IITM-CS5691 : Pattern Recognition and Machine Learning
Assignment 1

Release Date: March 9, 2021
Due Date : March 19, 23:59

Roll No: EE19B132
Name: Sagnik Ghosh
Collaborators (if any): ME19B012
References (if any): CMB, DHS

- Use LaTeX to write-up your solutions (in the solution blocks of the source LaTeX file of this assignment), and submit the resulting single pdf file at GradeScope by the due date. (Note: **No late submissions** will be allowed, other than one-day late submission with 10% penalty or four-day late submission with 30% penalty! Within GradeScope, indicate the page number where your solution to each question starts, else we won't be able to grade it! You can join GradeScope using course entry code **5VDNKV**).
- For the programming question, please submit your code (rollno.ipynb file and rollno.py file in rollno.zip) directly in moodle, but provide your results/answers in the pdf file you upload to GradeScope.
- Collaboration is encouraged, but all write-ups must be done individually and independently, and mention your collaborator(s) if any. Same rules apply for codes written for any programming assignments (i.e., write your own code; we will run plagiarism checks on codes).
- If you have referred a book or any other online material for obtaining a solution, please cite the source. Again don't copy the source *as is* - you may use the source to understand the solution, but write-up the solution in your own words.
- Points will be awarded based on how clear, concise and rigorous your solutions are, and how correct your code is. Overall points for this assignment would be **min**(your score including bonus points scored, 50).

1. (10 points) [GETTING YOUR BASICS RIGHT!]

   (a) (1 point) You have a jar of $1,000$ coins. 999 are fair coins, and the remaining coin will always land heads. You take a single coin out of the jar and flip it 10 times in a row, all of which land heads. What is the probability your next toss with the same coin will land heads? Explain your answer. How would you call this probability in Bayesian jargon?

   Solution: We have 1000 coins out of which 999 are fair and 1 coin always lands as a head.
   Let A be the event that a fair coin is chosen.
   $P(A) = \frac{999}{1000}$
   Let B be the event that all 10 flips land as heads.

   $$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A)+P(B|A^C)P(A^C)}$$
   $$P(A|B) = \frac{0.5^{10}\frac{999}{1000}}{0.5^{10}\frac{999}{1000}+1^{10}\frac{1}{1000}}$$
   $$P(A|B) = \frac{999}{2023}$$

$$P(A^C|B) = 1 - P(A|B)$$
$$P(A^C|B) = \frac{1024}{2023}$$

Let C be the event that the next flip lands as a head.

$$P(C|B) = P(C|A,B)P(A|B) + P(C|A^C,B)P(A^C|B)$$
$$P(C|B) = 0.5 * \frac{999}{2023} + 1 * \frac{1024}{2023}$$
$$P(C|B) = \frac{3047}{4046} = 0.7531$$

This is the required probability.

This probability can be called the posterior probability in Bayesian jargon.

(b) (3 points) Consider the i.i.d data $\mathbf{X} = \{x_i\}_{i=1}^n$, such that each $x_i \sim \mathcal{N}(\mu, \sigma^2)$. We have seen ML estimates of μ, σ^2 in class by setting the gradient to zero. How can you argue that the stationary points so obtained are indeed global maxima of the likelihood function? Next, derive the bias of the MLE of μ, σ^2.

Solution: Let us consider the likelihood L,

$$L = \prod_{i=1}^n \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{1}{2\sigma^2}(x_i - \mu)^2)$$

As log() is a concave function, if we can prove that the log-likelihood is concave, i.e., the log-likelihood has only global optima, it implies that the likelihood also has only global optima.

$$ln(L) = nln(\frac{1}{\sqrt{2\pi}\sigma}) - \frac{1}{2\sigma^2}\sum_{i=1}^n (x_i - \mu)^2$$

Let us make a substitution $\alpha = \frac{1}{\sigma}$,

$$ln(L) = nln(\frac{\alpha}{\sqrt{2\pi}}) - \frac{\alpha^2}{2}\sum_{i=1}^n (x_i - \mu)^2$$

Let us first take the double derivative of the log-likelihood with respect to μ,

$$\frac{\partial^2 ln(L)}{\partial \mu^2} = -n\alpha^2 < 0$$

As the double derivative is negative, the ML estimate of μ is the global maxima.

Let us now take the double derivative of the log-likelihood with respect to α,

$$\frac{\partial^2 ln(L)}{\partial \alpha^2} = -\frac{1}{\alpha^2} - \sum_{i=1}^n (x_i - \mu)^2 < 0$$

As the double derivative is negative, the ML estimate of α is the global minima and hence the ML estimate of σ is the global maxima.

Let us derive the bias of μ_{ML}.

$$E[\mu_{ML}] = E[\frac{1}{n}\sum_{i=1}^n x_i]$$
$$E[\mu_{ML}] = \frac{1}{n} n E[x] = \mu$$

Hence ML estimate of μ is unbiased.

(c) (2 points) Consider a hyperplane $\mathbb{H}$ in $\mathbb{R}^d$ passing through zero. Prove that $\mathbb{H}$ is a subspace of $\mathbb{R}^d$ and is of dimension $d - 1$.

Solution: A hyperplane $\mathbb{H}$ can be defined as follows,

$$\mathbb{H} = \{x = [x_1 x_2 x_3 ... x_d]^T | a_1 x_1 + a_2 x_2 + ... + a_d x_d = 0\}$$

This can be written as,

$$\mathbb{H} = \{x = [x_1 x_2 x_3 ... x_d]^T | Ax = 0\}$$

where $A = [a_1 a_2 ... a_n]$

As $Ax = 0$, x is in the null space of A. As $\mathbb{H}$ is the set of all such x, $\mathbb{H}$ is the null space of A.

Since every nullspace of matrices in R^d is a subspace of R^d, $\mathbb{H}$ is also a subspace of R^d.

Using the rank-nullity theorem,

rank(A) + nullity(A) = d

As rank(A) = 1, nullity(A) = d-1.

But, nullity(A) = number of dimensions in null space of A. $\mathbb{H}$ is the nullspace of A. Hence, $\mathbb{H}$ is of dimension d-1.

(d) (2 points) We saw a mixture of two 1D Gaussians ($N(\mu_1, \sigma_1^2)$ and $N(\mu_2, \sigma_2^2)$) in class with parameters π_1, π_2 for the mixing proportions. Is the likelihood of this model convex or not convex? Give proof to support your view.

Solution: If we assume that the maxima of the likelihood is obtained when $\{\pi_1, \mu_1, \sigma_1, \pi_2, \mu_2, \sigma_2\}$ take the values $\{a_1, a_2, a_3, a_4, a_5, a_6\}$, then the likelihood should also attain a maxima when $\{\pi_1, \mu_1, \sigma_1, \pi_2, \mu_2, \sigma_2\}$ take the values $\{a_4, a_5, a_6, a_1, a_2, a_3\}$. Because of this, there cannot be single global maxima and hence the likelihood cannot be strictly convex.

The above issue, is also known as **identifiability**.

(e) (2 points) Show that there always exists a solution for the system of equations, $A^T Ax = A^T b$, where $x \in \mathbb{R}^m$, $A \in \mathbb{R}^{n \times m}$ and $b \in \mathbb{R}^n$. Further, show that for some solution x^* of this system of equations, Ax^* is the projection of b onto the column space of A.

Solution: We are given a vector b in R^n. Let us take its projection along the column space of A. Let us call the projection b_p. Hence,

$$Ax = b_p$$

for some x.

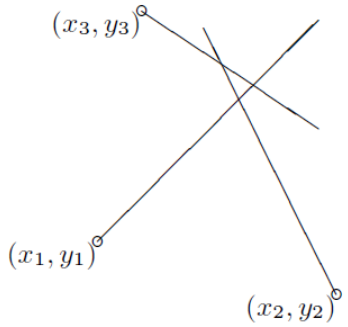
$b - b_p$ lies perpendicular to the column space of A. As the column space is orthogonal to null space of A^T, we have,

$$A^T(b - Ax) = 0$$

$$A^T Ax = A^T b$$

As we have taken arbitrary x, b, this equation always has a solution.

2. (5 points) [OF SAILORS AND BEARINGS...] A sailor infers his location (x, y) by measuring the bearings of three buoys whose locations (x_n, y_n) are given on his chart. Let the true bearings of the buoys be θ_n (measured from north as explained here). Assuming that his measurement $\tilde{\theta}_n$ of each bearing is subject to Gaussian noise of small standard deviation σ, what is his inferred location, by maximum likelihood?



The sailor's rule of thumb says that the boat's position can be taken to be the centre of the cocked hat, the triangle produced by the intersection of the three measured bearings as in the figure shown. Can you persuade him that the maximum likelihood answer is better?

Solution: We are given the locations (x_1, y_1), (x_2, y_2), (x_3, y_3) of three buoys and their corresponding measured bearings, $\tilde{\theta}_1$, $\tilde{\theta}_2$, $\tilde{\theta}_3$.

By considering the slopes of the lines joining (x_1, y_1), (x_2, y_2), (x_3, y_3) to (x, y) (true location of sailor) we can find expressions for the true bearings in terms of (x, y).

$$\frac{y-y_1}{x-x_1} = \tan(90^o - \theta_1)$$
$$\theta_1 = 90^o - \tan^{-1}(\frac{y-y_1}{x-x_1})$$

Similarly,

$$\theta_2 = 270^o + \tan^{-1}(-\frac{y-y_2}{x-x_2})$$
$$\theta_3 = 90^o + \tan^{-1}(-\frac{y-y_3}{x-x_3})$$

Please note that the above expressions for the bearings can be derived geometrically. It is also not possible to find the bearings for an arbitrarily located buoy because the function $\tan^{-1}(\tan())$ takes different forms in different quadrants.

Let us now try finding an expression for the **likelihood**.

$$p(\tilde{\theta} = \tilde{\theta}_i) = \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{1}{2\sigma^2}(\tilde{\theta}_i - \theta_i)^2)$$

Let us denote the likelihood by L.

$$L = \prod_{i=1}^{3} \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{1}{2\sigma^2}(\tilde{\theta}_i - \theta_i)^2)$$
$$ln(L) = 3ln(\frac{1}{\sqrt{2\pi}\sigma}) - \frac{1}{2\sigma^2}\sum_{i=1}^{3}(\tilde{\theta}_i - \theta_i)^2$$

By substituting the expressions for the errors in the above expression,

$$ln(L) =$$
$$3ln(\frac{1}{\sqrt{2\pi}\sigma}) - \frac{1}{2\sigma^2}((\tilde{\theta}_1 - 90^o + tan^{-1}(\frac{y-y_1}{x-x_1}))^2 + (\tilde{\theta}_2 - 270^o - tan^{-1}(-\frac{y-y_2}{x-x_2}))^2 + (\tilde{\theta}_3 - 90^o - tan^{-1}(-\frac{y-y_3}{x-x_3}))$$

We now need to maximize the above expression for the **log-likelihood** with respect to x and y.

As it is a complicated expression, it is not straightforward to maximize it.

We will instead take some values for the locations of the buoys and plot the log-likelihood function to see if it gives the right solution.

Let us take the **true** location (x, y) of the sailor as $(\frac{1}{3}, \frac{1}{3})$.

Let us take the following as the locations of the buoys.

$$(x_1, y_1) = (0, 0)$$
$$(x_2, y_2) = (1, 0)$$
$$(x_3, y_3) = (0, 1)$$

Using the above values, the actual values of the angles can be computed geometrically to be the following values,

$$\theta_1 = 45^o$$
$$\theta_2 = 270^o + tan^{-1}(0.5)$$
$$\theta_1 = 180^o - tan^{-1}(0.5)$$

In the code given below, we have taken the values of $\tilde{\theta}_i$s as Gaussian Noise corrupted values of the above true bearings.

```
theta1=45
theta2=270+np.degrees(np.arctan(1/2))
theta3=180-np.degrees(np.arctan(1/2))
sigma = 1
theta1t = np.random.normal(theta1,sigma)
theta2t = np.random.normal(theta2,sigma)
theta3t = np.random.normal(theta3,sigma)
```

Using the above values, we can plot a **contour plot** of the log-likelihood. This is done as follows,

```
import numpy as np
import matplotlib.pyplot as plt

def likelihood_func(x,y,x1,y1,theta1t,x2,y2,theta2t,x3,y3,theta3t,sigma):
    l
    1/(2*sigma*sigma))*
    np.square(theta1t-90+np.degrees(np.arctan((y-y1)/(x-x1))))-
    (1/(2*sigma*sigma))*
    np.square(theta2t-270-np.degrees(np.arctan(-(y-y2)/(x-x2))))-

    (1/(2*sigma*sigma))*
    np.square(theta3t-90-np.degrees(np.arctan(-(y-y3)/(x-x3))))
    return l

A = np.linspace(0.1, 0.9, 100)
B = np.linspace(0.1, 0.9, 100)

likelihood = np.zeros((len(A), len(B)))
x1=0
y1=0
x2=1
y2=0
x3=0
y3=1
theta1=45
theta2=270+np.degrees(np.arctan(1/2))
theta3=180-np.degrees(np.arctan(1/2))
sigma = 1
theta1t = np.random.normal(theta1,sigma)
theta2t = np.random.normal(theta2,sigma)
theta3t = np.random.normal(theta3,sigma)
for i in range(len(A)):
    for k in range(len(B)):
        likelihood[i][k]+=
        likelihood_func
        (A[i],B[k],x1,y1,theta1t,x2,y2,theta2t,x3,y3,theta3t,sigma)

(a, b) = np.meshgrid(A, B)
```

```
fig, ax = plt.subplots()
CS = ax.contour(a,b, likelihood, levels =30)
ax.clabel(CS)
ax.scatter([1/3], [1/3])
ax.title.set_text(r'Contour Plot of the Likelihood Function')
plt.savefig('sailor.png')
```

As it can be noticed, the likelihood function attains a maxima at the **true** (x, y) chosen initially, i.e., $(\frac{1}{3}, \frac{1}{3})$.

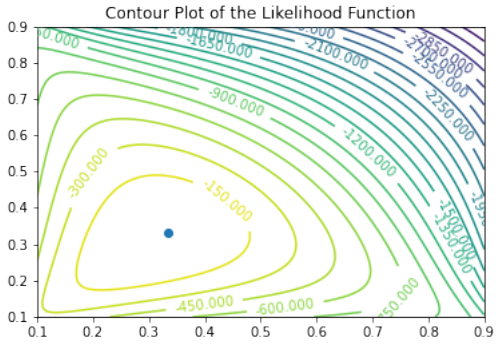


Figure 1: Plot of Likelihood

3. (5 points) [REVEREND BAYES DECIDES]

   (a) (2 points) Consider a classification problem in which the loss incurred on mis-classifying an input vector from class C_k as C_j is given by loss matrix entry L_{kj}, and for which the loss incurred in selecting the reject option is ψ. Find the decision criterion that will give minimum expected loss, and then simplify it for the case of 0-1 loss (i.e., when $L_{kj} = 1 - I_{kj}$, with I_{kj} being 1 for $k = j$ and 0 otherwise).

Solution: Consider m classes in all, $C_1, C_2, ..., C_m$. Loss incurred for classifying an input vector x into C_n is R_n.

$$R_n = \sum_{i=1}^{m} L_{in} P(C_i|x)$$

The Bayes Classifier will classify an input vector x into the class with the least loss incurred. If ψ is lesser than the loss incurred for all the classes then it chooses the reject option. This can be formulated as follows:

$$h(x) = \begin{cases} C_l & ; \quad R_l = min(R_1, R_2, .., R_m, \psi) \\ Reject & ; \quad \psi = min(R_1, R_2, .., R_m, \psi) \end{cases}$$

For the case of 0-1 loss,

$$L_{ij} = 1 \quad \forall \quad i \neq j$$
$$L_{ij} = 0 \quad \forall \quad i = j$$
$$R_n = \sum_{i=1}^{m} p(C_i|x) - p(C_n|x) = 1 - p(C_n|x)$$

Using the above formulation of $h(x)$ and substituting into that,

$$h(x) = \begin{cases} C_l & ; \quad 1\text{-}p(C_l|x) = min(1 - p(C_1|x), 1 - p(C_2|x), .., 1 - p(C_m|x), \psi) \\ Reject & ; \quad \psi = min(1 - p(C_1|x), 1 - p(C_2|x), .., 1 - p(C_m|x), \psi) \end{cases}$$

This can be simplified to,

$$h(x) = \begin{cases} C_l & ; \quad p(C_l|x) = max(p(C_1|x), p(C_2|x), .., p(C_m|x), 1 - \psi) \\ Reject & ; \quad 1\text{-}\psi = max(p(C_1|x), p(C_2|x), .., p(C_m|x), 1 - \psi) \end{cases}$$

(b) (2 points) Let L be the loss matrix defined by $L = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{bmatrix}$ where L_{ij} indicates the loss for an input x with i being the true class and j the predicted class. All the three classes are equally likely to occur. The class densities are $P(x|C_1 = 1) \sim N(-2, 1)$, $P(x|C_2 = 2) \sim N(0, 1)$ and $P(x|C_3) \sim N(2, 1)$. Find the Bayes classifier $h(x)$.

Solution: Given loss matrix $L = \begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{bmatrix}$

$$p(x|C_1) = \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x+2)^2)$$
$$p(x|C_2) = \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x-1)^2)$$
$$p(x|C_3) = \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x-2)^2)$$

As the class prior probabilities are equal, the Bayes Classifier derived using posterior probabilities is equivalent to the one derived using class conditional densities. Hence we will consider class conditional densities.

Loss incurred for classifying x into C_1 is

$$R_1 = (0)\frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x+2)^2) + (1)\frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x-1)^2) + (2)\frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x-2)^2)$$
$$R_1 = \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x-1)^2) + 2\frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x-2)^2)$$

Similarly,

$$R_2 = \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x+2)^2) + \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x-2)^2)$$
$$R_3 = 2\frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x+2)^2) + \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(x-1)^2)$$

The Bayes Classifier is as follows:

$$h(x) = \begin{cases} C_1 & ; \quad R_1 = \min(R_1, R_2, R_3) \\ C_2 & ; \quad R_2 = \min(R_1, R_2, R_3) \\ C_3 & ; \quad R_3 = \min(R_1, R_2, R_3) \end{cases}$$

where R_1, R_2, R_3 are defined above.

(c) (1 point) Consider two classes C_1 and C_2 with equal priors and with class conditional densities of a feature x given by Gaussian distributions with respective means μ_1 and μ_2, and same variance σ^2. Find equation of the decision boundary between these two classes.

Solution: We are given that

$$P(C_1) = P(C_2 = 0.5$$
$$p(x|C_1) = \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{1}{2\sigma^2}(x - \mu_1)^2)$$
$$p(x|C_2) = \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{1}{2\sigma^2}(x - \mu_2)^2)$$

As the prior probabilities are equal, we can instead work with the class conditional densities directly instead of calculating the posterior probabilities.

Hence, the Bayes Classifier would classify x into class C_1 if $p(x|C_1) > p(x|C_2)$ and vice versa.

The **decision boundary** between these two classes is defined by the equation,

$$p(x|C_1) = p(x|C_2)$$
$$\frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{1}{2\sigma^2}(x - \mu_1)^2) = \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{1}{2\sigma^2}(x - \mu_2)^2)$$
$$(x - \mu_1)^2 = (x - \mu_2)^2$$
$$2x(\mu_1 - \mu_2) + \mu_2^2 - \mu_1^2 = 0$$
$$x = \frac{\mu_1 + \mu_2}{2}$$

where we have made the assumption that $\mu_1 \neq \mu_2$.

4. (10 points) [DON'T MIX YOUR WORDS!]
Consider two documents D_1, D_2 and a background language model given by a Categorical distribution (i.e., assume $P(w|\theta)$ is known for every word w in the vocabulary V). We use the maximum likelihood method to estimate a unigram language model based on D_1, which will be denoted by θ_1 (i.e, $p(w \mid \theta_1)$ = "nos. of times word w occurred in D_1"/ $|D_1|$, where $|D_1|$ denotes the total number of words in D_1). Assume document D_2 is generated by sampling words from a two-component Categorical mixture model where one component is $p(w|\theta_1)$ and the other is $p(w|\theta)$. Let λ denote the probability that D_1 would be selected to generate a word in D_2. That makes $1 - \lambda$ the probability of selecting the background model. Let $D_2 = (w_1, w_2, ...w_k)$, where w_i is a word from the vocabulary V. Use the mixture model to fit D_2 and compute the ML estimate of λ using the EM (Expectation-Maximization) algorithm.

(a) (2 points) Given that each word w_i in document D_2 is generated independently from the mixture model, write down the log-likelihood of the whole document D_2. Is it easy to maximize this log-likelihood?

Solution: Let the set of all words in D_2 be denoted by W and let all the parameters for the mixture model defined for D_2 be denoted by θ_2.

$$p(w_i|\theta_2) = \lambda p(w_i|\theta_1) + (1-\lambda)p(w_i|\theta)$$

Hence, the likelihood can be written as,

$$p(W|\theta_2) = \prod_{i=1}^{k} p(w_i|\theta_2)$$

$$p(W|\theta_2) = \prod_{i=1}^{k} \lambda p(w_i|\theta_1) + (1-\lambda)p(w_i|\theta)$$

Hence, the log-likelihood can be written as,

$$ln(p(W|\theta_2)) = \sum_{i=1}^{k} ln(\lambda p(w_i|\theta_1) + (1-\lambda)p(w_i|\theta))$$

The above log-likelihood is not easy to maximize because we have a sum inside the logarithm. Due to this, we resort to algorithms like the EM Algorithm.

(b) (4 points) Write down the E-step and M-step updating formulas for estimating λ. Show your derivation of these formulas.

Solution: To implement the EM Algorithm for this problem, let us introduce a set of latent variables $Z = \{z_i\}_{i=1}^{k}$ for each word w_i in D_2.

z_i is defined as follows,

$$z_i = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ if } w_i \text{ is generated from the background model.}$$

$$z_i = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \text{ if } w_i \text{ is generated from the model associated with } D_1.$$

As is evident, W, Z is the complete data, whereas W is the incomplete data.

E-Step:

In the E-Step, we need to find an expression for the posterior distribution of the latent variables Z given the old parameters (In this case, we have only one unknown parameter, λ).

Note that z_{i0} is the first component of $z_i and z_{i1}$ is the second component of z_i.

$$p(z_{i0} = 1|w_i) = \frac{p(w_i|z_{i0}=1)p(z_{i0}=1)}{p(w_i|z_{i0}=1)p(z_{i0}=1)+p(w_i|z_{i1}=1)p(z_{i1}=1)}$$

$$p(z_{i0} = 1|w_i) = \frac{(1-\lambda_{old})p(w_i|\theta)}{(1-\lambda_{old})p(w_i|\theta)+\lambda_{old}p(w_i|\theta_1)}$$

Similarly,

$$p(z_{i1} = 1|w_i) = \frac{\lambda_{old}p(w_i|\theta_1)}{(1-\lambda_{old})p(w_i|\theta)+\lambda_{old}p(w_i|\theta_1)}$$

M-Step:

In the M-Step, we need to find the expectation of the complete data log-likelihood with respect to the posterior distribution of the latent variables computed using the old parameters. This expectation then needs to maximized with respect to the parameters to obtain the new parameters.

Let us first try to obtain an expression for the complete data log-likelihood.

$$p(w_i, z_i|\lambda) = [(1-\lambda)p(w_i|\theta)]^{z_{i0}}[\lambda p(w_i|\theta_1)]^{z_{i1}}$$
$$p(W, Z|\lambda) = \prod_{i=1}^{k}[(1-\lambda)p(w_i|\theta)]^{z_{i0}}[\lambda p(w_i|\theta_1)]^{z_{i1}}$$
$$ln(p(W, Z|\lambda)) = \sum_{i=1}^{k} z_{i0}[ln(1-\lambda) + ln(p(w_i|\theta))] + z_{i1}[ln(\lambda) + ln(p(w_i|\theta_1))]$$

We need to find the expectation of the above log likelihood with respect to the latent variables. Let us first find an expression for the expectation of z_{i0} and z_{i1}.

$$E[z_{i0}] = p(z_{i1} = 1|w_i)$$
$$E[z_{i0}] = \frac{(1-\lambda_{old})p(w_i|\theta)}{(1-\lambda_{old})p(w_i|\theta)+\lambda_{old}p(w_i|\theta_1)}$$

Let us call $E[z_{i0}]$ as γ_{i0} for convenience.

Similarly,

$$E[z_{i1}] = \frac{\lambda_{old}p(w_i|\theta_1)}{(1-\lambda_{old})p(w_i|\theta)+\lambda_{old}p(w_i|\theta_1)}$$

Let us call $E[z_{i1}]$ as γ_{i1} for convenience.

Now we can proceed to find an expression for the expectation of the complete data log likelihood.

$$E[ln(p(W, Z|\lambda))] = \sum_{i=1}^{k} \gamma_{i0}[ln(1-\lambda) + ln(p(w_i|\theta))] + \gamma_{i1}[ln(\lambda) + ln(p(w_i|\theta_1))]$$

Let us maximize this with respect to λ.

$$\frac{\partial E[ln(p(W,Z|\lambda))]}{\partial \lambda} = \sum_{i=1}^{k}(\frac{\gamma_{i1}}{\lambda} - \frac{\gamma_{i0}}{1-\lambda}) = 0$$
$$\lambda_{new} = \frac{1}{k}\sum_{i=1}^{k}\gamma_{i1}$$

where γ_{i1} is defined above.

(c) (4 points) In the previous parts of the question, we assume that the background language model $P(w|\theta)$ is known. How will your E-step and M-step change if you do not know the parameter θ and only know θ_1? Show your derivation.

Solution: Let us continue with the same setup as the previous part. In this case, we have to estimate λ and $\{p(w_i|\theta)\}_{i=1}^{k}$.

E-Step:

In the E-Step, we need to find an expression for the posterior distribution of the latent variables Z given the old parameters (In this case, we have λ and$\{p(w_i|\theta)\}_{i=1}^{k}$ as the unknown parameters).

$$p(z_{i0} = 1|w_i) = \frac{p(w_i|z_{i0}=1)p(z_{i0}=1)}{p(w_i|z_{i0}=1)p(z_{i0}=1)+p(w_i|z_{i1}=1)p(z_{i1}=1)}$$
$$p(z_{i0} = 1|w_i) = \frac{(1-\lambda_{old})p(w_i|\theta_{old})}{(1-\lambda_{old})p(w_i|\theta_{old})+\lambda_{old}p(w_i|\theta_1)}$$

Similarly,

$$p(z_{i1} = 1|w_i) = \frac{\lambda_{old} p(w_i|\theta_1)}{(1-\lambda_{old})p(w_i|\theta_{old}) + \lambda_{old} p(w_i|\theta_1)}$$

Note that the E-Step changes from the previous part because the above expressions are in terms of $p(w_i|\theta_{old})$ instead of $p(w_i|\theta)$ as we are updating and estimating the values of $p(w_i|\theta)$s too.

M-Step:

In the M-Step, we need to find the expectation of the complete data log-likelihood with respect to the posterior distribution of the latent variables computed using the old parameters. This expectation then needs to maximized with respect to the parameters to obtain the new parameters.

Let us first try to obtain an expression for the complete data log-likelihood.

$$p(w_i, z_i|\lambda) = [(1-\lambda)p(w_i|\theta)]^{z_{i0}} [\lambda p(w_i|\theta_1)]^{z_{i1}}$$

$$p(W, Z|\lambda) = \prod_{i=1}^{k} [(1-\lambda)p(w_i|\theta)]^{z_{i0}} [\lambda p(w_i|\theta_1)]^{z_{i1}}$$

$$ln(p(W, Z|\lambda)) = \sum_{i=1}^{k} z_{i0}[ln(1-\lambda) + ln(p(w_i|\theta))] + z_{i1}[ln(\lambda) + ln(p(w_i|\theta_1))]$$

We need to find the expectation of the above log likelihood with respect to the latent variables. Let us first find an expression for the expectation of z_{i0} and z_{i1}.

$$E[z_{i0}] = p(z_{i1} = 1|w_i)$$

$$E[z_{i0}] = \frac{(1-\lambda_{old})p(w_i|\theta_{old})}{(1-\lambda_{old})p(w_i|\theta_{old}) + \lambda_{old} p(w_i|\theta_1)}$$

Let us call $E[z_{i0}]$ as γ_{i0} for convenience.

Similarly,

$$E[z_{i1}] = \frac{\lambda_{old} p(w_i|\theta_1)}{(1-\lambda_{old})p(w_i|\theta_{old}) + \lambda_{old} p(w_i|\theta_1)}$$

Let us call $E[z_{i1}]$ as γ_{i1} for convenience.

Now we can proceed to find an expression for the expectation of the complete data log likelihood.

$$E[ln(p(W, Z|\lambda))] = \sum_{i=1}^{k} \gamma_{i0}[ln(1-\lambda) + ln(p(w_i|\theta))] + \gamma_{i1}[ln(\lambda) + ln(p(w_i|\theta_1))]$$

Let us maximize this with respect to λ.

$$\frac{\partial E[ln(p(W,Z|\lambda))]}{\partial \lambda} = \sum_{i=1}^{k} \left(\frac{\gamma_{i1}}{\lambda} - \frac{\gamma_{i0}}{1-\lambda}\right) = 0$$

$$\lambda_{new} = \frac{1}{k} \sum_{i=1}^{k} \gamma_{i1}$$

where γ_{i1} is defined above.

Now we have to maximize the expectation with respect to each of $\{p(w_i|\theta)\}_{i=1}^{k}$.

Note that this is a **constrained optimization** and hence we need to use the method of **Lagrange Multipliers**.

We have the following constraint,

$$\sum_{i=1}^{k} p(w_i|\theta) = 1$$

Let us take $f(\{p(w_i|\theta)\}_{i=1}^{k}) = E[ln(p(W, Z|\lambda))]$ and $g(\{p(w_i|\theta)\}_{i=1}^{k}) = \sum_{i=1}^{k} p(w_i|\theta)$

Let us take α as the lagrange multiplier.
Hence we have the following equations,

$$\nabla f(\{p(w_i|\theta)\}_{i=1}^k) = \alpha \nabla g(\{p(w_i|\theta)\}_{i=1}^k)$$
$$g(\{p(w_i|\theta)\}_{i=1}^k) = 1$$

We have $k+1$ equations and $k+1$ variables.
Solving this we get,

$$p(w_i|\theta) = \frac{\gamma_{i0}}{\alpha}$$

Hence,

$$\alpha = \sum_{i=1}^k \gamma_{i0}$$

From this, we get,

$$p(w_i|\theta)_{new} = \frac{\gamma_{i0}}{\sum_{i=1}^k \gamma_{i0}}$$

where γ_{i0} is defined above.

(d) (3 points) [BONUS] The previous parts of the question deal with MLE based density estimation. If you were to employ a Bayesian estimation method to infer λ, how will you proceed? That is, what prior would you choose for λ, and what is the the formula for the posterior? Is this posterior easily computable (i.e., has a closed-form expression or can be computed efficiently)? You can assume that both $P(w|\theta_1)$ and $P(w|\theta)$ are known and only λ is not known.

Solution:

5. (10 points) [DENSITY ESTIMATION - THE ONE RING TO RULE THEM ALL!] With density estimation ring already in your finger, you have all you need to master simple linear regression (even before seeing regression formally in class). Simple linear regression is a model that assumes a linear relationship between an input (aka independent) variable x and an output (aka dependent) variable y. Let us assume that the available set of observations, $\mathbb{D} = \{x_i, y_i\}_{i=1}^n$, are iid samples from the following model that captures the relationship between y and x:

$$y_i = w_0 + w_1 x_i + \epsilon_i; \qquad \epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

In this model, note that x_i is not a random variable, whereas ϵ_i and hence y_i are random variables, with ϵ_i being modeled as a Gaussian noise that is independent of each other and doesn't depend on x_i. Value of σ is assumed to be known for simplicity.
We would like to learn the parameters $\theta = \{w_0, w_1\}$ of the model, i.e., we would like to use MLE to estimate the exact parameter values or Bayesian methods to infer the (posterior) probability distribution over the parameter values.

(a) (2 points) Compute the probability distribution $P(y_i|x_i, \theta)$, and use it to write down the log likelihood of the model.

Solution: We are given that $y_i = w_1 x_i + w_0 + \epsilon_i$, where $p(\epsilon_i) = \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{1}{2\sigma^2}\epsilon_i^2)$. We can write the likelihood of y_i as follows,

$$P(y_i|x_i, \theta) = P(\epsilon_i = y_i - w_1 x_i - w_0)$$
$$P(y_i|x_i, \theta) = \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{1}{2\sigma^2}(y_i - w_1 x_i - w_0)^2)$$

Hence, the data log likelihood is as follows,

$$p(\mathbb{D}|\theta) = \prod_{i=1}^{n} P(y_i|x_i, \theta)$$
$$p(\mathbb{D}|\theta) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{1}{2\sigma^2}(y_i - w_1 x_i - w_0)^2)$$
$$ln(p(\mathbb{D}|\theta)) = -\frac{n}{2} ln(2\pi) - n ln(\sigma) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - w_1 x_i - w_0)^2$$

The above expression is the log likelihood of the model.

(b) (3 points) Derive the ML estimates for w_0 and w_1 by optimizing the above log likelihood.

Solution: We will first find the ML estimate of w_0. To do this, we need to compute the derivative of the data log likelihood with respect to w_0 and equate it to zero.

$$\frac{\partial ln(p(\mathbb{D}|\theta))}{\partial w_0} = \frac{1}{2\sigma^2} \sum_{i=1}^{n} 2(y_i - w_1 x_i - w_0) = 0$$
$$w_{0ML} = \frac{1}{n} \sum_{i=1}^{n} (y_i - w_{1ML} x_i)$$

Now, let us find the ML estimate of w_1.

$$\frac{\partial ln(p(\mathbb{D}|\theta))}{\partial w_0} = \frac{1}{2\sigma^2} \sum_{i=1}^{n} 2x_i(y_i - w_1 x_i - w_0) = 0$$
$$w_{1ML} = \frac{\sum_{i=1}^{n} x_i(y_i - w_{0ML})}{\sum_{i=1}^{n} x_i^2}$$

Here, we have two linear equations with two variables, w_{0ML}, w_{1ML}. Solving these,

$$w_{0ML} = \frac{\sum_{i=1}^{n}(y_i \sum_{i=1}^{n} x_{i^2} - x_i \sum_{i=1}^{n} x_i y_i)}{n\sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} x_i)^2}$$
$$w_{1ML} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} y_i \sum_{i=1}^{n} x_i}{n\sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} x_i)^2}$$

(c) (2 points) If σ is also not known before, derive the ML estimate for σ.

Solution: To find the ML estimate of σ, we need to find the derivative of the data log likelihood with respect to σ and equate it to zero.

$$\frac{\partial ln(p(\mathbb{D}|\theta))}{\partial \sigma} = -\frac{n}{\sigma} + \frac{1}{\sigma^3} \sum_{i=1}^{n} (y_i - w_1 x_i - w_0)^2 = 0$$
$$\sigma_{ML} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - w_{1ML} x_i - w_{0ML})^2}$$

where w_{0ML}, w_{1ML} are defined in the previous part.

(d) (3 points) For Bayesian inference, assume that the parameters w_0, w_1 are independent of each other and follow the distributions $\mathcal{N}(\mu_0, \sigma_0^2)$ and $\mathcal{N}(\mu_1, \sigma_1^2)$ respectively. Compute the posterior distributions for each parameter. How does the mode of this posterior (i.e., MAP estimate) relate to the MLE of w_0 and w_1 derived above?

Solution: We are given,

$$p(w_0) = \frac{1}{\sqrt{2\pi}\sigma_0} exp(-\frac{1}{2\sigma_0^2}(w_0 - \mu_0)^2)$$

The posterior of w_0 is proportional to the likelihood multiplied by the prior. The MAP Estimate would be the same as the estimate obtained by maximizing the likelihood multiplied by the prior. Hence,

$$p(D|w_0)p(w_0) = \frac{1}{\sqrt{2\pi}\sigma_0} exp(-\frac{1}{2\sigma_0^2}(w_0 - \mu_0)^2) \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{1}{2\sigma^2}(y_i - w_1 x_i - w_0)^2)$$

$$ln(p(D|w_0)p(w_0)) =$$

$$-\frac{n}{2}ln(2\pi) - nln(\sigma) - \frac{1}{\sigma^2}\sum_{i=1}^{n}((y_i - w_1 x_i - w_0)^2) - \frac{1}{2}ln(2\pi) - ln(\sigma_0) - \frac{1}{2\sigma_0^2}(w_0 - \mu_0)^2$$

Differentiating the above expression with respect to w_0 and equating it to zero, we get the estimate of w_0 as the following,

$$w_0 = \frac{\frac{\mu_0}{\sigma_0^2} + \frac{1}{\sigma^2}\sum_{i=1}^{n}(y_i - w_1 x_i)}{\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}}$$

Similarly,

$$ln(p(D|w_1)p(w_1)) =$$

$$-\frac{n}{2}ln(2\pi) - nln(\sigma) - \frac{1}{\sigma^2}\sum_{i=1}^{n}((y_i - w_1 x_i - w_0)^2) - \frac{1}{2}ln(2\pi) - ln(\sigma_1) - \frac{1}{2\sigma_1^2}(w_1 - \mu_1)^2$$

Differentiating the above expression with respect to w_1 and equating it to zero, we get the MAP estimate of w_1 to be the following,

$$w_1 = \frac{\frac{\mu_1}{\sigma_1^2} + \frac{1}{\sigma^2}\sum_{i=1}^{n} x_i(y_i - w_0)}{\frac{1}{\sigma_0^2} + \frac{\sum_{i=1}^{n} x_i^2}{\sigma^2}}$$

By looking at the expressions of the MAP Estimates, we can see how it relates to the ML Estimates.

If we set μ_0 to zero and σ_0 tends to infinity, the MAP estimate of w_0 tends to the its ML estimate. This can be verified intuitively as follows. As μ_0 and σ_0 tend to zero we get a flat prior and the MAP estimate with a flat prior is the same as the ML estimate because the prior does not give any additional information.

MAP Estimate of w_1 is related to its ML estimate in the same way as above.

6. (10 points) [LET'S ROLL UP YOUR CODING SLEEVES...] **Learning Binary Bayes Classifiers from data via Density Estimation**
   Derive Bayes classifiers under assumptions below and employing maximum likelihood approach to estimate class prior/conditional densities, and return the results on a test set.

   1. **BayesA** Assume $X|Y = -1 \sim \mathcal{N}(\mu_-, I)$ and $X|Y = 1 \sim \mathcal{N}(\mu_+, I)$
   2. **BayesB** Assume $X|Y = -1 \sim \mathcal{N}(\mu_-, \Sigma)$ and $X|Y = 1 \sim \mathcal{N}(\mu_+, \Sigma)$
   3. **BayesC** Assume $X|Y = -1 \sim \mathcal{N}(\mu_-, \Sigma_-)$ and $X|Y = 1 \sim \mathcal{N}(\mu_+, \Sigma_+)$

   Please see this folder for the template .ipynb file containing the helper functions, and you've to add

the missing code to this file (specifically, three functions function_for_A, function_for_B and function_for_C, and associated plotting/ROC code snippets) to implement the above three algorithms for the three datasets given in the same folder.

Please provide your results/answers in the pdf file you upload to GradeScope, but please submit your code separately in this moodle link. The code submitted should be a rollno.zip file containing two files: rollno.ipynb file (including your code as well as the exact same results/plots uploaded to Gradescope) and the associated rollno.py file.

(a) (3 points) Plot all the classifiers (3 classification algorithms on 3 datasets = 9 plots) on a 2D plot, Add the training data points also on the plots. (Color the positively classified area light green, and negatively classified area light red as in Fig 4.5 in Bishop's book).

Solution:

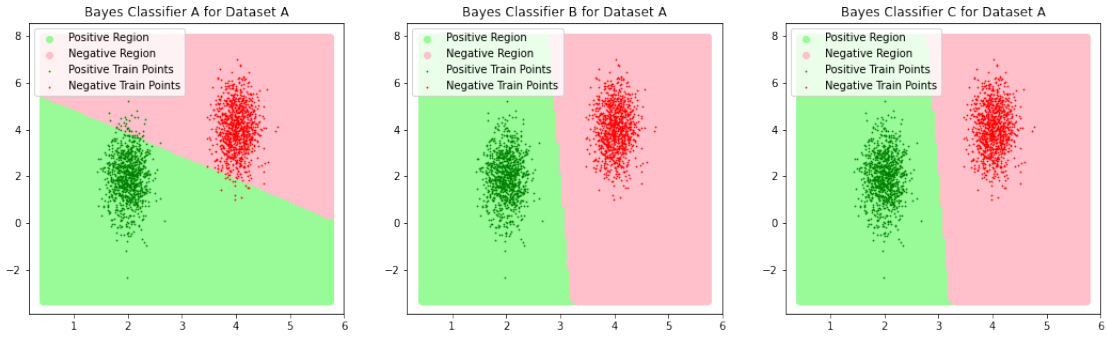


Figure 2: Plot of Dataset A

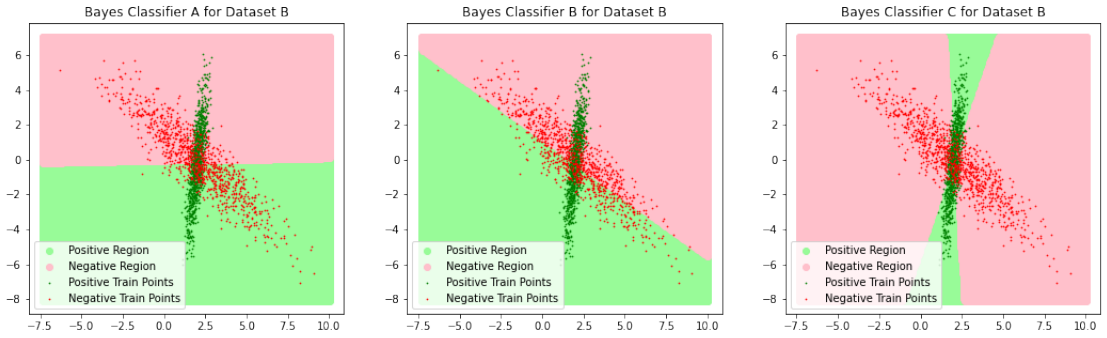


Figure 3: Plot of Dataset B

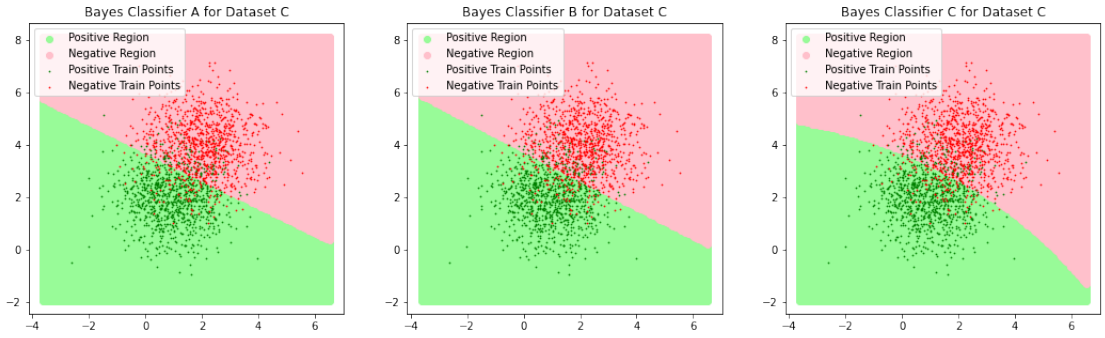


Figure 4: Plot of Dataset C

(b) (3 points) Give the ROC curves for all the classifiers. Note that a ROC curve plots the FPR (False Positive Rate) on the x-axis and TPR (True Positive Rate) on the y-axis. (9 plots)

Solution:

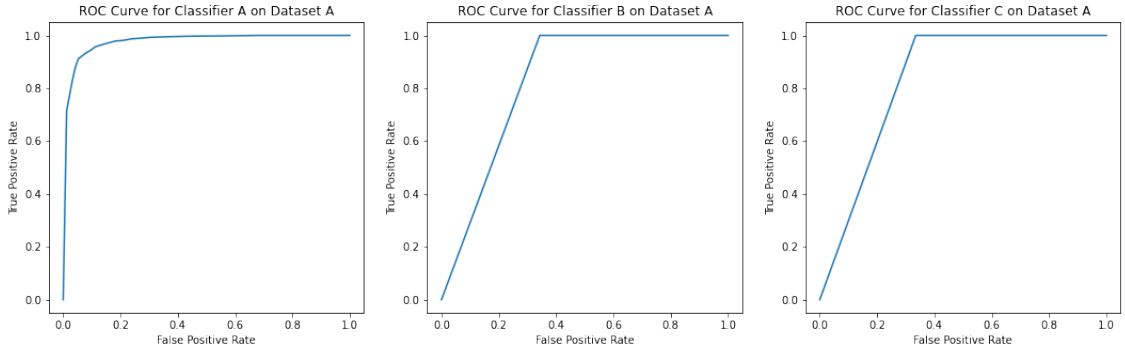


Figure 5: Plot of Dataset A ROC

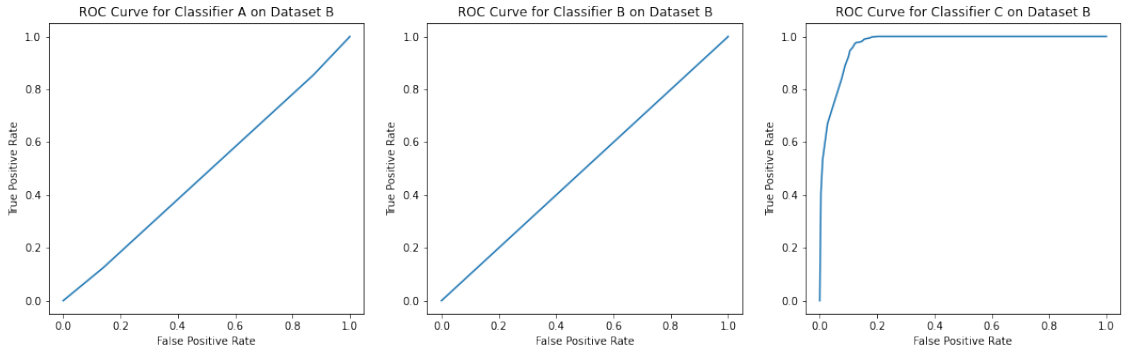


Figure 6: Plot of Dataset B ROC

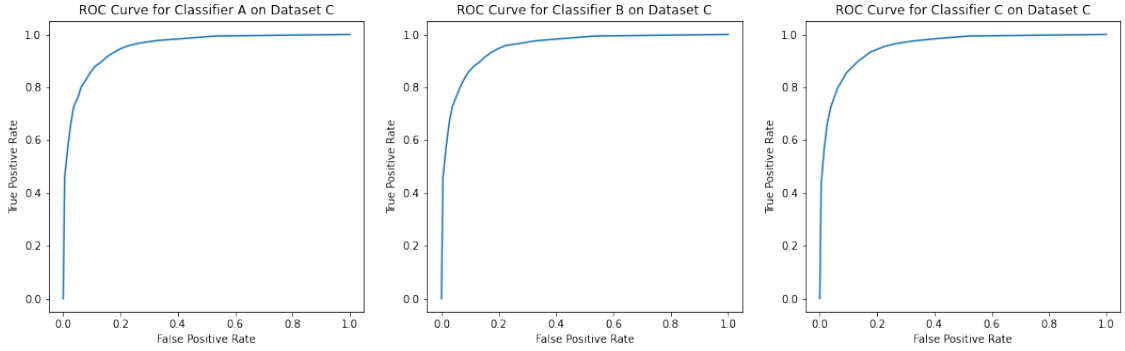


Figure 7: Plot of Dataset C

(c) (2 points) Provide the error rates for the above classifiers (three classifiers on the three datasets as 3×3 table, with appropriately named rows and columns).

Solution:

Error	Bayes A	Bayes B	Bayes C
Dataset A	9.79	22.89	22.55
Dataset B	50.85	50.40	7.45
Dataset C	11.75	11.65	11.8

Note that the above error rates are in percentage.

(d) (2 points) Summarise and explain your observations based on your plots and the assumptions given in the problem. Also briefly comment whether a non-parametric density estimation approach could have been used to solve this problem, and if so, what the associated pros/cons are compared to the parametric MLE based approach you have implemented.

Solution: The first thing to notice is that the decision boundary of classifiers A and B is always linear in nature whereas the decision boundary of classifier C can be quadratic. This is because classifier C is allowed to have different covariance matrices for its classes. Hence, the quadratic terms do not cancel out in general and it can have a non-linear boundary. This can be very clearly seen in the plots of dataset B.

Another thing to notice is the nature of the ROC curves. For dataset B, the ROC of classifier C has a much higher discriminability than classifiers A and B. This is because dataset B has different covariances of its classes and hence a non-linear decision boundary. This is why classifier C performs much better than the other classifiers.

Dataset C is linearly separable and the two classes have approximately the same covariance. Hence, all 3 classifiers perform similarly.

In dataset A, the test set does not represent the train set, hence the errors are high.

For these datasets, non-parametric methods would not perform much better than classifier C because the inherent decision boundaries of the datasets are quadratic and the datasets are generated from Gaussians in nature and classifier C can learn quadratic decision boundaries.

Also non-parametric methods would be more computationally expensive.