DC Optimization in Adversarial Sparse Support Vector Machine

Annabella Astorino[1], Massimo Di Francesco[2], Manlio Gaudioso[1], Enrico Gorgone[2], and Benedetto Manca[2]

[1] Dipartimento di Ingegneria Informatica, Modellistica, Elettronica e Sistemistica, Universitá della Calabria, 87036 Rende, Italy
annabella.astorino@dimes.unical.it - manlio.gaudioso@unical.it
[2] Dipartimento di Matematica e Informatica, Universitá degli Studi di Cagliari, 09124 Cagliari, Italy
mdifrance@unica.it - egorgone@unica.it - bmanca@unica.it

Abstract. In supervised classification models, such as Support Vector Machine, the main purpose is to predict the class membership of the incoming samples. In some real applications malicious inputs are inserted to mislead a vulnerable classifier, leading to a wrong prediction. In our work we focus first on the problem of introducing the smallest perturbation of a sample to induce incorrect classification and then on how to produce a significant downgrading of the classifier acting on a subset of the input samples.
The novelty of the proposed approach is in the attempt of calculating *sparse* perturbations by minimizing the relative ℓ_0-pseudo-norm, which gives rise to a Difference of Convex (DC) optimization model. We present the results of some preliminary experiments.

Keywords: Adversarial Machine Learning · SVM · Sparse optimization · ℓ_0-pseudo-norm · DC Optimization.

1 Introduction

In the classification framework of Adversarial Machine Learning (AML) there are two players in action: the Defender and the Attacker. The Defender tries to design a robust classifier, while the Attacker tries to introduce small modifications in data to mislead the classifier and cheating the Defender (see Fig. 1). Applications range from computer vision to cybersecurity where input perturbations could poison the predictions of learning-based pattern classifiers [2].

In this work, we consider the Attacker's perspective and, in detail, the following steps should be performed [12]:

- to enable different attack scenarios against learning algorithms;
- to implement the corresponding attack strategy;
- to define the attacker's objective;

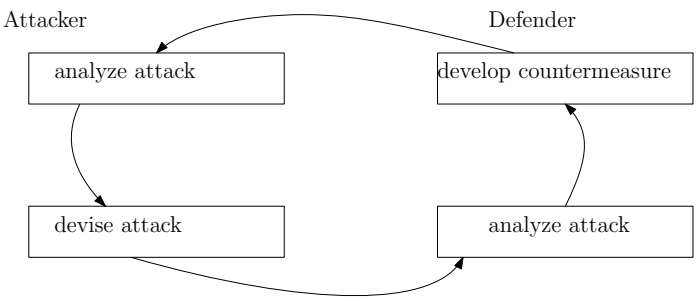


Fig. 1. Adversarial ML framework

– to identify the data to be manipulated;
– to formulate an optimization problem.

In the framework of supervised binary classification, we suppose that the Defender constructs a Support Vector Machine (SVM) classifier (see [10, 13, 23, 24]) by stating the following problem:

$$\begin{aligned} \min_{w,\gamma,\xi,\zeta} \ & \tfrac{1}{2} w^t w + C\Big(\sum_{i\in I} \xi_i + \sum_{l\in L} \zeta_l\Big) \\ s.t. \quad & a_i^t w - \gamma + 1 \le \xi_i \quad i \in I \\ & -b_l^t w + \gamma + 1 \le \zeta_l, \quad l \in L \\ & \xi_i \ge 0, \quad i \in I, \quad \zeta_l \ge 0, \quad l \in L \end{aligned}$$

where the two point-sets in the n-dimensional space, $\mathcal{A} = \{a_i\}$ $(i \in I)$ and $\mathcal{B} = \{b_l\}$ $(l \in L)$, are given. A separating hyperplane, defined by the couple $(w \in \mathbb{R}^n, \gamma \in \mathbb{R})$ is calculated in view of minimizing the sum of the classification errors $\xi_i, i \in I$ and $\zeta_l, l \in L$ for the two sets $\mathcal{A}$ and $\mathcal{B}$, respectively.

In the paper we propose two models. In both we assume that the Attacker has complete information on the dataset; in addition, he holds the control on one or more samples and, consequently, he has the possibility of perturbing them. In the first model (Section 2) the objective is to *move* some points of the sets so that they are wrongly classified by the available hyperplane; in the second one the objective is to generate the maximum disruption of the classifier acting on the samples that can be manipulated. The novelty of both models is the search of *sparse* perturbations, so that the smallest possible number of features is involved. To deal with sparsity, we resort to the k-norm based approach described in [17, 15, 18]. It results, in both models, in a DC programming formulation [3, 16, 19, 20, 1, 22].

2 Sparse sample perturbation

In the adversarial classification framework, malicious inputs are designed to cheat a vulnerable classifier leading to a wrong prediction. In particular, we focus on

the search of the smallest (in the ℓ_0-pseudo-norm sense) perturbations of samples producing a failure in the classification process.

Given a binary classifier [6, 7, 4, 8, 9, 5, 11] $F : \mathbb{R}^n \mapsto \{-1, 1\}$ and any $x \in \mathbb{R}^n$ we find the *smallest* perturbation $\delta \in \mathbb{R}^n$ such that

$$F(x + \delta) \neq F(x),$$

by changing a small number of attributes.

More specifically, adopting the SVM approach to separate two data sets $\mathcal{A}$ and $\mathcal{B}$, we take w.l.o.g. any $a \in \mathcal{A}$ and consider the separating hyperplane $H(w, \gamma)$. If a is correctly classified, the inequality $a^T w \leq \gamma - 1$ holds. Hence, we look for a perturbation $\delta \in \mathbb{R}^n$ to make the point a misclassified. To this aim, satisfaction of the following inequality $(a + \delta)^T w \geq \gamma + 1$ is required. It can be rewritten as $w^T \delta \geq \rho_a = -a^T w + \gamma + 1 > 0$. In order to keep small the number of attributes to be modified we add a cardinality constraint [21]. To bound the magnitude of all components of δ we introduce the minimization of its ℓ_∞ norm, coming out with the non-linear program $\min_\delta \{\|\delta\|_\infty : w^t \delta \geq \rho_a, \|\delta\|_0 \leq k\}$ for an appropriately selected value of $k \in \{1, \ldots, n\}$.

To replace the cardinality constraint $\|\delta\|_0 \leq k$ with a DC constraint we use the k-norm definition. We recall that the k-norm of x, indicated by $\|x\|_{[k]}$, is the sum of k maximal components (in modulus) of x, $k = 1, \ldots, n$. The following hold:

i) $\|x\|_\infty = \|x\|_{[1]} \leq \ldots \leq \|x\|_{[k]} \leq \ldots \|x\|_{[n]} = \|x\|_1$;
ii) $\|x\|_0 \leq k \;\; \Rightarrow \;\; \|x\|_1 - \|x\|_{[s]} = 0, \;\; k \leq s \leq n.$

In particular, for any $k = 1, \ldots, n$, the following equivalence holds:

$$\|x\|_0 \leq k \;\; \Leftrightarrow \;\; \|x\|_1 - \|x\|_{[k]} = 0,$$

which allows us to rewrite the problem in the form

$$\begin{aligned} &\min_\delta \|\delta\|_\infty \\ &s.t. \;\; w^t \delta \geq \rho_a \\ &\qquad \|\delta\|_1 - \|\delta\|_{[k]} \leq 0 \end{aligned}$$

By introducing the penalty parameter σ for the nonconvex, nonsmooth constraint $\|\delta\|_1 - \|\delta\|_{[k]} \leq 0$, we obtain

$$\begin{aligned} &\min_\delta \|\delta\|_\infty + \sigma(\|\delta\|_1 - \|\delta\|_{[k]}) \\ &s.t. \;\; w^t \delta \geq \rho_a \end{aligned} \tag{1}$$

Note that the objective function of the above problem is of DC type, thus we apply the Descent-Ascent DC (DADC) algorithm [14] after having penalized the linear constraint $w^T \delta \geq \rho_a$ by the scalar $\beta > 0$.

In figures 2 and 3 we report two examples in the 2-dimensional space. In Fig. 2 the sets $\mathcal{A}$ and $\mathcal{B}$ are linearly separable. We have used the following parameters: $k = 1$, $\sigma = 4$ and $\beta = 1.1$. The algorithm has provided the following solution:

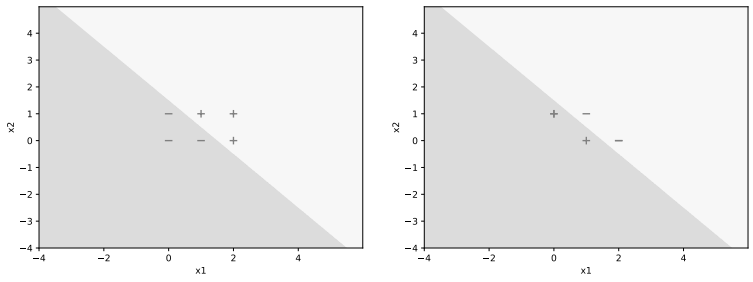
a_i	$a_i + \delta_i$	$b_i l$	$b_l + \delta_l$
$(0,0)$	$(2,0)$	$(1,1)$	$(0,1)$
$(0,1)$	$(1,1)$	$(2,0)$	$(1,0)$
$(1,0)$	$(2,0)$	$(2,1)$	$(0,1)$

Note that, having set $k = 1$, for all points considered only one attribute changes and, as expected, the classifier fails for all of them.

In Fig. 3 the sets $\mathcal{A}$ and $\mathcal{B}$ are not linearly separable. Even in this case, by using the same parameters, it is confirmed that only one attribute changes and the classifier fails for each sample. We have the following solution:

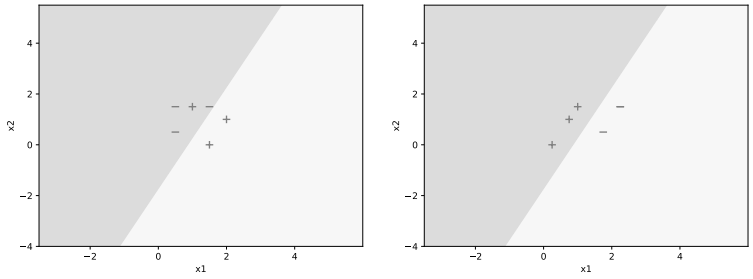
a_i	$a_i + \delta_i$	$b_i l$	$b_l + \delta_l$
$(0.5, 0.5)$	$(1.75, 0.5)$	$(1.5, 0)$	$(0.25, 0)$
$(0.5, 1.5)$	$(2.25, 1.5)$	$(2, 1)$	$(0.75, 1)$
$(1.5, 1.5)$	$(2.25, 1.5)$	$(1, 1.5)$	$(1, 1.5)$

The point $b_3 = (1, 1.5)$ is unchanged because it is already misclassified and we have not applied the method on it.



(a) (b)

Fig. 2. Example 1: SVM before (a) and after perturbation (b).



(a) (b)

Fig. 3. Example 2: SVM before (a) and after perturbation (b).

3 Classifier perturbation

For the two point-sets $\mathcal{A} = \{a_i\}$, $i \in I$ and $\mathcal{B} = \{b_l\}$, $l \in L$ in $\mathbb{R}^n$, we suppose that the Defender has found an optimal separating hyperplane (w', γ') such that

$$(w', \gamma') = \arg\min_{w,\gamma} \sum_{i \in I} \max\{0, a_i^\top w - \gamma + 1\} + \sum_{l \in L} \max\{0, -b_l^\top w + \gamma + 1\} \quad (2)$$

We suppose that the Attacker is able to manipulate the attributes of some samples, both from $\mathcal{A}$ and $\mathcal{B}$, corresponding to the index sets $I_M \subset I$ and $L_M \subset L$, respectively; variables δ_i, $i \in I_M$ and δ_l, $l \in L_M$ define the sample perturbations. The objective of maximising the change in the separating hyperplane is pursued by plugging into the objective function the modulus of the scalar product between the normal w to the perturbed hyperplane and w', the normal to the current one. Thus, we state the problem

$$\begin{aligned} \min_{w,\gamma,\delta} \; & C_1\Big(\sum_{i \in I \setminus I_M} \max\{0, a_i^\top w - \gamma + 1\} + \sum_{l \in L \setminus L_M} \max\{0, -b_l^\top w + \gamma + 1\} \\ & + \sum_{i \in I_M} \max\{0, (a_i + \delta_i)^\top w - \gamma + 1\} \\ & + \sum_{l \in L_M} \max\{0, -(b_l + \delta_l)^\top w + \gamma + 1\}\Big) + C_2 |w^\top w'| + \frac{1}{2} C_3 \|w\|^2 \\ s.t. \quad & \|\delta_i\|^2 \le \rho, \quad i \in I_M, \quad \|\delta_l\|^2 \le \rho, \quad l \in L_M \end{aligned} \quad (3)$$

Remark 1 *Note that the objective function encompasses nonsmooth and nonlinear terms of the type $\delta_i^\top w$ under the max operator. They can be put in DC form by observing that any function $f(x, y) = \max\{0, x^\top y\}$ can be rewritten as*

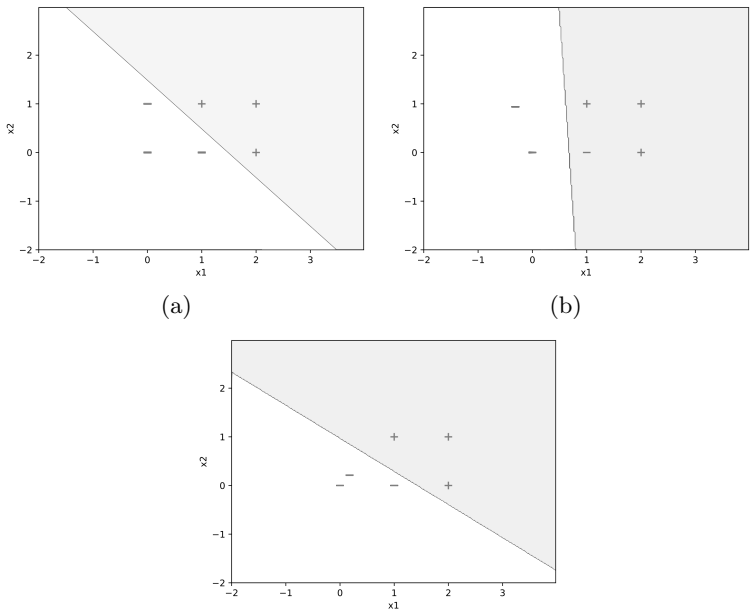
$$f(x,y) = \max\{0, \frac{1}{4}(\|x+y\|^2 - \|x-y\|^2)\} = \frac{1}{4}\max\{\|x+y\|^2, \|x-y\|^2\} - \|x-y\|^2$$

The above Remark allows us to rewrite the objective function in a DC form. In solving the problem we have again used the DADC algorithm [14], after having penalized the constraints $\|\delta_i\|^2 \le \rho$ ($i \in I_m$) and $\|\delta_l\|^2 \le \rho$ ($l \in L_m$) by the scalar $\beta > 0$.

In figures 4, 5, 6 and 7 we report four examples in the 2-dimensional space. In Fig. 4 the sets $\mathcal{A}$ and $\mathcal{B}$ are linearly separable. We have manipulated only the point $(0, 1)$ and we have used the following parameters: $C_2 = 20$, $C_3 = 0.1$, $\beta = 60$ and $\rho = 0.1$. As for the parameter C_1, in Fig. 4-(b) we set it equal to 80 obtaining a considerable perturbation of the separating hyperplane, but paying it with a misclassified point. On the contrary (see in Fig. 4-(c)) by setting $C_1 = 120$ we obtain correct classification of all samples at the cost of a weaker perturbation of the Defender's classifier.

Similar considerations hold for the other examples. In Fig. 5 the manipulated point is $(1, 0)$ and we have used the following parameters: $C_1 = 32$, $C_2 = 30$, $C_3 = 0.1$, $\beta = 40$ and $\rho = 0.1$. In Fig. 6 we have used the following parameters: $C_3 = 0.1$, $\beta = 40$ and $\rho = 0.1$ and the manipulated sample is $(0, 1)$. In Fig. 6-(b)

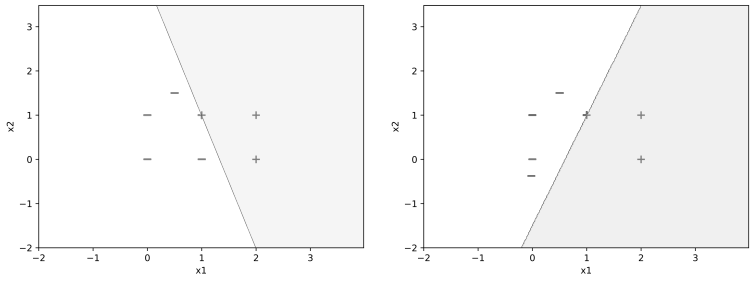
we have set $C_1 = 150$ and $C_2 = 30$ and in Fig. 6-(c) we have set $C_1 = 70$ and $C_2 = 60$. In Fig. 5 we have two manipulated points: $(1, 1)$ and $(0, 2)$ and we have used the following parameters: $C_1 = 10$, $C_2 = 1$, $C_3 = 0.1$, $\beta = 10$ and $\rho = 0.1$.



(a) (b)

(c)

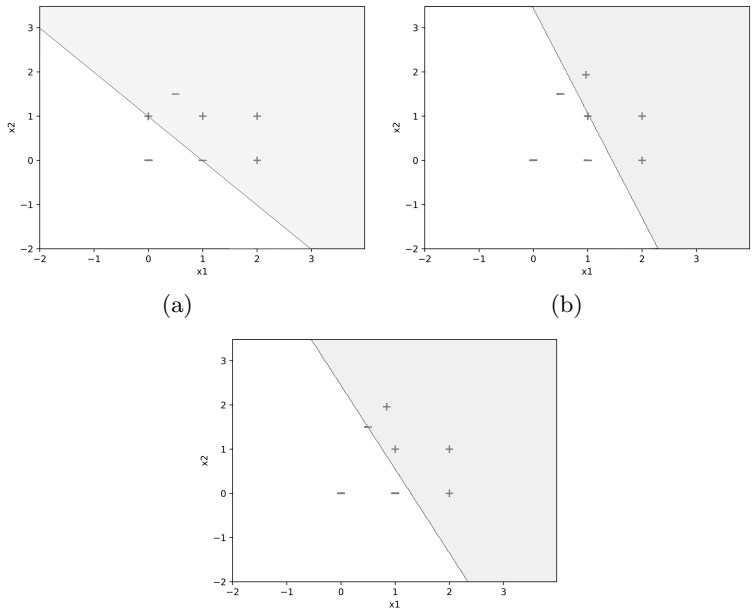
Fig. 4. Example 3: SVM before (a) and after manipulation (b) – (c).



(a) (b)

Fig. 5. Example 4: SVM before (a) and after manipulation (b).

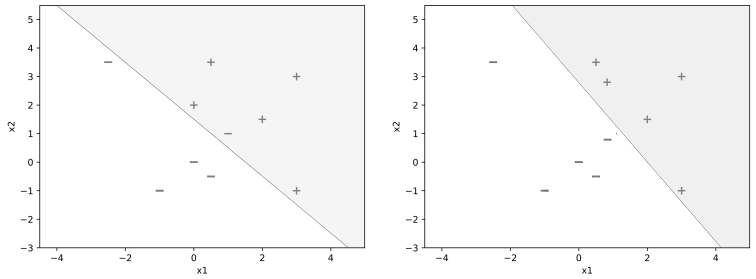
In conclusion, the preliminary numerical experiments show that the optimization program (1) is able to perform a sparse perturbation on the data points such that the SVM classifier fails. Moreover, the classifier perturbation model (3) disrupts the separating hyper-plane obtained via SVM acting on some of the data points, which results in a lower accuracy score on the initial dataset.



(a) (b)

(c)

Fig. 6. Example 5: SVM before (a) and after manipulation (b) – (c).



(a) (b)

Fig. 7. Example 6: SVM before (a) and after manipulation (b).

References

1. An, L.T.H., Tao, P.D.: The dc (difference of convex functions) programming and dca revisited with dc models of real world nonconvex optimization problems. Annals of operations research **133**, 23–46 (2005)
2. Astorino, A., Chiarello, A., Gaudioso, M., Piccolo, A.: Malicious url detection via spherical classification. Neural Computing and Applications **28**, 699–705 (2017)
3. Astorino, A., Di Francesco, M., Gaudioso, M., Gorgone, E., Manca, B.: Polyhedral separation via difference of convex (dc) programming. Soft Computing **25**, 12605–12613 (2021)
4. Astorino, A., Frangioni, A., Gorgone, E., Manca, B.: Ellipsoidal classification via semidefinite programming. Operations Research Letters **51**(2), 197–203 (2023)

5. Astorino, A., Fuduli, A.: Support vector machine polyhedral separability in semisupervised learning. Journal of Optimization Theory and Applications **164**, 1039–1050 (2015)
6. Astorino, A., Gaudioso, M.: Polyhedral separability through successive lp. Journal of Optimization theory and applications **112**(2), 265–293 (2002)
7. Astorino, A., Gaudioso, M.: Ellipsoidal separation for classification problems. Optimization Methods and Software **20**(2-3), 267–276 (2005)
8. Astorino, A., Gaudioso, M.: A fixed-center spherical separation algorithm with kernel transformations for classification problems. Computational Management Science **6**(3), 357–372 (2009)
9. Astorino, A., Gaudioso, M., Seeger, A.: Conic separation of finite sets i. the homogeneous case. Journal of Convex Analysis (2014)
10. Astorino, A., Gorgone, E., Gaudioso, M., Pallaschke, D.: Data preprocessing in semi-supervised svm classification. Optimization **60**(1-2), 143–151 (2011)
11. Bennett, K.P., Mangasarian, O.L.: Robust linear programming discrimination of two linearly inseparable sets. Optimization methods and software **1**(1), 23–34 (1992)
12. Biggio, B., Roli, F.: Wild patterns: Ten years after the rise of adversarial machine learning. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 2154–2156 (2018)
13. Cristianini, N., Shawe-Taylor, J.: An introduction to support vector machines and other kernel-based learning methods. Cambridge university press (2000)
14. d'Alessandro, P., Gaudioso, M., Giallombardo, G., Giovanna, M.: The descent-ascent algorithm for dc programming. INFORMS Journal of Computing, to appear (2023)
15. Gaudioso, M., Giallombardo, G., Miglionico, G.: Sparse optimization via vector k-norm and dc programming with an application to feature selection for support vector machines. Computational Optimization and Applications **86**(2), 745–766 (2023)
16. Gaudioso, M., Giallombardo, G., Miglionico, G., Bagirov, A.M.: Minimizing nonsmooth dc functions via successive dc piecewise-affine approximations. Journal of Global Optimization **71**, 37–55 (2018)
17. Gaudioso, M., Gorgone, E., Hiriart-Urruty, J.B.: Feature selection in svm via polyhedral k-norm. Optimization letters **14**(1), 19–36 (2020)
18. Gotoh, J.y., Takeda, A., Tono, K.: Dc formulations and algorithms for sparse optimization problems. Mathematical Programming **169**, 141–176 (2018)
19. Joki, K., Bagirov, A.M., Karmitsa, N., Mäkelä, M.M.: A proximal bundle method for nonsmooth dc optimization utilizing nonconvex cutting planes. Journal of Global Optimization **68**(3), 501–535 (2017)
20. Khalaf, W., Astorino, A., D'Alessandro, P., Gaudioso, M.: A dc optimization-based clustering technique for edge detection. Optimization Letters **11**(3), 627–640 (2017)
21. Mangasarian, O.L.: Arbitrary-norm separating plane. Operations Research Letters **24**(1-2), 15–23 (1999)
22. Pham Dinh, T., Le Thi, H.A.: Recent advances in dc programming and dca. Transactions on computational intelligence XIII pp. 1–37 (2014)
23. Schölkopf, B., Burges, C.J., Smola, A.J.: Advances in kernel methods: support vector learning. MIT press (1999)
24. Vapnik, V.: The nature of statistical learning theory. Springer science & business media (1999)