Université Sultan Moulay Slimane
ECOLE NATIONALE DES SCIENCES APPLIQUEES
Département de Mathématiques et Informatique

PROJET DU DATA MINING

CLASSIFICATION METHODES

Support Vector Machines

Réalisé par :

EL HAMID Youssef

DRIOUCH Latifa

DAKIR Ismahane

Sous la direction de : Pr. ORDOU Amal

Année universitaire : 2022 - 2023

TABLE DES MATIÈRES

INTRODUCTION

L'apprentissage automatique est un domaine de l'informatique qui a connu une croissance rapide ces dernières années, offrant des techniques permettant aux machines d'apprendre à partir de données et d'effectuer des tâches complexes sans être explicitement programmées. Les algorithmes d'apprentissage supervisé, en particulier la classification, sont largement utilisés dans divers domaines tels que la reconnaissance d'images, la biologie, la finance et bien d'autres encore. Les SVM (Support Vector Machines) sont l'un des algorithmes de classification les plus populaires utilisés dans l'apprentissage automatique en raison de leur capacité à gérer les problèmes de classification linéaire et non linéaire. Les SVM sont particulièrement utiles dans les situations où les données d'entraînement sont complexes et le bruit est important. Dans ce rapport, nous présentons une étude sur l'utilisation de SVM pour la classification de données de spam et de non-spam. Nous discuterons des principes de base de SVM, de leur fonctionnement, de leur entraînement, de leur validation et de leur application pour la classification de données de spam et de non-spam. Nous montrerons comment les SVM peuvent être utilisés pour améliorer la précision de la classification, tout en minimisant les faux positifs et les faux négatifs.

CHAPITRE

ONE

PRINCIPE ET CAS D'UTILISATION

Le principe des SVM consiste à ramener un problème de classification ou de discrimination à un hyperplan (feature space) dans lequel les données sont séparées en plusieurs classes dont la frontière est la plus éloignée possible des points de données (ou "marge maximale"). D'où l'autre nom attribué aux SVM : les séparateurs à vaste marge. Le concept de frontière implique que les données soient linéairement séparables. Pour y parvenir, les support vector machines font appel à des noyaux, c'est-à-dire des fonctions mathématiques permettant de projeter et séparer les données dans l'espace vectoriel, les "vecteurs de support" étant les données les plus proches de la frontière. C'est la frontière la plus éloignée de tous les points d'entraînement qui est optimale, et qui présente donc la meilleure capacité de généralisation.

Les cas d'utilisation de la méthode SVM sont très variés et couvrent de nombreux domaines, tels que la reconnaissance de caractères manuscrits, la détection de fraude bancaire, la prédiction de la qualité de l'air, la détection de cancer et bien d'autres encore.

En reconnaissance de caractères manuscrits, la méthode SVM est utilisée pour identifier les caractères dans une image numérique en se basant sur un ensemble d'exemples d'images pré-classifiées. Dans le domaine de la détection de fraude bancaire, la méthode SVM est utilisée pour identifier les transactions suspectes et les activités frauduleuses en se basant sur des schémas d'activités frauduleuses connus.

Dans le domaine de la prédiction de la qualité de l'air, la méthode SVM est utilisée pour prédire les niveaux de pollution de l'air en se basant sur des données historiques de qualité de l'air et des données météorologiques. Dans le domaine de la détection de cancer, la méthode SVM est utilisée pour identifier les tumeurs cancéreuses dans des images radiographiques en se basant sur un ensemble d'exemples pré-classifiés.

En somme, la méthode SVM est utilisée dans de nombreux domaines pour résoudre des problèmes de classification et de régression. Son utilisation est très répandue dans les domaines de la reconnaissance de formes, de la vision par ordinateur et de la bioinformatique, mais elle trouve également des applications dans des domaines tels que la finance, l'environnement et la médecine.

CHAPITRE

TWO

SVMS BINAIRES

2.1 Cas linéairement séparable

Le cas le plus simple est celui où les données d'entrainement viennent uniquement de deux classes différentes (+1 ou -1), on parle alors de classification binaire. L'idée des SVMs est de rechercher un hyperplan (droite dans le cas de deux dimensions) qui sépare le mieux ces deux classes. Si un tel hyperplan existe, c'est-à-dire si les données sont linéairement séparables, on parle d'une machine à vecteur support à marge dure (Hard margin). L'hyperplan séparateur est représenté par l'équation suivante :

$$H(x) = w^T x + b$$

Où w est un vecteur de m dimensions et b est un terme. La fonction de décision, pour un exemple x, peut être exprimée comme suit :

$$Classe = 1 \quad Si\ H(x) > 0$$
$$Classe = -1 \quad Si\ H(x) < 0$$

Puisque les deux classes sont linéairement séparables, il n'existe aucun exemple qui se situe sur l'hyperplan, c-à-d qui satisfait $H(x) = 0$. Il convient alors d'utiliser la fonction de décisions suivante :

$$Classe = 1 \quad Si\ H(x) > 1$$
$$Classe = -1 \quad Si\ H(x) < -1$$

Les valeurs +1 et -1 à droite des inégalités peuvent être des constantes quelconques +a et -a, mais en divisant les deux parties des inégalités par a, on trouve les inégalités précédentes qui sont équivalentes à l'équation suivante :

$$y_i(w^T x_i + b) \geq 1; i = 1 :: n$$

L'hyperplan $w^T x + b = 0$ représente un hyperplan séparateur des deux classes, et la distance entre cet hyperplan et l'exemple le plus proche s'appelle la marge. La région qui se trouve entre les deux hyperplans $w^T x + b = -1$ et $w^T x + b = +1$ est appelée la région de généralisation de la machine d'apprentissage. Plus cette région est importante, plus est la capacité de généralisation de la machine. La maximisation de cette région est l'objectif de la phase d'entrainement qui consiste, pour la méthode SVM, à rechercher l'hyperplan qui maximise la région de généralisation c-à-d la marge. Un tel hyperplan est appelé "hyperplan de séparation optimale". En supposant que les données d'apprentissage ne contiennent pas des données bruitées (mal-étiquetées) et que les données de test suivent la même probabilité que celle des données d'entraînement, l'hyperplan de marge maximale va certainement maximiser la capacité de généralisation de la machine d'apprentissage.
La détermination de l'hyperplan optimal passe par la détermination de la distance euclidienne minimale entre l'hyperplan et l'exemple le plus proche des deux classes. Puisque le vecteur w est orthogonal sur l'hyperplan séparateur, la droite parallèle à w et reliant un exemple x à l'hyperplan est donnée par la formule :

$$\frac{aw}{\|w\|} + x = 0$$

Où a représente la distance entre x et l'hyperplan. La résolution de cette équation, donne :

$$a = -\frac{w^T x + b}{\|w\|}$$

La distance de tout exemple de l'hyperplan doit être supérieure ou égale à la marge δ :

$$\frac{y_i(w^T x_i + b)}{\|w\|} \geq \delta$$

Si une paire $(w; b)$ est une solution alors $(aw; ab)$ est une solution aussi où a est un scalaire. On impose alors la contrainte suivante :

$$\|w\|\delta \geq 1$$

Pour trouver l'hyperplan séparateur qui maximise la marge, on doit déterminer, à partir des deux dernières inégalités, le vecteur w qui possède la norme euclidienne minimale et qui vérifie la contrainte de l'équation, de bonne classification des exemples d'entrainement. L'hyperplan séparateur optimal peut être obtenu en résolvant le problème de l'équation :

$$\begin{cases} Minimiser\ \frac{1}{2}\ \|w\|^2 \\ sous\ contraintes \\ y_i(w^T x_i + b) \geq 1 \forall i = 1..n \end{cases}$$

Remarquons que nous pouvons obtenir le même hyperplan même en supprimant toutes les données qui vérifient l'inégalité de la contrainte. Les données qui vérifient l'égalité de la contrainte s'appellent les vecteurs supports, et ce sont ces données seules qui contribuent à la détermination de l'hyperplan. Dans la figure, les données qui se trouvent sur les deux droites +1 et -1 représentent les vecteurs supports.

Le problème de l'équation est un problème de programmation quadratique avec contraintes linéaires. Dans ce problème, les variables sont w et b, c-à-d que le nombre de variables est égal à m + 1. Généralement, le nombre de variables est important ce qui ne permet pas d'utiliser les techniques classiques de programmation quadratique. Dans ce cas le problème est convertit en un problème dual équivalent sans contraintes de l'équation suivante qui introduit les multiplicateurs de Lagrange :

$$Q(w; b; \alpha) = \tfrac{1}{2} w^T w - \textstyle\sum_{i=1}^{n} \alpha_i (y_i(w^T x_i + b) - 1)$$

O$\grave{u}$ les α_i sont les multiplicateurs non négatifs de Lagrange. L'optimum de la fonction objective Q peut être obtenu en la minimisant par rapport à w et b et en la maximisant par rapport aux α_i. A l'optimum de la fonction objective, ses

dérivées par rapports aux variables w et b s'annulent ainsi que le produit des α_i aux contraintes :

$$\begin{cases} \frac{\partial Q(w,b,\alpha)}{\partial w} = 0 & (a) \\ \frac{\partial Q(w,b,\alpha)}{\partial b} = 0 & (b) \\ \alpha_i \left\{ y_i \left(w^T x_i + b \right) - 1 \right\} = 0 & (c) \\ \alpha_i \geq 0 & (d) \end{cases}$$

De (a) on déduit :

$$\begin{cases} w = \sum_{i=1}^{n} \alpha_i y_i x_i \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases}$$

En remplaçant dans la fonction objective, on obtient le problème dual à maximiser suivant :

$$\begin{cases} \text{Maximiser} & Q(\alpha) = \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{Sous contraintes} & \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & \alpha_i \geq 0 \end{cases}$$

Si le problème de classification est linéairement séparable, une solution optimale pour les α_i existe. Les exemples ayant des $\alpha_i \neq 0$ représentent les vecteurs supports appartenant aux deux classes. La fonction de décision est donnée par :

$$H(x) = \sum_{S} \alpha_i y_i x^T x_i + b$$

Où S représente l'ensemble des vecteurs supports. b peut être calculé à partir de n'importe quel vecteur support par l'équation :

$$b = y_i - w^T x_i$$

D'un point de vue précision, on prend la moyenne de b pour tous les vecteurs supports :

$$b = \frac{1}{|S|} \sum_{i \in S} y_i - w^T x_i$$

La fonction de décision H peut être calculée, donc, pour chaque nouvel exemple x par la fonction $H(x)$ et la décision peut être prise comme suit :

$$\begin{cases} x \in classe + 1 \; si \; H(x) > 0 \\ x \in classe - 1 \; si \; H(x) < 0 \\ x \; inclassifiable \; si \; H(x) = 0 \end{cases}$$

La zone $-1 < H(x) < 1$ est appelée la zone de généralisation. Si on prend un exemple x_k de l'ensemble d'entrainement appartenant à la classe y_k et on calcule sa fonction de décision $H(x_k)$, on peut se trouver dans l'un des cas suivants :

1. $y_k * H(x_k) > 1$: dans ce cas l'exemple est bien classé et ne se situe pas dans la zone de la marge. Il ne représente pas un vecteur support.
2. $y_k * H(x_k) = 1$: dans ce cas l'exemple est bien classé et se situe aux frontières de la zone de la marge. Il représente un vecteur support.
3. $0 < y_k * H(x_k) < 1$: dans ce cas l'exemple est bien classé et se situe dans de la zone de la marge. Il ne représente pas un vecteur support.
4. $y_k * H(x_k) < 0$: dans ce cas l'exemple se situe dans le mauvais coté, il est mal classé et ne représente pas un vecteur support.

2.2 Cas non linéairement séparable

En réalité, un hyperplan séparateur n'existe pas toujours, et même s'il existe, il ne représente pas généralement la meilleure solution pour la classification. En plus une erreur d'étiquetage dans les données d'entrainement (un exemple étiqueté +1 au lieu de -1 par exemple) affectera crucialement l'hyperplan.

Dans le cas où les données ne sont pas linéairement séparables, ou contiennent du bruit (outliers : données mal étiquetées) les contraintes ne peuvent être

vérifiées, et il y a nécessité de les relaxer un peu. Ceci peut être fait en admettant une certaine erreur de classification des données ce qui est appelé "SVM à marge souple (Soft Margin)".

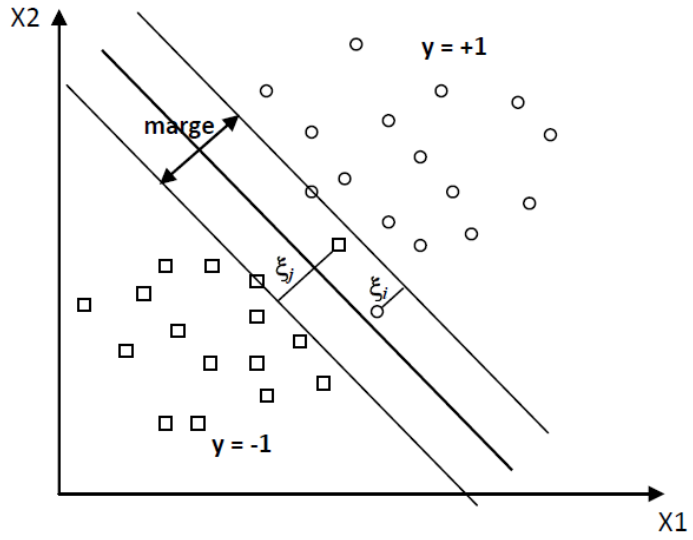


FIGURE 2.1 – SVM binaire à marge souple.

On introduit alors sur les contraintes des variables ξ_i dites de relaxation pour obtenir la contrainte de l'équation :

$$y_i \left(w^T x_i + b\right) \geq 1 - \xi_i, i = 1..n$$

Grâce aux variables de relaxation non négatives ξ_i, un hyperplan séparateur existera toujours.

Si $\xi_i < 1, x_i$ ne respecte pas la marge mais reste bien classé, sinon x_i est mal classé par l'hyperplan. Dans ce cas, au lieu de rechercher uniquement un hyperplan séparateur qui maximise la marge, on recherche un hyperplan qui minimise aussi la somme des erreurs permises c-à-d minimiser $Q(w) = \sum_{i=1}^{n} \xi_i$. Le problème dual devient donc :

$$\begin{cases} \text{Minimiser} & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \\ \text{sous contraintes} & \\ y_i \left(w^T x_i + b\right) \geq 1 - \xi_i & i = 1..n \\ \xi_i \geq 0 & \end{cases}$$

Où C est un paramètre positif libre (mais fixe) qui représente une balance entre les deux termes de la fonction objective (la marge et les erreurs permises)

c-à-d entre la maximisation de la marge et la minimisation de l'erreur de classification. On obtient le problème dual de l'équation suivante où on introduit les multiplicateurs de Lagrange α_i et β_i :

$$Q(w, b, \alpha, \xi, \beta) = \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i y_i \left(w^T x_i + b \right) - 1 + \xi_i - \sum_{i=1}^{n} \beta_i \xi_i$$

À la solution optimale, les dérivées par rapport aux variables w, b, α, β s'annulent ainsi que le produit des contraintes aux multiplicateurs. Les conditions suivantes sont alors vérifiées :

$$\begin{cases} \frac{\partial Q(w,b,\xi,\alpha,\beta)}{\partial w} = 0 & (a) \\ \frac{\partial Q(w,b,\xi,\alpha,\beta)}{\partial b} = 0 & (b) \\ \alpha_i \left\{ y_i \left(w^T x_i + b \right) - 1 + \xi_i \right\} = 0 & (c) \\ \beta_i \xi_i = 0 & (d) \\ \alpha_i \geq 0; \beta_i \geq 0; \xi_i \geq 0 & (e) \end{cases}$$

On déduit :

$$\begin{cases} w = \sum_{i=1}^{n} \alpha_i y_i x_i \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \\ \alpha_i + \beta_i = 0 \end{cases}$$

En remplaçant cette équation dans la fonction objective, on obtient le problème dual suivant :

$$\begin{cases} \text{Maximiser} & Q(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{sous contraintes} & \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \end{cases}$$

La seule différence avec la SVM à marge dure est que les α_i ne peuvent pas dépasser C, ils peuvent être dans l'un des trois cas suivants :

1. $\alpha_i = 0 \Rightarrow \beta_i = C \Rightarrow \xi_i = 0$: x_i est bien classé,
2. $0 < \alpha_i < C \Rightarrow \beta_i > 0 \Rightarrow \xi_i = 0 \Rightarrow y_i \left(w^T x_i + b \right) = 1$: x_i est un vecteur support et est appelé dans ce cas vecteur support non borné (unbounded),

3. $\alpha_i = C \Rightarrow \beta_i = 0 \Rightarrow \xi_i \geq 0 \Rightarrow y_i \left(w^T x_i + b\right) = 1 - \xi_i$: x_i est un vecteur support appelé dans ce cas vecteur support borné (bounded). Si $0 \leq \xi_i < 1, x_i$ est bien classé, sinon x_i est mal classé.

Ces conditions sur les α_i sont appelées les conditions de Karush-Kuhn-Tucker (KKT), elles sont très utilisées par les algorithmes d'optimisation pour rechercher les α_i optimums et par conséquent l'hyperplan optimal. La fonction de décision est alors calculée de la même manière que dans le cas des SVMs à marge dure mais uniquement à base des vecteurs supports non bornés par :

$$H(x) = \sum_{i \in U} \alpha_i y_i x_i^T x + b$$

Pour les vecteurs supports non bornés, nous avons :

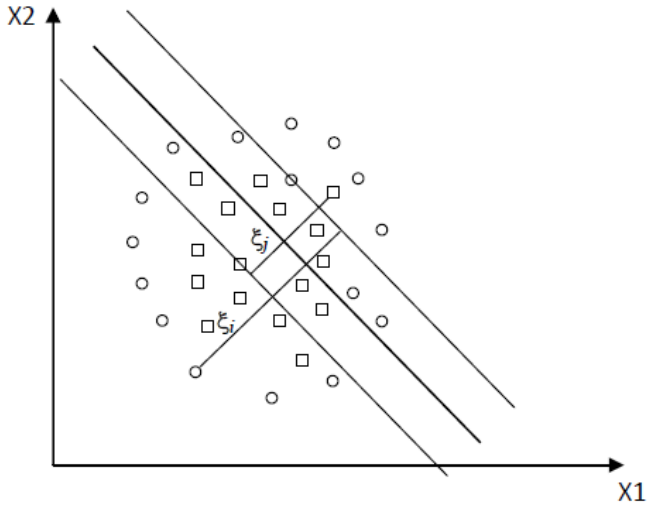
$$b = y_i - w^T x_i$$

Pour garantir une bonne précision, on prend la moyenne de b pour tous les vecteurs supports non bornés :

$$b = \frac{1}{|U|} \sum_{i \in U} y_i - w^T x_i$$

2.2.1 Utilisation des noyaux

Le fait d'admettre la mal-classification de certains exemples, ne peut pas toujours donner une bonne généralisation pour un hyperplan même si ce dernier est optimisé.



Plutôt qu'une droite, la représentation idéale de la fonction de décision serait une représentation qui colle le mieux aux données d'entrainement.

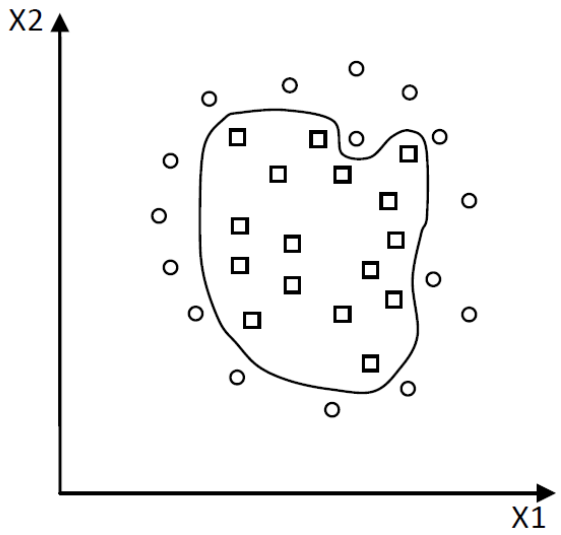


FIGURE 2.2 – Représentation idéale de la fonction de décision

La détermination d'une telle fonction non linéaire est très difficile voire impossible. Pour cela les données sont amenées dans un espace où cette fonction devient linéaire, cette astuce permet de garder les mêmes modèles de problèmes d'optimisation vus dans les sections précédentes, utilisant les SVMs basées essentiellement sur le principe de séparation linéaire. Cette transformation d'espace est réalisée souvent à l'aide d'une fonction $F = \{\phi(x) \mid x \in X\}$ appelé "Mapping function" et le nouvel espace est appelé espace de caractéristiques "Features space".

Dans ce nouvel espace de caractéristiques, la fonction objective à optimiser devient :

$$Q(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left\langle \phi\left(x_i\right), \phi\left(x_j\right) \right\rangle$$

Où $\left\langle \phi\left(x_i\right), \phi\left(x_j\right) \right\rangle$ est le produit scalaire des deux images des vecteurs x_i et x_j dans le nouvel espace et dont le résultat est un scalaire. Dans le calcul de l'optimum de la fonction, on utilise une astuce appelée "Noyau" ("Kernel"), au lieu de calculer $\phi\left(x_i\right), \phi\left(x_j\right)$ et leur produit scalaire, on calcule plutôt une fonction $K\left(x_i, x_j\right)$ qui représente à la fois les deux transformations (qui peuvent être inconnues) et leur produit scalaire. Cette fonction permet de surmonter

le problème de détermination de la transformation ϕ et permet d'apprendre des relations non linéaires par des machines linéaires. En pratique, il existe certains noyaux qui sont très utilisés et qui sont considérés comme standards. Une fois le noyau choisi, la fonction objective peut être calculée comme suit :

$$Q(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K\left(x_i, x_j\right)$$

Et la fonction de décision devient :

$$H(x) = \sum_{i \in S} \alpha_i y_i K\left(x_i, x\right) + b$$

Où S représente l'ensemble des vecteurs supports.

2.2.2 Exemples de noyaux

- Noyau linéaire : Si les données sont linéairement séparables, on n'a pas besoin de changer d'espace, et le produit scalaire suffit pour définir la fonction de décision :

$$K\left(x_i, x_j\right) = x_i^T x_j$$

- Noyau polynomial : Le noyau polynomial élève le produit scalaire à une puissance naturelle d :

$$K\left(x_i, x_j\right) = \left(x_i^T x_j\right)^d$$

Si $d = 1$ le noyau devient linéaire. Le noyau polynomial dit non homogène $K\left(x_i, x_j\right) = \left(x_i^T x_j + C\right)^d$ est aussi utilisé.

- Noyau RBF : Les noyaux RBF (Radial Basis functions) sont des noyaux qui peuvent être écrits sous la forme : $K\left(x_i, x_j\right) = f\left(d\left(x_i, x_j\right)\right)$ où d est une métrique sur X et f est une fonction dans $\Re$. Un exemple des noyaux RBF est le noyau Gaussien :

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

Où σ est un réel positif qui représente la largeur de bande du noyau.

CHAPITRE

THREE

SVMS MULTICLASSE

Les machines à vecteur support sont dans leur origine binaires. Cependant, les problèmes du monde réel sont dans la plupart des cas multiclasse, l'exemple le plus simple en est la reconnaissance des caractères optiques (OCR). Dans de tels cas, on ne cherche pas à affecter un nouvel exemple à l'une de deux classes mais à l'une parmi plusieurs, c-à-d que la décision n'est plus binaire et un seul hyperplan ne suffit plus. Les méthodes des machines à vecteur support multiclasse, réduisent le problème multiclasse à une composition de plusieurs hyperplans biclasses permettant de tracer les frontières de décision entre les différentes classes. Ces méthodes décomposent l'ensemble d'exemples en plusieurs sous ensembles représentant chacun un problème de classification binaire. Pour chaque problème un hyperplan de séparation est déterminé par la méthode SVM binaire. On construit lors de la classification une hiérarchie des hyperplans binaires qui est parcourue de la racine jusqu'à une feuille pour décider de la classe d'un nouvel exemple. On trouve dans la littérature plusieurs méthodes de décomposition :

3.1 Une-contre-reste

C'est la méthode la plus simple et la plus ancienne. Selon la formulation de Vapnik, elle consiste à déterminer pour chaque classe k un hyperplan Hk(wk ; bk) la séparant de toutes les autres classes. Cette classe k est considérée comme étant la classe positive (+1) et les autres classes comme étant la classe négative (-1), ce qui résulte, pour un problème à K classes, en K SVM binaires. Un hyper-

plan Hk est défini pour chaque classe k par la fonction de décision suivante :

$$H_k(x) = \text{signe}\left(\langle w_k, x\rangle + b_k\right)$$
$$= \begin{cases} +1 & \text{si} \quad f_k(x) > 0 \\ 0 & \text{sinon} \end{cases}$$

La valeur retournée de l'hyperplan permet de savoir si x appartient à la classe k ou non. Dans le cas où il n'appartient pas à $k(H_k(x) = 0)$, nous n'avons aucune information sur l'appartenance de x aux autres classes. Pour le savoir, on présente x à tous les hyperplans, ce qui donne la fonction de décision de l'équation suivante :

$$k^* = \underbrace{\text{Arg}}_{(1\leq k\leq K)} \text{Max}\left(H_k(x)\right)$$

Si une seule valeur $H_k(x)$ est égale à 1 et toutes les autres sont égales à 0, on conclut que x appartient à la classe k. Le problème est que l'équation peut être vérifiée pour plus d'une classe, ce qui produit des régions d'ambiguïté, et l'exemple x est dit non classifiable. La figure suivante représente un cas de séparation de 3 classes. Pour surmonter cette situation, la méthode 1vsR utilise

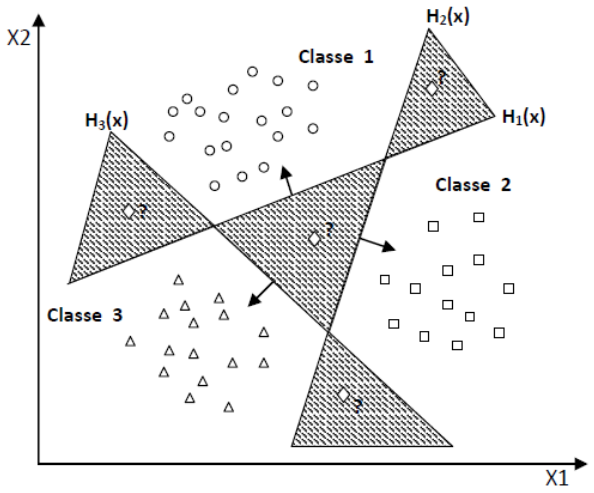


FIGURE 3.1 – Approche une-contre-reste avec des zones d'indécision

le principe de "le gagnant prend tout" ("winner-takes-all") : la classe k retenue est celle qui maximise $f_k(x) = \langle w_k; x_i\rangle + b_k$ de l'équation :

$$k^* = \underbrace{\text{Arg}}_{(1\leq k\leq K)} \text{Max}\left(\langle w_k; x_i\rangle + b_k\right)$$

Géométriquement interprétée, tout nouvel exemple x est affecté à la classe dont l'hyperplan est le plus loin de x, parmi les classes ayant H(x) = 1.

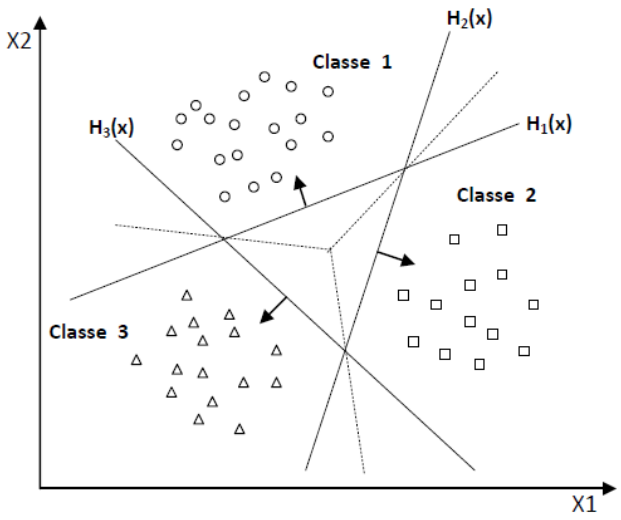


FIGURE 3.2 – Résolution des cas d'indécision dans la méthode 1vsR

La méthode 1vsR peut être utilisée pour découvrir même les cas de rejet où un exemple n'appartient à aucune des K classes. Pour cela, on prend les deux fonctions de décision les plus élevées, puis on calcule leur différence, si elle est au dessous d'un certain seuil, l'exemple est rejeté. Souvent, la méthode 1vsR est critiquée à cause de son asymétrie, puisque chaque hyperplan est entrainé sur un nombre d'exemples négatifs beaucoup plus important que le nombre d'exemples positifs. Par exemple dans le cas de l'OCR, le classifieur du caractère 'A' est entrainé sur des exemples positifs représentant 'A' et des exemples négatifs représentant tous les autres caractères. La méthode une contre une suivante est une méthode symétrique qui corrige ce problème.

3.2 Une-contre-une

Cette méthode, appelée aussi "pairwise", revient à Kner et ses co-auteurs qui l'ont proposée pour les réseaux de neurones. Elle consiste à utiliser un classifieur pour chaque paire de classes. Au lieu d'apprendre K fonctions de décisions, la méthode 1vs1 discrimine chaque classe de chaque autre classe, ainsi $K(K-1)/2$ fonctions de décisions sont apprises. Pour chaque paire de classes (k ; s), la méthode 1vs1 définit une fonction de décision binaire h_{ks} : $\mathbb{R} \longrightarrow \{-1, +1\}$. L'affectation d'un nouvel exemple se fait par liste de vote. On teste un exemple par le calcul de sa fonction de décision pour chaque hyperplan. Pour chaque test, on vote pour la classe à laquelle appartient l'exemple

(classe gagnante). On définit pour le faire la fonction de décision binaire $H_{ks}(x)$ de l'équation suivante :

$$H_{ks}(x) = \text{signe}\,(f_{ks}(x)) = \begin{cases} +1 & \text{si } f_{ks}(x) > 0; \\ 0 & \text{sinon} \end{cases}$$

Sur la base des $K(K-1)/2$ fonctions de décision binaires, on définit K autres fonctions de décision :

$$H_k(x) = \sum_{s=1}^{m} H_{ks}(x)$$

Un nouvel exemple est affecté à la classe la plus votée. La règle de classification d'un nouvel exemple x est donnée par l'équation :

$$k^* = \underbrace{\text{Arg}}_{(1 \leq k \leq K)} \text{Max}\,(H_k(x))$$

Malheureusement, cette fonction peut être vérifiée pour plusieurs classes, ce qui produit des zones d'indécisions. La méthode de vote affecte dans ce cas, un exemple aléatoirement à l'une des classes les plus votées. La Figure suivante représente un exemple de classification de trois classes avec la zone d'indécision. Bien que La méthode 1vs1 utilise, pour l'entrainement, un nombre

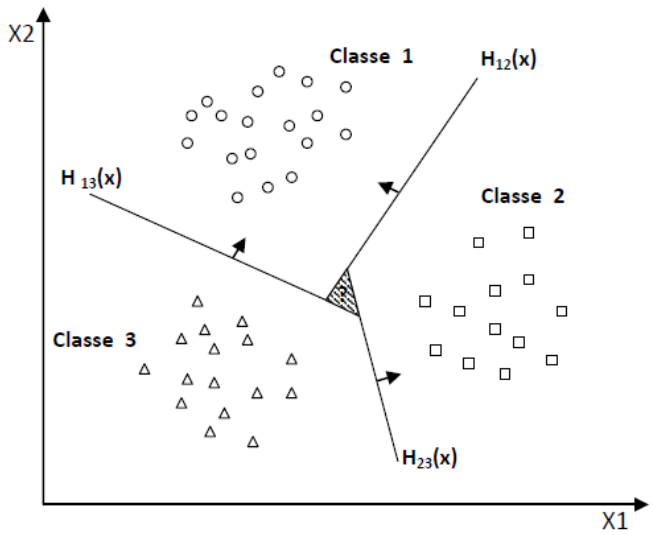


FIGURE 3.3 – Approche une-contre-une

plus important d'hyperplans que la méthode 1vsR, elle est souvent plus rapide. Cela est du, d'une part, au nombre limité d'exemples utilisés pour entrainer

chaque hyperplan, et d'autre part, à la simplicité des problèmes à résoudre. En effet, chaque deux classes prises à part sont moins chevauchées que toutes les classes.

CHAPITRE

FOUR

AVANTAGES ET DES INCONVÉNIENTS

La méthode SVM présente des avantages et des inconvénients, comme toute méthode d'apprentissage automatique.

Les avantages de la méthode SVM sont les suivants :

Efficacité : la méthode SVM est très efficace pour résoudre les problèmes de classification et de régression, même pour les données de grande dimensionnalité.

Régularisation : la méthode SVM utilise une technique de régularisation qui permet d'éviter le surapprentissage, ce qui est particulièrement utile lorsque les données sont bruyantes ou déséquilibrées.

Adaptabilité : la méthode SVM est adaptable à différentes tâches de classification et de régression, grâce à l'utilisation de différents types de noyaux.

Interprétabilité : contrairement à certaines autres méthodes d'apprentissage automatique, la méthode SVM fournit une interprétation claire des résultats.

Robustesse : la méthode SVM est robuste aux données aberrantes, ce qui en fait une méthode utile pour les données bruyantes.

Cependant, la méthode SVM présente également quelques inconvénients, tels que :

Sensibilité au choix des paramètres : la méthode SVM est sensible au choix des paramètres, tels que le choix du noyau et les paramètres de régularisation.

Temps de calcul : la méthode SVM peut prendre beaucoup de temps de calcul pour les grands ensembles de données.

Difficulté à gérer les données manquantes : la méthode SVM a du mal à

traiter les données manquantes ou incomplètes.

Pas approprié pour les problèmes de régression non linéaire : la méthode SVM est plus appropriée pour les problèmes de classification que pour les problèmes de régression non linéaire.

CONCLUSION

En conclusion, la méthode SVM est une technique de classification puissante et efficace qui peut être utilisée dans de nombreux domaines, tels que la reconnaissance de formes, la vision par ordinateur, la bioinformatique et bien d'autres encore. Elle se distingue des autres méthodes de classification en utilisant des hyperplans pour séparer les données en classes distinctes.

La méthode SVM offre de nombreux avantages, tels que son efficacité pour résoudre les problèmes de classification, sa régularisation pour éviter le surapprentissage, sa capacité à gérer les données de grande dimensionnalité et sa robustesse aux données bruyantes. Cependant, elle a également quelques inconvénients, tels que sa sensibilité au choix des paramètres et son temps de calcul pour les grands ensembles de données.

Dans l'ensemble, la méthode SVM est une méthode très utile pour la classification, qui est largement utilisée dans de nombreux domaines. Elle a aidé les scientifiques, les ingénieurs et les professionnels de l'informatique à résoudre des problèmes complexes de classification avec succès. Cependant, il est important de comprendre les avantages et les inconvénients de la méthode SVM et de prendre en compte les spécificités de chaque problème de classification avant de choisir cette méthode pour résoudre un problème de classification donné.