TP Supervised Learning Compte-rendu

Antoine Moulin

1 Emotion Recognition based on facial landmarks

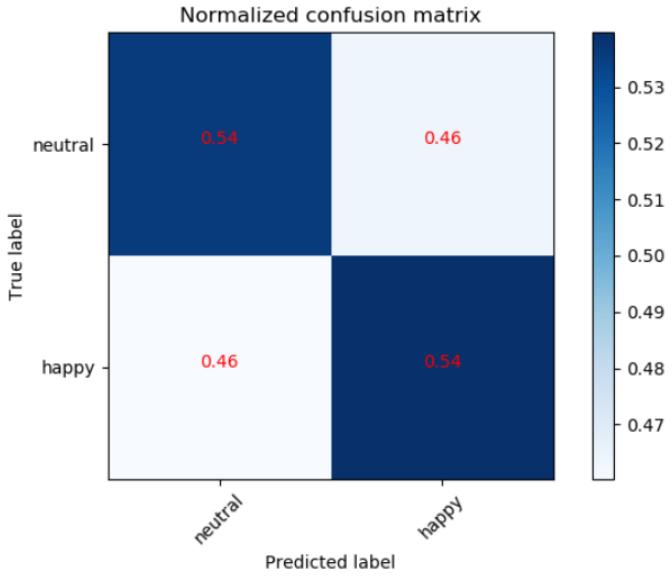
1. On importe les modules et fonctions nécessaires.

2. Les données contiennent des visages ainsi que des points de repère associés à ces derniers. Pour chacun de ces exemples, on a la classe associée ("neutre" ou "joyeux").

3. On mélange les données afin que l'ensemble d'entraînement soit le plus représentatif possible du cas général. Par exemple, si notre ensemble d'entraînement est rangé par classe, il est nécessaire de mélanger celui-ci.

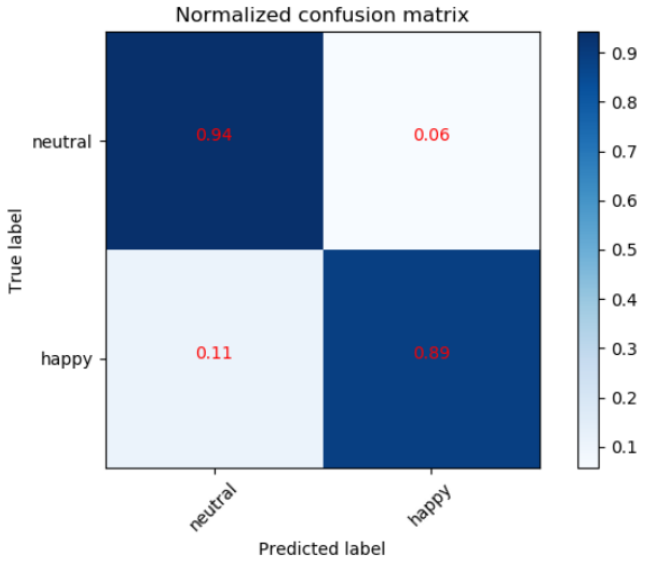
4. On exécute la cellule permettant d'afficher les points alignés avec une GPA.

5. On normalise les données afin que toutes les features aient le même ordre de grandeur. Cela permet notamment de donner la même importance à chaque feature. Après avoir utilisé une LDA, on obtient la matrice de confusion suivante :



Les résultats ne sont pas très bons. En effet, en lisant la première ligne, on constate que parmi tous les visages dont l'expression est "neutre", seulement 54% ont été classés comme "neutre". En lisant la première colonne, on remarque que sur tous les visages classés comme "neutre", seulement 54% sont effectivement dans la classe "neutre". On pourrait faire de même avec la classe "joyeux". Ainsi, on peut dire que le classifieur se trompe "plus ou moins une fois sur deux". Cela vient sans doute du fait qu'en utilisant les distances au lieu des positions, on perd l'information spatiale (une distance par rapport à la moyenne peut correspondre à n'importe quel point situé sur le cercle ayant pour centre la moyenne et pour rayon cette même distance).

6. On utilise directement les positions des points de repère comme features. Les résultats illustrés par la matrice de confusion semblent bien meilleurs :



7. La fonction `cross_val_score` appelle la fonction `score` du classifieur utilisé. Ainsi, dans chacune des méthodes (LDA, QDA, régression logistique ainsi que KNN), c'est la précision moyenne qui est renvoyée. Les résultats obtenus pour chaque méthode sont légèrement meilleurs que ceux obtenus à la question 5. Néanmoins, ils restent peu satisfaisants (de l'ordre de 0.6) :

```
Fitting LDA
done in 0.090s
 Average and std CV score : 0.56 +- 0.06585969936159747
Fitting QDA
done in 0.110s
 Average and std CV score : 0.5875 +- 0.032596012026013234
Fitting Logistic Regression
done in 0.198s
 Average and std CV score : 0.59 +- 0.0532681893816563
Fitting K-nearest neighbour
done in 0.081s
 Average and std CV score : 0.575 +- 0.03952847075210475
```

8. Cette fois, les résultats sont bien meilleurs. Excepté pour la QDA qui obtient un score de 0.68, les autres méthodes ont un score supérieur à 0.93, soit un résultat similaire à celui obtenu à la question 6. Nous obtenons effectivement des avertissements lors de l'utilisation d'une LDA ou d'une QDA :

```
C:\Users\User\PycharmProjects\TP2-IMA205\venv\lib\site-packages\sklearn\discriminant_analysis.py:686: UserWarning: Variables are collinear
  warnings.warn("Variables are collinear")
```

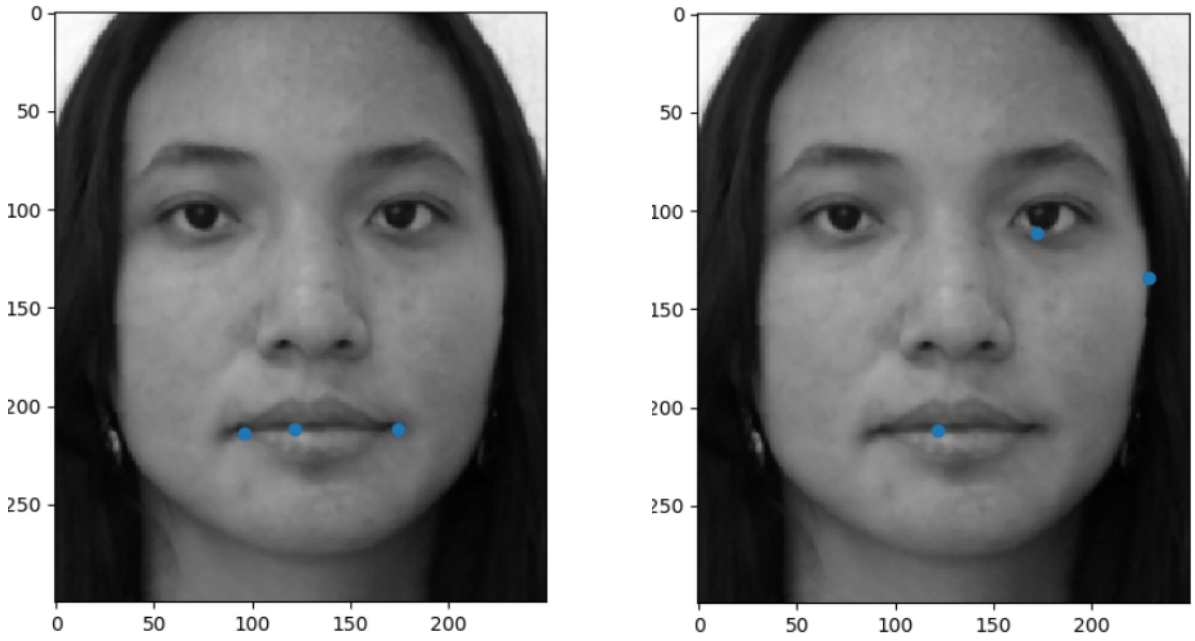
   Le fait qu'on obtienne des avertissements vient du fait que certaines variables sont colinéaires. C'est problématique car si des variables sont (presque) colinéaires, alors le déterminant de la matrice de covariance est proche de 0 et l'inversion de la matrice de covariance est peu précis. Le meilleur classifieur étant donné, dans le modèle LDA, par (cf cours) :

$$f^*(x_j) = arg\ \min_{C_k} -2x_j^T \left(\Sigma^*\right)^{-1} \mu_k^* + \left(\mu_k^*\right)^T \left(\Sigma^*\right)^{-1} \mu_k^* - 2log\left(\pi_{C_k}^*\right)$$

   on remarque que le problème d'inversion de la matrice de covariance a un impact non négligeable sur notre classifieur.

9. Dans le cadre d'une LDA, on suppose que les données suivent une distribution gaussienne. Il est donc possible d'utiliser une PCA. Dans notre cas, cela serait pertinent car la PCA nous fournit une base orthogonale, ce qui résout le problème de colinéarité. Les résultats obtenus sont globalement meilleurs. Si l'amélioration est faible pour ceux qui étaient déjà à 0.95, la QDA fournit maintenant de bien meilleurs résultats, passant de 0.68 à 0.95.

10. Nous pourrions choisir comme points de repère deux points situés aux extrémités gauche et droite de la bouche, ainsi qu'un troisième point situé au milieu de la lèvre supérieure car, a priori, une personne "joyeuse" sera en train de sourire tandis qu'une personne "neutre" ne sourira pas. Cela correspond par exemple aux points $0, 55, 61$. Nous pourrions également tenter de prendre les points qui maximisent la somme des variances selon l'axe des abscisses et l'axe des ordonnées. Dans les deux cas, voici ce que nous obtenons :

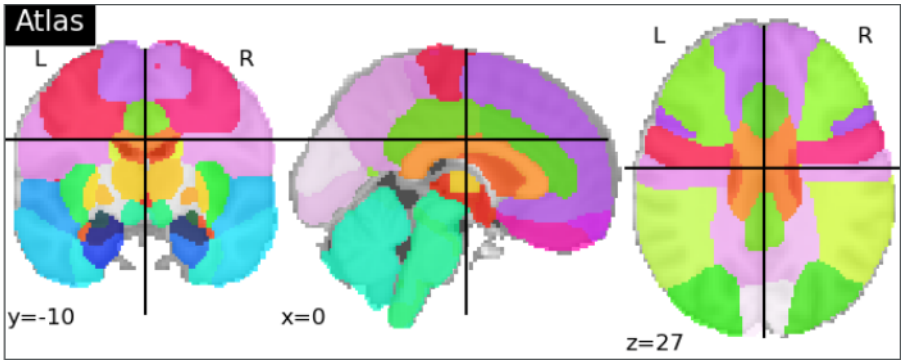


Dans le second cas, nous obtenons un point situé sur la bouche, un en dessous d'un oeil et un autre à l'extrêmité d'une joue, ce qui est cohérent puisque ces parties du visage bougent de façon importante lorsqu'une personne sourit.

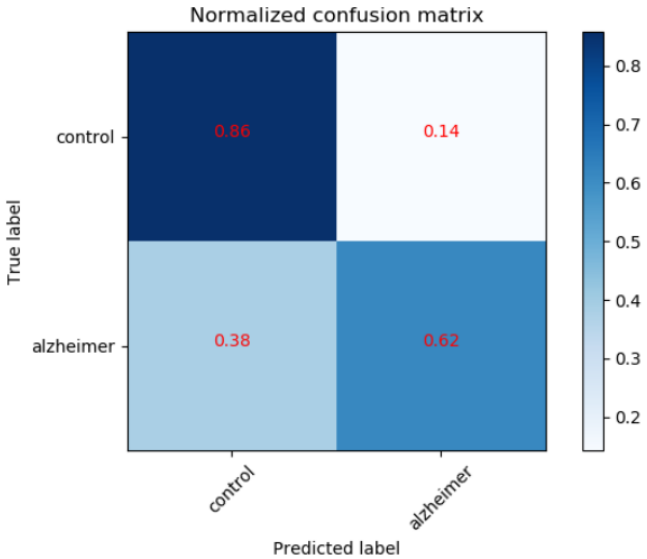
11. Il est possible que les images n'aient pas été correctement annotées et donc qu'il y ait quelques erreurs dans la vérité terrain (même si elles doivent être peu nombreuses). Après tout, l'erreur est humaine ...

2 Alzheimer prediction using PET signals

On charge les données avec le code donné dans l'énoncé et on l'affiche :



On commence par implémenter le script `main_ADNI.py` qui prédit si un patient est atteint de la maladie d'Alzheimer ou non. En utilisant une LDA comme dans la première partie, on obtient la matrice de confusion suivante :



- Si on utilise moins de sujets pour l'entraînement, on remarque que l'erreur d'entraînement a tendance à augmenter, tandis que l'erreur de test reste relativement élevée.
- Il n'est pas nécessaire d'utiliser toutes les régions d'intérêts. On pourrait utiliser les régions mentionnées, à savoir l'hippocampe gauche, l'hippocampe droit, lobe temporal médian gauche et lobe temporal médian droit. On réalise ensuite une PCA sur ces quatre régions en gardant quatre composantes. Cela permet d'avoir une base orthonormée et donc de régler les problèmes de colinéarité. Pour sélectionner les régions, on utilise la variable `labels` qui est à notre disposition.
- En essayant les différentes méthodes, on remarque que c'est la LDA qui donne de meilleurs résultats, avec un score de 0.87.

3 Theoretical questions

3.1 OLS

Avant de commencer les calculs, rappelons le contexte dans lequel nous nous plaçons dans cette partie. $\mathbf{x}$ est une matrice réelle de taille $n \times d$ et $\mathbf{y} \in \mathbb{R}^n$ est défini par :

$$\mathbf{y} = \mathbf{x}\beta + \epsilon \ t.q. \ \begin{cases} \mathbb{E}(\epsilon) = 0 \\ Var(\epsilon) = \sigma^2 I_n \end{cases}$$

L'inégalité portant sur des matrices symétriques (car $Var(\beta^*), Var\left(\tilde{\beta}\right) \in \mathbb{R}^{d\times d}$ sont symétriques), il faut également définir une relation d'ordre sur cet ensemble. Pour $V_1, V_2 \in \mathbb{R}^{d\times d}$, on dira que $V_1 \leq V_2$ si, pour tout $u \in \mathbb{R}^d$, $u^T V_1 u \leq u^T V_2 u$.

On définit maintenant deux estimateurs non biaisés de β : l'estimateur des moindres carrés $\beta^* = \left(\mathbf{x}^T\mathbf{x}\right)^{-1}\mathbf{x}^T\mathbf{y} = H\mathbf{y}$ et un estimateur linéaire $\tilde{\beta} = C\mathbf{y} = (H+D)\mathbf{y}$ avec D une matrice non nulle.

- Calcul de $\mathbb{E}\left(\tilde{\beta}\right)$. Comme $\tilde{\beta}$ est un estimateur non biaisé de β, on a $\mathbb{E}\left(\tilde{\beta}\right) = \beta$. Si on détaille le calcul, on a :

$$\begin{aligned} \mathbb{E}\left(\tilde{\beta}\right) &= (H+D)\,\mathbb{E}(\mathbf{y}) \\ &= (H+D)\,\mathbf{x}\beta \\ &= H\mathbf{x}\beta + D\mathbf{x}\beta \end{aligned}$$

  Or $H = \left(\mathbf{x}^T\mathbf{x}\right)^{-1}\mathbf{x}^T$ donc $H\mathbf{x} = I_d$. De plus, comme $\tilde{\beta}$ est non biaisé, alors $\mathbb{E}\left(\tilde{\beta}\right) = \beta$ d'où

$$\beta + D\mathbf{x}\beta = \beta \ i.e. \ D\mathbf{x}\beta = 0$$

  Ceci étant valable pour tout β, on a $D\mathbf{x} = 0$. Ainsi,

$$\boxed{\mathbb{E}\left(\tilde{\beta}\right) = \beta \ et \ D\mathbf{x} = 0}$$

- Calcul de $Var\left(\tilde{\beta}\right)$. On a :

$$\begin{aligned} Var\left(\tilde{\beta}\right) &= Var\left[(H+D)\,\mathbf{y}\right] \\ &= Var(H\mathbf{y}) + Var(D\mathbf{y}) + 2Cov(H\mathbf{y}, D\mathbf{y}) \end{aligned}$$

  De plus, $Cov(H\mathbf{y}, D\mathbf{y}) = Cov(H\epsilon, D\epsilon) = \mathbb{E}\left(H\epsilon\epsilon^T D^T\right) = H\mathbb{E}\left(\epsilon\epsilon^T\right)D^T = \sigma^2 HD^T$. Or d'après ce qui précède, $D\mathbf{x} = 0$ donc $x^T D^T = 0$ puis $HD^T = 0$. Ainsi, $Cov(H\mathbf{y}, D\mathbf{y}) = 0$.

  Comme $Var(D\mathbf{y}) = DVar(\mathbf{y})\,D^T = DVar(\epsilon)\,D^T = \sigma^2 DD^T$, on a

$$Var\left(\tilde{\beta}\right) = Var(\beta^*) + \sigma^2 DD^T$$

  Or

$$Var(\beta^*) = Var\left(\left(\mathbf{x}^T\mathbf{x}\right)^{-1}\mathbf{x}^T y\right) = \mathbb{E}\left(\left(\mathbf{x}^T\mathbf{x}\right)^{-1}\mathbf{x}^T\epsilon\epsilon^T\mathbf{x}\left(\mathbf{x}^T\mathbf{x}\right)^{-1}\right) = \left(\mathbf{x}^T\mathbf{x}\right)^{-1}\mathbf{x}^T\mathbb{E}\left(\epsilon\epsilon^T\right)\mathbf{x}\left(\mathbf{x}^T\mathbf{x}\right)^{-1} = \sigma^2\left(\mathbf{x}^T\mathbf{x}\right)^{-1}$$

  Donc

$$\boxed{Var\left(\tilde{\beta}\right) = \sigma^2\left(\mathbf{x}^T\mathbf{x}\right)^{-1} + \sigma^2 DD^T}$$

- Inégalité. Soit $u \in \mathbb{R}^d$. Alors :

$$u^T Var\left(\tilde{\beta}\right)u = u^T Var(\beta^*)\,u + \sigma^2 u^T DD^T u = u^T Var(\beta^*)\,u + \sigma^2\left|\left|D^T u\right|\right|^2 \geq u^T Var(\beta^*)\,u$$

  Il y a égalité si, et seulement si, D est nulle. Or D est par hypothèse non-nulle. On peut donc conclure :

$$\boxed{Var\left(\tilde{\beta}\right) > Var(\beta^*)}$$

3.2 Ridge regression

- On rappelle (cf cours) que l'estimateur Ridge est donné par

$$\beta^*_{ridge} = \left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}\mathbf{x}_c^T\mathbf{y}_c$$

Calculons son espérance.

$$\mathbb{E}\left(\beta^*_{ridge}\right) = \left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}\mathbf{x}_c^T\mathbb{E}\left(\mathbf{y}_c\right) = \left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}\mathbf{x}_c^T\mathbf{x}_c\beta = \left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}\left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d - \lambda I_d\right)\beta$$

Ainsi,

$$\boxed{\mathbb{E}\left(\beta^*_{ridge}\right) - \beta = -\lambda\left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}\beta \neq 0}$$

Donc β^*_{ridge} est un estimateur biaisé de β.

- On reprend l'expression de l'estimateur énoncée à la question précédente :

$$\beta^*_{ridge} = \left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}\mathbf{x}_c^T\mathbf{y}_c$$

On remplace $\mathbf{x}_c$ par UDV^T :

$$\beta^*_{ridge} = \left(\left(UDV^T\right)^T UDV^T + \lambda I_d\right)^{-1}\left(UDV^T\right)^T\mathbf{y}_c$$

C'est-à-dire

$$\beta^*_{ridge} = \left(VD^TU^TUDV^T + \lambda I_d\right)^{-1}VD^TU^T\mathbf{y}_c$$

On utilise le fait que U est une matrice orthogonale et que D est une matrice diagonale :

$$\beta^*_{ridge} = \left(VD^2V^T + \lambda I_d\right)^{-1}VDU^T\mathbf{y}_c$$

Comme $\lambda I_d = \lambda VV^T$, on a :

$$\beta^*_{ridge} = \left(V\left(D^2 + \lambda I_d\right)V^T\right)^{-1}VDU^T\mathbf{y}_c$$

Puis

$$\boxed{\beta^*_{ridge} = V\left(D^2 + \lambda I_d\right)^{-1}DU^T\mathbf{y}_c}$$

Comme $D^2 + \lambda I_d$ est une matrice diagonale, cette expression est utile car elle ne nécessite pas de calculer un inverse, étant donné que l'inverse du matrice diagonale est la matrice diagonale formée des inverses des coefficients.

- Calculons $Var\left(\beta^*_{ridge}\right)$ à l'aide de la décomposition en valeurs singulières. On utilise la formule de l'estimateur :

$$Var\left(\beta^*_{ridge}\right) = \sigma^2\left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}\mathbf{x}_c^T\mathbf{x}_c\left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}$$

$\mathbf{x}_c^T\mathbf{x}_c$ est une matrice symétrique donc d'après le théorème spectral, on peut écrire $\mathbf{x}_c^T\mathbf{x}_c = Q\Delta Q^T$ avec Δ une matrice diagonale et Q une matrice orthogonale. Ainsi, on peut écrire :

$$\mathbf{x}_c^T\mathbf{x}_c\left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1} = Q\Delta Q^T\left(Q\left(\Delta + \lambda I_d\right)Q^T\right)^{-1} = Q\Delta\left(\Delta + \lambda I_d\right)^{-1}Q^T$$

Or les matrices Δ et $\left(\Delta + \lambda I_d\right)^{-1}$ étant diagonales, elles commutent. Ainsi,

$$\mathbf{x}_c^T\mathbf{x}_c\left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1} = Q\left(\Delta + \lambda I_d\right)^{-1}\Delta Q^T = \left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-1}\mathbf{x}_c^T\mathbf{x}_c$$

On a donc, pour la variance,

$$Var\left(\beta^*_{ridge}\right) = \sigma^2\left(\mathbf{x}_c^T\mathbf{x}_c + \lambda I_d\right)^{-2}\mathbf{x}_c^T\mathbf{x}_c$$

On utilise la décomposition en valeurs singulières de $\mathbf{x}_c$:

$$Var\left(\beta^*_{ridge}\right) = \sigma^2V\left(D^2 + \lambda I_d\right)^{-2}D^2V^T$$

On a donc :

$$Var\left(\beta^*_{ridge}\right) = \sigma^2\sum_{k=1}^{d}\frac{s_k^2}{\left(s_k^2 + \lambda\right)^2}v_iv_i^T$$

où $(s_k)_{k\in[\![1,d]\!]}$ sont les valeurs singulières de $\mathbf{x}_c$. D'où

$$Var\left(\beta^*_{ridge}\right) = \sigma^2\sum_{k=1}^{d}\frac{\lambda_k}{\left(\lambda_k + \lambda\right)^2}v_iv_i^T$$

où les $(\lambda_k)_{k\in[\![1,d]\!]}$ sont les valeurs propres de $\mathbf{x}_c$. Or, de la même façon, on trouve :

$$Var\left(\beta^*_{OLS}\right) = \sigma^2 \sum_{k=1}^{d} \frac{1}{\lambda_k} v_i v_i^T$$

Comme, pour tout k, on a $\frac{\lambda_k}{(\lambda_k+\lambda)^2} < \frac{1}{\lambda_k}$, on peut conclure :

$$\boxed{Var\left(\beta^*_{OLS}\right) \geq Var\left(\beta^*_{ridge}\right)}$$

- D'après la formule précédente de la variance, on a clairement

$$\boxed{Var\left(\beta^*_{ridge}\right) \underset{\lambda \longrightarrow +\infty}{\longrightarrow} 0}$$

De plus, l'espérance tend vers 0 lorsque λ tend vers $+\infty$, donc

$$\boxed{\mathbb{E}\left(\beta^*_{ridge}\right) - \beta \underset{\lambda \longrightarrow +\infty}{\longrightarrow} -\beta}$$

- On reprend l'expression donné plus haut. On remplace $\mathbf{x}_c^T \mathbf{x}_c$ par I_d, on trouve $\beta^*_{ridge} = \frac{1}{1+\lambda} \mathbf{x}_c^T \mathbf{y}_c$. Par ailleurs, on a également $\beta^*_{OLS} = \mathbf{x}_c^T \mathbf{y}_c$. D'où

$$\boxed{\beta^*_{ridge} = \frac{1}{1+\lambda} \beta^*_{OLS}}$$

- On utilise le résultat précédent :

$$\lambda \left(\frac{||\beta^*_{OLS}||_2^2}{(1+\lambda)^2} - t \right) = 0$$

Lorsque $\lambda = 0$, on a

$$\boxed{\beta^*_{ridge} = \beta^*_{OLS}}$$

et il n'y a pas de relation entre λ et t puisque dans ce cas, il n'y a pas de régularisation.

Supposons $\lambda \neq 0$. L'équation précédente donne

$$||\beta^*_{ridge}||^2 = \frac{\|\beta^*_{OLS}\|^2}{(1+\lambda)^2} = t$$

Lorsque $\lambda \longrightarrow +\infty$, on a donc

$$\boxed{\beta^*_{ridge} = 0}$$

- La régularisation Ridge a pour effet de forcer les coefficients à être dans une boule de rayon $\sqrt{t}$ mais sans forcément en mettre certains à 0 tandis que la régularisation a pour propriété de mettre des coefficients à 0 (ceux qui sont inférieurs en valeur absolue à $\frac{\lambda}{2}$).

3.3 Elastic Net

On peut se référer au cours de SD204 pour cette partie.

3.4 LDA

- D'après le cours, nous avons

$$f^*(x_j) = \underset{C_k}{\operatorname{argmax}} P_k(X) \pi_{C_k}$$

$$f^*(x_j) = \underset{C_k}{\operatorname{argmin}} \; -2 \log\left[P_k(X) \pi_{C_k}\right]$$

De plus, à une constante près, nous avons

$$P_k(X) = |\Sigma_k|^{-\frac{1}{2}} \exp\left(-\frac{1}{2} \left(x - \mu_k\right)^T \Sigma_k^{-1} \left(x - \mu_k\right) \right)$$

D'où

$$f^*(x_j) = \underset{C_k}{\operatorname{argmin}} \; \left(x_j - \mu_k\right)^T \Sigma_k^{-1} \left(x_j - \mu_k\right) + \log|\Sigma_k| - 2\log\left(\pi_{C_k}\right)$$

Puis

$$f^*(x_j) = \underset{C_k}{\operatorname{argmin}} \; x_j^T \Sigma_k^{-1} x_j - x_j^T \Sigma_k^{-1} \mu_k - \mu_k^T \Sigma_k^{-1} x_j + \mu_k^T \Sigma_k^{-1} \mu_k - 2\log\left(\pi_{C_k}\right) + \log|\Sigma_k|$$

Finalement,

$$\boxed{f^*(x_j) = \underset{C_k}{\operatorname{argmin}}\, x_j^T \Sigma_k^{-1} x_j - 2x_j^T \Sigma_k^{-1} \mu_k - 2\log(\pi_{C_k}) + \log|\Sigma_k| + \mu_k^T \Sigma_k^{-1} \mu_k}$$

- La solution est donc bien une fonction quadratique, à cause du terme $x_j^T \Sigma_k^{-1} x_j$.