

Describing the Result of a Classifier to the End-User: Geometric-based Sensitivity

Isabelle Alvarez¹ and Sophie Martin² and Salma Mesmoudi³

Abstract. This paper addresses the issue of supporting the end-user of a classifier, when it is used as a decision support system, to classify new cases. We consider several kinds of classifiers: Statistical or machine learning classifiers, which are built on data, but also direct model-based classifiers that are built to solve a particular problem (like in viability or control problems). The end-user relies mainly on global information (like error rates or global sensitivity analysis) to assess the quality of the result given by the system. Class membership probability, if available, describes certainly the local statistical viewpoint. But it doesn't take into account other contextual information: Cases with high value of class membership probability can also be close to the decision boundary. In the case of numerical state space, we propose to use the decision boundary of the classifier (which always exists, even implicitly), to describe the situation of a particular case: The distance of a case to the decision boundary measures the robustness of the decision to a change in the input data. Other geometric concepts, such as the maximal maximal ball, can present a precise picture of the situation to the end-user. We show the interest of such a geometric study on different examples.

1 INTRODUCTION

Many real applications of Decision Support Systems are based on classification systems, and a great number of fields are concerned: Scoring, case analysis, diagnosis or pattern recognition, etc. (See [15] for examples of applications with decision trees). Numerous types of classifiers are developed: Rule-based systems; Statistical and machine learning classifiers, with a great number of methods and algorithms (see [9], [14] for references); And direct model-based classifiers, which are developed to solve a particular problem. Statistical and machine learning classifiers, when used as decision systems, provide as an outcome the class label of a new input case, possibly with some estimate of the class membership probability. Generally some information about the performance of the classifier as a model is also available (error rates or a confusion matrix). Works have also been done concerning data visualization (see for instance [6]). The end-user of a direct model relies in the best case on some sensitivity analysis at the model level, as does rarely the end-user of a machine learning classifier. The influence of parameter changes is studied at the model level (with experimental design and response surface analysis, see [19] for methods and applications). But sensitivity study or performance rate at the model level carry no information about a particular case. And even conditional probability estimates, which carry

information about the probability of the case to belong to the predicted class, don't say much about the link between a particular case and the predicted class [10]. Two cases which have the same class label and the same class membership probability can be very different in other respects. One case can be very close to the decision boundary (the boundary of the inverse image of the different classes in the input space), which means that a small change of its attribute values can change the decision. The other one can be far from the decision boundary, which means that a sizable perturbation is necessary to change the decision. This type of information concerns the context of the decision and not its probability. It can be seen as a local sensitivity analysis with the same viewpoint as in [18]. In the case of numerical state space, when it is possible to define a metric, we propose a geometric method in order to produce a contextual description of the result given to the end-user, using part of the information encompassed in the classification system but which is generally not used. We study the relative position of the decision boundary. This gives information about the robustness of the decision to perturbation or to uncertainty about the input case. If a case is far from the decision boundary, then a considerable change in its attribute values will be necessary to change the decision. On the contrary, if a case is close to the decision boundary, only a small change is sufficient.

The paper is organized as follows: Section 2 underlines the limits of the probabilistic information. Section 3 describes the geometric analysis concepts and algorithms. Section 4 presents some experimental results on both machine learning and direct model-based classifiers.

2 LIMITS OF PROBABILISTIC VIEWPOINT

The probabilistic viewpoint in classification problem considers that the result (the decision) associated to a case is completely described by the class membership conditional probability estimate. Obviously this information is very useful to assess the validity of the result. Choosing decision 'D' with probability 0.95 is rather different from choosing the same decision with probability 0.55, or worse with highest probability 0.4 in a three-class problem.

However, the probabilistic viewpoint lets aside important contextual information about the case. Figure 1 shows two cases whose class membership probability is the same and nevertheless the contextual situation is rather different. In particular, the robustness of each case to possible perturbation is very different: The distance of A from the decision boundary is much smaller than the distance of B to the same boundary (the distance from B to $p(B)$ is very large compared to the distance from A to $p(A)$). From what we know of the human perception of probabilities (see [17] for references), obviously the end-user would not consider the decision for case 'A' or

¹ Cemagref and LIP6, France, email: isabelle.alvarez@lip6.fr

² Cemagref, France

³ LIP6, France

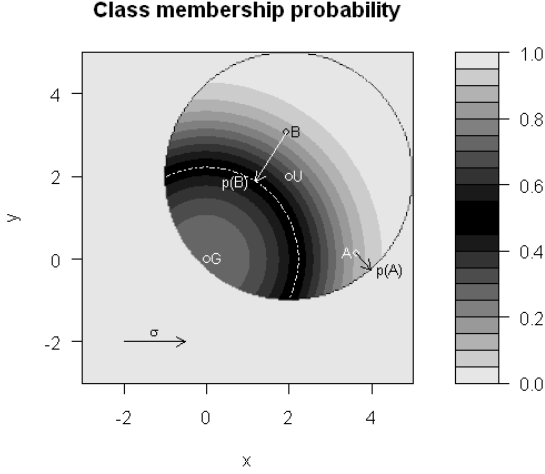


Figure 1. True class membership probability for a two class problem defined by a Gaussian distribution centered on G and a uniform distribution on a disc centered on U . The decision boundary is in black and white. Points A and B have the same class membership probability (0.86), although A is much closer to the decision boundary than B . ($p(A)$ and $p(B)$ are their respective projection onto the decision boundary).

for case 'B' in the same way, despite the probabilistic outcome.

Besides, in many situations, conditional probabilities are estimated, so they may be inaccurate. Even if the classifier result is provided as a class membership probability estimate, and if the end-user has some information to assess the reliability of this outcome (for example, error rates estimated on new cases, see [10]), it can be useful to have additional information. Probabilistic information is basically statistical; If the end-user uses the classifier repeatedly, (and when the estimates are correct), the success rate will match the probability estimate. But for a particular case, the end-user would appreciate a larger description of the situation.

3 A GEOMETRIC APPROACH TO CONTEXTUAL INFORMATION

3.1 Geometric robustness

We propose to evaluate the robustness of the decision to perturbation of the input case to give this information to the end-user as a mean to assess the situation described by the decision support system. After the building stage, which depends on the type of classifier (supervised learning, modeling, etc.), the end-user submits new cases to the classifier which predicts their class label. The classifier works like a decision function, it associates a class label $c(x)$ to each vector x of the input space. We consider here classification systems operating on numerical data, such that it is possible to define a metric on the input space E . For convenience we consider in the following that E is a finite-dimensional vector space, namely $\mathbb{R}^d$. Since a decision function performs a partition of the space, we define (classically) the robustness s of the decision at point x as the largest distance δ such that the class of points in the open ball $B(x, \delta)$ is the same as x class:

$$s(x) = \max\{\delta \geq 0 : \forall y, d(x, y) < \delta, c(y) = c(x)\}$$

We define a sensitive move $m(x)$ at x as the smallest vector v such that the robustness of the decision at $x + v$ is zero:

$$m(x) = \operatorname{argmin}_{v \in E} \{\|v\| : s(x + v) = 0\}$$

In a colloquial, non-technical sense, the decision at point x is very sensitive when it cannot resist perturbations. Formally here it means that the robustness at x is near zero. Basically, if the robustness at x is $s(x)$, it means that the decision given by the classifier will be the same even if a perturbation is applied to x , as long as the size of the perturbation is smaller than $s(x)$. For example, in figure 1, the robustness of case A is $\|p(A) - A\|$. It is very small. On the other hand, the robustness of case B , $\|p(B) - B\|$ is approximately the value of the standard error.

Robustness against perturbation gives complementary information to any statistical information given by the classifier (either the class with the highest probability estimate, or a vector of class membership probability estimates). Actually, a decision can be almost sure (a class with a probability estimate equals to or near one), and nevertheless very sensitive. This is the case for instance for determining the state of a dynamical system near bifurcation point. This is also the case for decision depending on a threshold, like means-tested benefit, choice of a device depending on weight or age, etc. In all these situations, it is important to inform the end-user about the robustness of the decision given by the classifier. With this additional information, the end-user can adapt his decision strategy (which can depend on decision cost, risk perception, etc., see [17].) On the same idea, sensitive moves carry relevant information to the end-user. Each sensitive move gathers a set of the smallest combined changes of each state variable that are necessary to change the decision. The end-user can use this information to attempt to change the decision if it is not satisfying (for instance, in diagnosis or corrective maintenance application); He can also use it to appreciate the risk of a change in the decision, when there is uncertainty on the value of the attributes.

Since every classifier induces a partition of the input space that defines a decision boundary Γ as the union of the boundaries of the different areas corresponding to the different classes, the distance of point x to the decision boundary is the robustness at x by construction. This distance is reached for some points of the boundary, which form the sensitive moves. The geometric study is then based on the computation of these projections onto the decision boundary.

When the decision boundary is described by an analytical formula, or by a set of constraints, it is possible to compute the exact value of the distance to this boundary, and also the projection points for all points of the input space, with efficient polynomial algorithms (see [4] for general convex problems; [2] and [22] for decision trees).

When the boundary is described with an implicit formula or in extension, it is necessary to approximate the distance and the projection with numerical algorithms, as in mathematical morphology.

For any compact subset of E , we propose to approximate the distance to the decision boundary in a discrete space. For simplicity, we assume that this compact subset is the unit hypercube H of E . We define G_N the regular grid with N points per dimension: G_N has N^d elements $\hat{x} = (k_1, \dots, k_d)$, $k_i \in \{0, N-1\}$ corresponding to $x = (k_1/(N-1), \dots, k_d/(N-1))$ in E . The discretized boundary $\hat{\Gamma}_N \subset G_N$ is defined as follows:

Let $\hat{x} \in G_N$ and let x be its corresponding point in E ,

$$\hat{x} \in \hat{\Gamma}_N \text{ if and only if } d(x, \Gamma) \leq \frac{\sqrt{d}}{N-1} \quad (1)$$

($\sqrt{d}$ is the diagonal length of the unit hypercube and $(N-1)$ the rescaling coefficient).

Theorem 1 Let $y \in H$ and $\hat{x}_N$, the nearest point of y in G_N . Then the exact distance of $\hat{x}_N$ to the discretized boundary $\hat{\Gamma}_N$, approximates the distance of y to the decision boundary Γ in H , $\Gamma \cap H$

Proof:

Let $P(y)$ be the projection of $y \in H$ onto $\Gamma \cap H$ and $\hat{P}(\hat{x}_N)$ be the projection of $\hat{x}_N \in G_N$ onto $\hat{\Gamma}_N$. For the Euclidean distance (but similar inequalities exist for other distances), let $\epsilon_N = \frac{\sqrt{d}}{N-1}$. By construction of $\hat{\Gamma}_N$, there exists $\hat{P}(y) \in \hat{\Gamma}_N$ such that $d(\hat{P}(y), P(y)) \leq \epsilon_N$ and $P(\hat{x}_N) \in \Gamma \cap H$ such that $d(\hat{P}(\hat{x}_N), P(\hat{x}_N)) \leq \epsilon_N$. Then, thanks to triangular inequality,

$$\begin{aligned} d(y, \Gamma \cap H) &\leq d(y, P(\hat{x}_N)) \\ &\leq d(y, \hat{x}_N) + d(\hat{x}_N, \hat{P}(\hat{x}_N)) + d(\hat{P}(\hat{x}_N), P(\hat{x}_N)) \\ &\leq d(\hat{x}_N, \hat{P}(\hat{x}_N)) + 2\epsilon_N \end{aligned}$$

For the same reason, $d(\hat{x}_N, \hat{P}(\hat{x}_N)) \leq d(y, \Gamma \cap H) + 2\epsilon_N$

When we consider the Euclidean distance, it is also possible to approximate the Euclidean projection onto the decision boundary.

Theorem 2 *Let $y \in H$ and $\hat{x}_N$, the nearest point of y in G_N . Then the projection of $\hat{x}_N$ onto the discretized boundary $\hat{\Gamma}_N$, approximates the Euclidean projection onto the decision boundary in H , $\Gamma \cap H$, outside the skeleton⁴ of the connected parts of the class areas, as N goes to infinity.*

Proof:

Let $P(y)$ be the projection of $y \in H$ onto $\Gamma \cap H$ and $\hat{P}(\hat{x}_N)$ be the projection of $\hat{x}_N \in G_N$ onto $\hat{\Gamma}_N$. $(\hat{P}(\hat{x}_N))_N$ is a bounded sequence, so it has a convergent subsequence toward a point Q . Since $d(\hat{P}(\hat{x}_N), \Gamma) \leq \frac{\sqrt{d}}{N-1}$, $Q \in \Gamma$. Moreover, from Theorem 1, $d(y, Q) = d(y, P(y))$. Since y doesn't belong to the skeleton, $P(y) = Q$. ■

3.2 Computation of the distance and projection

Algorithms that compute the distance over a grid have an exponential complexity with the dimension of the space. This is the reason why it is essential to consider algorithms with an optimal complexity.

When the distance and projection are computed at each point of the grid, the minimal complexity is at least linear in the number of points, that is in $O(N^d)$. Mathematic morphology provides such an algorithm, the Distance Transform (DT) algorithm, as described in [13]. We modified the algorithm in order to compute at the same time the Euclidean distance and the Euclidean projection.

This algorithm is defined on a hyper rectangle of $\mathbb{N}^d$, therefore the area of the input space for which the distance is computed has to be first mapped to a hyper rectangle in $\mathbb{N}^d$. This means in particular that two neighbors on any axis of the grid are at the same distance. The algorithm computes (with integer arithmetic) only the exact distance of each point of the discretized space to a subset $\bar{S}$ of $\mathbb{N}^d$.

The algorithm consists in two steps, the first step computes the distance and nearest point of the boundary on the first axis. It labels the points of the boundary if needed; The second step is called $d-1$ times and adds an axis to the previous subspace of dimension $k-1$, as described in Algorithm 1. It updates the distance value and the list of projection points in the new subspace. To each norm is associated a function that specifies how the distance in dimension k subspace is computed from the distance in dimension $k-1$.

For example, with the Euclidean distance, the square distance in dimension k subspace is given by a parabola: If $g_{k-1}(X)$ is the square distance between X and some point of the boundary, computed on the first $k-1$ axes, and if u_k is the unit vector of axis k ,

then the square distance in the dimension k subspace, $g_k(X)$ is at most $g_{k-1}(X)$. Applying the Pythagorean theorem, it is also at most $g_{k-1}(X - l \cdot u_k) + (x_k - l)^2$ for $0 < l < N$, and the min value gives the result. But the computation of all these values is suboptimal.

The algorithm is optimal, since instead of computing all the distance values, it considers a set of distance functions and computes their lower envelope. For example, for the Euclidean distance, it considers the set F of parabolas:

$$\{F^X(i) = g_{k-1}(X) + (i - x_k)^2\} (0 \leq x_k < N)$$

The square distance for all $x_k, 0 \leq x_k < N$ is given by the lower envelope of F .

The key point is the fact that two parabolas in F intersect at most once in $\mathbb{N}^2$, and that the intersection is very easy to compute: F^X intersects F^Y when $2(x_k - y_k)$ divides $(x_k^2 - y_k^2 + g_{k-1}(X) - g_{k-1}(Y))$.

This is the reason why it is optimal: The computation of the envelope is reduced to a matrix searching problem, which complexity is in $O(N)$ in this totally monotone case (see [7] and [1] for details), so the overall complexity is in $O(N^d)$. It is also the reason why the algorithm works for other distances, as long as this key point can be maintained.

For the Euclidean distance, the building function and the intersect function are defined by:

$$\begin{aligned} F^X(i) &= g_{k-1}(X) + (i - x_k)^2 \\ \text{intersect}(F^X, F^Y) &= \text{truncate}(x_k^2 - y_k^2 + g_{k-1}(X) - g_{k-1}(Y)) \\ &\quad \div 2(x_k - y_k) \quad (2) \end{aligned}$$

The same distance transform algorithm can be used to compute other distances. For example, the sup norm distance is more appropriate than the Euclidean distance when the attributes cannot substitute for one another. For the sup norm, the distance in dimension k subspace is computed from the distance in dimension $k-1$ with truncated V -shaped function:

$$F^X(i) = \max(g_{k-1}(X), |i - x_k|) \quad (3)$$

The V -shaped function has its vertex at abscissa x_k and is truncated at the value $g_{k-1}(X)$, since the distance of the sup norm in dimension k is the max of the distance for the first $k-1$ axes and of the distance along axis k , that is $|i - x_k|$ at step i . Contrary to [13] and [7], we use the following formula for the intersection of truncated V -shaped functions F^X and F^Y (with $x_k \leq y_k$):

$$\text{intersect}(F^X, F^Y) = \begin{cases} \max((x_k + g_{k-1}(Y)), ((x_k + y_k) \div 2)) & \text{if } g_{k-1}(X) \leq g_{k-1}(Y) \\ \min((y_k - g_{k-1}(X)), ((x_k + y_k) \div 2)) & \text{otherwise} \end{cases} \quad (4)$$

Algorithm 1 SquareEuclideanDistanceToSet

Input: The distance (or square distance) g_{k-1} computed on the $k-1$ first axes of G , the list p_{k-1} of labels of the projection points

Result: g_k computed on the k first axes for each point of $G = [0, N-1]^d$, the updated list of labels p_k

For all $X = (x_1, \dots, x_{k-1}, x_{k+1}, \dots, x_d) \in [0, N-1]^{d-1}$ do {

$q \leftarrow 0; s[0] \leftarrow 0; t[0] \leftarrow 0; f[0] \leftarrow g[s[0]]$

/* s is the stack of building functions (bf),
 t the stack of bf intersections, f the stack

⁴ The skeleton S of A is the set of the centers of all maximal open balls in A (see [12] for a study of its topological properties. We suppose here that the skeleton is the complementary set of a dense open set)

```

of bf minimums as in  $F_{s[q]}^X(i) = f[q] + (i - s[q])^2$  */
*Building function recruitment
For  $i \in [0, N - 1]$  do {
  While ( $q > 0$  and  $F_{s[q]}^X(t[q]) > F_i^X(t[q])$ ) do {
/*the present building function dominates*/
     $q \leftarrow q - 1$  }
  If  $q = 0$  Then {
/*present building function replaces all*/
     $s[0] \leftarrow i$ ;  $f[0] \leftarrow g[i]$  } Else {
     $w \leftarrow 1 + \arg(\text{intersect}(F_{s[q]}^X, F_i^X))$ 
    If  $w < N$  Then {
/* present building function is recruited
from  $q+1$  */
     $q \leftarrow q + 1$ ;  $s[q] \leftarrow i$ ;  $t[q] \leftarrow w$ ;  $f[q] \leftarrow g[i]$  } }
*Assignment: following the building functions envelope
flag  $\leftarrow$  NULL
For  $i$  from  $N - 1$  to 0 do {
  If flag  $\neq$  NULL Then {
/*skeleton: Add also previous value */
     $g_k^X(i) \leftarrow F_{s[q]}^X(i)$ ;  $p_k^X(i) \leftarrow p_k^X(s[q])$ , flag }
  Else {
/* normal case */
     $g_k^X(i) \leftarrow F_{s[q]}^X(i)$ ;  $p_k^X(i) \leftarrow p_k^X(s[q])$  }
  flag  $\leftarrow$  NULL
  If  $i = t[q]$  Then {
/* present and next points have two
projections (skeleton) */
    flag  $\leftarrow p_k^X(s[q])$ 
     $q \leftarrow q - 1$ 
/* add next projection point to the list */
     $p_k^X(i) \leftarrow p_k^X(i), p_k^X(s[q])$  } }
Return  $g_k, p_k$ 

```

With other algorithms coming from computational geometry [5], it is possible to compute the Voronoi diagram of the points of the discretized boundary. The complexity of these algorithms are in $O(n \log n)$ where n is the number of points of the boundary. Since a boundary is generally locally isomorph to a co-dimension 1 manifold, the complexity is in $O(N^{d-1} \log N)$, on a grid of size N . But when the boundary is implicit, it is given by the complementary set of one class, which is of the same magnitude than the grid itself ($O(N^d)$). So in this case the DT algorithm is more efficient.

The DT algorithm applies to nondimensionalized data, so model experts have to propose a suitable coordinate system in which the input state becomes an Euclidean space. Since the geometric study takes place after the building of the classifier, some information about the input space is generally available (range of the data, estimate of the mean, standard error, etc.), either from the data that were used during the building phase, or from domain expert. Two basic metrics can be used when only few information is available on the input space, the Min-Max (MM) metric and the standard (s) metric, corresponding to two different coordinate systems (see equation 5, where E_i and s_i are estimates of the mean and the standard deviation of attribute i , and $\text{Min}(i)$ and $\text{Max}(i)$ are the minimum and the maximum of the range of attribute i).

$$y_i^{MM} = \frac{x_i - \text{Min}_i}{\text{Max}_i - \text{Min}_i} \quad \text{or} \quad y_i^s = \frac{x_i - E_i}{s_i} \quad (5)$$

The Min-Max metric can be preferred when the variables are sensor measurements, since the accuracy of physical sensors is generally a function of the total range.

In the new coordinate system, the input space becomes an inner product space. Apart from the Euclidean distance, other distances like the sup norm (L^∞) or Manhattan (L^1) can be used. The sup norm is generally preferred when the uncertainties for each attributes can combine freely. It is the case for example, in process control, when different controls are applied each with its own uncertainty.

4 EXPERIMENTATION

Although the complexity of the algorithm is exponential with the dimension, table 1 shows that it can be used in many situations up to dimension 5. We have considered two different problems. In problem A each class is a half-space of the input space. In problem B one class is inside a half-sphere centered on one axis, and the other class is outside. We used one core AMD Opteron(tm) Processor 8378 clocked at 2,4 GHz and 90Go of DDR-2 RAM.

Table 1. Examples of computation time (and total number of points) depending on the state space dimension and the number N of points per axis

Halfspace problem: Dimension					
N	8	7	6	5	4
10	34m58s	2m37s	0m11s	0m1s	0m0.1s
20	*	195m44s	8m44s	0m19.45s	0m1s
points	*	1, 28.10 ⁹	6, 40.10 ⁷	3, 20.10 ⁶	1, 60.10 ⁵
25	*	*	30m52s	0m57s	0m2s
31	*	*	93m53	2m14s	0m3s
points	*	*	8, 8810 ⁸	2, 86.10 ⁷	9, 24.10 ⁵
60	*	*	*	69min1s	0m47s
100	*	*	*	*	5m53s
177	*	*	*	*	61m43s
Halfsphere problem: Dimension					
N	8	7	6	5	4
10	36m52.3s	2m44s	0m12s	0m1s	0m0.1s
20	*	175m35s	9m11s	0m20s	0min1s
25	*	*	32m47s	0m57s	0m2s
31	*	*	96m29s	2min18s	0min4s
60	*	*	*	68m42s	0m48s
100	*	*	*	*	6m5s
177	*	*	*	*	58m43s

4.1 Examples with Machine learning classifiers

We develop here an example of support vector machine (SVM), which is a very popular classification method (see [20] for references), since SVM algorithm finds the hyperplane H with the highest margin that separates the data and minimizes the norm of its normal vector (in a high dimensional space where the data can possibly become separable). We also develop an example of naive Bayes classifier (NBC), which is more appropriate to estimate class membership probabilities. But the method applies to any classifier as soon as the input space is numerical.

We built a SVM on a 2-class problem in a 5 dimensional Euclidean input space. 1000 points per class are drawn from a standard Gaussian distribution. One class is centered on (0, 0, 0, 0, 0) and the other on (2, 2, 2, 2, 2). Points are labeled respectively. A SVM classifier was built on these training points, using R kernlab package ([8]) with default parameters: C classification, radial basis kernel function with heuristic estimation of its sigma parameter, and estimation

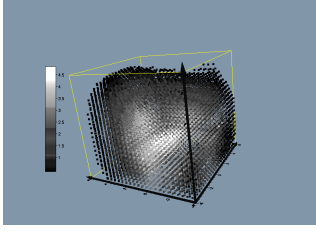


Figure 2. 3d-section ($u_5 = u_4 = 0$) of the distance computation to one class boundary in a 5 dimensional state space. The boundary has no color.

of the posterior probabilities with a sigmoid function fitted on the resulting SVM function values of a 3-fold cross validation performed on the training data (as in [16]). The error of the SVM on the training cases is small (0.0135). We then computed the class membership probability and the distance to the implicit boundary of the probabilistic model on a grid of about $11 \cdot 10^6$ points (26 points on each dimension). Figure 2 shows a section of the distance computation.

As it is explained in section 2, there is no reason why the posterior probability and the distance to the decision boundary should be correlated, except near the decision boundary. Table 2 sums up correlation results, obtained using Spearman's rank correlation coefficient (that is Pearson's correlation coefficient between the ranked variables). The distance and the posterior probability are highly correlated for values of the probability near 0.5. On the other hand, there is very few correlation for the high values of the class membership probability. Very similar values were obtained on a smaller grid (16 points on each variable): (for example, for class 1, 0.277, 0.160, 0.05, -0.02 instead of 0.267, 0.160, 0.044, -0.031 for respectively the 40%, 30%, 20%, and 10% of highest values of the posterior probability).

Table 2. Spearman's rank correlation coefficient between the distance to the decision boundary and the class membership probability p

ρ	Quantiles of $ p - 0.5 $					
	highest values				smallest values	
Class	10%	20%	30%	40%	30%	40%
$p > 0.5$	-0.031	0.044	0.160	0.267	*	0.782
$p < 0.5$	-0.105	-0.005	0.151	0.044	0.832	0.758

ρ is between -1 and 1 , 0 means no correlation.

It is also possible to find points with class membership probability near 0.5 (for example, above 0.45), and yet very far from the decision boundary. Conversely, points (more than 1000) relatively near the decision boundary, (closer than the standard deviation), can have a class membership probability over 0.99. For these points, the direction given by the projection point can help the end-user to assess the risk of a decision change associated with a perturbation in the input data.

We trained a naive Bayes classifier on a 2-class problem in a 4-dimensional Euclidean input space, with 400 points per class, from Gaussian distribution centered on (1,1,1,1) and (3,3,3,3) respectively, with rather different standard deviation vectors, (1/2,1,1,2) and (1.4,1.2,1,0.4) respectively. Since the NBC algorithm builds Gaussian distributions as estimate of the class membership probabilities, the distance to the decision boundary is necessarily correlated to the latter (since level curves are ellipsoids): For the 10% of highest values of the posterior probability, the correlation coefficient is still

0.40. But the main interest of the method is that it can help to focus on dangerous situation such as in Figure 1. Figure 3 shows an area of the input space where points with high class membership probability can be very close to the decision boundary.

4.2 Application to a direct model-based classifier: A viability kernel for the eutrophication lake problem

We consider here an example of classifier for which the decision boundary is not available analytically. This method is used in particular to model usage conflicts in environmental engineering, when the dynamics of the socio ecosystem are known as a dynamical system in a state space E . The set of acceptable states (from both ecological and socioeconomic viewpoints) is defined by a set K of constraints in the state space. Possible actions on the socio ecosystem are represented by controls. In that framework, using the concepts and methods of viability theory [3], the viability kernel delimits the area of the state space where there is always a sequence of controls that allows the system to verify indefinitely the constraints. The viability kernel can be seen as a classifier: Viable states are inside the viability kernel. The decision boundary is the boundary of the viability kernel.

The illustration we develop here is the problem of lake eutrophication, which is the sudden shift from clear-water (oligotrophic) to turbid-water (eutrophic) state. Oligotrophic state has obviously a tremendous higher economic value of ecosystem services. The issue is to determine whether the lake can remain in an oligotrophic state or if it is doomed to become eutrophic in finite time. The problem can be modeled by the dynamic between phosphorus concentration P and phosphorus inputs L (excess phosphorus is imported by farms, through fertilizer and animal feed supplements). The main constraints are that agriculture requires a minimum value for L , L_{min} , and oligotrophic state requires a maximum value for P , P_{max} . In this model, regulation laws are constraints on $|\frac{dL}{dt}| \leq V$: this means that the regulation law cannot set the maximum amount of phosphorus inputs, but rather imposes a decrease of the phosphorus inputs, for example by a percentage each year. This decrease is bounded by V . The viability kernel is the subset of the (L, P) -plane that gathers all states (L, P) such that there exists at least one regulation law that allows the oligotrophic state to be maintained [11].

The information of the distance to the viability kernel boundary is

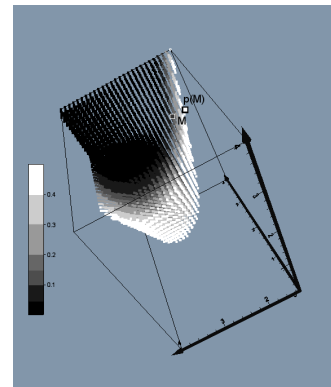


Figure 3. 3d-section ($u_1 \approx 1.74$) of the class membership probability (one class only) in a 4 dimensional state space. Point M has a low class membership probability (0.099), and this suggests that it is fairly in one class side, although it is very close to the boundary ($d=0.3$). Its projection $p(M)$ belongs to the same section, it is reached along the x-axis.

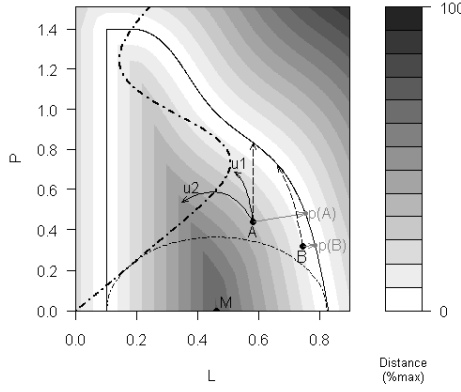


Figure 4. Distance map for the viability kernel boundary of the lake eutrophication problem (with $P_{max} = 1.4$, $L_{min} = 0.1$, $V = 0.09$). The black line is the viability kernel boundary. The dot dashed line is the set of equilibria. $p(A)$ ($p(B)$) shows the direction and size of the most sensitive disturbance at point A (B respectively). Some trajectories are computed with constant negative or null control. Long dashed trajectories that leave the viability kernel are not robust.

valuable because of measure uncertainty or exogenous disturbances (which may cause a sudden increase in phosphorus concentration for instance). Figure 4 shows the viability kernel with the level curves of the distance to its boundary. The geometric study shows when a state is dangerously close to the decision boundary. For example, point B stands very close to the boundary compare to A : it is less robust to perturbation. The projection onto the boundary shows the direction of the most dangerous disturbance. Figure 4 shows that when the lake is in state B , a small increase of phosphorus input (L) combined with a very small error in the phosphorus concentration (P) can shift the state of the lake outside the viability kernel, where it is doomed to become eutrophic at some time in the future.

The geometric study also gives global information concerning the problem: Point M is the center of the maximal maximal ball in the viability kernel. It is the farthest point from the boundary with $d(M, \Gamma) \approx 0.36$. This distance can be compared with (half) the size of the constraint set ($0.9/2 = 0.45$ along L). So this viability kernel has a reasonable size, it should be easy to find action policy that guarantees an oligotrophic state.

Furthermore, the distance to the viability kernel boundary allows to propose geometric definitions of the robustness of trajectories inside the viability kernel. For example:

Definition 1 Robustness of a trajectory.

Let $u : t \mapsto u(t)$ be a trajectory in the viability kernel $Viab(K)$. We note Γ_V its boundary. The robustness value of u is $r(u) = \min_t \{d(u(t), \Gamma_{Viab(K)})\}$.

With this definition, the trajectories starting from A and B which are dashed lines in Figure 4 have a robustness value of zero: they leave the viability kernel in finite time. Other trajectories starting from A with constant (negative) control have a strictly positive robustness value, and we can see in Figure 4 that $r(u_2) > r(u_1)$. This geometric indicator, based on the distance to the boundary, can help to choose between control sequences.

In this example, which was chosen for illustration purpose, the dimension of the input space is 2. But the same kind of geometric approach can be used up to dimension 5 problems, as in [21] for a dimension 5 problem of cheese ripening process.

5 CONCLUSION

Considering statistical or direct model-based classification systems, we aimed at providing information on the robustness of the decision facing a change in the input data. Robustness is measured as the maximal strength of the perturbation in the input data which does not provoke a change in the decision. Such a local information is neither contained in global information such as error rates or model sensitivity analysis, nor in local statistical information. Robustness is linked with geometric concepts : the diameter of the maximal ball contained in one class area provides an upper bound of the robustness for all the elements of this class, whereas the distance between a particular point and the boundary of its class area and its projection on this boundary provides the intensity and the direction of the perturbation with minimal norm that would lead to a decision change. We have proposed optimal algorithms to compute these geometric concepts and carried them out on some examples: machine learning and a direct model-based classifiers (viability kernel).

REFERENCES

- [1] A. Aggarwal, M. Klowe, S. Moran, P. Shor, and R. Wilber, 'Geometric applications of a matrix-searching algorithm', *Algorithmica*, **2**, 195–208, (1987).
- [2] I. Alvarez and S. Bernard, 'Ranking cases with decision trees: a geometric method that preserves intelligibility', in *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence 2005*, pp. 635–641, (2005).
- [3] J.P. Aubin, *Viability Theory*, Birkhauser, Basel, 1991.
- [4] H. Bauschke and J.M. Borwein, 'On projection algorithms for solving convex feasibility problems', *SIAM Review*, **38**(3), 367–426, (1996).
- [5] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry*, Springer-Verlag, 2000.
- [6] T.N. Do and F. Poulet, 'Enhancing svm with visualization', in *Proc. of Discovery Science 2004*, pp. 183–194. Springer-Verlag, (2004).
- [7] T. Hirata, 'A Unified Linear-Time Algorithm for Computing Distance Maps', *Information Processing Letters*, **58**(3), 129–133, (1996).
- [8] A. Karatzoglou, A. Smola, K. Hornik, and A. Zeileis, 'kernel package for kernel methods in R', *Journal of Statistical Software*, **11**(9), 1–20, (2004).
- [9] S. B. Kotsiantis, 'Supervised Machine Learning: A Review of Classification Techniques', *Informatica*, **31**, 249–268, (2007).
- [10] M. Kukar and I. Kononenko, 'eliable classifications with machine learning', in *Proc. ECML-02*, pp. 219–231. Springer, (2002).
- [11] S. Martin, 'The cost of restoration as a way of defining resilience: a viability approach applied to a model of lake eutrophication', *Ecology and Society*, **9**(2), (2004).
- [12] G. Matheron, *Examples of topological properties of skeletons*, 217–257, Image Analysis and Mathematical Morphology, Academic Press, 1988.
- [13] A. Meijster, J. Roerdink, and W.H. Hesselink, 'A General Algorithm for Computing Distance Transforms in Linear Time', *Morphology and Its Applications to Image and Signal Processing*, 331–340, (2000).
- [14] T. Mitchell, *Machine Learning*, McGraw Hill, 1997.
- [15] S.K. Murthy, 'Automatic construction of decision trees from data: A multi-disciplinary survey', *Data Mining and Knowledge Discovery*, **2**(4), 345–389, (1998).
- [16] J. Platt, *Probabilistic outputs for support vector machines*, 61–74, Advances in Large Margin Classifiers, MIT Press, 2000.
- [17] S. Plous, *The Psychology of Judgment and Decision Making*, McGraw-Hill, 1993.
- [18] H.M. Regan, Y. Ben-Haim, B. Langford, W. Wilson, and P. Lundberg, 'Robust decision-making under severe uncertainty for conservation management', *Ecological Applications*, **15**(4), 1471–1477, (2005).
- [19] A. Saltelli, K. Chan, and M. Scott, *Sensitivity Analysis*, Wiley, 2000.
- [20] J. Shawe-Taylor and N. Cristianini, *Support Vector Machines and other kernel-based learning methods*, Cambridge University Press, 2000.
- [21] M. Sicard, S. Martin, R. Reuillon, S. Mesmoudi, and N. Perrot, 'Viability theory to control complex food processes. Application to Camembert ripening process', in *ECCS*, (2009).
- [22] N. Toth and B. Pataki, 'On classification confidence and ranking using decision trees', in *Proc. of INES2007*, (2007).