Value of Information in a Support Vector Machine, an Exploration

Bachelor of Science in Economics, Management and Computer Science

Supervisor:
Professor Borgonovo Emanuele

Student:

Giancola Simone Maria

3074413

Academic Year 2020-2021

ACKNOWLEDGEMENTS

There are many individuals to which I am immensely thankful for this first step in higher education, I hope that this page will be at the same time sufficient and condensed enough to fit in the space allotted for it.

First of all, I would like to express my gratitude to my supervisor Professor Borgonovo Emanuele. He has been a key individual in my growth as a student in the past 3 years. Always available and full of ideas, he again proved to be a wonderful course director. Out of all the exceptional means of learning he provided, he also managed to set up a team for this piece of work.

Coming back to this team, Professor Plischke Elmar proved to be a great thinker, contributing with solid ideas to the exploration we carried out since January. Together with these two academic entities, I also have to thank Airoldi Federico, a clever, young and kind person, that just graduated and has chosen to get a job, but nevertheless found time to help and collaborate with myself and the other members of the team. His advanced knowledge in programming principles and languages has definitely been of pivotal importance for the progression of the work, and will be as long as it goes on.

Again on education, I also want to mention professor Maccheroni Fabio and Lijoi Antonio, two Bocconi Professors. Thank you for providing me with a solid knowledge and a strong interest for science.

On the family side, I exploit this opportunity to publicly thank my parents. Both have always pushed me forward in my education and personal development, often giving up something themselves. I also wish to mention personally each and every family member of a great and closely-knit group of people, that never runs away from each other. A great thank you to my grandmother, my grandfather, who taught me most of the things I know out of academics, my brother, aunts, uncles, and cousins.

Last (but not least) I really want this to be the first and not last moment I will also thank my friends. Thank you to the guys I know since we were 6, with whom I grew up, sharing many moments I will never forget.

Thank you to my rugby teammates, for struggling together each training session & match, always striving for winning, as winning is the only thing that matters in the end. I hope I will find the strength to restart after this awful historical moment that kept us out of the field.

Thank you my dear University friends, we had struggles and fun together. Even though we were in the same city for roughly one year and a half, we bonded, and found the same hunger for personal success in each other.

In & out of these diverse groups of people, I would like to give a special alphabetical mention as I did with family to those that have always been closer, constantly and indirectly pushing me forward in pursuing anything, teaching me, like my family, how most of the things are done not for yourself but for someone else. Thank you Andrea D.L., Chiara M., Gabriele Z., Gianluca F., Lorenzo C., Lorenzo D.N., Lorenzo R., Marta B., Nunzio F., Roberto C., Vittorio H.

I hope the future will hold more and more for everyone of you, and for myself.

Contents

Chapter 1

Introduction

This document aims to provide an overview on Support Vector Machines, Value of Information and a potential link between the two, together with some considerations on what is needed to accomplish it.

Purpose

The first two chapters explore concepts orbiting around Support Vector Machines and Value of Information. This is done mainly to provide a thorough understanding to anyone accessing this thesis, and especially with regard to the mathematical procedures implemented.
The last chapter is an explanation of what was obtained after a series of working sessions, held between the contributors of this project.
Creating such a system would be beneficial to each analysis pipeline that implements the algorithm at a corporate level, as the Value of Information is a key indicator of which elements eventually drive a result. It turns out to be even more important when business choices have to be made. In fact, differently from other methods[1], it focuses on existing dimensions and thus returns a more applicable result.

[1]Such as Principal Component Analysis, not included in this document

Methods

For what concerns the first two chapters, a heavy theoretical approach is proposed. When focusing on Support Vector Machines, many sources were exploited, ranging from publications to advanced lectures from accredited sources (when cited, a link to the pdf is provided in the bibliography). The chapter about Value of Information is a detailed overview made easier for beginners, authored by the two academic contributors and other collaborators.

The attempt to merge the two topics arose from a joint work that relied on the experience and collaboration of all the contributors, thus being more practical and experimental. It is paired with a code repository available upon request.

This piece of work evolved during the study, resulting in a series of scripts that together compute what is desired. Along with the programming elements, automatically saved graphs of the results are stored. It is not necessary for the purpose of understanding the topic, but rather to show how it is implemented at a coding level.

Sources

The usual format was followed, with numbers hyperlinking to the bibliography, automatically generated by the Zotero platform[2]. In addition to the academic and citable documents, it is worth mentioning that websites were exploited to grasp an introductory understanding of some topics. Wikipedia[3] is a good source for basic mathematical concepts or standard notation, and is well indexed in terms of web searching. Wolfram MathWorld[4] is another exceptional and more rigorous knowledge provider. The discussion Platforms on the Stack Exchange Environment[5], allow users to interact, and are the missing piece in this mosaic of internet websites for studies. All of the above websites were used consistently, thus I judged them worth mentioning.

In addition to the information obtained from external documents, it is evident that many

[2] www.Zotero.org
[3] www.Wikipedia.org
[4] www.WolphramMathWorld.com
[5] www.StackSites.com

of the methodologies and notations used in this production were possible thanks to the great input given by my Bachelor studies.

Chapter 2

Support Vector Machines

Support Vector Machines (often referred to as *SVMs*) are powerful tools for both classification and regression. In the following chapter, the classification version will be briefly introduced. This is an adaptation of a very insightful and advanced source of information [2].

Basic Definitions

Definition 2.1 (Euclidean Space)**.** *A Euclidean Space is a Vector Space* $\mathcal{X} \subseteq \mathbb{R}^d : d < \infty$ *equipped with an inner product denoted as* $\langle x, x' \rangle \, \forall x, x' \in \mathcal{X}$[1].

Definition 2.2 (Inner Product)**.** *An operation* $\langle x, x' \rangle$ *inside an Euclidean Space* $\mathcal{X}$ *is an inner product if it is an operation* $\langle \cdot, \cdot \rangle : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *such that* $\forall a \in \mathbb{R}, \quad \forall x, x' \in \mathcal{X}$ *:*

1. *Linearity*

$$a \langle x, x' \rangle = \langle ax, x' \rangle \tag{2.1}$$

2. *Symmetry*

$$\langle x, x' \rangle = \langle x', x \rangle \tag{2.2}$$

[1] One of the main papers cited for this chapter engages in a small digression on how the notions of Hilbert and Euclidean are interchangeable up to some point. For the sake of this production, focused on a dataset with a limited number of features d, we refer to Euclidean space as a finite space which can take up any dot product, and not only the canonical one for $\mathbb{R}^d$

3. Positive Definiteness

$$\langle x, x\rangle \geq 0 \qquad \langle x, x\rangle = 0 \Longleftrightarrow x = 0 \tag{2.3}$$

Intuitively, an inner product is a notion of similarity between two datapoints

Definition 2.3 (Norm in Euclidean Space)**.** *For every Euclidean Space equipped with an inner product the latter induces the existance of the norm which is defined as:*

$$||x|| = \sqrt{\langle x, x\rangle} \Longleftrightarrow ||x||^2 = \langle x, x\rangle$$

Definition 2.4 (Linearly Separable Data)**.** *Two Euclidean spaces with the same inner product $\mathcal{X}_{-1}, \mathcal{X}_1 \subseteq \mathbb{R}^n$ are linearly separable in $\mathbb{R}^n$ if*

$$\exists w \in \mathbb{R}^n,\ b \in \mathbb{R} \quad : \quad \langle w, x\rangle < b\ \forall\ x \in \mathcal{X}_{-1} \quad \langle w, x\rangle > b\ \forall\ x \in \mathcal{X}_1$$

Or, in other words, there is a hyperplane $f(x) = \langle w, x\rangle + b = 0$ that perfectly distinguishes the two spaces.

Problem Definition

In a real setting, given a dataset made of the tuples of data samples and labels:

$$\mathcal{D} \coloneqq \left\{ \{x_i, y_i\} \quad : i \in \{1, \ldots, n\} \quad x_i \in \mathbb{R}^d, \quad y_i \in \{1, -1\} \right\}$$

It is possible to identify two classes defined by the value of $y_i \forall i$. SVMs can be used to generate a solver for the problem of classifying unseen elements of this type, once properly set up. The result will be a function $\hat{y} = g(f(x))$ such that:

$$f(x) > 0 \Longrightarrow \hat{y} = 1 \tag{2.4}$$

$$f(x) < 0 \Longrightarrow \hat{y} = -1 \tag{2.5}$$

Linearly separable data

Given the existance of a hyperplane distinguishing the two classes, the most robust one will be oriented such that the distance from both sets is maximized [6]. This statement is simply proved by the fact that it identifies a separator which will hardly suffer from perturbations and datapoints added.
The distance between a random point x^* and the hyperplane $f(x) = \langle w, x\rangle + b$ such that $\langle w, x\rangle = \sum_i w_i x_i$ is:

$$d(x^*) = \frac{|f(x^*)|}{||w||} = \frac{|\langle w, x^*\rangle + b|}{||w||} \tag{2.6}$$

Considering for simplicity the sets $\mathcal{X}_{-1}, \mathcal{X}_1$, identified by the value of $y_i \forall i$, to maximize the margin the distances from the nearest datapoints of the two classes have to be maximized. The two are denoted as d_+ at x_+ and d_- at x_- and the margin will eventually be:

$$M = d_+ + d_-$$

By the fact that $\forall c \in \mathbb{R} \quad f(x) = cf(x) = 0$, it is possible to choose a normalization such that $f(x_+) = 1$ and $f(x_-) = -1$.
We will thus have that a good classifier has for all the training points:

$$\langle w, x_i\rangle + b \geq +1 \quad \forall x_i \in \mathcal{X}_1 \tag{2.7}$$

$$\langle w, x_i\rangle + b \leq -1 \quad \forall x_i \in \mathcal{X}_{-1} \tag{2.8}$$

Which can be seen as follows:

$$y_i f(x_i) - 1 = y_i(\langle w, x_i\rangle + b) - 1 \geq 0 \quad \forall i \tag{2.9}$$

Using Definition 2.4 in the most comfortable way possible.
In terms of margin, it will be that:

$$M = d_+ + d_- = \frac{|1|}{||w||} + \frac{|-1|}{||w||} = \frac{2}{||w||} \tag{2.10}$$

For these reasons, the solution is identified by the following constrained optimization problem of a quadratic function[2][2].

$$\min_{w\in\mathbb{R}^d} ||w||^2 \quad s.t. \quad y_i(f(x_i)) - 1 \geq 0 \forall i \tag{2.11}$$

Lagrangian Formulation, Dual, KKT Conditions

Being that the equations above correspond to a constrained optimization problem, we can formulate a solution exploiting the Lagrangian equivalent of it. For sake of completeness, the main theorems to extract this link are reported below. A rigorous approach is found in Appendix A of [3]

Theorem 2.1 (Lagrangian Sufficiency)**.** *Given a function $f : \mathcal{X} \subseteq R^d \to \mathbb{R}$, a minimization problem $\min_x f(x)$ subject to constraints $h(x_i) \geq b$ and a Lagrangian function:*

$$\mathcal{L}(x, \lambda) = f(x) + \langle \lambda, (b - h(x)) \rangle \tag{2.12}$$

Where $x \in \mathcal{X} \subseteq \mathbb{R}^d$, $\lambda \in \mathbb{R}^n$, $\lambda_i \geq 0 \forall i$, $h : \mathcal{X} \to \mathbb{R}$.
If $x^* \in \mathcal{X}$ and $\lambda^* \in \mathbb{R}^d$ are such that:

$$x^* = arg\min_{\mathcal{X}} \mathcal{L}(x, \lambda^*) \quad h(x^*) \geq b \Longrightarrow x^* = arg\min_{\mathcal{X}} f(x) \tag{2.13}$$

Proof

$$\min_{x\in\mathcal{X},\ h(x)\geq b} f(x) = \min_{x\in\mathcal{X},\ h(x)\geq b} f(x) + \langle \lambda^*, (b - h(x)) \rangle \tag{2.14}$$

$$\geq \min_{x\in\mathcal{X}} f(x) + \langle \lambda^*, (b - h(x)) \rangle \tag{2.15}$$

$$= f(x^*) + \langle \lambda^*, (h(x^*) - b) \rangle \tag{2.16}$$

$$= f(x^*) \tag{2.17}$$

[2]The shift from norm to squared norm is implemented to yield a unique and stable solution. The trade-off is that it will be more robust.

In order, 2.14 stems from the fact that we are adding 0 *from the following implications by arguments in [5]*

- *if* $h(x_i) > b_i$ *the constraint is inactive and* λ^* *can be set to* 0
- *if* $h(x_i) = b_i 0$ *then the item is* 0. *Being that the solution is on the boundary, we will also have that the gradients are in the same direction, otherwise we could decrease* f *and see* $b - h(x)$ *increasing above* 0. *This can be seen as* $\nabla f = \nabla((b - h(x)))$

2.15 is verified since we take away a constraint and thus explore new possible values which could be smaller. 2.16 Holds by what we assumed at the beginning of the theorem. Eventually, 2.17 holds since $h(x^*) - b = 0$, *and we claim that it is the minimum of* $f(x)$ *subject to the constraints since all the requirements hold.*

In the SVM problem the Lagrangian is[3]:

$$\mathcal{L}(w, b, \lambda) = \frac{1}{2}||w||^2 + \langle \lambda, 1 - y(f(w, x))\rangle = \frac{1}{2}||w||^2 + \sum_i \lambda_i - \langle \lambda, y(f(w, x))\rangle \quad (2.18)$$

Minimizing with respect to w, b leads to the following conditions:

$$\mathcal{C}_1 = \begin{cases} \dfrac{\partial \mathcal{L}}{w_j} = w_j - \sum_i \lambda_i y_i x_{ij} \Longrightarrow w_j = \sum_i \lambda_i y_i x_{ij} \quad for \quad j = 1, \ldots, d \\ \dfrac{\partial \mathcal{L}}{b} = -\sum_i \lambda_i y_i = 0 \end{cases} \quad (2.19)$$

[3]The coefficient of $||w||$ is set to $\frac{1}{2}$ to make the derivation simpler. Being a minimization the value will not change as the function is just *stretched*.

It is possible to replace those conditions in the formulation of $\mathcal{L}(w, b, \lambda)$ to get rid of w, b and have a function dependent on λ only [2]. Going on in this direction:

$$\mathcal{L}(w, b, \lambda) = \frac{1}{2}||w||^2 - \sum_i \lambda_i y_i(\langle w, x_i\rangle + b) + \sum_i \lambda_i \tag{2.20}$$

$$=\frac{1}{2}||w||^2 - \sum_i \lambda_i y_i \langle w, x_i\rangle - \sum_i \lambda_i y_i b + \sum_i \lambda_i \tag{2.21}$$

$$= \frac{1}{2}||w||^2 - \langle\sum_i \lambda_i y_i x_i, w\rangle - b\sum_i \lambda_i y_i + \sum_i \lambda_i \tag{2.22}$$

$$= \frac{1}{2}||w||^2 - ||w||^2 - 0 + \sum_i \lambda_i = -\frac{1}{2}||w||^2 + \sum_i \lambda_i \tag{2.23}$$

$$= \sum_i \lambda_i - \frac{1}{2}\sum_i\sum_j \lambda_i\lambda_j y_i y_j \langle x_i, x_j\rangle = \mathcal{L}'(\lambda) \tag{2.24}$$

$\mathcal{L}'(\lambda)$ is said to be the dual[4] of $\mathcal{L}(w, b, \lambda)$, with the former being a minimization and the latter being maximized subject to the conditions $\lambda_i \geq 0 \forall i$, $\sum_i \lambda_i y_i = 0$. This last formulation is useful since the training datapoints are expressed in terms of inner products only. Indeed, if we set $\phi(w, b) \coloneqq \min_{w,b} \max_{\lambda_i \geq 0} \mathcal{L}(w, b, \lambda)$ and $\psi(w, b) \coloneqq \max_{\lambda_i \geq 0} \min_{w,b} \mathcal{L}(w, b, \lambda)$, it is easy to see that $\phi(w, b)$ is the setup used in the primal approach and $\psi(w, b)$ in the dual one. In a general setting it holds that $\max\min(f) \leq \min\max(f)$, but in this case they will exactly coincide, since it is a convex problem, as claimed in [7].

A more general formulation is proposed in the following theorem.

Theorem 2.2 (Karush - Khun - Tucker (KTT) conditions)**.** *The minimization solution to the above stated problem* $\min_{x,\lambda} \mathcal{L}(x, \lambda)$ *is obtained by solving:*

$$\frac{\partial \mathcal{L}}{\partial x_j} = 0 \quad for\ j = 1, \ldots, d \tag{2.25}$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0 \tag{2.26}$$

$$\lambda_i \geq 0 \quad for\ i = 1, \ldots, n \tag{2.27}$$

$$h(x_i) - b \geq 0 \quad for\ i = 1, \ldots, n \tag{2.28}$$

$$\lambda_i(b - h(x_i)) = 0 \quad for\ i = 1, \ldots, n \tag{2.29}$$

[4]Here equality is not exact, as explained later, the derivation is correct but some constraints were enforced.

Non Linearly Separable Data

Assuming that in the given dimensionality d of the dataset there is no linear separation of the classes, it is possible to show that SVM can be slightly modified to return a meaningful solution.

Slack Variables

If the datapoints are not efficiently distinguishable by a straight line that maximizes the margin, it is possible to introduce values $\xi_i \geq 0$ that allow for margin misclassification. The literature refers to them as *slack* variables. Given that the classification is implemented by $sgn(f(x))$, setting the new constraints as $y_i(f(x_i)) - 1 + \xi_i \geq 0$ a misclassification will hold whenever $\xi_i > 1$, or no contribution will be added[5]. A new cost function is thus introduced in the Lagrangian which becomes:

$$\mathcal{L}(w, b, \xi, \lambda) = \frac{1}{2}||w||^2 + C\sum_i \xi_i - \sum_i \lambda_i \Big(y_i(f(x_i)) - 1 + \xi_i \Big) - \sum_i \eta_i \xi_i \tag{2.30}$$

Where the penalty is $C\sum_i(\xi_i)^k$, with a relative cost of C, and $k = 1$ which guarantees that the problem is a quadratic programming problem [2]. The lagrangian variables η_i are introduced to enforce the further constraint that $\xi_i \geq 0 \forall i$. Previous derivatives are the same, and to those we add:

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = C - \lambda_i - \eta_i = 0 \Longleftrightarrow \eta_i = C - \lambda_i \Longrightarrow C - \lambda_i \geq 0 \; by \; \eta_i \geq 0 \tag{2.31}$$

[5]By $\xi_i \geq 0$ if $\xi_i \leq 1$ the classification of the algorithm is still in line with y_i. If $\xi_i > 1$ the classification is incorrect and thus a mistake is made.

Expanding the result with the previous tricks and what was added in this new primal form:

$$\mathcal{L}'(\lambda) = \sum_i \lambda_i - \frac{1}{2}\sum_i \sum_j \lambda_i \lambda_j y_i y_j \langle x_i, x_j \rangle - \sum_i \lambda_i \eta_i + C\sum_i \xi_i - \sum_i \eta_i \xi_i \tag{2.32}$$

$$= \sum_i \lambda_i - \frac{1}{2}\sum_i \sum_j \lambda_i \lambda_j y_i y_j \langle x_i, x_j \rangle - \sum_i \lambda_i \eta_i + C\sum_i \xi_i - \sum_i (C - \lambda_i)\xi_i \tag{2.33}$$

$$= \sum_i \lambda_i - \frac{1}{2}\sum_i \sum_j \lambda_i \lambda_j y_i y_j \langle x_i, x_j \rangle \tag{2.34}$$

To maximize with constraints:

$$\begin{cases} 0 \leq \lambda_i \leq C \\ \sum_i \lambda_i y_i = 0 \end{cases} \tag{2.35}$$

Where the first condition summarizes the constraint on λ_i and the last implication of equation 2.31.

Feature Mapping through Kernels

In addition to margin misclassification, it could be beneficial to map the data-points to a higher space. Ideally, there is always a dimension where, if the inputs are mapped, then there is a hyperplane perfectly distinguishing the classes. This is proven by a theorem from Cover[4].

However, mapping to a higher dimension is not always beneficial. As the number of features increases, the time required to work out the calculations explodes. For these reasons, it is necessary to exploit specific feature mapping functions that allow datapoints in $\mathbb{R}^d$ to be mapped to a bigger space, together with kernel functions that *simplify* the notion of inner product in that space for that particular case. The easiest example of kernel is the quick formula $a^2 + 2ab + b^2$ for the calculation of $(a+b)^2$. A kernel is nothing but a shortcut for calculations.

Definition 2.5 (Feature Mapping)**.** *A feature mapping is a function* $\phi : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}^D$ *where usually* $D >> d$

Definition 2.6 (Kernel Function)**.** *A kernel function is a mapping such that the inner product of a higher dimensional feature mapping ϕ is worked out efficiently. Namely:*

$$\mathcal{K}(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \tag{2.36}$$

It is not needed to know what kind of operations the feature mapping actually computes as the kernel returns an answer by implicitly using it.

Mapping to a higher dimensional space is beneficial. This is easy to infer thanks to the following theorem. In a nutshell, it proves that, with high probability, if the dimension is high enough, a set of datapoints is linearly separable.
The original statement was formulated in [4], while an easy proof is provided in [9] and reported in an straightforward case to give an idea.

Theorem 2.3 (Cover's Theorem)**.** *Given $p+1$ points labelled with two classes in $\mathbb{R}^d$ arbitrarily the number of partitions that are linearly separable by a $d-1$ dimensional plane in $\mathbb{R}^d$ are:*

$$C(p, d) = 2 \sum_{i=0}^{d-1} \binom{p-1}{i} \tag{2.37}$$

And if $p = d$ this means that all possible partitions can be split with a suitable hyperplane.

Proof (by induction)

The number of such partitions is denoted by $C(p, d)$. Considering $C(p, d)$ and an added point there are two cases:

1. *A separating plane passes through the new point. It suffices to infinitesimally shift it away to make the classes separable. It could belong to either one of the two classes depending on the shift.*
2. *No separating plane passes through the point, returning a single new feasible partition for this new set of points*

Thus, when evaluating $C(p+1, d)$ it will be that solution 1 is counted twice and solution 2 is counted once. This is achieved by considering $C(p, d)$ for 2 since nothing will have changed in terms of feasible splits, and $C(p, d-1)$ for 1 since there is a constraint on the

plane passing through the introduced datapoint. This identifies the recursion:

$$C(p+1,d) = C(p,d) + C(p,d-1) \tag{2.38}$$
$$= C(p-1,d) + 2C(p-1,d-1) + C(p-1,d-2) \tag{2.39}$$
$$= \binom{p}{0}C(1,d) + \binom{p}{1}C(1,d-1) + \dots \binom{p}{p}C(1,d-p) \tag{2.40}$$
$$= 2\sum_{i=0}^{d-1}\binom{p}{i} \tag{2.41}$$

Where in 2.40 the recursion was carried out until all terms were exhausted, and in 2.41 the properties $C(1,k) = 0$ if $k < 1$ and $C(1,k) = 2$ if $k \geq 1$ allow for the final representation, in which i increases until $C(1,d-p)$ is meaningful.
This relation, by substituting $p+1 = p\prime$ and considering the easy case in which $p\prime = d$ states that the number of feasible partitions is:

$$C(p\prime,d) = 2\sum_{i=0}^{d-1}\binom{p\prime-1}{i} \xrightarrow{p\prime=d} 2\sum_{i=0}^{p\prime-1}\binom{p\prime-1}{i} \xrightarrow{\sum_{i=0}^{n}\binom{n}{i}=2^n} 2(2^{p\prime-1}) = 2^{p\prime} \tag{2.42}$$

And, given $p\prime$ points the exact number of binary partitions is $2^{p\prime}$, making all of them linearly separable.

This theorem proves that every binary dataset is linearly separable at a dimension equal to its size. This is rarely done for many reasons, including the following:

1. Computationally explosive as $n \to \infty$
2. ϕ is not known in advance
3. Most of all, d, D are known, but which specific kernel $\mathcal{K}$ to use to map $d \to D$ is not.

A more efficient path, pursued due to its great flexibility, is to exploit a Radial Basis Function (RBF) kernel.

Definition 2.7 (RBF kernel)**.** *Given a $\gamma = \dfrac{1}{2\sigma^2}$ spread parameter, an RBF kernel is a function:*

$$\mathcal{K}_{RBF}(x,x') \coloneqq e^{-\gamma||x-x'||^2} \tag{2.43}$$

The potential of such a kernel is harnessed by its peculiarity of mapping to a feature map of infinite dimension.

Theorem 2.4 (RBF is an infinite dimensional kernel)**.** *$\mathcal{K}_{RBF}$ is a kernel for a feature mapping*

$$\phi : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}^\infty$$

Proof

$$\mathcal{K}_{RBF}(x, x') = e^{-\gamma||x-x'||^2} = exp\Big[-\gamma\Big(||x||^2 + ||x'||^2 - 2\left\langle x, x' \right\rangle \Big)\Big] \tag{2.44}$$

$$= kost \cdot exp\Big[2\gamma \left\langle x, x' \right\rangle \Big] \qquad with\ kost \coloneqq exp(-\gamma(||x||^2 - ||x'||^2) \tag{2.45}$$

$$= kost \prod_{i=1}^{i=2\gamma} exp\Big[\left\langle x, x' \right\rangle \Big] \qquad and\ by\ e^{f(x)} = \sum_{n=0}^{\infty} \frac{f(x)^n}{n!} \tag{2.46}$$

$$= kost \prod_{i=1}^{i=2\gamma} \sum_{n=0}^{\infty} \frac{\langle x, x' \rangle^n}{n!} = kost \prod_{i=1}^{i=2\gamma} \sum_{n=0}^{\infty} \frac{\mathcal{K}_{POLY(n)}(x, x')}{n!} \tag{2.47}$$

Where the last identity is a valid kernel since sums and products of kernels are indeed kernels (see this lecture for a proof[8], not reported here). Thus RBF is a valid kernel mapping to an infinite dimension the datapoints in a finite (& efficient) time.

The potential of such an infinite dimensional kernel is reached once a value of γ is validated and an infinite dimensional mapping is made. As shown in Theorem 2.3, all the possible partitions are separable for p points in p dimensions. As a consequence, increasing the dimension to ∞, they will again be separable, as the amount of information that can be stored along the dimensions is *infinitely* higher.

For high values of γ, the risk is over-fitting the training set, as there will always be a separation, since we are in an ∞ feature space. Enforcing a high γ might not generalize well on the test set. A trade-off must be established for the solution to be elastic enough. The similarity measure induced by a kernel can be directly exploited in a SVM, especially when data is not linearly separable, as explained in [2] and below.

All methods together

By noticing that in the dual formulation the inner product between each pair of datapoints is used, it could be useful to pre-compute and store these values inside a matrix.

Definition 2.8 (Gram Matrix in an SVM setting)**.** *Given* $\{x_1, \dots, x_n\}$ *datapoints and a notion of inner product* $\mathcal{K}(x_i, x_j)$*, a Gram Matrix* $\mathcal{G}$ *is the matrix defined by all the possible combinations of inner products:*

$$\mathcal{G} = \Big\{\mathcal{K}(x_i, x_j)\Big\}_{i,j=1}^{i,j=n} = \langle \phi(x_i), \phi(x_j) \rangle \; \forall i, j \tag{2.48}$$

The properties of $\mathcal{G}$ *are out of the scope of an SVM implementation such as the one of this document.*

After having chosen a Kernel $\mathcal{K}$ and precomputed $\mathcal{G}$, the final dual formulation for an SVM classification algorithm is of the form:

$$\max_{\lambda \in \mathbb{R}^n} \left(\sum_i \lambda_i - \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j \mathcal{K}(x_i, x_j) \right) \tag{2.49}$$

$$s.t. \begin{cases} 0 \leq \lambda_i \leq C \\ \sum_i \lambda_i y_i = \lambda^T y = 0 \\ \mathcal{K}(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \quad for \; \phi : \mathcal{X} \to \mathbb{R}^D \end{cases} \tag{2.50}$$

Or the corresponding KKT conditions.

The methods to solve these kinds of constrained optimization problems are many, and are out of the scope of this introductory explanation of the theory behind a Support Vector Machine. Section 5 of [2] explores the topic and provides insights on some of the approaches.

Further features, observations, and advancements to the technique have been explored in [2] and the works cited therein, but are usually not part of an introductory lecture on the topic like this one.

Chapter 3

Value of Information

This chapter approaches Value of Information (often referred to as *VoI*) with one of the many frameworks implemented in literature. It is a brief introduction focused on [1], detailing for a less experienced audience part of its results, and also aiming to provide enough theoretical knowledge to understand the usage in a real setting. Other approaches to tackle this topic are also outlined in the *related literature* section of [1]. To avoid confusion, the same notation is used, though very detailed and not introductory as the explanation.

Probabilistic Sensitivity Measures

Useful definitions to formalize the notions that follow is given below.

Definition 3.1 (σ algebra)**.** *Given a set X, a sigma algebra $\mathcal{F}$ is a collection of subsets $X_i \subseteq X$ such that:*

- $X \in \mathcal{F}$
- $X_i \in \mathcal{F} \Longleftrightarrow X_i^C \in \mathcal{F}$
- *if X_n is a sequence* $\Longrightarrow \bigcup_{i=1}^{d} X_i \in \mathcal{F}$

Definition 3.2 (Borel σ algebra)**.** *A Borel σ algebra is a sigma algebra generated by Open sets, or equivalently by closed sets.*

These definitions are foundational elements about the notion of collection of events to which probabilities can be assigned.
The framework takes into account a set of variables $X_1 \dots X_d, Y \in \mathbb{R}^{d+1}$ on a measure space $(\Omega, \mathcal{F}, \mathcal{P})$, where:

- Ω is a probability space where the events represented by $X_i \forall i$ *take place*
- $\mathcal{F}$ is a Borel σ algebra
- $\mathcal{P}$ is the set of probability measures on $(\Omega, \mathcal{F})$, or also a reference probability measure.

In this environment, a piece of information of the form $g(X_1 \dots X_d) = X$ such that $g : \mathbb{R}^d \to \mathbb{R}^k$ $k \leq d$ is given, and can be exploited to infer conclusions about the target denoted by Y. The aim of Value of Information is to understand the usefulness of the action consisting in gathering this knowledge, and conclude if the costs and benefit are worth the time or investment. In more straightforward terms, the issue to tackle is:

> If I spend c to gather X as information, will I end up knowing more than c relative to Y, holding a positive knowledge profit in terms of approximating my model of Y?

In order to answer this question rigorously, objects such as the probability measure $\mathbb{P}$, the cumulative distribution function F and the density f are defined for both X & Y and conditionals in the subscripts.
To formalize the notion of added value of knowledge, a separation measure between probability measures, which corresponds to a distance, is introduced. The separation measure ζ follows the definition of all distances, and is assumed to take place in the space of $\mathcal{P}$.

Definition 3.3 (Separation or Distance measure ζ)**.** *A function $\zeta : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ such that:*

- $\zeta(\mathbb{P}, \mathbb{Q}) \geq 0 \ \ \forall \mathbb{P}, \mathbb{Q}$
- $\zeta(\mathbb{P}, \mathbb{Q}) = 0 \Longleftrightarrow \mathbb{P} = \mathbb{Q}$

Where the two items are required properties of any distance, namely being always positive, and equal to zero when the two elements coincide.

Given a tool to evaluate the amount of difference between two distributions, the following definition is directly implied.

Definition 3.4 (Probabilistic Sensitivity Measure ξ)**.** *Given X knowledge about Y with a separation measure ζ the probabilistic sensitivity measure is:*

$$\xi_X^Y = \mathbb{E}[\zeta(\mathbb{P}_Y, \mathbb{P}_{Y|X})] \tag{3.1}$$

Which can be seen as exactly the average difference in knowing X rather than not having X when assessing Y

Basic Probability Theory properties have useful applications in this setting, as shown below with an example theorem.

Theorem 3.1 (Nullity implies Independence)**.** *A null Probabilistic Sensitivity Measure implies that the two distributions are independent, namely:*

$$\xi_X^Y = 0 \implies X \perp Y \tag{3.2}$$

Proof

By definition ξ_X^Y is the expected value of a separation measure. By definition it is also the case that $\zeta(\mathbb{P}_Y, \mathbb{P}_{Y|X}) \geq 0$ since it has the properties of a distance. In addition to that, the distance will be zero if and only if the distributions are equal. This means that the conditionality on X does not influence Y, which is an equivalent definition of independence. Mathematically:

$$\zeta(\mathbb{P}_Y, \mathbb{P}_{Y|X}) \coloneqq \begin{cases} \geq 0 \ \forall X, Y \\ = 0 \Longleftrightarrow Y = Y|X \implies Y \sim Y|X \implies Y \perp X \end{cases} \tag{3.3}$$

$$\implies \xi_X^Y = \mathbb{E}[\zeta(\mathbb{P}_Y, \mathbb{P}_{Y|X})] \coloneqq \begin{cases} \geq 0 \ \forall X, Y \\ = 0 \Longleftrightarrow \zeta(\mathbb{P}_Y, \mathbb{P}_{Y|X}) = 0 \implies Y \perp X \end{cases} \tag{3.4}$$

Useful examples to prove where these methods emerge in many sensitivity measures are thoroughly discussed in [1].

Value of Information through a Probabilistic Sensitivity Measure

After having pointed out the basic tools, a more realistic setting needs further elements to construe a complete landscape and eventually observe the appearance of Value of Information.

Definition 3.5 (Report a)**.** *In a setting such that an analyst has to infer conclusions from a dataset, a report is a quantity lying in a suitable space $a \in \mathcal{A}$ which is a thoughtful guess of a characteristic θ_Y of Y.*

Definition 3.6 (Scoring Rule S)**.** *A scoring rule is a function that evaluates the fitness of the report and the actual realization of the target. In other words there is $S : \mathbb{R} \times \mathcal{A} \to \mathbb{R}$ which takes a value $S(y, a)$ when a is reported and $Y = y$.*
Being that there are conditional distributions $Y|X$ and that the results have to be observable the added requirements are:

- *Existance of conditional expectations in terms of X*

$$\forall i,\ \{X_{i1}, \dots X_{ik}\} \exists \mathbb{E}[S(Y, a)|X],\ \max_{a \in \mathcal{A}} \mathbb{E}[S(Y, a)|X] \tag{3.5}$$

- *Existance of Expectation in terms of $Y \subseteq \mathbb{R}$ and $a \in \mathcal{A}$*

$$\forall Y \subseteq \mathbb{R} \exists \mathbb{E}[S(Y, a)],\ \max_{a \in \mathcal{A}} \mathbb{E}[S(Y, a)] \text{ where one of the maximizers is } a = \theta_Y \tag{3.6}$$

In this case S is said to be proper and strictly proper in case of a unique maximizer.

Applying the notions outlined above, it is possible to introduce Information Value in the standard way.

Definition 3.7 (Information Value ϵ)**.** *Given a target Y, knowledge X and a scoring rule S the information value is:*

$$\epsilon_X^S = \mathbb{E}\left[\max_{a \in \mathcal{A}} \mathbb{E}[S(Y, a)|X]\right] - \max_{a \in \mathcal{A}} \mathbb{E}[S(Y, a)] \tag{3.7}$$

Which can be seen as the greatest score improvement achieved by learning X [1] in terms of best reports.
Moreover, to ensure that the first term always exists, it has to be the case that the function $\chi \to \mathbb{E}[S(Y,a)|X=\chi]$ is continuous in $\chi \forall a \in \mathcal{A}$

Assuming that

$$Y \sim \mathbb{P} \ : \ S(\mathbb{P},a) \coloneqq \mathbb{E}[S(Y,a)] \tag{3.8}$$

$$Y|X \sim \mathbb{Q} \ : \ S(\mathbb{Q},a) \coloneqq \mathbb{E}[S(Y,a)|X] \tag{3.9}$$

It is then possible to evalute the score of a report in terms of a probability distribution and not only a random variable, defining them as equal to the expectation of the variable itself. Further, taking as example $\mathbb{P}$, the set of reports $a \in \mathcal{A}$ that maximizes $S(\mathbb{P},a)$ is denoted as[1] $a^S_{\mathbb{P}}$.
The equivalence of Probabilistic Sensitivity Measures ξ and Information Values ϵ arises exactly thanks to equations 3.8 and 3.9, when a proper Scoring rule S is chosen, as the following theorem outlines.

Theorem 3.2 (Correspondence of Information Value and Probabilistic Sensitivity Measure)**.** *ϵ^S_X with S proper is a Probabilistic Sensitivity Measure ξ^Y_X for Y with separation measure:*

$$\zeta^S(\mathbb{P},\mathbb{Q}) = S(\mathbb{Q},a^S_{\mathbb{Q}}) - S(\mathbb{Q},a^S_{\mathbb{Q}}) \tag{3.10}$$

Proof

$$\epsilon^S_X = \mathbb{E}\left[\max_{a\in\mathcal{A}} \mathbb{E}[S(Y,a)|X]\right] - \max_{a\in\mathcal{A}} \mathbb{E}[S(Y,a)] \tag{3.11}$$

$$= \mathbb{E}\left[\mathbb{E}[S(Y,a^S_{\mathbb{Q}})|X]\right] - \mathbb{E}[S(Y,a^S_{\mathbb{P}})] \tag{3.12}$$

$$= \mathbb{E}\left[\mathbb{E}[S(Y,a^S_{\mathbb{Q}})|X]\right] - \mathbb{E}\left[\mathbb{E}[S(Y,a^S_{\mathbb{P}})|X]\right] \tag{3.13}$$

$$= \mathbb{E}\left[\mathbb{E}[S(Y,a^S_{\mathbb{Q}}) - S(Y,a^S_{\mathbb{P}})|X]\right] \tag{3.14}$$

$$= \mathbb{E}[S(\mathbb{Q},a^S_{\mathbb{Q}}) - S(\mathbb{Q},a^S_{\mathbb{P}})] \tag{3.15}$$

[1]The same reasoning leads to the notation for $a^S_{\mathbb{Q}}$

In order, at equation 3.12 the definition just introduced is used, and it can be so thanks to the fact that S is proper. At equation 3.13 the Law of Iterated Expectation is exploited (i.e.$\forall X, Y \quad \mathbb{E}[\mathbb{E}[X|Y]] = \mathbb{E}[X]$), at equation 3.14 linearity of expectation (i.e. $\forall X, Y \quad \mathbb{E}[X] + \mathbb{E}[Y] = \mathbb{E}[X + Y]$ implies the result, and eventually at equation 3.15 it is the case that:

- $\mathbb{Q}$ *appears in both terms in the Scoring function since it is* $Y|X$
- *The law of iterated expectation allows for the removal of the condition on* X

This last result is nothing but the definition of Probabilistic Sensitivity Measure in accordance with definition 3.4. A derivation is briefly reported below:

$$\epsilon_X^S = \mathbb{E}[S(\mathbb{Q}, a_{\mathbb{Q}}^S) - S(\mathbb{Q}, a_{\mathbb{P}}^S)] = \mathbb{E}[\zeta(\mathbb{P}_Y, \mathbb{P}_{Y|X})] = \xi_X^Y \ : \ \zeta(\mathbb{P}_Y, \mathbb{P}_{Y|X}) = S(\mathbb{Q}, a_{\mathbb{Q}}^S) - S(\mathbb{Q}, a_{\mathbb{P}}^S) \tag{3.16}$$

Thanks to these derivations, further specifications reported in [1], and the literature mentioned inside, it is thus possible to evaluate the actual value of a piece of information when the purpose is understanding a target variable.
A proper analysis of an algorithm and its functioning should lead to the determination and design of a structure that returns such a result.

Chapter 4

Unifying the concepts: SVM and VoI

This chapter deals with a baseline attempt to calculate the Value of Information stored in a Support Vector Machine. While exploring the topic, many challenges arose, with the effect of slowing down the process. For these reasons, the common framework and some observations concerning how the classifier behaves are reported, aiming to establish the basis for a further improvement in the future. Nevertheless, the topics introduced in the previous two chapters are made coexistent, both theoretically and under a coding perspective. What is missing is a fine tuning of the whole system, to adjust it in the best way possible.
Given that it will be an analysis of the issues encountered, many aspects might not merge together, and will resemble independent digressions.

Setting a Common Ground

Once trained, an SVM classifier can be seen as a system that is fed with a test dataset and returns an answer, which is nothing but an estimated distribution of the result. Analytically, if we have a function f_{SVM} and data sets $\left\{X_{train}, X_{test}, Y_{train}, Y_{test}\right\}$, in first place f_{SVM} is oriented in terms of $\left\{X_{train}, Y_{train}\right\}$. Secondly, the function will be

able to estimate how X_{test} determines a tentative target distribution of Y_{test}. Namely:

$$f_{SVM}(X_{test}) = \widehat{Y_{test}} \implies \text{answer is } \sim \widehat{Y_{test}}, \quad \text{reality is } \sim Y_{test} \tag{4.1}$$

In case of a multidimensional dataset (i.e. $x_i \in \mathbb{R}^d$), it could happen that some dimensions are more important than others or, in other words, that some directions have more weight in the classifier's choice for a new sample. In the framework of a simple SVM, the target variable y_i for each sample x_i is a binary number, either positive or negative. An estimation of its value $\hat{y}$ will be the realization of what was presented in chapter two as the characteristic θ_Y and merging all terms in a vector results in $\widehat{Y_{test}}$.
In order to save resources, it could be beneficial to understand beforehand which directions store more information, and thus find the most important influencers in the result. This could be implemented by providing different SVM classifiers with different pieces of information $g(X_{train})$ and $h(X_{train})$, so to evaluate how much they separate in the estimated distribution with a suitable distance measure $\zeta(Y_g, Y_h)$. If the information provided obscured some dimensions, it would be possible to assess the impact of a direction in determining the final result. When not presented in training, the result is likely to be different, leading to a different classifier.

A Theoretical Simplified Attempt

In order to understand how the whole system could work together, it was assumed that one classifier named f_{SVM} was trained on the whole dataset, and that another one, named f_{svm} knew about fewer dimensions. The determination of their difference is the key point. The easiest case would be that of a two dimensional dataset ($d = 2$) with n samples, such that one dimension is informative and the other one is not. From this moment onwards, this will be referred to as *Rectangular Dataset*. An image for $n = 10000$ is shown below:

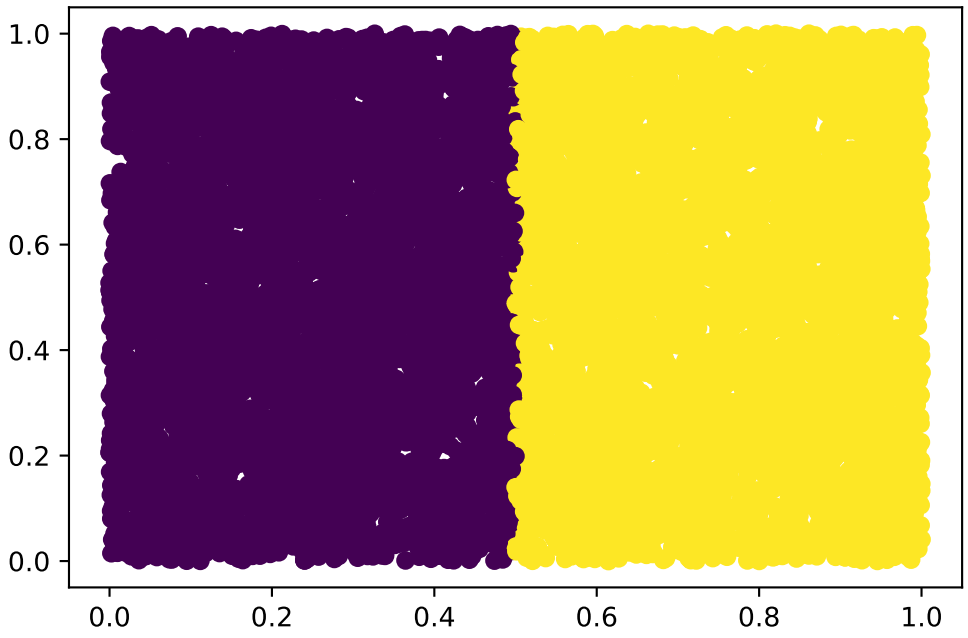
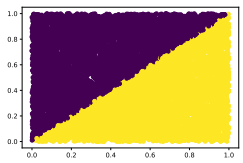
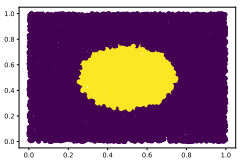


Figure 4.1: Rectangular Dataset

The *Diagonal Dataset* and the *Circular Dataset* are two further invented cases:



(a) Diagonal Dataset



(b) Circular Dataset

Figure 4.2: Sample Datasets for the first experiments

In the first case, a well functioning Value of Information analysis would suggest that only the dimension denoted as X_1 is informative, as X_2 is uniform in its shape, and the decision boundary could be evaluted for any sample as:

$$\forall x_i = (x_1, x_2)\ dec(x_1, x_2) = dec(x_1) \perp x_2\ \&\ dec(x_1) = \begin{cases} if\ x_1 \leq \frac{1}{2} \implies & y_i = 0 \\ else & y_i = 1 \end{cases} \tag{4.2}$$

To stabilize the calculation, it is possible to consider pieces (or better, strips) of the dataset for a single dimension (e.g. $d = 1$ and $X_{1a} = \{x_i \ : \ x_1 \in [0, 0.1]\}$ and so on). With these, a subclassifier f_{svm} is trained. Performances of the whole classifier f_{SVM}

and f_{svm} on the strip are compared. Averaging on all partitions of a dimension returns a result, which is a weighed comparison of all regions of data for an omniscient SVM, and a set of SVMs that are aware of partitions only. To even strengthen the evaluation, averages across every dimension will be observed for many different numbers of samples, to see how the values behave as $n \to \infty$. This said, an informal attempt would consist in the following:

Algorithm 1 VoI evaluation with strips of data

initialize $list_{datasetsize}$
for Each $datasetsize$ in $list_{datasetsize}$ **do**:
    recover $X_{train}, Y_{train}, X_{test}, Y_{test}$ with size $datasetsize$ in total
    $f_{SVM} = train(SVM_{empty}, X_{train}, Y_{train})$
    $VoI = []$ a matrix with $len(datasetsize)$ columns and a row per dimension
    for all dimensions **do**:
        create set of non overlapping substrips for Train and Test Data identified by the "sub" prefix
        $n = n_{substrips}$
        $VoI_{i^{th}\,dimension} = 0$
        for each substrip **do**
            $f_{svm} = train(SVM_{empty}, X_{subtrain}, Y_{subtrain})$
            $errors = f_{SVM}(X_{subtest}, Y_{subtest})$
            $suberrors = f_{svm}(X_{subtest}, Y_{subtest})$
            $VoI_{i^{th}\,dimension} += errors - suberrors$
        end for
        append $\dfrac{VoI_{i^{th}\,dimension}}{n}$ to VoI
    end for
end for
Compare trends for different dimensions as n increases (a (n, VoI) *plot*)
return VoI list for each value of n (a matrix)

While this method has some evident flaws and needs theoretical adjustment, it proved to be a good starting point to evaluate imminent criticalities in the realization of the project. Some of them will be explained in the following sections.

Concisely, for each sample size, firstly a classifier with *complete* knowledge is trained. In second place, the information set is split into many non-overlapping subsets along one of the dimensions, on which a different classifier is trained. This is done to compare the performance on the inner portion. These scores are gathered and averaged across one dimension, and appended to a list of information values for each direction that is the result of the algorithm. Ideally, as the inside classifier has specific knowledge, one would

expect that, as $n \to \infty$, the points will be enough to sufficiently train in both cases, and that the subclassifiers will perform better on the partitioned test set. The reasons for this claim can be summed up in the following points:

- **Saturation**

  Especially for simple datasets such as the one considered, it is expected that there will be a sufficient number of samples after which other points do not contribute to improving the classifier's training. If this number of points is reached in the subtrain dataset, then both classifiers will have saturated information and perform at their best. This ensures that f_{SVM} has no advantage on f_{svm} when considering the sample size used.
- **Overhead information of f_{SVM}**

  When a classifier *knows* about a specific strip, it is perfectly trained to respond to queries similar to that specific region. On the contrary, an omniscient classifier has wider and less specific knowledge, as it was trained on different information not necessarily needed.

Substrips Existance Conditions and Exceptions

One of the core problems is the abscence of a sufficiently big subtrain and subtest portion of data to achieve a meaningful result. Assuming that the information available is sufficiently big (i.e. $n \to \infty$) may not be enough. For instance, if data is generated randomly, even though the amount tends to infinity, there could be empty regions which are just too small, or with a single label in the train set. This happens if the condition of the split returns either zero datapoints or uniformly labeled datapoints. Thus, it becomes problematic to decide how to make the split as to make it function in each possible case, and simultaneously to avoid corner cases. This has been hypothetically achieved with the following choices:

- The number of strips, also referred to as bins in the code, depends on the total size but is limited

- Given that the dataset is enclosed in a square with vertices $[(0,0),(1,0),(0,1),(1,1)]$, the dimensions will range from 0 to 1 and the bins will consider equally lengthed rectangles that capture all the non-target dimensions and a $\frac{1}{n_{substrips}}$ length side of the target one.
- In the event that the train set has only one label, meaning that the subclassifier cannot train itself, the suberrors will be the number of errors that an hypothetical classifier always answering the label would make in the test set. In other words:

$$if\ Y_{subtrain} = [1,\dots,1] \Longrightarrow \widehat{Y_{subtest}} = [1,\dots,1] \tag{4.3}$$

$$\Longrightarrow suberrors = \sum_{i=1}^{n'} (1 - y_{subtest\ i})(mod 2) \text{ i.e. the number of different labels} \tag{4.4}$$

Thanks to these exception handlings, and a sufficiently large dataset, in most cases it is possible to evaluate the sought quantity with uniformly sized substrips. This leads to less granularity in the number of partitions, but prevents partitions from being so small that no datapoints belong to them.

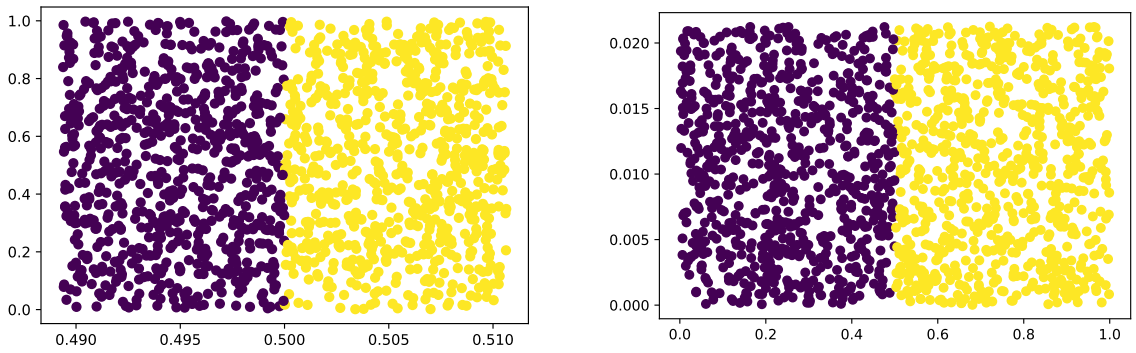
SVMs' Instability with *small* Datasets

Providing the algorithm with a thoughtful partition of data satisfies the sufficiency conditions of having information to work on. What comes after solving this issue can be directly linked to algorithm features that are used to train the classifier, which is not perfect and is subject to different weaknesses, and the way in which its parameters are tuned. Without going too deep inside which parameters a Support Vector Machine can have, Sklearn's documentation [10] mainly focuses on the kernel, gamma γ and the cost c. For the easy and explorative case, the cost was set to $c = 1$ and the kernel to RBF, which was introduced in definition 2.7, making the last parameter set to the default value of $\gamma = \frac{1}{n_{features}\sigma_X^2}$, where X is the train set.
In terms of the parameters' effect, it was already proved that a RBF kernel projects the datapoints in an infinitely dimensional space, where in particular the γ value determines the inverse of the influence radius of a single training point on the others, or more easily

how narrowly the *distance* bell of the infinite dimensional Gaussian curve is wrapped around the single datapoint. In the same chapter, from equation 2.30 onwards, the cost parameter (slack variable) is introduced in the perspective. Briefly, it is equivalent to the utility loss in misclassifying one datapoint of the train set, which allows for greater flexibility in the margin computation.

Although this is only one of the infinitely many available configurations, that ought to be explored after this single starting experiment, the expectations were very distant from the result obtained. More in detail, a strange instability in f_{svm} is found, in a rather easy case such as the rectangular dataset.

Thanks to the way in which the rectangular dataset is generated, when strips are extracted along the dimension denoted as X_1, they are *homogeneously labeled* apart from the one that includes the decision boundary of $X_1 = \frac{1}{2}$, while sampling partitions along X_2 returns *homogeneously shaped* subdatasets. Graphs are shown below for $datasetsize = 10000$, and have been proportionally adjusted to make them more user friendly, so the axis domains need to be observed[1].



(a) $X_{subtrain}$ at dimension X_1

(b) $X_{subtrain}$ at dimension X_2

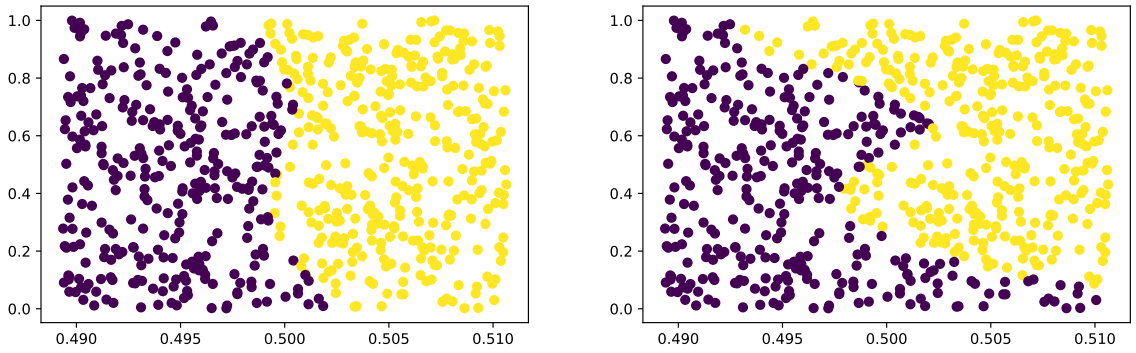
Figure 4.3: Training strips in both dimensions

The test set is obviously similar but less populated by points.

In a setting such as this one, given the potential of a powerful machine such as the SVM-RBF kernel, our expectation was that at least the subclassifier would have been as good as the complete classifier. On the contrary, what we found is that, when training is on a

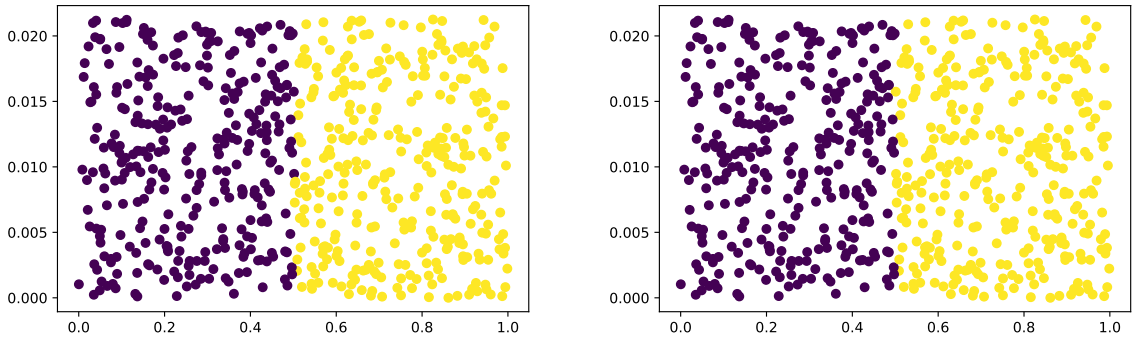
[1]The other strips of X_1 are corner cases with only one label, so the special treatment of the previous section is used.

strip, and the dimensions are stretched[2], the subclassifier performs worse making a split along X_1, learning a decision boundary which is not at all straight, as it can be noticed in the following images.



(a) f_{SVM} performance on test (b) f_{svm} performance on test

Figure 4.4: Dimension X_1



(a) f_{SVM} performance on test (b) f_{svm} performance on test

Figure 4.5: Dimension X_2

	f_{SVM} errors	f_{svm} errors	test size
X_1	25	102	660
X_2	2	0	639

Table 4.1: Dimensions/ Sizes matrix

The reasons for this strange behavior, with a classifier that is considered very powerful and flexible, especially with an infinite dimensional kernel, are yet to be explored. This *confusion* when domains are unbalanced could definitely be a factor to take into account,

[2]As in this case in which the subtraining dataset is no longer a square but a rectangle

but at the same time normalizing the dimensions would lead to losing the meaning of evaluating the strip *as is*, and adjust f_{svm} with a subdataset which has the same domain of its parent.

In addition to that, the best practice in a Machine Learning setting consists in attempting any operation with many different parameter combinations, to validate the mix that returns the best experimental result, and this is likely to be done as well in further explorations. It was agreed to allocate more time to study further the criticalities encountered.

Chapter 5

Conclusion

This last chapter gathers briefly and narratively what was reported in this document and attempts to provide the reader with an overview of how the project can be improved.

Summary

After an introduction to the topic and its methodology, the two main portions are a rigorous, publication-based first lecture about the topics of Support Vector Machines and Value of Information.

The former, a very famous classification[1] algorithm, is theoretically formulated, with digressions on tangent mathematical theorems and methods that support its strength and provide knowledge about its functioning. Being a very broad topic, many aspects are missing for sake of simplicity, among which efficient solving methods that are worth a dedicated document. Nevertheless, the advantage of programming languages is that this is done automatically and thus it can be avoided during a first experience.

The latter is a famous concept in many fields, ranging from physics to economics, approached with many different methods. One of these frameworks is proposed and explained.

Eventually, a procedure to extract an information measure from a Support Vector Ma-

[1] And Regression, but not in the scope of this specific explanation

chine is outlined, together with its challenges, and an interesting case in which such a powerful algorithm fails unexpectedly.

Current Concerns, Limitations, and Further Enhancements

Key points for a successful outcome will depend mostly on envisioning a meaningful procedure. This will consist in many adjustment steps that will also depend on what is found. First of all, a more rigorous workflow and design has to be attached to the current method. This will ensure a formulation consistent with the Value of Information. It is noticeable that a linear cost function of the errors might not be a correct distance measure between the two estimated distributions, or might at least need a proof to satisfy the sufficient conditions for that. It was chosen mainly to explore the problem in a directly observable way.
Secondly, as underlined in the prior chapter, parameter validation is fundamental to achieve a performance which is at least optimal in the range of parameters attempted. Again, this was not done due to the initial phase of the study.
Lastly, the instability that the solver has in narrow environments is to be understood and solved. Without a more precise subclassifier, which is to be expected given the *specific knowledge*, it is very difficult to assess the information stored in a dimension. This could imply that adjustments to parameters, transformations and further considerations might occur, together with a stronger literature analysis to identify if this was dealt with before.
This line of action is likely to require trial and error, formal study of the topics, and eventually time. In the event that the expected results were achieved it would lead to the creation of a tool that evaluates how SVM classifies from a different perspective, more oriented towards the final performance on a real dataset. This is a pivotal aspect of any application in the industry, as the final result is what drives the profit and ensures the sought success.

Bibliography

[1] Emanuele Borgonovo et al. "Probabilistic sensitivity measures as information value". In: *European Journal of Operational Research* 289.2 (2021), pp. 595–610. DOI: 10.1016/j.ejor.2020.07.010.

[2] Christopher J C Burges. "A Tutorial on Support Vector Machines for Pattern Recognition". In: (), p. 43.

[3] Costas Courcoubetis and Richard Weber. *Pricing Communication Networks: Economics, Technology and Modelling*. 1st ed. Wiley, Mar. 11, 2003. ISBN: 978-0-470-85130-2 978-0-470-86717-4. DOI: 10.1002/0470867175. URL: https://onlinelibrary.wiley.com/doi/book/10.1002/0470867175 (visited on 04/27/2021).

[4] Thomas M. Cover. "Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition". In: *IEEE Trans. Electron. Comput.* EC-14.3 (June 1965), pp. 326–334. ISSN: 0367-7508. DOI: 10.1109/PGEC.1965.264137. URL: http://ieeexplore.ieee.org/document/4038449/ (visited on 04/27/2021).

[5] Aaron Hertzmann, Marcus Brubaker, and David J. Fleet. *Lagrange Multipliers*. 2015. URL: http://www.cs.toronto.edu/~mbrubake/teaching/C11/Handouts/LagrangeMultipliers.pdf (visited on 04/27/2021).

[6] Aaron Hertzmann, David J. Fleet, and Marcus Brubaker. *Support Vector Machines*. 2015. URL: http://www.cs.toronto.edu/~mbrubake/teaching/C11/Handouts/SupportVectorMachines.pdf (visited on 04/27/2021).

[7] Michael I. Jordan. "1 SVM Non-separable Classification". In: (), p. 4.

[8] Roni Kardon. *More Kernels and their Properties*. 2008. URL: https://web.iitd.ac.in/~sumeet/CLT2008S-lecture18.pdf (visited on 04/27/2021).

[9] Emin Orhan. "Cover's Function Counting Theorem (1965)". In: (), p. 2.

[10] *sklearn.svm.SVC*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.