ML Assignment 1

Tomas Langebaek Carrizosa

February 16, 2020

[1]

1 Exercises:

- 8.3

Join Probabilities			
a	b	c	p(a, b, c)
0	0	0	0,192
0	0	1	0,144
0	1	0	0,048
0	1	1	0,216
1	0	0	0,192
1	0	1	0,064
1	1	0	0,048
1	1	1	0,096

a).

$p(a, b) = \sum_{n=0}^{1} p(a, b, c = 1)$

$= 0{,}048{+}0{,}096 = 0{,}144$

$p(a) = \sum_{n=0}^{1} \sum_{j=0}^{1} p(a, b = n, c = j)$

[1]Some of the exercises in this assignment were discussed with Chenxian Zhang, Nathan Hu, Valeria Marin, Richmond Horikawa and Amalia Riegelhith.

= 0,192 + 0,064 + 0,048 + 0,096 = 0,4

$p(b) = \sum_{n=0}^{1} \sum_{j=0}^{1} p(a=n, b, c=j)$

= 0,048 + 0,216 + 0,048 + 0,096 = 0,408

p(a)p(b) = (0,4)(0,408)

$= 0,1632 \neq 0,144 = p(a,b)$

q.e.d

b).

$p(a,b|c) \stackrel{?}{=} p(a|c)p(b|c)$

On c = 1, a = 1, b = 1:

$$p(a \mid c)p(b \mid c)$$

= (0,064 + 0,096)(0,216 + 0,096)

= (0,16 + 0,312)

0,04992/p(c=1)

p(c=1) = 0,144 + 0,216 + 0,064 + 0,096

= 0,52

0,04992/0,52

$$= 0,096 = p(a=1, b=1 \mid c=1)$$

q.e.d

On c = 1, a = 0, b = 1:

$$p(a \mid c)p(b \mid c)$$

= (0,144 + 0,216)(0,216 + 0,096)

= (0,36 + 0,312)

0,112/p(c=1)

p(c=1) = 0,144 + 0,216 + 0,064 + 0,096

= 0,52

0,112/0,52

$$= 0,216 = p(a = 0, b = 1 \mid c = 1)$$

q.e.d

On c = 1, a = 1, b = 0:

$$p(a \mid c)p(b \mid c)$$

= (0,064 + 0,096)(0,144 + 0,064)

= (0,16 + 0,208)

0,003328/p(c=1)

p(c=1) = 0,144 + 0,216 + 0,064 + 0,096

= 0,52

0,003328/0,52

$$= 0,064 = p(a = 1, b = 0 \mid c = 1)$$

q.e.d

On c = 1, a = 0, b = 0:

$$p(a \mid c)p(b \mid c)$$

= (0,144 + 0,216)(0,144 + 0,064)

= (0,36 + 0,208)

0,07488/p(c=1)

p(c=1) = 0,144 + 0,216 + 0,064 + 0,096

= 0,52

0,07488/0,52

$$= 0,144 = p(a = 0, b = 0 \mid c = 1)$$

q.e.d

On c = 0, a = 1, b = 1:

$$p(a \mid c)p(b \mid c)$$

= (0,192 + 0,048)(0,048 + 0,048)

= (0,24 + 0,096)

0,02304/p(c=0)

p(c=0) = 0,192 + 0,048 + 0,192 + 0,048

= 0,48

0,02304/0,48

$$= 0,048 = p(a = 1, b = 1 \mid c = 0)$$

q.e.d

On c = 0, a = 0, b = 1:

$$p(a \mid c)p(b \mid c)$$

= (0,192 + 0,048)(0,048 + 0,048)

= (0,24 + 0,096)

0,02304/p(c=0)

p(c=0) = 0,192 + 0,048 + 0,192 + 0,048

= 0,48

0,0204/0,48

$$= 0,048 = p(a = 0, b = 1 \mid c = 0)$$

q.e.d

On c = 0, a = 1, b = 0:

$$p(a \mid c)p(b \mid c)$$

= (0,192 + 0,048)(0,192 + 0,192)

= (0,24 + 0,384)

0,09216/p(c=0)

p(c=0) = 0,192 + 0,048 + 0,192 + 0,048

= 0,48

0,09216/0,48

$$= 0,192 = p(a = 0, b = 1 \mid c = 0)$$

q.e.d

On c = 0, a = 0, b = 0:

$$p(a \mid c)p(b \mid c)$$

= (0,192 + 0,048)(0,192 + 0,192)

= (0,24 + 0,384)

0,09216/p(c=0)

p(c=0) = 0,192 + 0,048 + 0,192 + 0,048

= 0,48

0,09216/0,48

$$= 0,192 = p(a = 0, b = 0 \mid c = 0)$$

q.e.d

- 8.4

$$p(a) = \sum_{n=0}^{1}\sum_{j=0}^{1} p(a, b = n, c = j)$$

= 0,4

$$p(b \mid c) = p(b \cap c)/p(c) = 0,216 + 0,096/0,52$$

= 0,6

$$p(c \mid a) = p(c \cap a)/p(a) = 0,064 + 0,096/0,4$$

= 0,4

$$p(a, b, c) \stackrel{?}{=} p(a)p(c \mid a)p(b \mid c)$$

= 0,096 = (0,4)(0,4)(0,6)

= 0,096

q.e.d

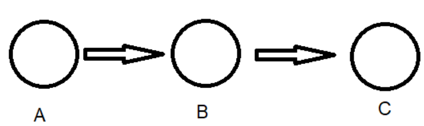


Figure 1: Graph representation

- 8.10

a).

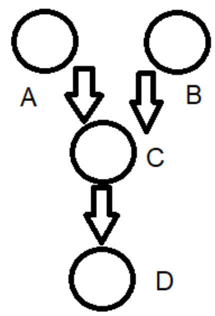


Figure 2: Graph representation

? $\rightarrow a \perp\!\!\!\perp | \emptyset$

p(a)p(b)p(c| $a, b)p(d \mid c)$

= p(a)p(b)

$$\sum_c \sum_d p(c \mid a, b) p(d \mid c)$$

=

$$p(a)p(b) \sum_c \sum_d p(d \mid c) p(c \mid a, b)$$

because we are adding over d given c the probability is 1

=

$$p(a)p(b) \sum_c p(c \mid a, b)$$

because we are adding over c given a, b the probability is 1

= p(a)p(b) = a $\perp\!\!\!\perp | \emptyset$

q.e.d

b).

D-Separation:

Nodes are head to head but it's descendants are in the set.

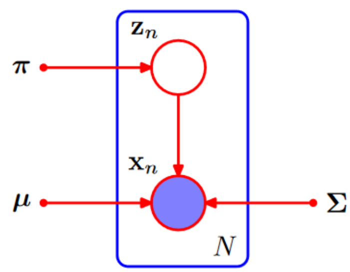
A node is head to tail but the node is not in the set.

None of the D-Separation conditions are satisfied.

The paths are not blocked, thus:

$a \not\perp\!\!\!\perp b \mid d$

- 9.5



There is a unique path from any Zn to any Zn going through:

$$\mu$$

and:

$$\sum$$

, which is tail to tail with respect to both:

$$\mu$$

and:

$$\sum$$

, so they are blocked. Each path from Xn to other Xn is head to tail with respect to to Zn, which is in the set, so they are blocked. Any path from Zn to any other Zn is blocked if we condition on:

$$\pi$$

treating it like an observed node (that is tail to tail), so the paths are blocked. Thus, Zns are D-Separated and is satisfied that

$$p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}) = \prod_{n=1}^{N} p(\mathbf{z}_n|\mathbf{x}_n, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}).$$

because all paths are blocked.

The following derivation can me made manualy:

2 Old Faithful:

- 1)

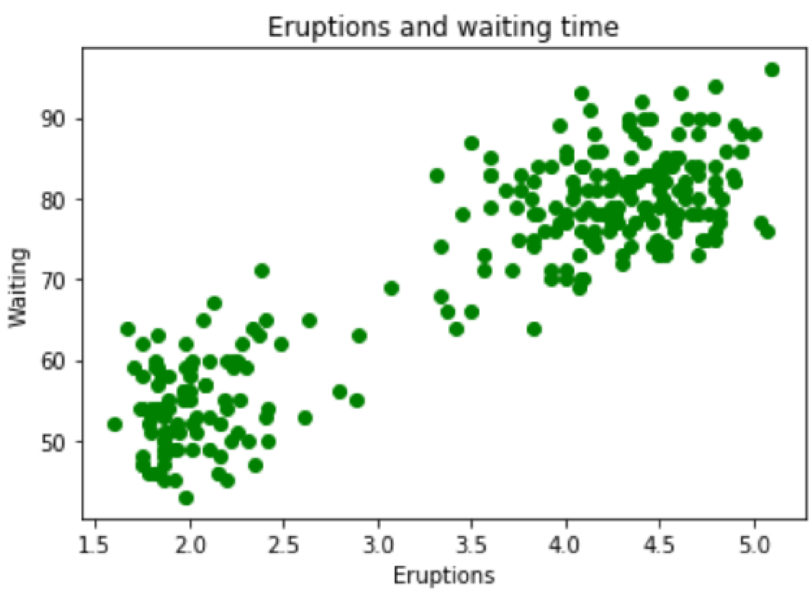


Figure 3: Old faithful data

- 2)

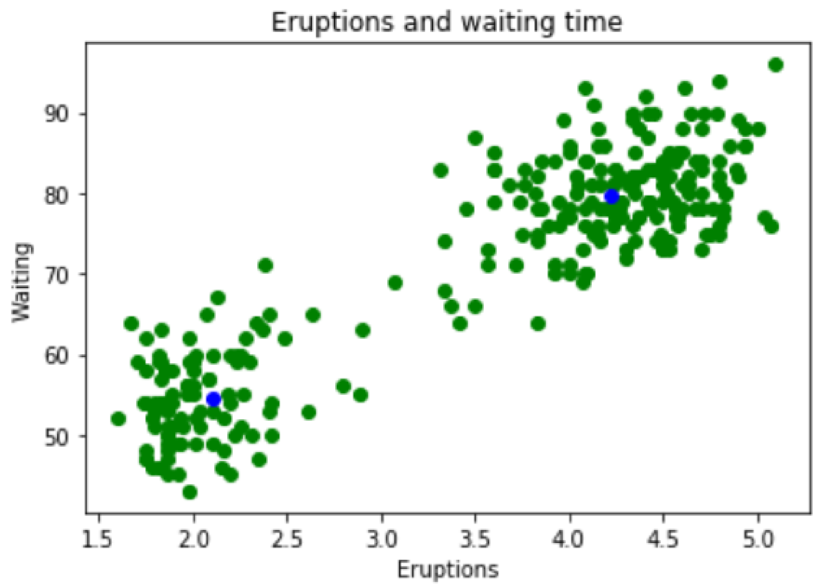


Figure 4: Old faithful data with means

```
Cluster #1:
cov1:
[[ 0.22240823  3.68439869]
 [ 3.68439869 61.03548314]]
mean1:
[ 4.22578402 79.67019661]
-----------
Cluster #2:
cov2:
[[ 0.2157424   3.57397302]
 [ 3.57397302 59.20617942]]
mean2:
[ 2.1111593  54.53216049]
```

EM algorithm:

```
while numberOfRepetitions > 0:

    #--------------------------------------------------------------------------
    #------------------------ E STEP ------------------------------------------
    #--------------------------------------------------------------------------
    # lists for storing the gamma values
    results1 = []
    results2 = []
    for n in range(len(x)):
        denominator = 0
        #Calculate Normal Cluster 1
        y1 = multivariate_normal.pdf([x[n],y[n]], mean=mean1, cov=cov1)
        y1 *= pi1
        denominator += y1
        #Calculate Normal Cluster 2
        y2 = multivariate_normal.pdf([x[n],y[n]], mean=mean2, cov=cov2)
        y2 *= pi2
        denominator += y2
        # Divide by denominator, the added previous values
        y1 /= denominator
        y2 /= denominator
        # add them to each list
        results1.append(y1)
        results2.append(y2)
    #--------------------------------------------------------------------------
    #------------------------ M STEP ------------------------------------------
    #--------------------------------------------------------------------------
    #Reset Covariance matrixes and means for Clusters:
    cov1 = [[0,0],[0,0]]
    cov2 = [[0,0],[0,0]]
    mean1 = [0,0]
```

```
mean2 = [0,0]
# add gammas
addedGammas1 = sum(results1)
addedGammas2 = sum(results2)
#new pySubKs
pi1 = addedGammas1/272 # 272 is the lenght of the data arrays
pi2 = addedGammas2/272 # 272 is the lenght of the data arrays
#calculation of new means:
for n in range(len(x)):
    # Multiplications of each component and it's gamma value
    mean1 += np.multiply([x[n], y[n]], results1[n])
    mean2 += np.multiply([x[n], y[n]], results2[n])
#divide by the sun of gammas
#assign NEW MEANS
mean1[0] = mean1[0]/addedGammas1
mean1[1] = mean1[1]/addedGammas1
mean2[0] = mean2[0]/addedGammas2
mean2[1] = mean2[1]/addedGammas2
#mean1 /= addedGammas1
#mean2 /= addedGammas2
#calculations of new covariance matrix

matrix1 = 0
matrix2 = 0

for n in range(len(x)):
    # Vector that stores one point
    vector1 = np.array([x[n],y[n]])
    # Reshape and trasnpose of vector
    # Matrix Multiplication
    matrix1 =  vector1.reshape(-1,1) @ vector1.reshape(1,2)
```

```
        # Multiplication of the matrix and it's gamma value
        matrixMu1 = np.multiply(matrix1, results1[n])
        # add to total matrix for this loop
        matrix1 = np.add(matrix1, matrixMu1)

        # Vector that stores one point
        vector2 = np.array([x[n],y[n]])
        # Reshape and trasnpose of vector
        # Matrix Multiplication
        matrix2 =  vector2.reshape(-1,1) @ vector2.reshape(1,2)
        # Multiplication of the matrix and it's gamma value
        matrixMu2 = np.multiply(matrix2, results2[n])
        # add to total matrix for this loop
        matrix2 = np.add(matrix2, matrixMu2)

    # Divide each total matrix by their added gamma values
    cov1 = np.divide(matrix1, addedGammas1)
    cov2 = np.divide(matrix2, addedGammas2)
    #decrease repetitions
    numberOfRepetitions -= 1
```

- 3)

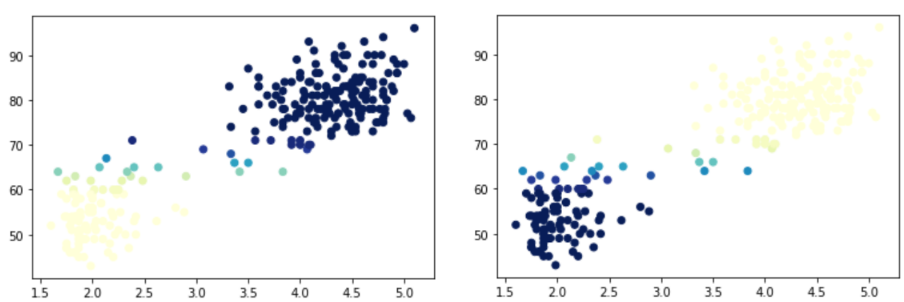


Figure 5: Marginal probabilities for each cluster

The code used for floating the marginal probabilities is the following:

```
plt.scatter(x,y,c=results2,cmap=cm.get_cmap('YlGnBu', 15),norm=mpl.colors.
Normalize(vmin=min(results2),vmax=max(results2)))
```

- 4)

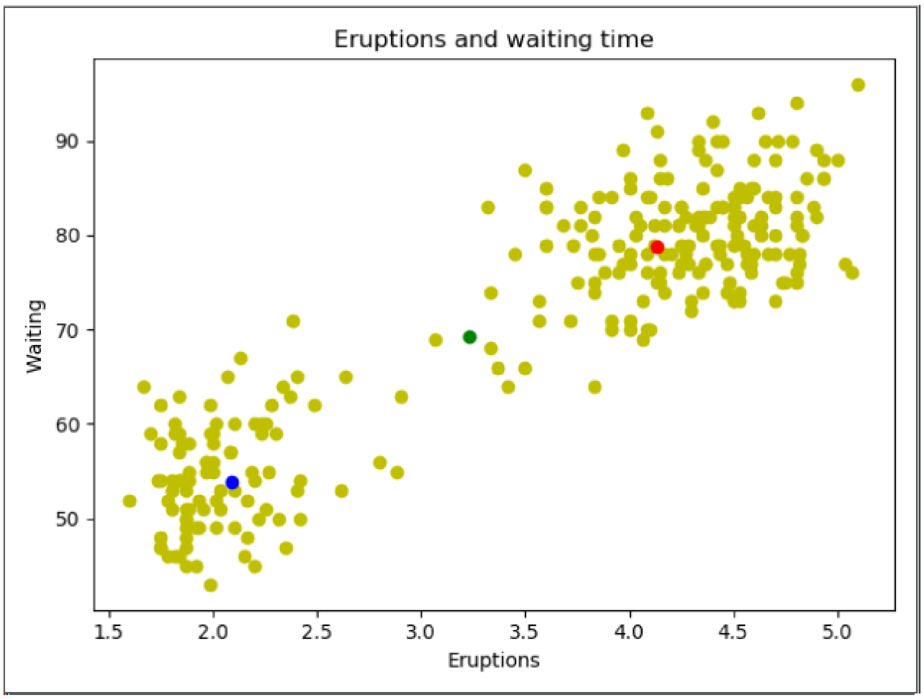


Figure 6: Graph representation

When using the EM algorithm for multiple mixtures with the Old Faithful data the graph in figure 6 was obtained. It shows how means tend to relocate. The third cluster appears to be on the mid-point between the two original mixtures. This is due to the fact that each mixture is maximized with respect to all the data points and taking into account the responsibilities of the other mixtures; therefore, the mean of the 3rd mixtures is located in a central point with respect to the data and the two other means. No more mixtures were added successfully to the EM implementation shown above, but for more mixtures, means are expected to group together near the two groups of points that surround the means in Figure 4. As the data is quite symmetrical (as a reflection on the x and y axis) this behavior is expected for even for than 4 mixtures.