objects>)



2 EM

1. Can Eq.3 be solved directly?

ANALYSIS Eq. 3 is our complete likelihood that is easier to solve for than our observed likelihood. We need to take the expectation of the latent variable (z, the probability that the result came from coin B or alternatively the outcome of flipping coin A) in order to maximize the parameters of interest (π, p, q). As discussed in class, we can only try to estimate z, which means our solution for theta is only as good as our initialization. If we could solve this equation directly, we would not need to have an expectation and maximization step.

1. Write code to implement the EM algorithm based on Eq.4-7. Are the results of parameters estimation the same with different initialization?

ANALYSIS Yes, the results of the parameters are different when you have different initialization values. The initialized theta works as a "prior" and influences the first estimation of μ. Your first estimation of μ will affect your revised theta, and so on until the parameters converge.

In [222]:

```
# Question 2.7

def e_step(theta, y):

    mu = []

    for yj in y:

        if yj > 0:

            muj = (theta['pi']*theta['p']) / ((theta['pi']*theta['p']) + ((1-theta[
'pi'])*theta['q']))

        elif yj < 1:

            muj = (theta['pi']*(1-theta['p'])) / ((theta['pi']*(1-theta['p'])) + ((
1-theta['pi'])*(1-theta['q'])))

        mu.append(muj)

    return mu

def m_step(mu, y):

    theta = {}

    pi = (1/len(mu))*sum(mu)

    theta['pi'] = pi

    p = sum(np.array(mu) * np.array(y)) / sum(mu)

    theta['p'] = p

    q = sum((1-np.array(mu)) * np.array(y)) / sum((1-np.array(mu)))

    theta['q'] = q

    return theta
```

In [225]:
```python
def em(theta, y):

    print('Initialized theta: ', theta, '\n')

    newtheta = theta

    for i in np.arange(20):

        mu = e_step(newtheta, y)
        newtheta = m_step(mu, y)

        print('Iteration ', i+1, ':\n')

        print('mu', i+1, ':', mu, '\n')

        print('theta', i+1, ':', newtheta, '\n')

theta = {'pi':0.4, 'p':0.6, 'q':0.7}

y = [1.0,1.0,0.0,1.0,0.0,0.0,1.0,0.0,1.0,1.0]

em(theta,y)
```

```
Initialized theta:  {'pi': 0.4, 'p': 0.6, 'q': 0.7}

Iteration  1 :

mu 1 : [0.36363636363636365, 0.36363636363636365, 0.47058823529411764, 0.363636363
63636365, 0.47058823529411764, 0.47058823529411764, 0.36363636363636365, 0.4705882
3529411764, 0.36363636363636365, 0.36363636363636365]

theta 1 : {'pi': 0.4064171122994653, 'p': 0.5368421052631579, 'q': 0.6432432432432
431}

Iteration  2 :

mu 2 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.36363636363
63638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.47058823529
411764, 0.3636363636363638, 0.3636363636363638]

theta 2 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.6432432432432
431}

Iteration  3 :

mu 3 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.36363636363
63638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.47058823529
411764, 0.3636363636363638, 0.3636363636363638]

theta 3 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.6432432432432
431}

Iteration  4 :

mu 4 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.36363636363
63638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.47058823529
411764, 0.3636363636363638, 0.3636363636363638]

theta 4 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.6432432432432
431}

Iteration  5 :

mu 5 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.36363636363
63638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.47058823529
411764, 0.3636363636363638, 0.3636363636363638]

theta 5 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.6432432432432
431}

Iteration  6 :

mu 6 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.36363636363
63638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.47058823529
411764, 0.3636363636363638, 0.3636363636363638]

theta 6 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.6432432432432
431}

Iteration  7 :

mu 7 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.36363636363
63638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.47058823529
411764, 0.3636363636363638, 0.3636363636363638]

theta 7 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.6432432432432
431}
```

```
Iteration  8 :

mu 8 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.36363636363
63638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.47058823529
411764, 0.3636363636363638, 0.3636363636363638]

theta 8 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.6432432432432
431}

Iteration  9 :

mu 9 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.36363636363
63638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.47058823529
411764, 0.3636363636363638, 0.3636363636363638]

theta 9 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.6432432432432
431}

Iteration  10 :

mu 10 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 10 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  11 :

mu 11 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 11 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  12 :

mu 12 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 12 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  13 :

mu 13 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 13 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  14 :

mu 14 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 14 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}
```

```
Iteration  15 :

mu 15 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 15 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  16 :

mu 16 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 16 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  17 :

mu 17 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 17 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  18 :

mu 18 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 18 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  19 :

mu 19 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 19 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}

Iteration  20 :

mu 20 : [0.3636363636363638, 0.3636363636363638, 0.47058823529411764, 0.3636363636
363638, 0.47058823529411764, 0.47058823529411764, 0.3636363636363638, 0.4705882352
9411764, 0.3636363636363638, 0.3636363636363638]

theta 20 : {'pi': 0.40641711229946537, 'p': 0.536842105263158, 'q': 0.643243243243
2431}
```

In [226]:

```
theta = {'pi':0.5, 'p':0.5, 'q':0.5}

y = [1.0,1.0,0.0,1.0,0.0,0.0,1.0,0.0,1.0,1.0]

em(theta,y)
```

```
Initialized theta:  {'pi': 0.5, 'p': 0.5, 'q': 0.5}

Iteration  1 :

mu 1 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 1 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  2 :

mu 2 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 2 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  3 :

mu 3 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 3 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  4 :

mu 4 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 4 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  5 :

mu 5 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 5 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  6 :

mu 6 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 6 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  7 :

mu 7 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 7 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  8 :

mu 8 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 8 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  9 :

mu 9 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 9 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  10 :

mu 10 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 10 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  11 :
```

```
mu 11 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 11 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  12 :

mu 12 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 12 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  13 :

mu 13 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 13 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  14 :

mu 14 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 14 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  15 :

mu 15 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 15 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  16 :

mu 16 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 16 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  17 :

mu 17 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 17 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  18 :

mu 18 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 18 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  19 :

mu 19 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 19 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}

Iteration  20 :

mu 20 : [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]

theta 20 : {'pi': 0.5, 'p': 0.6, 'q': 0.6}
```

(Question 1.2)

$$X = \begin{bmatrix} 2 & 3 & 3 & 4 & 5 & 7 \\ 2 & 4 & 5 & 5 & 6 & 8 \end{bmatrix}$$

$$XX^T = \begin{bmatrix} 2 & 3 & 3 & 4 & 5 & 7 \\ 2 & 4 & 5 & 5 & 6 & 8 \end{bmatrix} \begin{bmatrix} 2 & 2 \\ 3 & 4 \\ 3 & 5 \\ 4 & 5 \\ 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$= \begin{bmatrix} 4+9+9+16+25+49 & 4+12+15+20+30+56 \\ 4+12+15+20+30+56 & 4+16+25+25+36+64 \end{bmatrix}$$

$$= \begin{bmatrix} 112 & 137 \\ 137 & 170 \end{bmatrix}$$

Finding the eigen values + vectors.

1. Compute determinant of $A - \lambda I$ where $A = XX^T$

$$\det \begin{vmatrix} 112-\lambda & 137 \\ 137 & 170-\lambda \end{vmatrix} = (112-\lambda)(170-\lambda) - (137)^2$$

$$= \lambda^2 - 282\lambda + 271$$

2. Find roots of polynomial

$$\frac{282 \pm \sqrt{(-282)^2 + 4(1)(271)}}{2} = \frac{282 \pm \sqrt{282^2 - 4(271)}}{2}$$

$$= 141 \pm \sqrt{19610}$$

$$= (281.0357, 0.96429)$$

3. For each λ_j solve $(XX^T - \lambda I)x = 0$ to find eigenvector x.

$\lambda_1 = 281.0357$

$\lambda_2 = 0.96429$

$$XX^T = \begin{bmatrix} 112 & 137 \\ 137 & 170 \end{bmatrix}$$

- $\lambda_1 = 281.0357$

$$(XX^T - \lambda_1)I = \begin{bmatrix} -169.0357 & 137 \\ 137 & -111.0357 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Solve and get $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -0.810479 \\ -1 \end{pmatrix}$

Take the norm of the vector and divide to get normalized eigenvectors.

$$\text{norm}\begin{pmatrix} x \\ y \end{pmatrix} = 1.28719$$

$$\hat{\begin{pmatrix} x \\ y \end{pmatrix}}_{\lambda_1} = \begin{pmatrix} -.629647 \\ -.776816 \end{pmatrix}$$ → eigenvector for λ_1

- $\lambda_2 = 0.96429$

$$(XX^T - \lambda_2 I) = \begin{bmatrix} 111.0357 & 137 \\ 137 & 169.0357 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Solve and get $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -1 \\ 0.810479 \end{pmatrix}$

Divide by norm $= 1.28719$

$$\hat{\begin{pmatrix} x \\ y \end{pmatrix}}_{\lambda_2} = \begin{pmatrix} -0.7768 \\ 0.\cancel{62}964 \end{pmatrix}$$ → eigenvector for λ_2

(Question 1.4)

Find the standardized X matrix

$$X = \begin{bmatrix} 2 & 3 & 3 & 4 & 5 & 7 \\ 2 & 4 & 5 & 5 & 6 & 8 \end{bmatrix} \qquad \bar{x}_1 = 4 \qquad \bar{\sigma}_1 = 1.6329$$
$$\bar{x}_2 = 5 \qquad \bar{\sigma}_2 = 1.825$$

Take mean and std along each dimension.

$$X_{std} = \begin{bmatrix} -1.2247 & -0.612 & -0.612 & 0 & 0.612 & 1.8371 \\ -1.64 & -0.547 & 0 & 0 & 0.5479 & 1.6431 \end{bmatrix}$$

Take the covariance matrix by calculating $(X_{std} X_{std}^T)\frac{1}{6}$ or in other words $\frac{1}{6}\sum_{i=1}^{6} x_i y_i^T$ because the mean is centered at 0 from standardization.

$$cov_{X_{std}} = \begin{bmatrix} 1.2 & 1.140 \\ 1.140 & 1.2 \end{bmatrix} = \frac{1}{6}(X_{std})(X_{std})^T$$

Find the eigenvectors and eigenvalues of the matrix.

$$\det \begin{bmatrix} 1.2-\lambda & 1.14 \\ 1.14 & 1.2-\lambda \end{bmatrix} = (1.2-\lambda)^2 - (1.14)^2$$
$$= \lambda^2 - 2.4\lambda - 1.295$$

$\lambda = 2.34, 0.059$

Find the associated eigenvector for the largest λ since we are only going to one dimension.

$(Cov_{X_{std}} - \lambda I)x = 0$ where $\lambda = 2.34$

We solve to get $x = \begin{pmatrix} 0.7071 \\ 0.7071 \end{pmatrix}$

Now take our eigenvector and multiply by X_{std} to get our new data in 1 dimension

$$X_{std} = \begin{bmatrix} -1.22 & -0.612 & -0.612 & 0 & 0.612 & 1.837 \\ -1.64 & -0.547 & 0 & 0 & 0.5479 & 1.643 \end{bmatrix}$$

$$w = [0.7071 \quad 0.7071]$$

$$[0.7071 \quad 0.7071] \begin{bmatrix} -1.22 & -0.612 & -0.612 & 0 & 0.612 & 1.837 \\ -1.64 & -0.547 & 0 & 0 & 0.547 & 1.643 \end{bmatrix}$$

$$= [-2.027 \quad -0.8203 \quad -0.433 \quad 0 \quad 0.8203 \quad 2.460]$$

And you're done!

Q 2.2 Depending on y_j, $\mu_j^{(i+1)}$ can be written one of two ways.

WHEN $y_j = 1$

$$\mu_j^{(i+1)} = \frac{\pi^{(i)} p^{(i)}}{\pi^{(i)} p^{(i)} + (1-\pi^{(i)}) q^{(i)}}$$

WHEN $y_j = 0$

$$\mu_j^{(i+1)} = \frac{\pi^{(i)} (1-p^{(i)})}{\pi^{(i)} (1-p^{(i)}) + (1-\pi^{(i)})(1-q^{(i)})}$$

$$\begin{Bmatrix} y_j = 1 & \text{for } j = 1, 2, 4, 7, 9, 10 \\ y_j = 0 & \text{for } j = 3, 5, 6, 8 \end{Bmatrix}$$

$\boxed{\mu_j^{(1)} \text{ when } y_j = 1}$

$$\mu^{(1)} = \frac{(0.5)(0.5)}{(0.5)(0.5) + (0.5)(0.5)} = \frac{0.25}{0.5} = \frac{1}{2}$$

$\boxed{\mu_j^{(1)} \text{ when } y_j = 0}$

$$\mu^{(1)} = \frac{0.5(0.5)}{(0.5)(0.5) + (0.5)(0.5)} = \frac{0.25}{0.5} = \frac{1}{2}$$

Q. 2.3 we calculated $\mu^{(1)} = 1/2$ for all j.

$$\pi^{(1)} = \frac{1}{10} \sum^{10} \left(\frac{1}{2}\right) = 5/10 = 1/2$$

$$p^{(1)} = \sum_j^{10} (1/2)(y_j) \Big/ \sum^{10}(1/2) = (1/2)(6)/5 = 3/5$$

$$q^{(1)} = \sum_j^{10} (1/2)(y_j) \Big/ \sum^{10}(1/2) = (1/2)(6)/5 = 3/5$$

Q. 2.4. The other parameters calculated in 2.2, 2.3 are $\mu^{(1)} = 1/2$ and $\theta^{(1)} = (1/2, 3/5, 3/5) = (\pi^{(1)}, p^{(1)}, q^{(1)})$

$\boxed{\mu_j^{(2)} \text{ when } y_j = 1}$

$$\mu^{(2)} = \frac{(1/2)(3/5)}{(1/2)(3/5) + (1/2)(3/5)} = \frac{1}{2}$$

$\boxed{\mu_j^{(2)} \text{ when } y_j = 0}$

$$\mu^{(2)} = \frac{(1/2)(2/5)}{(1/2)(2/5) + (1/2)(2/5)} = \frac{1}{2}$$

Q 2.5 We calculated $\mu^{(2)} = 1/2$ for all j.

This is the same outcome as $\mu^{(1)}$.

So $\theta^{(1)} = \theta^{(2)}$ and $\theta^{(2)} = (\pi^{(2)}, p^{(2)}, q^{(2)}) = (1/2, 3/5, 3/5)$

Q 2.6 Now, do 2 iterations with $\theta^{(0)} = (0.4, 0.6, 0.7)$.
$\pi^{(0)}$ $p^{(0)}$ $q^{(0)}$

$\boxed{\mu_j^{(1)} \text{ when } y_j = 1}$

$$\mu^{(1)} = \frac{(0.4)(0.6)}{(0.4)(0.6) + (0.6)(0.7)} = 0.3636$$

$\boxed{\mu_j^{(1)} \text{ when } y_j = 0}$

$$\mu^{(1)} = \frac{(0.4)(0.4)}{(0.4)(0.4) + (0.6)(0.3)} = 0.47058$$

$\pi^{(1)} = 0.40641$
$p^{(1)} = 0.53684$
$q^{(1)} = 0.64323$

$\boxed{\mu_{(j)}^{(2)} \text{ when } y_j = 1}$

$$\mu^{(2)} = (0.40641)(0.53684) / [(0.40641)(0.53684) + (1-0.40641)(0.64323)]$$

$$= 0.36363$$

$\boxed{\mu_j^{(2)} \text{ when } y_j = 0}$

$$\mu^{(2)} = (0.406)(1-0.536) / [0.406(1-0.536) + (1-0.406)(1-0.6423)]$$
$$= 0.47057$$

We get roughly the same results as $\mu^{(1)}$, so we know that

$\pi(2) = 0.4061$

$p(2) = 0.53484$

$q(2) = 0.64323$

$\theta^{(2)}$ is not the same as it is in Question 2.5