ECE 595: Homework 3

Rahul Deshmukh, Class ID: 58

(Spring 2019)

Exercise 1

a Proof of MLE of Likelihood function

i To prove: $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = tr[\boldsymbol{A}\boldsymbol{x}\boldsymbol{x}^T]$

We will use the definition of trace: $tr(\boldsymbol{AB}) = A_{ij}B_{ji}$

Proof:

$$\begin{aligned}
\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} &= x_i A_{ij} x_j \\
&= A_{ij} x_j x_i \\
&= tr[\boldsymbol{A}(\boldsymbol{x} \otimes \boldsymbol{x})] \\
&= tr[\boldsymbol{A}\boldsymbol{x}\boldsymbol{x}^T] \\
&\text{hence proved}
\end{aligned}$$

ii Simplification of likelihood function:
Now we will be using the identity proved in part (i)

$$\begin{aligned}
p(\boldsymbol{x}_1, \boldsymbol{x}_2, ...\boldsymbol{x}_N|\Sigma) &= \prod_{n=1}^{N} \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\{-\frac{1}{2}(x_n - \mu)^T \Sigma^{-1}(x_n - \mu)\} \\
&= \prod_{n=1}^{N} \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\{-\frac{1}{2} tr[\Sigma^{-1}(x_n - \mu)(x_n - \mu)^T]\} \\
&= \frac{1}{(2\pi)^{Nd/2}} |\Sigma|^{-N/2} \exp\{-\frac{1}{2}\sum_{n=1}^{N} tr[\Sigma^{-1}(x_n - \mu)(x_n - \mu)^T]\} \\
&= \frac{1}{(2\pi)^{Nd/2}} |\Sigma^{-1}|^{N/2} \exp\{-\frac{1}{2}\sum_{n=1}^{N} tr[\Sigma^{-1}(x_n - \mu)(x_n - \mu)^T]\} \\
&= \frac{1}{(2\pi)^{Nd/2}} |\Sigma^{-1}|^{N/2} \exp\{-\frac{1}{2} tr[\Sigma^{-1}\sum_{n=1}^{N}(x_n - \mu)(x_n - \mu)^T]\}
\end{aligned}$$

For the last step in the above proof we are using the simplification that:

$$
\begin{aligned}
\sum_{n=1}^{N} tr[\Sigma^{-1}(x_n-\mu)(x_n-\mu)^T] &= \sum_{n=1}^{N} \Sigma_{ij}^{-1}(\Sigma_n)_{ij} \\
&= \Sigma_{ij}^{-1}\sum_{n=1}^{N}(\Sigma_n)_{ij} \\
&= tr[\Sigma^{-1}\sum_{n=1}^{N}(x_n-\mu)(x_n-\mu)^T]
\end{aligned}
$$

Also using the identity:

$$|A^{-1}| = |A|^{-1}$$

iii Given $A = \Sigma^{-1}\hat{\Sigma}_{MLE}$ and $\lambda_1, \lambda_2, ...\lambda_d$ are the eigenvalues of A

We will also be using the following properties:

$$
\begin{aligned}
|A| = |\Sigma^{-1}||\hat{\Sigma}_{MLE}| &= \prod_{i=1}^{d}\lambda_i \\
tr[A] &= \sum_{i=1}^{d}\lambda_i \\
tr[cA] &= c * tr[A] \text{ where c is a constant} \\
\hat{\Sigma}_{MLE} &= \frac{1}{N}\sum_{n=1} N(x_n-\mu)(x_n-\mu)^T
\end{aligned}
$$

Proof:

$$
\begin{aligned}
p(\boldsymbol{x}_1, \boldsymbol{x}_2, ...\boldsymbol{x}_N|\Sigma) &= \frac{1}{(2\pi)^{Nd/2}}|\Sigma^{-1}|^{N/2}\exp\{-\frac{1}{2}tr[\Sigma^{-1}\sum_{n=1}^{N}(x_n-\mu)(x_n-\mu)^T]\} \\
&= \frac{1}{(2\pi)^{Nd/2}}\Big(\frac{\prod_{i=1}^{d}\lambda_i}{|\hat{\Sigma}_{MLE}|}\Big)^{N/2}\exp\{-\frac{1}{2}tr[\Sigma^{-1}(N\hat{\Sigma}_{MLE})]\} \\
&= \frac{1}{(2\pi)^{Nd/2}}\Big(\frac{\prod_{i=1}^{d}\lambda_i}{|\hat{\Sigma}_{MLE}|}\Big)^{N/2}\exp\{-\frac{N}{2}tr[\Sigma^{-1}(\hat{\Sigma}_{MLE})]\} \\
&= \frac{1}{(2\pi)^{Nd/2}}\Big(\frac{\prod_{i=1}^{d}\lambda_i}{|\hat{\Sigma}_{MLE}|}\Big)^{N/2}\exp\{-\frac{N}{2}tr[A]\} \\
&= \frac{1}{(2\pi)^{Nd/2}}\Big(\frac{\prod_{i=1}^{d}\lambda_i}{|\hat{\Sigma}_{MLE}|}\Big)^{N/2}\exp\{-\frac{N}{2}\sum_{i=1}^{d}\lambda_i\}
\end{aligned}
$$

hence proved

iv Maximization of likelihood function:
Let us take the log() of the likelihood function and equate its gradient wrt λ to 0.

$$\begin{aligned} log(p(\mathcal{D}|\Sigma)) &= \frac{N}{2}(\sum_{i=1}^{d} log(\lambda_i)) - \frac{N}{2}(\sum_{i=1}^{d} \lambda_i) + \frac{N}{2}(log((2\pi)^d|\hat{\Sigma}_{MLE}|)) \\ \frac{\partial log(p(\mathcal{D}|\Sigma))}{\partial \lambda_j} &= \frac{N}{2}(\frac{1}{\lambda_j} - 1) = 0 \\ \Rightarrow \lambda_j &= 1 \end{aligned}$$

Therefore $\lambda_j = 1 \forall j$ maximizes the likelihood function. This would mean that all the eigenvalues of $A = \Sigma^{-1}\hat{\Sigma}_{MLE}$ are the same and thus we can choose the standard basis as our set of eigenvectors i.e.

$$\begin{aligned} \Rightarrow A &= I \\ \Sigma^{-1}\hat{\Sigma}_{MLE} &= I \\ \Sigma &= \hat{\Sigma}_{MLE} \end{aligned}$$

Thus $\hat{\Sigma}_{MLE}$ maximizes the likelihood function.

b Proof by directly taking a matrix derivative:

We will be using the following identities:

$$\begin{aligned} \frac{\partial tr[AX]}{\partial X} &= \frac{\partial tr[XA]}{\partial X} = A \\ \frac{\partial |X|}{\partial X} &= |X|X^{-1} \end{aligned}$$

Let us start with the result derived in part (ii) and replace Σ^{-1} by P:

$$\begin{aligned}
p(\mathcal{D}|\Sigma) &= \frac{1}{(2\pi)^{Nd/2}}|\Sigma^{-1}|^{N/2}\exp\{-\frac{1}{2}tr[\Sigma^{-1}\sum_{n=1}^{N}(x_n-\mu)(x_n-\mu)^T]\} \\
&= \frac{1}{(2\pi)^{Nd/2}}|P|^{N/2}\exp\{-\frac{N}{2}tr[P\hat{\Sigma}_{MLE}]\} \\
\mathbf{0} &= \frac{\partial p(\mathcal{D}|P)}{\partial P} \\
\mathbf{0} &= \frac{\partial}{\partial P}(\frac{1}{(2\pi)^{Nd/2}}|P|^{N/2}\exp\{-\frac{N}{2}tr[P\hat{\Sigma}_{MLE}]) \\
\mathbf{0} &= (\frac{\partial|P|^{N/2}}{\partial P}+|P|^{N/2}\frac{\partial(-\frac{N}{2}tr[P\hat{\Sigma}_{MLE}])}{\partial P})\exp\{-\frac{N}{2}tr[P\hat{\Sigma}_{MLE}]\} \\
\mathbf{0} &= \frac{N}{2}|P|^{N/2-1}*(|P|P^{-1})+|P|^{N/2}(-\frac{N}{2}*\hat{\Sigma}_{MLE}) \\
\mathbf{0} &= |P|^{N/2}*(P^{-1}-\hat{\Sigma}_{MLE}) \\
\text{as}|P| &\neq 0 \\
P^{-1} &= \hat{\Sigma}_{MLE} \\
P^{-1} = (\Sigma^{-1})^{-1} = \Sigma &= \hat{\Sigma}_{MLE}
\end{aligned}$$

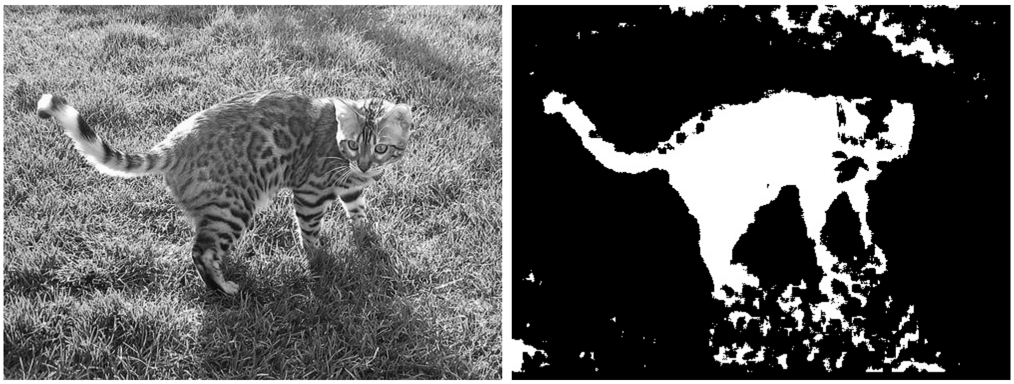
Therefore $\hat{\Sigma}_{MLE}$ maximizes the likelihood function.

Exercise 2

a Please refer to code on page 14

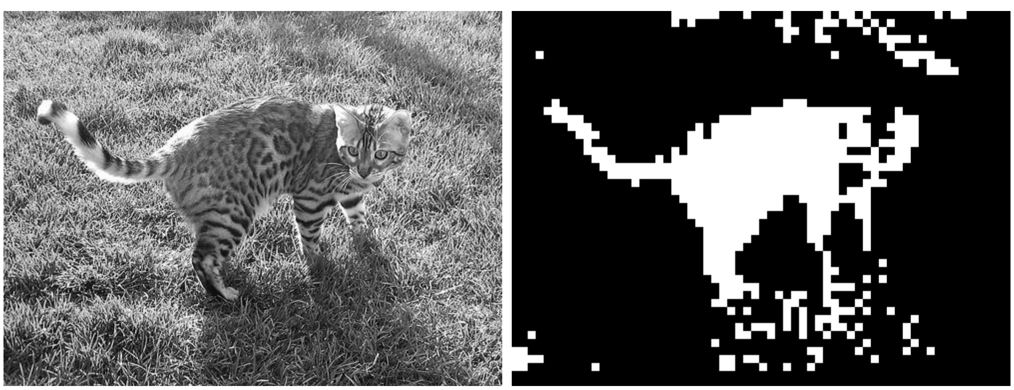
b (i)Please refer to code on page 14

ii Please refer to code on page 14



(a) original image (b) Output for overlapping patches

iii Please refer to code on page 14



(c) original image (d) Output for non overlapping patches

(iv) Please refer to code on page 14

The mean absolute errors reported for the two cases are:

```
Error for overlapping: 8.767922614379087%
and for non-overlapping:8.428352418300653%
```

(v) I had downloaded a picture of cheetah on grass from internet for testing our classifier. The discussion follows:

(e) original image

(f) Output for overlapping patches

(g) Output for non overlapping patches

As we can observe from the above results for the testing image, our classifier performs poorly. This is because:

- The parameters $\mu_1, \Sigma_1, \mu_2, \Sigma_2, \pi_1, \pi_2$ are computed from data only using MLE. Therefore, any testing data which does not conform to these parameters will have poor classification.

- Feature vector does not have enough information, alternatively we could use feature vectors like LBP(local binary pattern) which contain more information.

Exercise 3: Connection to Linear-Least Squares

a Expression for the decision boundary of linear Gaussian classifier:

$$g(x) = 0 \forall \{x \in \mathbb{R}^d | \beta^T x + \beta_0 = 0\}$$

Where,

$$\beta = \hat{\Sigma}^{-1}(\mu_1 - \mu_2) \qquad \beta_0 = -\frac{1}{2}\left(\mu_1^T \hat{\Sigma}^{-1}\mu_1 - \mu_2^T \hat{\Sigma}^{-1}\mu_2\right) + log\frac{\pi_1}{\pi_2}$$

b We are given the following:
We have a dataset with two classes $\mathcal{D}_1 = \{(x_i^{(1)}, y_i^{(1)})\}_{i=1}^{N_1}$ and $\mathcal{D}_2 = \{(x_i^{(2)}, y_i^{(2)})\}_{i=1}^{N_2}$ with $x_i^{(j)} \in \mathbb{R}^d$ and $y_i^{(j)} \in \mathbb{R}$ and $N = N_1 + N_2$

$$\hat{\Sigma} = \frac{1}{N_1 + N_2 - 2}\sum_{j=1}^{2}\sum_{i=1}^{N_j}(x_i^{(j)} - \hat{\mu}_j)(x_i^{(j)} - \hat{\mu}_j)^T$$

$$\hat{\mu}_1 = \frac{1}{N_1}\sum_{i=1}^{N_1} x_i^{(1)}$$

$$\hat{\mu}_2 = \frac{1}{N_2}\sum_{i=1}^{N_2} x_i^{(2)}$$

$$A^T A \begin{bmatrix} w \\ w_0 \end{bmatrix} = A^T b$$

if we recall from previous homework A and b are given by

$$A = \begin{bmatrix} X_1^T & 1 \\ X_2^T & 1 \end{bmatrix} \qquad b = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$$

where X_1^T are the feature vectors for class 1 in the form

$$X1_{(i)}^T = [x_1^{(1)}{}_{(i)}, x_2^{(1)}{}_{(i)}, ...x_d^{(1)}{}_{(i)}] \text{ for the i'th sample}$$

and Y_1 is the vector of class labels for class 1 in the form $Y_1^T = [c1, c1, c1, ...c1]$

(i) To prove:

$$A^T A = \begin{bmatrix} (N-2)\hat{\Sigma} + N_1\hat{\mu_1}\hat{\mu_1}^T + N_2\hat{\mu_2}\hat{\mu_2}^T & N_1\hat{\mu_1} + N_2\hat{\mu_2} \\ N_1\hat{\mu_1}^T + N_2\hat{\mu_2}^T & N \end{bmatrix} \tag{1}$$

Proof:
Lets simplify $A^T A$

$$
\begin{aligned}
A^T A &= \begin{bmatrix} X_1^T & 1 \\ X_2^T & 1 \end{bmatrix}^T \begin{bmatrix} X_1^T & 1 \\ X_2^T & 1 \end{bmatrix} \\
&= \begin{bmatrix} X_1 & X_2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} X_1^T & 1 \\ X_2^T & 1 \end{bmatrix} \\
&= \begin{bmatrix} X_1 X_1^T + X_2 X_2^T & X_1 + X_2 \\ X_1^T + X_2^T & N_1 + N_2 \end{bmatrix} \\
A^T b &= \begin{bmatrix} X_1 & X_2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} \\
&= \begin{bmatrix} X_1.Y_1 + X_2.Y_2 \\ 1.Y_1 + 1.Y_2 \end{bmatrix} \\
&= \begin{bmatrix} X_1.(c_1\mathbf{1}) + X_2.(c_2\mathbf{1}) \\ 1.(c_1\mathbf{1}) + 1.(c_2\mathbf{1}) \end{bmatrix} \qquad = \begin{bmatrix} c_1 \sum_{i=1}^{N_1} x_i^{(1)} + c_2 \sum_{i=1}^{N_2} x_i^{(2)} \\ c_1 N_1 + c_2 N_2 \end{bmatrix}
\end{aligned}
$$

Using equation (1) and simplifying the first element:

$$
\begin{aligned}
[A^T A]_{1,1} &= (N-2)\hat{\Sigma} + N_1 \hat{\mu}_1 \hat{\mu}_1{}^T + N_2 \hat{\mu}_2 \hat{\mu}_2{}^T \\
&= \sum_{j=1}^{2} \sum_{i=1}^{N_j} (x_i^{(j)} - \hat{\mu}_j)(x_i^{(j)} - \hat{\mu}_j)^T + N_1 \hat{\mu}_1 \hat{\mu}_1{}^T + N_2 \hat{\mu}_2 \hat{\mu}_2{}^T \\
&= \sum_{j=1}^{2} \sum_{i=1}^{N_j} \left(x_i^{(j)} x_i^{(j)T} - \hat{\mu}_j x_i^{(j)T} - x_i^{(j)} \hat{\mu}_j^T + \hat{\mu}_j \hat{\mu}_j^T \right) + N_1 \hat{\mu}_1 \hat{\mu}_1{}^T + N_2 \hat{\mu}_2 \hat{\mu}_2{}^T \\
&= \sum_{j=1}^{2} \sum_{i=1}^{N_j} \left(x_i^{(j)} x_i^{(j)T} \right) + \sum_{j=1}^{2} \left(-(N_j \hat{\mu}_j \hat{\mu}_j^T) - (N_j \hat{\mu}_j \hat{\mu}_j^T) + N_j \hat{\mu}_j \hat{\mu}_j^T \right) + N_1 \hat{\mu}_1 \hat{\mu}_1{}^T + N_2 \hat{\mu}_2 \hat{\mu}_2{}^T \\
&= \sum_{j=1}^{2} \sum_{i=1}^{N_j} \left(x_i^{(j)} x_i^{(j)T} \right) \\
&= X_1 X_1^T + X_2 X_2^T \\
[A^T A]_{1,2} &= N_1 \hat{\mu}_1 + N_2 \hat{\mu}_2 \\
&= \sum_{i=1}^{N_1} (x_i^{(1)}) + \sum_{i=1}^{N_2} (x_i^{(2)}) \\
&= X_1 + X_2 \\
\text{Similarly } [A^T A]_{2,1} &= X_1^T + X_2^T \\
\text{And} [A^T A]_{2,2} &= N = N_1 + N_2 \\
\text{hence proved} &
\end{aligned}
$$

(ii) To prove:

$$
A^T b = \begin{bmatrix} c_1 N_1 \hat{\mu}_1 + c_2 N_2 \hat{\mu}_2 \\ N_1 c_1 + N_2 c_2 \end{bmatrix}
$$

Proof:

$$
\begin{aligned}
[A^T b]_1 &= c_1 N_1 \hat{\mu}_1 + c_2 N_2 \hat{\mu}_2 \\
&= c_1 \sum_{i=1}^{N_1} x_i^{(1)} + c_2 \sum_{i=1}^{N_2} x_i^{(2)} \\
&\text{hence proved}
\end{aligned}
$$

(iii) We have:

$$
A^T A \begin{bmatrix} w \\ w_0 \end{bmatrix} = A^T b
$$

$$
\begin{bmatrix} (N-2)\hat{\Sigma} + N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T & N_1 \hat{\mu_1} + N_2 \hat{\mu_2} \\ N_1 \hat{\mu_1}^T + N_2 \hat{\mu_2}^T & N \end{bmatrix} \begin{bmatrix} w \\ w_0 \end{bmatrix} = \begin{bmatrix} c_1 N_1 \hat{\mu}_1 + c_2 N_2 \hat{\mu}_2 \\ N_1 c_1 + N_2 c_2 \end{bmatrix}
$$

$$
\begin{aligned}
((N-2)\hat{\Sigma} + N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T) w + (N_1 \hat{\mu_1} + N_2 \hat{\mu_2}) w_0 &= c_1 N_1 \hat{\mu}_1 + c_2 N_2 \hat{\mu}_2 N_1 c_1 + N_2 c_2 \\
(N_1 \hat{\mu_1}^T + N_2 \hat{\mu_2}^T) w + N w_0 &= N_1 c_1 + N_2 c_2
\end{aligned}
$$

$$
\Rightarrow w_0 = \frac{N_1 c_1 + N_2 c_2}{N} - \frac{(N_1 \hat{\mu_1}^T + N_2 \hat{\mu_2}^T) w}{N}
$$

$$
\begin{aligned}
\Rightarrow ((N-2)\hat{\Sigma} + N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T) w + (N_1 \hat{\mu_1} + N_2 \hat{\mu_2}) \Big(\frac{N_1 c_1 + N_2 c_2}{N} - \frac{(N_1 \hat{\mu_1}^T + N_2 \hat{\mu_2}^T) w}{N} \Big) \\
= c_1 N_1 \hat{\mu}_1 + c_2 N_2 \hat{\mu}_2 \\
\text{hence proved}
\end{aligned}
$$

(iv) Now we need to simplify the LHS:

$$
LHS = ((N-2)\hat{\Sigma} + N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T) w + (N_1 \hat{\mu_1} + N_2 \hat{\mu_2}) \Big(\frac{N_1 c_1 + N_2 c_2}{N} - \frac{(N_1 \hat{\mu_1}^T + N_2 \hat{\mu_2}^T) w}{N} \Big)
$$

We need to prove the following:

$$
N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T - (N_1 \hat{\mu_1} + N_2 \hat{\mu_2}) \frac{(N_1 \hat{\mu_1}^T + N_2 \hat{\mu_2}^T)}{N} = \frac{N_1 N_2}{N} (\hat{\mu}_2 - \hat{\mu}_1)(\hat{\mu}_2 - \hat{\mu}_1)^T
$$

$$
\begin{aligned}
LHS &= N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T - (N_1 \hat{\mu_1} + N_2 \hat{\mu_2}) \frac{(N_1 \hat{\mu_1}^T + N_2 \hat{\mu_2}^T)}{N} \\
&= N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T - \frac{N_1^2 \hat{\mu}_1 \hat{\mu}_1^T + N_1 N_2 \hat{\mu}_1 \hat{\mu}_2^T + N_2 N_1 \hat{\mu}_2 \hat{\mu}_1^T + N_2^2 \hat{\mu}_2 \hat{\mu}_2^T}{N1 + N_2} \\
&= \frac{(N_1 + N_2)(N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T) - N_1^2 \hat{\mu}_1 \hat{\mu}_1^T - N_1 N_2 \hat{\mu}_1 \hat{\mu}_2^T - N_2 N_1 \hat{\mu}_2 \hat{\mu}_1^T - N_2^2 \hat{\mu}_2 \hat{\mu}_2^T}{N1 + N_2} \\
&= \frac{N_1 N_2 \hat{\mu_2} \hat{\mu_2}^T + N_2 N_1 \hat{\mu_1} \hat{\mu_1}^T - N_1 N_2 \hat{\mu}_1 \hat{\mu}_2^T - N_2 N_1 \hat{\mu}_2 \hat{\mu}_1^T}{N1 + N_2} \\
&= \frac{N_1 N_2 (\hat{\mu_2} \hat{\mu_2}^T + \hat{\mu_1} \hat{\mu_1}^T - \hat{\mu}_1 \hat{\mu}_2^T - \hat{\mu}_2 \hat{\mu}_1^T)}{N1 + N_2} = \frac{N_1 N_2 (\hat{\mu}_2 - \hat{\mu}_1)(\hat{\mu}_2 - \hat{\mu}_1)^T}{N}
\end{aligned}
$$

Therefore we have:

$$((N-2)\hat{\Sigma} + \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)(\hat{\mu}_2-\hat{\mu}_1)^T}{N})w + (N_1\hat{\mu}_1 + N_2\hat{\mu}_2)\Big(\frac{N_1c_1+N_2c_2}{N}\Big) = c_1N_1\hat{\mu}_1 + c_2N_2\hat{\mu}_2$$

(v) We need to make some further simplification

$$((N-2)\hat{\Sigma} + \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)(\hat{\mu}_2-\hat{\mu}_1)^T}{N})w + (N_1\hat{\mu}_1 + N_2\hat{\mu}_2)\Big(\frac{N_1c_1+N_2c_2}{N}\Big) = c_1N_1\hat{\mu}_1 + c_2N_2\hat{\mu}_2$$

$$\begin{aligned}
((N-2)\hat{\Sigma} + \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)(\hat{\mu}_2-\hat{\mu}_1)^T}{N})w &= c_1N_1\hat{\mu}_1 + c_2N_2\hat{\mu}_2 - (N_1\hat{\mu}_1 + N_2\hat{\mu}_2)\Big(\frac{N_1c_1+N_2c_2}{N_1+N_2}\Big) \\
&= \frac{(N_1+N_2)(c_1N_1\hat{\mu}_1 + c_2N_2\hat{\mu}_2) - (N_1\hat{\mu}_1 + N_2\hat{\mu}_2)(N_1c_1+N_2c_2)}{N} \\
&= \frac{+c_2N_1N_2\hat{\mu}_2 + c_1N_2N_1\hat{\mu}_1 - c_2N_1N_2\hat{\mu}_1 - c_1N_1N_2\hat{\mu}_2}{N} \\
&= \Big(\frac{+c_2N_1N_2 - c_1N_1N_2}{N}\Big)(\hat{\mu}_2-\hat{\mu}_1)
\end{aligned}$$

hence proved

c To prove: $\boldsymbol{w} = k\boldsymbol{\beta}$, $k \in \mathbb{R}$
Lets recall what was β

$$\beta = \hat{\Sigma}^{-1}(\mu_i - \mu_j)$$

Proof:

$$\begin{aligned}
((N-2)\hat{\Sigma} + \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)(\hat{\mu}_2-\hat{\mu}_1)^T}{N})w &= \Big(\frac{+c_2N_1N_2 - c_1N_1N_2}{N}\Big)(\hat{\mu}_2-\hat{\mu}_1) \\
(N-2)\hat{\Sigma}w + \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)(\hat{\mu}_2-\hat{\mu}_1)^T}{N}w &= \Big(\frac{+c_2N_1N_2 - c_1N_1N_2}{N}\Big)(\hat{\mu}_2-\hat{\mu}_1) \\
(N-2)\hat{\Sigma}w + \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)^Tw}{N}(\hat{\mu}_2-\hat{\mu}_1) &= \Big(\frac{+c_2N_1N_2 - c_1N_1N_2}{N}\Big)(\hat{\mu}_2-\hat{\mu}_1) \\
(N-2)\hat{\Sigma}w &= \Big(\frac{+c_2N_1N_2 - c_1N_1N_2}{N}\Big)(\hat{}_2-\hat{\mu}_1) - \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)^Tw}{N}(\hat{\mu}_2-\hat{\mu}_1) \\
(N-2)\hat{\Sigma}w &= \Big(\frac{+c_2N_1N_2 - c_1N_1N_2}{N} - \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)^Tw}{N}\Big)(\hat{\mu}_2-\hat{\mu}_1) \\
w &= \frac{1}{(N-2)}\Big(\frac{+c_2N_1N_2 - c_1N_1N_2}{N} - \frac{N_1N_2(\hat{\mu}_2-\hat{\mu}_1)^Tw}{N}\Big)\hat{\Sigma}^{-1}(\hat{\mu}_2-\hat{\mu}_1) \\
w &= k\hat{\Sigma}^{-1}(\hat{\mu}_2-\hat{\mu}_1) \\
w &= k\beta
\end{aligned}$$

hence proved

d Given: $c_1 = 1, c_2 = -1, N_1 = N_2$. To prove: There exists some scalar a such that $a\beta = w$ and $a\beta_0 = w_0$

Let us recall our results from previous parts:

$$\beta = \hat{\Sigma}^{-1}(\mu_1 - \mu_2) \qquad \beta_0 = -\frac{1}{2}\Big(\mu_1^T\hat{\Sigma}^{-1}\mu_1 - \mu_2^T\hat{\Sigma}^{-1}\mu_2\Big) + log\frac{\pi_1}{\pi_2}$$

$$\begin{aligned} w_0 &= \frac{N_1c_1 + N_2c_2}{N} - \frac{(N_1\hat{\mu_1}^T + N_2\hat{\mu_2}^T)w}{N} \\ w &= \frac{1}{(N-2)}\Big(\frac{+c_2N_1N_2 - c_1N_1N_2}{N} - \frac{N_1N_2(\hat{\mu}_2 - \hat{\mu}_1)^Tw}{N}\Big)\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) \\ \pi_1 = \frac{N_1}{N} &, \pi_2 = \frac{N_2}{N} \\ N &= N_1 + N_2 \end{aligned}$$

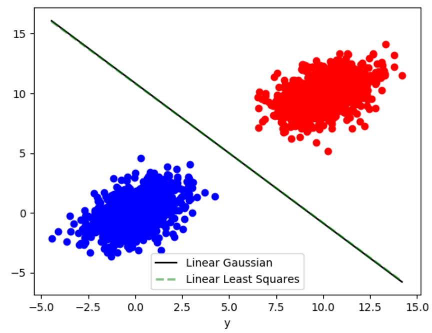
Proof: Using the fact that $c_1 = 1, c_2 = -1, N_1 = N_2$ and simplifying above expressions we get:

$$\begin{aligned} w &= \frac{1}{(2N-2)}\Big(\frac{-N^2}{N} - \frac{N^2(\hat{\mu}_2 - \hat{\mu}_1)^Tw}{N}\Big)\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) \\ \Rightarrow w &= \frac{1}{(2N-2)}\Big(-N - N(\hat{\mu}_2 - \hat{\mu}_1)^Tw\Big)\beta \\ w &= a\beta \text{ where } a = \frac{1}{(2N-2)}\Big(-N - N(\hat{\mu}_2 - \hat{\mu}_1)^Tw\Big) \\ \pi_1 = \frac{N}{2N} = \frac{1}{2} &, \pi_2 = \frac{N}{2N} = \frac{1}{2} \\ \beta_0 &= -\frac{1}{2}\Big(\mu_1^T\hat{\Sigma}^{-1}\mu_1 - \mu_2^T\hat{\Sigma}^{-1}\mu_2\Big) + log\frac{\pi_1}{\pi_2} \\ \Rightarrow \beta_0 &= -\frac{1}{2}\Big(\mu_1^T\hat{\Sigma}^{-1}\mu_1 - \mu_2^T\hat{\Sigma}^{-1}\mu_2\Big) + log(\frac{1/2}{1/2}) \\ \Rightarrow \beta_0 &= -\frac{1}{2}\Big(\mu_1^T\hat{\Sigma}^{-1}\mu_1 - \mu_2^T\hat{\Sigma}^{-1}\mu_2\Big) \\ w_0 &= \frac{N_1c_1 + N_2c_2}{N} - \frac{(N_1\hat{\mu_1}^T + N_2\hat{\mu_2}^T)w}{N} \\ \Rightarrow w_0 &= \frac{N - N}{2N} - \frac{(N\hat{\mu_1}^T + N\hat{\mu_2}^T)w}{2N} \\ &= -\frac{(\hat{\mu_1}^T + \hat{\mu_2}^T)w}{2} \\ &= -\frac{(\hat{\mu_1} + \hat{\mu_2})^Ta\beta}{2} \\ &= -\frac{a}{2}(\hat{\mu_1} + \hat{\mu_2})^T\hat{\Sigma}^{-1}(\mu_1 - \mu_2) \\ &= -\frac{a}{2}(\hat{\mu_1}^T\hat{\Sigma}^{-1}\mu_1 - \hat{\mu_1}^T\hat{\Sigma}^{-1}\mu_2 + \hat{\mu_2}^T\hat{\Sigma}^{-1}\mu_1 - \hat{\mu_2}^T\hat{\Sigma}^{-1}\mu_2) \\ &= -\frac{a}{2}(\hat{\mu_1}^T\hat{\Sigma}^{-1}\mu_1 - \hat{\mu_2}^T\hat{\Sigma}^{-1}\mu_2) \\ \Rightarrow w_0 &= a\beta_0 \end{aligned}$$

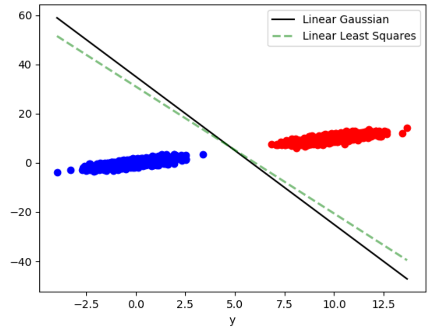
hence proved

e

(i),(ii),(iii) Please refer to page17 for code



(h) Plot 1



(i) Plot 2

Condition number for plot 1: 2.6993272947280484

$$\hat{\Sigma}_{plot1} \begin{bmatrix} 0.6141065 & 1.6696115 \\ 1.6696115 & 0.92650076 \end{bmatrix}$$

Condition number is for second plot : 7.664828236667515

$$\hat{\Sigma}_{plot2} \begin{bmatrix} 0.67383727 & 1.22490623 \\ 1.22490623 & 1.2645129 \end{bmatrix}$$

Discussion: We can observe from the plots that even with a good condition number for the variance the two decision lines do not always match.

f Given: $c_1 = 1, c_2 = -1, N_1 = N_2 = N/2$, We have to comment on the geometry of the following problem:

$$A^T A \begin{bmatrix} w \\ w_0 \end{bmatrix} = A^T b$$

Lets recall our previous results:

$$A^T A = \begin{bmatrix} (N-2)\hat{\Sigma} + N_1 \hat{\mu_1} \hat{\mu_1}^T + N_2 \hat{\mu_2} \hat{\mu_2}^T & N_1 \hat{\mu_1} + N_2 \hat{\mu_2} \\ N_1 \hat{\mu_1}^T + N_2 \hat{\mu_2}^T & N \end{bmatrix}$$

$$\Rightarrow A^T A = \begin{bmatrix} (N-2)\hat{\Sigma} + \frac{N}{2} \hat{\mu_1} \hat{\mu_1}^T + \frac{N}{2} \hat{\mu_2} \hat{\mu_2}^T & \frac{N}{2}(\hat{\mu_1} + \hat{\mu_2}) \\ \frac{N}{2}(\hat{\mu_1} + \hat{\mu_2})^T & N \end{bmatrix}$$

$$A^T b = \begin{bmatrix} c_1 N_1 \hat{\mu_1} + c_2 N_2 \hat{\mu_2} \\ N_1 c_1 + N_2 c_2 \end{bmatrix}$$

$$\Rightarrow A^T b = \begin{bmatrix} \frac{N}{2}(\hat{\mu}_1 - \hat{\mu}_2) \\ 0 \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} (N-2)\hat{\Sigma} + \frac{N}{2} \hat{\mu_1} \hat{\mu_1}^T + \frac{N}{2} \hat{\mu_2} \hat{\mu_2}^T & \frac{N}{2}(\hat{\mu_1} + \hat{\mu_2}) \\ \frac{N}{2}(\hat{\mu_1} + \hat{\mu_2})^T & N \end{bmatrix} \begin{bmatrix} w \\ w_0 \end{bmatrix} = \begin{bmatrix} \frac{N}{2}(\hat{\mu}_1 - \hat{\mu}_2) \\ 0 \end{bmatrix}$$

$$
\begin{aligned}
\Rightarrow w_0 &= -\frac{1}{2}(\hat{\mu}_1+\hat{\mu}_2)^T w \\
((N-2)\hat{\Sigma} + \frac{N}{2}\hat{\mu}_1\hat{\mu}_1{}^T + \frac{N}{2}\hat{\mu}_2\hat{\mu}_2{}^T)w + \frac{N}{2}(\hat{\mu}_1+\hat{\mu}_2)w_0 &= \frac{N}{2}(\hat{\mu}_1-\hat{\mu}_2) \\
((N-2)\hat{\Sigma} + \frac{N}{2}\hat{\mu}_1\hat{\mu}_1{}^T + \frac{N}{2}\hat{\mu}_2\hat{\mu}_2{}^T)w + \frac{N}{2}(\hat{\mu}_1+\hat{\mu}_2)(-\frac{1}{2}(\hat{\mu}_1+\hat{\mu}_2)^T w) &= \frac{N}{2}(\hat{\mu}_1-\hat{\mu}_2) \\
((N-2)\hat{\Sigma} + \frac{N}{2}\hat{\mu}_1\hat{\mu}_1{}^T + \frac{N}{2}\hat{\mu}_2\hat{\mu}_2{}^T - \frac{N}{4}(\hat{\mu}_1+\hat{\mu}_2)((\hat{\mu}_1+\hat{\mu}_2)^T)w &= \frac{N}{2}(\hat{\mu}_1-\hat{\mu}_2) \\
((N-2)\hat{\Sigma} + \frac{N}{4}\hat{\mu}_1\hat{\mu}_1{}^T + \frac{N}{4}\hat{\mu}_2\hat{\mu}_2{}^T - \frac{N}{2}\hat{\mu}_1\hat{\mu}_2{}^T)w &= \frac{N}{2}(\hat{\mu}_1-\hat{\mu}_2) \\
((N-2)\hat{\Sigma} + \frac{N}{4}(\hat{\mu}_1-\hat{\mu}_2)(\hat{\mu}_1-\hat{\mu}_2)^T)w &= \frac{N}{2}(\hat{\mu}_1-\hat{\mu}_2) \\
w &= k\hat{\Sigma}^{-1}(\hat{\mu}_1-\hat{\mu}_2)
\end{aligned}
$$

We get the expression for w and w_0 from the above procedure, we can also find the x_0 such that the decision line can be expressed in the form of $g(x) = w^T x + w_0 = w^T(x - x_0) = 0 \Rightarrow w_0 = -w^T x_0$

Therefore, $x_0 = \frac{1}{2}(\hat{\mu}_1+\hat{\mu}_2)$ ie mid point of the two means and $w = k\hat{\Sigma}^{-1}(\hat{\mu}_1-\hat{\mu}_2)$ ie the direction parallel to the difference in the means rotated by $\hat{\Sigma}^{-1}$.

Main Code for Exercise 2

Listing 1: Source Code

```
"""
ECE 595: ML-1
HW-3: main file
@author- Rahul Deshmukh
"""
#%% Import libraries
import numpy as np
import functions as myfun
import cv2
#%% 1 Read data from files
read_path = '../data/'
save_path = '../result/'
#------------------------------Exercise-2:CAT Classification------------------------------#
#(a)
#(i)
train_cat= np.matrix(np.loadtxt(read_path+'train_cat.txt',delimiter=','))
train_grass= np.matrix(np.loadtxt(read_path+'train_grass.txt',delimiter=','))
N_cat= np.shape(train_cat)[1]
N_grass= np.shape(train_grass)[1]
N_total =N_cat+N_grass
#(ii)
# calculate the mean
u_cat = np.mean(train_cat,axis=1)
u_grass = np.mean(train_grass,axis = 1)
# calculate the variance
sigma_cat = myfun.find_covariance(train_cat,u_cat)
sigma_grass = myfun.find_covariance(train_grass,u_grass)
#priors
pi_cat= N_cat/(N_total)
pi_grass= N_grass/N_total
#%% (b)
#(i)
cat_img = cv2.imread(read_path+'cat_grass.jpg',0) # read in grayscale
cat_img = cat_img/255.0 # normalizing

#(ii)
# prepare lists for classification
u = [u_grass,u_cat]
sigma = [sigma_grass,sigma_cat]
prior = [pi_grass,pi_cat]
user_def_label = [0,1]

overlapping_labels = myfun.classify_overlapping_patches(cat_img,u,sigma,prior,
    user_def_label)
cv2.imwrite(save_path+'overlapping_patches.jpg',255*overlapping_labels)

#(iii) non-overlapping patches
non_overlapping_labels = myfun.classify_non_overlapping_patches(cat_img,u,sigma,
    prior,user_def_label)
cv2.imwrite(save_path+'non_overlapping_patches.jpg',255*non_overlapping_labels)

#(iv)
#read ground truth
ground_truth = cv2.imread(read_path+'truth.png',0)/255
# find mean abs errors
```

```
error1 = myfun.mean_abs_error(overlapping_labels,ground_truth)
error2 = myfun.mean_abs_error(non_overlapping_labels,ground_truth)
print('Error for overlapping: '+str(100*error1)+'%\nand for non-overlapping:'+str(100*error2)+'%')

#(v) Read image of cheetah on grass
cheetah_img= cv2.imread(read_path+'cheetah3.jpg',0)

cheetah_overlapping_labels = myfun.classify_overlapping_patches(cheetah_img,u,sigma,prior,user_def_label)
cv2.imwrite(save_path+'cheetah_overlapping_patches.jpg',255*cheetah_overlapping_labels)

cheetah_non_overlapping_labels = myfun.classify_non_overlapping_patches(cheetah_img,u,sigma,prior,user_def_label)
cv2.imwrite(save_path+'cheetah_non_overlapping_patches.jpg',255*cheetah_non_overlapping_labels)
```

Functions for Exercise 2

Listing 2: Functions

```
"""
ECE 595: ML-1
HW-3: functions
@author- Rahul Deshmukh
"""
# import libraries
import numpy as np

#%%
# function for calculating mean abs error
def mean_abs_error(prediction,truth):
    """
    Input: prediciton: grayscale image normalized to 0-1
           truth: grayscale image normalized to 0-1
           both are of same sizes
    Output: error: scalar value
    """
    error = 0
    M,N = np.shape(prediction)
    error = (1/(M*N))*np.sum(np.abs(prediction-truth))
    return(error)

# function for classifying non overlapping patches of an image
def classify_non_overlapping_patches(img,u,sigma,prior,user_def_label):
    """
    Input: img: np array of image in grayscale(scaled to 0-1)
    u: class means as list of 1d arrays
    sigma: class covariances as list of 2d arrays
    prior: list of priors
    user_def_label: list of label values
    Output: predicted_labels: np array of predicted label
    """
    M,N = np.shape(img)
    predicted_label = np.zeros((M,N))
    for i in range(M//8-1):
        for j in range(N//8-1):
```

```
            test_sample  = img[8*i:8*i+8,8*j:8*j+8]
            # convert test sample to a vector
            test_sample = np.reshape(test_sample,(64,1))
            # find class of the sample
            predicted_label[8*i:8*i+8,8*j:8*j+8] = classify_sample(test_sample,u,sigma,prior,user_def_label)
    return(predicted_label)

#function for classifying the a test sample using MLE decision line
def classify_sample(x,u,sigma,prior,user_def_label):
    """
    Input: x: sample
    u: class means as list of 1d arrays
    sigma: class covariances as list of 2d arrays
    prior: list of priors
    user_def_label: list of labels
    Output: predicted_label (0,1,2,...K-1)
    """
    # find number of classes
    Num_class = len(u)
    disc = [] #discriminant
    for i in range(Num_class):
        iu = u[i];
        isigma = sigma[i];
        isigma_inv = np.linalg.inv(isigma)
        iprior = prior[i];
        idisc = -(1/2)*(x-iu).T@isigma_inv@(x-iu) \
                -(1/2)*np.log(np.linalg.det(isigma))\
                +np.log(iprior)
        disc.append(idisc)
    predicted_label = user_def_label[np.argmax(disc)]
    return(predicted_label)
# function for obtainting the image patches
def classify_overlapping_patches(img,u,sigma,prior,user_def_label):
    """
    Input: img: np array of image in grayscale(scaled to 0-1)
    u: class means as list of 1d arrays
    sigma: class covariances as list of 2d arrays
    prior: list of priors
    user_def_label: list of label values
    Output: predicted_labels: np array of predicted label
    """
    M,N = np.shape(img)
    predicted_label = np.zeros((M,N))
    for i in range(M-8):
        for j in range(N-8):
            test_sample  = img[i:i+8,j:j+8]
            # convert test sample to a vector
            test_sample = np.reshape(test_sample,(64,1))
            # find class of the sample
            predicted_label[i,j] = classify_sample(test_sample,u,sigma,prior,user_def_label)
    return(predicted_label)
# function for finding the covariance using samples
def find_covariance(x,u):
    """
    input: x: samples in the format of a np matrix with col vectors
           u: mean of samples
    output: covar: matrix
```

```
    """
    Ndim,Nsample=np.shape(x)
    # substract mean vector from samples
    X = x-np.kron(np.ones((1,Nsample)),u)
    covar = (1/Nsample)*(X@X.T)
    return(covar)
```

Code for Exercise 3

```
"""
ECE 595: ML-1
HW-3 Exercise 3
@author- Rahul Deshmukh
"""
#%% Import libraries
import numpy as np
import matplotlib.pyplot as plt
#%%
#(i) Generate data
NumPts = 1000
# priors
pi_1 = 1/2
pi_2 = 1/2
# define mean and variance
u1 = np.array([0,0])
u2= np.array([10,10])
sigma = np.random.rand(2,2)
sigma = sigma +sigma.T # becomes symmetric
d,v = np.linalg.eig(sigma)
d = np.abs(d)# now semi pos def
# improve the condition number
if min(d)<1e-6:
    d[np.argmin(d)]=1
if max(d)>1e6:
    d[np.argmax(d)]=1
print('Condition number is: '+str(np.linalg.cond(sigma)))
sigma = v@np.diag(d)@v.T
sigma_inv = np.linalg.inv(sigma)
# sample points from gaussian
x1 = np.random.multivariate_normal(u1,sigma,NumPts) #pts for class1
x2 = np.random.multivariate_normal(u2,sigma,NumPts) #pts for class1
x_min = min(min(x1[:,0]),min(x2[:,0]))
x_max = max(max(x1[:,0]),max(x2[:,0]))
#plot data for visual check
plt.scatter(x1[:,0],x1[:,1],c='b')
plt.scatter(x2[:,0],x2[:,1],c='r')
plt.show()
# (ii) plot the decision line
x_pts = np.linspace(x_min,x_max,100) # for decision line
# decision line using part(a)
beta =sigma_inv@(u1-u2)
beta_0 = -(1/2)*(u1.T@sigma_inv@u1-u2.T@sigma_inv@u2)+np.log(pi_1/pi_2)
# find y coords of line
y_pts = -(beta[0]*x_pts+beta_0)/(beta[1])
# plot line
plt.plot(x_pts,y_pts,c='black')
plt.xlabel('x')
plt.xlabel('y')
```

```
plt.show()

#(iii) Decision line using linear least squares
A = np.ones((2*NumPts,3))
A[:NumPts,:-1] = x1
A[NumPts:,:-1] = x2
b = np.vstack((np.ones((NumPts,1)),-1*np.ones((NumPts,1))))
# solve using LLS
theta = np.linalg.inv(A.T@A)@(A.T@b)
y_lls = -(theta[0]*x_pts+theta[2])/(theta[1])
#plot lls line
plt.plot(x_pts,y_lls,c='green',alpha=0.5,linewidth=2,linestyle='--')
plt.legend(['Linear Gaussian','Linear Least Squares'])
plt.show()
```