



AMIRKABIR UNIVERSITY OF TECHNOLOGY
(TEHRAN POLYTECHNIC)
DEPARTMENT OF COMPUTER ENGINEERING

STATISTICAL PATTERN RECOGNITION

Assignment II

*Soroush Mahdi
99131050*

supervised by
Dr. Mohammad Rahmati

December 30, 2020



Contents

1 Do You Think You May Have COVID-19?	2
2 Making Decisions using Bayes Decision Rule	3
2.1 a	3
2.2 b	4
2.3 c	5
2.4 d	6
2.5 e	6
3 Automatic Saffron Cleaning Machine	8
4 Dealing with Prediction Error in Bayes Decision Rule	12
5 Separating Messi from Football Field	14
5.1 a	14
5.2 b	14
5.3 c	16
5.4 d	16
5.5 e	17
6 Maximum Likelihood Approach for Parameter Estimation	18
7 Here's Why Hitler Hated MLE So Bad	21
8 Further Study: When Will Civilization End?	22
9 Some Explanatory Questions	22
9.1 a	22
9.2 b	22
9.3 c	22
9.4 d	22
9.5 e	23
9.6 f	23
9.7 g	23
9.8 h	23



1 Do You Think You May Have COVID-19?

1.

$$1.\alpha \quad P(D^+) = \frac{4}{100}, \quad P(T^+ | D^+) = \frac{83}{100}, \quad P(T^- | D^-) = \frac{17}{100}, \quad P(T^+ | D^-) = \frac{17}{100} \rightarrow ?$$

$$P(D^+|T^+) = \frac{P(T^+|D^+) P(D^+)}{P(T^+|D^+) P(D^+) + P(T^+|D^-) P(D^-)} = \frac{\frac{332}{17264}}{\frac{83}{100} \left(\frac{4}{100} \right) + \frac{17}{100} \left(\frac{99}{100} \right)} = 0.0192 \approx 1.9\%$$

↳ probability that you
Actually have covid

$$P(T^+ | D^+) = \frac{P(T^+ \cap D^+)}{P(D^+)} = \frac{\frac{1}{100} \cdot \frac{1}{100}}{\frac{1}{100} \cdot \frac{1}{100} + \frac{2}{100} \cdot \frac{99}{100}} = \frac{1}{100} = 0.01$$

$$\Rightarrow P(D^+ | T^+) = \frac{\frac{81}{100} \times \frac{5}{10}}{\frac{81}{100} \times \frac{5}{10} + \frac{27}{100} \times \frac{95}{10}} = \frac{405}{2970} = 0,1363, 13,63\%$$

$$P(T_+ | D_+) = \frac{76}{100}, \quad P(T_- | D_-) = \frac{68}{100}, \quad P(D_+) = \frac{25}{1000} \Rightarrow P(D_-) = \frac{975}{1000}$$

$$P(D_+ | T_-) = \frac{P(T_- | D_+) P(D_+)}{P(T_- | D_+) P(D_+) + P(T_- | D_-) P(D_-)} = \frac{(0,24) \cdot (0,25)}{(0,24) \cdot (0,25) + (0,68) \cdot (0,75)} = 0,1669$$

$\downarrow 1 - \frac{76}{100} = \frac{24}{100}$

$\rightarrow = 0,0089 = 0,89\%$

$$P(D_+ | T_+) = \frac{P(T_+ | D_+) P(D_+)}{P(T_+ | D_+) P(D_+) + P(T_+ | D_-) P(D_-)} = \frac{(0.76)(0.25)}{(0.76)(0.25) + (0.32)(0.975)} = 0.331$$

$\hookrightarrow 1 - \frac{68}{100}, \frac{32}{100}$

$$= 0.0574; 5.74\% \leftarrow$$

1.f $P(D_+|T_+) = 5,74\%$
 $P(D_+|T_-) = 89\%$ } \Rightarrow yes he is more likely to have covid when he tests Positive than when he test negative

Figure 1: a-f



1.g

$$P(D_+|T_-) \approx 89\% \Rightarrow P(D_-|T_-) \approx 99,11\% \Rightarrow \text{yes he is}$$

$$1.h \quad P(D_+) = \frac{10}{100}, \quad P(T_+|D_+) = P, \quad P(T_+|D_-) = 1-P$$

$$P(D_+|T_+) = \frac{P(T_+|D_+) P(D_+)}{P(T_+|D_+) P(D_+) + P(T_+|D_-) P(D_-)} = \frac{\frac{P}{100}}{\frac{P}{100} + \frac{(1-P)(99)}{100}} = \frac{P}{P + 99 - 99P} = \frac{P}{99 - 98P}$$

$$1.i \quad P(D_+|T_+) = q \quad q = 20\%$$

$$1.j \quad q = 20\% = \frac{P}{99 - 98P} \Rightarrow \frac{P}{99 - 98P} = \frac{1}{5} \Rightarrow 5P = 99 - 98P \Rightarrow P = \frac{99}{103}$$

Figure 2: g-j

2 Making Decisions using Bayes Decision Rule

2.1 a

```

1 #mu1 = [1, 2]
2 mu2 = [-1, -3]
3 cov1 = [[1.8, -0.7], [-0.7, 1.8]]
4 cov2 = [[1.5, 0.3], [0.3, 1.5]]
5
6 x1,y1 = np.random.multivariate_normal(mu1, cov1, 1000).T
7 x2,y2 = np.random.multivariate_normal(mu2, cov2, 1000).T
8
9 xx1, yy1 = np.meshgrid(x1,y1)
10 pos1 = np.dstack((xx1,yy1))
11 rv1 = multivariate_normal(mean=mu1, cov=cov1)
12 fig2 = plt.figure()
13 ax2 = fig2.add_subplot(111)
14 ax2.contourf(xx1, yy1, rv1.pdf(pos1))
15
16 fig2.savefig('2a1.png')
17
18 xx2, yy2 = np.meshgrid(x2,y2)
19 pos2 = np.dstack((xx2,yy2))
20 rv2 = multivariate_normal(mean=mu2, cov=cov2)
21 fig2 = plt.figure()
22 ax2 = fig2.add_subplot(111)
23 ax2.contourf(xx2, yy2, rv2.pdf(pos2))
24
25 fig2.savefig('2a2.png')

```

Listing 1: 2a

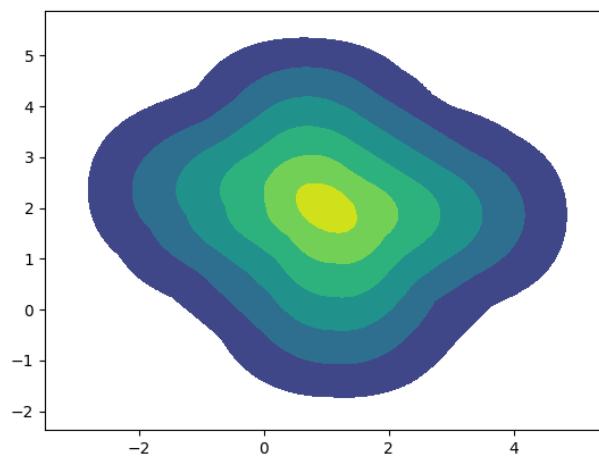


Figure 3: class 1

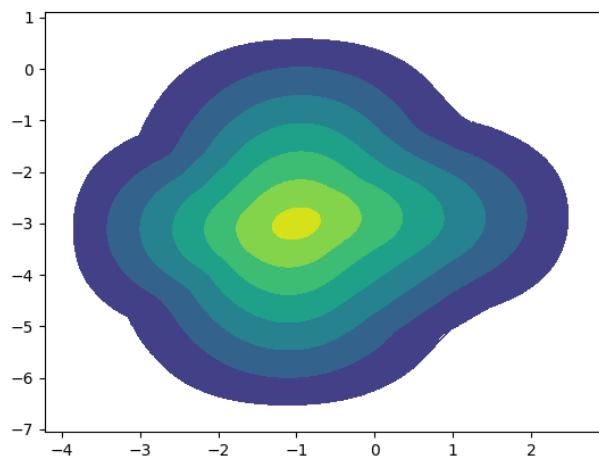


Figure 4: class 2

2.2 b

```

1 x = np.linspace(-30, 30, 15000)
2 plt.plot(x,-0.4*x-0.5 )
3
4 Y, X = np.mgrid[-30:30:300j, -30:30:300j]
5 plt.contour(X, Y, (63*X**2 - 128*X +36*Y**2 + 28*X*Y -172*Y + 236)/55 -
6 (25*X**2 + 20 * X + 25 *Y**2 -10*X*Y+140*Y+220)/36 +0.24 , levels=[0])
7 plt.grid()
8
9 plt.savefig('2b.png')

```

Listing 2: 2b

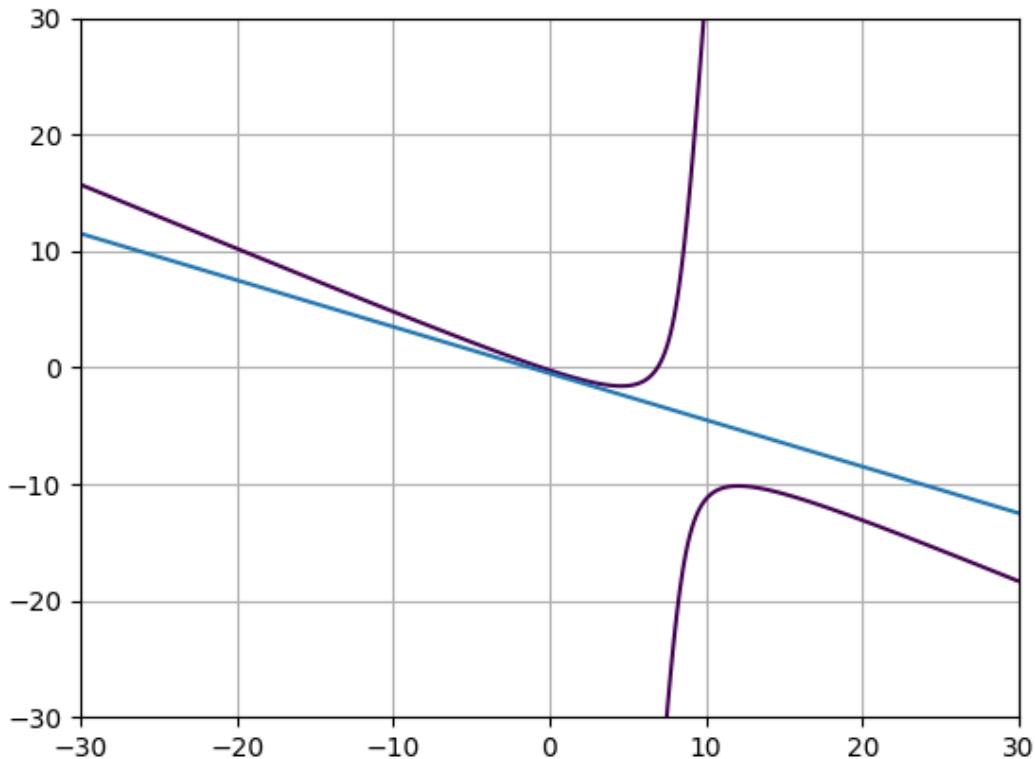


Figure 5: decision boundary

2.3 c

```

1  def MDC (x):
2      return 2*x[0] + 5 * x[1] + 0.4
3
4  def bayes(x):
5      mu1 = np.array([1, 2])
6      mu2 = np.array([-1, -3])
7      cov1 = np.array([[1.8, -0.7], [-0.7, 1.8]] )
8      cov2 = np.array([[1.5, 0.3], [0.3, 1.5]] )
9
10     return -0.5 * ((x-mu1).T@np.linalg.inv(cov1)@(x-mu1)-0.5*np.log(np.abs(np.linalg
11         .det(cov1)))) - (-0.5 * ((x-mu2).T@np.linalg.inv(cov2)@(x-mu2)-0.5*np.log(np.abs(
12             np.linalg.det(cov2)))))

13  mu1 = [1, 2]
14  mu2 = [-1, -3]
15  cov1 = [[1.8, -0.7], [-0.7, 1.8]]
16  cov2 = [[1.5, 0.3], [0.3, 1.5]]
17
18  x0 = np.random.multivariate_normal(mu1, cov1, 1000)
19  x1 = np.random.multivariate_normal(mu2, cov2, 1000)
20
21  mdc_err = 0
22  bayes_err = 0
23
24  for i in x0:
25      if MDC(i)<0:
26          mdc_err+=1
27

```



```
28     if bayes(i)<0:
29         bayes_err +=1
30
31 for i in x1:
32     if MDC(i)>=0:
33         mdc_err+=1
34
35     if bayes(i)>=0:
36         bayes_err +=1
37
38 print('MDC error: ',mdc_err/2000)
39 print('bayes error: ',bayes_err/2000)
40
41 #MDC error:  0.018
42 #bayes error:  0.012
```

Listing 3: 2c

2.4 d

```
1  TP_bayes_cost = 0
2  TP_bayes = 0
3
4  for i in x0:
5      if bayes(i)>=np.log(3):
6          TP_bayes_cost +=1
7
8      if bayes(i)>=0:
9          TP_bayes +=1
10
11
12 recall_cost = TP_bayes_cost/2000
13 recall_bayes = TP_bayes/2000
14
15 FP_cost = 0
16 FP_bayes = 0
17
18 for i in x1:
19     if bayes(i)>=np.log(3):
20         FP_cost+=1
21
22     if bayes(i)>=0:
23         FP_bayes +=1
24
25 precision_cost = TP_bayes_cost / (TP_bayes_cost + FP_cost)
26 precision_bayes = TP_bayes/(TP_bayes + FP_bayes)
27
28 fscore_cost = 2*precision_cost*recall_cost/(precision_cost+recall_cost)
29 fscore_bayes = 2*precision_bayes*recall_bayes/(precision_bayes+recall_bayes)
30
31 print('bayes with cost F score: : ',fscore_cost)
32 print('bayes F score: ',fscore_bayes)
33
34
35 # bayes with cost F score: :  0.6552534407519301
36 # bayes F score:  0.6606666666666667
```

Listing 4: 2d

2.5 e

```
1  avg_error_mdc = 0
2  avg_error_bayes = 0
3  for i in range(20):
4      x0 = np.random.multivariate_normal(mu1, cov1,1000)
5      x1 = np.random.multivariate_normal(mu2, cov2, 1000)
6
```



```
7     mdc_err = 0
8     bayes_err = 0
9
10    for i in x0:
11        if MDC(i)<0:
12            mdc_err+=1
13
14        if bayes(i)<0:
15            bayes_err +=1
16
17    for i in x1:
18        if MDC(i)>=0:
19            mdc_err+=1
20
21        if bayes(i)>=0:
22            bayes_err +=1
23
24    avg_error_bayes+=bayes_err/2000
25    avg_error_mdc+=mdc_err/2000
26
27 print('avg MDC error: ',avg_error_mdc/20)
28 print('avg bayes error: ',avg_error_bayes/20)
29
30 #avg MDC error:  0.015175000000000008
31 #avg bayes error:  0.012925000000000002
```

Listing 5: 2e



3 Automatic Saffron Cleaning Machine

3)

a) $g_i(x) = \frac{1}{2} (x - \mu_i)^T (x - \mu_i) + \log(p(\omega_i))$ \rightarrow in case we have $\Sigma_i = I$

$\Rightarrow g_i(x) = g_i(n) \Rightarrow (x - \mu_i)^T (x - \mu_i) = (x - \mu_j)^T (x - \mu_j)$ and normal distributions the bayes classifier discriminant functions become

Class 1: petal, Class 2: stamen, Class 3: stigma $\left\{ \begin{array}{l} \mu_1 = 48.748, \mu_2 = 56.86 \\ \mu_3 = 46.084 \end{array} \right.$

feature set 1: $\arg \min_i (x - \mu_i)^2 \Rightarrow \arg \min_i ((x - 48.748)^2, (x - 56.86)^2, (x - 46.084)^2)$

$\Rightarrow \arg \min ((48.748^2 + \mu_1^2), (56.86^2 + \mu_2^2), (46.084^2 + \mu_3^2))$

$\Rightarrow \begin{cases} \omega_1 & x < -\frac{32}{25} \\ \omega_2 & -\frac{32}{25} < x < 46 \\ \omega_3 & 46 < x \end{cases}$

Feature set 2: $\left\{ \begin{array}{l} \mu_1 = [11.6] \\ \mu_2 = [2.6] \\ \mu_3 = [22] \end{array} \right. \Rightarrow \arg \min_i (x - \mu_i)^T (x - \mu_i)$

$\Rightarrow \arg \min_i [(x_1 - \mu_{11})^2 + (x_2 - \mu_{12})^2] \Rightarrow \arg \min_i [x_1^2 + \mu_{11}^2 - 2x_1\mu_{11} + x_2^2 + \mu_{12}^2 - 2x_2\mu_{12}]$

$\Rightarrow \arg \min_i [(134.56 - 2x_1 \cdot 11.6), (47.72 - 5.2x_1 - 12.8x_2), (485 - 44x_1 - 2x_2)]$

Feature set 3: $\left\{ \begin{array}{l} \mu_1 = [742] \\ \mu_2 = [0.622] \\ \mu_3 = [3.696] \end{array} \right. \arg \min_i (x - \mu_i)^T (x - \mu_i) \Rightarrow \arg \min_i [(x_1 - \mu_{11})^2 + (x_2 - \mu_{12})^2 + (x_3 - \mu_{13})^2]$

$\Rightarrow \arg \min_i [\mu_{11}^2 + \mu_{12}^2 + \mu_{13}^2 + 2(x_1\mu_{11} + x_2\mu_{12} + x_3\mu_{13})]$

Figure 6: a



3-b)	feature set 1	feature set 2	feature set 3	true label
1:	$x > 46 \rightarrow \omega_3$	$\omega_2 \checkmark$	ω_1	ω_2
2:	$\omega_3 \checkmark$	ω_2	ω_2	ω_3
3:	$\omega_1 \checkmark$	ω_2	$\omega_1 \checkmark$	ω_1
4:	$\omega_3 \checkmark$	ω_2	ω_2	ω_3
5:	$\omega_1 \checkmark$	$\omega_2 \checkmark$	$\omega_2 \checkmark$	ω_3
6:	ω_3	$\omega_2 \checkmark$	$\omega_1 \checkmark$	ω_1
7:	$\omega_1 \checkmark$	ω_2	$\omega_2 \checkmark$	ω_2
8:	ω_3	$\omega_2 \checkmark$	$\omega_1 \checkmark$	ω_1

Figure 7: b



c) classifier 1:

true/predict	ω_1	ω_2	ω_3
ω_1	3	0	0
ω_2	0	0	3
ω_3	0	0	2

classifier 2:

		Predict		
		ω_1	ω_2	ω_3
true	ω_1	0	3	0
	ω_2	0	3	0
	ω_3	0	2	0

classifier 3

		Predict		
		ω_1	ω_2	ω_3
true	ω_1	3	0	0
	ω_2	1	2	0
	ω_3	0	2	0

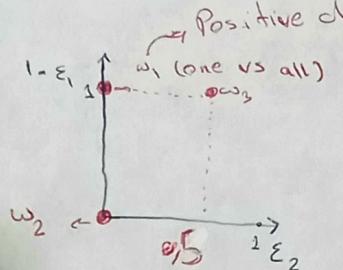
Figure 8: c

d) \Rightarrow classifier 1 \rightarrow error: $\frac{3}{8}$

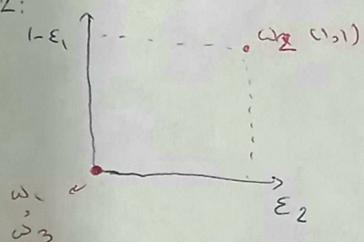
classifier 2 \rightarrow error: $\frac{5}{8}$

classifier 3: \rightarrow error: $\frac{3}{8}$

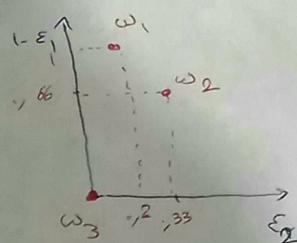
c) classifier 1:



classifier 2:



classifier 3:



f) we know that in bayes classifier if we have $\{\Sigma_i\}$, I and normal distributions and equal priors the bayes discriminant function is equivalent to MDC classifier so we will have same results.

Figure 9: d-f



4 Dealing with Prediction Error in Bayes Decision Rule

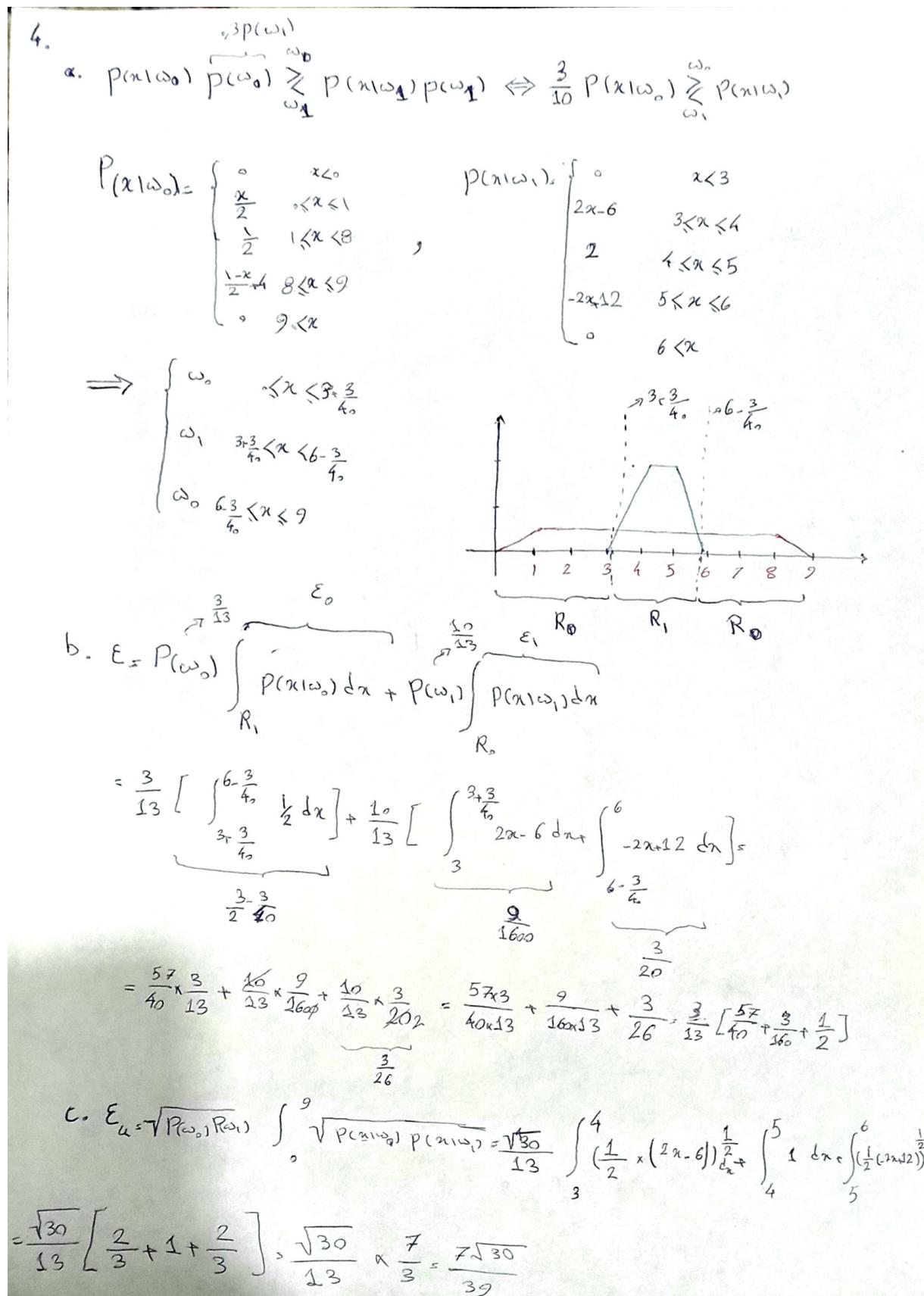


Figure 10: a-c

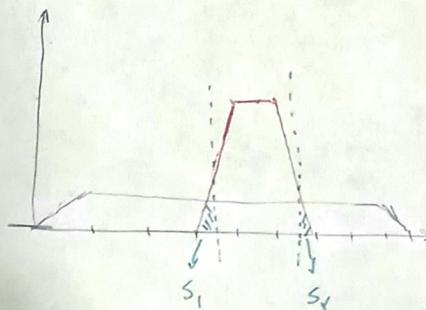
d. $E_{\omega_1} \left(\frac{3}{13} \right) \underbrace{\left(\frac{10}{13} \right)^{1-s}}_A \int_{-\infty}^{\infty} p(x|\omega_1)^s p(x|\omega_2)^{1-s} = A \left[\int_3^4 \left(\frac{1}{2} \right)^s (2x-6)^{1-s} + \int_4^5 \left(\frac{1}{2} \right)^s (2)^{1-s} \right. \\ \left. + \int_5^6 \left(\frac{1}{2} \right)^s (-2x+12)^{1-s} \right]$

 $= \left(\frac{3}{13} \right)^s \left(\frac{10}{13} \right)^{1-s} \left[\int_3^4 \left(2x-6 \right)^{1-s} + 2^{1-2s} + \int_5^6 \left(-2x+12 \right)^{1-s} \right]$

e. $E_{1s} \int_{R_0} P(x|\omega_1) d_{n_s} \approx \int_3^{3+\frac{1}{2\sqrt{10}}} 2x-6 dx + \int_{6-\frac{1}{2\sqrt{10}}}^6 -2x+12 dx \Rightarrow \int_3^6 2x-6 dx = \frac{1}{40}$

 $\Rightarrow z^2 - 6z - 9 + 18 = \frac{1}{40} \Rightarrow z \in 3 \pm \frac{1}{2\sqrt{10}} \Rightarrow R_{1s} [3 \pm \frac{1}{2\sqrt{10}}, 6 - \frac{1}{2\sqrt{10}}] \cup R_{0s} [6, 9] - R_{1s}$
 $E_p P(\omega_0) \int_{R_1} p(x|\omega_0) d_n + P(\omega_1) \times \frac{5}{100} = \frac{1}{26} + \frac{3}{13} \int_{3 - \frac{1}{2\sqrt{10}}}^{6 - \frac{1}{2\sqrt{10}}} \frac{1}{2} d_n = \frac{1}{26} + \frac{3}{13} \left[\frac{3}{2} - \frac{\sqrt{10}}{20} \right]$
 $= \frac{1}{26} + \frac{9}{26} - \frac{3\sqrt{10}}{260} = \frac{1}{26} \left(10 - \frac{3\sqrt{10}}{10} \right)$

We know that for E_p to be equal to $\frac{5}{100}$ our decision region would be like this:



and $s_1 + s_2 = \frac{5}{100}$ so to minimize E_p , s_1 and s_2 must be equal so $s_1 = s_2 = \frac{1}{40}$

Figure 11: d-e



5 Separating Messi from Football Field

5.1 a

```
1 def get_dct_feature(block):
2     """make zigzag dct feature for a 8*8 normal block
3
4     Args:
5         block (numpy array 8*8)
6
7     """
8     zigzag = np.array([[0, 1, 5, 6, 14, 15, 27, 28],
9                         [2, 4, 7, 13, 16, 26, 29, 42],
10                        [3, 8, 12, 17, 25, 30, 41, 43],
11                        [9, 11, 18, 24, 31, 40, 44, 53],
12                        [10, 19, 23, 32, 39, 45, 52, 54],
13                        [20, 22, 33, 38, 46, 51, 55, 60],
14                        [21, 34, 37, 47, 50, 56, 59, 61],
15                        [35, 36, 48, 49, 57, 58, 62, 63]]))
16
17     zigzag_vec = zigzag.reshape(64,)
18     dct = cv2.dct(block) # DCT transformer
19     dct_vec = np.abs(dct.reshape(64,))
20     sec_larg_index = np.argpartition(dct_vec,-2)[-2] #finding the second largest abs
21     value of DCTs
22     return zigzag_vec[sec_larg_index]
23
24 #loading training images
25 img = cv2.imread('leo1.png',0)
26 mask_img = cv2.imread('leo1_mask.png',0)
27 #normalize image for dct
28 noraml_img = img/255
29 features = np.zeros((img.shape[0]//8 , img.shape[1]//8))
30
31 #calculating feature for each block
32 for i in range(img.shape[0]//8):
33     for j in range(img.shape[1]//8):
34         block = noraml_img[8*i:8*(i+1),8*j:8*(j+1)]
35         features[i,j] = get_dct_feature(block)
36
37 #calculating priors
38 prior_messi = np.sum(mask_img/255)/(mask_img.shape[0]*mask_img.shape[1])
39 prior_field = 1 - prior_messi
40
41 print('prior messi = ', prior_messi)
42 print('prior field = ', prior_field)
43 #prior messi = 0.35684313725490197
44 #prior field = 0.6431568627450981
```

Listing 6: 5a

5.2 b

```
1 features_vec = features.reshape(img.shape[0]//8 * img.shape[1]//8,)
2 mask_features = np.zeros((img.shape[0]//8 , img.shape[1]//8))
3
4 #deciding if each block in training image belongs to feild or messi
5 for i in range(mask_img.shape[0]//8):
6     for j in range(mask_img.shape[1]//8):
7         block = mask_img[8*i:8*(i+1),8*j:8*(j+1)]/255
8         if np.sum(block) > 32:
```



```
9         mask_features[i,j] = 1
10        else:
11            mask_features[i,j] = 0
12
13 mask_features_vec = mask_features.reshape(mask_img.shape[0]//8 * mask_img.shape
14 [1]//8,)
15 post_messi = []
16 post_field = []
17 #saving each feature to its class
18 for i in range(mask_img.shape[0]//8 * mask_img.shape[1]//8):
19     if mask_features_vec[i] == 1:
20         post_messi.append(features_vec[i])
21     else:
22         post_field.append(features_vec[i])
23
24 #plotting histograms
25 plt.hist(post_messi)
26 plt.savefig('messi likelihood.png')
27 plt.clf()
28 plt.hist(post_field)
29 plt.savefig('field likelihood.png')
30
31 #calculating likelihoods
32 counter_messi = collections.Counter(post_messi)
33
34 counter_field = collections.Counter(post_field)
35
36 for i in counter_field.keys():
37     counter_field[i]/= len(post_field)
38
39 for i in counter_messi.keys():
40     counter_messi[i]/= len(post_messi)
41
42 for i in range(64):
43     if i not in counter_field.keys():
44         counter_field[i] = 0
45
46 if i not in counter_messi.keys():
47     counter_messi[i] = 0
```

Listing 7: 5b

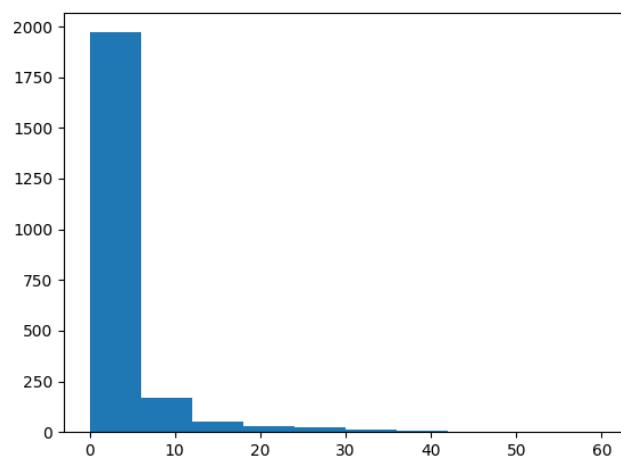


Figure 12: messi likelihood histogram

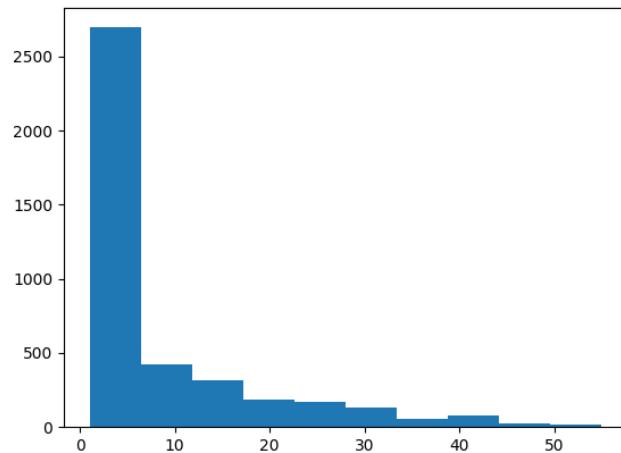


Figure 13: field likelihood histogram

5.3 c

```

1 #loading test image
2 test_img = cv2.imread('leo2.png',0)
3 normal_test = test_img / 255
4
5 test_features = np.zeros((test_img.shape[0]//8 , test_img.shape[1]//8))
6 # calculating the X feature for each block
7 for i in range(test_img.shape[0]//8):
8     for j in range(test_img.shape[1]//8):
9         block = normal_test[8*i:8*(i+1),8*j:8*(j+1)]
10        test_features[i,j] =get_dct_feature(block)

```

Listing 8: 5c

5.4 d

```

1 #loading test image
2 test_img = cv2.imread('leo2.png',0)
3 normal_test = test_img / 255
4
5 test_features = np.zeros((test_img.shape[0]//8 , test_img.shape[1]//8))
6 # calculating the X feature for each block
7 for i in range(test_img.shape[0]//8):
8     for j in range(test_img.shape[1]//8):
9         block = normal_test[8*i:8*(i+1),8*j:8*(j+1)]
10        test_features[i,j] =get_dct_feature(block)
11        x = test_features[i,j]
12        if counter_messi[x]*prior_messi>counter_field[x]*prior_field:
13            #test_img[8*i:8*(i+1),8*j:8*(j+1)]=np.ones((8,8))*255
14            pass
15        else:
16            test_img [8*i:8*(i+1),8*j:8*(j+1)]=np.zeros((8,8))
17
18 cv2.imwrite('result.png',test_img)

```

Listing 9: 5d



Figure 14: result

5.5 e

```
1 #loading test image
2 test_img = cv2.imread('leo2.png',0)
3 normal_test = test_img / 255
4 test_mask = cv2.imread('leo2_mask.png',0)
5
6 test_features = np.zeros((test_img.shape[0]//8 , test_img.shape[1]//8))
7 error = 0
8 # calculating the X feature for each block
9 for i in range(test_img.shape[0]//8):
10    for j in range(test_img.shape[1]//8):
```



```

12     block = normal_test[8*i:8*(i+1), 8*j:8*(j+1)]
13     test_features[i,j] = get_dct_feature(block)
14     x = test_features[i,j]
15     if counter_messi[x]*prior_messi > counter_field[x]*prior_field:
16         if np.sum(test_mask[8*i:8*(i+1), 8*j:8*(j+1)]/255) < 33:
17             error+=1
18         else:
19             if np.sum(test_mask[8*i:8*(i+1), 8*j:8*(j+1)]/255) > 32:
20                 error+=1
21
22 error = error / ((test_mask.shape[0]//8) * (test_mask.shape[1]// 8))
23
24 print("prob of error: ", error)
25
26 #prob of error:  0.2912941176470588

```

Listing 10: 5e

6 Maximum Likelihood Approach for Parameter Estimation

6.

$$\alpha) L(\theta), f_{(1,2,1,3,4,2,3,1,1,3)}(\theta) = f_{(1|\theta)}^4 f_{(2|\theta)}^2 f_{(3|\theta)}^3 f_{(4|\theta)} = \left(\frac{3\theta}{5}\right)^4 \left(\frac{2(1-\theta)}{5}\right)^2 \left(\frac{2\theta}{5}\right)^3 \left(\frac{3(1-\theta)}{5}\right)$$

$$\begin{aligned} b) \log L(\theta) &= 4 \lg \frac{3\theta}{5} + 2 \lg \frac{2(1-\theta)}{5} + 3 \lg \frac{2\theta}{5} + \lg \frac{3(1-\theta)}{5} = 4 \lg \frac{3}{5} + 4 \lg \theta + 2 \lg \frac{2}{5} + 2 \lg (1-\theta) + \\ &\quad 3 \lg \frac{2}{5} + 3 \lg \theta + \lg \frac{3}{5} + \lg (1-\theta) \\ \Rightarrow \log L(\theta) &= 5 \lg \frac{3}{5} + 5 \lg \frac{2}{5} + 7 \lg \theta + 3 \lg (1-\theta) \end{aligned}$$

$$c) \frac{d}{d\theta} \ln L(\theta) = \frac{7}{\theta} + \frac{-3}{1-\theta} = 0 \Rightarrow \frac{7}{\theta} = \frac{3}{1-\theta} \Rightarrow 3A = 7 - 7\theta \Rightarrow 10\theta = 7 \Rightarrow \hat{\theta} = \frac{7}{10}$$

$$d) L(\theta) = \prod_{i=1}^n \frac{1}{(m-1)!} \times \theta^{-m} \times x_i^{m-1} \times e^{\frac{-x_i}{\theta}} = \frac{1}{(m-1)!} \times \theta^{-nm} \times \prod_{i=1}^n x_i^{m-1} \times e^{\frac{-\sum x_i}{\theta}}$$

$$\Rightarrow \ln L(\theta) = -n \ln(m-1)! - mn \ln \theta + (m-1)(\sum \ln x_i) + \frac{-\sum x_i}{\theta} \Rightarrow \frac{d}{d\theta} \ln L(\theta) = \frac{-mn}{\theta} + \frac{\sum x_i}{\theta^2}$$

Figure 15: a-d



6)

$$d) \frac{d}{d\theta} \ln L(\theta) = -\frac{mn}{\theta} + \frac{\sum y_i}{\theta^2} = 0 \Rightarrow \frac{\sum y_i}{\theta^2} = \frac{mn}{\theta} \Rightarrow \hat{\theta} = \frac{\sum y_i}{mn}$$

$$e) L(\theta) = \prod_{i=1}^n \frac{e^{\theta y_i}}{y_i!} = e^{n\theta} \times \prod_{i=1}^n \frac{\theta^{y_i}}{y_i!} \Rightarrow \ln L(\theta) = -n\theta + \sum_{i=1}^n (y_i \ln \theta - \ln(y_i!))$$

$$= -n\theta + \sum_{i=1}^n y_i \ln \theta - \sum_{i=1}^n \ln(y_i!) \Rightarrow \frac{d}{d\theta} \ln L(\theta) = -n + \sum_{i=1}^n \frac{y_i}{\theta} = -n + \frac{1}{\theta} \sum y_i = 0$$

$$\Rightarrow \frac{\sum y_i}{\theta} = n \Rightarrow \hat{\theta} = \frac{\sum y_i}{n}$$

$$f) E(\hat{\theta}_{ML}) = E\left[\frac{\sum y_i}{n}\right] = \frac{1}{n} \sum_{i=1}^n E[y_i] = \frac{1}{n} \times n\mu = \mu$$

$$\text{Var}(\hat{\theta}_{ML}) = E[\hat{\theta}^2] - E[\hat{\theta}]^2 = E\left[\frac{\sum_{i=1}^n y_i y_j}{n^2}\right] - \mu^2,$$

$$E\left[\frac{\sum_{i=1}^n y_i y_j}{n^2}\right] = \frac{1}{n^2} \sum_{j=1}^n \sum_{i=1}^n E[y_i y_j] = \frac{1}{n^2} \left[(n-1)\mu^2 + n(\mu^2 + \sigma^2) \right]$$

$E[y_i y_j], i \neq j, \text{ independent}$

$$\Rightarrow \text{Var}(\hat{\theta}_{ML}) = \mu^2 - \frac{1}{n}\mu^2 + \frac{n-1}{n}\mu^2 + \frac{\sigma^2}{n} = \frac{\sigma^2}{n}$$

$$g) \hat{\theta}_{MAP}(y) = \arg \max_{\theta} f(\theta|y) p(\theta) \Rightarrow \arg \max_{\theta} \prod_{i=1}^n \frac{e^{\theta y_i}}{y_i!} \times \underbrace{\frac{\lambda^\alpha \theta^{\alpha-1} e^{-\lambda \theta}}{\Gamma(\alpha)}}_A$$

$$\ln A = \sum_{i=1}^n \left(\ln \frac{e^{\theta y_i}}{y_i!} + \ln \frac{\lambda^\alpha \theta^{\alpha-1} e^{-\lambda \theta}}{\Gamma(\alpha)} \right) = \sum_{i=1}^n \left[-\theta + y_i \ln \theta - \ln y_i! + \alpha \ln \lambda + (\alpha-1) \ln \theta - \lambda \theta - \ln(\Gamma(\alpha)) \right]$$

$$\frac{d}{d\theta} \ln A = \sum_{i=1}^n \left[-1 + \frac{y_i}{\theta} + \frac{\alpha-1}{\theta} - \lambda \right] = -n - n\lambda + \frac{n(\alpha-1)}{\theta} + \frac{1}{\theta} \sum_{i=1}^n y_i = 0 \Rightarrow \frac{1}{\theta} (n(\alpha-1) + \sum y_i) = n + n\lambda$$

$$\Rightarrow \hat{\theta}_{MAP} = \frac{n(\alpha-1) + \sum y_i}{n + n\lambda} = \frac{\alpha-1 + \frac{\sum y_i}{n}}{1 + \lambda}$$

$$E[\hat{\theta}_{MAP}] = E\left[\frac{\alpha-1 + \frac{\sum y_i}{n}}{1 + \lambda}\right] = \frac{\alpha-1}{1+\lambda} + \frac{1}{n(1+\lambda)} \left(\sum_{i=1}^n E[y_i] \right) = \frac{\alpha-1}{1+\lambda} + \frac{\mu}{1+\lambda} = \frac{\mu + \alpha - 1}{1+\lambda}$$

$$\text{if } \alpha = 1 \Rightarrow \frac{\mu}{1+\lambda}$$

Figure 16: d-g



6)

$$g) \hat{\theta}_{ML} = \frac{\sum y_i}{n}, \hat{\theta}_{MAP} = \frac{\alpha-1}{1+\lambda} + \frac{\sum y_i}{n} = \frac{\alpha-1}{1+\lambda} + \frac{\hat{\theta}_{ML}}{1+\lambda} \text{ if } \alpha=1 \Rightarrow \hat{\theta}_{MAP} = \frac{\hat{\theta}_{ML}}{1+\lambda} \Rightarrow \lambda > 0$$

$$E[\hat{\theta}_{ML}] = \mu, E[\hat{\theta}_{MAP}] = \frac{\alpha-1}{1+\lambda} + \frac{\mu}{1+\lambda} \text{ if } \alpha=1 \Rightarrow E[\hat{\theta}_{MAP}] = \frac{\mu}{1+\lambda} = \frac{E[\hat{\theta}_{ML}]}{1+\lambda}$$

$$h) \hat{\theta} = \frac{\sum y_i}{n} \Rightarrow E(\hat{\theta}) = \frac{1}{n} \sum_{i=1}^n E[y_i] = \mu$$

$$\text{Var}(\hat{\theta}) = E[\hat{\theta}^2] - E[\hat{\theta}]^2$$

$$\hookrightarrow E\left[\sum_{j=1}^n \sum_{i=1}^n y_i y_j\right] = (n-1)\mu^2 + n(\mu^2 + \sigma^2)$$

$$\Rightarrow \text{Var}(\hat{\theta}) = (n-1)\mu^2 + n(\mu^2 + \sigma^2) = (n-1)\mu^2 + n\sigma^2$$

i) Let T be any unbiased estimator of θ based on data y_1, y_2, \dots, y_n (i.i.d)

$$\text{then } \text{Var}[T] \geq \frac{1}{I(\theta)} \text{ where } I(\theta) = -E\left[\frac{\partial^2}{\partial \theta^2} \ln L(y|\theta)\right]$$

$$\Rightarrow I(\theta) = -E\left[\frac{\partial^2}{\partial \theta^2} \left(-n\theta + \sum y_i \ln \theta - \sum \ln y_i \right) \right] = -E\left[\frac{\partial}{\partial \theta} \left[-n + \frac{\sum y_i}{\theta} \right]\right]$$

$$= -E\left[-\frac{\sum y_i}{\theta^2}\right] = \frac{1}{\theta^2} \sum_{i=1}^n E[y_i] = \frac{n\mu}{\theta^2} \Rightarrow \text{Var}[T] \geq \frac{1}{\frac{n\mu}{\theta^2}}.$$

$$\text{Var}(\hat{\theta}_{ML}) = \frac{\sigma^2}{n} \geq \frac{\theta^2}{n\mu} \quad \text{our distribution is poisson distribution}$$

and in this distribution $\theta = \mu = \sigma^2$

$$\Rightarrow \frac{\sigma^2}{n} = \frac{\theta}{n} = \frac{\theta^2}{n\theta} = \frac{\theta^2}{n\mu} \Rightarrow \hat{\theta}_{ML} \text{ is efficient because it's unbiased and its var is minimum}$$

Figure 17: g-i



7 Here's Why Hitler Hated MLE So Bad

7.

a) num of all possible outcomes $\rightarrow \binom{n}{k}$

num of cases when highest serial number is $m \rightarrow \binom{m-1}{k-1}$ because when the largest serial number is m we should pick other $k-1$ serial numbers from 1 to $m-1$

$$\text{so } P(m|n, k) = \frac{\binom{m-1}{k-1}}{\binom{n}{k}}$$

b) $P(n|m, k) = \frac{P(m|n, k) P(n|k)}{P(m|k)}$ ①

$P(n|k) = \frac{1}{\Omega - k} \rightarrow$ because $k \leq n < \Omega$ and n is uniformly distributed

$$P(m|k) = P(m|k) \times 1 = P(m|k) \times \sum_{n=0}^{\Omega-1} P(n|m, k) \stackrel{①}{=} P(m|k) \sum_{n=m}^{\Omega-1} \frac{P(m|n, k) P(n|k)}{P(m|k)}$$

$$= \sum_{n=m}^{\Omega-1} P(m|n, k) P(n|k) = \frac{\sum_{n=m}^{\Omega-1} P(m|n, k)}{\frac{1}{\Omega - k}}$$

$$\Rightarrow P(n|m, k) = \frac{P(m|n, k) \times \frac{1}{\Omega - k}}{\sum_{n=m}^{\Omega-1} P(m|n, k)} = \frac{\binom{m-1}{k-1} \binom{n}{k}^{-1}}{\sum_{n=m}^{\Omega-1} \binom{m-1}{k-1} \binom{n}{k}^{-1}} = \frac{\binom{n}{k}^{-1}}{\frac{k}{k-1} \left[\frac{1}{(n-1)} - \frac{1}{(\Omega-1)} \right]}$$

$\sum_{n=m}^{\Omega-1} \frac{1}{\binom{n}{k}}$ using Hint

$$\text{if } \Omega \rightarrow \infty \text{ then } \binom{\Omega-1}{k-1} \rightarrow \infty \Rightarrow P(n|m, k) = \frac{\binom{n}{k}^{-1}}{\frac{k}{\binom{m-1}{k-1}}} = \frac{(k-1)(m-1)}{k \binom{n}{k}}$$

Figure 18: a-b



C) $\mu_{\sum p(n)} = \sum_{n=-\infty}^{+\infty} n P(n|m, k) = \sum_{n=m}^{\infty} n \left[\frac{(k-1)(m-1)}{k \binom{n}{k}} \right] = \left(\frac{k-1}{k} \right) \binom{m-1}{k-1} \sum_{n=m}^{\infty} n \binom{n}{k}^{-1}$

$$\binom{n}{k} = \frac{n!}{\frac{n!}{k!(n-k)!}} = \frac{k!(n-k)!}{(n-1)!} \Rightarrow \mu_s A \underbrace{\left(\frac{k-1}{k} \right) \binom{m-1}{k-1} k!}_{A} \sum_{n=m}^{\infty} \frac{(n-k)!}{(n-1)!}$$

$$\Rightarrow \mu_s A \sum_{n=m}^{\infty} \frac{(n-k)!}{(n-1)!} \stackrel{j=n-1}{=} \mu_s A \sum_{j=m-1}^{\infty} \frac{(j-(k-1))!}{j!} = A \frac{1}{k-x} \left[\frac{\binom{m-k+1}{m-x}}{\binom{m-x}{m-x}} - \frac{\binom{m-k+1}{m-x+1}}{\binom{m-x+1}{m-x+1}} \right]$$

Using Hint ↑

if $k \geq 3$ & $\infty \rightarrow \infty \Rightarrow \frac{(n-k)!}{(n-x)!} \rightarrow 0 \Rightarrow \mu_s A \frac{k-1}{k} \binom{m-1}{k-1} k! \times \frac{1}{k-x} \times \frac{(m-k)!}{(m-x)!}$

$$= \frac{k-1}{x} \times \frac{\cancel{(m-1)!}}{\cancel{(k-1)!} \cancel{(m-k)!}} \times k! \times \frac{1}{k-x} \times \frac{\cancel{(m-k)!}}{\cancel{(m-x)!}} = \frac{k-1}{k-x} \binom{m-1}{k-1}$$

d) if $k=2$ then $k-2=0$ and $\frac{k-1}{k-x}$ is undefined and if $k<2$ then $\frac{(n-k)!}{(n-x)!}$ is undefined so the minimum k is 3

Figure 19: c-d

8 Further Study: When Will Civilization End?

9 Some Explanatory Questions

9.1 a

when the prior probabilities are equal and two classes have normal distributions with the same covariance matrix. and the covariance matrix has to be equal to the identity matrix.

9.2 b

in MDC when we want to choose class representative for example mean of each class we have to calculate it if we dont have it and its training phase but if we have class means we dont need to calculate. similarly in minimum error classifier if we dont have distributions or parameters of distribution we have to estimate them using data and this could be training phase but if we have distributions we dont need to this phase.

9.3 c

the Bayes error is the best error that we can achieve because we have the exact distributions and probabilities and we have an infinitely large dataset and we know in this situation Bayes error is optimum.

9.4 d

yes we can use one vs all method or we can plot ROC curve for each pair of classes.



9.5 e

yes we can assume a distribution for target and use the bayes rule for calculating targets

9.6 f

yes it is. because we have a decision boundary in bayes decision rule and its fixed.

9.7 g

if in MAP we have uniform prior then the MAP and MLE are the same. in MAP we use more information for estimation so in general its better than MLE but if we dont have a correct prior its better to use MLE.

9.8 h

in this method we add $-\lambda P(\theta)$ to the function that we want to maximize. when we have a small amount of data its better to apply penalized MLE