Machine à vecteurs de support dans le cadre de données fonctionnelles - classification

March 22, 2020

François LE GAC

Shon AMSALHEM

M2 ISIFAR

1 Introduction

Ce travail s'appuie sur un papier de recherche intitulé : *Support vector machine for functional data classification*[1]. L'objectif de cette étude est de définir le cadre d'utilisation du modèle de machine à vecteurs de support (SVM) dans le cas où les données sont fonctionnelles. Tout au long de cette étude, nous illustrerons nos propos à l'aide d'une application, la classification d'échantillons d'essence.

Commençons par rappeler ce que sont les données fonctionnelles. Une variable aléatoire est dite fonctionnelle si ses valeurs sont dans un espace de dimension infini. Une observation d'une variable fonctionnelle est appelée donnée fonctionnelle. Ce qui veut dire que chaque individu correspond non plus à un vecteur mais à une courbe, à un continuum. Il existe de nombreuses applications pratiques où les données fonctionnelles interviennent comme par exemple les données spectrométriques, la reconnaissance vocale ou encore les séries temporelles (chaque observation est une série temporelle).

2 Application

Pour illustrer la partie théorique de cette étude, nous allons tenter de résoudre un problème de classification binaire avec des données fonctionnelles. Nous avons sélectionné le *jeu de données Octane*[2] dont on donne un aperçu ci-dessous :

	Col 1	Col 2		Col 400	Var. cible
Ech-1	$X_1(\lambda_{900})$	$X_1(\lambda_{902})$	...	$X_1(\lambda_{1700})$	0
...	...	...	...	...	...
Ech-60	$X_{60}(\lambda_{900})$	$X_{60}(\lambda_{902})$	...	$X_{60}(\lambda_{1700})$	1

Fig 1. Jeu de données Octane

Ce jeu de donnée comprend les spectres de 60 échantillons d'essence, mesurés pour une longueur d'onde de 900 à 1700 nanomètres. L'absorbance pour différentes longueurs d'ondes permet de prédire l'indice d'octane dans le mélange. Cet indice va mettre en avant la résistance ou non de l'échantillon d'essence à l'auto allumage (inflammation spontanée et accidentelle du mélange carburé). Plus l'indice d'octane est élevé, plus l'essence est résistante à l'auto allumage.

Remarque :

Pour nous ramener à un problème de classification, nous choisissons arbitrairement un indice seuil : 87. Si l'indice d'un échantillon est supérieur à cette valeur seuil, il est classé comme "résistant à l'auto allumage" (1), sinon il est classé comme "non résistant à l'auto allumage" (0).

3 Problème posé par les données fonctionnelles

Les données fonctionnelles sont par définition dans un espace de dimension infini. Cela peut poser des problèmes d'un point de vue pratique et théorique. Pour illustrer ces difficultés, prenons un exemple simple : la régression linéaire. On rappelle que ce modèle d'apprentissage supervisé consiste à modéliser la variable cible Y de la façon suivante :

$$\mathbb{E}(Y|X) = X\beta$$

Si $X \in \mathbb{R}^d$, alors on peut aisément estimer le paramètre $\beta \in \mathbb{R}^d$ à l'aide de la méthode des moindres carrés. Cette dernière nécessite d'inverser la matrice de covariance de X.
Cependant, en pratique, si N (nb d'obs) $<< d$ comme c'est le cas pour les données fonctionnelles, la méthode des moindres carrés se heurte à une impasse. La matrice de covariance est non inversible.

Il existe deux solutions à ce problème : le filtrage (projection) et la pénalisation. La première consiste à réduire l'espace initial des données d'apprentissage en un espace de dimension finie. On utilise des projections dont on parlera davantage dans la suite. La seconde méthode "la pénalisation" dont on parlera peu dans cette étude, consiste à réduire la complexité du problème. Dans le cadre

de notre exemple précédent, la régression linéaire, on peut penser à la pénalisation Lasso (norme L1) ou Ridge (norme L2).

4 Principe des machines à vecteurs de support (SVM) :

4.1 Cas linéairement séparable

L'algorithme de machine à vecteurs de support (SVM) est une méthode d'apprentissage supervisée. Prenons le cadre de la classification binaire où $Y \in \{-1, 1\}, X \in \mathbb{R}^d$. Pour simplifier encore davantage le problème, on considère dans un premier temps des données linéairement séparables. Le but de la méthode SVM est de déterminer l'hyperplan séparateur qui maximise la marge. En effet, sur la figure ci-dessous, on voit qu'un grand nombre de droites permettent de séparer les données. Intuitivement, si l'on ajoute une donnée proche du groupe des +1, on aura tendance à vouloir le classer comme tel. On va donc choisir la droite qui se trouve le "plus au milieu possible", ce qui revient à maximiser la marge. La marge est la distance entre les vecteurs support, c'est à dire les points à la frontière de chaque groupe et l'hyperplan (en l'occurrence la droite).

La fonction $x \mapsto sign(< w, x > +b)$ est une règle de discrimination potentielle.

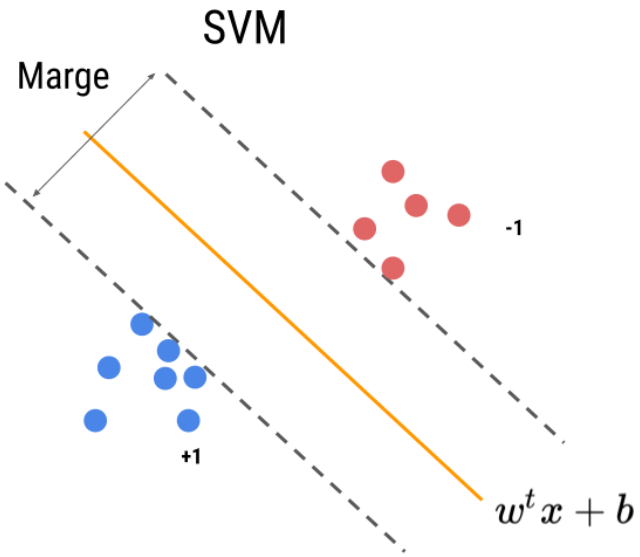

Fig 2. Support vector machine

3

Même si l'hyperplan optimal est unique, il existe plusieurs couples (w, b), qui décrivent ce même hyperplan. On décide donc de ne considérer que l'unique paramètrage (w, b) tel que :

$\forall i, y_i(w^t x_i + b) \geq 1$ et l'égalité est atteinte si x_i est un vecteur support.

Après avoir utilisé la distance d'un point à une droite on peut aisément montrer qu'on se retrouve avec le problème suivant :

$$\begin{cases} Maximiser \ \frac{1}{||w||} \\ tel \ que \ \forall i, y_i(w^t x_i + b) \geq 1 \end{cases} \tag{1}$$

Que l'on reformule de la manière suivante pour simplifier le problème d'optimisation :

Hard Margin (Primal)

$$\begin{cases} Minimiser\ \frac{1}{2}||w||^2 \\ tel\ que\ \forall i, y_i(w^t x_i + b) \geq 1 \end{cases} \tag{2}$$

C'est un problème d'optimisation convexe sous contrainte linéaire. Ce problème qu'on appelle problème primal est difficile à résoudre dans la pratique. En utilisant les coefficients multiplicateurs de Lagrange on se ramène au problème suivant, plus facile à résoudre :

Hard Margin (Dual)

$$\begin{cases} Minimiser\ L_d(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i=1}^n \sum_{k=1}^n \alpha_i \alpha_k y_i y_k x_i^t x_k \\ tel\ que\ \alpha_i \geq 0\ et\ \sum_{i=1}^n \alpha_i y_i = 0 \end{cases} \tag{3}$$

4.2 Cas non linéairement séparable

Dans le cas de données non linéairement séparables on reformule notre problème d'optimisation de la manière suivante :

Soft Margin (Primal)

$$\begin{cases} Minimiser\ \frac{1}{2}||w||^2 + C\sum_{i=1}^n \xi_i \\ tel\ que\ \forall i, y_i(w^t x_i + b) \geq 1 - \xi_i\ avec\ \xi_i \geq 0, i = 1, ..., n \end{cases} \tag{4}$$

On vient donc rajouter un terme dans chaque condition. On peut voir le paramètre C comme un compromis entre maximiser la marge et minimiser l'erreur de classification.

De la même manière on peut exprimer ce problème en problème dual.

On obtient w^* (le w optimal) :

$w^* = \sum_{i=1}^n \alpha_i^* x_i$

Si l'on remplace w^* dans notre règle de décision : $X \in \mathbb{R}^d, x \mapsto sign(w^t x + b)$
devient $x \mapsto sign(\sum_{i=1}^n \alpha_i^* x_i^t x_i + b^*)$

On s'aperçoit que la fonction de décision dépend des produits scalaire des x_i ! Cette remarque est très importante puisqu'elle va nous permettre d'utiliser les méthodes à noyau (Kernel).

4.3 Un point sur les méthodes à noyau (Kernel)

Dans le cas où l'on souhaite utiliser SVM comme un algorithme non linéaire, on va utiliser une fonction ϕ pour envoyer nos données dans un espace où elles seront séparables linéairement par un hyperplan.

$$\phi : E \to E'$$
$$X \mapsto \phi(X)$$

On appelle fonction noyau la fonction :

$$E \times E \to \mathbb{R}$$

$$(x_1, x_2) \mapsto K(x_1, x_2) = \phi(x_1)^t \phi(x_2)$$

(Remarque : le théorème de Mercer indique que K doit être continue, symétrique, semi-définie positive)

On rappel que le calcul de l'hyperplan séparateur ne nécessite ni la connaissance de E' ni de ϕ mais seulement de K ! Voici deux exemples de noyau dont on se servira par la suite :

Linéaire : $K(x_1, x_2) = \langle x_1, x_2 \rangle + c$ où c est une constante.

Gaussien : $K(x_1, x_2) = exp(-\gamma ||x_1 - x_2||^2)$, $\gamma \geq 0$

5 SVM et les données fonctionnelles

Définir SVM dans le cadre de données fonctionnelles est aussi facile que lorsque les données d'entrée sont dans $\mathbb{R}^d$, notamment parce que l'on fait l'hypothèse que les données d'entrée sont dans un espace de Hilbert. Toutefois, la nature fonctionnelle des données a un impact : l'algorithme présente de mauvaise performances.

6 Transformation fonctionnelles

Comme on peut le voir sur les figures 3 et 4 ci-dessous, il est difficile de distinguer les deux classes. On y arrivera peut être mieux en utilisant les dérivées secondes de chaque courbe.

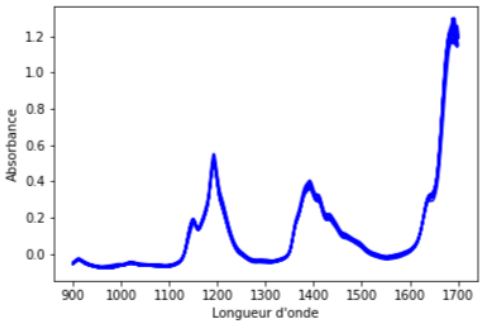

Fig 3. Courbe - non résistant à l'auto allumage (0)

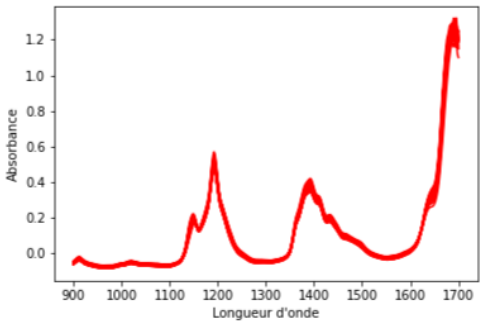

Fig 4. Courbe - résistant à l'auto allumage (1)

En effet, on peut par exemple s'attendre à ce qu'une classe présente un nombre supérieur de maximum locaux que l'autre classe. Comment obtenir les dérivées secondes de chaque courbe ? On commence par projeter chaque courbe sur un sous espace spline. On utilise une méthode d'interpolation avec 1000 points. On rappelle que l'interpolation est une opération qui permet d'obtenir une courbe généralement plus simple à partir d'un nombre infini de points. L'interpolation va nous permettre de faire des calculs de dérivées secondes. Voici ci-dessous un exemple pour un individu :

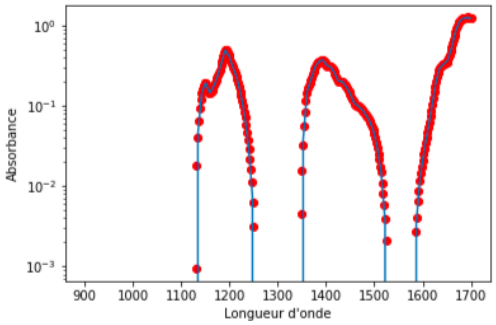

Fig 5. Interpolation

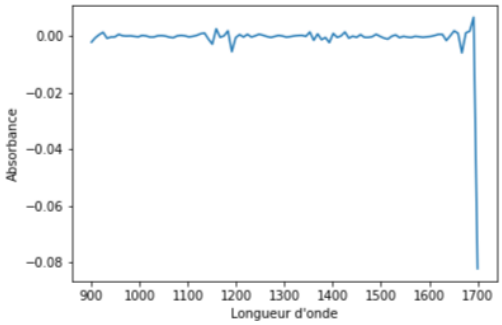

Fig 6. Dérivée seconde pour une obs.

Voici ci-dessous les résultats obtenus pour les deux classes :

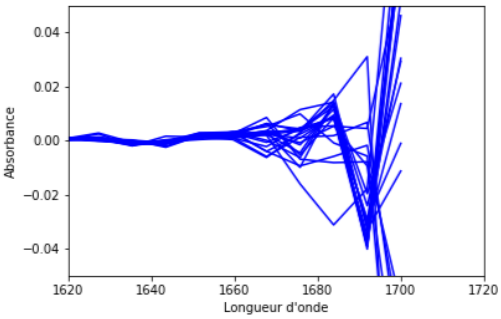

Fig 7. Dérivée seconde - non résistant à l'auto allumage (0)

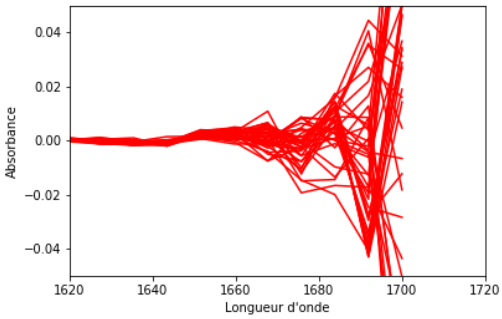

Fig 8. Dérivée seconde - résistant à l'auto allumage (1)

Malheureusement on n'observe pas non plus de différences significative à l'oeil nu. Essayons tout de même de tester SVM pour différents Kernels sur ces données.

7 Résultat

Revenons en à notre application : on souhaite classer 60 échantillons en deux catégories (résistant et non résistant à l'auto allumage). Pour évaluer la performance de nos différentes méthodes, on utilise la précision c'est à dire le pourcentage de données bien classées. En outre, nous utilisons la cross-validation : on échantillonne notre jeu de données initial en un jeu de données d'entraînement (70% des valeurs), et un jeu de données test (30% des valeurs). La performance de notre modèle dépend très largement de la séparation en deux de notre jeu de donnée initial. On va donc répéter la cross validation 500 fois pour chaque méthode et utiliser la précision moyenne.

	KNN	SVM			
	Méthode de base (données brut)	Linéaire - données brutes	Gaussien - données brutes	Linéaire - **dérivée seconde**	Gaussien - **dérivée seconde**
Accuracy	**0.81**	**0.63**	**0.63**	**0.65**	**0.60**

Fig 9. Performances des modèles

Comme nous l'avons évoqué précédemment, les performances de l'algorithme SVM sur les données brutes sont mauvaises : 0.63 en précision pour les Kernels linéaire et Gaussien. La méthode KNN que l'on utilise à titre de comparaison donne de bien meilleurs résultats : 0.81. Le SVM à Kernel Gaussien appliqué aux dérivées secondes présente le plus mauvais résultat : 0.59. La mauvaise performance du Kernel Gaussien peut s'expliquer par le fait que la comparaison directe de ces spectres basée sur la norme L2 est en général dominée par la valeur moyenne de ces spectres, qui n'est pas la bonne caractéristique pour distinguer les deux types de spectres. Finalement, le SVM à Kernel Linéaire appliqué sur les dérivées secondes présente le meilleur résultat parmi les différents SVM : 0.65. Il faut toutefois tempérer notre enthousiasme, cette précision est largement inférieure à celle de la méthode KNN.

On a montré que l'on peut utiliser SVM directement sur des données fonctionnelles mais qu'on obtient systématiquement de mauvais résultats. En revanche, le fait d'utiliser SVM sur les dérivées secondes permet de prendre en compte la structure spécifique des données fonctionnelles. Cependant, nous n'avons malheureusement pas réussi à obtenir des résultats satisfaisant en comparaison avec la méthode de base KNN.

Références :

- 1. Fabrice,Rossia. Nathalie, Villab. Support vector machine for functional data classification : Elsevier : 2006.
- 2. Information on Octane's data set