Edward Mina

Intro to Machine Learning, Pr. Deniz Erdogmus

Question 1:

The provided data set has two define classes irrespective of the features such that class 1 (ω_1) is defined as the absence of a person and class 2 is the contrary, the presence of someone in the room(ω_2). Given each feature (x_1,x_2) with respect to the sensor and class, there exists the following sets of likelihood probability distributions under each circumstance:

1. $P(x_1 \mid \omega_1)$ = Gaussian distribution with mean of m_o and variance of s^2. This distribution is labelled by the feature space of sensor one (x_1) describing temperature under the conditions that no one is in the room.
2. $P(x_1 \mid \omega_2)$ = Gaussian(m_1, s^2). This distribution also describes x_1 under the conditions that someone is in the room.
3. $P(x_2 \mid \omega_1)$ = Gaussian(0, r^2). This distribution is labelled by the feature space of sensor two (x_2) under the conditions that no one is in the room.
4. $P(x_2 \mid \omega_2)$ = Gaussian(n, r^2). Probability of feature x_2 given that someone is in the room (ω_2.)

Risk for these decisions will be defined as

1. $R(a_1 \mid x) = \lambda_{11} * P(\omega_1 \mid x) + \lambda_{12} * P(\omega_2 \mid x)$ where x is a feature vector space x = [x_1,x_2], a_1 is the action of determining that the room is empty, $\lambda_{ij} = \lambda(a_i \mid w_j)$ which is the associated loss with the knowledge that a room is empty or not and acting either correctly or incorrectly. In the case of 0-1 loss, the loss function is specifically defined as:

$$\lambda(a_i/\omega_j) = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases}$$

   With this loss function, the risk is simplified as: $R(a_1 \mid x) = P(\omega_2 \mid x)$

2. $R(a_2 \mid x) = \lambda_{21} * P(\omega_1 \mid x) + \lambda_{22} * P(\omega_2 \mid x)$. As above, the following describes the risk associated with choosing that a room is occupied given the features from both sensors. More specifically, the loss of deciding a room is occupied when it's not multiplied by the probability that is not summed to the associated loss of deciding the room is empty when it is multiplied by its posterior.
   Using the loss function above, the risk is simplified as: $R(a_2 \mid x) = P(\omega_1 \mid x)$

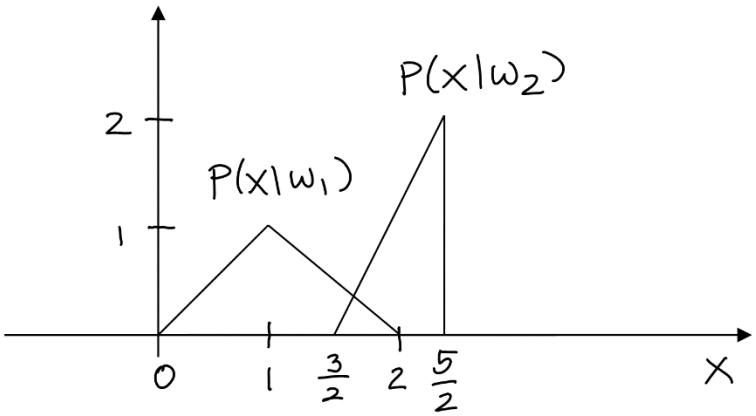
In order to minimize risk a decision rule is established below modelled by the likelihood ratio.

1. Decide room is empty if $R(a_1 \mid x) < R(a_2 \mid x) = P(\omega_2 \mid x) < P(\omega_1 \mid x)$
2. Likelihood Equation
   a. Deciding Class 1 (room is empty) when $\frac{P(x \mid \omega 1)}{P(x \mid \omega 2)} > \frac{P(\omega 2)}{P(\omega 1)} = 1$ (Priors are deemed equivalent in the question.)
   b. Is it important to note that x is a vector consisting of features such that:
      i. $P(x \mid \omega 1) = P\left(\begin{bmatrix} x1 \\ x2 \end{bmatrix} \middle| \omega 1\right) = P(x1|\omega 1) * P(x2|\omega 1) = Gaussian(m_o, s^2) * Gaussian(0, r^2)$
      ii. $P(x \mid \omega 2) = P\left(\begin{bmatrix} x1 \\ x2 \end{bmatrix} \middle| \omega 2\right) = P(x1|\omega 2) * P(x2|\omega 2) = Gaussian(m_1, s^2) * Gaussian(n, r^2)$
3. $\frac{P(x|\omega 1)}{P(x|\omega 2)} \underset{<(2)}{>(1)} 1 = \frac{Gaussian(m_o, s^2) * Gaussian(m_1, s^2)}{Gaussian(0, s^2) * Gaussian(n, s^2)} \underset{<(1)}{>(2)} 1$
   a. Note that Gaussian(x,m, σ_i) = $\frac{1}{(2\pi)^{\frac{1}{2}}\sigma_i} e^{-\frac{\frac{1}{2}(x-m)^2}{\sigma_i^2}}$
   b. $\frac{P(x|\omega 1)}{P(x|\omega 2)} \underset{<(2)}{>(1)} 1 = \frac{\frac{1}{(2\pi)^{\frac{1}{2}}s} e^{-\frac{\frac{1}{2}(x1-m_o)^2}{s^2}} * \frac{1}{(2\pi)^{\frac{1}{2}}r} e^{-\frac{\frac{1}{2}(x2)^2}{r^2}}}{\frac{1}{(2\pi)^{\frac{1}{2}}s} e^{-\frac{\frac{1}{2}(x1-m_1)^2}{s^2}} * \frac{1}{(2\pi)^{\frac{1}{2}}r} e^{-\frac{\frac{1}{2}(x2-n)^2}{r^2}}} = \frac{e^{-\frac{\frac{1}{2}(x1-m_o)^2}{s^2}} * e^{-\frac{\frac{1}{2}(x2)^2}{r^2}}}{e^{-\frac{\frac{1}{2}(x1-m_1)^2}{s^2}} * e^{-\frac{\frac{1}{2}(x2-n)^2}{r^2}}}$
   c. $lnP(x|\omega 1) - lnP(x|\omega 2) \underset{<(2)}{>(1)} 0 = \frac{(x1-m_o)^2}{s^2} + \frac{(x2)^2}{r^2} - \frac{(x1-m_1)^2}{s^2} - \frac{(x2-n)^2}{r^2} \underset{<(2)}{>(1)} 0$
   d. $A = \frac{-2x_1m_o + 2x_1m_1 + m_o^2 + m_1^2}{s^2} - \left(\frac{-2x_2n + n^2}{r^2}\right) \underset{<(2)}{>(1)} 0$

Therefore, in designing the algorithms that use the sensor data, a simple Bayesian decision process can be done such that the main question being solved is whether a person in the room or not. Mathematically speaking, this is the probability given data from each sensor of what is the chance someone in in the room (Posterior = $P(\omega_2 \mid x)$). Data from the sensors are likelihood distributions based on the premise of whether or not someone is in the room with respect to the feature space each sensor models. Using Bayes decision theory, the algorithm will use the probability distributions it has been given from the sensor data to solve to the decision with the highest probability and lowest risk assessment. The priors are irrelevant in this decision since they are equal. Therefore the algorithm will look something like this:

1. Observe sensor 1 data and with distribution mean and variance $P(x_1 \mid \omega_1)$ & $P(x_1 \mid \omega_2)$
2. Repeat for sensor 2 data $P(x_2 \mid \omega_1)$ & $P(x_2 \mid \omega_2)$
3. For decision rule A (shown in step d above):
    a. If $A < 0$, decide someone is in the room
    b. Else: Room is empty
        i. $A = \frac{-2x_1m_o+2x_1m_1+m_o^2+m_1^2}{s^2} - \left(\frac{-2x_2n+n^2}{r^2}\right)$

Question 2:



$$P(x|\omega_1) = \begin{cases} x, & 0 \le x \le 1 \\ 2-x, & 1 \le x \le 2 \\ 0, & \text{else} \end{cases}$$

$$P(x|\omega_2) = \begin{cases} 2x-3, & \frac{3}{2} \le x \le \frac{5}{2} \\ 0, & \text{else} \end{cases}$$

(a) Minimum Risk Decision Rule using a discrimination function:

$Given\ the\ assumption\ of\ General\ Case\ of\ Loss\ (Not\ 0-1\ Loss)$:

$Discriminant\ Function\ g(x) = g1(x) - g2(x) = -R(a1|x) - \left(-R(a2|x)\right)$

$Given: R(a1\ |x)\ =\ \lambda_{11} * P(\omega1 \mid x)\ + \lambda_{12} * P(\omega2 \mid x)$

$\&\ R(a2\ |x)\ =\ \lambda_{21} * P(\omega1 \mid x)\ + \lambda_{22} * P(\omega2 \mid x)$

$g(x) = -\lambda_{11} * P(\omega1 \mid x) - \lambda_{12} * P(\omega2 \mid x) + \lambda_{21} * P(\omega1 \mid x)\ + \lambda_{22} * P(\omega2 \mid x)$

$g(x) = P(x \mid \omega1)(\lambda_{21}-\lambda_{11}) + P(\omega2 \mid x)(\lambda_{22} - \lambda_{12})$

Discriminant Function:

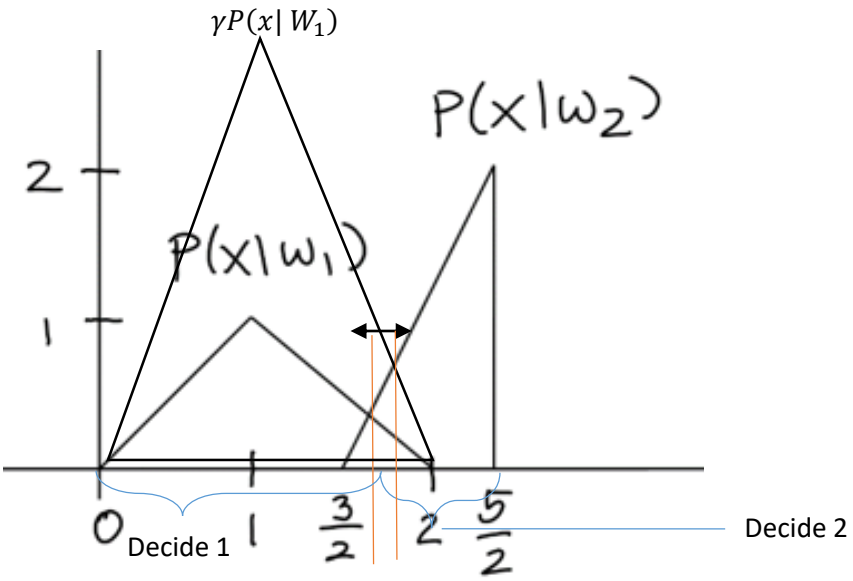
$Application\ of\ Bayes \rightarrow g(x) = P(x|\ \omega1)P(\omega1)(\lambda_{21}-\lambda_{11}) + P(x|\ \omega2)P(\omega2)(\lambda_{22} - \lambda_{12})$

$Decision\ Rule$:

$\frac{P(x|\omega2)}{P(x|\ \omega1)} \begin{matrix} > (2) \\ < (1) \end{matrix} \frac{(\lambda_{21}-\lambda_{11})P(\omega1)}{(\lambda_{12}-\lambda_{22})P(\omega2)} = \gamma(\lambda, P(\omega))$

$\frac{P(x|\omega2)}{\gamma * P(x|\ \omega1)} \begin{matrix} > (2) \\ < (1) \end{matrix} = 1$

(b) ROC Curve:



Threshold (Orange Line) $\tau(\gamma)$ *such that it intersects the two curves*: $2 - x = 2x - 3 \rightarrow x = \tau = \frac{5}{3}$.

For $\gamma = 1$:

$$P(Decision = W_2 | Truth = W_2) = \int_{\tau}^{\frac{5}{2}} P(x|W_2)dx = \int_{\tau}^{\frac{5}{2}} (2x - 3)dx = \frac{35}{36}$$

$$P(Decision = W_2 | Truth = W_1) = \int_{\tau}^{2} P(x|W_1)dx = \int_{\tau}^{2} (2 - x)dx = \frac{1}{18}$$

For $\gamma = 2$:

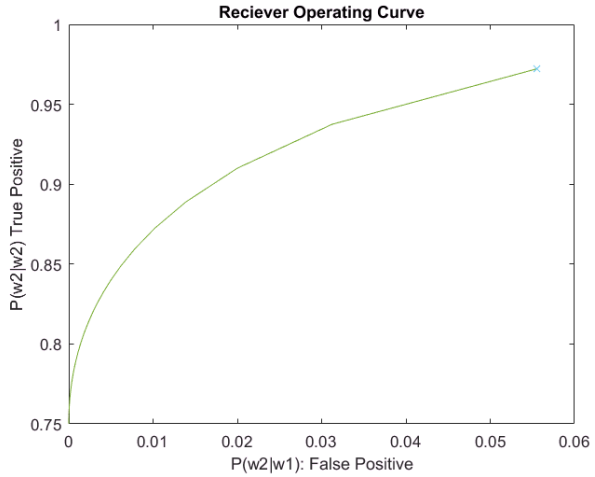
$$\gamma = 2: \tau = \frac{7}{4}, P(W_2|W_1) = \frac{1}{32}, P(W_2|W_2) = \frac{15}{16}$$

For *General Procedure* $\rightarrow \gamma(2 - x) = 2x - 3 \rightarrow x = \tau(\gamma) = \frac{2\gamma+3}{2+\gamma}$

$$P(W_2|W_1) = 2 - 2\tau + \frac{\tau^2}{2}$$

$$P(W_2|W_2) = -\frac{5}{4} + 3\tau - \tau^2$$

Below is a Rough Approximation of the ROC curve for a limited range since the distribution is piecewise:




```
for y= 1:1000
    t(y)= (2*y+3)/(2+y);
    x1=2-2*t+t.^2/2;
    x2=-5/4+3*t-t.^2;
end
plot(x1,x2)
```

*Note that the operator curve extends further to the right with a max probability of 1 (not .6), but this was done so that the blue x is the first point represents gamma at 0-1 loss (see below).

(c) Minimum expected loss decision for 0-1 Loss

$$General\ Loss: \frac{P(x|\omega 2)}{P(x|\ \omega 1)} \gtrless \frac{(2)}{(1)}\ \gamma = \frac{(\lambda_{21}-\lambda_{11})P(\omega 1)}{(\lambda_{12}-\lambda_{22})P(\omega 2)}$$

$$0-1\ Loss, Equal\ Priors: \frac{P(x|\omega 2)}{P(x|\ \omega 1)} \gtrless \frac{(2)}{(1)}\ \gamma = 1$$

Thus when γ=1, then which is at the point marked on the plot as the blue dot. This is because the for loop starts at 1 and gamma increases each respective probability decreases. The math for this is proven above at the points (1/18, 35/36).

Question 3:

a.) Maximum Likelihood Estimate (MLE) of parameter λ for a Poisson distribution:

(1) $P(k) = e^{-\lambda}\frac{\lambda^k}{k!}$

(2) $For\ data\ set\ \{k_1, k_2, \dots, k_N\}, Likelihood\ is\ defined\ as\ P(k_1, k_2, \dots, k_N|\lambda)$

(3) $\text{Log} - \text{Likelihood} = \sum \ln P(k_1, k_2, \dots, k_N|\lambda) = \sum(-\lambda + kln\lambda - lnk!) = ln\lambda \sum k - n\lambda - \sum lnk!$ *

(4) Differentiation: $\frac{d}{d\lambda} ln\lambda \sum k - n\lambda - \sum lnk! = \frac{1}{N}\sum k$ *

(5) Solution: $\hat{\lambda} = \overline{K}$

**summation is from 1 to N for all values of K*

Poisson Variance (NOT MLE)=

$$Var(K) = E(K^2) - \big(E(K)\big)^2 = E(K^2 - K + K) - \lambda^2 = E(K(K-1) + K) - \lambda^2$$

$$E(K(K-1) + K) + \lambda^2 = \sum k(k-1) * e^{-\lambda}\frac{\lambda^k}{k!} + \lambda - \lambda^2 \rightarrow Previous\ Derivation\ \sum k * e^{-\lambda}\frac{\lambda^k}{k!} = \lambda$$

Computing MLE mean and variance. (Proof of unbiased estimators)

General Expectations for Poisson Distribution

Mean:

$$E(K) = \sum k * Pr(K = k)$$

$$E(K) = \sum k * e^{-\lambda}\frac{\lambda^k}{k!}$$

$$E(K) = \lambda e^{-\lambda} e^{\lambda} = \lambda$$

$$E\big(\hat{\lambda}\big) = E(\overline{K}) = \frac{1}{N}\sum E(K) = \frac{1}{N}\sum \lambda = \lambda$$

Variance:

$$Expectation\ of\ Negative\ 2nd\ derivative\ of\ Log - Likelihood$$

$$-E[-\frac{d^2}{d\lambda^2}(\sum \ln P(k_1, k_2, \dots, k_N|\lambda)]$$

Fisher Information (Measurement of PDF Sharpness which is inversely related to Variance)

$$= -E\left[\frac{1}{\lambda^2}\sum k\right] = n\frac{\lambda}{\lambda^2} = \frac{1}{Var(MLE)} \rightarrow Var = \frac{\lambda}{n}$$

b.) Determine MAP Decision for $P(\omega_i \mid D)$ (Given data set D, what is the probability it belongs to some class ω_i.)

MAP: (Determine class that maximizes the Probability Function)

$$\hat{\omega} = \text{argmax}\ P(\omega i \mid D) = \text{argmax}\ P(D|\omega i\ )P(\omega i)$$

$$\hat{\omega} = \text{argmax}[\sum l\text{n}P(D|\omega i\ ) + \ln P(\omega i)]$$

$$\hat{\omega} = \text{argmax}[\sum \ln e^{-\lambda}\frac{\lambda^k}{k!} + \ln P(\omega i)]$$

$$\hat{\omega} \rightarrow argmax\left[\sum -\lambda + kln\lambda\ - ln k! + lnP(\omega i)\right]$$

$$\hat{\omega} \rightarrow argmax\left[\sum kln\lambda\ + lnP(\omega i)\right]\ (remove\ constants)$$

Question 4:

Part A:

A Gaussian multivariate distribution can be verified under the following statement X= Az + b such that X is a Gaussian centered at mean b, and having a covariance equal to: AAT. On the other hand, z is a Gaussian with a 0 mean and identity covariance. Gaussian (b, AA^T) = A * Gaussian(0,I) + b. Note, in order for AA^T (presumed to be a symmetric covariance matrix), to be positive semidefinite, A is a regular (invertible) matrix according to basic spectral theory. Below is a simple plot of x in python's distplot function where A= [.6325 0; 0 .6325] regular matrix such that its product to its transpose it the symmetric covariance matrix used. The mean vector, b, is some arbitrarily selected value [1 1]. The plot of these figures is shown directly below. Further below this figure is a plot of A*z+b where z has an identity covariance and 0 mean. Proof of x=Az+b is shown in the nature of how the plots overlap well at the same mean and share similar stretches of points (variance).

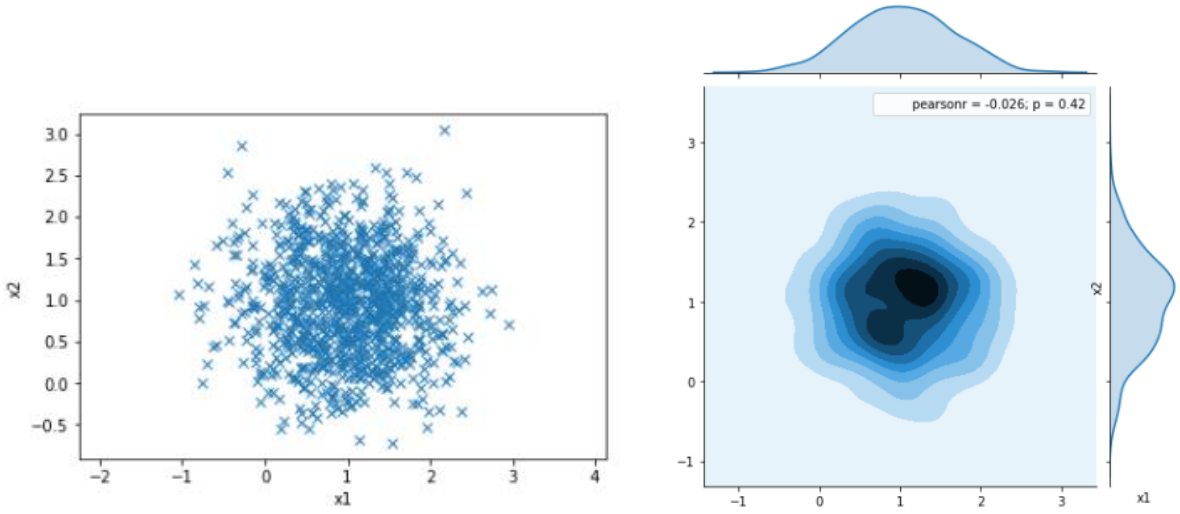


Figure A: The following are two plots of x, with a predefined mean and covariance which is a function of some regular matrix A.

Choosing a value of A is done by predefining a symmetric covariate matrix such that it is a product of a regular matrix and its transpose. This is validated using Eigen decomposition of the covariate matrix:

$$\Sigma = A * A^T = \begin{bmatrix} .4 & 0 \\ 0 & .4 \end{bmatrix}$$

$$\Sigma = V * D * V^T$$

$$\Sigma = V * D^{\frac{1}{2}} * V^T * V * D^{\frac{T}{2}} * V^T, where\ V^T * V = I\ and\ D = D^{\frac{1}{2}} * D^{\frac{T}{2}}$$

$$\Sigma = A * A^T = V * D^{\frac{1}{2}} * V^T * V * D^{\frac{T}{2}} * V^T, therefore\ A = V * D^{\frac{1}{2}} * V^T$$

$$A = \begin{bmatrix} .6325 & 0 \\ 0 & .6325 \end{bmatrix}$$

```
cov=[.4 0; 0 .4];
[V D]=eig(cov)
```

```
V = 2×2
     1     0
     0     1

D = 2×2
    0.4000         0
         0    0.4000
```

```
A=(V*(D^(1/2))'*V')
```

```
A = 2×2
    0.6325         0
         0    0.6325
```

```
A*A'
```

```
ans = 2×2
    0.4000         0
         0    0.4000
```

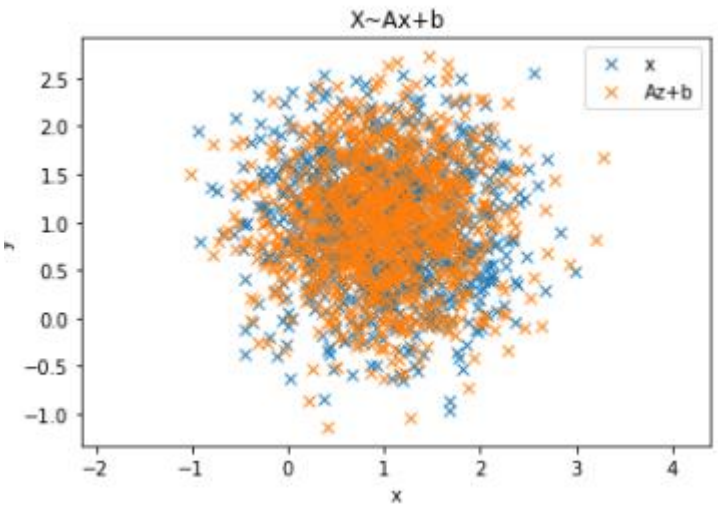


*Figure B: The following is a visual proof the of distribution x equating to Az+b where A*A^T is the covariate matrix for x, b is the mean, and z is a distribution with 0 mean and identity variance. The plots overlap with equal variance and are centered around the same mean.*

Part B:

Shown in question 2, the discriminant for a function assuming general loss is:

$$g(x) = \ln P(\omega i \mid x) = \ln P(x|\omega) + lnP(\omega i)$$

$$P(\omega j \mid x) = Gaussian\ multivariate = p(x) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} * e^{-\frac{1}{2}([(x-\mu)^T \Sigma^{-1}(x-\mu)])}$$

$$g(x) = \ln P(\omega i \mid x) = \ln(\frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}}) - 1/2 * ([(x-\mu)^T \Sigma^{-1}(x-\mu)]) + lnP(\omega i)$$

$$g(x) = -\frac{d}{2}\ln(2\pi) - \frac{1}{2} ln|\Sigma| - \frac{1}{2}([(x-\mu)^T \Sigma^{-1}(x-\mu)]) + lnP(\omega i)$$

CASE III Clusters (Distributions have different shapes and sizes)

$$gi(x) = -\frac{1}{2} ln|\Sigma| - \frac{1}{2}([(x-\mu)^T \Sigma^{-1}(x-\mu)]) + lnP(\omega i) = x^T W_i x + w\hat{}T_i\, x + w_{io}$$

$$\begin{cases} W_i = -\frac{1}{2} * \Sigma i^{-1} \\ w_i = \Sigma i^{-1} * \mu i \\ w_{io} = -\frac{1}{2} * (\mu i)^T \Sigma i^{-1} (\mu i) - \frac{1}{2} ln|\Sigma i| + lnP(\omega i) \end{cases}$$

$$gj(x) = -\frac{1}{2} ln|\Sigma| - \frac{1}{2}([(x-\mu)^T \Sigma^{-1}(x-\mu)]) + lnP(\omega j)$$

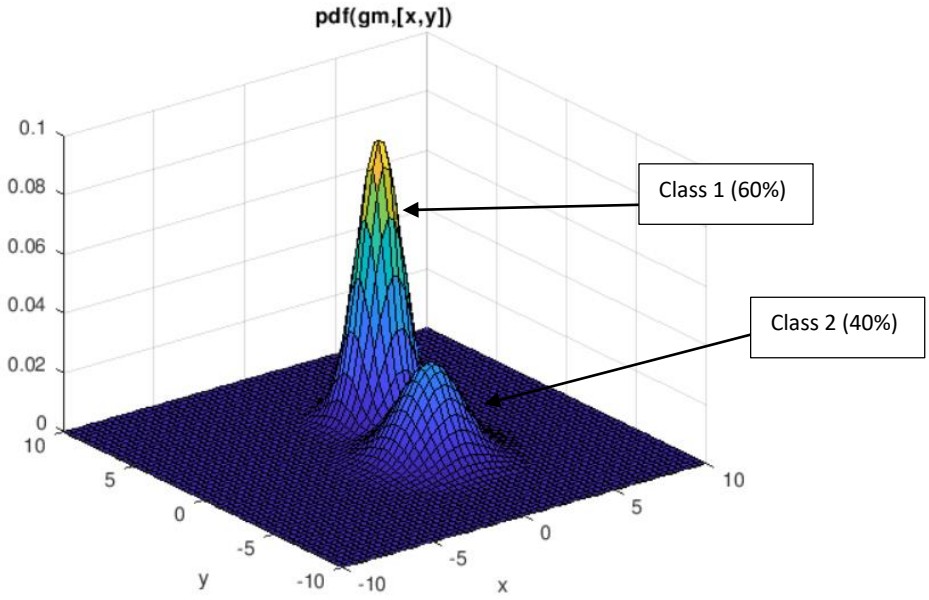
$$Boundary@\ gi(x) - gj(x) = P(x|\omega i)P(\omega i) - P(x|\omega j)P(\omega j)$$

$$x^T * -\frac{1}{2} * \Sigma i^{-1} x + (\Sigma i^{-1} * \mu i)^T * x - \frac{1}{2} * (\mu i)^T \Sigma i^{-1} (\mu i) - \frac{1}{2} ln|\Sigma i| + lnP(\omega i)$$
$$= x^T * -\frac{1}{2} * \Sigma j^{-1} x + (\Sigma j^{-1} * \mu j)^T * x - \frac{1}{2} * (\mu j)^T \Sigma j^{-1} (\mu j) - \frac{1}{2} ln|\Sigma j| + lnP(\omega j)$$

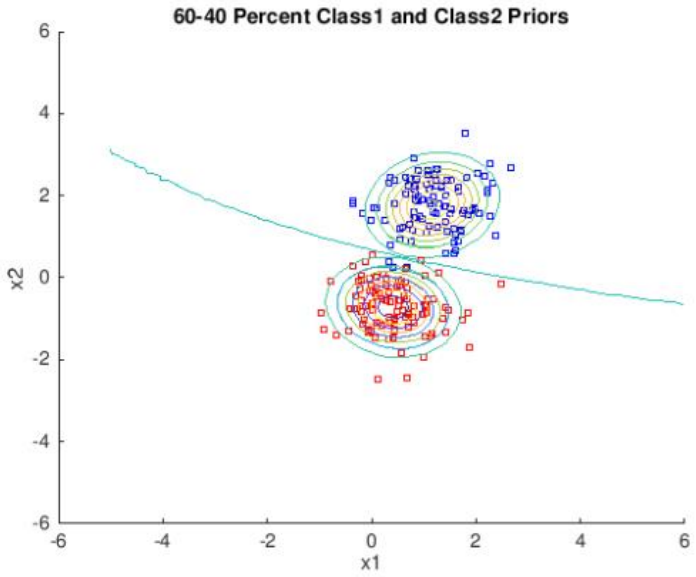
$x^T = \binom{x1}{x2}$ → Solve for x2 in terms of x1 or plot the function as a single level contour with as the difference in the given PDF's.

QDA for Unequal Priors:

3D Plot: Gaussian Mixture Model with respective mixing portions of 60% and 40%.

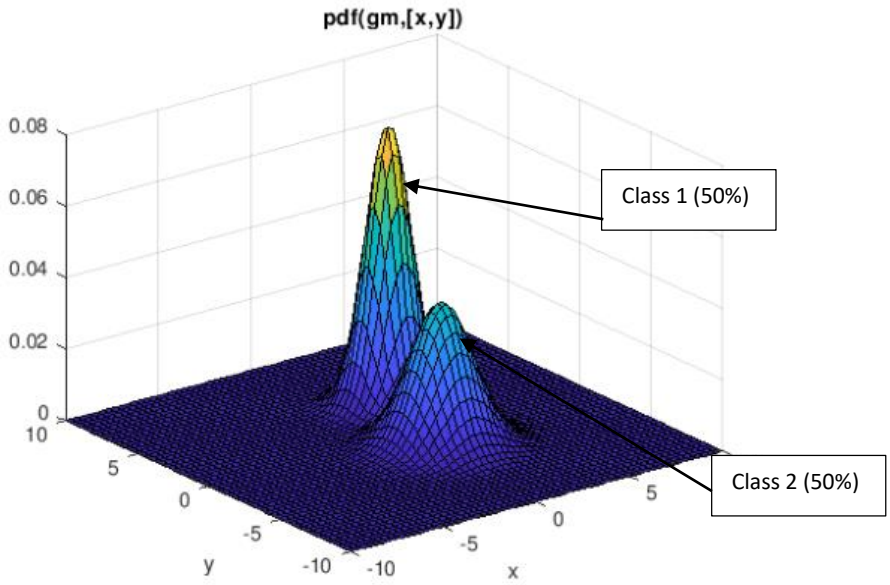


Contour Plot and associated discriminant function:

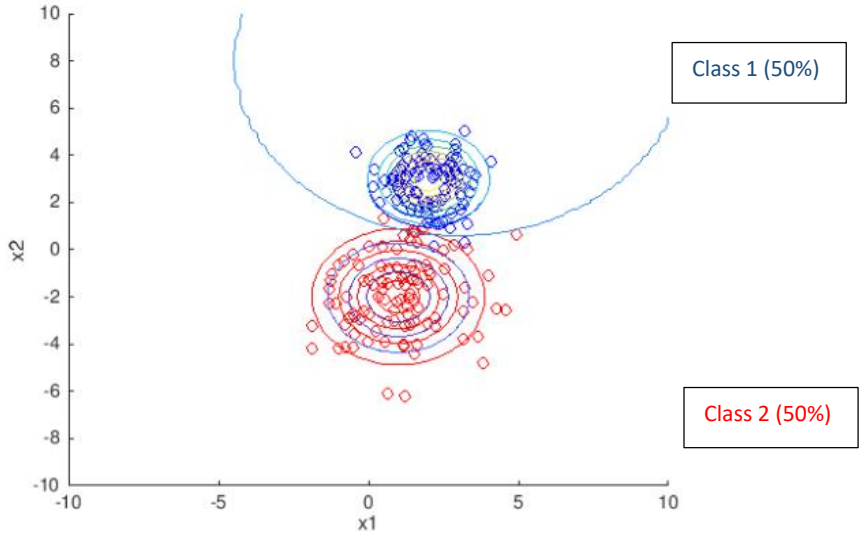


QDA for Equal Priors (.5):

3D Plot: Gaussian Distributions with mean1= 2,3 and mean2= 1,-2 with respective mixing portions of 50% and 50%. Note that the peak for class two is higher because of its larger contribution the to the model.



Development of the quadratic decision boundary using MAP Estimation:



The QDA has the following Error Count

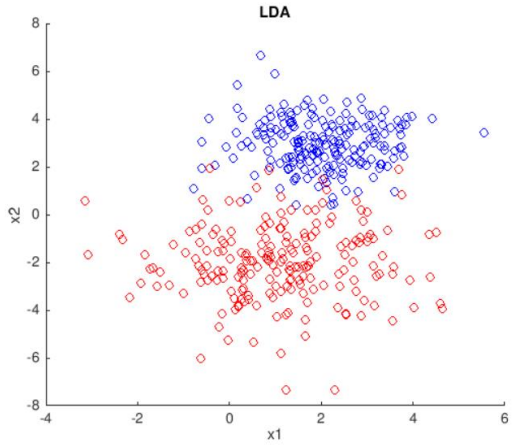
Error Total = Error(Choosing 1: True=2)+ Error(Choosing 2: True=1)= 2+2 = 4

LDA:

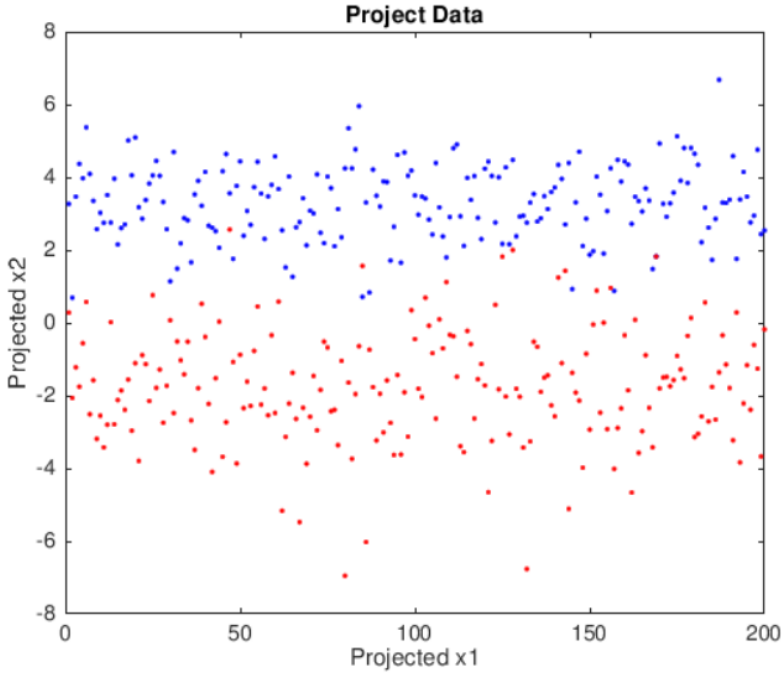
Linear Discriminant Analysis is a similar and often less accurate technique that generates some boundary that maximizes the distance between the projected means normalized by within class scatter. What this effectively means is that the projected samples from the same class are put as close as possible together while the means are as far apart as can be. All in all this optimization function simple is the maximization of the ratio of each data sets Between Class Scatter and Within Class Scatter.

Mathematically speaking, this means computing the associated eigenpairs associated with the following equation: $S_w^{-1}*S_Bw=\lambda w$, where S_w^{-1} is the inverted within scatter values of the unprojected data for a certain class, and S_B is the between class scatter. Note that the eigenvector with the largest eigenvalue is will maximize the optimization technique. In this code sequence the following pseudo technique was developed: (Images of the coded sections are shown)

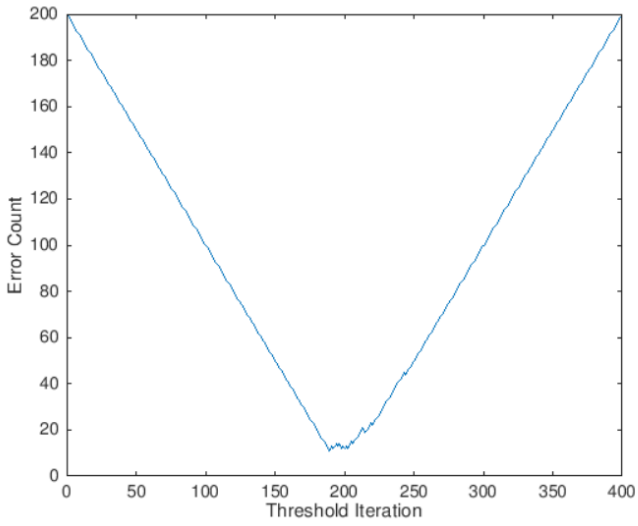
1. Let A and B some 100 samples from a multivariate Gaussian Distribution.



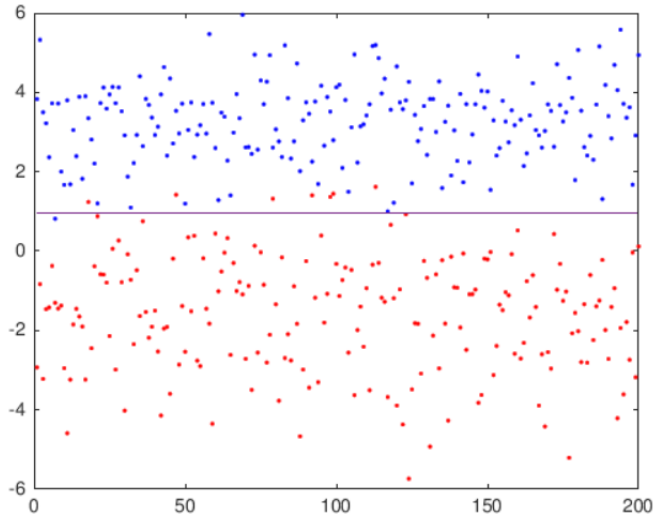
2. (Within Scatter) = S_w= sigma1+sigma2
3. (Between Scatter)= S_B= (mu1-mu2)*(mu1-mu2)^T
4. Find max eigenvalue, and associated eigenvector for S_w^{-1}
    a. W= eigenvector resulting from max eigenvalue
5. Projection of multivariate data
    a. Y = W^TA
    b. Z = W^TB



6. Error formulae (With respect to some threshold t)=
    a. #Total Number of Sample of Class A -#Number of samples A on correct side of threshold) + (#Total Number of Sample of Class B -#Number of samples B on correct side of threshold)
        i. (A_{TOTAL}-(#A>t))+ (B_{TOTAL}-(#B<t)), where # represents count of
    b. = #Incorrectly Classified A + #Incorrectly classified B



7. Determine threshold set as midpoints of plot and find threshold that has the lowest threshold count.



In this instance the minimum error count is 8, and the line above is a projection of the best threshold that minimizes error.

Looking at both instances of discriminant analysis, QDA performed better in this specific case. After running several trials QDA has proved to general perform better than LDA, however at times each model was relatively comparable. However, each model uses a different minimization technique where QDA is largely based on the MAP which include an impact from not only the distribution but also its prior. On the other hand, LDA computes a maximization function for the ration of between and within class scatter for each distribution. As a result, it creates a linear discriminant that exists with the lowest error for each class classification. Looking at both models, QDA has more flexibility for the covariance of each model than LDA, however is more computationally more expensive because it has to estimate a larger number of parameters. Therefore, LDA is a better model in the case of multiple classes and fewer data points, where for many classes, QDA will have to deal with many covariance's which will be a problem.

Question 5:

1.) $argmax\ P\left(\begin{pmatrix}x\\y\end{pmatrix}\middle|r\right)$ The following is the function that ought to be maximized which is the probability given some data r which is a rough approximation of sensor distance (true sensor distance + noise), one can locate the true location of the car at $\begin{pmatrix}x\\y\end{pmatrix}$

2.) $argmax\ P\left(\begin{pmatrix}x\\y\end{pmatrix}\middle|r\right) = argmax\ \prod P\left(ri\middle|\begin{pmatrix}x\\y\end{pmatrix}\right) * P\left(\begin{pmatrix}x\\y\end{pmatrix}\right) = argmax\ \sum lnP\left(ri\middle|\begin{pmatrix}x\\y\end{pmatrix}\right) + lnP\left(\begin{pmatrix}x\\y\end{pmatrix}\right)$ (Solution is sum of log likelihood and log prior which combine to be the Maximum A Posterior using Bayes Theorem. Note that the evidence (P(r)) is excluded since is it is not relevant in the maximization of the function.

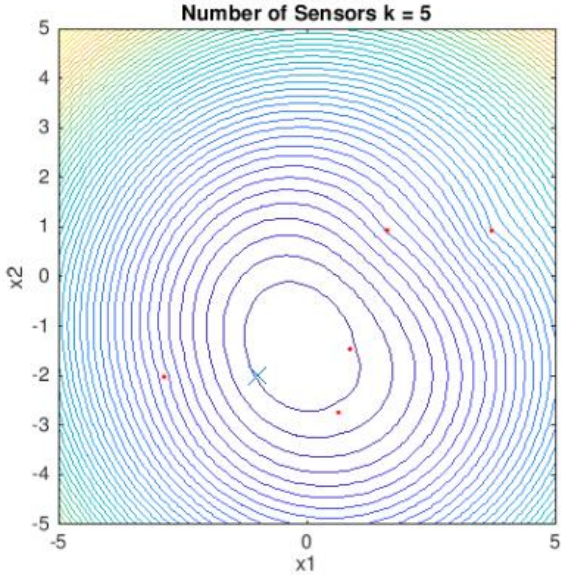
3.) $= argmax\ \sum ln\frac{1}{\frac{1}{(2\pi)^2}\sigma_i} e^{-\frac{\frac{1}{2}(ri-d)^2}{\sigma_i^2}} + ln\frac{1}{2\pi\sigma_x\sigma_y} e^{-\frac{1}{2}[x\ \ y]\begin{bmatrix}\sigma_x^2 & 0\\0 & \sigma_y^2\end{bmatrix}^{-1}\begin{bmatrix}x\\y\end{bmatrix}} = argmax \sum\ ln\frac{1}{\frac{1}{(2\pi)^2}\sigma_i} - \frac{\frac{1}{2}(ri-d)^2}{\sigma_i^2} + ln\frac{1}{2\pi\sigma_x\sigma_y} - \frac{1}{2}[x\ \ y]\begin{bmatrix}\sigma_x^2 & 0\\0 & \sigma_y^2\end{bmatrix}^{-1}\begin{bmatrix}x\\y\end{bmatrix}$. This step involves the substitution of the distributions. The prior distribution is provided in the problem. However the likelihood or the probability of r given the true coordinates of the car x,y can be inferred. This is because r = d+n where d is the Euclidean distance between each reference sensor and its associated point, and n is a Gaussian noise model with a mean of 0 and a specified variance. Therefore the distribution of the likelihood is a mean centered at d, the Euclidean distance, under the same variance as the noise model.

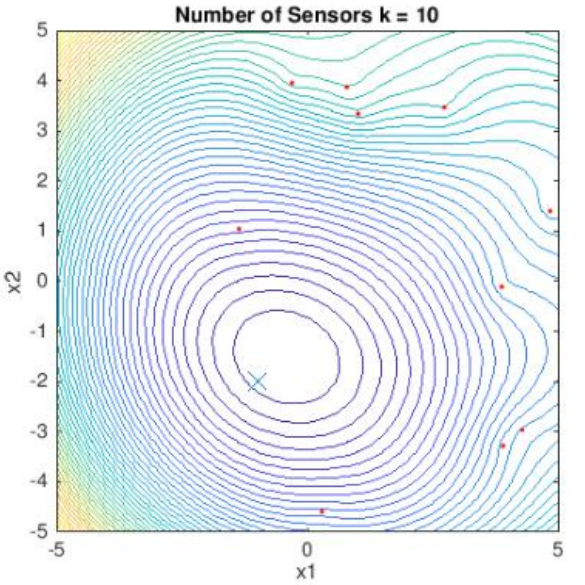
4.) $argmax \sum -\frac{\frac{1}{2}(ri-d)^2}{\sigma_i^2} - \frac{1}{2}[x\ \ y]\begin{bmatrix}\sigma_x^2 & 0\\0 & \sigma_y^2\end{bmatrix}^{-1}\begin{bmatrix}x\\y\end{bmatrix}$. Constants in the equation are removed. The solution is then multiplied by -2 for further simplification. The negative multiplication results in a minimization problem now.

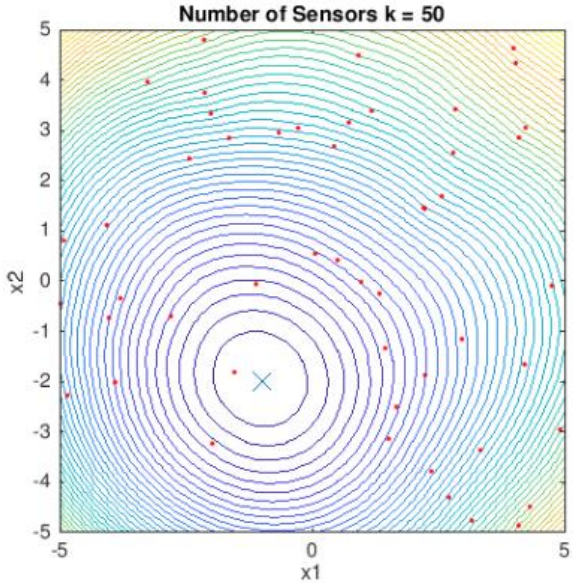
5.) $argmin \sum \frac{(r_i - d(x,y))^2}{\sigma_i^2} + \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}$. As explained before, the final solution is to incorporate the Euclidean distance formula.

6.) $argmin \sum \frac{\left(r - \sqrt{(x_i-x)^2+(y_i-y)^2}\right)^2}{\sigma_i^2} + \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}$.

Below are a series of plots having an x, the true (unknown location of the car), and a contour of the optimization function above such that its center minimizes the function described above. The red dots represent the number and location of the references.







It is clear to see with each trail that as the number of sensors (k) increases, the function better approximates the true location in the minima of the contour function. This makes more sense since an increase of sensors increases the likelihood formula where the log-likelihoods are summed. Therefore a better representation of the center of the plot can be determined. It is important to note that the prior in the MAP estimate pulls the contours towards origin since the prior itself has a 0 centered mean.

Question 6:

For GMM where M = 3 (# number distributions)

Proof of Gaussian Mixture Models and EM ALGO:

Let z be a distribution made of 3 other Gaussians such that Z~multinomial(π1, π2, π3), where π is the probability of picking a sample from a Gaussians 1,2, or 3. When using EM algorithm, if the Gaussian distributions are not known for a given GMM, there are latent variables z. The GMM is made of *observables* which are from a distribution where in P(x|z)*P(z=k), P(z=k) is π_k. Therefore, we define a GMM as:

$$GMM = \sum_{k=1}^{K} \pi_k * P(x|z=k), \sum \pi_k * Gaussian(\mu_k, \Sigma_k) \; where \; in \; the \; case \; of \; 3 \; distributions:$$

$$P(x) = \pi_1 * P(x|z=1) + \pi_2 * P(x|z=2) + \pi_3 * P(x|z=3), where \; P(x) \; is \; a \; probailistic \; weighted \; average$$

(Convex Combination)

$$\pi_k = \frac{N_k}{N} = \frac{\#\ of\ times\ k\ distribution\ appears}{\#\ observations}$$

EM Algorithm

Expectation Step: (Re)Assign Probabilities and Classify (K)

Computation of the responsibilities which represents how strong a data point belongs to a component class distribution (1,2,3).

$$r_{k=1} = P(z=k=1|x) = \frac{\pi_1 P(x|z=1)}{\pi_1 P(x|z=1) + \pi_2 P(x|z=2) + \pi_2 P(x|z=3)}$$

$$r_k = \frac{\pi_k P(x|z=k)}{\sum \pi_{k'} P(x|z=k')} \sim \pi_k P(x|z=k) \; (Normalization \; Constant \; is \; the \; same)$$

Pseudo Code: For every value of x and y, determine π_1, π_2, π_3 such that the highest probability given a guessed mu and sigma for each value will determine the class for that iteration. More specifically

$$\pi_{k\ iteration(i)} = \pi_{k\ guess} * Gaussian\ (x, \mu_{k\ guess}, \Sigma_{k\ guess}) * Gaussian\ (y, \mu_{k\ guess}, \Sigma_{k\ guess})$$

Class k for a given {x, y} is k for max $[\pi_1, \pi_2, \pi_3]$: π_k

Maximization Step: Compute and Update Parameters for new Guessed Classes

Apply the maximum likelihood estimate update such that new parameters replace old ones.

$$argmax \sum r_k * ln\ Gaussian\ (x, \mu_k, \Sigma_k) = argmax \sum r_k * ln\ P(x|z=k) = argmax \sum \frac{\pi_k P(x|z=k)}{\sum \pi_{k'} P(x|z=k')} * ln\ P(x|z=k)$$

$$\mu_{k\ MLE} = \frac{1}{\sum \frac{\pi_k P(x|z=k)}{\sum \pi_{k'} P(x|z=k')}} \sum \frac{\pi_k P(x|z=k)}{\sum \pi_{k'} P(x|z=k')} * x$$

$$\Sigma_{k\ MLE} = \frac{1}{\sum \frac{\pi_k P(x|z=k)}{\sum \pi_{k'} P(x|z=k')}} \sum \frac{\pi_k P(x|z=k)}{\sum \pi_{k'} P(x|z=k')} * (x - \mu_{k\ MLE})\text{^}2$$

Pseudo Code:

$$\frac{\#values\ for\ Guessed\ Class\ k}{Total\ Number\ of\ Values} = \pi_k$$

$$\mu_{k\ guess} = [\ mean(\#\text{x}\ values\ for\ Guessed\ Class\ k), mean(\#\text{y}\ values\ for\ Guessed\ Class\ k) = [\mu_{kx}\ \ \mu_{ky}\ ]$$

$$\Sigma_{k\ guess} = [std(\#values\ for\ Guessed\ Class\ n)\ 0, 0\ std(\#values\ for\ Guessed\ Class\ n)] = \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix}$$

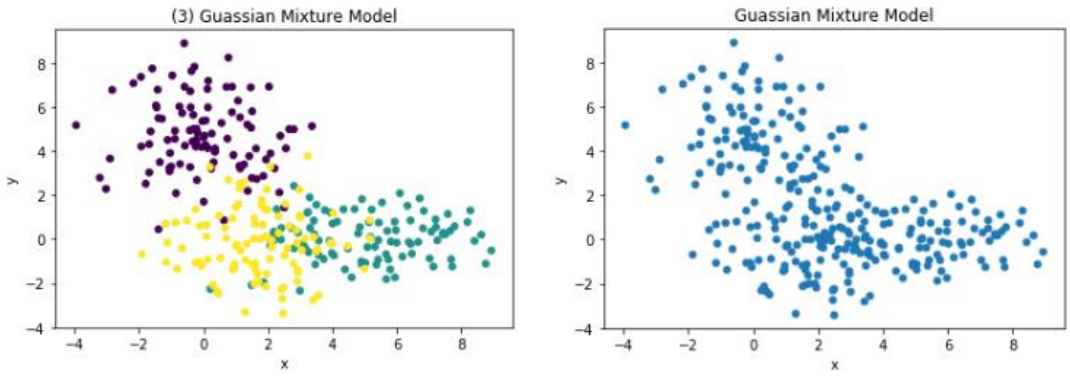
Main Basis: While the distance between old parameters and update parameters > threshold, the algorithm will continue updating:

While $(\text{sum}(\text{oldparams} - \text{updatedparams})^2)^{.5} > \text{Threshold}$

1.) Update Labels/Classes (E-Step)
2.) MLE of newly labelled data and update parameters (M-Step)

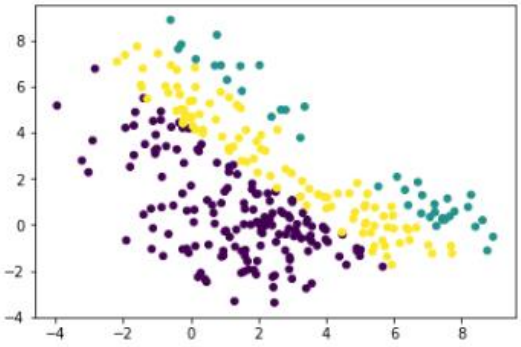
Part A:

Below is a Gaussian mixture model of M=3. The right shows the plots that make up the Gaussian and the right shows the model that is fed into the EM Algorithm.

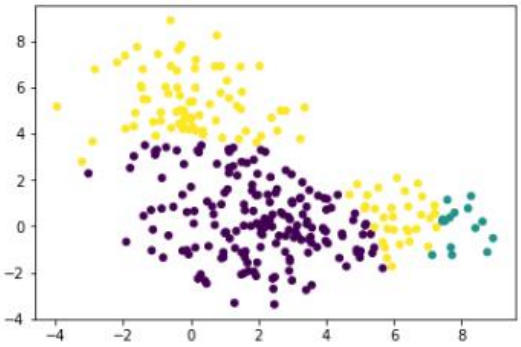


Part B:

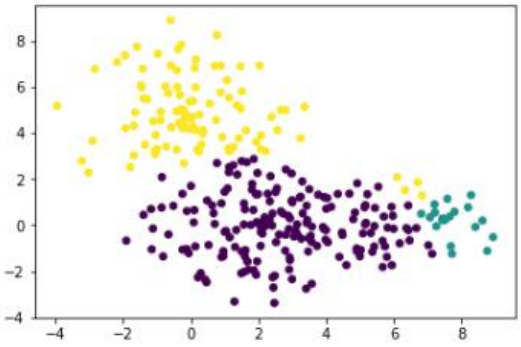
Given some guesses for the distribution means, after three iterations one can see how well the model performs:



Iteration 1



Iteration 2



Iteration 3

Comparing the models:

True Params:

mu1 = [0, 5]

sig1 = [2, 0; 0, 3]

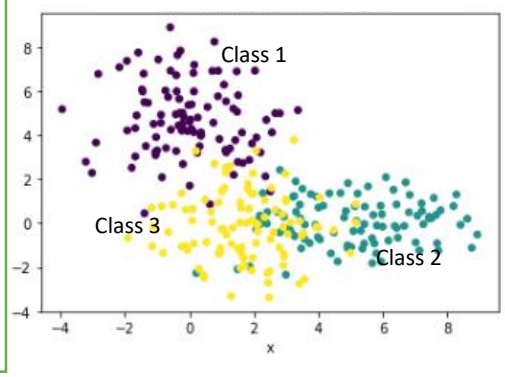
mu2 = [5, 0]

sig2 = [4, 0; 0, 1]

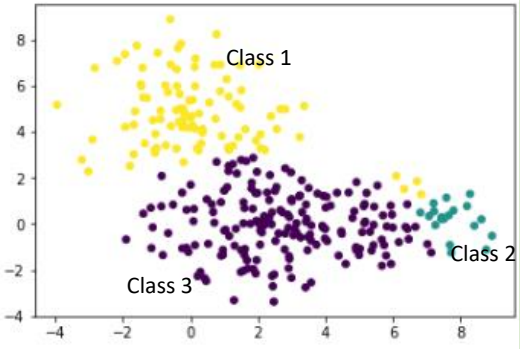
mu3 = [2, 0]

sig3 = [2, 0; 0, 2]

True Distribution



EM Approximated Distribution



EM Params:

mu1 = [.37, 4.74]

sig1 = [1.97, 0; 0, 1.52]

mu2 = [7.79, -.081]

sig2 = [1.07, 0; 0, .986]

mu3 = [2.82, -.019]

sig3 = [1.67, 0; 0, 1.247]

Looking at the true results in comparison, the EM Algorithm has performed relatively well in locating the three distributions however, has struggled with splitting classes 2 and 3. Class 2 is much larges in the model and this makes sense for an error to most likely occur due to the fact that the classes 2 and 3 intersect more strongly than class 1. Likewise, making class 2 larger in the approximation has thus made class 3 smaller. Note that the lambda values are representation of pi or the percentage mixture of each distribution in the GMM. The EM model shows a higher contribution from class 3 as opposed to class 1.

Below is a log of iterations throughout EM Algo:

```
iteration 1, shift 2.2058632680307917
        mu1                       sig1       mu2                       sig2
0  1.458051  [1.8582185424224675, 0]  5.559697  [3.1455886188022792, 0]
1  0.599416   [0, 2.105444814189806]  2.638507   [0, 3.064044851672928]
2  0.000000                       0  0.000000                       0

        mu3                       sig3    lambda
0  2.712905  [2.5493306323094758, 0]  0.513333
1  2.643890  [0, 2.5177383146617967]  0.143333
2  0.000000                       0  0.343333
iteration 2, shift 3.976982220370052
        mu1                       sig1       mu2                       sig2
0  2.108270  [1.5170011907442302, 0]  7.907754  [1.0272706648840335, 0]
1  0.328107  [0, 1.6022251846942723] -0.007450  [0, 1.0015009022304535]
2  0.000000                       0  0.000000                       0

        mu3                       sig3    lambda
0  1.678074  [2.8922700679600375, 0]  0.553333
1  3.960953  [0, 2.4715950920061096]  0.090000
2  0.000000                       0  0.356667
iteration 3, shift 1.720032833518873
        mu1                       sig1       mu2                       sig2
0  2.818827  [1.6736121291253587, 0]  7.795966  [1.0735055165727552, 0]
1 -0.019387  [0, 1.2473962319306153]  0.081247  [0, 0.9864993997407729]
2  0.000000                       0  0.000000                       0

        mu3                       sig3    lambda
0  0.373786  [1.9705896942446681, 0]  0.566667
1  4.742835   [0, 1.521135084723379]  0.096667
2  0.000000                       0  0.336667
```