A physical approach to binary classification

Alexandre Quemy
IBM Software Lab
Cracow, Poland
Faculty of Computing, Poznań University of Technology
Poznań, Poland
aquemy@pl.ibm.com

ABSTRACT

In this paper, we model a training set as a distribution supposedly behaving as described by the heat equation. We propose a classification algorithm that numerically solves the problem using a spectral decomposition to determine the classification boundary. This iterative algorithm updates the initial source of heat depending on the classification accuracy. We study the impact on the different parameters and compare the classification accuracy to standard machine algorithms on three types of datasets. The approach provides consistently good results, reaching similar scores than some of the state-of-the-art learning techniques such as Gradient Boosting or Extra Tree classifier.

1 INTRODUCTION

By the number of situations that can be modeled as a classification problem, it is one of the most widely studied problem in machine learning. Concretely, it consists in distinguishing between objects of different classes. Over the years, some theoretical frameworks emerged to unify and mathematically justify the different methods. With elegant formalism using a mixture between algebra, probabilities and analysis, most of the approaches start by modeling the problem as an optimizing problem in a certain hypothesis space, i.e. a space of potential solutions to the problem. The optimal argument of the optimization program provides the *best* fitting model to solve the problem. The different techniques to solve this optimization problem, s.a. gradient descent, are then studied to prove mathematical properties on the solution and its existence: existence and uniqueness of solutions, smoothness, convergence with the number of training examples, etc.

In this paper, we adopt a different approach by modeling the classification problem using a purely physical point of view and validating the model pertinence a posteriori, by numerical experiments. No direct optimization problem is used, and the model provides an explicit analytic form. We justify the usage of the model by few reasonable assumptions that are, in general, implicit in all machine learning algorithms.

The contribution of this paper are twofold. First, we introduce in Section 4 a new classification algorithm based on the direct resolution of the diffusion equation. In particular, we discuss in Section 4.1 the difference with kernel-based SVM methods, and in particular, with Radial basis function (RBF) SVM [4]. Last, we provide numerical experiments to validate the approach in Section 5.

The plan of this paper is as follows: in Section 2, we present related work on classification problems. In particular, we present non-linear SVM techniques. Indeed, the approach presented in this paper directly relates to SVM with a Gaussian kernel, with a totally different point of view on the way to solve the problem. In Section 3, we introduce the heat diffusion problem and provide the key steps to prove the existence of solutions using Lions-Lax-Milgram theorem, and explicit them with a spectral decomposition that can be applied to arbitrarily large spaces. In Section 4, we present the application of those solutions to the binary classification problem, including the initial hypothesis and an iterative training algorithm to adjust the heat source such that it improves the decision boundary over iterations. Section 5 provides a numerical validation, including a comparison with other standard machine learning methods and a study on the influence of the main model parameters.

2 RELATED WORK

2.1 Classification as an optimization problem

We suppose the existence of an unknown function f over Ω that allows to correctly classify any point of Ω. The problem of classification is to find $\bar{f}$, an approximation of f [7]. To do so, a hypothesis space H is defined which represents the space in which to look for the approximation. This space needs to be large enough to include the exact function but restricted enough for the algorithm to provide a good approximation in a reasonable time. The definition of H and the search method defines the machine learning technique.

The learning problem can be written as an optimization problem as follows:

$$f^* = \operatorname*{argmin}_{\bar{f} \in H} \int_{\Omega} ||f(\mathbf{x}) - \bar{f}(\mathbf{x})||_{\mathrm{Y}} \mu(\mathbf{x}) d\mathbf{x} \tag{1}$$

where μ is the distribution of appearance of elements in Ω and $||.||_{\mathrm{Y}}$ a certain norm on the output space.

In practice, it is impossible to evaluate the generalization error, such that we minimize the empirical error. Given a training set $(\mathbf{X}, \mathbf{y})$, solving the classification problem is practically equivalent to

$$f^* = \operatorname*{argmin}_{\bar{f} \in H} \sum_{(\mathbf{x}, y) \in (\mathbf{X}, \mathbf{y})} \mathbb{I}_{f(\mathbf{x}) = \bar{f}(\mathbf{x})} \tag{2}$$

.

To give an example, linear classification methods formulate the hypothesis space as the space of M-hyperplanes which can be uniquely defined by a vector $\mathbf{w}$. and a bias w_0 s.t. $h_{\mathbf{w}}(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0 = 0$. Thus, finding a solution to the classification problem consists in solving

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w}} \sum_{\mathbf{x} \in \mathbf{X}} \ell(\mathbf{w}, \mathbf{x}) + \lambda R(\mathbf{x}) \tag{3}$$

where ℓ is a loss function, R is a regularization term and λ an hyperparameter to control its effect [12].

2.2 Non-linear SVM and Gaussian kernel

The Equation (3) is the base of SVM [16], logistic regression [5] or the Perceptron algorithm [15] . However, it can solve only linearly separable problems. To handle non-linearly separable problems, the main idea is to use the kernel *trick* to map the training set into a higher dimensional space s.t. it becomes linearly separable [10]. Let denote by $\phi : \Omega \longrightarrow \mathcal{V}$ this feature map.

A kernel is a symmetric continuous function $K : [a,b] \times [a,b] \longrightarrow \mathbb{R}$ s.t. $K(x,y) = K(y,x)$. The kernel is positive semi-definite iff $\sum_i^n \sum_j^n K(x_i, y_j)c_i c_j \geq 0$ for any finite sequence of x_i and any choice of real for c_i. A linear classifier computes a prediction for an unlabeled input x' as $\bar{y} = \text{sgn} \sum_{i=0}^n w_i y_i < x_i, x' >$. ϕ image being in a high dimensional space, it makes the scalar product in $\mathcal{V}$ impossible to compute in practice. The kernel trick thus consists in replacing the costly scalar product by a kernel function such that $K(x,y) =< \phi(x), \phi(y) >$. More precisely, Mercer's Theorem states that for any positive semi-definite kernel on $\Omega \times \Omega$ there exists a well define scalar product in $\mathcal{V}$ such that $K(x,y) =< \phi(x), \phi(y) >$. This means that in practice, we select K a priori rather than determining it depending on a certain ϕ. A typical kernel-based classifier then computes $\bar{y} = \text{sgn} \sum_{i=0}^n w_i y_i K(x_i, x')$ where the kernel is used as a similarity measure.

In summary, the kernel trick allows linear classifier to learn non-linear boundaries by replacing a scalar product in a high dimensional space by an easy-to-compute kernel function. For a more formal and exhaustive presentation of kernel-based methods, we refer the reader to [6, 10, 13]

One popular kernel for SVM is the Gaussian kernel [17], also known as Radial basis function, defined by $K_G(x,x') = \exp(\frac{||x-x'||^2}{2\sigma^2})$ where σ how far is the influence of examples around them.

3 HEAT EQUATION

The heat equation is trivially derived from the continuity equation that describes the continuous transport of conserved quantities. In this case, the quantity is heat and the equation refers to the diffusion of heat in a solid. Fourier was the first to understand that the rate of heat through a material is proportional to the negative gradient of temperature [9]. He also was the first to provide a solution to this equation for the simple cases.

3.1 The problem

$$\begin{cases} \frac{\partial u}{\partial t} - \Delta u = f(x,y,t) \in L^2(\Omega), \Omega =]0,1[^2 \times]0,T[\\ u(x,0,t) = 0 \\ u(x,1,t) = 0 \\ \frac{\partial u}{\partial n}(0,y,t) = 0 \\ \frac{\partial u}{\partial n}(1,y,t) = 0 \\ u(x,y,0) = 0 \end{cases} \tag{HE}$$

This standard equation models heat diffusion in a two dimensional surface of an homogeneous material, heated according to a source f of finite energy. Dirichlet and Neumann conditions specify respectively the behavior of heat on the vertical edges and the heat direction on the horizontal edges. Finally, the initial condition provides the initial heat distribution over the spatial space. We assume the problem to be homogeneous, i.e. the boundary conditions are equalized to zero.

The general technique to solve the problem would work for a N dimensional hypercube and is easy to adapt to regular domain such as a sphere or torus.

3.2 Solving the problem

In this section, we provide the main steps to solve the problem (HE). For a formal demonstration of Lions-Lax-Milgram theorem and further reading on functional analysis, we refer the reader to [1, 3].

First, we considering only the spectral part of the problem:

$$-\Delta w = \lambda w \tag{SP}$$

Thanks to Lions-Lax-Milgram theorem [11], we know that a weak solution exists. Using a variational approach coupled to the spectral decomposition of self-adjoint compact operators, we can explicit the solutions that exists almost everywhere in $L^2(\Omega)$.

Theorem (Spectral theorem): *Let us assume V and H two real Hilbert spaces of infinite dimensions. We suppose $V \subset H$ with compact injection (i.e. the inclusion operator is continuous and compact), and that V is dense in H. Let $a(.,.)$ be a bilinear symmetric continuous form satisfying V-ellipticity. Then, the eigenvalues of a s.t. $\forall v \in V, a(u,v) = \lambda < u,v >_H$ form an increasing sequence $(\lambda_k)_{k\geq 1}$ of positive reals that goes to infinity, and there exists a Hilbert base of H, $(u_k)_{k\geq 1}$ of eigenvectors:*

$$u_k \in V, \ et\ a(u_k, v) = \lambda_k < u_k, v_k >_H, \ \forall v \in V \tag{4}$$

In addition, $(\frac{u_k}{\sqrt{\lambda_k}})_{k\geq 1}$ is an Hilbert base of V embeded with the scalar product $a(.,.)$.

Let $H_0^1(\Omega)$ be the closure of $C_c^1(\Omega)$ in the Sobolev space $H^1(\Omega)$, and $\mathcal{D}(\Omega)$ the space of infinitely differentiable functions with compact support, also known as test functions. To take into account the Dirichlet conditions, we define $V = H_0^1(\Omega)$, and H as the Lebesgue space $L^2(\Omega)$. Using a variational approach on the spectral problem (SP), we deduce that $a(u,v) = \int_\Omega \nabla u.\nabla v dx = \lambda \int_\Omega uv dx = \lambda < u,v >_{L^2(\Omega)}$. According to Rellich theorem $H_0^1(\Omega)$ is compactly included in $L^2(\Omega)$. As $\mathcal{D}(\Omega)$ is dense in both spaces, we can deduce that $H_0^1(\Omega)$ is dense in $L^2(\Omega)$. Therefore, we can apply the spectral theorem above which demonstrates the existence of solutions to (SP) in $H_0^1(\Omega)$.

To come back to the original problem, a Green formula and a trace theorem is enough to conclude. A simpler alternative is to use the space of distributions:

$$\begin{aligned} \int_\Omega \nabla u.\nabla v dx &= \lambda \int_\Omega uv dx \\ \Leftrightarrow < \nabla u, \nabla v > &= \lambda < u,v > \\ \Leftrightarrow - < \Delta u, v > &= \lambda < u,v > \end{aligned} \tag{5}$$

, thus $-\Delta u = \lambda u$ in the sense of distribution. Thanks to the density of $\mathcal{D}$ in V and H, and using the compact canonic injection, we deduce the equivalence in L^2.

The existence of solution proven, we look for explicit solutions. A remarkable property of the heat equation is that the solution is separable. In this paper, we consider only the problem with a two dimension spatial space, but the technique generalizes to any dimension. In other words, we assume that the solution is of the following form:

$$w(x,y) = X(x)Y(y) \tag{6}$$

Using the initial conditions, the solution are given by the following classic result:

$$\begin{cases} w_{k,l}(x,y) = Csin(l\pi y)cos(k\pi x) \\ \lambda_k = \pi^2(l^2 + k^2) \end{cases} \quad , \forall k \geq 0 \tag{7}$$

The normalization constant is easily obtained by the following:

$$||w_{l,k}||_{L^2(\Omega)} = 1 \Leftrightarrow \begin{cases} C = \sqrt{2} & \text{if } k = 0 \\ C = 2 & \text{otherwise} \end{cases} \tag{8}$$

The solution of (SP) can be writen as follows:

$$w(x, y) = \sum_{l,k} \xi_{l,k}(t) w_{l,k}(x, y) \tag{9}$$

As f is assumed to be in $L^2(\Omega)$, it can also be expressed in the Hilbert base by $f(x, y) = \sum_{l,k} f_{l,k}(t) w_{l,k}(x, y)$ with $f_{l,k}(t) =< f(x, y, t), w_{l,k}(x, y) >_{L^2(\Omega)}$. We can finally rewrite the problem in the Hilbert base:

$$\begin{aligned} \frac{\partial}{\partial t} \sum_{l,k} \xi_{l,k}(t) w_{l,k}(x, y) - \Delta \sum_{l,k} \xi_{l,k}(t) w_{l,k}(x, y) \\ = \sum_{l,k} f_{l,k}(t) w_{l,k}(x, y) \end{aligned} \tag{10}$$

By linearity,

$$\sum_{l,k} [\xi'_{l,k}(t) - \lambda_{l,k} \xi_{l,k}(t)] w_{l,k}(x, y) = \sum_{l,k} f_{l,k}(t) w_{l,k}(x, y) \tag{11}$$

, which is equivalent to solve ordinary differential equations

$$\xi'_{l,k}(t) - \lambda_{l,k} \xi_{l,k}(t) = f_{l,k}(t), \ \forall k, l \geq 0 \tag{12}$$

4 CLASSIFICATION USING EXPLICIT HEAT EQUATION

In this paper, we simplify the problem by removing the time dimension. This removes the need to compute the ODEs:

$$\xi_{l,k} = \frac{f_{l,k}}{\lambda_{l,k}}, \ \forall k, l > 0 \tag{13}$$

The hypothesis behind using the heat equation for binary classification is that for a point of a given class, the confidence level for this class around this point evolves continuously as described by the heat equation.

Consider Ω as the unit hypercube of dimension N. We define the source f as a certain prior with the following constraint: $\forall \mathbf{x} \in X, \ f(\mathbf{x}) \in [-1, 1]$. Given a training set $(\mathbf{X}, \mathbf{y})$ with $\forall y \in Y, y \in \{0, 1\}$, we define $\forall \mathbf{x}, y \in \mathbf{X} \times Y, f(\mathbf{x}, t) = y$ and $\forall \mathbf{x} \in \Omega \setminus \mathbf{X}, f(\mathbf{x}, t) = 0$.

To compute the solution, we use an homogeneous grid of step h in every dimension. We denote by Ω_h the discretized space. The projection on the Hilbert base is done using Simpson numerical integration using the M first eigenvectors. Once the solution $u \in L(\Omega_h)$ is obtained and normalized within $[-1, 1]$, the decision boundary is defined by a level set defined by $\beta \in [-1, 1]$. In practice, β is set to 0 and can be automatically adjusted as hyperparameter using e.g. a grid search or more advanced hyperparameter tuning techniques.

The solution is then evaluated on the training set. For each misclassified input vector, its corresponding heat value in the source is updated depending on its margin, and a regularization factor as described by Algorithm 1. The problem is then solved again for u with the updated source. This procedure is performed K times.

4.1 Relation with Gaussian Kernel

In this section, we try to clarify the similarities and differences between non-linear SVM using gaussian kernel and the approach presented in this paper, namely solving the heat equation, considering the heat as class probability for a given point.

Algorithm 1 Model generation for hypercubes

```
Input:
 - $(\mathbf{X}, y)$: a training set
 - $N$: hypercube dimensions
 - $M$: number of eigenvalues
 - $h$: grid step size
 - $K \geq 1$: number of source adjustment iterations
 - $\alpha \geq 1$: regularization factor
 - $\beta \in [0, 1]$: level set to use as decision boundary
Output:
 - $u_h \in L^2(\Omega_h)$
 1: $(\lambda_i, w_i)_i \leftarrow$ precompute_eigenvalues()
 2: $f^{(0)} \leftarrow$ generates_source(X, y)
 3: for $k \in \{1..K\}$ do
 4:     $(f_i^{(k)})_i \leftarrow \int_{\Omega_h} f^{(k)}(\mathbf{x}) w_i d\mathbf{x}$
 5:     $(\xi_i^{(k)})_i \leftarrow \frac{f_i^{(k)}}{\lambda_i}$
 6:     for $(\mathbf{x}, y) \in \mathbf{X} \times Y$ do
 7:         $u(\mathbf{x}) \leftarrow \sum_{i=0}^{\frac{N}{H}} (\xi_i^{(k)})^T w_i$
 8:     end for
 9:     for $(\mathbf{x}, y) \in \mathbf{X} \times Y$ do
10:         $\bar{y} = 1$ if $u(\mathbf{x}) \geq \beta$ else $-1$
11:         if $\bar{y} \neq y$ then
12:             $f^{k+1} \leftarrow f^k + y\alpha(1 - |u(\mathbf{x}) - \beta|)$
13:         end if
14:     end for
15: end for
```

The heat kernel of $\mathbb{R}^N$ is defined as follows:

$$K_H(t, x, y) = \frac{1}{(4\pi t)^{\frac{N}{2}}} \exp(-\frac{|x - y|^2}{4t}) \tag{14}$$

This time-varying function is solution of $\frac{\partial K}{\partial t}(t, x, y) = \Delta K(t, x, y)$ for all $t > 0; x, y \in \mathbb{R}^N$ under the following condition:

$$\lim_{t \to 0} \int_{\mathbb{R}^N} K_H(t, x, y) \phi(y) = \phi(y) \tag{15}$$

for any test function $\phi \in \mathcal{D}(\mathbb{R}^N)$. This is the limit in the sense of distribution s.t. the value of the heat kernel next to zero tends toward the dirac distribution $\delta(x - y)$. Therefore, K is a Green's function of the heat equation [2].

In comparison, the gaussian kernel does not depend on time and is defined by:

$$K_G(x, y) = \exp(\frac{||x - y||^2}{2\sigma^2}) \tag{16}$$

To allow the kernel trick in RBF SVM, Mercer's therem is used. In particular, it maps the kernel to a linear operator of the following form:

$$T_K \phi(x) \int_a^b K_G(x, y) \phi(y) dy \tag{17}$$

for any function $\phi \in L^2([a, b])$. This is this operator on which a spectral decomposition is applied and justified by Mercer's theorem [8].

In the physical approach of Section 3, the model describing heat diffusion is defined, then Lions-Lax-Milgram theorem used to prove the existence and uniqueness of solution under restrictive conditions. To explicit the solution and being able to numerically approach it, the spectral theorem is applied because

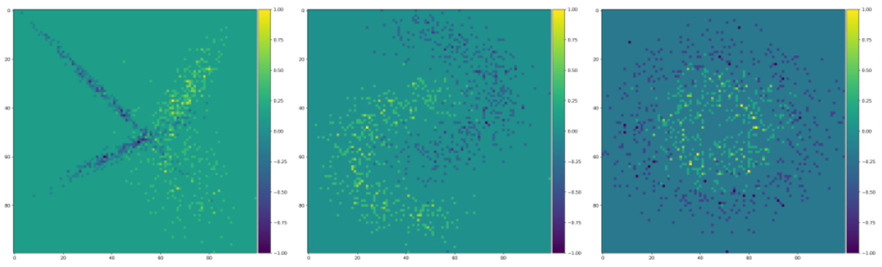

Figure 1: Synthetic datasets made out of 1000 points on a 100x100 grid.

the Laplacian is a compact operator. We are never interested in finding a solution in another space. The space V in the variational approach is an auxiliary space to solve the problem. Density relations and canonic injection allows then to make sure the solution makes sense in the original space.

In the machine learning approach, the shape of the solution is determined in advance and an optimization problem is defined. The solution is assumed to live in a finite dimensional space. The existence of a feature map ϕ from the original space to a higher dimensional space is assumed to exist such that the solution in the higher dimensional space is better than in the original space. The kernel is then selected to fulfill Mercer's theorem that gives a spectral decomposition of the kernel seen as a compact self-adjoint integral operator. This decomposition provides a proper inner product on the the higher dimensional space V such that the explicit representation of ϕ is not necessary. In other words, the space V plays a preponderant role in the resolution, and the spectral decomposition is used only to allow the *kernel trick*.

We mentioned that we do not need to assume or solve any optimization problem contrarily to the machine learning approach. In fact, there is an underlying optimization program that is solved by a direct corollary of Lions-Lax-Milgram theorem. Indeed, the theorem states that if $a(.,.)$ is symmetric, as is the Laplacian, then $\exists! u \in H,\ J(u) = \min_{v\in H} \frac{1}{2}a(v,v) - L(v)$. As $a(.,.)$ is a bilinear symmetric form, and assuming any basis, there exists a matrix A s.t. it is equivalent to $\exists! u \in H,\ J(u) = \min_{v\in H} \frac{1}{2}v^T Av - Lv$ [13].

Kernel-based methods solve a similar optimization program. Using the same notations, the program is defined by $\min_{v\in H} \frac{1}{2}v^T Qv - e^T v$ with $e = [1, ..., 1]^T$ and $Q_{ij} = y_i y_j K(x_i, x_j)$. Although its nature is the same, A dimension depends only on the number of eigenvectors kept in the basis while Q depends directly on the training set size. This formulation illustrates one major difference between the two kernels: the heat kernel depends on the dimension N which is the computational bottleneck of the physical approach. On the contrary, kernel-based SVM computational time is primarily influenced by the training size set.

5 NUMERICAL EXPERIMENTS

In this section, we study the property of the parameters of Algorithm 1 and compare it to some standard classification algorithms. We use a 10-folds cross-validation with a stratified shuffle sampling method. The value of β is selected by an exhaustive search for values in {-0.2, -0.1, 0, 0.1, 0.2 } on a validation set. We used three dataset generators with various difficulties. The datasets used to study the model parameters are shown in Figure 1. The first dataset generates two clusters per class whose barycenters are located on a randomly generated polytope. The second dataset is made out of two interleaving half circles with 20% noise (points that are flipped to belong to the opposite class). The third one is made out of two imbricated circular clusters, with 20% noise. Those datasets are non-linearly separable and present some difficulties that a modern machine learning algorithm must be able to tackle.

In the following sections, the solutions are displayed with their isolines from -1 to 0 by 0.1 step. The bold line represents the boundary for level 0 and the dashed line, the best boundary found by the algorithm. In case the best boundary is 0, only the dashed line is displayed. The points displayed on the solution are the test set.

5.1 Influence of model parameters

Eigenvalues number: Figures 2 and 3 shows the impact of the number of eigenvectors kept in the decomposition, for dataset 1 and 2. Low values allows only very regular and symmetric solutions in terms of level sets. Increasing the value allows the level sets and thus the decision boundary to be less regular as it increases the numerical approximation of the exact solution to the heat problem. In particular, in Figure 2, for five eigenvectors, it is impossible to distinguish between the two blue clusters, while for 20 and 50 there exists an isoline that clearly allows to distinguish between them. Above 20 eigenvalues, the gain in precision does not seem to increase the quality of the solution. The number of eigenvalues could even be lower depending on the datasets. For instance, on dataset 3, because of the axial symmetry through the center by any axis, less eigenvectors are requires to properly describe the solution. Conversely, more complex datasets, with many clusters, various cluster shapes may need even more eigenvectors to describe the solution.

Iterations number: Figures 4, 5 and 6 display the evolution of the solution depending on the number of iterations using 20 eigenvalues and α set to 1. For dataset 1 in Figure 4, the boundary in the area where the clusters intersect is pushed by decreasing the temperature of misclassified points of class -1. Two small isolated components appear after seven iterations in the *hot* area. After 10 iterations, those components are merged with the main cold area. However, as it leads to massively misclassifying hot points, the boundary is almost brought back to the initial one after 13 iterations. Notice however, that the isolines clearly display the shape of the clusters. Finally, after 18 iterations, the solution clearly displays the four clusters, including the parts across both classes. This oscillating behavior comes from the noise in the center: modifying the heat of some points leads some other to be misclassified, themselves being misclassified after the next iteration, and so forth. The value α is probably too high and in the future, we could use a decaying schema such that the total variation of heat on the source between iterations

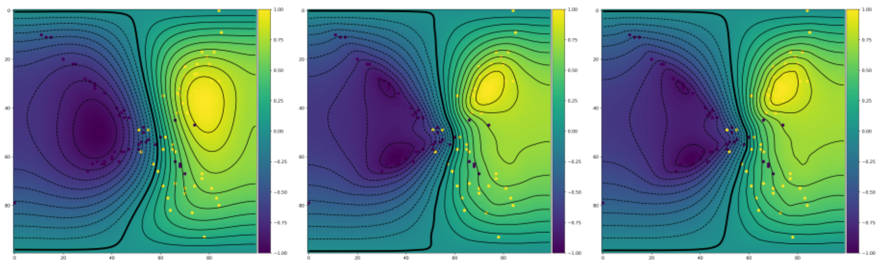

Figure 2: Solution for 5, 20 and 50 eigenvalues on dataset 1

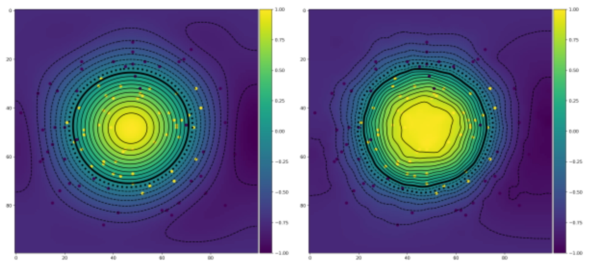

Figure 3: Solution for 5 and 50 eigenvalues on dataset 2

decreases iteration after iteration (similarly to the simulated annealing algorithm). Another possibility would be to solve the non-homogeneous problem such that the diffusion coefficient is not constant: increased if a point is correctly classified, decreased otherwise such that even if the heat is drastically increased and the point correctly classified at the next iteration, this heat will not be propagated to far around.

In Figure 5, after three iterations the solutions is still very close to the initial one (not displayed here). Contrarily to the first dataset, there exists a rather smooth boundary to separate the clusters and thus, there is no oscillation. Iteration after iteration, the isolines are torn to fit the clusters. The process is rather slow, most likely due to a low value for α.

Dataset 3 is probably the easiest one for the method due to the Laplacian properties. In Figure 6, we can observe that the final solution conserves its circular shape while adjusting the local characteristics of the training set to the point it clearly overfits. For instance, after 18 iterations, there is an irrelevant isolated component above the main isoline.

α **value:** Figures 7 and 8 display the solution depending on α after 10 iterations with 20 eigenvectors for dataset 1 and 2 (dataset is omitted for space reason). As expected, the solutions highly differ depending on the value of α. In particular, despite the solution provides in general good accuracy, the solution feels less fitting the training set topology.

On dataset 2, Figure 8, a higher value than 1 for α speeds up the training process, as displayed for $\alpha = 3$ and $\alpha = 5$. However, above a certain threshold, the solution will oscillate with the training iterations without guarantee that the solution will provide good results. Actually, α controls the radius of the neighborhood around a misclassified point that will be affected by the heat modification of this specific point. A too high value implies that a lot of neighbors will be misclassified, themselves misclassifying their neighbors in a sort of chain reaction. Empirically, 3 or 5 seems to provide the best results. Notice that α is different from the parameter σ of the Gaussian kernel. A given point might see its temperature changed several times in the process, with possible different values of α in case of adaptive schema. These temperature changes imply a local change in the radius of influence. On the contrary, the latter fixes a common radius of influence for all points on the domain. The value is not update during the optimization process.

5.2 Comparison to other methods

We compared our method to eight other methods provided by Scikit-Learn [14]: linear SVM, RBF SVM, Random Forest, Extra Tree, Gaussian Process for classification, Multilayer Perceptron, Gaussian Naive Bayes and Gradient Boosting. We used three dataset generators and a 10-fold cross-validation as mentioned above. However, to obtain really statistically robust results, we generated five datasets per generator, and then averaged the cross-validation scores to obtain the final result. As the datasets and the split for the cross-validation are balanced, the accuracy is a proper indicator of model performance. We also report the average standard deviation.

We used the default parameters for the algorithm implementation. For our method, we used 20 eigenvalues, 10 iterations and $\alpha = 1$ which we consider being the default configuration.

The results are provided in Table 1. As expected, Linear SVM provides the worst results as none of the dataset are separable. In general, the Laplacian approach presented here performs well on all datasets. It ranks first on dataset 3 which is not surprising knowing the properties of the Laplacian operator. On dataset 2, it ranks second, only 0.20% behind Gradient Boosting. The worst results are achieved on the first dataset, where it ranks 4th behind Gradient Boosting (-1.46%), Gaussian Process (-1.32%) and slightly after Extra Tree (-0.12%). In addition, the standard deviation belongs to the same range as the other good performing methods.

Those results are encouraging since the model performed well against some state-of-the-art methods s.a. Gradient Boosting or Extra Tree. Even more interesting, it provides better results than Random Forest or RBF SVM. Although the selected datasets propose some common difficulties in classification, additional validation is still needed. In particular, while the size of the training set has little impact on the execution time, the number of dimensions is the main bottleneck as computing the solution requires computing as many integral as there are dimensions.

6 CONCLUSION

In this paper, we assumed the confidence in a label is continuous across the space and behave as a diffusion model. A training dataset can thus be viewed as an initial distribution of heat such that solving the heat equation propagates the belief or information in the whole space. Numerical validation over three different

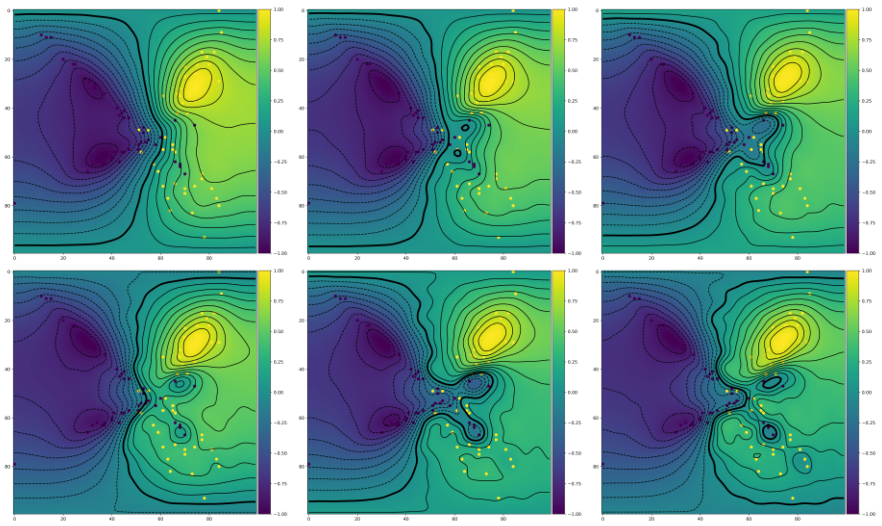
Figure 4: Solution for 3, 7, 10, 13, 18 and 25 iterations with 20 eigenvectors and $\alpha = 1$ on dataset 1

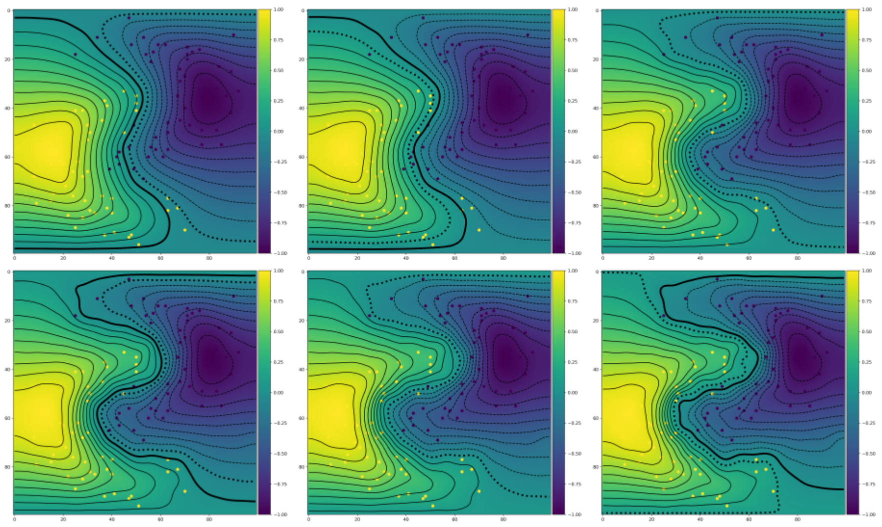
Figure 5: Solution for 3, 7, 10, 13, 18 and 25 iterations with 20 eigenvectors and $\alpha = 1$ on dataset 2

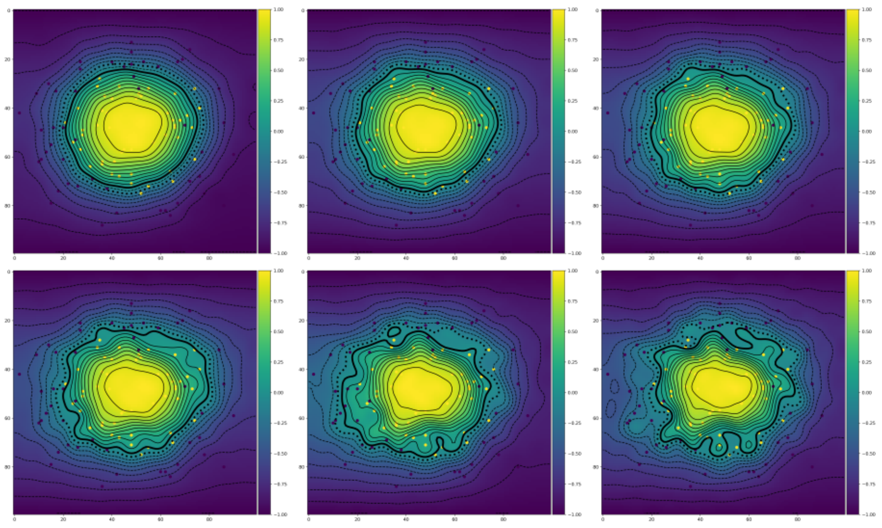
Figure 6: Solution for 3, 7, 10, 13, 18 and 25 iterations with 20 eigenvectors and $\alpha = 1$ on dataset 3

Table 1: Average accuracy obtained over five datasets for each dataset generator.

	Dataset gen. 1	Dataset gen. 2	Dataset gen. 3
Laplacian	0.8472 (0.035797)	0.9628 (0.020220)	**0.9020** (0.024278)
Linear SVM	0.3982 (0.433634)	0.4338 (0.480578)	0.2577 (0.449427)
RBF SVM	0.8338 (0.034809)	0.9292 (0.026755)	0.8526 (0.034984)
Random Forest	0.8140 (0.034044)	0.9074 (0.025235)	0.8648 (0.040679)
Extra Tree	0.8484 (0.030553)	0.9638 (0.019310)	0.8720 (0.030403)
Gaussian Process	0.8604 (0.030100)	0.9004 (0.016497)	0.9001 (0.032232)
Multilayer Perceptron	0.7730 (0.038215)	0.8794 (0.032538)	0.7428 (0.034757)
Gaussian Naive Bayes	0.8116 (0.034721)	0.8964 (0.032232)	0.8978 (0.030916)
Gradient Boosting	**0.8618** (0.026415)	**0.9648** (0.019082)	0.8920 (0.029064)

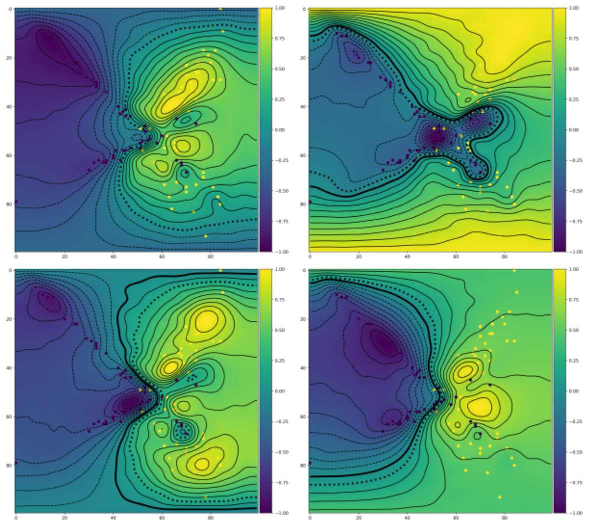

Figure 7: Solution after 10 iterations for values of α in $\{3, 5, 10, 25\}$ with 20 eigenvectors on dataset 1

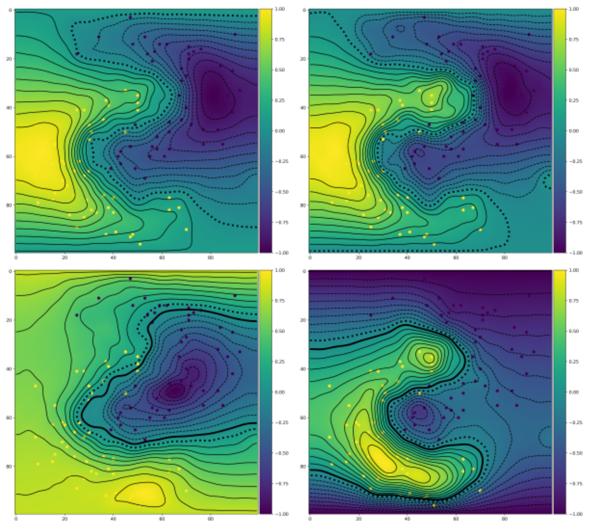

Figure 8: Solution after 10 iterations for values of α in $\{3, 5, 10, 25\}$ with 20 eigenvectors on dataset 2

types of non-separable datasets has shown the pertinence of such approach. The diffusion model managed to be as accurate as the state-of-the-art machine learning techniques such as Gradient Boosting or Extra Tree.

One advantage of adopting this approach is the well understood mathematics behind, in particular the proven existence and uniqueness of solution. Notice also that the method is purely deterministic. We do believe that leveraging robust mathematical models rather than traditional "learned" black-box models is a key to provide more transparent and explainable solutions to problems.

Beyond this proof of concept, several axes of improvement appear. First, we mentioned several ideas to select the correct parameter to speed-up the algorithm while preventing overfitting and solution oscillations between iterations: adaptive parameters and heat equation with non-homogeneous diffusion coefficient.

More theoretically, we do not provide a proof of convergence nor under which condition the solution converges to a perfectly classified training set. The intuition tells us the algorithm converges towards a cycle of distributions that depends on α. But it remains to be formally proved.

In this preliminary work, we studied only two dimension problems with a hypercube as space domain, although the theory adapts immediately to any dimension and pretty easily to other type of regular domains. Future work should study the scalability of the method in high dimensions. It is clear that for a given hypercube, the training set size does not affect the computation time, however, more dimensions imply more costly numerical integration. We hope to improve the scalability by using the different symmetries available.

Another axis to speed-up the method and increase the accuracy is the way to partition the space. We used a homogeneous grid of a given step, but it might be interesting to generate a more ad-hoc grid using e.g. a bins discretizer based on k-means or quantiles. Another approach could be to solve the problem using finite element method that automatically refines the space where it needs to be, according to the local density of training set points in the domain. Also, finite elements are usually faster due to their sparse support, especially for the heat equation.

REFERENCES

[1] G. Allaire. 2007. *Numerical analysis and optimization : an introduction to mathematical modelling and numerical simulation.* Oxford University Press, Oxford.

[2] N. Berline, E. Getzler, and M. Vergne. 2003. *Heat kernels and Dirac operators.* Springer Science & Business Media.

[3] H. Brezis. 2010. *Functional analysis, Sobolev spaces and partial differential equations.* Springer Science & Business Media.

[4] Y.-W. Chang, C.-J. Hsieh, K.-W. K. Chang, M. Ringgaard, and C.-J. Lin. 2010. Training and testing low-degree polynomial data mappings via linear SVM. *J. Mach. Learn. Res.* 11, Apr (2010), 1471–1490.

[5] D. R. Cox. 1958. The regression analysis of binary sequences. *J. R. Stat. Soc.* (1958), 215–242.

[6] F. Cucker and S. Smale. 2002. On the mathematical foundations of learning. *Bulletin of the American mathematical society* 39, 1 (2002), 1–49.

[7] F. Cucker and D. X. Zhou. 2007. *Learning theory: an approximation theory viewpoint.* Vol. 24. Cambridge University Press.

[8] J. C. Ferreira and V. A. Menegatto. 2009. Eigenvalues of integral operators defined by smooth positive definite kernels. *Integral Equations and Operator Theory* 64, 1 (2009), 61–81.

[9] I. Grattan-Guinness. 2005. Joseph Fourier, Théorie analytique de la chaleur (1822). In *Landmark Writings in Western Mathematics 1640-1940.* Elsevier, 354–365.

[10] T. Hofmann, B. Schölkopf, and A. J. Smola. 2008. Kernel methods in machine learning. *Annals of Statistics* (2008), 1171–1220.

[11] P. D. Lax and A. N. Milgram. 1955. IX. Parabolic Equations. In *Contributions to the Theory of Partial Differential Equations. (AM-33).* Princeton University Press, 167–190.

[12] Y. Lin. 2004. A note on margin-based loss functions in classification. *Stat. Prob. Lett.* 68, 1 (2004), 73–82.

[13] M. Mohri, A. Rostamizadeh, and A. Talwalkar. 2018. *Foundations of machine learning.*

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12 (2011), 2825–2830.

[15] F. Rosenblatt. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.* 65, 6 (1958), 386.

[16] V. Vapnik. 2013. *The nature of statistical learning theory.* Springer Science & Business Media.

[17] C. KI Williams and C. E. Rasmussen. 2006. *Gaussian processes for machine learning.* Vol. 2. MIT press Cambridge, MA.