Vers des Machines à Vecteurs de Support "Actionnables" : Une Approche Fondée sur le Classement

Ansaf Salleb-Aouissi Bert C. Huang David L. Waltz

Center for Computational Learning Systems
Columbia University, New York, NY 10115
{ansaf@ccls, bert@cs, waltz@ccls}.columbia.edu
http://www.ccls.columbia.edu/

Résumé. Une des principales critiques que l'on puisse faire aux Séparateurs à Vaste Marge (SVM) est le manque d'intelligibilité des résultats. En effet, il s'agit d'une technique "boite noire" qui ne fournit pas d'explications ni d'indices quant aux raisons d'une classification. Les résultats doivent être pris tels quels en faisant confiance au système qui les a produits. Pourtant selon notre expérience pratique, les experts du domaine préfèrent largement une méthode d'apprentissage avec explications et recommandation d'actions plutôt qu'une boite noire, aussi performante et prédictive soit-elle.
Dans cette thématique, nous proposons une nouvelle approche qui consiste a rendre les SVM plus "actionnables". Ce but est atteint en couplant des modèles de classement des résultats des SVM à des méthodes d'apprentissage de concepts. Nous présentons une application de notre méthode sur diverses données dont des données médicales concernant des patients de l'athérosclérose. Nos résultats empiriques semblent très prometteurs et montrent l'utilité de notre approche quant à l'intelligibilité et l'actionnabilité des résultats produits par SVM.
Mots clés : Séparateurs à Vaste Marge (SVM), classement, apprentissage de règles, actionnabilité.

1 Introduction

Durant la dernière décade, les machines à vecteurs de support (ou Séparateurs à Vaste Marge : SVM) ont connu un immense succès, principalement comme puissants classifieurs. Cependant, une des principales limitations des SVM est le manque d'intelligibilité des résultats. En effet, les SVM ne produisent pas d'explications ni d'indices quant aux raisons d'une classification et les résultats produits doivent être pris tels quels, en faisant confiance au système. Nous proposons de rendre les SVM actionnables en classant (ordonnant) les exemples, pas seulement en les classifiant. En effet, les moyens d'action sont la plupart du temps limités, ce qui ne permet d'agir que sur une petite partie des exemples de la population. De plus, le classement peut être très utile pour "tamiser" les exemples d'apprentissage afin de ne garder que les exemples réellement importants, représentatifs des classes. L'idée sous-jacente de notre

méthodologie consiste à contraster les résultats ordonnés des SVM afin de découvrir les principales propriétés caractéristiques discriminantes entre la partie haute (désignée Top dans la suite) et la partie basse (désignée par Bottom) du classement produit par SVM.
Nous sommes concernés par le problème d'intelligibilité, car de notre expérience pratique, les experts du domaine sont clairement beaucoup plus confiants quand il s'agit d'agir sur des exemples hautement classés, si les raisons d'un tel classement sont fournies avec la liste ordonnée des exemples. De plus, il peut être aussi important de comprendre la partie basse du classement. Ceci peut être d'un grand intérêt pour l'expert du domaine pour diriger des actions et comprendre le système.

Le schéma général de notre approche est comme suit :

- Ordonner les exemples en utilisant les SVM.
- Créer deux sous-ensembles des données ordonnées : les n exemples du haut (Top n) et les m exemples du bas (Bottom m) de l'ordonnancement. Typiquement, $n = m$, et $||n + m|| = 5 - 20\%$ du total des exemples.
- Extraire l'ensemble des propriétés les plus importantes en analysant les motifs d'attributs dans les sous-ensembles Top et Bottom.

Notons qu'ignorer la partie milieu de l'ordonnancement et se concentrer seulement sur les extrémités, permet de faciliter l'extraction de motifs intéressants. Nous voulons identifier les caractéristiques des exemples du haut de la liste en les contrastant aux exemples du bas. En effet, il est possible d'analyser la fréquence relative des différentes propriétés (attribut=valeur) et de calculer l'importance de telles propriétés en utilisant des indices statistiques tels que le “Leverage”. Nous présentons une application de notre méthode sur diverse données dont des données médicales concernant des patients de l'athérosclérose. Nos résultats empiriques semblent très prometteurs et montrent l'utilité de notre approche quant à l'intelligibilité et l'actionnabilité des résultats produits par SVM.

2 État de l'Art

Un certain nombre de travaux récents (Barakat et Diederich (2004); Barakat et Bradley (2006); da Costa F. Chaves et al. (2005); Fung et al. (2005); Nunez et al. (2002); Fu et al. (2004); Zhang et al. (2005); Martens et al. (2007)) se sont intéressés au problème d'extraction d'explications à partir des SVM.

Nunez et al. (Nunez et al. (2002)) ont suggéré une méthode géométrique permettant de transformer un classifieur SVM en règles de classification. Pour ce faire, les auteurs utilisent les k-moyennes pour déterminer un ensemble de vecteurs prototypes. Lorsqu'ils sont combinés avec les vecteurs de support se trouvant sur la marge SVM, ces vecteurs aident à construire les frontières d'ellipsoïdes ou d'hyper-rectangles. Ces derniers sont traduits en équations *Si-Alors* ou à des règles d'intervalles respectivement. Une approche similaire est proposée par Zhang et al. (Zhang et al. (2005)) et suggère un algorithme pour l'extraction de règles d'hyper-rectangles (HRE) pour SVM. La principale différence avec l'approche précédente est que le regroupement des vecteurs de support (Ben-Hur et al. (2002)) est utilisé pour trouver les vecteurs prototypes de chaque classe au lieu des k-moyennes. Ceci évite de choisir le nombre de groupes à priori.

Barakat et Bradley (Barakat et Diederich (2004)) combinent les arbres de décision aux SVM pour produire les explications. Ceci est réalisé comme suit : d'abord, construire un classifieur SVM. Ensuite, sélectionner les vecteurs de support générés par le modèle et écarter

leurs étiquettes de classes. Le modèle SVM est ensuite utilisé pour prédire la classe des vecteurs de support ce qui conduit à une nouvel ensemble de données. Enfin, construire un arbre de décision en utilisant les nouvelles données pour produire des règles symboliques. Les règles de décision ainsi produites sont alors évaluées sur un ensemble test afin de vérifier que les nouveaux exemples sont classés correctement par l'arbre de décision. Les mesures utilisées ici, principalement exactitude (accuracy) et fidélité, ont été étendues dans (Barakat et Bradley (2006)) à l'aire sous la courbe ROC.

Fung et al. (Fung et al. (2005)) proposent une approche pour convertir les SVM linéaires en un ensemble de règles ne se chevauchant pas de la forme : $\wedge_{i=1}^{n} l_i \leq x_i < u_i$ équivalentes au classifieur linéaire. Ceci est réalisé en résolvant un simple problème de programmation linéaire a $2n$ variables, n étant le nombre d'attributs. Chaque règle représente un hyper-cube dans un espace de dimension n avec des surfaces à axes parallèles. L'ensemble des règles optimales est calculé soit en utilisant un critère de maximisation de volume de l'hyper-cube ou un critère de couverture qui maximise le nombre de points dans le demi-espace. Les règles sont exprimées sous forme de disjonctions de conjonctions.

Une approche d'extraction de règles floues est proposée par da Costa et al. dans (da Costa F. Chaves et al. (2005)). L'extraction de règles est accomplie en 3 étapes : d'abord les vecteurs de support sont projetés sur les axes des coordonnées. Ensuite, pour chaque coordonnée, un ensemble de règles floues est construit. Ceux-ci constituent les antécédents des règles. Finalement, pour chaque vecteur support, une règle floue est extraite. Deux mesures sont définies pour évaluer la qualité des règles générées : exactitude et couverture de règle floue. Cette approche génère autant de règles qu'il y a de vecteurs de support, ce qui peut conduire à un grand nombre de règles.

Fu et al. (Fu et al. (2004)) s'attaquent au problème d'extraction de règles *Si-Alors* à partir d'un SVM non linéaire. L'idée est comme suit : étant donné un vecteur support pour une classe, un hyper-rectangle est construit en utilisant les points d'intersection des vecteurs de support avec la bordure SVM. Les intervalles de l'hyper-rectangle conduisent a une règle initiale. Par la suite, cette règle est réglée pour exclure les exemples hors classes afin de réduire son taux d'erreur. Enfin, les règles sont fusionnées afin d'obtenir un ensemble de règles plus concis. Cette approche a l'avantage de mettre en évidence les attributs les plus importants dans les règles extraites. De plus, l'utilisation d'un critère de fidélité qui évalue à quel point un système fondé sur des règles est proche du classifieur SVM. Le principal désavantage de cette approche est qu'elle est particulièrement couteuse lorsque le nombre de vecteurs de support est élevé.

Récemment, Martens et al. (2007) ont conduit une étude comparative de différentes approches. Ils ont montré qu'il était possible d'atteindre un degré élevé d'exactitude presque aussi bon que celui obtenu avec les SVM tout en produisant un modèle interprétable.

L'approche que nous proposons diffère fondamentalement des approches décrites ci-dessus dans le sens où nous n'utilisons pas les vecteurs de support pour extraire les explications des SVM. A la place, nous focalisons plutôt sur les exemples se trouvant en haut (Top) et en bas (Bottom) des exemples ordonnés par SVM, ce qui est moins couteux que d'utiliser un ensemble de vecteurs de support potentiellement grand. Notre principal argument est que l'utilisation des vecteurs de support n'est pas nécessairement un bon choix. En effet, la zone autour de la marge SVM est très bruitée et le fait que les vecteurs de support séparent les classes ne veut nullement dire qu'ils sont représentatifs de ces classes.

3 Notre Approche

Notre approche se décompose en deux étapes. La première consiste à classer (ordonner) l'ensemble des exemples en utilisant les SVM. Cette étape est décrite en Section 3.1. La seconde étape concerne l'extraction des explications proprement dites et ce à partir de l'ensemble ordonné d'exemples. Nous décrivons cette étape dans 3.2 et donnons l'algorithme YSVM.

3.1 Classement via les SVM

Nous nous intéressons au problème de classement des données. Le terme classement désigne le processus qui consiste à considérer un ensemble de données et les classer dans un ordre significatif et utile. Le classement supervisé permet d'atteindre tel objectif en utilisant les attributs des exemples ainsi que leurs étiquettes de classe. Plus formellement, nous voudrions classer des exemples $(x_1, y_1), \ldots, (x_n, y_n)$, où $x_1, \ldots, x_n$ sont des vecteurs attributs décrivant les objets $o_1, \ldots, o_n$, et chaque objet o_i est étiqueté par la classe $y_i \in \{+1, -1\}$. Même si le but est de produire un classement, les données du problème sont similaires à ceux d'un problème de classification. C'est pourquoi, nous utilisons une méthode de classification (SVM) et convertissons les résultats fournis en classement. Plus précisément, nous classons les objets en triant les objets par leur valeurs de décision de SVM linéaires (Vapnik (1995)) :

$$\begin{aligned} \min_{w,\xi} \quad & \frac{1}{2}||w||^2 + C \sum_{y_i=-1} \xi_i + RC \sum_{y_j=+1} \xi_j \\ \textit{s. t.} \quad & \forall k,\ y_k[w^T x_k + b] \geq 1 - \xi_k \ . \end{aligned} \tag{1}$$

où les ξ sont des variables dites *ressort* qui pénalisent l'erreur commise et le paramètre C détermine le compromis entre régularisation et pénalisation des erreurs de classification. Le paramètre R ajuste la pénalité pour la classe positive. Puisque nous voudrions pénaliser les erreurs d'étiquetage d'un exemple par la proportion de la popultation de la classe, nous pouvons fixer le paramètre R par :

$$R = \frac{\text{nombre de vrais négatifs}}{\text{nombre de vrais positifs}} \tag{2}$$

Typiquement, SVM produit un classifieur qui étiquette les exemples x par $y = sign(w^T x + b)$, mais nous ne mettons pas de seuil à nos résultats de telle manière à pouvoir ordonner nos exemples selon la fermeté avec laquelle le classifieur prédit la classe de chaque exemple. Autrement dit, nous gardons plutôt le score donné par SVM à l'exemple et non pas seulement son signe. Nous utilisons les courbes ROC (Receiver Operating Characteristic) (Bradley (1997)) pour évaluer la qualité de notre classement. Ils procurent une bonne façon de mesurer la qualité du classement lorsque la seule vérité dont nous disposons est si un exemple appartient au haut du classement (étiqueté $+1$) ou plutôt en bas (étiqueté -1). ROC est essentiellement normalisé par la cardinalité de la classe ce qui est similaire à la normalisation de la fonction de perte (loss) que nous avons pris pour apprendre le SVM. La qualité d'une courbe ROC est facilement mesurée à l'aide de l'aire sous la courbe (Area Under the Curve : AUC), qui se trouve dans l'intervalle $[0, 1]$. Une aire de 0.5 peut être atteinte avec un classement aléatoire des données alors qu'une aire 1.0 est atteinte en ordonnant parfaitement les exemples positifs en haut et les négatifs en bas.

Exemple Considérons un ensemble de 100 composants électriques. Chaque composant est décrit par son numéro de série, âge, taille et fabriquant et est étiqueté par son statut de panne (label=1 pour *en panne*, -1 *autrement*). Un classement SVM permet d'ordonner les composants selon leur susceptibilité aux pannes. Le haut du classement aurait dans ce cas les composants les plus sensibles aux pannes alors que les composants du bas du classement ont une moindre tendance aux pannes. Ainsi, l'expert du domaine peut focaliser sur les n composants du haut du classement pour agir, par exemple en planifiant des inspections/remplacements.

3.2 Extraction d'Explications

La phase d'extraction d'explications consiste à sélectionner et comparer d'abord les exemples du haut et du bas du classement. Ainsi, nous groupons les exemples en trois ensembles, où les exemples les plus "purs", i.e. les "très positifs" et les "très négatifs" se trouvent en haut et bas du classement respectivement. Les exemples du milieu du classement, autour de la marge SVM, sont plutôt les exemples bruités. Une question que l'on peut se poser est pourquoi considérer le haut et le bas du classement au lieu de comparer les exemples de la classe positive et ceux de la classe négative. La raison pour laquelle nous ne comparons pas simplement les deux classes est que certains exemples sont considérés comme étant négatifs alors que leur classe est en vérité inconnue. Ceci peut arriver fréquemment dans les applications réelles. Dans l'exemple précédent, nous ne sommes pas surs si les exemples négatifs le sont réellement : même si ces composants ne sont pas en panne, ils peuvent l'être dans un avenir proche. Les SVM ne devraient pas classer ces exemples profondément dans le demi-espace de la classe négative, mais plutôt autour de la marge séparant les deux classes. L'approche que nous suggérons ici écarterait ces exemples en focalisant seulement sur le haut et le bas du classement, là ou les étiquettes des exemples sont les plus fiables. Une fois que l'on a focalisé sur les extrémités du classement, nous recherchons l'ensemble des règles intéressantes de la forme :

$$R : Propriete \rightarrow Concept \tag{3}$$

où $Propriete$ est une paire attribut-valeur et $Concept$ est soit le concept "être en haut" ou "être en bas". Nous évaluons l'importance des propriétés en utilisant l'indice statistique de *Leverage* (Piatetsky-Shapiro (1991)). La raison pour laquelle nous avons choisi cette mesure est le fait qu'elle combine bien un haut pouvoir discriminant avec la capture des propriétés associées les plus fréquentes (support élevé). La mesure de Leverage a été utlisé dans d'autres tâches d'apprentissage telles que la caractérisation (voir par exemple Turmeaux et al. (2003)), On la trouvera aussi dans la littérature avec d'autres noms tels que *Nouveauté*. Le Leverage de la règle ci-dessus est donné par :

$$Leverage(R) = P(Propriete \wedge Concept) - P(Propriete) * P(Concept) \tag{4}$$

La mesure de Leverage évalue la proportion d'exemples additionnels couverts par la partie gauche et droite de la règle au dessus de ceux attendus si les deux cotés de la règle sont indépendants l'un de l'autre. Clairement, nous avons $-0.25 \leq Leverage(R) \leq +0.25$. Une propriété est dite intéressante pour un concept donné si la valeur de son Leverage est fortement positive. Ceci indique une forte association entre la propriété et le concept, alors qu'une forte valeur négative indique une forte association entre la propriété et la négation du concept. Dans

notre approche, le Leverage d'une règle peut etre estimé empiriquement par :

$$\frac{|\{x \in Concept \wedge \mathcal{V}_p(x) = true\}|}{|T \cup B|} - \frac{|\{x \in (T \cup B) \wedge \mathcal{V}_p(x) = true\}|}{|T \cup B|} * \frac{|Concept|}{|T \cup B|} \quad (5)$$

où Concept est soit T (l'ensemble Top) ou B (l'ensemble Bottom), p est une propriété et la notation $\mathcal{V}_p(x)$ est une fonction booléenne telle que pour un exemple x, nous avons $\mathcal{V}_p(x) = vrai$ ou $faux$ ce qui veut dire que la propriété p peut être satisfaite par x ou non.

Algorithme 1 : YSVM

Input :
- une liste classée d'exemples $\mathcal{L}$
- un seuil de Leverage MinLev
- Pourcentages Top et Bottom

Output :
- 2 ensembles de propriétés $\mathcal{P}_T$ and $\mathcal{P}_B$
- un ensembe d'histogrammes $\mathcal{H}$ un par attribut

```
1  T ← {exemples au Top de L}
2  B ← {exemples auBottom de L}
3  P ← ∅
4  pour chaque attribut faire
5      pour chaque valeur faire
6          p ← (attribut = valeur)
7          si Leverage(p → Top) ≥ MinLev alors
8              P_T ← P_T ∪ p
9          sinon si Leverage(p → Bottom) ≥ MinLev alors
10             P_B ← P_B ∪ p
11     h = Histogramme(attribut)
12     H = H ∪ h
13 retourner P_T, P_B, H
```

L'algorithme YSVM (donné en Algorithme 1) explore l'espace de recherche des propriétés possibles pour découvrir celles qui sont les plus importantes (parties gauches de règles) qui ont conduit SVM à classer des exemples avant d'autres. Pour une meilleure visualisation, l'algorithme extrait aussi un histogramme pour chaque attribut donnant la fréquence relative de ses valeurs dans Top et Bottom. Nous avons étendu YSM pour traiter des conjonctions de propriétés. Nous l'avons également étendu pour essayer différentes valeurs de Top et Bottom afin de sélectionner les tailles qui conduisent au plus grand nombre de propriétés intéressantes.

Exemple Considérons la liste ordonnée de composants électriques illustrée dans la Table 1. Extraire les principales propriétés comme le montre le Tableau 2 aide à identifier les facteurs de pannes. Par exemple, il peut être important de trouver des motifs dans les attributs des exemples ordonnés, comme savoir que des composants particuliers d'un certain fabriquant sont disproportionnellement responsables de failles. Le but ultime est d'aider l'expert dans son choix quant à l'achat de composants fiables, planification des inspections, etc.

	Rank	#Série	Age	Taille	Fabriquant
TOP	1	15B25	2	500	B
	2	13B28	8	500	B
	3	58C25	12	1000	C
	4	88A25	1	500	A
	5	18B22	17	500	B
	.	.	.	.	.
	.	.	.	.	.
	.	.	.	.	.
BOTTOM	96	63A11	27	500	A
	97	12A25	2	2000	A
	98	15A54	8	2000	A
	99	55A95	12	2000	A
	100	41B77	25	2500	B

TAB. 1 – *Exemple de liste ordonnée.*

Propriete	Freq_top	Leverage_top	Freq_bottom	Leverage_bottom
Taille=[2000,250)	0	-0.15	3	0.15
Taille=[500,1000)	4	0.15	1	-0.15
Fabriquant=A	1	-0.15	4	0.15
Age=[25,+inf)	0	-0.10	2	0.10
Fabriquant=A AND Taille=[2000,2500)	0	-0.15	3	0.15
Fabriquant=B AND Taille=[500,1000)	3	0.15	0	-0.15
Age=(-inf,3) AND Taille=[500,1000)	2	0.10	0	-0.10

TAB. 2 – *Ensemble de propriétés extraites pour l'exemple. Le fabriquant A est plutôt bon et en particulier pour les composants de grande taille alors que le fabriquant B a plutôt de petits et mauvais composants.*

4 Tests Empiriques

Nous avons implémenté YSVM en Python et avons conduit des tests empiriques sur divers benchmarks. Nous avons utilisé SVMLight[1] pour obtenir les classements SVM des différentes bases d'exemples.

4.1 Données Synthéthiques

Nous avons d'abord vérifié si YSVM capturait les "bons attributs". Pour cela, nous avons généré une base d'exemples aléatoires synthétiques de 1000 exemples. Chaque exemple est décrit par 50 attributs tels que $X \in \{-1, 1\}^{50}$. Les étiquettes de classes sont assignées comme suit : $Y = sign(\sum_{k=1}^{k=11} X_k)$. En d'autres termes, l'étiquette de classe est une combinaison linéaire des 11 premiers attributs parmi les 50 attributs. YSVM a réussi à re-découvrir ces attributs en focalisant seulement sur le top 5% et bottom 5% du classement avec Leverage minimum de 0.08. Il n'a pas été possible de découvrir ces attributs en utilisant tous les exemples jusqu'á ce qu'on ait réduit le Leverage minimum à une valeur très faible. On en conclut qu'il y a plus de pouvoir discriminant dans Top+Bottom que dans toute la base d'exemples.

[1] `http://svmlight.joachims.org/`

4.2 Données de l’Atherosclerose

Nous décrivons dans ce qui suit les tests que nous avons effectué sur des données médicales dans le contexte du projet Stulong [2]. Les données concernent une étude qui s’est étalée sur 20 ans concernant les facteurs de risque de l’athérosclérose dans une population de 1 419 hommes. Nous avons utilisé un ensemble de données préparé par (Lucas et al. (2002)) en se fixant le but d’identifier les principaux facteurs de risque de cette maladie. Les attributs utilisés sont donnés en Annexe 1 Tableau 4. Les patients ont été classés en 3 groupes : un groupe normal, un groupe à risque et enfin un groupe ayant la pathologie. Alors que cet attribut n’a pas été utilisé durant l’aprentissage du classement avec les SVM, les patients ayant la maladie et ceux qui sont à risques ont été classés avant les patients normaux. La cible d’apprentissage est l’attribut “death”. La Figure 1 montre la courbe ROC pour les résultats s’apprentissage avec les parties Top 10% et Bottom 10% mises en évidence où Top regroupe les patients les plus malades et Bottom ceux en meilleure santé par rapport à la maladie. Nous avons utilisé YSVM avec différentes valeurs de Top et Bottom et avons retenu les valeurs Top=5% et Bottom=5% donnant le plus grand nombre de propriétés intéressantes. Les résultats sont reportés en Figure 2 et les histogrammes associés en Annexe 1, Figure 5. Nos tests avec différents Top et Bottom ont montré que plus on augmentait ||T+B||, moins on obtenait de propriétés intéressantes. Conernant le temps d’éxecution, il faut compter quelques secondes pour générer des propriétés de taille ≤ 2. (par manque de place, les détails ne sont pas fournis ici). Les données de Stulong ont déjà fait l’objet de nombreuses publications (voir par exemple Lucas et al. (2002)). Les facteurs de l’athérosclérose sont connus pour être principalement la consommation et durée de consommation de tabac, le surpoids, l’activité physique alors qu’il n’y pas d’évidence quant à l’impact de la consommation d’alcool comme facteur de la maladie. Tous ces facteurs ont bien été découverts par YSVM comme le montre la Figure 3 et la Figure 5. Enfin, la Table 3 compare le nombre de propriétés intéressantes lorsque l’on utilise Top et Bottom versus tous les exemples d’apprentissage et les vecteurs de support. Pour ce test, nous avons fixé MinLeverage=0.08, et la taille des propriétés à ≤ 2, nous avons utilisé d’autres benchmarks de l’UCI (`http://mlearn.ics.uci.edu/`).

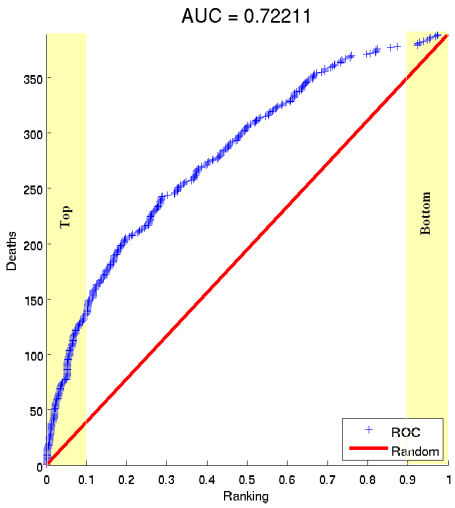

FIG. 1 – *Courbe ROC des données de l’athérosclerose. L’axe des X représente le classement normalisé.*

5 Discussion et Conclusion

Cet article décrit une approche simple et utile pour rendre les résultats des SVM actionnables. La principale originalité de notre approche est d’aider l’utilisateur à comprendre les

[2] `http://euromise.vse.cz/STULONG`

Propriété	Freq_top	Leverage_top	Freq_bottom	Leverage_bottom
ACTIV_JOB=1	16	-0.12	50	0.12
ACTIV_JOB=3	29	0.09	4	-0.09
TIME_JOB=5	36	-0.08	58	0.08
TIME_JOB=6	24	0.06	7	-0.06
BIRTH_YEAR=[25,30)	52	0.18	2	-0.18
BIRTH_YEAR=[35,40)	4	-0.17	51	0.17
ALCO_CONS=[1.10,1.20)	23	-0.05	37	0.05
TOBA_CONSO=0	0	-0.13	36	0.13
TOBA_CONSO=0.5	0	-0.07	20	0.07
TOBA_CONSO=0.85	33	0.10	5	-0.10
TOBA_CONSO=1.25	37	0.12	2	-0.12
TOBA_DURA=20	67	0.24	1	-0.24
EDUCATION=0	60	0.21	2	-0.21
EDUCATION=1	10	-0.21	68	0.21
TOBA_DURA=20 AND EDUCATION=0	57	0.20	0	-0.20
TOBA_DURA=20 AND RSK_TOBA=1 AND EDUCATION=0	57	0.20	0	-0.20

FIG. 2 – *Liste de quelques importantes propriétés découvertes dans les données de l'athérosclérose telles que extraires par YSVM. Une propriété est caractéristique du concept (Top/Bottom) si son Leverage fortement positif.*

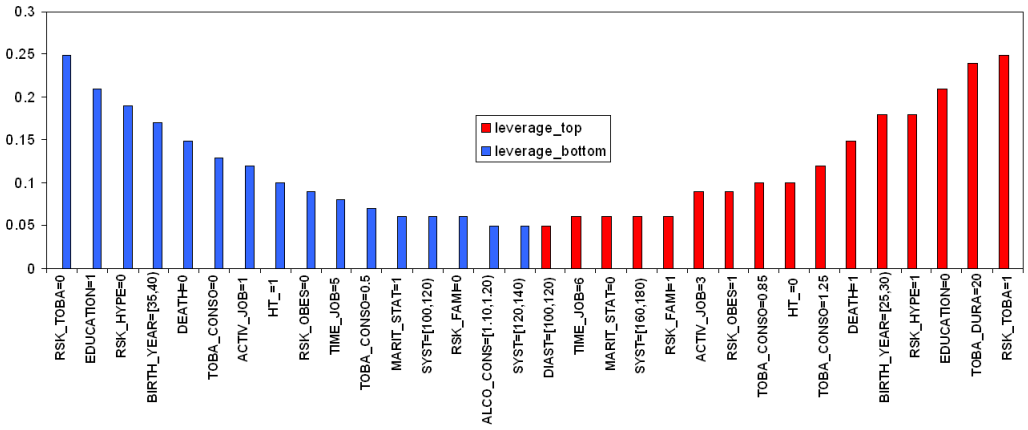

FIG. 3 – *Histogramme de Leverage pour les propriétés simples de l'athérosclérose. Les barres bleues (resp. rouges) représentent les propriétés les plus importantes pour Bottom (resp. Top) du classement.*

	taille prop=1			**taille prop=2**		
	Top5% et Bottom5%	Tous les exemples	Vecteurs Support	Top5% et Bottom5%	Tous les exemples	Vecteurs Support
Atherosclerosis	20	2	2	463	63	63
Australian	15	7	4	135	71	40
Heart	21	11	2	186	78	36
Synthetic	26	2	2	975	180	191

TAB. 3 – *Nombre de propriétés découvertes lorsque l'on considère Top+Bottom versus toute la base d'exemples et versus les vecteurs de support.*

raisons d'une classification donnée à travers le classement des résultats de SVM. L'idée sous-jacente consiste à contraster les résultats de SVM, principalement le Top et Bottom du classement, afin de détecter les propriétés qui différencient les classes, propriétés très utiles pour l'expert du domaine pour diriger des actions et comprendre le système. De plus, le fait d'ignorer le milieu du classement en se concentrant sur les extrémités simplifie l'extraction de propriétés intéressantes. Notons qu'en plus des SVM, notre algorithme peut être utilisé avec n'importe quelle liste classée quelle que soit la méthode de classement utilisée.
En faisant varier la taille des populations dans Top et Bottom, nous permettons un compromis entre généralisation et exactitude (accuracy). En utilisant de petites tailles, nous obtenons des règles claires avec fort Leverage mais qui peuvent connaître un sur-apprentissage (overfitting). Avec de grands tailles, nous obtenons des règles de moins bon Leverage qui généralisent bien mais qui peuvent ne pas offrir suffisamment d'information utiles. Selon les applications, différentes valeurs peuvent être utilisées pour la validation croisée selon les tailles de Top et Bottom, incluant la moyenne du Leverage et l'exactitude des règles.
Il convient de discuter notre approche par rapport aux arbres de décision. Quoique interprétables, les arbres de décision produisent des descriptions discriminantes des classes ; c'est l'une des raisons pour lesquelles ils ne sont pas actionnables selon Gamberger et Lavrač (2002). Ce que nous proposons, ce sont les descriptions longues discriminantes caractéristiques qui permettent de mettre la lumière sur le modèle et non de le reproduire.
Concenant les mesures d'évaluation des régles, nous avons choisi d'utiliser le Leverage comme indice de pertinence pour les raisons évoquées plus haut. Cependant, cette mesure n'est pas monotone et ne permet donc pas d'élaguer l'espace de recherche lorsque l'on combine les propriétés comme dans *Apriori* Agrawal et al. (1993). Nous proposons d'utiliser le Leverage comme mesure de filtrage des bonnes règles et d'explorer l'utilisation d'autres mesures statistiques pour le parcours efficace de l'espace de recherche.
Remerciements : Ce travail a été partiellement financé par un contrat de recherche avec Consolidated Edison New York. Nous remercions Sergey Sigelman pour l'implantation de YSVM. Merci également aux relecteurs pour leurs commentaires et suggestions.

Références

Agrawal, R., T. Imielinski, et A. Swami (1993). Mining association rules between sets of items in large databases. In P. Buneman et S. Jajodia (Eds.), *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*, New York, pp. 207–216. ACM Press.

Barakat, N. et A. P. Bradley (2006). Rule extraction from support vector machines : Measuring the explanation capability using the area under the roc curve. In *ICPR (2)*, pp. 812–815.

Barakat, N. et J. Diederich (2004). Learning-based rule-extraction from support vector machines. In *12th Int'l Conf. Computer Theory and Applications*.

Ben-Hur, A., D. Horn, H. T. Siegelmann, et V. Vapnik (2002). Support vector clustering. *J. Mach. Learn. Res. 2*, 125–137.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition 30*(7), 1145–1159.

da Costa F. Chaves, A., M. M. B. R. Vellasco, et R. Tanscheit (2005). Fuzzy rule extraction from support vector machines. In *Proceedings of the Fifth International Conference on*

Hybrid Intelligent Systems, Washington, DC, USA, pp. 335–340. IEEE Computer Society.

Fu, X., C. Ong, K. S. S., G. G. Hung, et L. Goh (2004). Extracting the knowledge embedded in support vector machines. In *2004 IEEE International Joint Conference on Neural Networks*.

Fung, G., S. Sandilya, et R. B. Rao (2005). Rule extraction from linear support vector machines. In *KDD '05 : Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, New York, NY, USA, pp. 32–40. ACM Press.

Gamberger, D. et N. Lavrač (2002). Generating actionable knowledge by expert-guided subgroup discovery. In *PKDD'02 : Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, pp. 163–174. Springer-Verlag.

Lucas, N., J. Azé, et M. Sebag (2002). Atherosclerosis Risk Identification and Visual Analysis. In *ECML/PKDD 2002 Discovery Challenge Workshop program*.

Martens, D., B. Baesens, T. V. Gestel, et J. Vanthienen (2007). Comprehensible credit scoring models using rule extraction from support vector machines. *European Journal of Operational Research 183, 3*, 1466–1476.

Nunez, H., C. Angulo, et A. Catala (2002). Support vector machines with symbolic interpretation. In *SBRN '02 : Proceedings of the VII Brazilian Symposium on Neural Networks (SBRN'02)*, Washington, DC, USA, pp. 142. IEEE Computer Society.

Piatetsky-Shapiro, G. (1991). Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, pp. 229–248. AAAI/MIT Press.

Turmeaux, T., A. Salleb, C. Vrain, et D. Cassard (2003). Learning characteristic rules relying on quantified paths. In *7th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, pp. 471–482. Springer–Verlag, LNCS.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. New York, NY, USA : Springer-Verlag New York, Inc.

Zhang, Y., H. Su, T. Jia, et J. Chu (2005). Rule extraction from trained support vector machines. In *PAKDD*, pp. 61–70.

Summary

Support Vector Machines (SVMs) have attracted a great deal of attention and achieved huge success mainly as powerful classifiers. However, one of the main drawbacks of SVMs is the lack of intelligibility of the results. SVMs are "black box" systems that do not provide insights on the reasons of a classification or explanations - the results produced must be taken on faith. We are concerned about the problem of intelligibility because from our practical experience, domain experts strongly prefer Machine Learning with explanations. In that context, we have developed a new approach to provide explanations and make SVMs results more actionable. The underlying idea is to produce explanations by applying symbolic Machine Learning models to SVM-produced ranking results. More precisely, we are contrasting SVM results from the top and bottom of rankings to detect the main characteristic properties of the classes which can be useful for the practitioner to direct actions and understand the system. We applied our approach on several datasets. Our empirical results seem promising and show the utility of our methodology with regard to the intelligibility and actionability of an SVM output.
Key words: Support Vector Machines (SVMs); Ranking, Rule Extraction; Actionability.

Appendix 1

Attribute	Type	Description	Attribute	Type	Description
ICO	C	Identification of a patient	HYPLL	C	Medicines in hyperlipidemia
ACTIV_JOB	C	Physical activity in a job. 1 :sits, 2 : stands, 3 :walks, 4 :carries heavy loads,5 : not stated	MOC_SUC	C	Urine sugar
ACTIV_AFT	C	Physical activity after a job. 1 : sits, 2 : moderate activity, 3 :great activity, 4 : not stated	MOC_ALB	C	Urine albumen
TRANSP_JOB	C	Transport to go to work. 1 :on foot, 2 : by bike, 3 :public means of transport, 4 : by car, 9 : not stated	BOLHR	C	Chest pain
TIME_JOB	C	Time to get to work. 5 : half hour, 6 : 1 hour, 7 : 2 hours, 8 : >2 hours, 9 :not stated	CHLST	N	Cholesterol in mg%
BIRTH_YEAR	N	Year of birth	TRIGL	N	Triglycerides in mg%
ENTRY_YEAR	N	Year of entry into the study	SYST	N	Blood pressure systolic
ALCO_CONS	N	Alcohol consumption	DIAST	N	Blood pressure diastolic
TOBA_CONS	N	Tobacco consumption	HEIGHT	N	Height (cm)
TOBA_DURA	N	Smoking duration	WEIGHT	N	Weight (kg)
MARIT_STAT	C	Marital status. 1 :married, 0 : not married	BMI	N	Body Mass Index
EDUCATION	C	Reached education. 1 : university, 0 : not university	TRIC	N	Skin fold triceps (mm)
IM	C	Myocardial infarction	SUBSC	N	Skin fold subscapularis (mm)
ICT	C	Ictus	RSK_FAMI	C	Family risk
HT	C	Hypertension	RSK_OBES	C	Obesity risk
HTL	C	Medicines in HT	RSK_TOBA	C	Smoking risk
DIAB	C	Diabetes	RSK_HYPE	C	Hypertension risk
DIABD	C	Diet in DIAB	RSK_CHOL	C	Cholesterol risk
HYPL	C	Hyperlipidemia	GROUP	C	Normal, Risk, Pathological
DEATH	C	Patient dead or not			

FIG. 4 – *Attributs de la table de l’athérosclérose. ’C’ désigne catégorique et ’N’ numérique.*

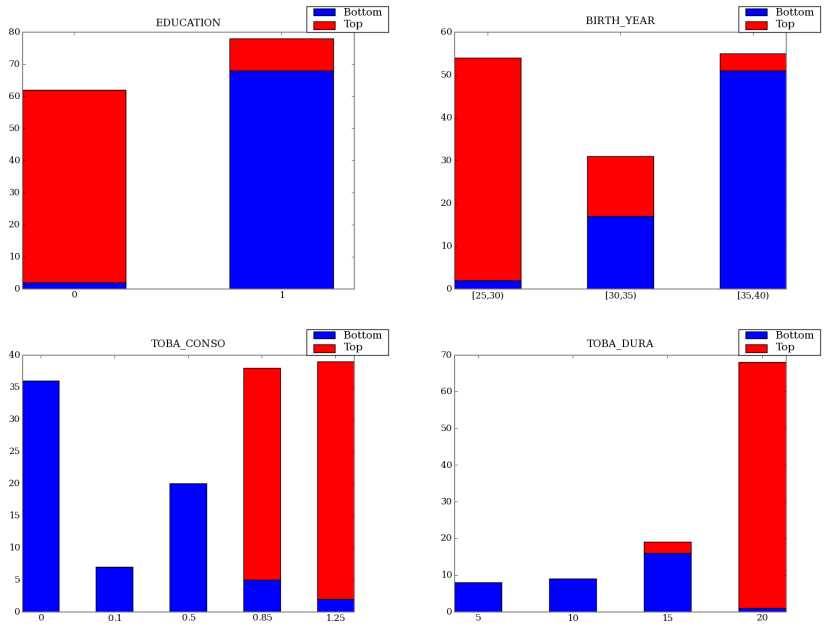

FIG. 5 – *Histogrammes de quelques attributs importants selon YSVM*