TECHNICAL REPORT

No. CCLS-08-02

Title: Toward Actionable Support Vector Machines: A Ranking-based Approach

Authors: Ansaf Salleb-Aouissi, Bert C. Huang, David L. Waltz

Toward Actionable Support Vector Machines: A Ranking-based Approach

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

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

Abstract. During the last decade, 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 this learning method 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 rather than a black box even if the black box system achieves a high predictive performance.
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 discriminative properties between classes which can be quite useful for the practitioner to direct actions and understand the system.
We applied our approach on several datasets. Our empirical results seem very 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.

1 Introduction

One of the main limitations of SVMs is the lack of intelligibility of the results. SVMs indeed do not provide insights on the reasons of a classification or explanations - the results produced must be taken on faith. We propose to make SVMs' results actionable by ranking items, not simply classifying, since it is often the case that resource limitations allow actions on only a very small fraction of possible items, making a classification task highly imbalanced. Basically, ranking can be quite useful in sifting the data in order to keep only what are really the most important examples that represent the classes. We are concerned about the problem of

intelligibility because from our practical experience, decision-makers are understandably much more confident in acting on high-ranked items if they can be given reasons for the rankings than if they are simply given an ordered list of examples. Moreover understanding the bottom of a ranking can also be important.

The general schema of our proposed methodology is as follows:

- Rank examples using Support Vector Machine learning system.
- Create two subsets of the ranking data, consisting of the top n and bottom m items of the ranking. Typically $n = m$, and $||n + m|| = 5 - 20\%$ of the total data.
- Extract the important properties by analyzing attribute patterns in the top and bottom subsets.

Note that since we are ignoring the middle of the ranking and concentrating on the extremes, this may simplify the extraction of interesting patterns. We want to identify the patterned characteristics of the examples from the top of the list in contrast to those from the bottom. Our experiments show that there is hope of success: we can look for the relative frequency of various attribute values. One can compute simply the importance of feature using some statistical measures such as the Leverage measure.

2 Related Work

Quite a lot of recent work (Barakat and Diederich (2004); Barakat and 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)) has addressed the problem of extracting explanations from SVMs.

Nunez et al. (Nunez et al. (2002)) suggested a geometric method to transform an SVM into a rule-based classifier. They use k-means clustering to determine a set of prototype vectors. When combined with the support vectors lying on the margin, these vectors help to construct the boundaries of ellipsoids or hyper-rectangles. These are mapped to if-then equations or interval rules respectively. This approach does not scale well to large datasets and has been experimentally tested only on small benchmarks. Moreover, rules are not necessarily intelligible to the expert of the domain since their main goal is to achieve classification with the SVM-derived classifier performance that is comparable to the SVM itself. Similar work proposed by Zhang et al. (Zhang et al. (2005)) suggests a hyper-rectangle rule extraction (HRE) algorithm for SVMs. The main difference is that support vector clustering (Ben-Hur et al. (2002)) is used to find the vector prototypes of each class rather than k-means. This avoids choosing the number of clusters a priori.

Barakat and Bradley (Barakat and Diederich (2004)) combine decision trees and SVMs to produce explanations. This is achieved as follows: First, train an SVM to constuct a classification model with acceptable accuracy, precision and recall. Second, select the support vectors generated by the model and discard their class labels [1]). The SVM model is then used to predict their class labels which leads to a new data set. Third, train a decision tree on this new dataset to produce symbolic rules. Decision rules are then evaluated on a blind test to check whether unseen examples are classified correctly by the classification rules. The evaluation measures

[1] The reason support vectors are re-classified is not clear in (Barakat and Diederich (2004)

used in (Barakat and Diederich (2004)) – mainly accuracy and fidelity – have been extended to the area under the ROC curve (AUC) in (Barakat and Bradley (2006)). Again, this approach aims to find a set of decision rules with a high predictive ability as compared to SVM rather than producing explanatory knowledge for a domain expert.

Fung et al. (Fung et al. (2005)) propose an approach to convert a linear SVM to a set of non-overlapping rules of the form: $\wedge_{i=1}^{n} l_i \leq x_i < u_i$ equivalent to the linear classifier. This is achieved by solving simple linear programming problems in $2n$ variables, n being the number of features. Each rule represents a hypercube in an n-dimensional space with axis-parallel surfaces. The set of optimal rules is computed either by using a volume-maximization criteria (that maximizes the volume of the hyper-cube) or point-coverage criteria (that maximizes the number of training points in the half-space). In this work, the rules are expressed by disjunction of conjunctions and may not be human-understandable explanations.

A fuzzy rule extraction approach is proposed by da Costa et al. in (da Costa F. Chaves et al. (2005)). Fuzzy rules are extracted in 3 steps: First, support vectors obtained from SVM are projected onto the coordinate axes. Second, for each coordinate, a number of fuzzy sets are constructed. These constitute the antecedents of the rules. Finally, for each support vector a fuzzy rule is extracted. Two evaluation measures are defined in (da Costa F. Chaves et al. (2005)) to assess the quality of the rules generated: fuzzy rule accuracy and fuzzy rule coverage. Note that this method generates as many rules as there are support vectors which can be computationally expensive and may lead to a large number of rules.

Fu et al. (Fu et al. (2004)) address the problem of generating explicit if-then rules from non-linear SVMs. The idea is as follows: Given a support vector for a class, a hyper-rectangle is constructed using the crossing points of the support vector with the SVM boundary. The intervals of the hyper-rectangle lead to an initial rule. Then, this rule is tuned to exclude the out-of-class examples so as to improve its accuracy. Finally, rules are merged to get a more concise rule-set. This approach has the main advantage of highlighting the most important attributes in the extracted rules. Moreover, the extracted rules can follow decisions of SVM classifiers very well. This is achieved by using a *fidelity* criterion that assesses how well a system based on the rule extraction matches the SVM classifier. As pointed out in (Fu et al. (2004)), the main drawback of this method is that it is computationally very expensive, especially when the number of support vectors is high.

The approach we propose here differs fundamentally from the approaches above-cited in the sense that we are not relying on the support vectors to achieve interpretability. We use rather top and bottom examples of an SVM-based ranking which is computationally less expensive than considering a potentially large set of support vectors. We argue that choosing support vectors is not necessarily a good choice. Indeed, the area around the margin is very noisy and the fact that support vectors separate the classes does not mean that they are representative of the classes.

3 Our Framework

Our approach is basically a 2-step process. The first step consists in ranking the examples using SVMs. This step is described in Section 3.1. The second step concerns the extraction of explanations from a ranked list of objects. We describe this step in 3.2 along with the YSVM algorithm that we propose.

3.1 Ranking using SVMs

We address the problem of *supervised ranking* of data. The term "ranking" refers to the process of taking a collection of data and ordering it in a meaningful and useful order. Supervised ranking outputs such order using the features and guided by the label assigned to each object.

More formally, we would like to order a set of examples

$$(x_1, y_1), \ldots, (x_n, y_n)$$

where $x_1, \ldots, x_n$ are vectors of features describing a set of object $o_1, \ldots, o_n$, and each object o_i is given a label $y_i \in \{+1, -1\}$.

Even though we want to output a ranking, our problem inputs are more similar to a classification problem. Therefore, we use a classification method and convert its output into a ranking. Specifically, we rank objects by sorting the decision values of a linear support vector machine (SVM) (Vapnik (1995)). The SVM is:

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

Where the ξ variables are slack variables and the C parameter determines the tradeoff between regularization and penalizing misclassification. The R parameter scales the penalty for the positive class. Since we want to penalize mislabeling of an example by the proportion of the total population of the class, we can set parameter R by the following:

$$R = \frac{\text{number of true negatives}}{\text{number of true positives}} \tag{2}$$

Typically, the SVM produces a classifier that labels examples x with $y = sign(w^T x + b)$, but we do not threshold our outputs so we can sort and rank our examples by how strongly the linear classifier predicts the class of each example.

We use receiver order characteristic (ROC) curves (Bradley (1997)) to analyze our rankings, since they provide a good way of measuring the quality of a ranking when the only ground truth we have is whether or not each data point belongs on the top of the ranking (labeled $+1$) or belongs on the bottom (labeled -1). ROC essentially normalizes by the class cardinality which is similar to the normalization of the loss function we did to train the SVM. The quality of an ROC curve is easily measured by the area under the curve (AUC), which is in the range $[0, 1]$, where an AUC of 0.5 can be achieved by a random ordering of the data and AUC of 1.0 is achieved by perfectly ranking the positive examples at the top and the negative ones at the bottom.

Example Let's consider a list of 100 electronic components. Each component is described by its serial number, age, size and manufacturer, and is labeled according to its failure status (label=1 for failed, -1 otherwise). An SVM ranking would allow us to order the components according to likelihood of failure. The Top of the ranking would have the components that are the most susceptible to failure while components at the Bottom ranking are less prone to failure. Thus, the domain expert can focus on the n top components and act on them, e.g. by scheduling inspections/replacements.

3.2 Getting explanations

Given a a good ranking list produced by the previous step, getting explanations consists in selecting the Top and Bottom examples of the ranking list. By doing so, we group the examples into three sets, where the most 'pure' examples, i.e. the 'Very positive' and 'Very negative' examples are on the Top and Bottom of the ranked list respectively and the examples in the middle, around the SVM margin, are rather noisy. One question that arises is why we compare the Top and the Bottom of a ranking rather than comparing the examples in positive class to the negative class. The reason we are not simply contrasting the positive class to the negative class is that some examples are considered as negative while they are actually unknown. This occurs in many real-life applications. In the previous example, we are not sure whether the negative examples are truly negative; Although these components are considered as negative because they have not failed yet, they may fail anytime soon. The SVM should not classify these examples deep in the negative class but rather around the margin separating the two classes. The approach we suggest here would discard such examples and focus only on the most valuable examples, the ones we are most certain are either good or bad..

Once we focus on the ranking extremities, we look for the set of interesting rules of the form:

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

where $Property$ is an attribute-value pair and $Concept$ is either the concept 'being on the Top' or 'being on the Bottom'. We assess the importance of properties by using the *Leverage* measure (Piatetsky-Shapiro (1991)). Basically, the reason we chose this statistical measure is that it combines high discriminative power with the capture of the most highly associated properties (high support). Leverage measure has been used in other learning tasks and is called also *Novelty* (See for e.g. (Turmeaux et al. (2003)) in learning characteristic rules). The Leverage of the rule above is given by:

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

The Leverage measure evaluates the proportion of additional examples covered by both the left-hand side and right-hand side of the rule above those expected if both sides of the rule were independent of each other. Obviously, we have:

$$-0.25 \leq Leverage(R) \leq +0.25$$

We say that a property is interesting for a given concept if it has a strongly positive Leverage. This indicates a strong association between the property and the concept, while a strongly negative value indicates a strong association between the property and the negation of the concept. In our framework, the Leverage of a rule can be estimated by:

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

where Concept is either T (the Top set) or B (the Bottom set), p is a property and the notation $\mathcal{V}_p(x)$ is a boolean function such that for an example x, we have $\mathcal{V}_p(x) = true$ or $false$ which means that the property p may be satisfied by x or not.

Example Let's consider again the ranked list of electronic components illustrated in Table 1. Identifying the main properties as shown in Table 2 helps identify what is responsible for failures. For example, it can be extremely important to find patterns in the attributes of ranked items, for instance to realize that particular components from a particular manufacturer are disproportionately responsible for failures. The ultimate goal is to help the domain expert in setting policies for purchasing the most reliable components, scheduling inspections, etc.

	Rank	Serial#	Age	Size	Manufacturer
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 – *A toy example.*

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

TAB. 2 – *Set of properties extracted for the toy example. Manufacturer A is rather good especially for large-size components, while manufacturer B has rather bad small components.*

The aim of YSVM (given in Algorithm 1) is to uncover the set of the most important properties that lead an SVM classification to rank some examples above some others. The algorithm explores the search space of possible properties by contrasting Top and Bottom parts of the ranking. The notations $\mathcal{P}_T$ and $\mathcal{P}_B$ are used to denote the set of properties (left-hand sides of the rules) for Top and Bottom respectively. For a better visualization of the proportions of the properties in Top and Bottom, the algorithm outputs also a histogram for each attribute giving the relative frequency of its various values. This allows us to clearly visualize the most contrasting single properties. We have extended YSVM to handle conjunctions of properties. We have also extended our algorithm to try several values of Top and Bottom percentages in order to select the best sizes for Top and Bottom, the sizes that lead to the highest number of interesting properties and the highest average Leverage.

Algorithm 1: YSVM algorithm

Input:
- a ranked list of examples $\mathcal{L}$
- a Leverage threshold MinLev
- Top and Bottom percentages

Output:
- 2 sets of properties $\mathcal{P}_T$ and $\mathcal{P}_B$
- a set of histograms $\mathcal{H}$ one for each attribute

$T \leftarrow \{examples\ in\ Top\ of\ \mathcal{L}\}$
$B \leftarrow \{examples\ in\ Bottom\ of\ \mathcal{L}\}$
$\mathcal{P}_T \leftarrow \emptyset$
$\mathcal{P}_B \leftarrow \emptyset$
foreach *attribute* **do**
  foreach *value* **do**
    $p \leftarrow (attribute = value)$
    if $Leverage(p \rightarrow Top) \geq MinLev$ **then**
      $\mathcal{P}_T \leftarrow \mathcal{P}_T \cup p$
    else if $Leverage(p \rightarrow Bottom) \geq MinLev$ **then**
      $\mathcal{P}_B \leftarrow \mathcal{P}_B \cup p$
  $h = Histogram(attribute)$
  $\mathcal{H} = \mathcal{H} \cup h$
return $\mathcal{P}_T, \mathcal{P}_B, \mathcal{H}$

4 Experiments

We have implemented YSVM[2] in Python and have conducted an experimental evaluation of our algorithm on several benchmarks. We have used first SVMLight[3] to train Support Vector Machines on the different datasets in order to get the ranking lists.

4.1 A synthetic dataset

We did a sanity check to verify whether YSVM is catching the right attributes. For this purpose, we have generated randomly a synthetic dataset of 1000 examples each described by 50 features such that $X \in \{-1, 1\}^{50}$. Class labels are assigned as follows:

$$Y = sign(\sum_{k=1}^{k=11} X_k)$$

In other words, the class label is a linear combination of the first 11 attributes. YSVM succeeded in finding these attributes by focusing only on the Top 5% and bottom 5% of the ranking list and with a minimum Leverage of 0.08. It was not possible to uncover this set of properties

[2]Source code will be available at `http://www.ccls.columbia.edu/ansaf`
[3]`http://svmlight.joachims.org/`

by using the full dataset with the same parameters until we decreased the Leverage to a very low value. This means that we have more discriminative power in Top+Bottom than in the full dataset.

4.2 Atherosclerosis dataset

In this section, we describe the experiments we have conducted on a medical dataset in the context of the Stulong project[4]. The dataset concerns a twenty years lasting study of the risk factors of the atherosclerosis in a population of 1 419 middle aged men. We use a compiled dataset (Lucas et al. (2002)) with the goal to identify the main factors for this disease. The attributes used in the dataset are given in Table 4 in the appendix. The patients have been classified into three groups, namely the normal group, the risk group and the pathological group. While this attribute has not been used in learning the ranking, the SVM succeed in ranking the pathological and risky patient above the healthy ones. The ranking target attribute used is "death". Figure 1 shows the ROC curve for the learning results with Top 10% and Bottom 10% emphasized and where Top gathers the illest patients and Bottom the healthiest patients. We have used YSVM with different values of Top and Bottom and kept the couple (mainly Top=5% and Bottom=5%) giving the highest number of good properties. The results are reported in Figure 2 and the associated histograms are given in Figure 4. Our experiments with different Top and bottom values have shown that the more we increase ||T+B||, the less we get interesting properties (for space limitation, the details are not reported here).

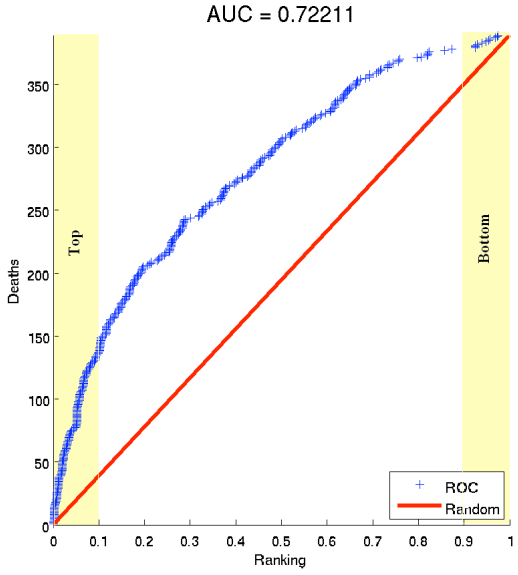

FIG. 1 – *Atherosclerosis ROC Curve. The X-axis represents the ranking in proportions.*

[4] http://euromise.vse.cz/STULONG

Property	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
MARIT_STAT=0	21	0.06	3	-0.06
MARIT_STAT=1	49	-0.06	67	0.06
EDUCATION=0	60	0.21	2	-0.21
EDUCATION=1	10	-0.21	68	0.21
HT_=0	70	0.10	41	-0.10
HT_=1	0	-0.10	29	0.10
SYST=[100,120)	2	-0.06	18	0.06
SYST=[120,140)	12	-0.05	26	0.05
SYST=[160,180)	18	0.06	0	-0.06
DIAST=[100,120)	14	0.05	0	-0.05
RSK_FAMI=0	50	-0.06	67	0.06
RSK_FAMI=1	19	0.06	3	-0.06
RSK_OBES=0	41	-0.09	66	0.09
RSK_OBES=1	28	0.09	4	-0.09
RSK_TOBA=0	0	-0.25	70	0.25
RSK_TOBA=1	70	0.25	0	-0.25
RSK_HYPE=0	15	-0.19	69	0.19
RSK_HYPE=1	50	0.18	1	-0.17
TOBA_DURA=20 AND EDUCATION=0	57	0.20	0	-0.20
ICT_=0 AND EDUCATION=0	60	0.21	2	-0.21
MARIT_STAT=1 AND RSK_HYPE=0	5	-0.22	66	0.22
TOBA_DURA=20 AND RSK_TOBA=1 AND EDUCATION=0	57	0.20	0	-0.20
MARIT_STAT=1 AND RSK_HYPE=0 AND RSK_OBES=0	3	-0.21	62	0.21
...	...	...	...	...

FIG. 2 – *List of some of the important properties discovered in atherosclerosis dataset as generated by YSVM. A property is characteristic of a concept (Top/Bottom) if its Leverage is strongly positive.*

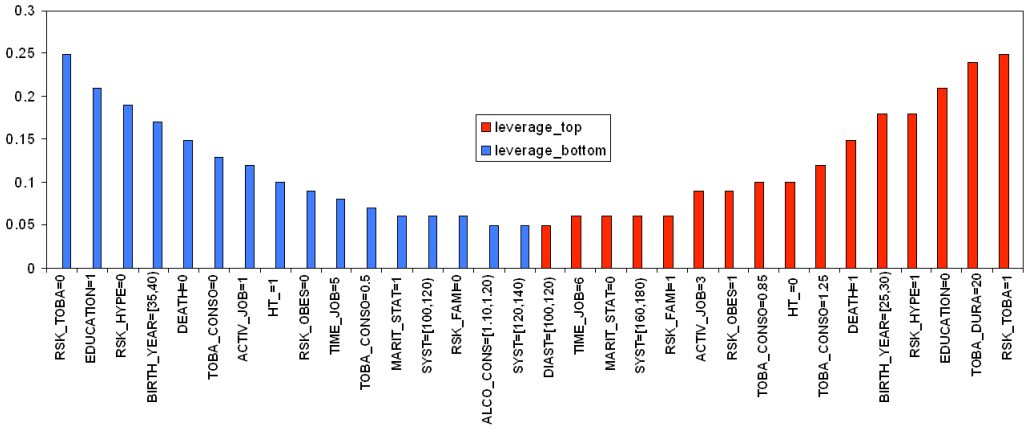

FIG. 3 – *Histogram of Leverage for atherosclerosis properties. Only single properties are represented here. The blue (resp. red) bars represent the Leverage of the most important properties for Bottom (resp. Top) ranking.*

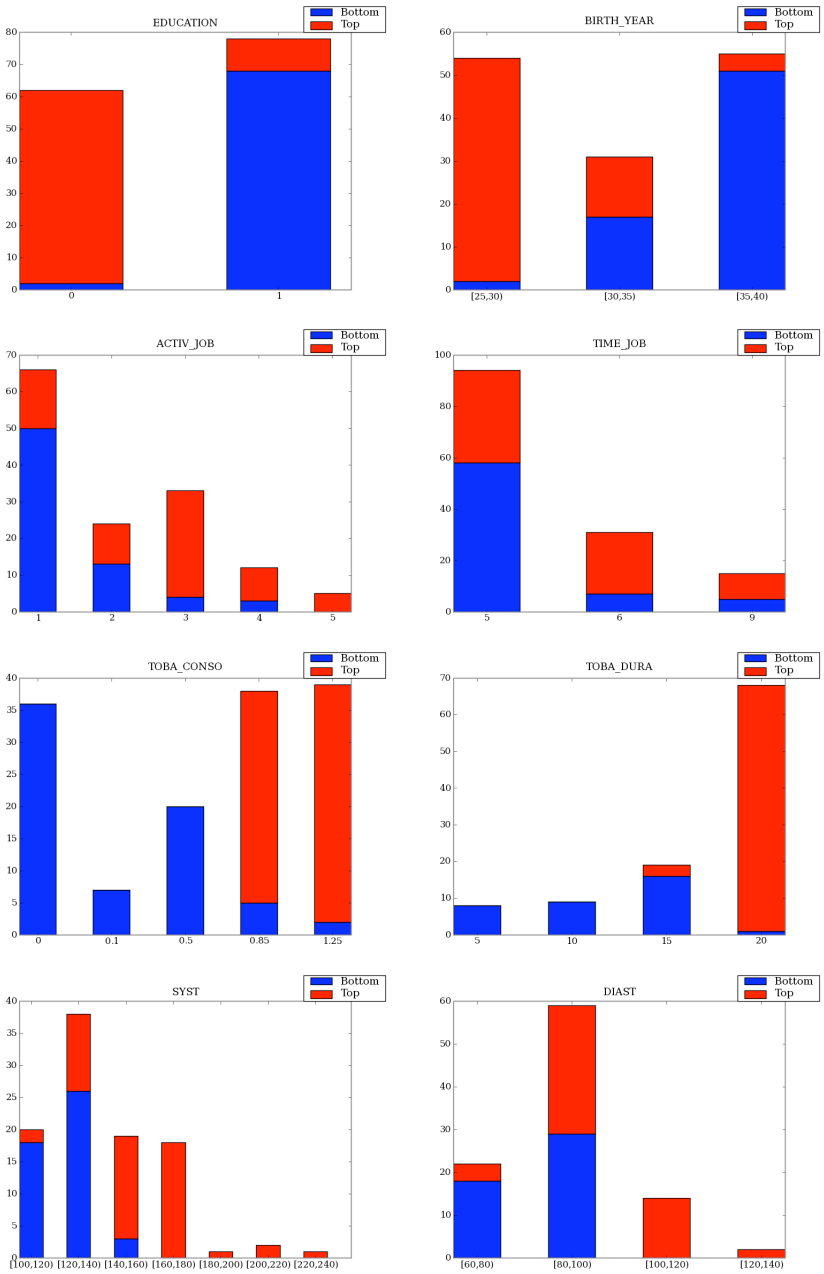

FIG. 4 – *Histograms of some attributes as extracted by YSVM*

Stulong data has been already used in a discovery challenge and has been subject to many publications (See for instance Lucas et al. (2002)). Atherosclerosis factors are known to be mainly tobacco consumption and duration, overweight, physical activity while there is no evidence on the impact of alcohol consumption on the atherosclerosis. All these factors have been uncovered by our approach as shown in Figure 3 and Figure 4. Table 3 compares the number of interesting properties when we use the full dataset versus considering only the Top and Bottom parts of the ranked list of examples. For this experiment, we have used MinLeverage=0.08, Top=Bottom=5% and size of properties ≤ 2 on other UCI datasets.

	Top+Bottom	Full dataset
Atherosclerosis	463	63
Australian	135	67
German	134	32
Heart	280	268

TAB. 3 – *Number of properties discovered when we consider the full datasets vs. Top+Bottom.*

5 Conclusion and Future Work

This paper describes a simple yet useful approach to make Support Vector Machines results actionable. The main originality of our approach is its ability to get insights on the reasons of a given classification through ranking the SVMs results. The underlying idea is to contrast SVMs results, mainly Top and Bottom ranking, to detect the main discriminative properties between classes which can be quite useful for the practitioner to direct actions and understand the system. Moreover, since we are ignoring the middle of the ranking and concentrating on the extremes, this may simplify computationally the task of patterns extraction. Notice here that our algorithm would work with any ranking method as it needs as input a ranked list of objects, no matter what learning methodology has been used to rank the objects.

By varying the size of the top and bottom populations, we allow for a tradeoff parameter between generalization and accuracy. By using smaller sizes, we obtain cleaner rules with higher confidence, but could experience overfitting. With larger sizes, we obtain less confident rules that generalize better but may not offer enough useful information. Depending on the application, there are various values that can be used for cross validation with respect to the size of the top and bottom, including average leverage or accuracy of the rules extracted.

In future work, we would like to integrate the readability/actionability ability into the learning algorithm itself. We plan to investigate other methods to tackle the problem of generating stronger and interesting action rules as well as investigate the use of our tool in feature selection problems.

Acknowledgments

This work has been partly supported by a research contract from Consolidated Edison New York. Thanks to Sergey Sigelman for YSVM implementation.

References

Barakat, N. and 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. and 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, and 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, and 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, and L. Goh (2004). Extracting the knowledge embedded in support vector machines. In *2004 IEEE International Joint Conference on Neural Networks*.

Fung, G., S. Sandilya, and 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.

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

Nunez, H., C. Angulo, and 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, and 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, and J. Chu (2005). Rule extraction from trained support vector machines. In *PAKDD*, pp. 61–70.

Appendix 1

Résumé

Durant la dernière décade, les machines a vecteurs support (ou Séparateur à Vaste Marge: SVM) ont connu un immense succès. Ce succès est du a leurs fondements théoriques solides et a leur pouvoir prédictif sans précédent. Cependant, l'une des principales critiques faites aux

SVMs est le manque d'intelligibilité des résultats. 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 soit-elle. Dans cette thématique, nous proposons une nouvelle approche qui consiste à contraster les résultats des SVMs afin de détecter les principales propriétés discriminantes. Ce but est atteint en couplant des modèles d'ordonnancement des résultats des SVMs à des méthodes d'apprentissage de concepts. Nous présentons une application de notre méthode sur des données médicales concernant des patients de l'athérosclérose. Les tests expérimentaux montrent l'intérêt de notre approche quant à l'intelligibilité et l'actionabilite des résultats.

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

TAB. 4 – *Attributes of the atherosclerosis table. Note that* Group*, is not used in the learning.*