CS 229, Fall 2018
Problem Set #3 Solutions:
Deep Learning & Unsupervised learning

1. A Simple Neural Network

a) Gradient w.r.t $w_{1,2}^{[1]}$.

Forward propagation:

$$\begin{aligned} z^{[1]} &= W^{[1]}x^{(i)} \\ h &= g\left(z^{[1]}\right) \\ z^{[2]} &= W^{[2]}h \\ o^{(i)} &= g\left(z^{[2]}\right) \end{aligned}$$

Cost function:

$$\ell = \frac{1}{m}\Sigma_i^m\left(o^{(i)} - y^{(i)}\right)^2$$

Derivative of $W^{(2)}$ (Not needed in this problem).

$$\begin{aligned} \frac{\partial \ell}{\partial W^{[2]}} &= \frac{\partial}{\partial W^{[2]}}\left(o^{(i)} - y^{(i)}\right)^2 \\ &= 2\left(o^{(i)} - y^{(i)}\right)\frac{\partial}{\partial W^{[2]}}\left(g\left(z^{[2]}\right)\right) \\ &= 2\left(o^{(i)} - y^{(i)}\right)g\left(z^{[2]}\right)\left(1 - g\left(z^{[2]}\right)\right)h \\ &= 2\left(o^{(i)} - y^{(i)}\right)o^{(i)}\left(1 - o^{(i)}\right)h \end{aligned}$$

Derivative of $W^{[1]}$

$$\begin{aligned} \frac{\partial \ell}{\partial W^{[1]}} &= \frac{\partial \ell}{\partial o}\frac{\partial o}{\partial z^{[2]}}\frac{\partial z^{[2]}}{\partial h}\frac{\partial h}{\partial z^{[1]}}\frac{\partial z^{[1]}}{\partial W^{[1]}} \\ &= 2(o - y)o(1 - o)W^{[2]}h(1 - h)x \end{aligned}$$

Add subscription and sum up

$$\frac{\partial \ell}{\partial W_{1,2}^{[1]}} = \frac{1}{m}\sum_{i=1}^{m} 2\left(o^{(i)} - y^{(i)}\right)o^{(i)}\left(1 - o^{(i)}\right)W^{[2]}h_2(1 - h_2)x^{(i)}$$

b) In the dataset plot, find three points clock wise which can form a triangle to separate the dataset in two:

$$\begin{aligned} a^{(1)} &= (0.5, 0.5) \\ a^{(2)} &= (0.5, 3.5) \\ a^{(3)} &= (3.5, 0.5) \end{aligned}$$

Subtract with the next to form three vectors, and add intercept in front:

$$\begin{aligned} \mathrm{v}^{(1)} &= (0, -3.0) \\ \mathrm{v}^{(2)} &= (-3.0, 3.0) \\ \mathrm{v}^{(3)} &= (3.0, 0) \end{aligned}$$

Equations for the above three vectors. For $v^{(1)}$ and $v^{(2)}$:

$$\frac{\mathrm{x}_1 - \mathrm{a}_1^{(2)}}{\mathrm{v}_1^{(1)}} = \frac{\mathrm{x}_2 - \mathrm{a}_2^{(2)}}{\mathrm{v}_2^{(1)}}$$

Which can re rewritten as:

$$\mathrm{v}_2^{(1)}\mathrm{x}_1 - \mathrm{v}_1^{(1)}\mathrm{x}_2 + \left(\mathrm{v}_1^{(1)}\mathrm{a}_2^{(2)} - \mathrm{v}_2^{(1)}\mathrm{a}_1^{(2)}\right) = 0$$

This gives us the first component of matrix $W^{[1]}$:

$$W_1^{[1]} = \begin{pmatrix} \mathrm{v}_2^{(1)} \\ -\mathrm{v}_1^{(1)} \\ \mathrm{v}_1^{(1)}\mathrm{a}_2^{(2)} - \mathrm{v}_2^{(1)}\mathrm{a}_1^{(2)} \end{pmatrix} = \begin{pmatrix} -3.0 \\ 0 \\ 1.5 \end{pmatrix}$$

Similarly, for $v^{(2)}$ and $v^{(3)}$ we have:

$$W_2^{[1]} = \begin{pmatrix} \mathrm{v}_2^{(2)} \\ -\mathrm{v}_1^{(2)} \\ \mathrm{v}_1^{(2)}\mathrm{a}_2^{(3)} - \mathrm{v}_2^{(2)}\mathrm{a}_1^{(3)} \end{pmatrix} = \begin{pmatrix} 3.0 \\ 3.0 \\ -12.0 \end{pmatrix}$$

And for $v^{(3)}$ and $v^{(1)}$ we have:

$$W_3^{[1]} = \begin{pmatrix} \mathrm{v}_2^{(3)} \\ -\mathrm{v}_1^{(3)} \\ \mathrm{v}_1^{(3)}\mathrm{a}_2^{(1)} - \mathrm{v}_2^{(3)}\mathrm{a}_1^{(1)} \end{pmatrix} = \begin{pmatrix} 0 \\ -3.0 \\ 1.5 \end{pmatrix}$$

Stack them to from a matrix $W^{[1]}$:

$$W^{[1]} = \begin{pmatrix} -3.0 & 3.0 & 0 \\ 0 & 3.0 & -3.0 \\ 1.5 & -12.0 & 1.5 \end{pmatrix}$$

Explanation: given any point x, if its projection on the normal of vector $v^{(i)}$ is positive then it is on the right hand of $v^{(i)}$, otherwise on the left. If it lies on the right of all the vectors, it is inside the triangle. (note the intercept is the last component of W which can be easily adjusted to be the first component)

Output layer matrix can be:

$$W^{[2]} = \begin{pmatrix} -1 \\ -1 \\ -1 \\ 2.5 \end{pmatrix}$$

Because the input can only be one of the eight values of h:

$$h = \begin{pmatrix} 0/1 \\ 0/1 \\ 0/1 \\ 1 \end{pmatrix}$$

This guarantees that only for dataset inside the triangle could the product $W^{[2]}h$ be negative, which will then make f(x) = 0. All the other cases will have $W^{[2]}h \geq 0$, which makes f(x) = 1.

c) The activation function for h_1, h_2, h_3 is changed to $f(x) = x$, provide a set of weights that makes it achieve 100% accuracy.

Answer:
If the activation function is just f(x)=x, the NN is degraded to logistic regression. It is impossible to achieve 100% accuracy since the feature mappings doesn't include high order components. The decision boundary is going to be a straight line and will have misclassification.

2. KL divergence

2 KL divergence and Maximum Likelihood

(a) Nonnegativity

Prove the following:

$$\forall P, Q, D_{KL}(P \parallel Q) \geq 0$$

And

$$D_{KL}(P||Q) = 0 \iff P = Q$$

PROOF 1:

$$\begin{aligned}
D_{KL}(P \parallel Q) &= \sum_x P(x) log \frac{P(x)}{Q(x)} \\
&= -\sum_x P(x) log \frac{Q(x)}{P(x)} \\
&>= -log(\sum_x P(x) \frac{Q(x)}{P(x)}) \\
&= -log(\sum_x Q(x)) \\
&= -log(1) = 0
\end{aligned}$$

PROOF 2:

$$D_{KL}(P \parallel Q) = 0 \iff P = Q$$

a. If $P = Q$, $D_{KL}(P \parallel Q) = \sum_{x \in X} P \log \frac{P}{Q} = \sum_{x \in X} P \log(1) = 0$

b. If $D_{KL}(P \parallel Q) = 0$, given $-\log x$ is strictly convex, then

$$\begin{aligned}
E[-log(\frac{P}{Q})] \geq -log(E[\frac{P}{Q}]) = 0 \\
= log(E[\frac{Q}{P}]) \\
= log(\sum [P\frac{Q}{P}]) \\
= log(1) \\
= 0
\end{aligned}$$

The equality holds iff $\frac{Q}{P}$ is a constant with probability 1, given the fact that both P and Q are pdf, we can only have $P = Q$.

(b) Chain rule for KL divergence

Prove that:

$$D_{KL}(P(X,Y) \parallel Q(X,Y)) = D_{KL}(P(X) \parallel Q(X)) + D_{KL}(P(Y|X) \parallel Q(Y|X))$$

PROOF:

$$\begin{aligned}
D_{KL}(P(X) \parallel Q(X)) + D_{KL}(P(Y|X) \parallel Q(Y|X)) &= \sum_x P(x) \log \frac{P(x)}{Q(x)} + \sum_x P(x) (\sum_y P(y|x) \log \frac{P(y|x)}{Q(y|x)}) \\
&= \sum_x P(x) \left(\log \frac{P(x)}{Q(x)} + \sum_y P(y|x) \log \frac{P(y|x)}{Q(y|x)} \right) \\
&= \sum_x P(x) \left(\sum_y P(y|x) \log \frac{P(x)}{Q(x)} + \sum_y P(y|x) \log \frac{P(y|x)}{Q(y|x)} \right) \\
&= \sum_x P(x) \left(\sum_y P(y|x) \left(\log \frac{P(x)}{Q(x)} + \log \frac{P(y|x)}{Q(y|x)} \right) \right) \\
&= \sum_x \left(\sum_y P(y|x) P(x) (\log \frac{P(x)P(y|x)}{Q(x)Q(y|x} \right) \\
&= \sum_x \sum_y P(x,y) \log \frac{P(x,y)}{Q(x,y)} \\
&= D_{KL}(P(X,Y) \parallel Q(X,Y))
\end{aligned}$$

(c) KL and maximum likelihood

Prove that

$$\arg\min_{\theta} D_{KL}(\hat{P} \parallel P_\theta) = \arg\max_{\theta} \sum_{i=1}^{m} \log P_\theta(x^{(i)})$$

$$\begin{aligned}
D_{KL}(\hat{P} \parallel P_\theta) &= \sum_x \hat{P}(x) \log \frac{\hat{P}(x)}{P_\theta(x)} \\
&= -\sum_x \hat{P}(x) \log \frac{P_\theta(x)}{\hat{P}(x)} \\
&= -\sum_x \frac{1}{m} \sum_{i=1}^{m} 1\{x^{(i)} = x\} \log \frac{P_\theta(x)}{\frac{1}{m}\sum_{i=1}^{m} 1\{x^{(i)} = x\}} \\
&= -\frac{1}{m} \sum_{i=1}^{m} \log \frac{P_\theta(x^{(i)})}{\frac{1}{m}\sum_{i=1}^{m}} \\
&= -\frac{1}{m} \sum_{i=1}^{m} \log P_\theta(x^{(i)}) \\
&= -\frac{1}{m} \text{log-likelihood}
\end{aligned}$$

Which implies that

$$\arg\min_{\theta} D_{KL}(\hat{P} \parallel P_\theta) = \arg\max_{\theta} \sum_{i=1}^{m} \log P_\theta(x^{(i)})$$

3. KL divergence, Fisher Information, Natural gradient

PS3.3 [For simplicity, $\nabla_{\theta'} \log p(y;\theta')|_{\theta'=\theta}$ is sometimes written as $\nabla_\theta \log p(y;\theta)$ directly]

(a) Suppose the score function is

$$s(y;\theta) = \nabla_\theta \log p(y;\theta);$$

prove that:

$$\mathbb{E}_{y\sim p(y;\theta)}[S(y;\theta')|_{\theta'=\theta'}] = 0$$

PROOF:

$$\mathbb{E}_{y\sim p(y;\theta)}[S(y;\theta')|_{\theta'=\theta}] = \int p(y;\theta) S(y;\theta)\, dy$$

$$= \int p(y;\theta) \nabla_\theta \log p(y;\theta)\, dy = \int p(y;\theta) \cdot \frac{1}{p(y;\theta)} \cdot \frac{\partial p(y;\theta)}{\partial \theta}\, dy$$

$$= \frac{\partial}{\partial\theta} \int p(y;\theta)\, dy = \frac{\partial}{\partial\theta}(1) = 0. \quad \text{Done.}$$

(b) Fisher Information:

$$I(\theta) = \mathrm{Cov}_{y\sim p(y;\theta)}[\nabla_{\theta'} \log p(y;\theta')|_{\theta'=\theta}]$$

show that

$$I(\theta) = \mathbb{E}_{y\sim p(y;\theta)}[\nabla_{\theta'} \log p(y;\theta') \nabla_{\theta'} \log p(y;\theta')^T|_{\theta'=\theta}]$$

PROOF: By definition of Covariance, Given a vector $X = \nabla_\theta \log p(y;\theta)|_{\theta'=\theta}$

$$I(\theta) = \mathrm{Cov}_{y\sim p(y;\theta)}[\nabla_{\theta'} \log p(y;\theta')|_{\theta'=\theta}]$$

$$= \mathbb{E}_{y\sim p(y;\theta)}[\nabla_{\theta'} \log p(y;\theta') \nabla_{\theta'} \log p(y;\theta')^T|_{\theta'=\theta}]$$

$$- \mathbb{E}_{y\sim p(y;\theta)}[\nabla_{\theta'} \log p(y;\theta')]_{\theta'=\theta} \cdot \mathbb{E}_{y\sim p(y;\theta)}[\nabla_{\theta'} \log p(y;\theta)]|_{\theta'=\theta}$$

$$= \mathbb{E}_{y\sim p(y;\theta)}[\nabla_{\theta'} \log p(y;\theta') \nabla_{\theta'} \log p(y;\theta')^T|_{\theta'=\theta}]$$

~~The last part is from the fact that~~ $\mathbb{E}_{y\sim p(y;\theta)}[S(y;\theta)] = 0$.

$$\mathrm{Cov}(X) = \begin{bmatrix} \mathbb{E}(X_1^2) - \mathbb{E}^2(X_1) & \mathbb{E}(X_1X_2) - \mathbb{E}(X_1)\mathbb{E}(X_2) & \cdots & \mathbb{E}(X_1X_n) - \mathbb{E}(X_1)\mathbb{E}(X_n) \\ \mathbb{E}(X_2X_1) - \mathbb{E}(X_2)\mathbb{E}(X_1) & \mathbb{E}(X_2^2) - \mathbb{E}^2(X_2) & \cdots & \mathbb{E}(X_2X_n) - \mathbb{E}(X_2)\mathbb{E}(X_n) \\ \mathbb{E}(X_3X_1) - \mathbb{E}(X_3)\mathbb{E}(X_1) & \cdots & \cdots & \cdots \\ \vdots & & & \\ \mathbb{E}(X_nX_1) - \mathbb{E}(X_n)\mathbb{E}(X_1) & \mathbb{E}(X_nX_2) - \mathbb{E}(X_n)\mathbb{E}(X_2) & \cdots & \mathbb{E}(X_nX_n) - \mathbb{E}^2(X_n) \end{bmatrix}$$

From (a) we know that $\mathbb{E}(X_i) = 0$; so:

$$Cov(X) = \begin{bmatrix} \mathbb{E}(X_1^2) & \mathbb{E}(X_1X_2) & \cdots & \mathbb{E}(X_1X_n) \\ \mathbb{E}(X_2X_1) & \mathbb{E}(X_2^2) & \cdots & \mathbb{E}(X_2X_n) \\ \vdots & & & \\ \mathbb{E}(X_nX_1) & & & \mathbb{E}(X_n^2) \end{bmatrix} = \mathbb{E}\begin{bmatrix} X_1^2 & X_1X_2 & \cdots & X_1X_n \\ X_2X_1 & X_2^2 & \cdots & X_2X_n \\ \vdots & & & \\ X_nX_1 & & & X_n^2 \end{bmatrix}$$

$= \mathbb{E}\left[XX^T\right]$ replace $X = \nabla_{\theta'} \log p(y;\theta')|_{\theta'=\theta}$, Proved.

(c). Show that:

$$I(\theta) = \mathbb{E}_{y\sim p(y;\theta)}\left[-\nabla_{\theta'}^2 \log p(y;\theta')|_{\theta'=\theta}\right]$$

PROOF:

$$\begin{aligned} \nabla_\theta^2 \log p(y;\theta) &= \nabla_\theta\left(\nabla_\theta \log p(y;\theta)\right) \\ &= \nabla_\theta\left(\nabla_\theta p(y;\theta)/p(y;\theta)\right) \\ &= \nabla_\theta^2 p(y;\theta)/p(y;\theta) - \left(\nabla_\theta p(y;\theta)/p(y;\theta)\right)\left(\nabla_\theta p(y;\theta)/p(y;\theta)\right)^T \\ &= \nabla_\theta^2 p(y;\theta)/p(y;\theta) - \left(\nabla_\theta \log p(y;\theta)\right)\left(\nabla_\theta \log p(y;\theta)\right)^T \end{aligned}$$

Sum over y on both sides:

$$\mathbb{E}_{y\sim p(y;\theta)}\left[\nabla_\theta^2 \log p(y;\theta)\right] = \mathbb{E}_\theta\left[\nabla_\theta^2 p(y;\theta)/p(y;\theta)\right] - \mathbb{E}_\theta\left[\nabla_\theta \log p(y;\theta) \nabla_\theta \log p(y;\theta)^T\right]$$

Consider the first part of the right hand side:

$$\mathbb{E}_{y\sim p(y;\theta)}\left[\nabla_\theta^2 p(y;\theta)/p(y;\theta)\right] = \int p(y;\theta)\cdot\frac{\nabla_\theta^2 p(y;\theta)}{p(y;\theta)}dy = \int \nabla_\theta^2 p(y;\theta)\,dy = \nabla_\theta^2 \int p(y;\theta)\,dy = \nabla_\theta^2(1) = 0;$$

The second part of the right hand side:

$$\mathbb{E}_\theta\left[\nabla_\theta \log p(y;\theta)\, \nabla_\theta \log p(y;\theta)^T\right] = I(\theta)$$

So:

$$\mathbb{E}_\theta\left[\nabla_\theta^2 \log p(y;\theta)\right] = -I(\theta)$$

$$I(\theta) = \mathbb{E}_\theta\left[-\nabla_\theta^2 \log p(y;\theta)\right] = \mathbb{E}_{y\sim p(y;\theta)}\left[-\nabla_{\theta'}^2 \log p(y;\theta')|_{\theta'=\theta}\right]$$

Done!

(d) $D_{KL}(p_\theta \| p_{\theta+d}) \approx \frac{1}{2} d^T I(\theta) d$.

PROOF:

$D_{KL}(p_\theta \| p_{\theta+d}) = \mathbb{E}_\theta[\log p_\theta - \log p_{\theta+d}] = \mathbb{E}_\theta[\log p_\theta] - \mathbb{E}_\theta[\log p_{\theta+d}]$

Consider $\log p_{\theta+d}$ using Taylor's expansion:

$\log p_{\theta+d} = \log p_\theta + d \nabla_\theta \log p_\theta + \frac{1}{2} d^T (\nabla_\theta^2 \log p_\theta) d$

Sum over $y \sim p(y;\theta)$ for both sides:

$$\mathbb{E}_\theta[\log p_{\theta+d}] \approx \mathbb{E}_\theta[\log p_\theta] + \mathbb{E}_\theta[d \nabla_\theta \log p_\theta] + \mathbb{E}_\theta[\tfrac{1}{2} d^T (\nabla_\theta^2 \log p_\theta) d]$$

$$= \mathbb{E}_\theta[\log p_\theta] + d\,\mathbb{E}_\theta[\nabla_\theta \log p_\theta] + \tfrac{1}{2} d^T \mathbb{E}_\theta[\nabla_\theta^2 \log p_\theta] d$$

$$= \mathbb{E}_\theta[\log p_\theta] - \tfrac{1}{2} d^T I(\theta) d$$

Which implies:

$D_{KL}(p_\theta \| p_{\theta+d}) \approx \frac{1}{2} d^T I(\theta) d$. done!

(e). $l(\theta) = \log p(y;\theta)$; $\theta \in \mathbb{R}^n$;

The Tylor expansion for $D_{KL}(p_\theta \| p_{\theta+d}) \approx \frac{1}{2} d^T I(\theta) d$;

The Tylor expansion for $l(\theta+d) \approx l(\theta) + d^T \nabla_\theta l(\theta)$;

the Original problem:

$$d^* = \arg\max_d l(\theta+d), \quad s.t.\ D_{KL}(p_\theta \| p_{\theta+d}) = c,$$

Replace both Tylor expansions:

$$d^* = \arg\max l(\theta) + d^T \nabla_\theta l(\theta), \quad s.t.\ D_{KL}(p_\theta \| p_{\theta+d}) \doteq \tfrac{1}{2} d^T I(\theta) d = c$$

→ the Lagrangean is:

$$L(d, \lambda) = l(\theta) + d^T \nabla_\theta l(\theta) - \lambda \cdot \left(\tfrac{1}{2} d^T I(\theta) d - c\right)$$

→ Derivatives of linear equations:

$$\frac{\partial L}{\partial d} = \nabla_\theta l(\theta) - \lambda I(\theta) d = 0 \quad ; \quad ⓐ$$

$$\frac{\partial L}{\partial \lambda} = c - \tfrac{1}{2} d^T I(\theta) d = 0 \quad ; \quad ⓑ$$

→ From ⓐ: $\tilde{d} = \lambda^{-1} I(\theta)^{-1} \nabla_\theta l(\theta)$;

→ ~~From~~ plug d to ⓑ:

$$c = \tfrac{1}{2}(\lambda^{-1} I(\theta)^{-1} \nabla_\theta l(\theta))^T I(\theta) (\lambda^{-1} I(\theta)^{-1} \nabla_\theta l(\theta))$$

$$= \frac{1}{2\lambda^2} \nabla_\theta l(\theta)^T I(\theta)^{-T} I(\theta) I(\theta)^{-1} \nabla_\theta l(\theta)$$

$$= \frac{1}{2\lambda^2} \nabla_\theta l(\theta)^T I(\theta)^{-1} \nabla_\theta l(\theta)$$

$$\Rightarrow \lambda = \frac{1}{\sqrt{2c}} \sqrt{\nabla_\theta l(\theta)^T I(\theta)^{-1} \nabla_\theta l(\theta)}$$

→ plug λ above back to (a), get d^*;

$$d^* = \sqrt{2c} \cdot \left(\nabla_\theta l(\theta)^T I(\theta)^{-1} \nabla_\theta l(\theta)\right)^{-\frac{1}{2}} I(\theta)^{-1} \nabla_\theta l(\theta)$$

the dimension of d^* is easy to know: $d^* \in \mathbb{R}^n$;

(f). Relation to Newton's method. -

Compare the two updates (compare the Δs for both)

natural gradient: $\tilde{d} = \lambda^{-1} \cdot I(\theta)^{-1} \nabla_\theta l(\theta)$;

Newton's method: $\Delta\theta = H^{-1} \nabla_\theta l(\theta)$

The Hessian H is just given by:

$H = \nabla_\theta^2 l(\theta)$;

~~$= Var(y;\eta)\, xx^T$, this is already proved in PS1.4.(c);~~

~~Also in PS1.4.(c) we know:~~

~~$\nabla_\theta l(\theta) =$~~

in ~~PS1~~ PS1.4.(c) we already proved:

$\nabla_\theta^2 l(\theta) = Var(y;\eta)\, xx^T$ this is with natural variable $\eta = \theta^T x$; if ~~we~~ we simple use $\eta = \theta$, we have:

$\nabla_\theta^2 l(\theta) = Var(y;\theta)$ which is just the same as $I(\theta)$;

λ^{-1} is simply a scalar. the direction is the same for both, because:

$I(\theta)^{-1} = H^{-1}$;

4. Semi-supervised EM

4. Semi-supervised EM

(a): PROOF.

For ease of hand writing, we $l(\theta)$ as $l_{semi\text{-}sup}(\theta)$.

Jensen's inequality ~~tells~~ tells us:

$$l(\theta)=\sum_{i=1}^{m}\log\sum_{z^{(i)}}p(x^{(i)},z^{(i)};\theta)+\alpha\left(\sum_{i=1}^{\tilde{m}}\log p(\tilde{x}^{(i)},\tilde{z}^{(i)};\theta)\right)$$

$$=\sum_{i=1}^{m}\log\sum_{z^{(i)}}Q(z^{(i)})\frac{p(x^{(i)},z^{(i)};\theta)}{Q(z^{(i)})}+\alpha\left(\sum_{i=1}^{\tilde{m}}\log p(\tilde{x}^{(i)},\tilde{z}^{(i)};\theta)\right)$$

$$\geq\sum_{i=1}^{m}\sum_{z^{(i)}}Q_i(z^{(i)})\log\frac{p(x^{(i)},z^{(i)};\theta)}{Q_i(z^{(i)})}+\alpha\left(\sum_{i=1}^{\tilde{m}}\log p(\tilde{x}^{(i)},\tilde{z}^{(i)};\theta)\right)$$

it is already shown in lecture note that the tight equality only happenes when:

$$Q_i(z^{(i)})=p(z^{(i)}\mid x^{(i)};\theta),$$ ~~suppose~~

suppose $\theta^{(t)}$ is the current tight point, it means:

$$l(\theta^{(t)})=\sum_{i=1}^{m}\sum_{z^{(i)}}Q_i^{(t)}(z^{(i)})\log\frac{p(x^{(i)},z^{(i)};\theta^{(t)})}{Q_i^{(t)}(z^{(i)})}+\alpha\left(\sum_{i=1}^{\tilde{m}}\log p(\tilde{x}^{(i)},\tilde{z}^{(i)};\theta^{(t)})\right)$$

For ~~any other~~ parameter $\theta^{(t+1)}$ will have

$$l(\theta^{(t+1)})\geq\sum_{i=1}^{m}\sum_{z^{(i)}}Q_i^{(t)}(z^{(i)})\log\frac{p(x^{(i)},z^{(i)};\theta^{(t+1)})}{Q_i^{(t)}(z^{(i)})}+\alpha\left(\sum_{i=1}^{\tilde{m}}\log p(\tilde{x}^{(i)},\tilde{z}^{(i)};\theta^{(t+1)})\right)$$

because the right part of the inequality is a lower bound of $l(\theta)$;

Consider the right part again, because $\theta^{(t+1)}$ is an arg max (g) (g here denotes the above right part). we again have:

$$g\geq\sum_{i=1}^{m}\sum_{z^{(i)}}Q_i^{(t)}(z^{(i)})\log\frac{p(x^{(i)},z^{(i)};\theta^{(t)})}{Q_i^{(t)}(z^{(i)})}+\alpha\left(\sum_{i=1}^{\tilde{m}}\log p(\tilde{x}^{(i)},\tilde{z}^{(i)};\theta^{(t)})\right)$$

$$=l(\theta^{(t)})$$

So. $l(\theta^{(t+1)})\geq l(\theta^{(t)})$. Done!

~~(b)~~. Semi-supervised GMM

(b). $z^{(i)}$s are the latent randoms which can take value $\{1, 2, \dots k\}$,

The updated values are:

$$w_j^{(i)} = Q_i(z^{(i)}=j) = p(z^{(i)}=j \mid x^{(i)}; \phi, \mu_j, \Sigma_j)$$

$$= \frac{p(x^{(i)} \mid z^{(i)}=j; \mu_j, \Sigma_j)\, p(z^{(i)}=j; \phi)}{\sum_{l=1}^{k} p(x^{(i)} \mid z^{(i)}=l; \mu_l, \Sigma_l)\, p(z^{(i)}=l; \phi)}$$

$$= \frac{\frac{1}{\sqrt{2\pi}|\Sigma_j|} \exp\left(-\frac{1}{2}(x^{(i)}-\mu_j)^T \Sigma_j^{-1}(x^{(i)}-\mu_j)\right) \cdot \phi_j}{\sum_{l=1}^{k} \frac{1}{\sqrt{2\pi}|\Sigma_l|} \exp\left(-\frac{1}{2}(x^{(i)}-\mu_l)^T \Sigma_l^{-1}(x^{(i)}-\mu_l)\right)\phi_l}$$

$$= \frac{\frac{1}{\sqrt{2\pi}|\Sigma_j|} \exp\left(-\frac{1}{2}(x^{(i)}-\mu_j)^T \Sigma_j^{-1}(x^{(i)}-\mu_j)\right) \phi_j}{\sum_{l=1}^{k} \frac{1}{\sqrt{2\pi}|\Sigma_l|} \exp\left(-\frac{1}{2}(x^{(i)}-\mu_l)^T \Sigma_l^{-1}(x^{(i)}-\mu_l)\right)\phi_l}$$

(The parameters are: $\mu_1, \mu_2 \dots \mu_k, \Sigma_1, \Sigma_2, \dots \Sigma_k, \phi_1, \phi_2 \dots \phi_k$)

(c) in M-step, just plug $w_j^{(i)}$ to $l(\theta)$ and take derivatives w.r.t μ, Σ and ϕ. The first part is the same as it is in lecture notes. and we also need to plus the supervised part:

$$l(\theta) = \sum_{i=1}^{m} \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)}; \phi, \mu, \Sigma)}{Q_i(z^{(i)})} + \alpha\left(\sum_{i=1}^{\tilde{m}} \log p(\tilde{x}^{(i)}, \tilde{z}^{(i)}; \tilde{\mu}, \tilde{\phi}, \tilde{\Sigma}\right)$$

the above parameters are without subscripts, ~~so add~~ expand and add subscript:

$$l(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{k} w_j^{(i)} \log\left(\frac{1}{(2\pi)^{n/2}|\Sigma_j|^{1/2}} \exp\left(-\frac{1}{2}(x^{(i)}-\mu_j)^T \Sigma_j^{-1}(x^{(i)}-\mu_j)\right)\phi_j / w_j^{(i)}\right) +$$

$$\alpha\left(\sum_{i=1}^{\tilde{m}} \log p(\tilde{x}^{(i)}, \tilde{z}^{(i)}; \phi_{\tilde{z}^{(i)}}, \mu_{\tilde{z}^{(i)}}, \Sigma_{\tilde{z}^{(i)}}\right)$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{k} w_j^{(i)} \log\left(\frac{1}{(2\pi)^{n/2}|\Sigma_j|^{1/2}} \exp\left(-\frac{1}{2}(x^{(i)}-\mu_j)^T \Sigma_j^{-1}(x^{(i)}-\mu_j)\right)\phi_j / w_j^{(i)}\right) +$$

$$\alpha\left(\sum_{i=1}^{\tilde{m}} \log\left(\frac{1}{(2\pi)^{n/2}|\Sigma_{\tilde{z}^{(i)}}|^{1/2}} \exp\left(-\frac{1}{2}(\tilde{x}^{(i)}-\mu_{\tilde{z}^{(i)}})^T \Sigma_{\tilde{z}^{(i)}}^{-1}(\tilde{x}^{(i)}-\mu_{\tilde{z}^{(i)}})\right)\phi_{\tilde{z}^{(i)}}\right)\right)$$

take the derivative w.r.t μ_l, the first part is just the same as lecture notes. and the second part can be written out similar to the first part:

$$\nabla_{\mu_l} = \sum_{i=1}^{m} w_l^{(i)}(\Sigma_l^{-1} x^{(i)} - \Sigma_l^{-1}\mu_l) + \sum_{i=1}^{\tilde{m}} 1\{\tilde{z}^{(i)}=l\}(\Sigma_{\tilde{z}^{(i)}}^{-1}\tilde{x}^{(i)} - \Sigma_{\tilde{z}^{(i)}}^{-1}\mu_{\tilde{z}^{(i)}})$$

Set it to $\vec{0}$ and left multiply with Σ_l give us:

$$\sum_{i=1}^{m} w_l^{(i)}(x^{(i)}-\mu_l) + \alpha\sum_{i=1}^{\tilde{m}} 1\{\tilde{z}^{(i)}=l\}(\tilde{x}^{(i)}-\mu_l) = \vec{0}$$

$$\mu_l = \frac{\sum_{i=1}^{m} w_l^{(i)} x^{(i)} + \alpha\sum_{i=1}^{\tilde{m}} 1\{\tilde{z}^{(i)}=l\}\tilde{x}^{(i)}}{\sum_{i=1}^{m} w_l^{(i)} + \alpha\sum_{i=1}^{\tilde{m}} 1\{\tilde{z}^{(i)}=l\}}$$

~~Take derivative w.r.t ϕ_j~~ :

Now derive the update rule for ϕ_j. Group all terms that depend on ϕ_j, we have the following:

$$\sum_{i=1}^{m}\sum_{j=1}^{k} w_j^{(i)} \log \phi_j + \alpha \sum_{i=1}^{\tilde m} \log \phi_{\tilde z^{(i)}}$$

Similar to lecture notes, the Lagrangian is:

$$L(\phi) = \sum_{i=1}^{m}\sum_{j=1}^{k} w_j^{(i)} \log \phi_j + \alpha \sum_{i=1}^{\tilde m} \log \phi_{\tilde z^{(i)}} + \beta\left(\sum_{j=1}^{k} \phi_j - 1\right)$$

take derivative w.r.t. ϕ_j:

$$\frac{\partial}{\partial \phi_j} = \sum_{i=1}^{m} \frac{w_j^{(i)}}{\phi_j} + \alpha \sum_{i=1}^{\tilde m} \frac{1\{\tilde z^{(i)} = j\}}{\phi_j} + \beta = 0 \Rightarrow$$

$$\phi_j = \frac{\sum_{i=1}^{m} w_j^{(i)} + \alpha \sum_{i=1}^{\tilde m} 1(\tilde z^{(i)} = j)}{-\beta}$$

Use the fact that $\sum_{j=1}^{k} \phi_j = 1$, its easy to come to:

$$-\beta = \sum_{i=1}^{m}\sum_{j=1}^{k} w_j^{(i)} + \alpha \sum_{i=1}^{\tilde m} 1\{\tilde z^{(i)} \text{ in } \{1, 2, \dots k\}\}$$
$$= m + \alpha \tilde m,$$

Therefore:

$$\phi_j = \frac{\sum_{i=1}^{m} w_j^{(i)} + \alpha \sum_{i=1}^{\tilde m} 1\{\tilde z^{(i)} = j\}}{m + \alpha \tilde m}.$$

Finally lets do Σ_l. Go to next page. →

Terms related to Σ_j's:

$$l(\theta) = \sum_{i=1}^{m}\sum_{j=1}^{k} w_j^{(i)}\left(-\frac{1}{2}(x^{(i)}-\mu_j)^T\Sigma_j^{-1}(x^{(i)}-\mu_j) - \log\left(\cancel{(2\pi)^{n/2}}|\Sigma_j|^{1/2}\right)\right)$$

$$+\alpha\left(\sum_{i=1}^{\tilde m}\left(-\frac{1}{2}(\tilde x^{(i)}-\mu_{\tilde z^{(i)}})^T\Sigma_{\tilde z^{(i)}}^{-1}(\tilde x^{(i)}-\mu_{\tilde z^{(i)}}) - \log\left(\cancel{(2\pi)^{n/2}}|\Sigma_{\tilde z^{(i)}}|^{1/2}\right)\right)\right)$$

$$= -\sum_{i=1}^{m}\sum_{j=1}^{k} w_j^{(i)}\left(\frac{1}{2}(x^{(i)}-\mu_j)^T\Sigma_j^{-1}(x^{(i)}-\mu_j) + \frac{1}{2}\log(|\Sigma_j|)\right)$$

$$-\alpha\left(\sum_{i=1}^{\tilde m}\left(\frac{1}{2}(\tilde x^{(i)}-\mu_{\tilde z^{(i)}})^T\Sigma_{\tilde z^{(i)}}^{-1}(\tilde x^{(i)}-\mu_{\tilde z^{(i)}}) + \frac{1}{2}\log(|\Sigma_{\tilde z^{(i)}}|)\right)\right)$$

take derivatie w.r.t Σ_l, it's still an $n\times n$ matrix:

$$\nabla_{\Sigma_l} = \frac{1}{2}\sum_{i=1}^{m}\left((x^{(i)}-\mu_l)(x^{(i)}-\mu_l)^T\Sigma_l^{-2} - \Sigma_l^{-1}\right)w_l^{(i)}$$

$$+\frac{1}{2}\alpha\left(\sum_{i=1}^{\tilde m} 1\{\tilde z^{(i)}=l\}(\tilde x^{(i)}-\mu_l)(\tilde x^{(i)}-\mu_l)^T\Sigma_l^{-2} - \sum_{i=1}^{\tilde m}1\{\tilde z^{(i)}=l\}\Sigma_l^{-1}\right)$$

$$= \frac{1}{2}\left(\sum_{i=1}^{m} w_l^{(i)}(x^{(i)}-\mu_l)(x^{(i)}-\mu_l)^T\Sigma_l^{-2} - \sum_{i=1}^{m} w_l^{(i)}\Sigma_l^{-1}\right)$$

$$+\frac{1}{2}\alpha\left(\sum_{i=1}^{\tilde m} 1\{\tilde z^{(i)}=l\}(\tilde x^{(i)}-\mu_l)(\tilde x^{(i)}-\mu_l)^T\Sigma_l^{-2} - \sum_{i=1}^{\tilde m}1\{\tilde z^{(i)}=l\}\Sigma_l^{-1}\right)$$

Set it to $\vec{0}$ and right multiply with Σ_l^2:

$$\sum_{i=1}^{m} w_l^{(i)}(x^{(i)}-\mu_l)(x^{(i)}-\mu_l)^T + \alpha\sum_{i=1}^{\tilde m} 1\{\tilde z^{(i)}=l\}(\tilde x^{(i)}-\mu_l)(\tilde x^{(i)}-\mu_l)^T$$

$$= \left(\sum_{i=1}^{m} w_l^{(i)} + \sum_{i=1}^{\tilde m} 1\{\tilde z^{(i)}=l\}\right)\Sigma_l$$

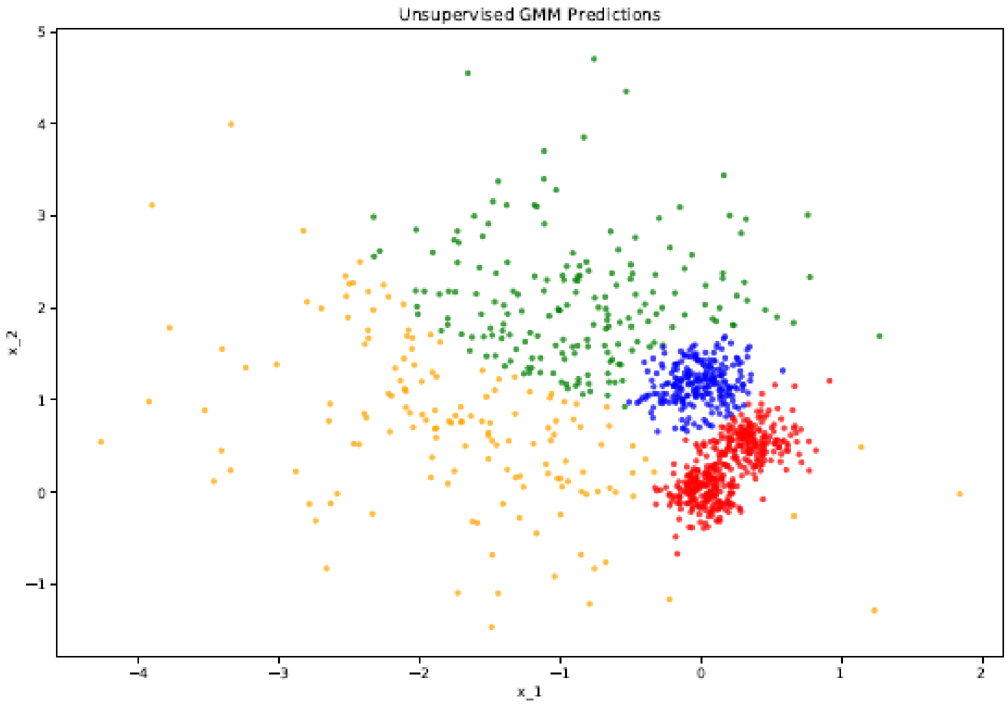
$$\Rightarrow \Sigma_l = \frac{\sum_{i=1}^{m} w_l^{(i)}(x^{(i)}-\mu_l)(x^{(i)}-\mu_l)^T + \alpha\sum_{i=1}^{\tilde m} 1\{\tilde z^{(i)}=l\}(\tilde x^{(i)}-\mu_l)(\tilde x^{(i)}-\mu_l)^T}{\sum_{i=1}^{m} w_l^{(i)} + \alpha\sum_{i=1}^{\tilde m} 1\{\tilde z^{(i)}=l\}}$$

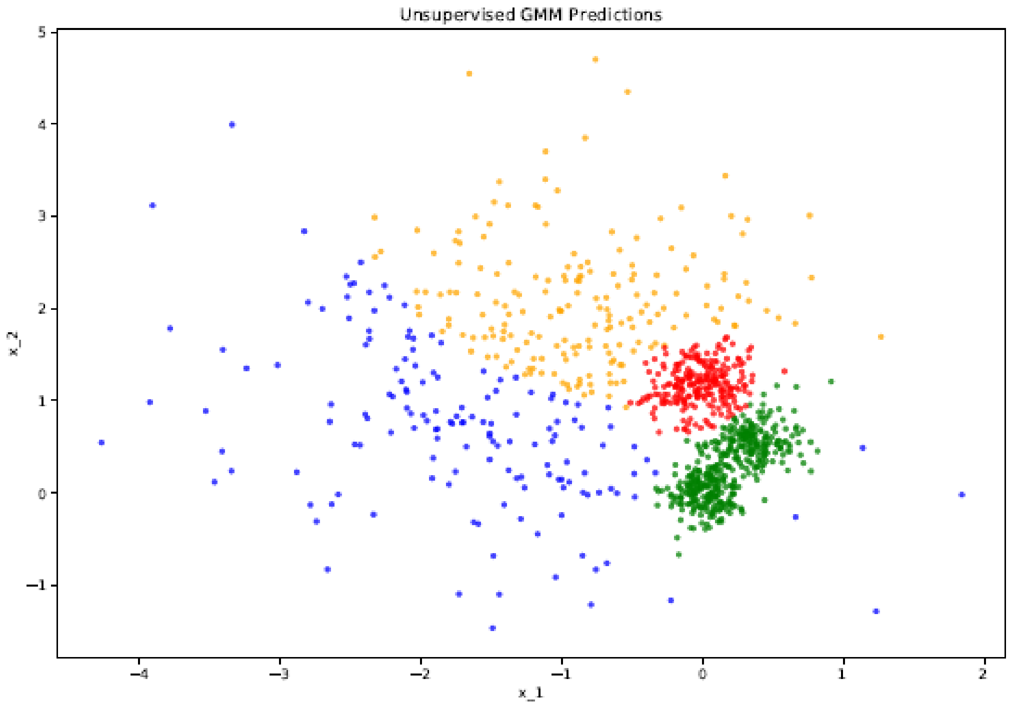
the denominator is a scalar, the mumoritor is an $n\times n$ matrix.
Done!

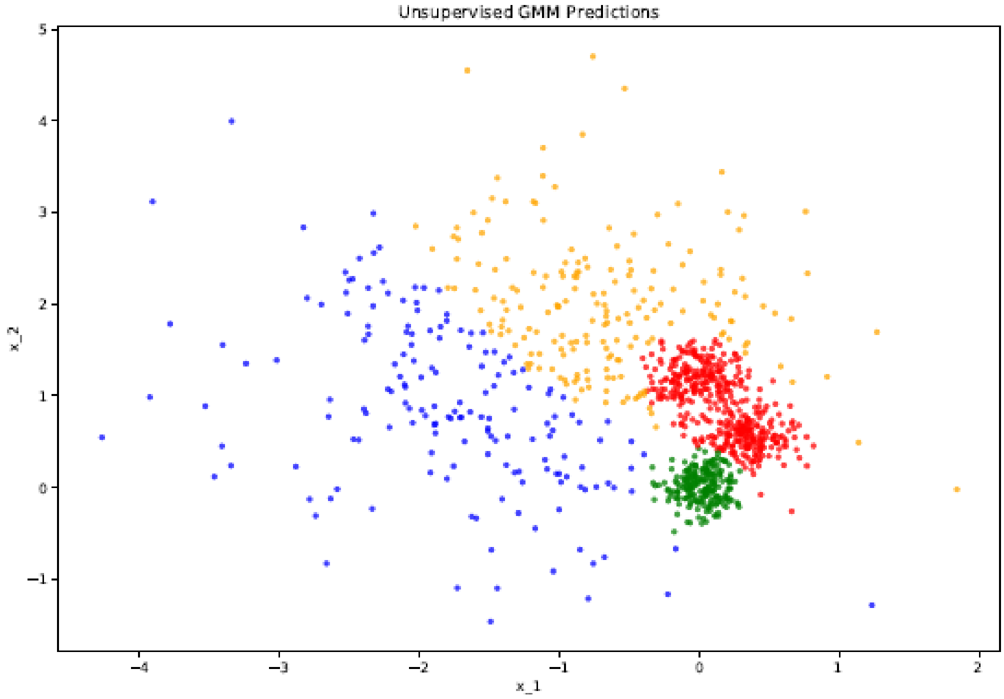
(d) plot.

(e) plot.

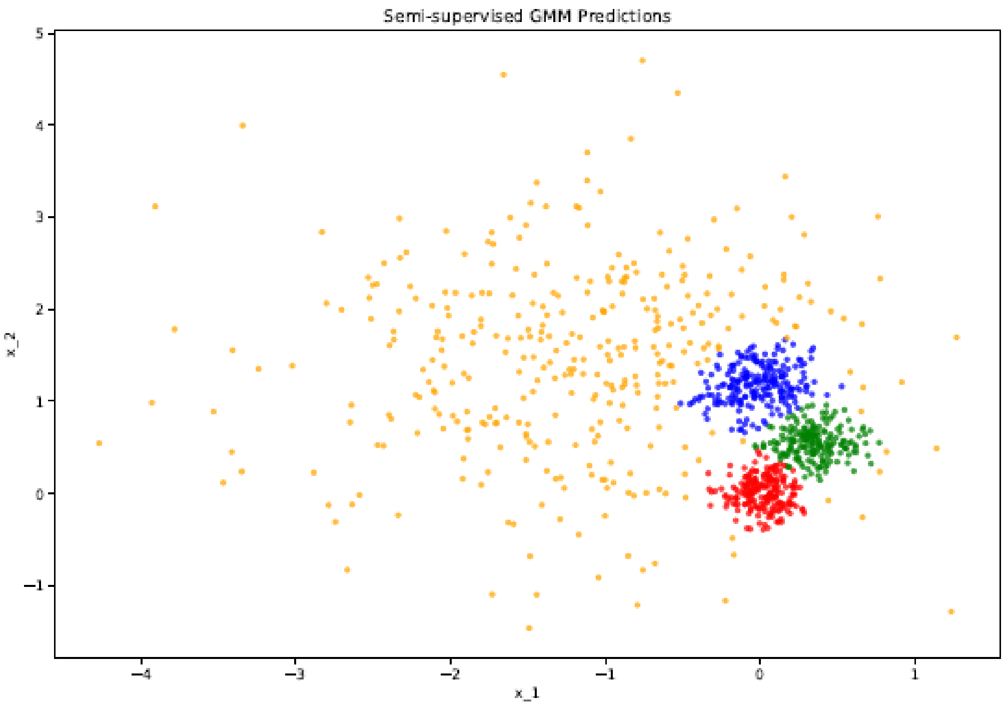
d) Unsupervised GMM predictions

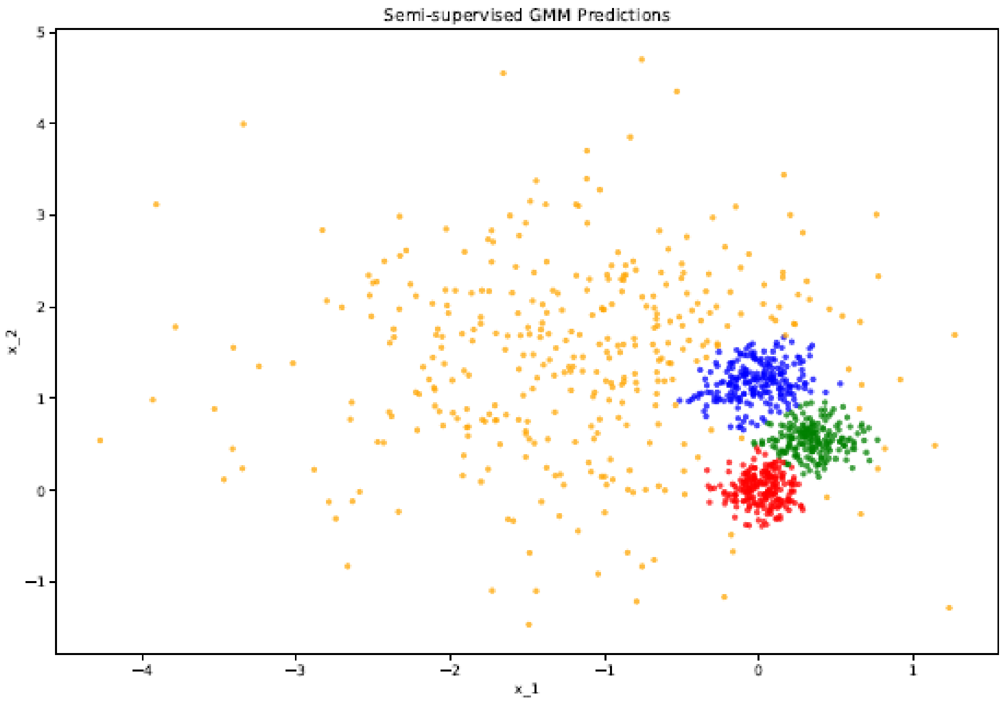




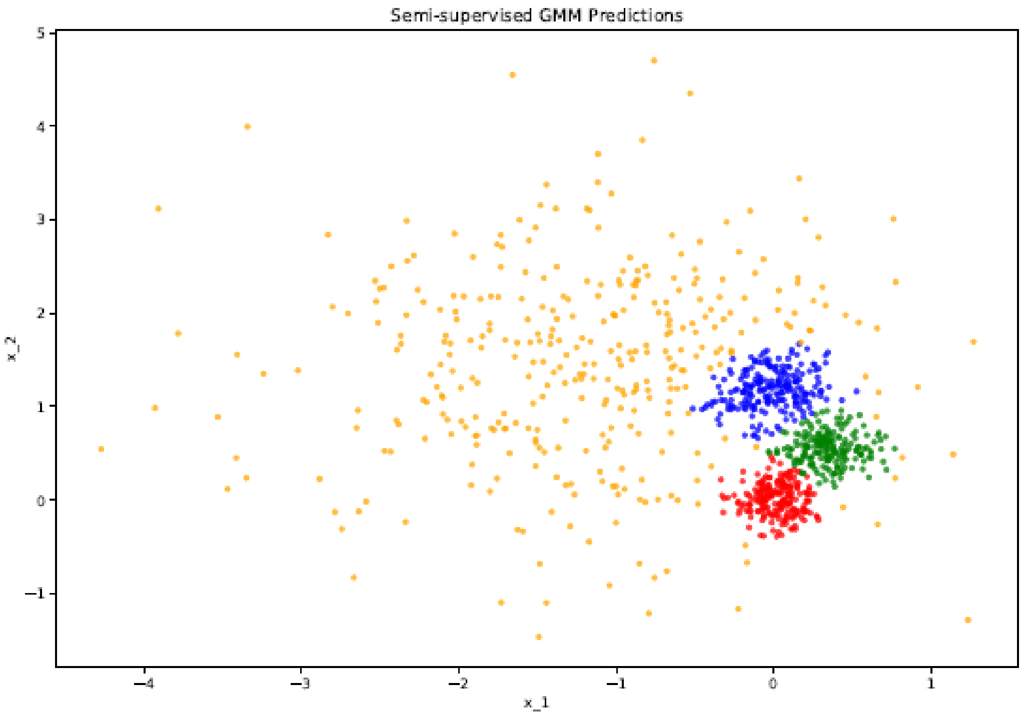


e) Semi-supervised GMM predictions:





Semi-supervised GMM Predictions
x_1
x_2



Semi-supervised GMM Predictions
x_1
x_2

f) Comparison of Unsupervised and Semi-supervised EM

i) Number of iterations taken to converge

Answer:

Unsupervised EM takes 60 ~ 100 iterations to converge;

Semi-supervised EM only takes 20~30 iterations to converge.

ii) Stability

Answer:

As can be seen in the above plot, Unsupervised EM is very sensitive to initialization and the class assignment changes dramatically with different random initializations. Semi-supervised EM is very stable and class assignments do not change much.

iii) Overall quality of assignments

Answer:

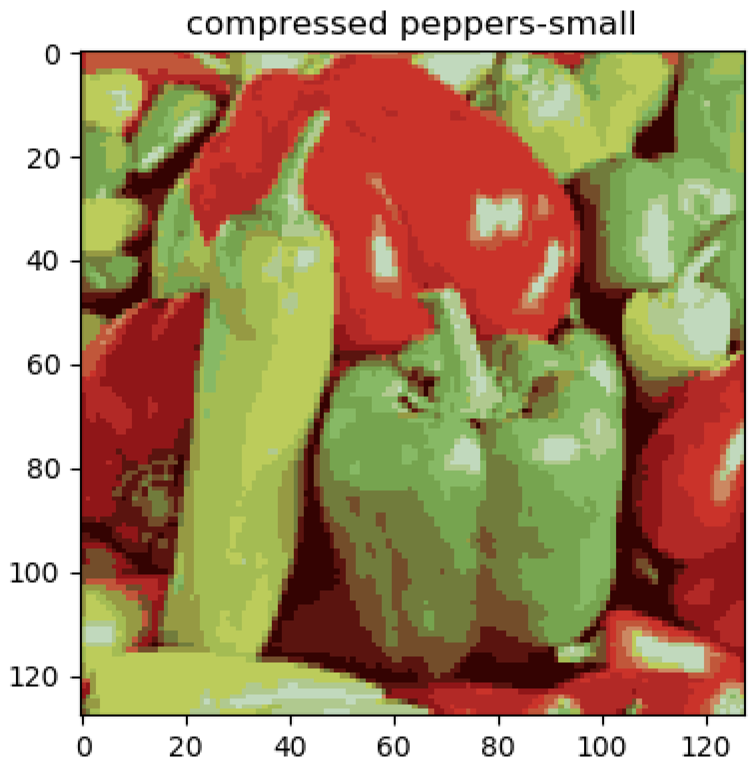
Overall the Semi-supervised EM has a much better quality compared to Unsupervised EM.

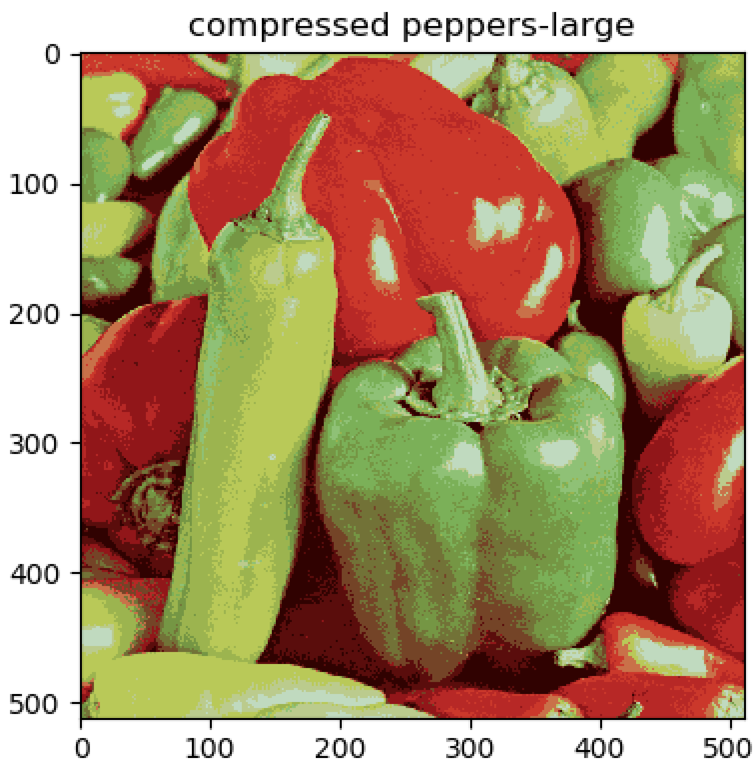
5. K-means for compression.

a) K-means compression implementation.

```
from matplotlib.image import imread
import matplotlib.pyplot as plt
import numpy as np

# reduce to n colors
def kmeans_compress(X, n = 16):
    X = X.reshape(-1, 3)
    centroids = X[np.random.choice(np.arange(X.shape[0]), size=n, replace=False)]
    centers = np.zeros([n, 3])
    #
    while not np.array_equal(centers, centroids):
        centers = centroids
        norms = [np.sqrt(np.sum((centers[i] - X) ** 2, axis=1)) for
            i in range(0, len(centers))]
        idxs = np.stack(norms).argmin(axis=0)
        #recalc centroids
        centroids = np.array([np.mean(X[np.where(idxs == i,True,False)],
            axis = 0).round() for i in range(n)])
    #you need the data type conversion, otherwise you are in big trouble!!
    return np.uint8(centroids), np.uint8(idxs)

def show_img(X, ax, s):
    ax.imshow(X)
    ax.set_title(s)

def main():
    fig, axes = plt.subplots(2, 1, figsize=(12, 4))
    axb1, axb2 = axes.ravel()

    A1 = imread('..\data\peppers-small.tiff')
    centroids,idxs = kmeans_compress(A1)
    B1 = np.array([centroids[idxs[i]] for i in range(0, len(idxs))])
    B1 = B1.reshape(A1.shape)
    show_img(B1, axb1,'compressed peppers-small')

    A2 = imread('..\data\peppers-large.tiff')
    centroids,idxs = kmeans_compress(A2)
    B2 = np.array([centroids[idxs[i]] for i in range(0, len(idxs))])
    B2 = B2.reshape(A2.shape)
    show_img(B2, axb2, 'compressed peppers-small')

    plt.show()

if __name__ == '__main__':
    main()
```

Compressed images:





b) K-means compression Factor

Answer:

To store the original image, each pixel requires 24bit for the RGB value (255 x 255 x 255 colors);

To store the compressed image, each pixel will only need 16 colors, which need only 4bit storage.

The compression factor is thus 24:4 = 6:1.