Projet analyse de données

SCHLÖGEL Benjamin BOSWELL Ethan

Mai 2023

1 Extension 1 et 2

1.1 Extension 1 : SVM à noyau gaussien

Pour répondre au problème de séparation des données qui ne peut pas se faire selon un modèle linéaire, nous appliquons au point x_i une transformation non-linéaire notés Φ. La fonction duale du Lagrangien s'écrit :

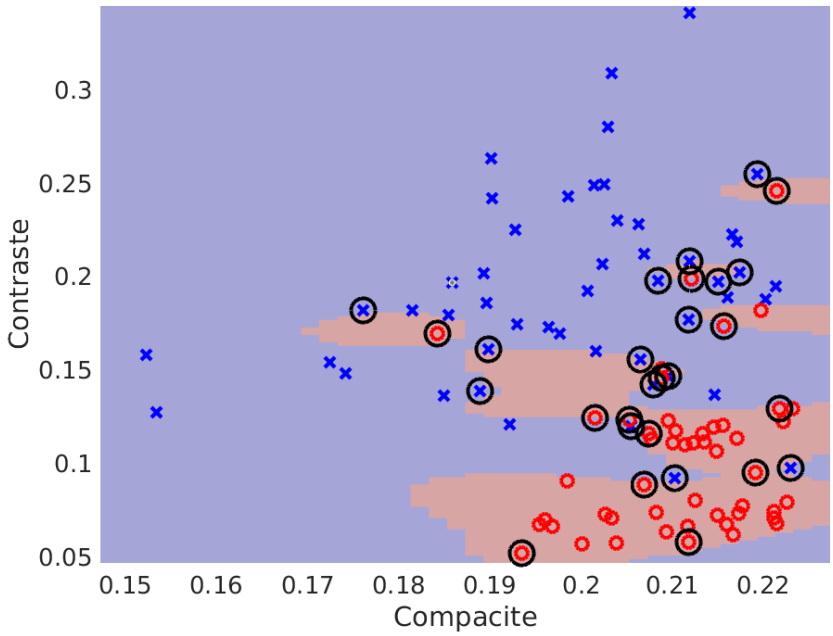
$$\overline{\mathcal{L}}(\alpha_1, ..., \alpha_n) = \frac{1}{2}\Sigma_{i=1}^{n}\Sigma_{j=1}^{n}\alpha_i y_i \Phi(x_i)^T \Phi(x_j) y_j \alpha_j + \Sigma_{i=1}^{n}\alpha_i$$

Le "coup de noyau" consiste à remplacer le produit scalaire $\Phi(x_i)^T\Phi(x_j)$ par :

$$K(x_i, x_j) = e^{-\frac{||x_i - x_j||^2}{2\sigma^2}}$$

Cela nous réduit le cout de calcul : nous n'avons pas à calculer les coordonnées de chaque point dans un espace d'une grande dimension. Avec le noyau Gaussien, nous travaillons sur les coordonnées transformées.

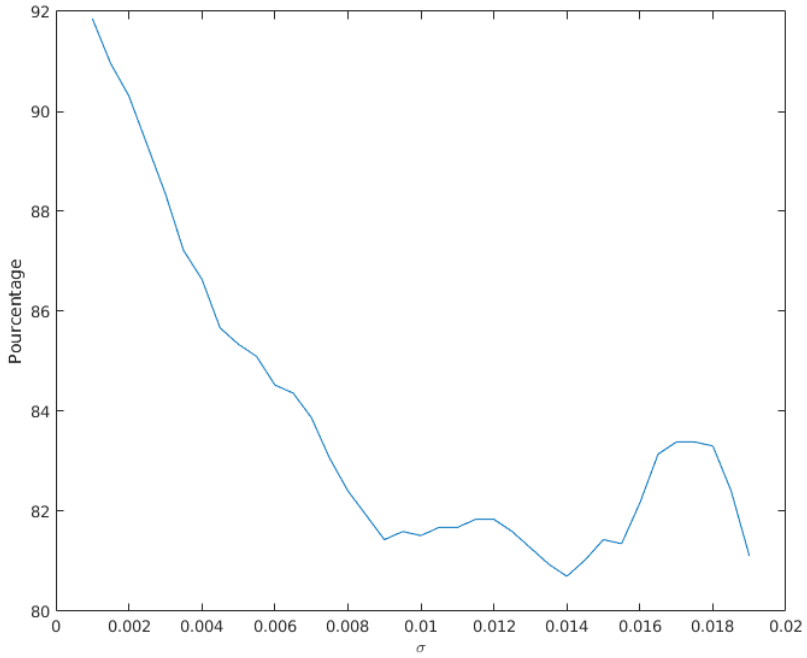
Figure 1: Résultat de la classification de l'extension 1 ($\sigma = 0.012$)



Avec cette valeur de σ, on constate que nous avons un apprentissage correct. En effet, en ajoutant un autre jeu de donnée, il est probable que ce jeu soit bien classé. On obtient un pourcentage de bonnes classifications de 81,2% pour la figure 1, ce qui est un résultat satisfaisant. Nous avons des zones pour les classes réalistes.

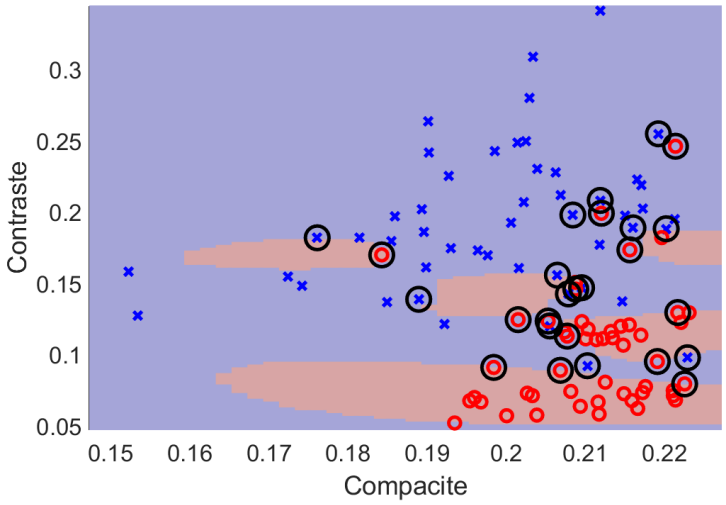
1.2 Extension 2 : optimisation du SVM à noyau gaussien

Figure 2: Pourcentage de bonnes classifications en fonction de σ



On remarque que le pourcentage augmente lorsque σ diminue : cependant, nous n'avons pas choisi σ trop petit, car nous risquons le sur-apprentissage. On observe également un maximum local en 0.017, mais on observe des zones qui semblent mal adaptées aux données d'entraînement, par exemple la zone rouge du bas qui est vide à gauche (voir Figure 3). C'est pourquoi nous avons choisi $\sigma = 0.012$.

Figure 3: Résultat de la classification avec $\sigma = 0.017$



2 Extension 3 : SVM linéaire à marge souple

Ici, nous reprenons l'extension 1 en implémentant une marge souple à notre SVM. Nous utilisons les variables ressorts ξ_i qui permettent de rendre plus souple les contraintes du classifieur. Ainsi la contrainte de constructions du séparateur linéaire s'écrit :

$$y_i(w^T x_i - c) - 1 \geq \xi_i$$

Mais pour éviter de trop assouplir cette contrainte, nous utilisons un terme de régularisation λ qui permet de contrôler le poids de $\Sigma_{i=1}^{n}\xi_i$. Cette nouvelle configuration permet d'élargir la zone d'apprentissage, et donc de mieux s'adapter lors de l'apparition de nouveaux jeux de données.

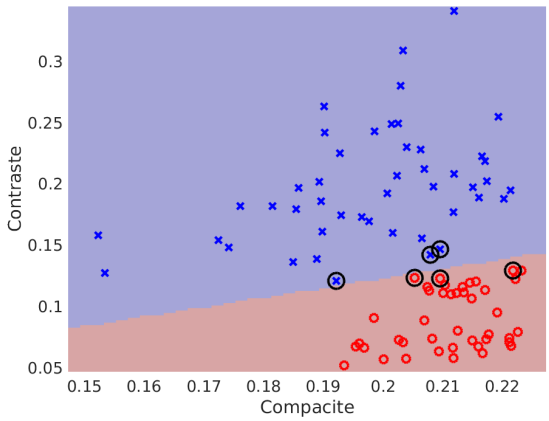


Figure 4: Résultat de la classification de l'extension 3 ($\lambda = 1750$)

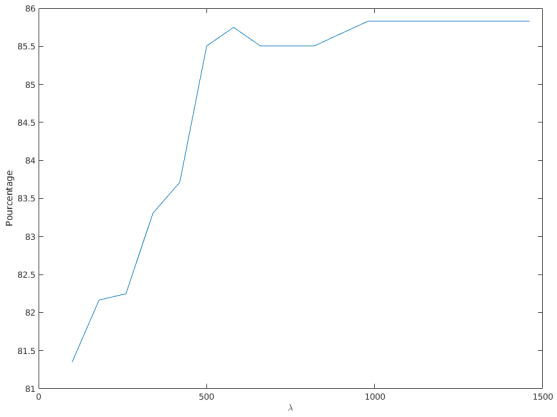


Figure 5: Pourcentage de bonnes classifications en fonction de λ

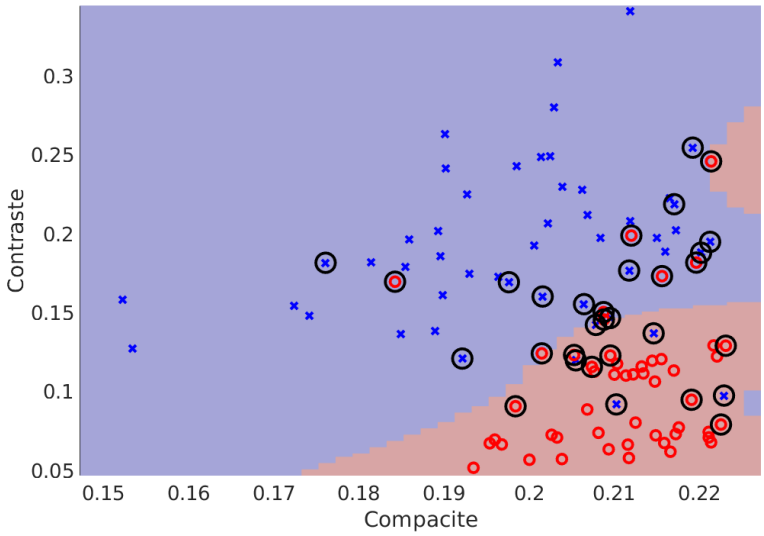
La manipulation fondamentale qui change par rapport à l'extension 1 est le calcul de c. Au lieu de prendre un vecteur support quelconque, nous choisissons un qui vérifie la condition $\alpha_i < \lambda$, qui est le résultat de la combinaison des contraintes précédentes. On peut chercher à trouver une valeur optimale de λ en calculant le pourcentage de bonnes classifications en fonction de λ. Nous obtenons ainsi la figure 4.

On constate que le pourcentage stagne à partir $\lambda = 500$. On peut donc choisir n'importe quelle valeur à partir de 500, nous avons donc choisi $\lambda = 1750$ pour la figure 3. Nous le voyons sur la figure 3, les deux jeux de données sont bien séparés et la reconnaissance des données proches de la frontière sont exactes.

3 Extension 4 : SVM à noyau gaussien et marge souple

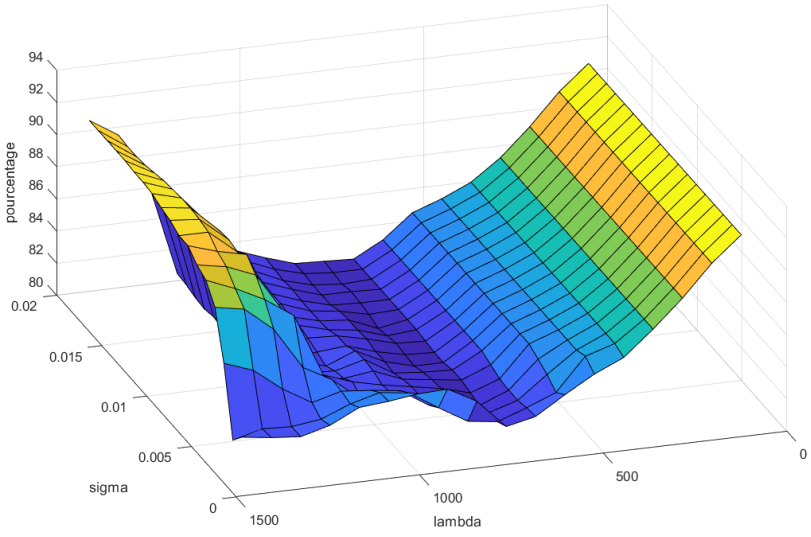
Comme précédemment, nous implémentons une SVM à noyau Gaussien de la même manière, mais le calcul de c se fait en récupérant le vecteur support qui nous intéresse. Comme pour la SVM linéaire à marge souple, nous utilisons la condition $\alpha_i < \lambda$. Pour les valeurs $\lambda = 10000$ et $\sigma = 0.075$ nous obtenons la figure suivante :

Figure 6: Résultat de la classification de l'extension 4



On peut noter que malheureusement le calcul des vecteurs de support ne nous donne pas un résultat satisfaisant. De plus, nous obtenons un résultat plus intéressant qu'à l'extension 1 : la zone d'apprentissage est plus élargie, même si certaines données sont mal classées. Mais cette configuration est plus intéressante que l'extension 1, car l'ajout de nouveaux jeux de données sera très probablement mieux classé que si on le faisait sur l'extension 1. De plus, sur la figure 7, nous pouvons voir l'influence de σ et de λ en même temps.

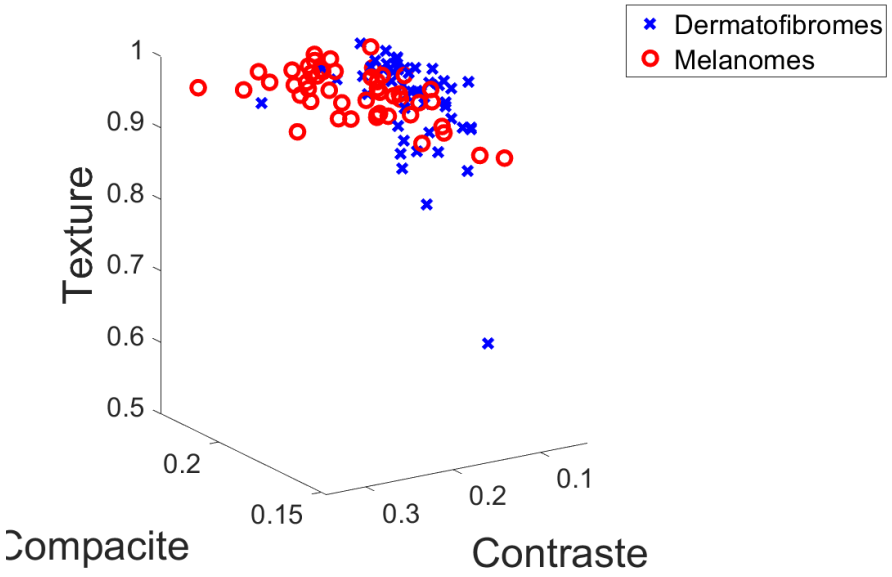
Figure 7: Pourcentage de bonnes classifications en fonction de λ et σ



4 Extension 5 : classification de données en dimension 3

Dans cette partie, nous ajoutons une troisième caractéristique, la texture, si bien que les données sont désormais des points en dimension 3.

Figure 8: Répartition des données en dimension 3



La fonction quadprog ne converge plus si on remplace directement les matrices en entrée par des matrices contenant trois colonnes, il faut donc adapter l'approche pour obtenir une classification des points.