IFT 6369, Devoir 2

Adel Nabli, ID: 20121744

03 Octobre 2018

1 Generative model (Fisher LDA)

1. On a deux variables aléatoires X et Y tel que $Y \sim Bernouilli(\pi)$ et $X|Y = j \sim \mathcal{N}(\mu_j, \Sigma)$. Nous avons à notre disposition une collection $\mathcal{D} = (x_n, y_n)_{n \in [\![1,N]\!]}$ de réalisations **i.i.d** de ces variables aléatoires.

   Nous voulons estimer le paramètre $\theta = (\pi,\ \mu_0,\ \mu_1,\ \Sigma)$ en utilisant l'estimateur de maximum de vraissemblance. Pour cela, nous allons commencer par écrire la log-vraissemblance:

$$\begin{aligned} l(\theta|\mathcal{D}) &= \log[\prod_{n=1}^{N} p(y_n|\pi)p(x_n|y_n, \mu_0, \mu_1, \Sigma)] \\ &= \underbrace{\sum_{n=1}^{N} \log p(y_n|\pi)}_{f(\pi)} + \underbrace{\sum_{n=1}^{N} \log p(x_n|y_n, \mu_0, \mu_1, \Sigma)}_{g(\mu_0,\mu_1,\Sigma)} \end{aligned}$$

   En remarquant que le paramètre π n'intervient que dans la première somme de l'équation, on peut écrire:

$$\hat{\pi} = \underset{\pi}{\text{argmax}} \sum_{n=1}^{N} \log p(y_n|\pi) = \underset{\pi}{\text{argmax}} \sum_{n=1}^{N} y_n \log(\pi) + (1 - y_n) \log(1 - \pi)$$

   La fonction f de π qui apparaît étant **concave**, on peut la maximiser en trouvant son point critique.

$$f'(\pi) = \sum_{n=1}^{N} \frac{y_n}{\pi} - \frac{1 - y_n}{1 - \pi} = \frac{-N}{1 - \pi} + \frac{\sum_{n=1}^{N} y_n}{\pi(1 - \pi)}$$

   Ainsi, en résolvant $f'(\pi) = 0$, on trouve $\hat{\pi} = \dfrac{\sum_{n=1}^{N} y_n}{N}$.

   Pour trouver les estimateurs pour le reste des paramètres, nous allons pouvoir nous intéresser uniquement à la fonction g vu qu'ils n'interviennent pas dans la première somme:

$$
\begin{aligned}
g(\mu_0,\mu_1,\Sigma) &= \sum_{n=1}^{N} \log[p(x_n|y_n=1,\mu_0,\mu_1,\Sigma)^{y_n} p(x_n|y_n=0,\mu_0,\mu_1,\Sigma)^{1-y_n}] \\
&= \sum_{n=1}^{N} \{y_n(-\log(2\pi) - \frac{1}{2}\log(\det(\Sigma)) - \frac{1}{2}(x_n-\mu_1)^T\Sigma^{-1}(x_n-\mu_1))\} \\
&+ \sum_{n=1}^{N} \{(1-y_n)(-\log(2\pi) - \frac{1}{2}\log(\det(\Sigma)) - \frac{1}{2}(x_n-\mu_0)^T\Sigma^{-1}(x_n-\mu_0))\} \\
&= -N\log(2\pi) + \frac{N}{2}\log(\det(\Sigma^{-1})) \\
&- \frac{1}{2}\sum_{n=1}^{N} y_n(x_n-\mu_1)^T\Sigma^{-1}(x_n-\mu_1) - \frac{1}{2}\sum_{n=1}^{N}(1-y_n)(x_n-\mu_0)^T\Sigma^{-1}(x_n-\mu_0)
\end{aligned}
$$

Ainsi, en utilisant l'expression du gradient de la fonction $x \mapsto x^T Ax$ et le fait que Σ^{-1} soit symétrique, on a:

$$
\begin{cases}
\nabla_{\mu_1}(g) = -\Sigma^{-1}\sum_{n=1}^{N} y_n(\mu_1 - x_n) \\
\nabla_{\mu_0}(g) = -\Sigma^{-1}\sum_{n=1}^{N}(1-y_n)(\mu_0 - x_n)
\end{cases}
$$

Donc, en écrivant $\nabla_{\mu_1}(g) = \nabla_{\mu_0}(g) = 0$, on trouve les estimateurs $\hat{\mu_1} = \dfrac{\sum_{n=1}^{N} x_n y_n}{\sum_{n=1}^{N} y_n}$
et $\hat{\mu_0} = \dfrac{\sum_{n=1}^{N} x_n(1-y_n)}{N - \sum_{n=1}^{N} y_n}$.
Cela se comprend comme des moyennes empiriques sur les sous ensembles de points labellisés "1" et "0".

Pour estimer le dernier paramètre, on va remarquer que les deux sommes qui apparaissent dans l'expression de g sont des sommes de scalaires et donc égales à leur traces.

Or, en posant $\tilde{\Sigma_1} = \dfrac{1}{N}\sum_{n=1}^{N} y_n(x_n-\mu_1)(x_n-\mu_1)^T$ et $\tilde{\Sigma_0} = \dfrac{1}{N}\sum_{n=1}^{N}(1-y_n)(x_n-\mu_0)(x_n-\mu_0)^T$, on peut ré-écrire notre expression sous la forme:

$$
g(\mu_0,\mu_1,\Sigma) = -N\log(2\pi) + \frac{N}{2}\log(\det(\Sigma^{-1})) - \frac{N}{2}\mathbf{Tr}(\Sigma^{-1}\tilde{\Sigma_1}) - \frac{N}{2}\mathbf{Tr}(\Sigma^{-1}\tilde{\Sigma_0})
$$

Ainsi, en utilisant l'expression des gradients des fonctions $A \mapsto \log(\det(A))$ et $A \mapsto \mathbf{Tr}(A)$, on a:

$$
\nabla_{\Sigma^{-1}}(g) = \frac{N}{2}\Sigma - \frac{N}{2}(\tilde{\Sigma_1} + \tilde{\Sigma_0})
$$

Et on trouve donc $\hat{\Sigma} = \dfrac{1}{N}\sum_{n=1}^{N} y_n(x_n-\hat{\mu_1})(x_n-\hat{\mu_1})^T + \dfrac{1}{N}\sum_{n=1}^{N}(1-y_n)(x_n-\hat{\mu_0})(x_n-\hat{\mu_0})^T$.

2. On a:

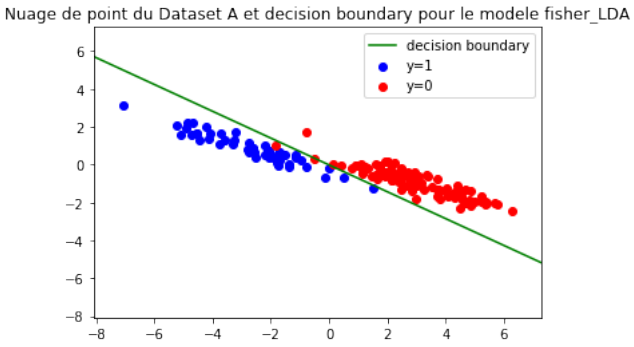
$$
\begin{aligned}
p(y=1|x,\theta) &= \frac{p(x|y=1,\theta)p(y=1|\pi)}{p(x|y=1,\theta)p(y=1|\pi)+p(x|y=0,\theta)p(y=0|\pi)} \\
&= \frac{\pi \exp\Big(-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\Big)}{\pi \exp\Big(-\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\Big)+(1-\pi)\exp\Big(-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)\Big)} \\
&= \frac{1}{1+\frac{1-\pi}{\pi}\exp\Big(-\frac{1}{2}(x-\mu_0)^T\Sigma^{-1}(x-\mu_0)+\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)\Big)} \\
&= \frac{1}{1+\exp\Big(-(\mu_1-\mu_0)^T\Sigma^{-1}x+\frac{1}{2}(\mu_1-\mu_0)^T\Sigma^{-1}(\mu_0-\mu_1)+\log(\frac{1-\pi}{\pi})\Big)}
\end{aligned}
$$

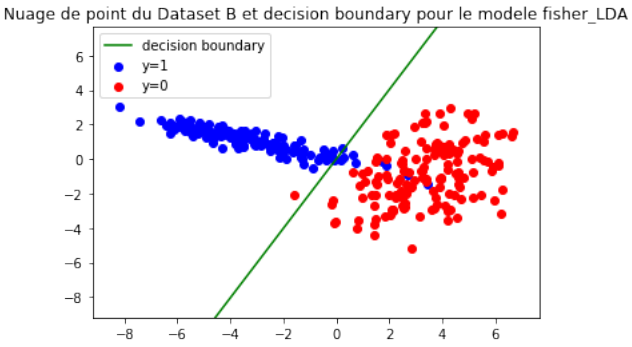
Ainsi, en posant $\beta^T = (\mu_1-\mu_0)^T\Sigma^{-1}$ et $\gamma = \frac{1}{2}(\mu_1-\mu_0)^T\Sigma^{-1}(\mu_1+\mu_0)+\log(\frac{1-\pi}{\pi})$, on a:

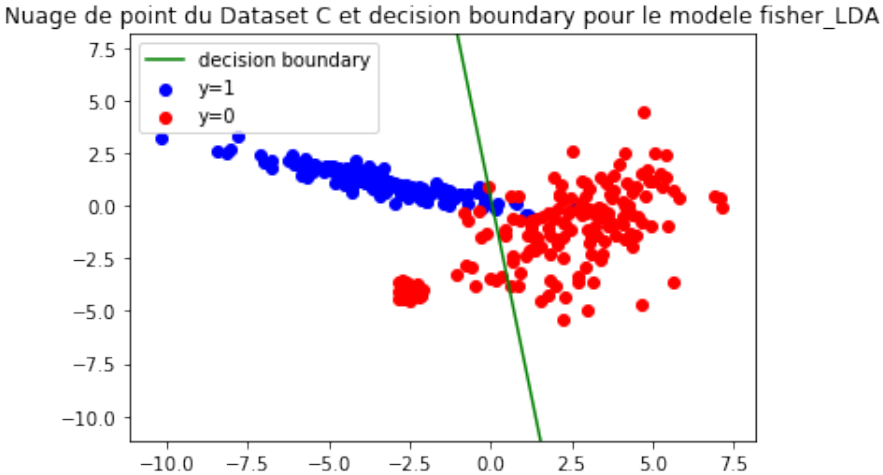
$$
p(y=1|x,\theta) = \frac{1}{1+\exp\Big(-\beta^T x+\gamma\Big)}
$$

et on retrouve donc une expression de la même forme que celle d'une régression logistique.

3. Après avoir implémenté le modèle (voir *Notebook*), nous avons les graphiques suivants:





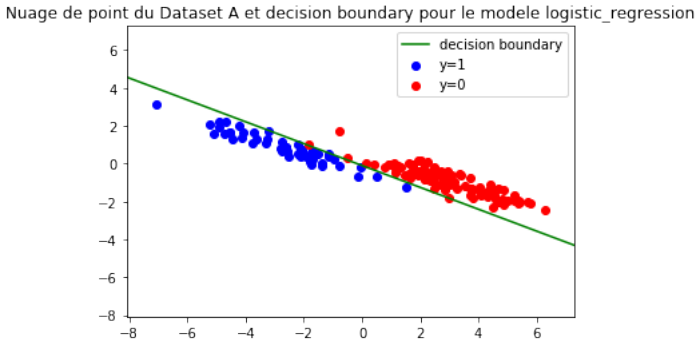


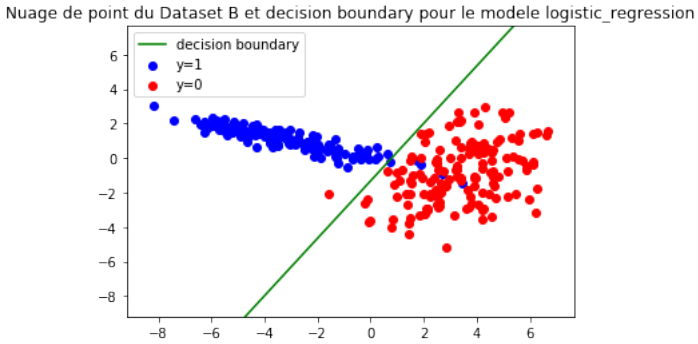
2 Logistic regression

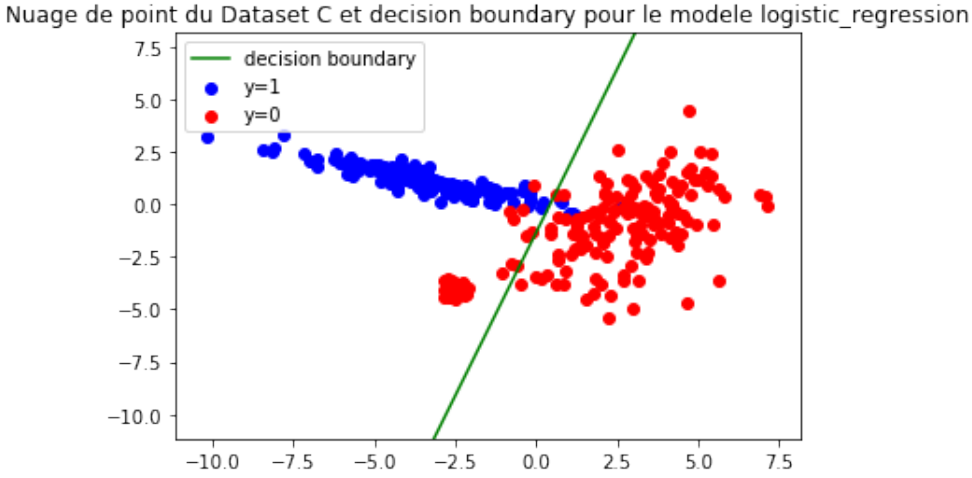
Nous sommes maintenant dans le cas où $p(y=1|x) = \sigma\big(f(x)\big)$ avec σ la fonction sigmoïde et $f(x) = \omega^T x + b$. Nous cherchons donc les valeurs du vecteur ω et celle du scalaire b. On trouve, après implémentation (voir *Notebook*):

	Dataset A	**Dataset B**	**Dataset C**
ω	$\begin{pmatrix}-189.31\\-327.60\end{pmatrix}$	$\begin{pmatrix}-1.71\\1.02\end{pmatrix}$	$\begin{pmatrix}-2.20\\0.71\end{pmatrix}$
b	-31.84	1.35	0.96

Cela donne les graphiques suivants:





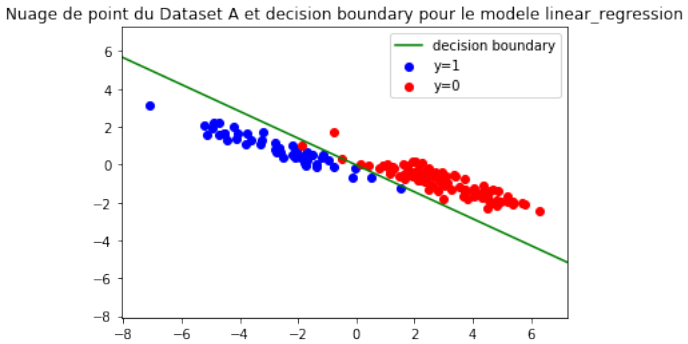


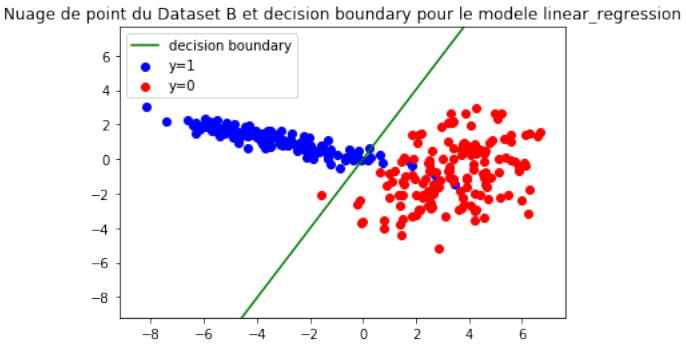
3 Linear regression

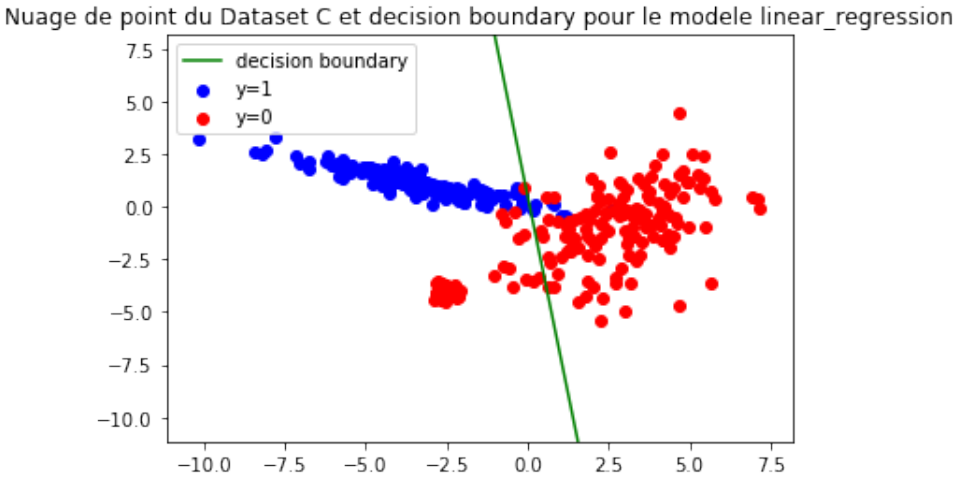
Nous sommes maintenant dans le cas où $p(y=1|x) = f(x) = \omega^T x + b$. Nous cherchons donc les valeurs du vecteur ω et celle du scalaire b. On trouve, après implémentation (voir *Notebook*):

	Dataset A	**Dataset B**	**Dataset C**
ω	$\begin{pmatrix}-0.26\\-0.37\end{pmatrix}$	$\begin{pmatrix}-0.10\\0.05\end{pmatrix}$	$\begin{pmatrix}-0.13\\-0.02\end{pmatrix}$
b	0.49	0.50	0.51

Cela donne les graphiques suivants:







4 Performances

Après calcul des différents taux d'erreurs de classifications pour les différents modèles et jeux de données (voir *Notebook*), nous pouvons fournir le tableau suivant:

	Dataset A		**Dataset B**		**Dataset C**	
	Train	Test	Train	Test	Train	Test
Fisher LDA	$\frac{2}{150} \simeq 0.013$	$\frac{30}{1500} = 0.02$	$\frac{9}{300} = 0.03$	$\frac{83}{2000} = 0.0415$	$\frac{22}{400} = 0.055$	$\frac{127}{3000} \simeq 0.042$
Logistic regression	$\frac{0}{150} = 0$	$\frac{52}{1500} \simeq 0.035$	$\frac{6}{300} = 0.02$	$\frac{86}{2000} = 0.043$	$\frac{16}{400} = 0.04$	$\frac{68}{3000} \simeq 0.023$
Linear regression	$\frac{2}{150} \simeq 0.013$	$\frac{32}{1500} \simeq 0.021$	$\frac{10}{300} \simeq 0.033$	$\frac{84}{2000} = 0.042$	$\frac{25}{400} = 0.0625$	$\frac{138}{3000} = 0.046$

Tablaux regroupant les taux d'erreurs de classifications observés

- Sur le **Dataset A**, on remarque que c'est la *régression logistique* qui performe le mieux sur les données d'entraînement mais c'est aussi ce modèle qui performe le moins bien sur les données test: il semblerait donc que la régression logistique sur-apprend sur cette distribution de données en collant de trop prêt les données d'entraînement.

  Les deux autres modèles donnent des résultats similaires à l'entraînement et en test.

On remarque cependant que le modèle performant le mieux en test est le modèle *Fisher LDA*, ce qui est compréhensible vu que les données du Dataset A sont des réalisations d'une loi de probabilité qui a exactement les mêmes caractéristiques que les hypothèses utilisées pour obtenir le modèle Fisher LDA, on peut donc s'attendre à ce qu'il se généralise bien face à de nouvelles données (le modèle appris devrait être proche de celui qui a généré les données).

- Sur le **Dataset B**, tous les modèles ont l'air de se comporter de la même manière et créent peu ou prou les même *decision boundaries*. Au vu de la manière dont a été généré le jeux de donnés, on peut s'attendre à ce que le modèle QDA performera mieux que les trois listés ci-dessus.

- Sur le **Dataset C**, seule la *régression logistique* semble arriver à produire une *decision boundary* qui réussit un tant soit peu à capturer la complexité du modèle générant les données (il y a deux centroïdes pour les points labellisés "y=0") en produisant une droite inclinée dans le "bon sens". Les deux autres modèles échouent de manière comparable sur ce jeux de données.

  Cependant, on remarque pour les 3 modèles que les taux d'erreurs en entraînement sont **supérieurs** à ceux observés en phase de test, ce qui peut paraître contre-intuitif au premier abord. On peut expliquer ce fait en supposant que le modèle qui a généré les données du Dataset C ne génère qu'avec une faible probabilité des points situés au niveau du centroïde *"non capturé par les modèles"*. Ainsi, en remarquant que les données de test sont en nombre $10\times$ plus élevés que les données d'entraînements, on peut supposer que les points présents dans le "centroïde non capturé" dans le *train set* sont anormalement sur-représentés (comparé à la réelle probabilité pour un point d'être généré à cet endroit) dû au faible nombre de tirages totaux dans le train set. Ce "bruit" est ainsi lissé en faisant un plus grand nombre de tirages, ce qu'il se passe dans le test set. Ainsi, la proportion de *"points bien capturés par les modèles"* augmente sur le test set.

5 QDA model

Cette fois, nous avons deux variables aléatoires X et Y tel que $Y \sim Bernouilli(\pi)$ et $X|Y = j \sim \mathcal{N}(\mu_j, \Sigma_j)$, ce qui donne le paramètre $\theta = (\pi,\ \mu_0,\ \mu_1,\ \Sigma_1,\ \Sigma_2)$ à estimer par maximum de vraissemblance. Le cas étant très proche de ce qui a déjà été calculé pour la question 1, nous reprendrons directement certains des résultats montrés précédemment. Ainsi:

$$
\begin{aligned}
l(\theta|\mathcal{D}) = &\sum_{n=1}^{N} \Big(y_n \log(\pi) + (1-y_n)\log(1-\pi) \Big) \\
&- N\log(2\pi) + \frac{1}{2}\sum_{n=1}^{N} y_n \log(\det(\Sigma_1^{-1})) + \frac{1}{2}\sum_{n=1}^{N}(1-y_n)\log(\det(\Sigma_0^{-1})) \\
&- \frac{1}{2}\sum_{n=1}^{N} y_n (x_n-\mu_1)^T\Sigma_1^{-1}(x_n-\mu_1) - \frac{1}{2}\sum_{n=1}^{N}(1-y_n)(x_n-\mu_0)^T\Sigma_0^{-1}(x_n-\mu_0)
\end{aligned}
$$

On remarque ainsi que nous pouvons reprendre sans problèmes les estimateurs déjà calculés dans le cas *Fisher LDA* pour π, μ_0 et μ_1, les seuls estimateurs qu'il nous reste à trouver sont donc ceux de Σ_0 et Σ_1.

En utilisant de nouveau les gradients de fonctions connus et les matrices $\tilde{\Sigma_0}$ et $\tilde{\Sigma_1}$, on trouve:

$$
\begin{cases}
\nabla_{\Sigma_1^{-1}}(l) = \frac{1}{2}\sum_{n=1}^{N} y_n\Sigma_1 - \frac{N}{2}\tilde{\Sigma_1} \\
\\
\nabla_{\Sigma_0^{-1}}(l) = \frac{1}{2}\sum_{n=1}^{N}(1-y_n)\Sigma_0 - \frac{N}{2}\tilde{\Sigma_0}
\end{cases}
$$

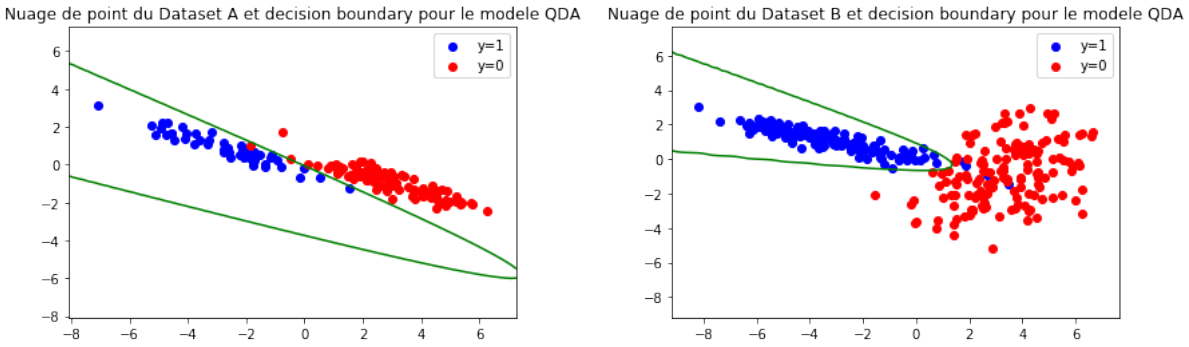
Ce qui donne:

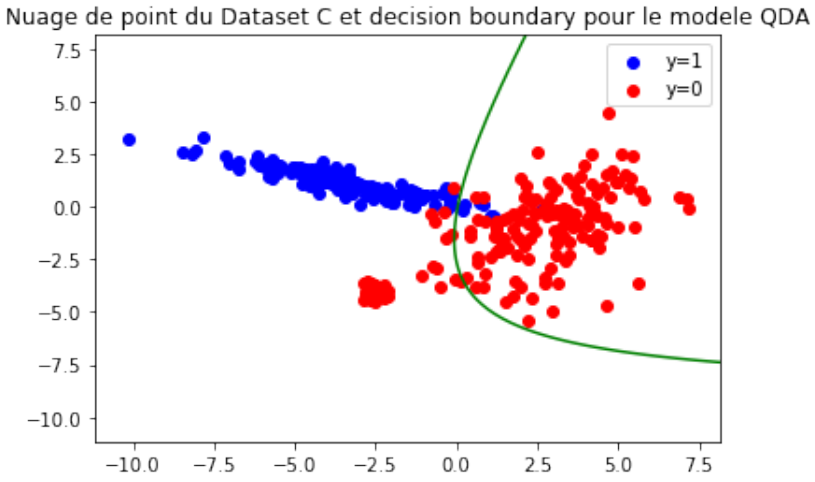
$$\hat{\Sigma_1} = \frac{\sum_{n=1}^{N} y_n (x_n - \hat{\mu_1})(x_n - \hat{\mu_1})^T}{\sum_{n=1}^{N} y_n} \quad ; \quad \hat{\Sigma_0} = \frac{\sum_{n=1}^{N} (1 - y_n)(x_n - \hat{\mu_0})(x_n - \hat{\mu_0})^T}{N - \sum_{n=1}^{N} y_n}$$

1. Après implémentation de ces estimateurs (voir *Notebook*), nous trouvons les valeurs suivantes pour les paramètres:

	Dataset A	**Dataset B**	**Dataset C**
π	0.333	0.5	0.625
μ_0	$\begin{pmatrix} 2.90 \\ -0.89 \end{pmatrix}$	$\begin{pmatrix} 3.34 \\ -0.84 \end{pmatrix}$	$\begin{pmatrix} 2.79 \\ -0.84 \end{pmatrix}$
μ_1	$\begin{pmatrix} -2.69 \\ 0.87 \end{pmatrix}$	$\begin{pmatrix} -3.22 \\ 1.08 \end{pmatrix}$	$\begin{pmatrix} -2.94 \\ -0.96 \end{pmatrix}$
Σ_0	$\begin{pmatrix} 2.31 & -1.05 \\ -1.05 & 0.58 \end{pmatrix}$	$\begin{pmatrix} 2.54 & 1.06 \\ 1.06 & 2.96 \end{pmatrix}$	$\begin{pmatrix} 2.90 & 1.25 \\ 1.25 & 2.92 \end{pmatrix}$
Σ_1	$\begin{pmatrix} 2.70 & -1.30 \\ -1.30 & 0.69 \end{pmatrix}$	$\begin{pmatrix} 4.15 & -1.33 \\ -1.33 & 0.52 \end{pmatrix}$	$\begin{pmatrix} 2.87 & -1.76 \\ -1.76 & 6.56 \end{pmatrix}$

2. Cela donne les graphiques suivants:





3. Nous avons les performances suivantes avec le modèle:

	Dataset A		**Dataset B**		**Dataset C**	
	Train	Test	Train	Test	Train	Test
QDA	$\frac{1}{150} \simeq 0.007$	$\frac{30}{1500} = 0.02$	$\frac{4}{300} \simeq 0.013$	$\frac{40}{2000} = 0.02$	$\frac{21}{400} = 0.0525$	$\frac{115}{3000} \simeq 0.038$

Tablaux regroupant les taux d'erreurs de classifications pour le modèle QDA

4. On observe donc des résultats similaires à ceux du modèle *Fisher LDA* sur les **Dataset A & C**. Sur le **Dataset A**, cela se comprend car le modèle *Fisher LDA* n'est jamais qu'un cas particulier du modèle *QDA*.

   Pour le **Dataset C**, nous pouvons expliquer de la même manière que précédemment la plus grande valeur du taux d'erreur sur le *train set* que sur le *test set*.

   Comme on pouvait s'y attendre, le modèle *QDA* performe bien mieux (2× mieux) que les 3 modèles précédents sur le **Dataset B**: la loi génératrice de ces données étant de la même famille que celle de notre modèle, on peut s'attendre à avoir réussi à trouver par entraînement des paramètres estimés proches des paramètres réels.