Hyperbolic Support Vector Machine and Kernel design

Aya El Dakdouki

École doctorale Sciences pour l'Ingénieur Université Lille Nord-de-France-072

THÈSE DE DOCTORAT

Discipline

Mathématiques appliquées

présentée par

Aya EL DAKDOUKI

MACHINE À VECTEURS DE SUPPORT HYPERBOLIQUE ET INGÉNIERIE DU NOYAU

Soutenue publiquement le 11 Septembre 2019 devant le jury composé de

Directeur de thèse:	Pr. Nicolas WICKER	Université de Lille 1, France
Co-directeur de thèse:	Pr. Yann GUERMEUR	CNRS, Inria-Loria, France
Rapporteurs:	Pr. Khalid BENABDESLEM	Université de Lyon, France
	Pr. Faïcel CHAMROUKHI	Université de Caen Normandie, France
Examinateurs:	Pr. Marianne CLAUSEL	Université de Lorraine, France
	Pr. Sophie DABO-NIANG	Univérsité de Lille, France

Contents

Remerciements

En tout premier lieu, je remercie le bon Dieu, Allah le tout puissant, de m'avoir donné la force pour survivre, la patience ainsi que le courage pour dépasser toutes les difficultés qui m'ont immensément guidé dans l'achèvement de cette thèse.

Un grand merci à mon directeur de thèse Monsieur Nicolas Wicker professeur à l'université de Lille, et à mon co-directeur Monsieur Yann Guermeur, Directeur de recherche au CNRS, affecté au LORIA INRIA à Nancy, qui m'ont accompagné pendant ces années de travail. Je vous suis reconnaissante pour l'opportunité de faire cette thèse que vous avez su m'accorder en me permettant de réaliser mon rêve, celui de devenir un docteur en mathématiques appliquées. Merci Nicolas, merci Yann, pour tout et notamment pour vos qualités humaines et scientifiques. Je ne pourrai jamais oublier votre gentillesse, votre patience, votre générosité, votre esprit de recherche, vos commentaires efficaces et votre soutien précieux au cours de ces années. Merci Yann pour les invitations à Nancy, cela a été un grand plaisir de travailler avec vous.

Je remercie ensuite l'ensemble des membres du jury, qui m'ont fait l'honneur de bien vouloir étudier avec attention mon travail: Khalid Benabdeslem et Faïcel Chamroukhi pour avoir accepté d'être rapporteurs de cette thèse. J'adresse mes remerciements à Marianne Clausel et Sophie Dabo-Niang pour avoir accepté de faire partie de mon jury de thèse en tant qu'examinatrices.

Ma gratitude va aussi à la fondation Walid Joumblatt pour les études universitaires de me donner l'attestation de bourse, et je remercie Monsieur Alaa Terro qui m'a facilité toutes les démarches administratives.

Je tiens à exprimer mes sincères remerciements à tous les professeurs qui m'ont enseigné à l'université Libanaise et à l'université de Nice Sophia Antipolis et qui par leurs compétences m'ont soutenu dans la poursuite de mes études.

Je remercie tout le personnel du Laboratoire surtout un grand merci aux membres des équipes Probabilités et Statistiques du laboratoire Paul Painlevé au sein duquel je me suis

épanoui durant ces années.

Je tiens à remercier les secrétariats Ludivine Fumery, Fatima Soulaimani, Soledad Cuenca pour l'ordre de missions et les informaticiens Sebastien Huart et Mohammed Khabzaoui. Je voudrais remercier aussi le secrétariat de direction Sabine Hertsoen, le secrétariat de Master 2 et de doctorat Aurore Smets et le responsable de l'imprimerie Frédérique Maréchal à leur gentillesse. Je profite de ces quelques lignes pour leur témoigner ma reconnaissance.

Je tiens à remercier M. Nicolas Wicker qui m'a donné l'opportunité pour assumer des charges d'enseignements au sein de l'université de Lille 1. Je remercie ensuite l'université de Lille 3 pour m'accepté pour un poste d'ATER pour enseigner pendant ma thèse qui était une expérience agréable et formidable.

J'adresse unn grand merci aux membres de l'université Lille 3 (UFR MIME) surtout avec les enseignants que j'ai enseigné avec eux, Mme Laurence Broze, M. Olivier Torres, M. Baba Thiam, Mme Aurore Lavigne, Mme Sophie Dabo, Mme Ophélie Guin, M. Aboubakar Amiri, M. Camille Sabbah, Mme Amélie Zill et Mme Mylène Maida.

Pendant cette thèse, j'ai eu le plaisir d'échanger dans la bonne humeur avec de nombreux enseignants chercheurs et avec de nombreux doctorants.

Ma gratitude va à mes amis en France pour leur soutien et pour tous les moments passés ensemble au cours de ces années, et j'adresse un grand merci à mes amis au Liban (Amal Hamieh, Jihan, Joyce, Hiba, Diala,...) pour leur encouragement et leur soutien moral...simplement pour leur amitié.

Je remercie chaleureusement ma famille, mes parents, mes frères Adel et Khalil et mes soeurs Nadima et Nour et mes grands-parents qui m'ont permis de mener sans contrainte ces longues études et m'ont toujours soutenu et encouragé dans ce long parcours et aidé dans la vie. Merci pour votre amour sans fin, vos encouragements et votre soutien émotionnel. Grâce aux sacrifices inestimables que vous m'avez consentis, vous avez tant souhaité que je parvienne à ce but. Mon succès est le vôtre! Je vous serais reconnaissant de toute ma vie, que Dieu vous accorde longue vie en santé!

Enfin, je réserve les derniers mots de ces remerciements à mon amour OMAR, merci chéri de ton soutien permanent dans les moments difficiles de mon travail, merci de tes conseils, ton encouragement, ton écoute et ta patience tout au long de ces années.

A vous tous ♡

Aya ♡

Acknowledgments

First and foremost, I thank God almighty, Allah for giving me the strength to survive, patience, as well as the courage to overcome all difficulties that has immensely guided me in finishing this thesis.

I would like to express my sincere appreciation to my thesis advisor Nicolas Wicker professor at the University of Lille, and to my co-director Mr Yann Guermeur, Research Director at the CNRS, assigned to LORIA INRIA Nancy, who accompanied me during these years of work. I am grateful for the opportunity to do this thesis that you were able to grant me by allowing me to realize my dream, that of becoming a doctor in applied mathematics. Thank you Nicolas, thank you Yann, for everything and especially for your human and scientific qualities. I will never forget your kindness, your patience, your generosity, your research spirit, your effective comments and your valuable support during these years. Thank you Yann for the invitations to nancy, this has been a great pleasure working with you.

Many thanks to the members of the jury, who have done me the honor to study my work carefully: Khalid Benabdeslem and Faïcel Chamroukhi for having accepted to be rapporteurs of this thesis. I am also grateful to Marianne Clausel and Sophie Dabo-Niang for their commitment to take part in my jury committee as examiners.

My gratitude also goes to the foundation Walid Joumblatt for the university studies to give me the certificate of scholarship, and I thank Mr. Alaa Terro who facilitated all the administrative procedures.

I would like to express my sincere thanks to all the teachers who taught me in the Lebanese university and university of Nice Sophia Antipolis and who by their skills supported me in the pursuit of my studies.

I thank all the staff of the Laboratory especially a big thank you to the members of the Probability and Statistics teams of the Paul Painlevé Laboratory in which I flourished during these years.

I would like to thank the secretariats Ludivine Fumery, Fatima Soulaimani, Soledad Cuenca for the order of missions and the computer scientists Sebastien Huart and Mohammed Khabzaoui. I would also like to thank the Executive Secretary Sabine Hertsoen, the Secretariat of Master 2 and PHD Student Aurore Smets and the Head of the printing Frédérique Maréchal for their kindness. I take these few lines to express my gratitude.

I would like to thank Mr. Nicolas Wicker who gave me the opportunity to take charge of teaching at the University of Lille 1. I then thank the University of Lille 3 for accepting me for a post ATER to teach during my thesis which was a pleasant and wonderful experience.

I extend a big thank you to the members of Lille 3 University (UFR MIME) especially with the teachers I taught with them, Mrs. Laurence Broze, Mr. Olivier Torres, Mr. Baba Thiam, Mrs. Aurore Lavigne, Ms. Sophie Dabo, Mrs. Ophélie Guin, Mr. Aboubakar Amiri, Mr. Camille Sabbah, Mrs. Amélie Zill and Mrs. Mylène Maida.

During this thesis, I had the pleasure of exchanging in good humor with many research professors and with many doctoral students. My gratitude goes to my friends in France for their support and for all the times we have spent together during these years, and I extend a big thank you to my friends in Lebanon (Amal Hamieh, Jihan, Joyce, Hiba, Diala, ...) for their encouragement and moral support ... simply for their friendship.

I warmly thank my family, my parents, my brothers (Adel, Khalil), my sisters (Nadima, Nour) and my grandparents who allowed me to lead these long studies without constraint and have always supported and encouraged me in this long journey and helped in the life. Thank you for your endless love, your encouragement and your emotional support. Thanks to the invaluable sacrifices you have given me, you have desired so much that I reach this goal. My success is for you! I would be grateful to you all my life, may God grant you long life in health!

Finally, I reserve the last words of these thanks to my love OMAR, thank you darling for your constant support in the difficult moments of my work, thank you for your advice, your encouragement, your listening and your patience throughout these years.

To you all ♡

Aya ♡

Résumé

La théorie statistique de l'apprentissage est un domaine de la statistique inférentielle dont les fondements ont été posés par Vapnik à la fin des années 60. Il est considéré comme un sous-domaine de l'intelligence artificielle. Dans l'apprentissage automatique, les machines à vecteurs de support (SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de discrimination et de régression.

Dans cette thèse, notre objectif est de proposer deux nouveaux problèmes d'apprentissage statistique: un portant sur la conception et l'évaluation d'une extension des SVM multi-classes et un autre sur la conception d'un nouveau noyau pour les machines à vecteurs de support.

Dans un premier temps, nous avons introduit une nouvelle machine à noyau pour la reconnaissance de modèle multi-classe: la machine à vecteur de support hyperbolique. Géométriquement, il est caractérisé par le fait que ses surfaces de décision dans l'espace de redescription sont définies par des fonctions hyperboliques. Nous avons ensuite établi ses principales propriétés statistiques. Parmi ces propriétés nous avons montré que les classes de fonctions composantes sont des classes de Glivenko-Cantelli uniforme, ceci en établissant un majorant de la complexité de Rademacher. Enfin, nous établissons un risque garanti pour notre classifieur.

Dans un second temps, nous avons créé un nouveau noyau s'appuyant sur la transformation de Fourier d'un modèle de mélange gaussien. Nous procédons de la manière suivante: d'abord, chaque classe est fragmentée en un nombre de sous-classes pertinentes, ensuite on considère les directions données par les vecteurs obtenus en prenant toutes les paires de centres de sous-classes d'une même classe. Parmi celles-ci, sont exclues celles permettant de connecter deux sous-classes de deux classes différentes. On peut aussi voir cela comme la recherche d'invariance par translation dans chaque classe. Nous l'avons appliqué avec succès sur plusieurs jeux de données dans le contexte d'un apprentissage automatique utilisant des machines à vecteurs support multi-classes.

Mots-clés. Apprentissage statistique, Classifieur multi-classe à marge, Glivenko-

Cantelli uniforme, Risque garanti, Complexité de Rademacher.

Abstract

Statistical learning theory is a field of inferential statistics whose foundations were laid by Vapnik at the end of the 1960s. It is considered a subdomain of artificial intelligence. In machine learning, support vector machines (SVM) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis.

In this thesis, our aim is to propose two new statistical learning problems: one on the conception and evaluation of a multi-class SVM extension and another on the design of a new kernel for support vectors machines.

First, we introduced a new kernel machine for multi-class pattern recognition : the hyperbolic support vector machine. Geometrically, it is characterized by the fact that its decision boundaries in the feature space are defined by hyperbolic functions. We then established its main statistical properties. Among these properties we showed that the classes of component functions are uniform Glivenko-Cantelli, this by establishing an upper bound of the Rademacher complexity. Finally, we establish a guaranteed risk for our classifier.

Second, we constructed a new kernel based on the Fourier transform of a Gaussian mixture model. We proceed in the following way: first, each class is fragmented into a number of relevant subclasses, then we consider the directions given by the vectors obtained by taking all pairs of subclass centers of the same class. Among these are excluded those allowing to connect two subclasses of two different classes. We can also see this as the search for translation invariance in each class. It successfully on several datasets in the context of machine learning using multiclass support vector machines.

Keywords. Machine learning, Margin multi category classifiers, Uniform Glivenko-Cantelli, Guaranteed risk, Rademacher complexity.

List of Figures

List of Tables

Outline of the thesis

This thesis is organized in four main chapters that can be read independently of the others to some extent.

Chapter 1 is an autonomous introductory chapter, we will first present the state of the art in the field of statistical learning introducing the scientific context: supervised and unsupervised learning. Then we will present the problem of binary classification in both cases: linearly and non-linearly separable, and we will discuss the problem of multi-class classification. Finally, we will present the capacity measures among these, the Rademacher complexity.

Chapter 2, will introduce fundamental elements of kernel theory. The goal of this chapter is to explain the interest of kernel methods through motivating examples, to introduce essential notions such as kernels, positive semidefinite property of kernel function and kernel matrix and reproducing kernel Hilbert spaces. We also give the properties of the operations on the kernel function that will provide a new kernel. Finally we provide several examples of kernels that includes a large number of widely used kernels such as the polynomial kernels and the Gaussian kernel.

Chapter 3, we will introduce a new kernel machine for multi-class pattern recognition: the hyperbolic support vector machine. Its decision boundaries in the feature space are defined by hyperbolic functions. We will establish its main statistical properties.

Chapter 4, we will construct a novel kernel function obtained as a Fourier transform of a Gaussian mixture model with the purpose of detecting translation invariance inside classes, which is applied successfully on several datasets in the context of machine learning using multiclass support vector machines (MSVM).

Chapter 1

Preliminaries

Abstract In this chapter, some concepts, background methods and results are presented. We begin first with an introduction about the data classification method. Then we present the problem of discrimination. Support vector machines (SVM) are then presented in two steps: the algorithms of the SVM bi-classes and the use of SVM to realize polytomies. Finally, we presented the capacity measures that will help us in chapter 3.

1.1 Introduction

Statistical learning [35] is a paradigm that combines a set of methods and algorithms for extracting relevant information from the data, or learning behaviors from examples. Its applications are numerous and present in fields as varied as the search for information in large data sets (thematic segmentation of text, image mining, etc.) or biology (behavior of population, DNA chips, etc.). We distinguish two main issues in statistical learning: supervised learning on the one hand, and unsupervised learning on the other hand.

Supervised learning is a type of machine learning that involves establishing rules of behaviour from a database containing examples of previously labelled cases. More precisely, this database is a set of input-output pairs $\{(x_i, y_i)\}_{1 \leqslant i \leqslant n} \in \mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the output space. The objective is then to learn how to predict, for any new input x, the output y.

According to the structure of $\mathcal{Y}$ we can be in two typical situations:

- When the space $\mathcal{Y}$ (which we will also call space of predictions) is finite, we call this task a problem of discrimination or classification, which is to assign a label to each input [42]. A function from $\mathcal{X}$ to $\mathcal{Y}$ is usually called a classifier or a decision rule.
- When $\mathcal{Y}$ is a continuous set, typically $\mathcal{Y} = \mathbb{R}$, we are talking about problem of regression [41], the prediction function is then called a regressor.

The unsupervised learning theory is a branch of machine learning that deals with the case where only the inputs $\{x_i\}_{1\leqslant i\leqslant n}$ are available, without the outputs, i.e that learns from test data that has not been labeled. The most important problem is then to partition the data, also called clustering. This is to group the observations into different homogeneous groups (clusters), by ensuring that the data in each subset share common characteristics.

As part of this thesis, we focus specifically on the supervised learning problem.

1.2 Data classification methods

Data classification sets the basic steps for anticipating corresponding labels of new points on the basis of a pre-defined set of labeled training points specially in the domain of machine learning and statistics. It is considered to be one of the supervised learning problems and it is applied in various domains such as document classification, handwriting recognition, internet search engines, etc.

Instance-based learning methods, neural networks, decision trees, support vector machines and many other methods have been developed to deal with data classification problems.

Instance-based learning is a family of learning algorithms that compares instances seen in training, which have been stored in memory with new problem instances, instead of performing explicit generalization. The idea is based on the assumption that features which are used to describe labels are similar when instances are close. It is reasonable to use labels of closest instances to predict labels of new instances. However, it is tough to understand the relationship between labels and features from the unstructured algorithms. Such methods are very efficient to manage real life problems, for example handwritten digits and satellite image scenes. For example the k-nearest neighbor (k-NN) method, given a training set, to predict the label of a new data point, one assigns k closest points of the new point in the training set, and labels it with the majority label in the k closest neighbors. The closest neighbors are found with a distance which can be seen as a similarity measurement. For example, the distance can be chosen as the L_p distance or the Minkowski distance. k is usually chosen using cross-validation [4] in real applications. A detailed discussion of instance based methods can be found in [[1], [2], [3]].

Neural Networks (NNets) were firstly defined by the neurophysiologist Warren McCulloch and the logician Walter Pitts in 1943. NNets are basically modeled referring to the model of the neural structure in the brain. In human brain, a neuron collects signals from other neurons through structures called dendrites, and sends out electrical activities through an axon which has thousands of branches. A synapse which is at the end of each branch converts activities from the axon to the connected neurones. The brain can perform highly complex computations due to the complication of the neurons networks. We focus on NNets which contain no cycles, called feedforward networks. NNets can be described as directed acyclic graphs, in which the nodes correspond to neurons and edges correspond to links between them. Each node accepts a weighted sum of outputs of the connected nodes related to the coming edges as input. NNets may contain several layers of nodes. In recent years, NNets have proven to be extremely proficient for numerous learning tasks because of the increased computational power to handle large datasets, and developments of new algorithms. Networks with multiple hidden layers, or what is called "Deep networks", have been successfully applied to many practical domains. A detailed overview can be found in [[5], [6]].

Decision trees (DT) are tree models that describe the classification paths for training data, which are constructed by nodes and directed edges. DT tasks are an admired method for various machine learning. A node is called a leaf node if it has no children. A leaf node presents a label of one of the classes, otherwise it is an internal node. Each internal node shows a rule for splitting the input space by one of the features or a predefined criterion. A classification tree predicts the label of a new point by going from the root node of the tree model to a leaf. A general method for constructing a classification tree is a recursive procedure that allows you to choose the "best characteristic" on each node, then split the feature space in reference of the "best feature". The best decision is made at each node by following this way. Applying it recursively, until all learning points are well classified or no proper feature can be used as a node, a classification tree is obtained. Occasionally, classification trees may discriminate training data well, considering a high prediction error. However, limiting the number of nodes generated helps avoiding errors. A commonly used methodology in practice is to prune the tree after it is built. Algorithms for building a decision tree include C4.5 [9], ID3 [10] , and CART [11]. Classification trees are simple but powerful. With each division on each node, the feature space partition is fully described, which makes the classification path more readable. Yet, the partition is sensitive to small changes in input data points. Due to the hierarchical nature of the process, small changes can guide to a significantly different series of splits.

The random forest (RF) algorithm [6] [7], proposed in 2001 by Leo Breiman and Adèle Cutler, a general purpose classification and regression method has been extremely successful. The approach, which combines several randomized decision trees and aggregates their predictions by averaging, has shown excellent performance in settings where the number of variables is much larger than the number of observations. These trees are grown separately with randomly selected subsets of input data points and randomly selected subsets of variables. To predict a label of a new data point, RF takes the majority vote from the predicted labels of all trees therein. RF can handle many input variables without doing a feature selection that's why it is said to run efficiently on large scale datasets and that is one of its greatest advantages. It can approximate missing data effectively as well, even when the missing data correspond to a large portion of the dataset. Interestingly, RF provides a measure of the importance of each variable.

Support vector machines (SVM) aim to construct a linear separation maximizing the margin between data belonging to different classes. They can be applied to a nonlinear separation problem by using a kernel function to transform the input space to a higher feature space (see "kernel trick", based on Mercer's Theorem [40]) which makes the problem linear. SVMs have already been well studied and achieve high accuracy in many applications. A detailed discussion of SVMs can be found in [[12], [13], [14], [15], [16], [17]].

In this thesis, we are interested among these methods, in the classification problem based on the support vector machine, we give a detailed introduction of this method in the next section.

1.3 Support Vector Machines

In machine learning, SVM are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. SVM were first invented by Vapnik and Chervonenkis [[17]]. They have been widely studied by many researchers [[18], [19],[12],[20], [21], [22], [23], [24], [25], [26]]. In 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik suggested a way to create non-linear classifiers by applying the kernel trick to maximum-margin hyperplanes [27]. The current standard incarnation (soft margin) was proposed by Corinna Cortes and Vapnik in 1993 and published in 1995 [28].

1.3.1 General operating principle of the SVMs

The perceptron is an algorithm for supervised learning of binary classifiers . A SVM, as a perceptron, finds a linear separator between the data points of two differents classes. In general, there may be several separators possible between classes (assuming the problem linearly separable) and a perceptron has no preference among them. In the SVM, however, we make a particular choice among all the possible separators: we want the one with the maximum "margin" [29].

1.3.1.1 Basic knowledge: Hyperplane, margin and support vectors

For two classes of examples given, the goal of SVM is to find a classifier that will separate the data and maximize the distance between these two classes. With SVM, this classifier is a linear classifier called **hyperplane**. In the following figure, we determine a hyperplane that separates the two sets of points.

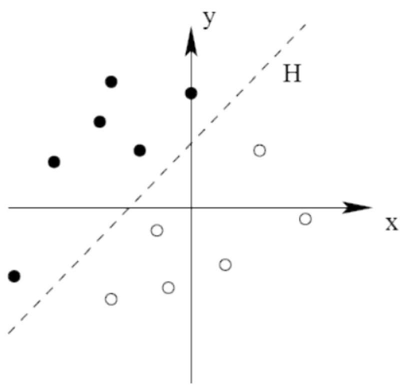


Figure 1.1: The hyperplane H that separates the two sets of points.

The nearest points, which alone are used for determining the hyperplane, are called support vectors.

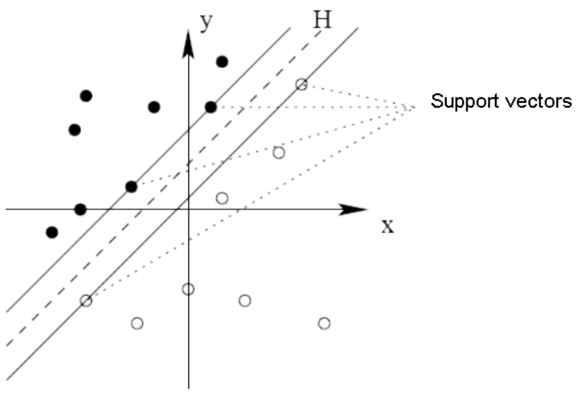


Figure 1.2: Support vectors.

There are many hyperplanes that separate the two classes of examples. The principle of the SVM is to choose the one that will maximize the minimum distance between the hyperplane and the training examples (i.e. the distance between the hyperplane and the support vectors), this distance is called the margin [30].

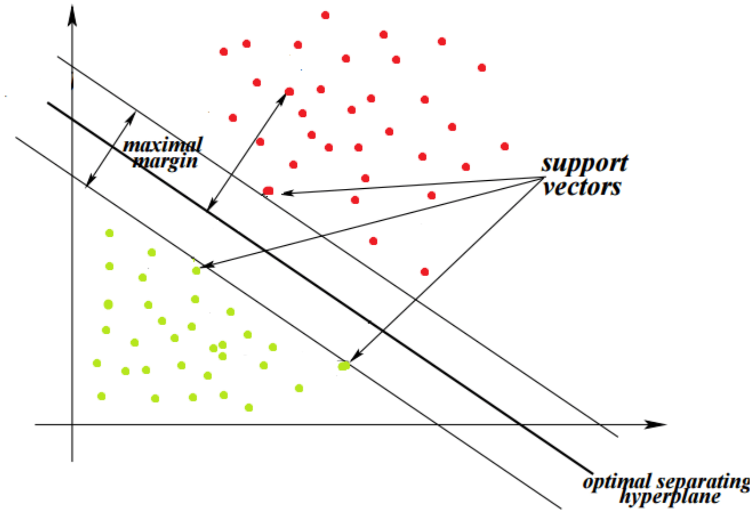


Figure 1.3: Optimal separating hyperplane, Support vectors and maximal margin.

1.3.1.2 Why maximize the margin?

Intuitively, having a wider margin provides more security when classifying a new example. Moreover, if we find the classifier that behaves best with respect to the learning data, it is clear that it will also be the one that will best classify the new examples. In the following figure, the right side shows us that with an optimal hyperplane, a new example remains well classified so that it falls in the margin. We see on the left side that with a smaller margin, the example is misclassified.

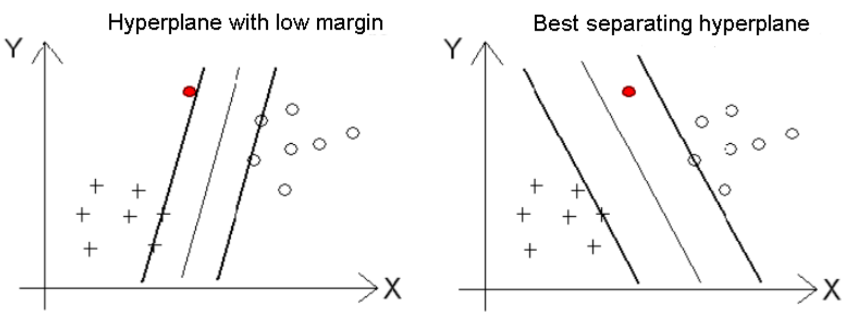
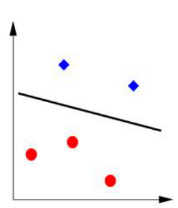


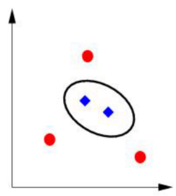
Figure 1.4: Best separating hyperplane.

1.3.1.3 Linearity and non-linearity

Among the SVM models, one can observe linearly separable cases and nonlinearly separable cases. The first ones are the simplest of SVM because they allow to easily find the linear classifier. In most real problems there isn't possible linear separation between the data, the maximum margin classifier can not to be used because it only works if the classes of training data are linearly separable.



(a) Linearly separable case



(b) Non-Linearly Separable case

To overcome the disadvantages of non-linearly separable case, the idea of SVM is to change the data space. The nonlinear transformation of the data can allow a linear separation of the examples in a new space. So we will have a change of dimension. This new space is called "feature space" (redescription space). Indeed, intuitively, the larger the dimension of the redescription space, the greater the probability of finding a separating hyperplane between examples is high. This is illustrated by the following figure 1.6.

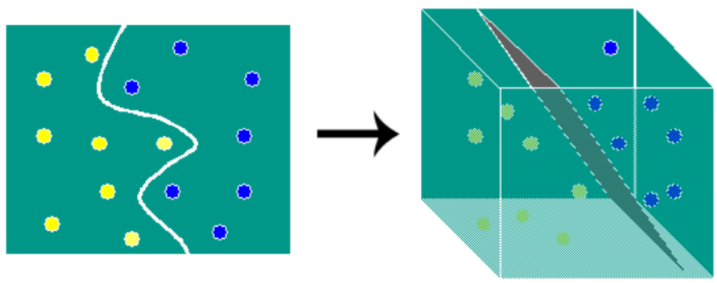


Figure 1.6: The nonlinear transformation of the data.

So we have a transformation of a problem of nonlinear separation in the representation space into a problem of linear separation in a space of redescription of larger dimension. This nonlinear transformation is performed via a kernel function. In practice, some families of kernel functions are known and it returns to the user of SVM to perform tests to determine which one is best for their application. Examples of the following kernel: polynomial, Gaussian, sigmoïd and Laplacian.

We consider problems of discrimination in C categories. Let $\mathcal{X}$ denote the description space and $\mathcal{Y}$ the set of categories. Each object is represented by its description $x \in \mathcal{X}$ and the set $\mathcal{Y}$ of the categories y can be identified with the set of indices of the categories: $[\![1, C]\!]$.

1.3.2 SVM for binary classification

In the first part, we will talk about the hard-SVM which are used for linearly separable training sets, next we introduce soft-SVM for non-linearly separated datasets. Finally we present Soft-SVM with kernels.

1.3.2.1 Hard-SVM

In this section, we consider a binary classification problem ($C = 2$), for which the set of categories $\mathcal{Y}$ is identifiable at $\{-1, +1\}$. Assume that we are given n observations $(x_1, y_1), \cdots, (x_n, y_n) \in \mathcal{X} \times \{-1, 1\}$. Given a test point $x \in \mathcal{X}$, the goal is to guess the corresponding $y \in \{-1, +1\}$ based on the n observations.
Consider for instance $\mathcal{X} = \mathbb{R}^d$, with $d \in \mathbb{N}^\star$. A way of fulfilling this goal is to find the "maximum-margin hyperplane" that divides the group of points x_i for which $y_i = 1$ from the group of points for which $y_i = -1$, which is defined so that the distance between the hyperplane and the nearest point x_i from either group is maximized.

We define the function g be a classifier, with real values from a class of functions $\mathcal{G}$ by: $g(x) = \langle w, x\rangle + b$, where $w \in \mathbb{R}^d$ is the normal vector to the hyperplane and $b \in \mathbb{R}$ is its relative position to the origin.

The separating hyperplane defined by $\{x \in \mathbb{R}^d : g(x) = 0\}$. Thus a new point $x \in \mathbb{R}^d$ is assigned to the prediction function dr_g defined as

$$\mathrm{dr}_g(x) = sgn(g(x)),$$

with $sgn()$ is the the sign function of its argument.

We assume that the two classes are linearly separable, which means that there is a hyperplane able to classify the data. However, there is usually an infinity of separator hyperplanes that classify the data correctly. From all these hyperplanes, we seek to find a hyperplane that maximizes the margin between the samples and the separating hyperplane.

One approach is to maximize the margin between two parallel hyperplanes that separate the two classes of data. This method is called the Hard-SVM. Since the separating hyperplane is $g(x) = 0$, the supporting hyperplane can be written as $g(x) \geqslant k$ for class $y_i = 1$ and $g(x) \leqslant -k$ for class $y_i = -1$, in which $k > 0$. After a rescaling, we require the supporting hyperplane to be $g(x) \geqslant 1$ for class $y_i = 1$ and $g(x)$ for class $y_i = -1$. Geometrically, the distance between these two hyperplanes is $2/\|w\|_2$, so to maximize the distance between the planes (margin) is equivalent to minimize $\|w\|_2/2$. Therefore, we obtain the following Hard-SVM formulation:

$$\begin{cases} \min\limits_{w,b} & \frac{1}{2}\|w\|^2 \\ \textit{subject to} & y_i\left(\langle w, x_i\rangle + b\right) \geqslant 1,\ i = 1, \ldots, n. \end{cases} \tag{1.1}$$

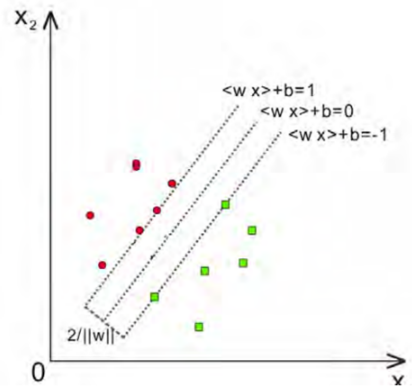


Figure 1.7: Linearly separable classification problem in the (x_1, x_2) space. Green squares and red dots represent different classes of points. The graph shows that for linearly separable training set, there are many possible hyperplanes separators and that the margin of two parallel supporting hyperplanes is equal to $2/\|w\|_2$.

In general, it is also not possible to find a linear separator in the redescription space. It may also be that samples are mislabeled and that the separating hyperplane is not the best solution to the problem of classification. In 1995, Corinna Cortes and Vladimir Vapnik [12] proposed a technique called the Soft-SVM, which tolerates bad classifications.

1.3.2.2 Soft-SVM

If the two classes are not separable linearly, these two classes are found mixed around the separation hyperplane. The technique of the Soft-margin looks for a separator hyperplane

that minimizes the number of errors (points on the wrong side of the supporting hyperplanes) by introducing slack variables ξ_i, which make it possible to release the constraints on the learning vectors.
Let $\xi_i > 0, i = 1, \ldots, n$, be slack variables. The constraints in the Hard-SVM formulation can be reformulated as:

$$y_i\left(\langle w, x_i\rangle + b\right) + \xi_i \geqslant 1,$$
$$\xi_i \geqslant 0,\ i = 1, \ldots, n.$$

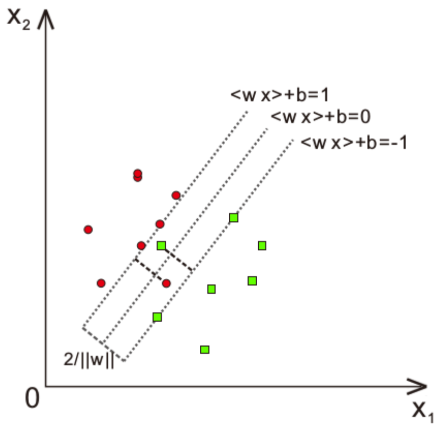


Figure 1.8: Non-linearly separable classification problem. Green squares and red dots are on the wrong side of the separating hyperplane. Soft-SVMs minimize the distances (ξ_i) of the points on the wrong side of the hyperplane and maximize the margin at the same time.

We see that a slack variable ξ_i penalizes an error vector (see Figure 1.8). Therefore the optimization problem in the case of non-separable data (Soft-SVM) is [12, 13, 26]:

$$\begin{cases} \min\limits_{w,b,\xi} & \frac{1}{2}\|w\|^2 + \zeta \sum\limits_{i=1}^{n} \xi_i \\ \textit{subject to} & y_i\left(\langle w, x_i\rangle + b\right) + \xi_i \geqslant 1, \qquad i = 1, \ldots, n \\ & \xi_i \geqslant 0, \qquad\qquad\qquad\qquad\quad i = 1, \ldots, n, \end{cases} \tag{1.2}$$

where ζ is a penalty parameter for misclassified points that compromise between margin width and misclassified points.

In constraints of Problem (1.2), since $\xi_i \geqslant 0$ and $\xi_i \geqslant 1 - y_i\left(\langle w, x_i\rangle + b\right)$, we get an equality as follows:

$$\xi_i = \max\{0, 1 - y_i\left(\langle w, x_i\rangle + b\right)\}. \tag{1.3}$$

So we have two situations:

- No error: $\xi_i = 0$, which means that the i-th point is either correctly classified,
- Error: $\xi_i = 1 - y_i\left(\langle w, x_i\rangle + b\right)$.

Using Eq. 1.3, we can now formulate the problem of optimization 1.2 as an unconstrained optimization problem:

$$\min_{w,b} \qquad \frac{1}{2}\|w\|^2 + \zeta \sum_{i=1}^{n} \max\{0, 1 - y_i\left(\langle w, x_i\rangle + b\right)\}. \tag{1.4}$$

The term $\max\{0, 1 - y_i\left(\langle w, x_i\rangle + b\right)$ is called "hinge loss" in statistics. From the problem (1.4), the SVM can be considered as regularized minimization problems where $\|w\|^2$ is a part of the regulation. In general, we can write SVM as follows,

$$\min_{w,b} \qquad \frac{1}{2}\|w\|^2 + \zeta \sum_{i=1}^{n} l\left(y_i, \left(\langle w, x_i\rangle + b\right)\right),$$

where l is a loss function.

To obtain the dual formulation of problem 1.2, we introduce the multipliers of Lagrange β_i and μ_i, the Lagrangian function is given by:

$$L(w, b, \xi, \beta, \mu) = \frac{1}{2}\|w\|^2 + \zeta \sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\beta_i\left(y_i\left(\langle w, x_i\rangle + b\right) - 1 + \xi_i\right) - \sum_{i=1}^{n}\mu_i\xi_i. \tag{1.5}$$

The Lagrangian must be optimized with respect to w ,b, ξ_i and the multipliers of Lagrange β_i and μ_i. By setting the partial derivatives to zero of the Lagrangian with respect to w ,b, ξ_i , we obtain the following:

$$\begin{array}{lll}
\dfrac{\partial L}{\partial w} = 0 & \Rightarrow & w = \displaystyle\sum_{i=1}^{n}\beta_i y_i x_i, \\
\dfrac{\partial L}{\partial \xi_i} = 0 & \Rightarrow & \beta_i = \zeta - \mu_i, \quad \forall i, \\
\dfrac{\partial L}{\partial b} = 0 & \Rightarrow & 0 = \displaystyle\sum_{i=1}^{n}\beta_i y_i, \\
\beta_i, \mu_i, \xi_i & \geqslant & 0, \qquad\qquad \forall i.
\end{array}$$

Substituting these results in the equation of Lagrangian 1.5 we obtain the dual problem as follow:

$$\begin{cases}
\max\limits_{\beta} \qquad -\frac{1}{2}\displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n}\beta_i\beta_j y_i y_j\langle x_i, x_j\rangle + \sum_{i=1}^{n}\beta_i \\
\textit{subject to} \qquad \displaystyle\sum_{i=1}^{n}\beta_i y_i = 0 \\
\qquad\qquad\qquad \zeta \geqslant \beta_i \geqslant 0, \qquad i = 1, \ldots, n.
\end{cases} \tag{1.6}$$

The decision function to classify a new observation x is always

$$\mathrm{dr}_g\,(x) = sgn\,(\langle \sum_{i=1}^{n} \beta_i y_i x_i, x\rangle + b)\,.$$

1.3.2.3 Soft-SVM with kernels

In the case where the two classes are slightly non-linearly separable, the Soft-SVM are sufficient as the figure 1.8. In this part, we present a method, called soft-SVM with kernels, deals with the case where the two classes are strongly non-linearly separable (as Figure 1.9). In order to remedy the problem of the absence of a linear separator, the idea of kernel trick is to reconsider the problem in a space high-dimensional, possibly of infinite dimension. In this new space, it is then probable that there is a linear separation.

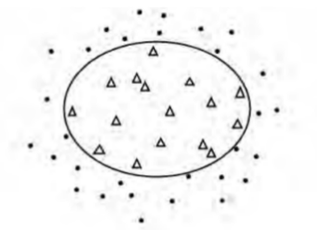


Figure 1.9: Classification case when two classes are strongly non-linearly separable.

The projection on a space of higher dimension makes it possible to perform linear operations equivalent to nonlinear operations on the input space, this projection is performed by the function of projection Φ defined as:

$$\begin{aligned} \Phi :\ \ & \mathcal{X} \longrightarrow H \\ & x_i \longrightarrow \Phi(x_i) \end{aligned}$$

where the mapping space H is a Hilbert space and $dim(H) \gg d$. Hence the optimization problem 1.6 will be reformulated as:

$$\begin{cases} \max\limits_{\beta} & -\frac{1}{2}\sum\limits_{i=1}^{n}\sum\limits_{j=1}^{n}\beta_i\beta_j y_i y_j \langle \Phi(x_i), \Phi(x_j)\rangle + \sum\limits_{i=1}^{n}\beta_i \\ subject\ to & \sum\limits_{i=1}^{n}\beta_i y_i = 0 \\ & \zeta \geqslant \beta_i \geqslant 0, \qquad i = 1,\dots,n. \end{cases} \tag{1.7}$$

The inner product imposed by the projection is more complex and very expensive in computing due to the large dimension of Φ, other functions called Kernel function can realize this computation without making explicit projection towards other spaces, the use of function Kernel to avoid projection is known as "Kernel Trick" [15]. In the following , the image space H through Φ is instantiated by H_κ, where the mapping space H_κ is the corresponding reproducing kernel Hilbert space (RKHS), with κ is a real-valued positive type function/kernel.

A kernel function is defined as:

$$\begin{aligned} \kappa : \quad & \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R} \\ & (x_i, x_j) \longrightarrow \kappa(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle_{H_\kappa}. \end{aligned}$$

To replace the projection function, a Kernel function must verify the Mercer's theorem which states that a kernel function represents the scalar product if it is positive definite.

We present some examples of kernel functions:

Degree d polynomial:

$$\kappa(u, v) = (1 + \langle u, v \rangle)^d,$$

Radial Basis (RBF kernel):

$$\kappa(u, v) = \exp\left(\frac{-\|u - v\|^2}{2\sigma^2} \right),$$

Two-layer Neural Network:

$$\kappa(u, v) = S\left(\eta \langle u, v \rangle + c\right),$$

in which S is the sigmoïd function,

$$S(t) = \frac{1}{1 + e^t} \ .$$

We will present in Chapter 2 more details on the kernel theory.

By substituting the Kernel trick into the dual problem 1.6, the optimization problem is formulated as:

$$\begin{cases} \max\limits_{\beta} & -\frac{1}{2} \sum\limits_{i=1}^{n} \sum\limits_{j=1}^{n} \beta_i \beta_j y_i y_j \kappa(x_i, x_j) + \sum\limits_{i=1}^{n} \beta_i \\ \textit{subject to} & \sum\limits_{i=1}^{n} \beta_i y_i = 0 \\ & \zeta \geqslant \beta_i \geqslant 0, \qquad i = 1, \ldots, n. \end{cases} \tag{1.8}$$

The decision function is given by:

$$\mathrm{dr}_g\,(x) = sgn\,(\sum_{i=1}^{n}\beta_i y_i \kappa(x_i, x) + b)\,.$$

1.3.3 SVM for multi-class classification

We place ourselves in the context of discrimination in C categories with $C \geqslant 3$.
The principle of the SVM explained in the previous section is summarised in solving of binary classification problems, but the most classification problems are a multi-class problem. Hence the importance of extending the principle of SVM to the problems of more than two classes, there have been several attempts to combine binary classifiers to identify this problem (multi classes) [32], there are also attempts to incorporate the classification of several classes into the SVM process so that all classes are treated simultaneously [33].

In this section, we will discuss on strategies based on reducing the multi-class problem to multiple binary classification problems. We will then briefly explain some of the most widely used methods.

1.3.3.1 The combination of SVM bi-classes: a first step towards multi-class SVM

Decomposition methods can be used to address a multi-category discrimination problem ($C \geqslant 3$) as a combination of dichotomous calculation problems. We are dealing here only with the two main decomposition methods.

1.3.3.2 One versus All

We are given a training dataset of n points of the form $(x_1, y_1), \cdots, (x_n, y_n)$ where x_i, $i = 1, \cdots, n$ is a vector of length d and $y_i \in \mathcal{Y} = \{1, \cdots, C\}$ representing the class of the sample. A first approach is to create a classifier for each class, which separates the points of this class from all the other points. This method is called one-versus-rest (OVR abbreviated) or one-versus-all (OVA abbreviated).

It consists in using a binary classifier (with real values) by category. The k^{th} classifier is intended to distinguish the index category k from all the others. To assign an example, it is thus presented to C classifiers, and the decision is obtained according to the principle winner-takes-all: the label chosen is that associated with the classifier who returned the highest value. It is commonly quoted as older works evoking the use of this strategy with SVM [34] (see also [35]). In [35], the authors support the thesis that this approach, as simple as it is, when implemented with correctly parameterized SVM, obtains performances that are not significantly lower than those of the other methods. It should be emphasized, however, that it involves learning to allocate to very unbalanced categories, which often raises practical difficulties.

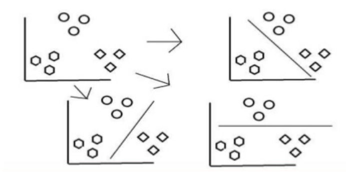


Figure 1.10: Three classes separated by the method of one versus all with linear separator.

1.3.3.3 One versus one

Another approach of decomposition, just as intuitive, is the "one-versus-one" method (OVO for short) [37]. Usually attributed to Knerr and his coauthors [38], it consists in using a classifier by couple of categories. The classifier indexed by the pair (k, l) (with $1 \leqslant k < l \leqslant C$), is intended to distinguish the index category k from that of index l. To assign an example, so it is presented to $C(C-1)/2$ classifiers, and the decision is usually obtained by performing a majority vote (max-wins voting).

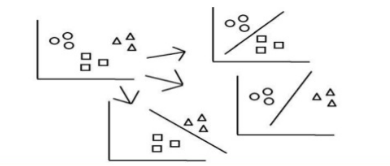


Figure 1.11: Multi-class classification by the method of one versus one.

There are several models of SVM multi classes (M-SVM) including Weston and Watkins model (WW) [33], Crammer and Singer (CS) [43], model of Lee, Lin and Wahba (LLW) [44] and Guermeur and Monfrini (MSVM2) [45]. We now describe the model of Weston and Watkins.

1.3.3.4 Weston and Watkins model

The first publication describing a multiclass SVM is [33] (see also [39]). It presents a model proposed independently by Vapnik and Blanz in slightly earlier oral communications (personal communication by Volker Blanz), and later by other authors in various forms.

The binary SVM optimisation problem [35] is generalised to the following:

$$\begin{cases} \min\limits_{w,b,\xi} & \frac{1}{2}\sum\limits_{k=1}^{C}\|w_k\|^2 + \zeta \sum\limits_{i=1}^{n}\sum\limits_{k\neq y_i}\xi_{ik} \\ subject\ to & \forall i, k \neq y_i, \quad \langle w_{y_i}, \Phi(x_i)\rangle + b_{y_i} \geqslant \langle w_k, \Phi(x_i)\rangle + b_k + 2 - \xi_{ik}, \\ & \xi_{ik} \geqslant 0, \qquad i = 1,\dots,n, \quad k = \{1,\dots,C\}\backslash y_i. \end{cases} \tag{1.9}$$

This gives the decision function :

$$\mathrm{dr}_g\,(x) = \underset{k}{\mathrm{argmax}}\ [\langle w_k, \Phi(x)\rangle + b_k], \quad k = 1,\dots,C.$$

We can use the Lagrangian to find the solution to this optimisation problem.

In the first part of thesis, we are interested in the problem of multiclass classification, we introduced a new margin multi category classifier based on classes of vector valued functions with one component function per category, it is a kernel machine. We found that separation surfaces are hyperbolic and this classifier generalizes the SVMs. We also exhibited the statistical properties of this classifier, among which Fisher consistency [50] and we showed that the classes of component functions are uniform Glivenko-Cantelli [51].

1.4 Hyperbolic kernel machine

This section presents the essential of our contributions to the theory of multi class SVM. We initially present the definition of Uniform Glivenko-Cantelli class . The definition of this property calls for the introduction of an intermediate definition.

Definition 1.1. ***(Empirical probability measure)*** *Let $(\mathcal{T}, \mathcal{A}_{\mathcal{T}})$ be a measurable space and let T be a random variable with values in $\mathcal{T}$, distributed according to a probability measure P_T on $(\mathcal{T}, \mathcal{A}_{\mathcal{T}})$. For $n \in \mathbb{N}^*$, let $\mathbf{T}_n = (T_i)_{1\leqslant i\leqslant n}$ be an n-sample made up of independent copies of T. The empirical measure supported on this sample, $P_{T,n}$, is given by*

$$P_{T,n} = \frac{1}{n}\sum_{i=1}^{n}\delta_{T_i},$$

where δ_{T_i} denotes the Dirac measure centered on T_i.

Definition 1.2 (Uniform Glivenko-Cantelli class [46, 47]). *Let the probability measures P_T and $P_{T,n}$ be defined as in Definition 1.1. Let $\mathcal{F}$ be a class of real-valued functions on $\mathcal{T}$. Then for $\epsilon \in \mathbb{R}_+^*$, $\mathcal{F}$ is an* ϵ-uniform Glivenko-Cantelli class *if*

$$\lim_{n \longrightarrow +\infty} \sup_{P_T} \mathbb{P}\left(\sup_{n' \geqslant n} \sup_{f \in \mathcal{F}} \left| \mathbb{E}_{T' \sim P_{T,n'}} \left[f\left(T'\right) \right] - \mathbb{E}_{T \sim P_T} \left[f\left(T\right) \right] \right| > \epsilon \right) = 0,$$

where $\mathbb{P}$ denotes the probability with respect to the sample. $\mathcal{F}$ is said to be a uniform Glivenko-Cantelli class *if $\mathcal{F}$ is an ϵ-Uniform Glivenko-Cantelli class for all value of ϵ.*

In this following, we shall refer to Uniform Glivenko-Cantelli classes by the abbreviation "GC classes". We can use the following three capacity measures to demonstrate that the component function class is GC class.

1.4.1 Capacity measures

We start by giving the definition of the capacity measures which are the covering-numbers that characterize "GC classes". The definition of these concepts, as well as those of the underlying concepts of cover and net, have been originally introduced in [52].

Definition 1.3. ***(ϵ-cover, ϵ-net, covering numbers, and ϵ-entropy)*** *Let (E, ρ) be a pseudo-metric space, $E' \subset E$ and $\epsilon \in \mathbb{R}_+^*$. An* ϵ-cover *of E' is a coverage of E' with open balls of radius ϵ the centers of which belong to E. These centers form an* ϵ-net *of E'. A* proper ϵ-net *of E' is an* ϵ-net *of E' included in E'. If E' has an ϵ-net of finite cardinality, then its* covering number $\mathcal{N}(\epsilon, E', \rho)$ *is the smallest cardinality of its ϵ-nets:*

$$\mathcal{N}\left(\epsilon, E', \rho\right) = \min\left\{ |E''| : \ (E'' \subset E) \wedge \left(\forall e \in E', \ \rho\left(e, E''\right) < \epsilon\right) \right\}.$$

If there is no such finite net, then the covering number is defined to be infinite. The corresponding logarithm, $\log_2\left(\mathcal{N}\left(\epsilon, E', \rho\right)\right)$, is called the minimal ϵ-entropy *of E', or simply the* ϵ-entropy *of E'. $\mathcal{N}^{(p)}\left(\epsilon, E', \rho\right)$ will designate a covering number of E' obtained by considering proper ϵ-nets only. In the finite case, we have thus:*

$$\mathcal{N}^{(p)}\left(\epsilon, E', \rho\right) = \min\left\{ |E''| : \ (E'' \subset E') \wedge \left(\forall e \in E', \ \rho\left(e, E''\right) < \epsilon\right) \right\}.$$

In the following, we will define the functional pseudo-metric based on the L_2-norm and on the uniform convergence norm.

Definition 1.4. ***(Pseudo-distance $d_{2,\mathbf{t}_n}$ and $d_{\infty,\mathbf{t}_n}$)*** *Let $\mathcal{F}$ be a class of real-valued functions on $\mathcal{T}$. For $n \in \mathbb{N}^*$, let $\mathbf{t}_n = (t_i)_{1 \leqslant i \leqslant n} \in \mathcal{T}^n$. Then,*

$$\forall \left(f, f'\right) \in \mathcal{F}^2, \ \ d_{2,\mathbf{t}_n}\left(f, f'\right) = \|f - f'\|_{L_2(\mu_{\mathbf{t}_n})} = \left(\frac{1}{n} \sum_{i=1}^{n} \left(f\left(t_i\right) - f'\left(t_i\right) \right)^2 \right)^{\frac{1}{2}},$$

$$\forall (f, f') \in \mathcal{F}^2, \;\; d_{\infty,\mathbf{t}_n}(f, f') = \|f - f'\|_{L_\infty(\mu_{\mathbf{t}_n})} = \max_{1 \leqslant i \leqslant n} |f(t_i) - f'(t_i)| .$$

where $\mu_{\mathbf{t}_n}$ *denotes the uniform probability measure on* $\{t_i : 1 \leqslant i \leqslant n\}$.

Definition 1.5. ***(Uniform covering numbers [49])*** *Let* $\mathcal{F}$ *be a class of real-valued functions on* $\mathcal{T}$ *and* $\bar{\mathcal{F}} \subset \mathcal{F}$. *For* $p \in \{2, \infty\}$, $\epsilon \in \mathbb{R}_+^*$, *and* $n \in \mathbb{N}^*$, *the* uniform covering number $\mathcal{N}_p\left(\epsilon, \bar{\mathcal{F}}, n\right)$ *is defined as follows:*

$$\mathcal{N}_p\left(\epsilon, \bar{\mathcal{F}}, n\right) = \sup_{\mathbf{t}_n \in \mathcal{T}^n} \mathcal{N}\left(\epsilon, \bar{\mathcal{F}}, d_{p,\mathbf{t}_n}\right).$$

We define accordingly $\mathcal{N}_p^{(p)}\left(\epsilon, \bar{\mathcal{F}}, n\right)$ *as:*

$$\mathcal{N}_p^{(p)}\left(\epsilon, \bar{\mathcal{F}}, n\right) = \sup_{\mathbf{t}_n \in \mathcal{T}^n} \mathcal{N}^{(p)}\left(\epsilon, \bar{\mathcal{F}}, d_{p,\mathbf{t}_n}\right).$$

There are several results which connect the Uniform Glivenko-Cantelli condition of a given class of functions to estimates on the covering numbers of that class. All the results are stated for classes of functions which are bounded by 1 . The results remain valid for classes of functions with a uniformly bounded range up to a constant which depends only on that bound.
The next result gives the notion of GC class and the covering numbers, this result is due to Dudley, Ghiné and Zinn [47].

Theorem 1.1. *Let* $\mathcal{F}$ *be a class of real-valued bounded functions on* $\mathcal{T}$. *Then, the following are equivalent:*

1. $\mathcal{F}$ *is a GC class.*

2. *For* $1 \leqslant p \leqslant \infty$,

$$\lim_{n \to \infty} \frac{\log_2\left(\mathcal{N}_p\left(\epsilon, \mathcal{F}, n\right)\right)}{n} = 0 \;, \quad for\ all\ \epsilon > 0.$$

Another important of the capacity measure used to analyse GC classes is the generalization of the Vapnik-Chervonenkis (VC) dimension [53]. It characterizes the learnability of the class of real-valued (binary) classifiers is the fat-shattering dimension [48], also known as the γ-dimension.

Definition 1.6. ***(Fat-shattering dimension [48])*** *Let* $\mathcal{F}$ *be a class of real-valued functions on* $\mathcal{T}$. *For* $\gamma \in \mathbb{R}_+^*$, *a subset* $s_{\mathcal{T}^n} = \{t_i : 1 \leqslant i \leqslant n\}$ *of* $\mathcal{T}$ *is said to be* γ-shattered *by $\mathcal{F}$ if there is a vector $\mathbf{b}_n = (b_i)_{1\leqslant i\leqslant n}$ in $\mathbb{R}^n$ such that, for all vector $\mathbf{l}_n = (l_i)_{1\leqslant i\leqslant n}$ in $\{-1,1\}^n$, there is a function $f_{\mathbf{l}_n}$ in $\mathcal{F}$ satisfying*

$$\forall i \in [\![1,n]\!], \quad l_i\left(f_{\mathbf{l}_n}(t_i) - b_i\right) \geqslant \gamma.$$

The vector $\mathbf{b}_n$ is called a witness *to the γ-shattering. The* fat-shattering dimension with margin γ *of the class $\mathcal{F}$, γ-dim$(\mathcal{F})$, is the maximal cardinality of a subset of $\mathcal{T}$ γ-shattered by $\mathcal{F}$, if such maximum exists. Otherwise, $\mathcal{F}$ is said to have infinite fat-shattering dimension with margin γ.*

The connection between GC classes and the notion of the fat-shattering dimension defined above is the following fundamental result [53]:

Theorem 1.2. *Let $\mathcal{F}$ be a class of uniformly bounded real-valued functions on $\mathcal{T}$, then it is a GC class if and only if it has a finite fat-shattering dimension for every $\gamma > 0$, i.e. for every $\gamma \in \mathbb{R}_+^*$, γ-dim$(\mathcal{F})$ is finite.*

The last capacity measure that gives the Uniform Glivenko Cantelli class is the Rademacher complexity. For $n \in \mathbb{N}^*$, a Rademacher sequence $\boldsymbol{\sigma}_n$ is a sequence $(\sigma_i)_{1\leqslant i\leqslant n}$ of independent random signs, i.e., independent and identically distributed (i.i.d.) random variables taking the values -1 and 1 with probability $\frac{1}{2}$ (symmetric Bernoulli or Rademacher random variables).

Definition 1.7. ***(Rademacher complexity)***

Let $(\mathcal{T}, \mathcal{A}_{\mathcal{T}})$ be a measurable space and let T be a random variable with values in $\mathcal{T}$, distributed according to a probability measure P_T on $(\mathcal{T}, \mathcal{A}_{\mathcal{T}})$. For $n \in \mathbb{N}^*$, let $\mathbf{T}_n = (T_i)_{1\leqslant i\leqslant n}$ be an n-sample made up of independent copies of T and let $\boldsymbol{\sigma}_n = (\sigma_i)_{1\leqslant i\leqslant n}$ be a Rademacher sequence. Let $\mathcal{F}$ be a class of real-valued functions with domain $\mathcal{T}$. The empirical Rademacher complexity *of $\mathcal{F}$ is*

$$\hat{R}_n(\mathcal{F}) = \mathbb{E}_{\boldsymbol{\sigma}_n}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i f(T_i) \,\middle|\, \mathbf{T}_n\right].$$

The Rademacher complexity *of $\mathcal{F}$ is*

$$R_n(\mathcal{F}) = \mathbb{E}_{\mathbf{T}_n}\left[\hat{R}_n(\mathcal{F})\right] = \mathbb{E}_{\mathbf{T}_n\boldsymbol{\sigma}_n}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \sigma_i f(T_i)\right].$$

Theorem 1.3 ([55])**.** *Let $\mathcal{F}$ be a class of uniformly bounded real-valued functions on $\mathcal{T}$. Then, the following are equivalent:*

1. $\mathcal{F}$ is a GC class.

2. For $n \in \mathbb{N}^*$,

$$\lim_{n\to\infty} R_n(\mathcal{F}) = 0.$$

Chapter 3 shows that the class of function of our classifier is GC class by using the third capacity measure which is the Rademacher complexity. It means that we use the theorem 1.3, it suffices to establish that the Rademacher complexity of this class of function converges to 0, as n goes to infinity.

1.4.2 Useful properties of the (empirical) Rademacher complexity

The following theorem summarizes some of the properties of the Rademacher averages we shall use.

Theorem 1.4. *Let $(\mathcal{T}, \mathcal{A}_\mathcal{T})$ be a measurable space. Let $\mathcal{F}$ and $\mathcal{F}'$ be classes of real-valued functions on $\mathcal{T}$. Then, for $n \in \mathbb{N}^*$,*

1. *If $\mathcal{F} \subset \mathcal{F}'$,*
$$R_n(\mathcal{F}) \leqslant R_n(\mathcal{F}').$$
2. *For every $c \in \mathbb{R}$, let $c\,\mathcal{F} = \{cf : f \in \mathcal{F}\}$,*
$$R_n(c\mathcal{F}) = |c|\ R_n(\mathcal{F}).$$
3. *Let $\mathcal{F} + \mathcal{F}' = \{f + f' : f \in \mathcal{F}, f' \in \mathcal{F}'\}$,*
$$R_n(\mathcal{F} + \mathcal{F}') \leqslant R_n(\mathcal{F}) + R_n(\mathcal{F}').$$
4. *If $\phi : \mathbb{R} \to \mathbb{R}$ is a Lipschitz function with a constant L_ϕ and satisfies $\phi(0) = 0$, then*
$$R_n(\phi \circ \mathcal{F}) \leqslant 2\ L_\phi\ R_n(\mathcal{F}),$$
where $\phi \circ \mathcal{F} = \{\phi(f(.)) : f \in \mathcal{F}\}$.

In the next section, we want to upper bound of the empirical Rademacher complexity of kernel classes.

1.4.3 Empirical Rademacher complexity of the kernel classes

Theorem 1.5. *Let $\kappa : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ be a positive definite, continuous functions. Suppose that κ is a bounded kernel with $\sup_{x\in\mathcal{X}} \sqrt{\kappa(x,x)} = B < \infty$, and let $(H_\kappa, \langle,\rangle_{H_\kappa})$ be its RKHS. For $M > 0$ be fixed, let $\mathcal{F} = \{f \in H_\kappa : \|f\|_{H_\kappa} \leqslant M\}$. Then for any $S = (x_1, \cdots, x_n)$,*

$$R_S(\mathcal{F}) \leqslant \frac{MB}{\sqrt{n}}. \tag{1.10}$$

Proof. Fix $S = (x_1, \ldots, x_n)$. Then

$$
\begin{aligned}
R_S(\mathcal{F}) &= \frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}_n}\left[\sup_{f\in\mathcal{F}} \sum_{i=1}^{n} \sigma_i f(x_i)\right] \\
&= \frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}_n}\left[\sup_{f\in\mathcal{F}} \sum_{i=1}^{n} \sigma_i \left\langle f, \kappa(., x_i)\right\rangle_{H_\kappa}\right] \\
&= \frac{1}{n}\mathbb{E}_{\boldsymbol{\sigma}_n}\left[\sup_{f\in\mathcal{F}} \langle f, \sum_{i=1}^{n} \sigma_i \kappa(., x_i)\rangle_{H_\kappa}\right] \textbf{(linearity of inner product)} \\
&\leqslant \frac{1}{n}\sup_{f\in\mathcal{F}} \|f\|_{H_\kappa} E_{\sigma_n}\left(\left\|\sum_{i=1}^{n} \sigma_i \kappa(., x_i)\right\|_{H_\kappa}\right) \textbf{(Cauchy Schwarz inequality)} \\
&\leqslant \frac{M}{n} E_{\sigma_n}\left(\left\|\sum_{i=1}^{n} \sigma_i \kappa(., x_i)\right\|_{H_\kappa}\right) \\
&\leqslant \frac{M}{n}\left[E_{\sigma_n}\left(\left\|\sum_{i=1}^{n} \sigma_i \kappa(., x_i)\right\|_{H_\kappa}^2\right)\right]^{\frac{1}{2}} \textbf{(Jensen's inequality)} \\
&= \frac{M}{n}\left[E_{\sigma_n}\left(\left\langle \sum_{i=1}^{n} \sigma_i \kappa(., x_i), \sum_{j=1}^{n} \sigma_j \kappa(., x_j)\right\rangle_{H_\kappa}\right)\right]^{\frac{1}{2}} \\
&= \frac{M}{n}\left[E_{\sigma_n}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} \sigma_i\sigma_j \left\langle \kappa(., x_i), \kappa(., x_j)\right\rangle_{H_\kappa}\right]\right]^{\frac{1}{2}} \\
&= \frac{M}{n}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} \kappa(x_i, x_j) E_{\sigma_n}\left[\sigma_i\sigma_j\right]\right]^{\frac{1}{2}} \\
&= \frac{M}{n}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} \kappa(x_i, x_j)\delta_{i,j}\right]^{\frac{1}{2}} \textbf{(where } \delta_{i,j} \textbf{ is the Kronecker symbol)} \\
&= \frac{M}{n}\left[\sum_{i=1}^{n} \|\kappa(., x_i)\|_{H_\kappa}^2\right]^{\frac{1}{2}} \quad (E_{\sigma_n}\left[\sigma_i\sigma_j\right] = 0, i \neq j) \\
&= \frac{M}{n}\left[\sum_{i=1}^{n} \kappa(x_i, x_i)\right]^{\frac{1}{2}} \textbf{(reproducing property 2.4)} \\
&\leqslant \frac{M}{n}\sqrt{nB^2} \\
&= \frac{MB}{\sqrt{n}}.
\end{aligned}
$$

Inequality 1.5 has been established, which concludes the proof. □

Note. $\frac{M}{n}\sqrt{\sum_{i=1}^{n}\kappa(x_i,x_i)} = \frac{M}{n}\sqrt{trace(K)}$, where K is the kernel matrix with entries $K_{ij} = \kappa(x_i,x_j)$.

The proof of this theorem 1.5 will help us in the proof of the lemma 3.1 of chapter 3.

Bibliography

[1] D. Aha, D. Kibler, and M. Albert, Instance-based learning algorithms, Machine learning, 6(1):37-66, 1991.

[2] D. Aha, Lazy learning: Special issue editorial. Artificial Intelligence Review, 11:7- 10, 1997.

[3] D. Wettschereck, D. Aha, and T. Mohri, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, Artificial Intelligence Review, 11(1-5):273-314, 1997.

[4] C. Bishop, Pattern recognition and machine learning, Springer, 2007.

[5] C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1996.

[6] S. Haykin, Neural Networks and Learning Machines, Prentice Hall, 2008.

[7] T. Ho, The Random Subspace Method for Constructing Decision Forests. IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (8): 832844, 1998.

[8] Rao, C. The use and interpretation of principal component analysis in applied research. Sankhya A 26, 329–358, 1964.

[9] J. Quinlan, C4.5: Programs for machine learning, Morgan-Kaufmann Publishers, 1993.

[10] J. R. Quinlan, Introduction of decision trees, Machine learning, 1(1):81-106,1986.

[11] L. Breiman, Classification and regression trees. CRC Press, 1993.

[12] C. Cortes, V. Vapnik. Support-vector networks. Machine Learning, 20(3): 273, 1995.

[13] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel based Learning Methods, Cambridge University Press, 2000.

[14] L. Hamel, Knowledge Discovery with Support Vector Machines, Wiley, 2009.

[15] B. Schölkopf, A. J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, Cambridge University Press, 2001.

[16] I. Steinwart, A. Christmann, Support Vector Machines, Springer, 2008.

[17] V. Vapnik, and A. Chervonenkis, A note on one class of perceptrons. Automation and Remote Control, 25, 1964.

[18] L. Bottou, Large-Scale Machine Learning with Stochastic Gradient Descent, Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010), 177-187, 2010.

[19] O. Chapelle, V. Vapnik, Model selection for support vector machines, Advances in Neural Information Processing Systems, 12, ed. S.A. Solla, T.K. Leen and K.-R. Muller, MIT Press, 2000.

[20] N. Couellan, W. Wang, Bi-level Stochastic Gradient for Large Scale Support Vector Machine, Neurocomputing, vol. 153, pp. 300-308, 2014.

[21] N. Cristianini, C. Campbell, J. Shawe-Taylor, Dynamically adapting kernels in support vector machines, Advances in Neural Information Processing Systems, 11, edM. Kearns, S. A. Solla, and D. Cohn, MIT Press, pp. 204-210, 1999.

[22] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel based Learning Methods, Cambridge University Press, 2000.

[23] C. Hsieh, K. Chang, C. Lin, S. Keerthi, S. Sundararajan, A dual coordinate descent method for large-scale linear SVM, In ICML, 2008.

[24] S. Lee, S.J. Wright, Sparse Nonlinear Support Vector Machine via Stochastic Approximation, University of Wisconsin Report, 2010.

[25] A. K. Menon, Large-Scale Support Vector Machines: Algorithms and Theory. Research Exam, University of California, San Diego, 2009.

[26] B. Schölkopf, A. Smola, Learning with Kernels, MIT, Cambridge, 2002.

[27] Boser, Bernhard E., Isabelle M. Guyon, and Vladimir N. Vapnik. "A training algorithm for optimal margin classifiers." Proceedings of the fifth annual workshop on Computational learning theory. ACM, 1992.

[28] Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." Machine learning 20.3 (1995): 273-297.

[29] Cornuéjols, Antoine. "Une nouvelle méthode d'apprentissage: Les SVM. Séparateurs à vaste marge." Bulletin de l'AFIA 51 (2002): 14-23.

[30] Hasan, Mohamadally, and Fomani Boris. "Svm: Machinesa vecteurs de support ou séparateursa vastes marges." Rapport technique, Versailles St Quentin, France. Cité (2006): 64.

[31] Vapnik, Vladimir. "Pattern recognition using generalized portrait method." Automation and remote control 24 (1963): 774-780.

[32] Moreira, Miguel, and Eddy Mayoraz. "Improved pairwise coupling classification with correcting classifiers." European conference on machine learning. Springer, Berlin, Heidelberg, 1998.

[33] J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, May, 1998.

[34] B. Schölkopf, C. Burges, and V. Vapnik. Extracting supp ort data for a given task. In KDD'95 , pages 252-257, 1995.

[35] V.N. Vapnik. The Nature of Statistical Learning Theory . Springer-Verlag, New York, 1995.

[36] Rifkin, Ryan, and Aldebaro Klautau. "In defense of one-vs-all classification." Journal of machine learning research 5.Jan (2004): 101-141.

[37] Friedman, Jerome H. "Another approach to polychotomous classification." Technical Report, Statistics Department, Stanford University (1996).

[38] S. Knerr, L. Personnaz, and G. Dreyfus. "Single-layer learning revisited: A stepwise procedure for building and training a neural network." Neurocomputing. Springer, Berlin, Heidelberg, pages 41-50, 1990.

[39] J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In Esann'99, pages 219-224, 1999.

[40] J Mercer, B. A. "XVI. Functions of positive and negative type, and their connection the theory of integral equations." Phil. Trans. R. Soc. Lond. A 209.441-458 (1909): 415-446.

[41] L. Györfi , M. Kohler, A. Krzyzak and H. Walk. A distribution-free theory of nonparametric regression. Springer Science and Business Media, 2006.

[42] L. Devroye, L. Györfi, and G. Lugosi. A Probabilistic Theory of Pattern Recognition, volume 31 of Applications of Mathematics (New York). Springer-Verlag, New York, 1996.

[43] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. Journal of Machine Learning Research, 2:265-292, 2001.

[44] Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines: Theory and ap- plication to the classification of microarray data and satellite radiance data. Journal of the American Statistical Association, 99(465); 67-81, 2004.

[45] Y. Guermeur and E. Monfrini. A quadratic loss multi-class SVM for which a radius-margin bound applies. Informatica, 22(1):73-96, 2011.

[46] N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. Journal of the ACM, 44(4):615-631, 1997.

[47] R.M. Dudley, E. Giné, and J. Zinn. Uniform and universal Glivenko-Cantelli classes. Journal of Theoretical Probability, 4(3):485-510, 1991.

[48] M.J. Kearns and R.E. Schapire. Efficient distribution-free learning of probabilistic concepts. Journal of Computer and System Sciences, 48(3):464-497, 1994.

[49] R.C. Williamson, A.J. Smola, and B. Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. IEEE Transactions on Information Theory, 47(6):2516-2532, 2001.

[50] Y. Liu. Fisher consistency of multicategory support vector machines. In Eleventh International Conference on Artificial Intelligence and Statistics, pages 289296, 2007.

[51] R.M. Dudley, E. Gin, and J. Zinn. Uniform and universal Glivenko-Cantelli classes. Journal of Theoretical Probability, 4(3) :485510, 1991.

[52] A.N. Kolmogorov, and V.M Tihomirov. "ϵ-entropy and ϵ-capacity of sets in function spaces." American Mathematical Society Translations series 2, 17:277-364, 1961.

[53] V. Vladimir, and A. Chervonenkis. "Necessary and sufficient conditions for the uniform convergence of means to their expectations." Theory of Probability and Its Applications 26.3 (1982): 532-553.

[54] Alon, Noga, et al. "Scale-sensitive dimensions, uniform convergence, and learnability." Journal of the ACM (JACM) 44.4 (1997): 615-631.

[55] Bousquet, O. (2002). Concentration inequalities and empirical processes theory applied to the analysis of learning algorithms.

Chapter 2

Background: an Overview of Kernel Methods

Abstract In this chapter, we provide the fundamental theory of kernel methods. Firstly, we give a short introduction of kernel method. We then recall the basic concepts necessary for the kernel theory. We will present the positive semidefinite property of kernel function and kernel matrix, and we give some operations on kernel function which will provide a new kernel. Finally we introduce some popular kernels in applications.

2.1 Introduction

In machine learning, kernel methods are a class of pattern recognition algorithms, whose best known member is the support vector machine (SVM). The idea of kernel trick is to transform the representation space of input data into a higher dimension space, where a linear classifier can be used and obtain good performance. The linear discrimination in high-dimensional space (also called feature space) is equivalent to a non-linear discrimination in the original space. However, the computation of the inner product in the feature space can be calculated by the kernel function.

A **kernel** is a two-argument real-valued function over $\mathcal{X} \times \mathcal{X}$ ($\kappa : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$) such that for any $x, y \in \mathcal{X}$

$$\kappa(x, y) = \langle \Phi(x), \Phi(y) \rangle_H \tag{2.1}$$

for some inner-product space H such that Φ is a mapping from $\mathcal{X}$ to a feature space H

$$\Phi : \mathcal{X} \longrightarrow H \tag{2.2}$$

Kernel methods have achieved great success and should clarify the following aspects:

- Data points are mapped from an input space to a higher-dimensionnal feature space
- It is not necessary to know the coordinates in the feature space, we could be obtained by similarity information which are the inner products.
- Pairwise inner products can be calculated by the kernel function.
- The initial problem is anticipated to be a linear problem in the feature space even though it is not linear in the input space.

This can be illustrated by the Figure 2.1.

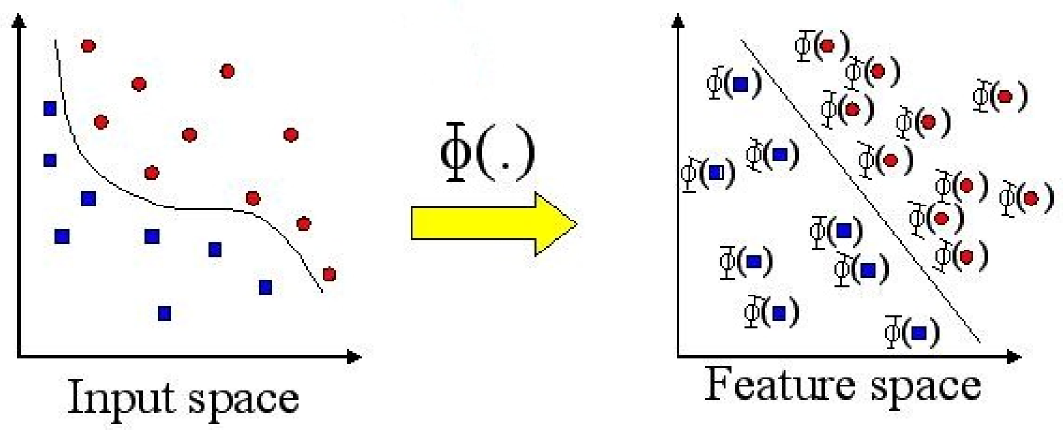


Figure 2.1: The initial data are transformed from the input space, by the transformation Φ, to a large space where the nonlinear problem becomes linear.

Example 2.1.1. *In this example, we consider a feature space two-dimension input space* $\mathcal{X} \subseteq \mathbb{R}^2$ *and its corresponding feature map*

$$\Phi : x = (x_1, x_2) \longrightarrow \Phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2) \in H \subseteq \mathbb{R}^3$$

We can compute the inner product between two points in the feature space, then we get

$$\begin{aligned}\langle \Phi(x), \Phi(y) \rangle_H &= \langle (x_1^2, x_2^2, \sqrt{2}x_1x_2), (y_1^2, y_2^2, \sqrt{2}y_1y_2) \rangle_H \\ &= x_1^2y_1^2 + x_2^2y_2^2 + 2x_1x_2y_1y_2 \\ &= (x_1y_1 + x_2y_2)^2 \\ &= \langle x, y \rangle_H^2.\end{aligned}$$

So the kernel function is

$$\kappa(x, y) = \langle \Phi(x), \Phi(y) \rangle_H = \langle x, y \rangle_H^2.$$

2.2 Fundamental elements of the theory of kernel functions

This section presents the definitions and properties essential to understanding the theory of kernel. To well comprehend the kernel theory, the following fundamental elements need to be presented.

2.2.1 Hilbert space

Definition 2.1. ***(Inner Product Space)*** *An inner product space $\mathcal{X}$ is a vector space with an associated inner product $\langle ., . \rangle : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ that satisfies:*

1. *Symmetry*

$$\langle x, y \rangle = \langle y, x \rangle \quad \forall x, y \in \mathcal{X}.$$

2. *Bilinearity*

$$\langle \alpha x + \beta y, z \rangle = \alpha \langle x+, z \rangle + \beta \langle y, z \rangle \quad \forall x, y, z \in \mathcal{X}, \ \forall \alpha, \beta \in \mathbb{R}.$$

3. *Positive Semi-Definiteness*

$$\langle x, x \rangle \geqslant 0 \quad \forall x \in \mathcal{X}.$$

The inner product space is strict if

$$\langle x, x \rangle = 0 \Leftrightarrow x = 0.$$

Note.

- A strict inner product space $\mathcal{X}$ has a natural norm given by $\|x\|_2 = \sqrt{\langle x, x \rangle}$. The associated metric is $d(x, y) = \|x - y\|_2$.

- The space $\mathbb{R}^d$ has the inner product $\langle x, y\rangle = x^T y$ which yields the **Euclidien norm:**

$$\|x\|_2^2 = \sum_{i=1}^{d} x_i^2.$$

We now give the definition of Hilbert space.

Definition 2.2. *A strict inner product space H is a Hilbert space if it is*

- ***Complete:*** *Every Cauchy sequence $\{h_i \in H\}_{i=1}^{\infty}$ such that*

$$\lim_{n \longrightarrow \infty} \sup_{m>n} \|h_n - h_m\| = 0.$$

- ***Separable:*** *There is a countable subset $\hat{H} = \{h_i \in H\}_{i=1}^{\infty}$ such that for all $h \in H$ and $\epsilon > 0$, there exists $h_i \in \hat{H}$ such that*

$$\|h_i - h\| < \epsilon.$$

Example 2.2.1. ***(Hilbert Space Examples:)*** *We give here several examples of Hilbert space*

- $\langle x, y\rangle = x^T y.$
- $\langle x, y\rangle = \sum_{i=1}^{\infty} x_i y_i.$
- *Inner-product generalized as*

$$\langle f, g\rangle = \int_{\mathcal{X}} f(x) g(x) dx,$$

where functions f satisfy $\int_{\mathcal{X}} f(x)^2 dx < \infty$.

2.2.2 Reproducing Kernel Hilbert Space

The definition of a Reproducing Kernel Hilbert Space (RKHS) calls for the following concepts that are essential in kernel definition.

Definition 2.3. ***(Positive Semi-Definite Matrix)*** *Matrix $\boldsymbol{A}$ is positive semi-definite (PSD) [1] if all its eigenvalues are non-negative $(\forall\ i\ \lambda_i(\boldsymbol{A}) \geqslant 0)$, i.e., for all $x \in \mathcal{X}$:*

$$x^T \boldsymbol{A} x \geqslant 0.$$

Definition 2.4. ***(Positive Definite Matrix)*** *Matrix $\boldsymbol{A}$ is positive definite if all its eigenvalues are positive $(\forall\ i\ \lambda_i(\boldsymbol{A}) > 0)$, i.e., $\boldsymbol{A}$ is PSD and*

$$x^T \boldsymbol{A} x = 0 \Leftrightarrow x = 0.$$

Definition 2.5. *A symmetric function $f : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ is positive semi-definite if*

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j f(x_i, x_j) = \alpha^T F\alpha \geqslant 0,$$

$\forall\ (\alpha_1, \cdots, \alpha_n) \in \mathbb{R}^n, \forall\ (x_1, \cdots, x_n) \in \mathcal{X}^n.$

In the following, we introduce the main property of reproducing kernel Hilbert spaces:

Definition 2.6. ***(Reproducing Kernel Function [3])*** *$\kappa(.,.)$ is a* ***reproducing kernel*** *[2] of a Hilbert space H if*

$$f(x) = \langle f, \kappa(x,.)\rangle_H,\ \forall f \in H. \tag{2.3}$$

Further, the space is called a Reproducing Kernel Hilbert Space (RKHS).

Remark. A reproducing kernel κ is finitely positive semi-definite function (FPSD).

Proof. The reproducing property is given by:

$$\forall x, y \in \mathcal{X},\ \langle \kappa(x,.), \kappa(y,.)\rangle_H = \kappa(x, y). \tag{2.4}$$

$\forall\ \alpha_1, \cdots, \alpha_n \in \mathbb{R},\ \forall\ x_1, \cdots, x_n \in \mathcal{X}$ we have:

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\kappa(x_i, x_j) &= \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\langle \kappa(x_i,.), \kappa(x_j,.)\rangle_H \\ &= \langle \sum_{i=1}^{n}\alpha_i\kappa(x_i,.), \sum_{j=1}^{n}\alpha_j\kappa(x_j,.)\rangle_H \\ &= \|\sum_{i=1}^{n}\alpha_i\kappa(x_i,.)\|_H^2 \geqslant 0.\end{aligned}$$

□

Definition 2.7. ***(Reproducing Kernel Hilbert Space [5])*** *A Reproducing Kernel Hilbert Space is a Hilbert space with a reproducing kernel, where the evaluation functions $f(.)$ are bounded, i.e.*

$$\exists M > 0 : |\delta_x(f)| = |f(x)| \leqslant M\|f\|_H.$$

2.2.3 Characterization of kernels

In this section, we characterize the kernel function using another way, which also allows us to build new kernels.

Theorem 2.1. *[4] $\kappa : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ is finitely positive semi-definite (FPSD) if and only there exists a Hilbert space H with feature map $\Phi : \mathcal{X} \longrightarrow H$ such that*

$$\kappa(x, y) = \langle \Phi(x), \Phi(y) \rangle_H.$$

Theorem 2.2. ***(Moore-Aronszajn Theorem)***

If κ is symmetric and positive definite kernel on a set $\mathcal{X}$. Then there is a unique Hilbert space of functions on $\mathcal{X}$ for which κ is a reproducing kernel.

Theorem 2.3. ***(Mercer Theorem)*** *Let κ is a continuous kernel function that takes two variables x and y and map them to a real value such that $\kappa(x, y) = \kappa(y, x)$.*
A kernel is non-negative definite if and only if:

$$\int\int f(x)\kappa(x, y)f(y)dxdy \geqslant 0.$$

In association with a kernel κ, we can define an integral operator T_κ, which, when applied to a function $f(x)$, generates another function:

$$T_\kappa(f(x)) = \int \kappa(x, y)f(y)dy = [T_\kappa\ f](x).$$

The eigenvalues and their correponding eigenfunctions of this operation are defined as:

$$T_\kappa(\phi_i(x)) = \int \kappa(x, y)\phi(y)dy = \lambda_i\phi_i(x).$$

The eigenvalues λ_i are non-negative and the eigenfunctions $\phi_i(x)$ are orthonomal:

$$\int \phi_i(x)\phi_j(x)dx = \delta_{ij}.$$

The eigenfunctions corresponding to the non-zero eigenvalues form a set of basis functions so that the kernel can be decomposed in terms of them:

$$\kappa(x, y) = \sum_{i=1}^{\infty} \lambda_i\phi_i(x)\phi_j(y).$$

2.2.4 Kernel matrix

Given the fundamental concepts mentioned above which are necessary to build the theory of kernel, now we offer a formal definition of the kernel:

Definition 2.8. *$\kappa : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ is a valid kernel if it satisfies the following conditions*

- *κ is symmetric: $\forall x, y \in \mathcal{X}$, $\kappa(x, y) = \kappa(y, x)$.*

- *κ is positive semi-definite.*

Definition 2.9. *A kernel matrix (or Gram matrix) K is the matrix that results from applying κ to all pairs of data points in set $\{x_i\}_{i=1}^n \in \mathcal{X}^n$*

$$K = \begin{pmatrix} \kappa(x_1, x_1) & \kappa(x_1, x_2) & \cdots & \kappa(x_1, x_n) \\ \kappa(x_2, x_1) & \kappa(x_2, x_2) & \cdots & \kappa(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(x_n, x_1) & \kappa(x_n, x_2) & \cdots & \kappa(x_n, x_n) \end{pmatrix}$$

that is, $K_{ij} = \langle \Phi(x_i), \Phi(x_j) \rangle_H = \kappa(x_i, x_j)$.

Remark. The Gram matrix is symmetric since $K_{ij} = K_{ji}$ and it is positive semi-definite.

Proof. For any vector α we have

$$\begin{aligned} \alpha^T K \alpha &= \sum_{i,j=1}^{n} \alpha_i \alpha_j K_{ij} \\ &= \sum_{i,j=1}^{n} \alpha_i \alpha_j \langle \Phi(x_i), \Phi(x_j) \rangle_H \\ &= \langle \sum_{i=1}^{n} \alpha_i \Phi(x_i), \sum_{j=1}^{n} \alpha_j \Phi(x_j) \rangle_H \\ &= \| \sum_{i=1}^{n} \alpha_i \Phi(x_i) \|_H^2 \geqslant 0. \end{aligned}$$

□

This property ensures that we have a valid kernel (positive semidefinite property), which allows us to manipulate the kernels regardless of the feature space.

2.3 Kernel constructions

In the previous section, we have seen that the necessary and sufficient condition for a function to be a reproducing kernel is that it be semi-definite positive. In this section we present some aspects of kernel engineering. More examples and properties can be found in [6, 7]. We will enumerate some properties also called closure properties, which allow us to manipulate kernel functions to create more complex kernels.

Proposition 2.1. ***(Closure Properties of Kernels [8, 9])*** *κ_1 and κ_2 are assumed to be valid kernel functions on $\mathcal{X} \times \mathcal{X}$, let $f : \mathcal{X} \longrightarrow \mathbb{R}$, $\Phi : \mathcal{X} \longrightarrow \mathbb{R}^N$, and κ_3 is a valid kernel on $\mathbb{R}^N \times \mathbb{R}^N$. Then, the function $\kappa : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ which is defined by one of the following expressions for any $x, y \in \mathcal{X}$ is also a valid kernel.*

1. $\kappa(x, y) = \kappa_1(x, y) + \kappa_2(x, y)$.

2. $\kappa(x, y) = a\kappa_1(x, y)$, $\forall a \in \mathbb{R}^+$.

3. $\kappa(x, y) = \kappa_1(x, y)\kappa_2(x, y)$.

4. $\kappa(x, y) = f(x)f(y)$.

5. $\kappa(x, y) = \kappa_3(\Phi(x), \Phi(y))$.

Proof. Let K_1 and K_2 be the $(n \times n)$ Gram matrices of kernels κ_1 and κ_2 respectively and $\alpha \in \mathbb{R}^n$ be any vector. Recall that K is a positive semi-definite matrix if and only if $\forall \alpha$, $\alpha^T K \alpha \geqslant 0$.

1. $K_1 + K_2$ is kernel matrix corresponding to $\kappa_1 + \kappa_2$. We have

$$\alpha^T K \alpha = \alpha^T K_1 \alpha + \alpha^T K_2 \alpha \geqslant 0.$$

So $K_1 + K_2$ is the positive semi-definite, i.e. the sum of two symmetric positive kernel is a valid kernel function.

2. $K = aK_1 \Longrightarrow \alpha^T K \alpha = a.\alpha^T K_1 \alpha \geqslant 0$. Then the validity of a kernel is conserved after multiplication by a positive scalar.

3. Schur product theorem states that the Schur product (Hadamard product) of two positive semi-definite matrix is a also positive semi-definite matrix since the eigenvalues of the product are product of corresponding eigenvalues of the two matrices which are positive. Then the product of two kernel functions is a valid kernel function.

4. Using the feature map $\Phi : x \longrightarrow f(x)$, we have

$$\kappa(x, y) = f(x)f(y) = \langle \Phi(x), \Phi(y) \rangle,$$

thus κ is PSD.

5. Since κ_3 is a kernel, applying it to any set of vectors $\{\Phi(x_i)\}_{i=1}^n$ yields a PSD matrix.

□

Proposition 2.2. ***(Polynomial functions of a kernel output)*** *Given a polynomial $P : \mathbb{R} \longrightarrow \mathbb{R}$ with positive coefficients and let κ_1 is a valid kernel, then the function*

$$\kappa(x, y) = P(\kappa_1(x, y))$$

is a valid kernel.

Proof. The polynomial P is a linear combination of powers of the kernel κ_1 with positive coefficients. Since the powers of κ_1 are products of κ_1 by itself and thus valid kernels, their linear combination is also a valid kernel. □

Proposition 2.3. ***(Exponential function of a kernel output)*** *Let κ_1 is a valid kernel, the function*

$$\kappa(x,y) = \exp(\kappa_1(x,y))$$

is a valid kernel.

Proof. We consider the Taylor series of $\exp(x) = 1 + x + \frac{x^2}{2!} + \cdots$. Thus, it is a limit of polynomials case.
For more details of the proof of propositions 2.2 and 2.3 could be found in [7]. □

We have seen in this subsection some methods for modifying kernel from existing kernels while maintaining properties of symmetry and positivity.

2.4 Basic kernels

In this section, we introduce a family of symmetric, positive definite functions (hence kernels): translation-invariant kernels, and give examples of most popular kernel functions.

2.4.1 Polynomial kernel

Example 2.4.1. *For a polynomial of degree s ($s \in \mathbb{N}^\star$), the polynomial kernel is defined as*

$$\kappa(x,y) = (\langle x,y\rangle + R)^s \quad \forall x,y \in \mathcal{X} = \mathbb{R}^d,$$

where $R \geqslant 0$ is a free parameter trading off the influence of higher-order versus lower-order terms in the polynomial. This kernel is called Inhomogeneous Polynomial kernel. When $R = 0$ the kernel is called homogeneous.

The following kernels are also very popular in applications that are actually special cases of polynomial kernel.

Example 2.4.2. ***(Linear kernels:)*** *The Linear kernel is the simplest kernel function. It is given by the inner product.*

$$\kappa(x,y) = \langle x,y\rangle.$$

The linear kernel is a special case of polynomial kernels with $R = 0$ and $s = 1$. The mapping function Φ is the identity function, i.e. $\Phi(x) = x \quad \forall x \in \mathcal{X}$.

It is obvious to show that the linear kernel is positive definite since

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\kappa(x_i,x_j)=\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\langle x_i,x_j\rangle=\|\sum_{i=1}^{n}\alpha_i x_i\|^2\geqslant 0,$$

by linearity of the dot product.

Example 2.4.3. ***(Quadratic kernels:)*** *Quadratic kernels are widely used in Speech Recognition.*

$$\kappa(x,y)=\langle x,y\rangle^2.$$

2.4.2 Translation-invariant kernels

We consider the class of translation invariant kernel functions which includes the class of radial kernels. This class contains exactly the kernels which can be defined only according to the difference of the kernel arguments. The property of this kernel as follow:

Proposition 2.4. *Let us set $\mathcal{X}=\mathbb{R}^d$ for some $d\geqslant 1$ and define κ as*

$$\forall x,y\in\mathcal{X},\kappa(x,y)=\mathcal{K}(x-y),$$

where $\mathcal{K}:\mathbb{R}^d\longrightarrow\mathbb{R}$.

The function $\mathcal{K}$ is said to be positive definite if the corresponding kernel κ is positive definite. The general form of continuous translation invariant kernels on $\mathbb{R}^d$ was discovered by Bochner [13]. Suppose that $\mathcal{K}$ is continuous, Bochner's theorem states that $\mathcal{K}$ is positive definite if and only if $\mathcal{K}$ is the Fourier transform of a bounded positive measure on $\mathbb{R}^d$ ([14], Theorem 20), that is

$$\mathcal{K}(z)=\int_{\mathbb{R}^d}e^{iw^Tz}\,dV(w),$$

for some bounded positive measure V on $\mathbb{R}^d$.

Several instances of the family of translation-invariant kernels used kernels on $\mathbb{R}^d$ such as:

Example 2.4.4. ***(Gaussian kernel)***

Gaussian kernels have been proposed by Boser, Guyon and Vapnik [10, 11, 12], they are the most used kernels in the field of machine learning. The definition of a gaussian kernel is given in the following definition 2.10.

Definition 2.10. *A Gaussian kernel is defined as*

$$\kappa(x,y)=e^{-\frac{\|x-y\|^2}{2\sigma^2}}, \tag{2.5}$$

where σ is the positive parameter, called the kernel width, which controls the flexibility of gaussian kernels.

Alternatively, it could also be implemented using

$$\kappa(x,y) = e^{-\gamma\|x-y\|^2}. \tag{2.6}$$

This type of kernel corresponds to $\mathcal{K}(z) = e^{-\frac{\|z\|^2}{2\sigma^2}}$ *which is positive definite since* $\mathcal{K}$ *is the characteristic function of a* $\mathcal{N}(0, \frac{1}{\sigma^2} I_d)$ *Gaussian distribution.*

The properties of Gaussian kernel are given as follow:

- As $\|\Phi(x)\|^2 = \kappa(x,x) = 1,\ \forall x \in \mathcal{X}$, then all the points have a norm equal to 1 in the feature space induced by a gaussian kernel.
- As $\langle \Phi(x), \Phi(y) \rangle = \kappa(x,y) > 0,\ \forall x, y \in \mathcal{X}$, then, all the points lie inside the same orthant in feature space.
- The mapping function Φ can not been given explicitly, because the feature space induced by a gaussian kernel is infinite-dimensional.

Example 2.4.5. ***(Exponential Kernel)*** *The Exponential Kernel is closely related to the Gaussian kernel, with only the square of the norm left out.*

$$\kappa(x,y) = e^{-\frac{\|x-y\|}{2\sigma^2}}. \tag{2.7}$$

It is also a translation-invariant kernel with $\mathcal{K}(z) = e^{-\frac{\|z\|}{2\sigma^2}}$.

Example 2.4.6. ***(Laplace kernel)*** *The Laplace kernel is completely equivalent to the exponential kernel, except that it is less sensitive to changes in the sigma parameter.*

$$\kappa(x,y) = e^{-\frac{\|x-y\|}{\sigma}}. \tag{2.8}$$

Here $\mathcal{K}(z) = e^{-\frac{\|z\|}{\sigma}}$ *corresponds to the characteristic function of a random vector* rU*, where* U *is uniform on the unit sphere of* $\mathbb{R}^d$ *and* r *is independent of* U *and follows a Cauchy distribution with density function* $f(t) = \frac{\sigma}{\pi(1+(\sigma t)^2)}$.

Example 2.4.7. ***(Cauchy kernel:)*** *The Cauchy kernel comes from the Cauchy distribution [15]. It is a long-tailed kernel and can be used to give long-range influence and sensitivity over the high dimension space.*

The Cauchy kernel is a parametric kernel (with parameter $\sigma > 0$*) with formula*

$$\kappa(x,y) = \frac{1}{1 + \frac{\|x-y\|^2}{\sigma^2}}\ , \tag{2.9}$$

for every $x, y \in \mathbb{R}^d$.

Example 2.4.8. ***(B-spline kernels:)*** *Let $B_0 = \mathbb{1}_{B_{1,d}}$ denoting the indicator function of the unit ball $B_{1,d}$ of $\mathbb{R}^d$. For every function $f, g : \mathbb{R}^d \longrightarrow \mathbb{R}$, let $f \circledast g$ denote the convolution of f and g, that is $(f \circledast g)(x) = \int_{\mathbb{R}^d} f(x')g(x' - x)\ dx'$. Then, the B-Spline kernel is given by the recursive formula:*

$$\kappa(x, y) = B_{2p+1}(x - y) \quad forallx, y \in \mathbb{R}^d,$$

where $p \in \mathbb{N}$ with B_i is a real-valued function on $\mathbb{R}^d$ such that $B_{i+1} = B_i \circledast B_0$ for each $i \geqslant 0$.

The function B_{2p+1} are positive definite and the kernel κ defines a translation-invariant kernel [16].

Some translation-invariant kernels such as the Gaussian kernel, the exponential kernel, the Laplace kernel and the Cauchy kernel admit actually in the more specific form

$$\kappa(x, y) = f(d(x, y)),$$

where d is a metric on $\mathcal{X}$, and $f : \mathbb{R}^+ \longrightarrow \mathbb{R}$ is a function. Usually, the metric arises from the dot product, $d(x, y) = \|x - y\| = \sqrt{\langle x, y \rangle}$.
These particular kernels are called radial kernels or radial basis function kernels (RBF kernels).

We will present several examples in chapter 4 as the kernels on graphs, fisher kernel, jittering kernels, etc. and we will construct a new kernel based on a Gaussian mixture model. We recall in the following the definition of the gaussian mixture model.

Definition 2.11. ***(Gaussian Mixture Model)*** *A Gaussian Mixture Model (GMM) is a parametric probability density function represented as a weighted sum of Gaussian component densities. The density of a Gaussian mixture model is given by:*

$$f(x) = \sum_{j=0}^{M} \tau_j g_j(x),$$

where τ_j is a mixture proportion such that $\sum_{j=0}^{M} \tau_j = 1$, and $g_j(x)$ is a Gaussian function.

Bibliography

[1] Meyer, Carl D. Matrix analysis and applied linear algebra. Vol. 71. Siam, 2000.

[2] Girosi, F., Jones, M. B., and Poggio, T. (1995). Regularization theory and neural networks architectures. Neural computation, 7(2):219-269.

[3] N. Aronszajn. Theory of reproducing kernels. Transactions of the American Mathematical Society, 68(3):337-404, 1950.

[4] Saitoh, S. (1988). Theory of reproducing kernels and its applications, volume 189. Longman

[5] Haussler, D. (1999). Convolution kernels on discrete structures. Technical report, Citeseer.

[6] V. N. Vapnik. The nature of statistical learning theory. Springer-Verlag New York, Inc., New York, NY, USA, 1995. 12, 17, 21.

[7] J. Shawe-Taylor and N. Cristianini. Kernel methods for pattern analysis. Cambridge University Press, June 2004. 17.

[8] Herbrich, R. (2002). Learning kernel classifiers. MIT Press, Cambridge.

[9] Micchelli, C. A. (1984). Interpolation of scattered data: distance matrices and conditionally positive definite functions. Springer.

[10] B. E. Boser, I. M. Guyon, and V. Vapnik. "A training algorithm for optimal margin classifiers." Proceedings of the fifth annual workshop on Computational learning theory. ACM, 1992.

[11] I. Guyon, B. Boser, and V. Vapnik. "Automatic capacity tuning of very large VC-dimension classifiers." Advances in neural information processing systems. 1993.

[12] V.N. Vapnik. The Nature of Statistical Learning Theory . Springer-Verlag, New York, 1995.

[13] S. Bochner. Monotone functions, Stieltjessche integrale und harmonische analyse. Mathematische Annalen, 108:378-410, 1933.

[14] A. Berlinet and C. Thomas-Agnan. Reproducing Kernel Hilbert Spaces in Probability and Statistics. Springer US, 2004.

[15] J. Basak "A least square kernel machine with box constraints." Pattern Recognition, 2008. ICPR 2008. 19th International Conference on. IEEE, 2008.

[16] V. Vapnik, S. E. Golowich, A. Smola, et al. "Support vector method for function approximation, regression estimation, and signal processing". In: Advances in neural information processing systems (1997), pp. 281-287.

Chapter 3

Hyperbolic Kernel Machine

A new kernel machine for multi-class pattern recognition is introduced: the hyperbolic kernel machine. Its decision boundaries in the feature space are defined by hyperbolic functions. We establish its main statistical properties.

3.1 Introduction

The support vector machine (SVM) [5] is the first and main kernel machine [25] for pattern classification. Over the last two decades, a great many multi-class extensions (M-SVMs) have been introduced (see [6, 4] for a survey). They were assessed in the framework of comparative experimental studies [8, 4]. Their statistical properties and generalization performance were also extensively investigated (see for instance [12, 11]). This chapter introduces a new kernel machine inspired by the M-SVMs. From a geometrical point of view, it is characterized by the fact that its decision boundaries in the feature space are defined by hyperbolic functions. We establish its main statistical properties.

The chapter is organized as follows. Section 3.2 is devoted to the definition of the new machine. Its statistical properties are established in Section 3.3. At last, we draw conclusions and outline our ongoing research in Section 3.4.

3.2 Hyperbolic kernel machine

The new classifier is devised in the following theoretical framework.

3.2.1 Theoretical framework

The learning problems we are interested in are C-category pattern classification problems. Let $\mathcal{X}$ denote the description space and $\mathcal{Y}$ the set of categories. $\mathcal{Y}$ can be identified with the set of indices of the categories, i.e., the set of the integers ranging from 1 to C, hereafter denoted by $[\![1; C]\!]$ (we do not assume any structure in $\mathcal{Y}$). We assume that

$(\mathcal{X}, \mathcal{A}_\mathcal{X})$ and $(\mathcal{Y}, \mathcal{A}_\mathcal{Y})$ are measurable spaces and denote by $\mathcal{A}_\mathcal{X} \otimes \mathcal{A}_\mathcal{Y}$ the tensor-product sigma-algebra on the Cartesian product $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. We make the hypothesis that the link between descriptions and categories can be characterized by an unknown probability measure P on the measurable space $(\mathcal{X} \times \mathcal{Y}, \mathcal{A}_\mathcal{X} \otimes \mathcal{A}_\mathcal{Y})$. Let $Z = (X, Y)$ be a random pair with values in $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, distributed according to P. The only access to P is via an m-sample $\mathbf{Z}_m = (Z_i)_{1 \leqslant i \leqslant m} = ((X_i, Y_i))_{1 \leqslant i \leqslant m}$ made up of independent copies of Z (in short $\mathbf{Z}_m \sim P^m$). In this context of *agnostic learning* [9], a classifier is characterized by a triplet made up of a function class, a decision rule and an inductive principle. We now introduce the new kernel machine through the specification of the corresponding triplet.

3.2.2 Function class and decision boundaries

In the sequel, κ is a real-valued positive type function/kernel [3] on $\mathcal{X}^2$ and $\left(\mathbf{H}_\kappa, \langle \cdot, \cdot \rangle_{\mathbf{H}_\kappa}\right)$ is the corresponding reproducing kernel Hilbert space (RKHS).

Definition 3.1 (Function classes $\tilde{\mathcal{H}}$ and $\mathcal{H}$)**.** *Let κ be a kernel. The function class $\tilde{\mathcal{H}}$ is the class of all real-valued functions $\tilde{h}$ on $\mathcal{X}$ such that*

$$\forall x \in \mathcal{X}, \ \ \tilde{h}(x) = R - \|O - \kappa_x\|_{\mathbf{H}_\kappa},$$

for some $R \in \mathbb{R}_+^*$ and $O \in \mathbf{H}_\kappa$. Then the function class at the basis of a C-category hyperbolic kernel machine is the class $\mathcal{H} = \tilde{\mathcal{H}}^C$.

Definition 3.2 (Decision rule)**.** *For every function $h = (h_k)_{1 \leqslant k \leqslant C} \in \mathcal{H}$, a* decision rule *$dr_h$ is specified in the following way:*

$$\forall x \in \mathcal{X}, \ \ \begin{cases} \left|\underset{1 \leqslant k \leqslant C}{\operatorname{argmax}}\, h_k(x)\right| = 1 \Longrightarrow dr_h(x) = \underset{1 \leqslant k \leqslant C}{\operatorname{argmax}}\, h_k(x) \\ \left|\underset{1 \leqslant k \leqslant C}{\operatorname{argmax}}\, h_k(x)\right| > 1 \Longrightarrow dr_h(x) = * \end{cases}$$

where $|\cdot|$ returns the cardinality of its argument and $*$ stands for a dummy category.

Let the function h of $\mathcal{H}$ be characterized by the vectors $\mathbf{R}_C = (R_k)_{1 \leqslant k \leqslant C} \in \left(\mathbb{R}_+^*\right)^C$ and $\mathbf{O}_C = (O_k)_{1 \leqslant k \leqslant C} \in (\mathbf{H}_\kappa)^C$. It stems from Definitions 3.1 and 3.2 that the boundaries between pairs of categories associated with h are either hyperbolic or linear in the feature space, depending on the value of $R_k - R_l$. Indeed, the formula defining the decision boundary between the categories k and l (for $\{k, l\} \subset [\![1; C]\!]$) is

$$R_k - \|O_k - \kappa_x\|_{\mathbf{H}_\kappa} - R_l + \|O_l - \kappa_x\|_{\mathbf{H}_\kappa} = 0.$$

When $R_k = R_l$, this simplifies into

$$\|O_k\|_{H_\kappa}^2 - \|O_l\|_{H_\kappa}^2 + 2\langle O_l - O_k, \kappa_x \rangle_{\mathbf{H}_\kappa} = 0,$$

meaning that the classifier is linear in $\mathbf{H}_\kappa$. If $R_k \neq R_l$, the boundary is a sheet of a hyperboloid of two sheets, whose foci are O_k and O_l. The nature of the sheet depends on the sign of $R_k - R_l$. For further details, see Appendix .1

3.2.3 Function selection

To perform function selection on the class $\mathcal{H}$, we specify a training problem that consists in minimizing a penalized data-fit term. This calls for the selection of a (margin) loss function. We use the parameterized truncated hinge loss, applied to the margin functions, a choice that bears the advantage to ensure Fisher consistency (see Section 3.3).

Definition 3.3 (Class of margin functions)**.** *Let $\mathcal{G}$ be a class of functions from $\mathcal{X}$ into $\mathbb{R}^C$. For every $g \in \mathcal{G}$, the* margin function *ρ_g is the real-valued function on $\mathcal{Z}$ defined by:*

$$\forall (x, k) \in \mathcal{Z}, \ \ \rho_g (x, k) = \frac{1}{2} \left(g_k (x) - \max_{l \neq k} g_l (x) \right).$$

Then, the class $\rho_\mathcal{G}$ is defined as follows: $\rho_\mathcal{G} = \{\rho_g : \ g \in \mathcal{G}\}$.

Definition 3.4 (Parameterized truncated hinge loss $\phi_{2,\gamma}$)**.** *For $\gamma \in (0, 1]$, the* parameterized truncated hinge loss *$\phi_{2,\gamma}$ is defined by:*

$$\forall t \in \mathbb{R}, \ \ \phi_{2,\gamma} (t) = \mathbb{1}_{\{t \leqslant 0\}} + \left(1 - \frac{t}{\gamma}\right) \mathbb{1}_{\{t \in (0,\gamma]\}}.$$

When using a margin loss function, the behavior of the margin functions outside the interval $[0, \gamma]$ becomes irrelevant to characterize the generalization performance. The idea to exploit this property by means of a combination with a piecewise-linear squashing function can be traced back to [1]. The piecewise-linear squashing function that fits best with $\phi_{2,\gamma}$ is the function π_γ.

Definition 3.5 (Piecewise-linear squashing function π_γ)**.** *For $\gamma \in (0, 1]$, the* piecewise-linear squashing function *π_γ is defined by:*

$$\forall t \in \mathbb{R}, \ \ \pi_\gamma (t) = t\mathbb{1}_{\{t \in (0,\gamma]\}} + \gamma \mathbb{1}_{\{t > \gamma\}}.$$

Thus, when possible, the class $\rho_\mathcal{G}$ is replaced with the following function class.

Definition 3.6 (Function class $\rho_{\mathcal{G},\gamma}$)**.** *Let $\mathcal{G}$ be a class of functions from $\mathcal{X}$ into $\mathbb{R}^C$ and $\rho_\mathcal{G}$ the corresponding class of margin functions. For every pair $(g, \gamma) \in \mathcal{G} \times (0, 1]$, the function $\rho_{g,\gamma}$ from $\mathcal{Z}$ into $[0, \gamma]$ is defined by:*

$$\rho_{g,\gamma} = \pi_\gamma \circ \rho_g.$$

Then, the class $\rho_{\mathcal{G},\gamma}$ is defined as follows:

$$\rho_{\mathcal{G},\gamma} = \{\rho_{g,\gamma} : \ g \in \mathcal{G}\}.$$

With these definitions at hand, the training problem can be defined as follows.

Definition 3.7 (Training problem of the hyperbolic kernel machine)**.** *Let κ be a kernel and $\mathcal{H}$ the function class associated with κ according to Definition 3.1. For $\mathbf{z}_m = (z_i)_{1 \leqslant i \leqslant m} \in \mathcal{Z}^m$, $\gamma \in (0, 1]$ and $\lambda \in \mathbb{R}_+^*$, the* hyperbolic kernel machine *associated with κ, $\mathbf{z}_m$, γ and λ, is obtained by solving the following optimization problem:*

Problem 1.

$$\min_{h \in \mathcal{H}} \left\{ \sum_{i=1}^{m} \phi_{2,\gamma} \circ \rho_h (z_i) + \lambda \left\| \mathbf{R}_C \right\|_2^2 \right\}$$
$$s.t.\ \ \forall k \in [\![1; C]\!],\ O_k \in conv(\{\kappa_{x_i} : y_i = k\}) .$$

Problem 1 is a non-convex optimization problem. There are many methods to solve it, among which: Metropolis-Hastings algorithm, Constrained Optimization by Linear Approximations (cobyla) with R, Solve Optimization problem with Nonlinear Objective and Constraints (solnl) with R, etc...

3.3 Statistical properties

The statistical properties considered in the sequel regard the consistency of the inductive principle and the generalization performance. Central in their formulations are the concepts of risk and margin risk, that we define now.

Definition 3.8 (Risk and margin risk)**.** *Let $\mathcal{G}$ be a class of functions from $\mathcal{X}$ into $\mathbb{R}^C$. The* expected risk *of any function $g \in \mathcal{G}$, $L(g)$, is given by:*

$$L(g) = \mathbb{E}_{(X,Y)\sim P} \left[\mathbb{1}_{\{\rho_g(X,Y) \leqslant 0\}} \right] = P\left(dr_g(X) \neq Y \right).$$

Its empirical risk *measured on the m-sample $\mathbf{Z}_m$ is:*

$$L_m(g) = \mathbb{E}_{Z' \sim P_m} \left[\mathbb{1}_{\{\rho_g(Z') \leqslant 0\}} \right] = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\{\rho_g(Z_i) \leqslant 0\}}$$

(where P_m is the empirical measure supported on $\mathbf{Z}_m$). Given a class of margin loss functions ϕ_γ parameterized by $\gamma \in (0, 1]$, for every (ordered) pair $(g, \gamma) \in \mathcal{G} \times (0, 1]$, the risk with margin γ *of g, $L_\gamma(g)$, is defined as:*

$$L_\gamma(g) = \mathbb{E}_{Z \sim P} \left[\phi_\gamma \circ \rho_g(Z) \right].$$

$L_{\gamma,m}(g)$ designates the corresponding empirical risk, measured on the m-sample $\mathbf{Z}_m$:

$$L_{\gamma,m}(g) = \mathbb{E}_{Z' \sim P_m} \left[\phi_\gamma \circ \rho_g(Z') \right] = \frac{1}{m} \sum_{i=1}^{m} \phi_\gamma \circ \rho_g(Z_i).$$

The first property we establish is Fisher consistency [12].

3.3.1 Fisher consistency

Proposition 3.1. *Let $\mathcal{G}$ be the class of all the functions from $\mathcal{X}$ into $\mathbb{R}^C$. The minimizer g^* of $\mathbb{E}_{(X,Y)}\left[\phi_{2,\gamma} \circ \rho_g(X,Y)\right]$ over $\mathcal{G}$ satisfies the following:*

$$\forall x \in \mathcal{X}, \exists k(x) \in \operatorname*{argmax}_{1 \leqslant k \leqslant C} P(Y = k \mid X = x): \ \rho_{g^*}(x, k(x)) \geqslant \gamma.$$

Proof. By disintegration (see Lemma 1.2.1 in [5]), there exists a map $x \mapsto P(\cdot \mid x)$ from $\mathcal{X}$ into the set of all probability measures on $\mathcal{Y}$ such that P is the joint distribution of $(P(\cdot \mid x))_{x \in \mathcal{X}}$ and of the marginal distribution $P_{\mathcal{X}}$ of P on $\mathcal{X}$. Consequently,

$$\begin{aligned}
\mathbb{E}_{(X,Y)}\left[\phi_{2,\gamma} \circ \rho_g(X,Y)\right] &= \int_{\mathcal{X} \times \mathcal{Y}} \phi_{2,\gamma} \circ \rho_g(x,y)\, dP(x,y) \\
&= \int_{\mathcal{X}} \left\{ \sum_{k=1}^{C} \phi_{2,\gamma} \circ \rho_g(x,k)\, P(Y = k \mid X = x) \right\} dP_{\mathcal{X}}(x) \\
&= \mathbb{E}_X \left[\sum_{k=1}^{C} \phi_{2,\gamma} \circ \rho_g(X,k)\, P(Y = k \mid X) \right],
\end{aligned}$$

from which it springs that

$$\forall x \in \mathcal{X}, \quad g^* \in \operatorname*{argmin}_{g \in \mathcal{G}} \sum_{k=1}^{C} \phi_{2,\gamma} \circ \rho_g(x,k)\, P(Y = k \mid X = x).$$

Given $x \in \mathcal{X}$ and $g \in \mathcal{G}$, by definition of ρ_g, there is at most one value of k in $[\![1; C]\!]$ such that $\rho_g(x,k) > 0$. Suppose that there is none. Then according to Definition 3.4,

$$\begin{aligned}
\sum_{k=1}^{C} \phi_{2,\gamma} \circ \rho_g(x,k)\, P(Y = k \mid X = x) &= \sum_{k=1}^{C} P(Y = k \mid X = x) \\
&= 1. \qquad (3.1)
\end{aligned}$$

Suppose on the contrary that there exists $k^* \in [\![1; C]\!]$ such that $\rho_g(x, k^*) > 0$. Then

$$\begin{aligned}
\sum_{k=1}^{C} \phi_{2,\gamma} \circ \rho_g(x,k)\, P(Y = k \mid X = x) &= 1 + \left(\phi_{2,\gamma} \circ \rho_g(x,k^*) - 1\right) P(Y = k^* \mid X = x) \qquad (3.2) \\
&< 1. \qquad (3.3)
\end{aligned}$$

It springs from (3.1) and (3.3) that g^* satisfies:

$$\forall x \in \mathcal{X}, \exists!\, k(x) \in [\![1; C]\!]: \ \rho_{g^*}(x, k(x)) > 0.$$

Furthermore, due to (3.2),

$$\forall x \in \mathcal{X}, \begin{cases} k(x) \in \operatorname{argmax}_{1 \leqslant k \leqslant C} P(Y = k \mid X = x) \\ \rho_{g^*}(x, k(x)) \geqslant \gamma \end{cases},$$

so that

$$\sum_{k=1}^{C} \phi_{2,\gamma} \circ \rho_{g^*}(x, k) P(Y = k \mid X = x) = 1 - \max_{1 \leqslant k \leqslant C} P(Y = k \mid X = x).$$

□

We now establish a guaranteed risk for our classifier.

3.3.2 Guaranteed risk

The capacity measure involved in our guaranteed risk is a Rademacher complexity.

Definition 3.9 (Rademacher complexity)**.** *Let $(\mathcal{T}, \mathcal{A}_{\mathcal{T}})$ be a measurable space and let T be a random variable with values in $\mathcal{T}$, distributed according to a probability measure P_T on $(\mathcal{T}, \mathcal{A}_{\mathcal{T}})$. For $m \in \mathbb{N}^*$, let $\mathbf{T}_m = (T_i)_{1 \leqslant i \leqslant m}$ be an m-sample made up of independent copies of T and let $\boldsymbol{\sigma}_m = (\sigma_i)_{1 \leqslant i \leqslant m}$ be a Rademacher sequence. Let $\mathcal{F}$ be a class of real-valued functions with domain $\mathcal{T}$. The* empirical Rademacher complexity *of $\mathcal{F}$ given $\mathbf{T}_m$ is*

$$\hat{R}_m(\mathcal{F}) = \mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^{m} \sigma_i f(T_i) \,\middle|\, \mathbf{T}_m\right].$$

The Rademacher complexity *of $\mathcal{F}$ is*

$$R_m(\mathcal{F}) = \mathbb{E}_{\mathbf{T}_m}\left[\hat{R}_m(\mathcal{F})\right] = \mathbb{E}_{\mathbf{T}_m \boldsymbol{\sigma}_m}\left[\sup_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^{m} \sigma_i f(T_i)\right].$$

Theorem 3.1 (Theorem 5 in [7])**.** *Let $\mathcal{G}$ be a class of functions from $\mathcal{X}$ into $\mathbb{R}^C$. For $\gamma \in (0, 1]$, let $\rho_{\mathcal{G},\gamma}$ be the function class deduced from $\mathcal{G}$ according to Definition 3.6. For a fixed $\gamma \in (0, 1]$ and a fixed $\delta \in (0, 1)$, with P^m-probability at least $1 - \delta$,*

$$\sup_{g \in \mathcal{G}} (L(g) - L_{\gamma,m}(g)) \leqslant \frac{2}{\gamma} R_m(\rho_{\mathcal{G},\gamma}) + \sqrt{\frac{\ln\left(\frac{1}{\delta}\right)}{2m}}, \tag{3.4}$$

where the margin loss function defining the empirical margin risk is the parameterized truncated hinge loss (Definition 3.4).

To upper bound the Rademacher complexity of interest, $R_m(\rho_{\mathcal{H},\gamma})$, we resort to a structural result. The sharpest result of this kind is due to Maurer [13]. It is an improvement of the ones introduced in [10, 11].

With Theorem 3.1 and Lemma 3.1 at hand, deriving a guaranteed risk for the hyperbolic kernel machine boils down to bounding from above $R_m\left(\tilde{\mathcal{H}}\right)$.

Lemma 3.1. *Let $\tilde{\mathcal{H}}$ be a function class satisfying Definition 3.1 with the following restrictions:* $\sup_{x\in\mathcal{X}} \|\kappa_x\| \,|_{\mathbf{H}_\kappa} \leqslant \Lambda_{\mathcal{X}}$, $\sup_{\tilde{h}\in\tilde{\mathcal{H}}} R \leqslant R_{max}$ *and* $\sup_{\tilde{h}\in\tilde{\mathcal{H}}} \|O\|_{\mathbf{H}_\kappa} \leqslant \Lambda_O$. *Then,*

$$R_m\left(\tilde{\mathcal{H}}\right) \leqslant \frac{R_{max}}{2\sqrt{m}} + \frac{1}{2m^{\frac{1}{4}}}\left(1 + \frac{\Lambda_O^2}{2} + \Lambda_{\mathcal{X}}^2 + 2\Lambda_O\Lambda_{\mathcal{X}}\right). \tag{3.5}$$

3.4 Conclusion

We are introduced a novel kernel machine for multi-class pattern recognition. First, the new margin multi category classifier is presented. This classifier is a kernel machine whose separation surfaces are hyperbolic and generalizes the SVM. Then, we exhibited its main statistical properties. We established an upper bound of the Rademacher complexity for this classifier. Finally, we deduced a guaranteed risk for the hyperbolic kernel machine.

3.5 Proof of Lemma 3.1

Proof. Due to the subadditivity of the supremum,

$$\begin{aligned}
\hat{R}_m\left(\tilde{\mathcal{H}}\right) &= \mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\frac{1}{m}\sum_{i=1}^{m}\sigma_i\tilde{h}\left(x_i\right)\right] \\
&\leqslant \frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i R + \sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\left\|O-\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}\right] \\
&= \frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i R\right] + \frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\left\|O-\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}\right]. \qquad (3.6)
\end{aligned}$$

The first Rademacher complexity in the right-hand side of (3.6) can be upper bounded thanks to Lemma 3.2, which gives:

$$\frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i R\right] \leqslant \frac{R_{\max}}{2\sqrt{m}}.$$

To upper bound the second Rademacher complexity, we resort to Lemma 3.3 and choose Φ to be the square root.

This is possible due to the following inequality:

$$\forall\left(u,v,w\right)\in\mathbb{R}_+^3, \quad \left|\sqrt{v}-\sqrt{u}\right| \leqslant \sqrt{w} + \frac{1}{2\sqrt{w}}\left|v-u\right|, \qquad (3.7)$$

which enables us to set $c = m^{-\frac{1}{4}}$ and $L_\Phi = \frac{1}{2}m^{\frac{1}{4}}$ (corresponding to $w = m^{-\frac{1}{2}}$). Then,

$$\frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\left\|O-\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}\right] \leqslant \frac{1}{2m^{\frac{1}{4}}}\left(1+\frac{1}{\sqrt{m}}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\left\|O-\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}^2\right]\right). \qquad (3.8)$$

We now bound the expectation in the right-hand side of (3.8).

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\left\|O-\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}^2\right] \\
&\leqslant \mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\left\|O\right\|_{\mathbf{H}_\kappa}^2 + \sum_{i=1}^{m}\sigma_i\left\|\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}^2 + 2\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\langle O,\kappa_{x_i}\rangle_{\mathbf{H}_\kappa}\right] \\
&\leqslant \frac{\Lambda_O^2}{2}\sqrt{m} + \mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sum_{i=1}^{m}\sigma_i\left\|\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}^2\right] + 2\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\langle O,\kappa_{x_i}\rangle_{\mathbf{H}_\kappa}\right], \qquad (3.9)
\end{aligned}$$

where $\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\left\|O\right\|_{\mathbf{H}_\kappa}^2\right]$ has been bounded from above by means of Lemma 3.2. The first expectation in the right-hand side of (3.9) can be bounded by means of Jensen's inequality, which gives:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sum_{i=1}^{m}\sigma_i\left\|\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}^2\right] &\leqslant \left(\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\left(\sum_{i=1}^{m}\sigma_i\left\|\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}^2\right)^2\right]\right)^{\frac{1}{2}}\\ &\leqslant \sqrt{m}\Lambda_{\mathcal{X}}^2.\end{aligned}$$

At last, the second expectation in the right-hand side of (3.9), associated with a class of linear functions, can be bounded by means of the Cauchy-Schwarz inequality and Jensen's inequality.

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\sum_{i=1}^{m}\sigma_i\langle O,\kappa_{x_i}\rangle_{\mathbf{H}_\kappa}\right] &= \mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{\tilde{h}\in\tilde{\mathcal{H}}}\left\langle O,\sum_{i=1}^{m}\sigma_i\kappa_{x_i}\right\rangle_{\mathbf{H}_\kappa}\right]\\ &\leqslant \Lambda_O\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\left\|\sum_{i=1}^{n}\sigma_i\kappa_{x_i}\right\|_{\mathbf{H}_\kappa}\right]\\ &\leqslant \Lambda_O\Lambda_{\mathcal{X}}\sqrt{m}.\end{aligned}$$

Putting things together gives (3.5), thus concluding the proof. □

3.6 Technical lemmas

Lemma 3.2. *Let $\mathcal{F}$ be the class of constant functions on $\mathcal{T}$ whose values range from* 0 *to $M_{\mathcal{F}}$. Then*

$$\forall m\in\mathbb{N}^*,\ R_m(\mathcal{F})\leqslant\frac{M_{\mathcal{F}}}{2\sqrt{m}}.$$

Proof. Let $\mathbf{t}_m=(t_i)_{1\leqslant i\leqslant m}\in\mathcal{T}^m$.

$$\begin{aligned}R_m(\mathcal{F}) &= \mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{f\in\mathcal{F}}\frac{1}{m}\sum_{i=1}^{m}\sigma_i f(t_i)\right]\\ &\leqslant \frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[M_{\mathcal{F}}\mathbb{1}_{\left\{\sum_{i=1}^{m}\sigma_i>0\right\}}\sum_{i=1}^{m}\sigma_i\right]\\ &= \frac{M_{\mathcal{F}}}{2m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\left|\sum_{i=1}^{m}\sigma_i\right|\right].\end{aligned}\tag{3.10}$$

The expectation in (3.10) can be upper bounded in the classical way using Jensen's inequality and the linearity of the expectation:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\left|\sum_{i=1}^{m}\sigma_i\right|\right] &\leqslant \left(\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\left(\sum_{i=1}^{m}\sigma_i\right)^2\right]\right)^{\frac{1}{2}} \\
&= \left(\sum_{i=1}^{m}\sum_{j=1}^{m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sigma_i\sigma_j\right]\right)^{\frac{1}{2}} \\
&= \left(\sum_{i=1}^{m}\sum_{j=1}^{m}\delta_{i,j}\right)^{\frac{1}{2}} \\
&= \sqrt{m}. \qquad (3.11)
\end{aligned}$$

A substitution of (3.11) into (3.10) concludes the proof. □

Lemma 3.3. *Let $\mathcal{F}$ be a class of real-valued functions on $\mathcal{T}$. If $\Phi : \mathbb{R} \longrightarrow \mathbb{R}$ is a function such that there exist $L_\Phi \in \mathbb{R}_+^*$ and $c \in \mathbb{R}$ satisfying:*

$$\forall (u, v) \in \mathbb{R}^2, \quad |\Phi(u) - \Phi(v)| \leqslant L_\Phi |u - v| + c,$$

then

$$\hat{R}_m(\Phi \circ \mathcal{F}) \leqslant L_\Phi \hat{R}_m(\mathcal{F}) + \frac{c}{2}.$$

The proof is basically that of Talagrand's contraction lemma (see for instance Lemma 4.2 in [14]).

Proof.

$$\begin{aligned}
\hat{R}_m(\Phi \circ \mathcal{F}) =& \ \mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{f\in\mathcal{F}}\frac{1}{m}\sum_{i=1}^{m}\sigma_i\Phi\circ f(t_i)\right] \\
=& \ \frac{1}{2m}\left\{\mathbb{E}_{\boldsymbol{\sigma}_{m-1}}\left[\sup_{f\in\mathcal{F}}\left(\sum_{i=1}^{m-1}\sigma_i\Phi\circ f(t_i) + \Phi\circ f(t_m)\right)\right]\right. \\
&\left. +\mathbb{E}_{\boldsymbol{\sigma}_{m-1}}\left[\sup_{f'\in\mathcal{F}}\left(\sum_{i=1}^{m-1}\sigma_i\Phi\circ f'(t_i) - \Phi\circ f'(t_m)\right)\right]\right\} \\
=& \ \frac{1}{2m}\mathbb{E}_{\boldsymbol{\sigma}_{m-1}}\left[\sup_{(f,f')\in\mathcal{F}^2}\left(\sum_{i=1}^{m-1}\sigma_i\left(\Phi\circ f(t_i) + \Phi\circ f'(t_i)\right) + \Phi\circ f(t_m) - \Phi\circ f'(t_m)\right)\right] \\
\leqslant& \ \frac{c}{2m} + \frac{1}{2m}\mathbb{E}_{\boldsymbol{\sigma}_{m-1}}\left[\sup_{(f,f')\in\mathcal{F}^2}\left(\sum_{i=1}^{m-1}\sigma_i\left(\Phi\circ f(t_i) + \Phi\circ f'(t_i)\right) + L_\Phi\left|f(t_m) - f'(t_m)\right|\right)\right].
\end{aligned}$$

Since f and f' are interchangeable, the absolute value can be removed, so that the inequality simplifies into

$$\hat{R}_m(\Phi \circ \mathcal{F}) \leqslant \frac{c}{2m} + \frac{1}{m}\mathbb{E}_{\boldsymbol{\sigma}_m}\left[\sup_{f\in\mathcal{F}}\left(\sum_{i=1}^{m-1}\sigma_i\Phi\circ f(t_i) + L_\Phi\sigma_m f(t_m)\right)\right].$$

Iterating the process for i equal to $m-1$ down to 1 concludes the proof. □

Proof of Equation 3.7.
For all $w > 0$ as the derivative is always decreasing. Indeed, if we suppose without loss of generality $v \geqslant u$. The formula is obtained by considering the following various cases:

- If, $u < w \leqslant v$, then $\sqrt{v} \leqslant \sqrt{w} + \frac{1}{2\sqrt{w}}(v-w) \leqslant \sqrt{w} + \frac{1}{2\sqrt{w}}(v-u)$. Then, $\sqrt{v} - \sqrt{u} \leqslant \sqrt{w} + \frac{1}{2\sqrt{w}}(v-u)$.
- If, $u \leqslant v < w$, then $\sqrt{v} - \sqrt{u} \leqslant \sqrt{w}$ and thus $\sqrt{v} - \sqrt{u} \leqslant \sqrt{w} + \frac{1}{2\sqrt{w}}(v-u)$.
- Otherwise, if $w \leqslant u \leqslant v$, then $\sqrt{v} - \sqrt{u} \leqslant \frac{1}{2\sqrt{w}}(v-u)$ and thus $\sqrt{v} - \sqrt{u} \leqslant \sqrt{w} + \frac{1}{2\sqrt{w}}(v-u)$.

For our application we will take $w = \frac{1}{\sqrt{m}}$. □

Appendices

.1 Geometric locus

Definition .10 (Hyperbola Definition). *Formally, a hyperbola can be defined geometrically as follows: For two given points, the foci, a hyperbola is a set of points (locus of points) such that the difference between the distances to each focus is constant.*

Shape of the separation surface We will determine the geometric locus of the points such that $h_k(x) - h_l(x) = 0$ to know the type of classifier. There are two cases:

In the first case if $R_k = R_l$ then, $h_k(x) - h_l(x) = \|O_k - \kappa_x\|_{H_\kappa} - \|O_l - \kappa_x\|_{H_\kappa}$. Consequently if, $h_k(x) - h_l(x) = 0$ then we have, $\|O_k - \kappa_x\|_{H_\kappa} - \|O_l - \kappa_x\|_{H_\kappa} = 0$, this implies $\|O_k\|^2_{H_\kappa} - \|O_l\|^2_{H_\kappa} + 2\langle O_l - O_k, \kappa_x\rangle_{H_\kappa} = 0$. Thus, in this case the classifier is linear, and we recognize the classical form of a SVM. Other way, we can find the type of classifier geometrically. The geometric locus of the points such that $h_k(x) - h_l(x) = 0$ is the mediator plane on segment $[O_k O_l]$, so it is a linear classifier.

In the second case if $R_k \neq R_l$, the geometric locus of the points such that $h_k(x) - h_l(x) = 0$ is a hyperbola by definition .10, so in this case it is a non linear classifier.

Eccentricity of a hyperbola We have $\|O_l - \kappa_x\|_{H_\kappa} - \|O_k - \kappa_x\|_{H_\kappa} = R_l - R_k$, then $a = \frac{R_l - R_k}{2}$, where a is the distance between the center of the hyperbola and one of its vertices. The distance c of the foci to the center is called the focal distance, it's given by: $c = \frac{\|O_l - O_k\|_{H_\kappa}}{2}$. Thus, the eccentricity e of a hyperbola is given by: $e = \frac{c}{a} = \frac{\|O_l - O_k\|_{H_\kappa}}{R_l - R_k}$.

.1.1 Geometric locus in dimension 2

Let $(-c, 0)$ and $(c, 0)$ be the foci of a hyperbola centered at the origin. The hyperbola is the set of all points (x, y) such that the difference of the distances from (x, y) to the foci is constant.

If $(a, 0)$ is a vertex of the hyperbola, the distance from $(-c, 0)$ to $(a, 0)$ is $a - (-c) = a + c$. The distance from $(c, 0)$ to $(a, 0)$ is $c - a$. The sum of the distances from the foci to the vertex is

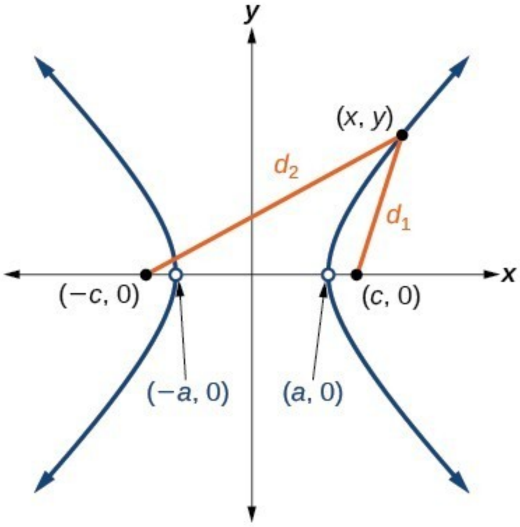
$$(a + c) - (c - a) = 2a.$$

If (x, y) is a point on the hyperbola, we can define the following variables:

$$d2 = \text{the distance from } (c, 0) \ \ to\ (x, y)$$

$$d1 = \ \text{the distance from } (c, 0)\ to\ (x, y)$$

By definition of a hyperbola, $d_2 - d_1$ is constant for any point (x, y) on the hyperbola. We know that the difference of these distances is $2a$ for the vertex $(a, 0)$. It follows that



$d_2 - d_1 = 2a$ for any point on the hyperbola. we will begin by applying the distance formula.

$$d_2 - d_1 = \sqrt{(x-(-c))^2 + (y-0)^2} - \sqrt{(x-c)^2 + (y-0)^2} = \sqrt{(x+c)^2 + y^2} - \sqrt{(x-c)^2 + y^2} = 2a$$

Move radical to opposite side

$\sqrt{(x+c)^2 + y^2} = \sqrt{(x-c)^2 + y^2} + 2a$

Square both sides

$(x+c)^2 + y^2 = (x-c)^2 + y^2 + 4a^2 + 4a\sqrt{(x-c)^2 + y^2}$

$x^2 + 2cx + c^2 + y^2 = x^2 - 2cx + c^2 + y^2 + 4a^2 + 4a\sqrt{(x-c)^2 + y^2}$

$4cx - 4a^2 = 4a\sqrt{(x-c)^2 + y^2}$

Divide by 4

$cx - a^2 = a\sqrt{(x-c)^2 + y^2}$

Square both sides

$(cx - a^2)^2 = a^2\left((x-c)^2 + y^2\right)$

$c^2x^2 - 2a^2cx + a^4 = a^2\left(x^2 - 2cx + c^2 + y^2\right)$

Set $b^2 = c^2 - a^2$

$(c^2 - a^2)x^2 - a^2y^2 = a^2(c^2 - a^2)$

$x^2b^2 - a^2y^2 = a^2b^2$

Divide both sides by a^2b^2

$\dfrac{x^2b^2}{a^2b^2} - \dfrac{a^2y^2}{a^2b^2} = \dfrac{a^2b^2}{a^2b^2}$

$\dfrac{x^2}{a^2} - \dfrac{y^2}{b^2} = 1.$

This equation defines a hyperbola centered at the origin with vertices $(\pm a, 0)$ and co-vertices $(0, \pm b)$.

.1.2 Geometric locus in dimension 3

Let $O_k = (0,0,c)$ and $O_l = (0,0,-c)$ be the foci of a hyperboloid. The hyperboloid of two sheets is the set of all points (x, y, z) such that the difference of the distances from (x, y, z) to the foci is constant.

$\|O_k - \kappa_x\| - \|O_l - \kappa_x\| = R_k - R_l$. We set $K = R_k - R_l$,

$\|O_k - \kappa_x\| = \|O_l - \kappa_x\| + K$

$\sqrt{x^2 + y^2 + (z-c)^2} = \sqrt{x^2 + y^2 + (z+c)^2} + K$

Square both sides

$x^2 + y^2 + (z-c)^2 = x^2 + y^2 + (z+c)^2 + K^2 + 2K\sqrt{x^2 + y^2 + (z+c)^2}$

$x^2 + y^2 + z^2 - 2cz + c^2 = x^2 + y^2 + z^2 + 2cz + c^2 + K^2 + 2K\sqrt{x^2 + y^2 + (z+c)^2}$

Simplify expressions

$-4cz - K^2 = 2K\sqrt{x^2 + y^2 + (z+c)^2}$

Square both sides

$(-4cz - K^2)^2 = \left(2K\sqrt{x^2 + y^2 + (z+c)^2}\right)^2$

$16c^2z^2 + 8cK^2z + K^4 = 4K^2\left(x^2 + y^2 + (z+c)^2\right)$

Divide by 4

$4c^2z^2 + 2cK^2z + \frac{K^4}{4} = K^2\left(x^2 + y^2 + (z+c)^2\right)$

$4c^2z^2 + 2cK^2z + \frac{K^4}{4} = K^2\left(x^2 + y^2 + z^2 + 2cz + c^2\right)$

$-K^2x^2 - K^2y^2 + (4c^2 - K^2)z^2 = K^2c^2 - \frac{K^4}{4} = K^2\left(c^2 - \frac{K^2}{4}\right) = K^2\left(\frac{4c^2 - K^2}{4}\right)$

Set $b^2 = c^2 - \frac{K^2}{4} = \frac{4c^2 - K^2}{4}$

$-K^2x^2 - K^2y^2 + 4b^2z^2 = K^2b^2$

If $K^2 \neq 0$, divide both sides by K^2b^2

$-\frac{1}{b^2}x^2 - \frac{1}{b^2}y^2 + \frac{4}{K^2}z^2 = 1.$

This equation defines a hyperboloid of two sheets. Also, this equation has one positive eigenvalue and two negative eigenvalues.

.1.3 Geometric locus in dimension n

Let $O_k = (0, 0, \cdots, 0, c)$ and $O_l = (0, 0, \cdots, 0, -c)$ be the foci of a hyperboloid, The hyperboloid of two sheets is the set of all points $(x_1, \cdots, x_n)$ such that the difference of the distances from $(x_1, \cdots, x_n)$ to the foci is constant.

$\|O_k - \kappa_x\| - \|O_l - \kappa_x\| = R_k - R_l$. We set $K = R_k - R_l$,

$\|O_k - \kappa_x\| = \|O_l - \kappa_x\| + K$

$\sqrt{x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n - c)^2} = \sqrt{x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2} + K$

Square both sides

$x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n - c)^2 = x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2 + K^2$

$+ 2K\sqrt{x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2}$

$(x_n - c)^2 = (x_n + c)^2 + K^2 + 2K\sqrt{x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2}$

$x_n^2 - 2cx_n + c^2 = x_n^2 + 2cx_n + c^2 + K^2 + 2K\sqrt{x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2}$

Simplify expressions

$-4cx_n - K^2 = 2K\sqrt{x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2}$

Square both sides

$(-4cx_n - K^2)^2 = \left(2K\sqrt{x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2}\right)^2$

$16c^2x_n^2 + 8cK^2x_n + K^4 = 4K^2\left(x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2\right)$

Divide by 4

$4c^2x_n^2 + 2cK^2x_n + \dfrac{K^4}{4} = K^2\left(x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + (x_n + c)^2\right)$

$4c^2x_n^2 + 2cK^2x_n + \dfrac{K^4}{4} = K^2\left(x_1^2 + x_2^2 + \cdots + x_{n-1}^2 + x_n^2 + 2cx_n + c^2\right)$

$-K^2x_1^2 - K^2x_2^2 - \cdots - K^2x_{n-1}^2 + (4c^2 - K^2)x_n^2 = K^2c^2 - \dfrac{K^4}{4} = K^2\left(c^2 - \dfrac{K^2}{4}\right) = K^2\left(\dfrac{4c^2 - K^2}{4}\right)$

Set $b^2 = \dfrac{4c^2 - K^2}{4}$

$-K^2x_1^2 - K^2x_2^2 - \cdots - K^2x_{n-1}^2 + (4c^2 - K^2)x_n^2 - K^2b^2 = 0.$

This equation is a quadratic form. The matrix form of this equation is:

$$\mathbf{A} = \begin{pmatrix} -K^2 & 0 & \cdots & 0 \\ 0 & -K^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 4c^2 - K^2 \end{pmatrix}$$

The eigenvalues of A are: $\lambda_1 = -K^2, \cdots, \lambda_{n-1} = -K^2$ and $\lambda_n = 4c^2 - K^2$.

Thus, there are one positive eigenvalue because $c > \frac{K}{2}$ and $(n-1)$ negative eigenvalues, then the locus of the set of all points $(x_1, \cdots, x_n)$ is a two-sheet hyperboloid.

Bibliography

[1] P.L. Bartlett. The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network. IEEE Transactions on Information Theory, 44(2):525-536, 1998.

[2] A. Berlinet and C. Thomas-Agnan. Reproducing Kernel Hilbert Spaces in Probability and Statistics. Kluwer Academic Publishers, Boston, 2004.

[3] C. Cortes and V.N. Vapnik. Support-vector networks. Machine Learning, 20(3):273-297, 1995.

[4] U. Dogan, T. Glasmachers, and C. Igel. A unified view on multi-class support vector classification. Journal of Machine Learning Research, 17(45):1-32, 2016.

[5] R.M. Dudley. A course on empirical processes. In P.L. Hennequin, editor, Ecole d'Eté de Probabilités de Saint-Flour XII - 1982, volume 1097 of Lecture Notes in Mathematics, pages 1-142. Springer-Verlag, 1984.

[6] Y. Guermeur. A generic model of multi-class support vector machine. International Journal of Intelligent Information and Database Systems, 6(6):555-577, 2012.

[7] Y. Guermeur. L p -norm Sauer-Shelah lemma for margin multi-category classifiers. Journal of Computer and System Sciences, 89:450-473, 2017.

[8] C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. IEEE Transactions on Neural Networks, 13(2):415-425, 2002.

[9] M.J. Kearns, R.E. Schapire, and L.M. Sellie. Toward efficient agnostic learning. Machine Learning, 17(2-3):115-141, 1994.

[10] V. Kuznetsov, M. Mohri, and U. Syed. Multi-class deep boosting. In NIPS 27, pages 2501-2509, 2014.

[11] Y. Lei, U. Dogan, A. Binder, and M. Kloft. Multi-class SVMs: From tighter data-dependent generalization bounds to novel algorithms. In NIPS 28, pages 2026-2034, 2015.

[12] Y. Liu. Fisher consistency of multicategory support vector machines. In Eleventh International Conference on Artificial Intelligence and Statistics, pages 289-296, 2007.

[13] A. Maurer. A vector-contraction inequality for Rademacher complexities. In ALT'16, pages 3-17, 2016.

[14] M. Mohri, A. Rostamizadeh, and A. Talwalkar. Foundations of Machine Learning. The MIT Press, Cambridge, MA, 2012.

[15] B. Schölkopf and A.J. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. The MIT Press, Cambridge, MA, 2002.

Chapter 4

Consolidation Kernel

In this chapter, we introduce a novel kernel function obtained as a Fourier transform of a Gaussian mixture model with the purpose of detecting translation invariance inside classes. It is close to existing kernels but has never been expressed in this precise way. We have applied it successfully on several datasets in the context of machine learning using multiclass support vector machines.

4.1 Introduction

Kernel methods are robust statistical learning techniques [25, 27], widely applied to various learning problems due to their versatility and good performance. Applications are wide and cover all possible structured or unstructured data types, e.g., general discrete structures [12], strings [18], weighted automata [4], etc. The only theoretical constraint is to have a positive type function or symmetric positive semidefinite (PSD) kernel [3], which implicitly specify an inner product in a reproducing kernel Hilbert space (RKHS). Thus, every data analysis algorithm that only makes use of inner products between data vectors can be transformed into a kernel method by the kernel trick, which consists in replacing the inner product by an arbitrary kernel function.

Kernel methods have become popular for various kinds of machine learning tasks, the most famous being the support vector machine (SVM) for classification [5]. SVM with a positive semidefinite kernel matrix has been applied successfully in many classification tasks including image retrieval, face recognition, and micro-array gene expression data analysis ([6, 25, 31]). Furthermore, in practice, it can prove successful even with indefinite kernels [1, 8, 10, 19, 28].

In this chapter, we address the problem of incorporating transformation invariance in a kernel used for classification. We introduce a new kernel constructed from the Fourier transform of a Gaussian mixture function. The underlying principle is that if a translation inside a class cannot be used to make closer points of different classes, then it could be used as a translation invariant. This is achieved by clustering classes into subclasses and reviewing the translations between subclasses centers. The simplest example we have in mind is the XOR case.

The rest of the chapter is organized as follows. In the following section, we review existing work dealing with the kernel design. Section 4.3 presents our kernel. The comparative study allowing its evaluation is the subject of Section 4.4. At last, we draw conclusions in Section 4.5.

4.2 State of the art on the transformation invariant kernels

We give a brief review of existing literature on kernel design. Kernels have been designed for a variety of data: graphs [15, 9, 30], string kernels [18, 14, 23] and hidden Markov models [13, 32], to name just a few. For a good review, we can refer to the article of [16].

When it comes to transformation invariance, the simplest idea is based on the generation of virtual examples [22, 21]. In this approach, new examples are created using the transformation at hand (translation or rotation for example) to enlarge the training set. A variant of it is the virtual support vector method [24]. There, the virtual examples are only generated from the support vectors (that utterly define the boundaries between the categories). The drawback is the enlarged memory and time complexities due to additionnal points.

Very close kernels to the virtual support vector method are the jittering kernels [7, 8], where the transformation invariance is in the kernel itself, for instance $\kappa^*(x, x')$ may be computed from a kernel κ using $T^* = \text{argmin}_{T \in \mathcal{T}} \kappa(x, x) + \kappa(Tx', Tx') - 2\kappa(x, Tx')$, where $\mathcal{T}$ is a transformation group and $\kappa^*(x, x')$ is equal to $\kappa(x, T^*x')$. A similar approach is the tangent distance kernels which rely on the computation of the distance between sets of points R_x and R'_x associated to the original points x and x' and obtained by all possible transformations. This has been originally incorporated in SVMs as TD kernels in [10] and extensively studied in [29] for neural networks.

All these kernels can be generalized by computing an average kernel over all transformations. This gives rise to the Haar-integration kernel [26, 11] defined for a standard kernel κ_0 and a transformation group $\mathcal{T}$, which contains the admissible transformations [**?**, see]for a complete definition]ShulzMirbach1994. The idea is to compute the average of the kernel output $\kappa_0\left(Tx, T'x'\right)$ over all pairwise combinations of the transformed examples $(Tx, T'x')$, $\forall\,(T, T') \in \mathcal{T}^2$. The HI-kernel κ of κ_0 with respect to $\mathcal{T}$ is thus

$$\int_{\mathcal{T}^2} \kappa_0\left(Tx, T'x'\right) \mathrm{d}T \mathrm{d}T',$$

under the condition of existence of the integral.

Finally, in this chapter we will make a particular use of the spectral mixture base kernels [34, 35] in Equation (4.1):

$$\kappa_{SM}(x,x') = \sum_{q=1}^{Q} a_q \frac{|\Sigma_q|^{1/2}}{(2\pi)^{p/2}} \exp\Big\{-\frac{1}{2}\left\|\Sigma_q^{1/2}(x-x')\right\|_2^2\Big\} \cos\left(\langle x-x', 2\pi\mu_q\rangle_2\right). \tag{4.1}$$

The kernel κ_{SM} has been defined for any parameters $\theta = (a_q, \mu_q, \Sigma_q)_{1\leqslant q\leqslant Q}$ thanks to Bochner's theorem and the flexibility of Gaussian mixture models. The underlying idea was to mimick the expressive power of deep learning architectures. Here, we will benefit from its flexibility to express translation invariance in data sets.

4.3 Consolidation kernel

The present work is inspired from a simple observation: a class can be so sparse that it becomes separated into several clusters. The idea is here to design a kernel bringing together these clusters.

We proceed in the following way. First, each class is fragmented into a number of relevant subclasses. Second, we consider the directions given by all the vectors c_1c_2 obtained in the following way:

1. c_1 and c_2 are subclass centers associated with the same class;

2. the vector c_1c_2 does not connect two subclasses of two different classes.

Let $\{c_{i1}c_{i2} : i \in [\![1;M]\!]\}$ be the set of all such vectors.

Along each of them, we want the kernel value to oscillate somehow according to the periodic function h_d defined on $\mathbb{R}$ as follows:

$$\forall k \in \mathbb{Z},\ \forall t \in [0,d),\ \ h_d\,(kd+t) = \frac{4}{d^2}t^2 - \frac{4}{d}t + 1,$$

and depicted in Figure 4.1.

As, $h_d(t)$ is too complex to handle, we resort to a m-order Fourier expansion of h_d which stands as follows:

$$h_d^m(t) = \frac{1}{3} + \sum_{j=1}^{m} a_j \cos\left(\frac{2\pi j t}{d}\right)$$

where a_j are the Fourier coefficients of h_d and are given by $a_j = \frac{4}{j^2\pi^2}$. The details are provided in the appendix. More precisely, our kernel, named consolidation kernel, is defined in the following way:

Definition 4.1. *The* consolidation kernel κ *is defined by:* $\forall (x,x') \in (\mathbb{R}^p)^2$,

$$\kappa(x,x') = \tau_0 \exp\{-\frac{1}{2}\sigma_0^2\|x-x'\|_2^2\} + \sum_{i=1}^{M} \tau_i h_{d_i}\left(\left\langle \frac{c_{i_1}-c_{i_2}}{d_i}, x-x'\right\rangle_2\right) \exp\{-\frac{1}{2}\sigma_i^2\|x-x'\|_2^2\}$$

with $\sum_{i=0}^{M}\tau_i = 1$ *and* $d_i = \|c_{i_1}-c_{i_2}\|_2$.

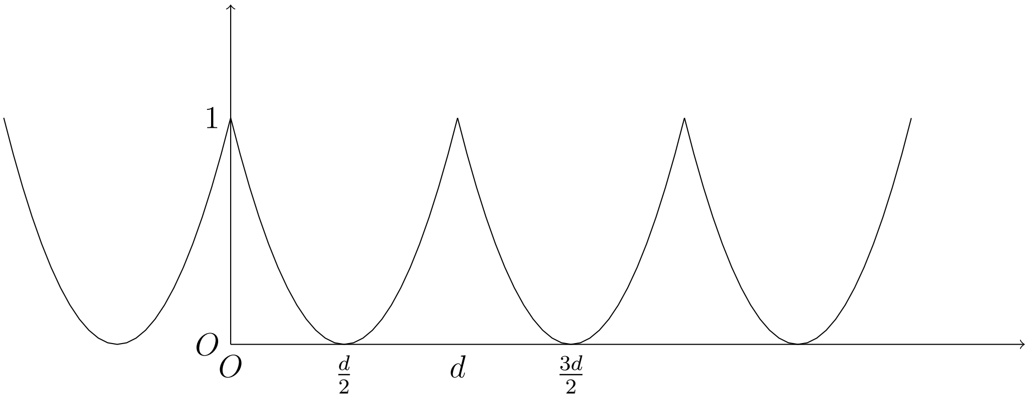


Figure 4.1: Graph of function h_d.

Then, kernel κ is truncated into kernel κ^m in the following way by replacing $h_d(t)$ by $h_d^m(t)$.

Definition 4.2. *The* consolidation kernel κ^m *is defined as:* $\forall (x, x') \in (\mathbb{R}^p)^2$,

$$\kappa^m(x, x') = \tau_0 \exp\{-\frac{1}{2}\sigma_0^2 \|x - x'\|_2^2\} + \sum_{i=1}^{M} \tau_i \sum_{j=0}^{m} a_j \cos\big(\langle \mu_{ij}, x - x' \rangle\big) \exp\{-\frac{1}{2}\sigma_i^2 \|x - x'\|_2^2\},$$

where $a_0, \dots, a_m$ are the first terms of the Fourier series of h_{d_i} and $\mu_{ij} = \frac{c_{i_1} - c_{i_2}}{\|c_{i_1} - c_{i_2}\|_2^2} j2\pi$.

Proposition 4.1 (Properties of the kernel κ)**.**

1. *$k(x, x) = 1$ for all $x \in \mathbb{R}^p$.*
2. *If $M = 1$ and $\tau_0 = 0$, the kernel is translation invariant in the following sense $\kappa(x + \nu_{1j}, y) = \kappa(x, y)$ with $\nu_{1j} = j(c_{1_1} - c_{1_2})$.*

Proof. This is immediate, indeed on one hand we have:

$$\kappa(x, x) = \tau_0 e^0 + \sum_{i=1}^{M} \tau_i h_{d_i}(0) e^0 = 1$$

and on the other hand:

$$\begin{aligned} \kappa(x + \nu_{1j}, y) =& h_{d_1}\Big(\big\langle \frac{c_{1_1} - c_{1_2}}{d_1}, x + \nu_{1j} - y \big\rangle\Big) \\ =& h_{d_1}\Big(jd_1 + \big\langle \frac{c_{1_1} - c_{1_2}}{d_1}, x - y \big\rangle\Big) \\ =& k(x, y) \text{ by periodicity of } h_{d_1}(t) \end{aligned}$$

□

Now, we just have to show that kernel κ^m is close to kernel κ.

Proposition 4.2 (Properties of the kernel k^m)**.**
If $\tau_0 = \cdots = \tau_M = \frac{1}{M+1}$ *then:*

$$|\kappa(x,y) - \kappa^m(x,y)| \leqslant \frac{4}{\pi^2 m}.$$

Proof.

$$\begin{aligned}
|\kappa^m(x,y) - \kappa(x,y)| &= \left| \sum_{i=1}^{M} \tau_i \sum_{j=m+1}^{\infty} a_j \cos(\mu_{ij}^T(x-y)) \exp\{-\frac{1}{2}\sigma_i^2 \|x-x'\|_2^2\} \right| \\
&\leqslant \frac{1}{M+1} \left| \sum_{i=1}^{M} \sum_{j=m+1}^{\infty} \frac{4}{j^2\pi^2} \cos(\mu_{ij}^T(x-y)) \right| \\
&\leqslant \frac{1}{M+1} \sum_{i=1}^{M} \sum_{j=m+1}^{\infty} \frac{4}{j^2\pi^2} \\
&\leqslant \frac{4}{\pi^2 m} \text{ by series integral comparison }.
\end{aligned}$$

□

Consequently, in practice if one wants ϵ lower than 10^{-2}, m should be taken at least equal to 41. By the preceding Proposition 4.2, we deduce that the kernel is not computationally too demanding as the number of terms is linear in M.

4.4 Application

4.4.1 Experimental Setup

The new kernel is assessed in the framework of a comparative study where the reference is provided by the Gaussian kernel. Both kernels are incorporated in a multi-class SVM (M-SVM): the one of [33], hereafter referred to as the WW-M-SVM. The package implementing it is MSVMpack [17]. For each data set and each class, five subclasses are obtained by means of the k-means algorithm.

4.4.2 XOR Problem

To gain insight into the way our method works, we start by studying a toy example: the well-known XOR problem. Then, the number of subclasses can be obviously set to two. In each subclass in the square $[1,9]^2$, 20 points are taken represented by rounds and crosses according to their class. This is the training set which is represented in the first drawing, once training is done, we test the classifier for the grid of 1000000 points represented in Figure 4.2 with the consolidation kernel k^m and the Gaussian kernel. The translation invariance on this example can be clearly observed.

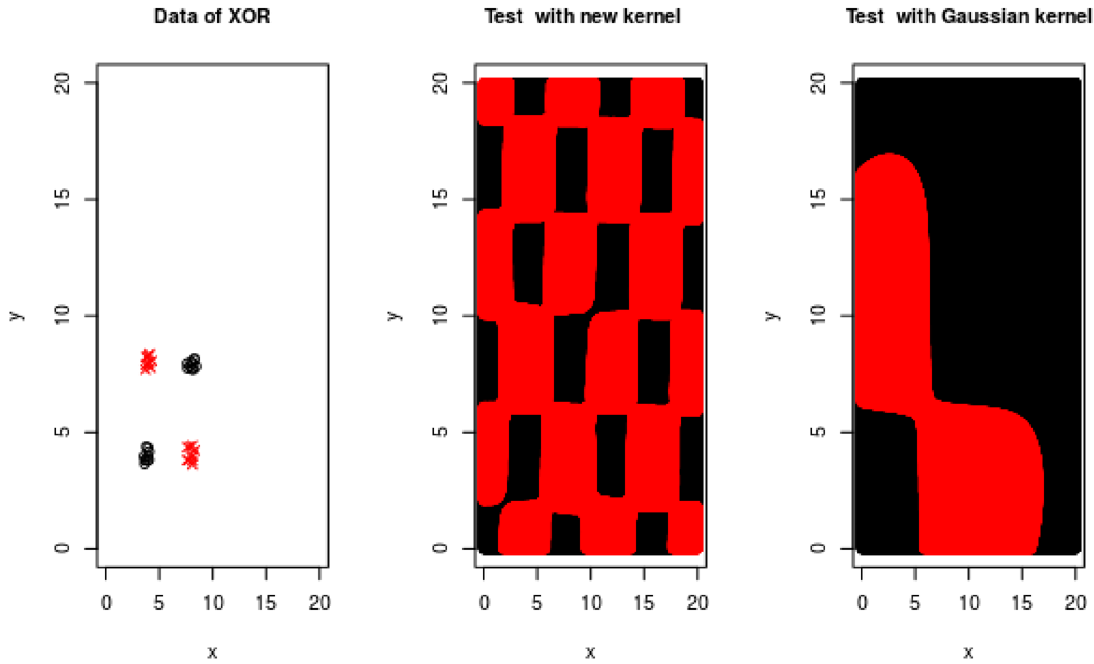


Figure 4.2: WW-M-SVM applied to XOR with both the consolidation kernel and the Gaussian kernel

4.4.2.1 Data Set Description

We have used different data sets in the experiments: Yeast, glass identification, BUPA liver disorders,vertebral column, abalone, madelon, letter recognition, avila, breast, image segmentation, MAGIC Gamma Telescope, credit, EEG Eye Statex, HTRU2, wine and covertype datasets from the UCI machine learning repository [2] and the Banana dataset and Digit Recognizer from Kaggle.com. The USPS-500 data set is a subset of the USPS data provided with the MCSVM[1] software (K. Crammer's own implementation in C of his M-SVM model (CS) named MCSVM).

Each data set is divided into a training set and a test set. These data sets we used in this experiment are described in Table 4.1. In this table, the names of the real data sets are shown with the size of the data sets, number of attributes and number of classes.

[1]http://www.cis.upenn.edu/∼crammer/code-index.html

Datasets	Train	Test	#Attributes	#Classes
Banana	4240	1060	2	2
Yeast	1187	297	8	10
Glass	171	43	9	6
Liver	276	69	6	2
Vertebral Column	248	62	6	2
Abalone	3341	836	8	3
Madelon	2080	520	500	2
Digit Recognizer	42000	28000	784	10
Letter Recognition	16000	4000	16	26
Avila	16693	4174	10	12
USPS-500	500	500	256	10
Breast	92	24	9	2
Segmentation	1848	462	19	7
Magic	15216	3804	10	2
Credit	24000	6000	23	2
Eye	11984	2996	14	2
Htru	14318	3580	8	2
Wine	142	36	13	3
Covertype	522911	58101	54	7

Table 4.1: Information about the UCI data sets and Kaggle used in the experiments.
The results produced by the WW-M-SVM for these data sets are given in Table 4.1. For each data set, we compare in Table 4.2 the training error rate and the recognition rate on the test set for the two kernels, with the parameter of σ given by $\sqrt{5 * \dim(\text{data})}$. We observe that the Gaussian kernel is systematically outperformed.

	Gaussian kernel		Kernel k^m	
Datasets	Training error rate [%]	Recognition rate [%]	Training error rate [%]	Recognition rate [%]
Banana	22.2406	77.64	8.0660	89.91
Yeast	50.7161	47.47	1.4322	53.87
Glass	38.5965	55.81	5.8480	81.40
Liver	23.9130	69.57	23.9130	75.36
Vertebral Column	14.1129	85.48	13.7097	90.32
Abalone	45.4355	52.87	40.3771	54.31
Madelon	22.2596	55.38	8.1250	65.00
Digit Recognizer	4.2083	94.65	2.5167	96.09
Letter Recognition	4.5	92.08	0.00	95.67
Avila	31.9954	68.04	8.9139	80.69
USPS-500	28.6250	66.50	0.00	92.00
Breast	16.3043	58.33	17.3913	66.67
Segmentation	8.1710	93.51	5.3571	95.67
Magic	14.0641	86.17	13.5844	86.88
Credit	17.95	81.95	17.8958	82.05
Eye	31.6923	69.63	30.2904	70.46
Htru	2.0534	97.85	2.0324	97.85
Wine	0.00	100.00	0.00	100.00
Covertype	23.8970	76.11	23.6960	76.70

Table 4.2: Comparison of WW-M-SVM with Gaussian kernel and k^m

To ascertain that the score difference between the Gaussian kernel and the consolidation kernel, we have tested statistically whether the difference in performance of two kernels is significant or not. Let p_1 and p_2 be the recognition rates of the Gaussian kernel and k^m respectively. The hypothesis tests are given by:

$$H_0 : p_1 = p_2 \quad \text{against} \quad H_1 : p_1 \neq p_2$$

The test statistic is calculated by the following formula:

$$T = \frac{p_1 - p_2}{\sqrt{2\ (p_c(1 - p_c)/n}},$$

with

$$p_c = \frac{p_1 + p_2}{2}.$$

Decision rule: Reject H_0 at the level 5% if $|T| > 1.96$, this means that there is a significant difference the two proportions.
The test statistic and p-values are given in the following table for each data sets.

Datasets	Test Statistic (T)	p-value
Banana	7.661837	1.832917e-14
Yeast	1.559955	0.1187704
Glass	2.55671	0.01056674
Liver	0.7613449	0.4464511
Vertebral Column	0.8263025	0.4086325
Abalone	0.590341	0.554962
Madelon	3.168863	0.001530364
Digit Recognizer	8.108306	5.133039e-16
Letter Recognition	6.695475	2.149715e-11
Avila	13.23582	5.449987e-40
USPS-500	6.288275	3.210138e-10
Breast	0.5967618	0.5506664
Segmentation	1.451238	0.1467137
Magic	0.9068353	0.3644939
Credit	0.1425665	0.8866326
Eye	0.7013112	0.4831088
Htru	0	1
Wine	NaN	NaN
Covertype	2.368416	0.01786445

Table 4.3: Test Statistic and p-values of the data sets

So, in 8 cases, the difference is significant and in favour of the consolidation kernel. Besides, for some datasets results can also be found in the literature for sanity check, they are reported in table 4.4.

Datasets	Recognition rate of literature [%]	Recognition rate of k^m [%]
Glass	71.028	81.40
Abalone	27.51	54.31
Digit Recognizer	91.84	96.09
Segmentation	97.576	98.05 $(\sigma = \sqrt{2})$
Wine	98.876	100.00
Covertype	72.40	76.70

Table 4.4: Comparison of WW-M-SVM with literature and k^m

4.5 Conclusion

In this chapter, we have introduced a particular case of the spectral mixture kernel where emphasis has been put on translation invariance. Experimental results have proved that this kernel is worth being used considering in particular that it does not involve much more computation compared to a simple kernel as the Gaussian kernel. Perspectives would involve other ways of adding properties to the kernels and dealing with high-dimensional data.

Proposition .3. *The Fourier series associated to the function of h_d is written as follows:*

$$h_d(t) = \frac{1}{3} + \sum_{j=1}^{+\infty} \frac{4}{j^2\pi^2} \cos(jtw) \ \textit{with}\ w = \frac{2\pi}{d}.$$

Proof. Since the function h_d is even then its coefficients $b_j(h_d) = 0$ and the coefficients $a_0(h_d)$ and $a_j(h_d)$ stand as follows:

$$a_0(h_d) = \frac{1}{d}\int_0^d h_d(t)\,\mathrm{d}t = \frac{1}{3}.$$

$$\begin{aligned} a_j(h_d) = \frac{2}{d}\int_0^d h_d(t)\cos(jtw)\,\mathrm{d}t =& \frac{2}{d}\left[\int_0^d \frac{4}{d^2}l^2\cos(jtw)\,\mathrm{d}t - \int_0^d \frac{4}{d}t\cos(jtw)\,\mathrm{d}t + \int_0^d \cos(jtw)\,\mathrm{d}t\right] \\ =& \frac{8}{d^3}\int_0^d t^2\cos(jtw)\,\mathrm{d}t - \frac{8}{d^2}\int_0^d t\cos(jtw)\,\mathrm{d}t + \frac{2}{d}\int_0^d \cos(jtw)\,\mathrm{d}t \\ =& \frac{8}{d^3}\frac{d^3}{2j^2\pi^2} \\ =& \frac{4}{j^2\pi^2} \end{aligned}$$

□

Bibliography

[1] C. Bahlmann, B. Haasdonk, and H. Burkhardt. Online handwriting recognition with support vector machines-a kernel approach. In Frontiers in handwriting recognition, 2002.

[2] C. Blake, E. Keogh, and C.J. Merz. Uci repository of machine learning databases. 1998.

[3] A. Berlinet and C. Thomas-Agnan. Reproducing Kernel Hilbert Spaces in Probability and Statistics. Kluwer Academic Publishers, Boston, 2004.

[4] C. Cortes, P. Haffner, and M. Mohri. Rational kernel. 2003.

[5] C. Cortes and V.N. Vapnik. Support-vector networks. Machine Learning, 20(3):273-297, 1995.

[6] N. Cristianini and J. Shawe-Taylor. An introduction to support vector machines and other kernel-based learning methods. Cambridge university press, 2000.

[7] D. Decoste and M. Burl. Distortion-invariant recognition via jittered queries. In CVPR, page 1732, 2000.

[8] D. Decoste and B. Schölkopf. Training invariant support vector machines. Machine learning, 46(1-3):161-190, 2002.

[9] T. Gärtner, P. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. Springer, Berlin, Heidelberg, 2003.

[10] B. Haasdonk and D. Keysers. Tangent distance kernels for support vector machines. In Proceedings 16th International Conference on Pattern Recognition, volume 2, 2002.

[11] B. Haasdonk, A. Vossen, and H. Burkhardt. Invariance in kernel methods by haar-integration kernels. In Scandinavian Conference on Image Analysis. Springer, Berlin, Heidelberg, 2005.

[12] D. Haussler. Convolution kernels on discrete structures. Technical Report UCSD-CRL-99-10, Departement of Computer Science, University of California at Santa Cruz, 1999.

[13] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. 1999.

[14] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In European conference on machine learning. Springer, Berlin, Heidelberg, 1998.

[15] R.I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In Proceedings of the 19th international conference on machine learning, 2002.

[16] F. Lauer and G. Bloch. Incorporating prior knowledge in support vector regression. Machine Learning, 70(1):89-118, 2008.

[17] F. Lauer and Y. Guermeur. MSVMpack: a multi-class support vector machine package. Journal of Machine Learning Research, 12:2293-2296, 2011.

[18] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. Journal of machine learning research, pages 419-444, 2002.

[19] P.J. Moreno, P.P. Ho, and N. Vasconcelos. A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. In NIPS 16, pages 1385-1392, 2004.

[20] K. Muandet, K. Fukumizu, B. Sriperumbudur, and B. Schölkopf. Kernel mean embedding of distributions: A review and beyond. Technical report, arXiv:1605.09522v3, 2017.

[21] P. Niyogi, F. Girosi, and T. Poggio. Incorporating prior information in machine learning by creating virtual examples. In Proceedings of the IEEE 86.11, pages 2196-2209, 1998.

[22] T. Poggio and T. Vetter. Recognition and structure from one 2d model view: Observations on prototypes, object classes and symmetries, 1992.

[23] G. Salton, A. Anita Wong, and C.-S. Chung-Shu Yang. A vector space model for automatic indexing. Communications of the ACM, 18(11):613-620, 1975.

[24] B. Schölkopf, C. Burges, and V. Vapnik. Incorporating invariances in support vector learning machines. In International Conference on Artificial Neural Networks. Springer, Berlin, Heidelberg, 1996.

[25] B. Schölkopf and A.J. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. The MIT Press, Cambridge, MA, 2002.

[26] H. Schulz-Mirbach. Constructing invariant features by averaging techniques. In Proceedings of the 12th IAPR International. Conference on Pattern Recognition, volume 2, 1994.

[27] J. Shawe-Taylor and N. Cristianini. Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge, 2004.

[28] H. Shimodaira, K.I. Noma, M. Nakai, and S. Sagayama. Dynamic time-alignment kernel in support vector machine. In Advances in neural information processing systems, 2002.

[29] P. Simard, Y. Le Cun, J. Denker, and B. Victorri. Transformation invariance in pattern recognition, tangent distance and tangent propagation, volume 1524. 1998.

[30] A.J. Smola and R. Kondor. Kernels and regularization on graphs. Springer, Berlin, Heidelberg, 2003.

[31] S. Tong and E. Chang. Support vector machine active learning for image retrieval. In Proceedings of the ninth ACM international conference on Multimedia., 2001.

[32] K. Tsuda, K. Taishin, and A. Kiyoshi. Marginalized kernels for biological sequences. Bioinformatics, 18:S268-S275, 2002.

[33] J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Royal Holloway, University of London, Department of Computer Science, 1998.

[34] A.G. Wilson and R.P. Adams. Gaussian process kernels for pattern discovery and extrapolation. In ICML, 2013.

[35] A.G. Wilson, H. Zhiting, R. Salakhutdinov, and E.P. Xing. Deep kernel learning. In AISTAT, 2016.

Conclusion

This thesis contributes to provide a new kernel machines inspired by the multi-class support vector machine (M-SVM) and a novel kernel function obtained as a Fourier transform of a Gaussian mixture model.

In chapters 3, we are interested in are C-category pattern classification problems, under the assumption that all categories are finite. We introduced a new margin multi category classifier based on classes of vector valued functions with one component function per category. For each description, it provides a score per category and the selected category is the one associated with the highest score. This classifier is a kernel machine whose separation surfaces are hyperbolic and generalizes the SVM. We then established its main statistical properties. We founded an upper bound of the Rademacher complexity of this classifier this establishing by used Ledoux Talagrand Lemma. This bound converges to 0 at infinity, this showed that the classes of component functions are uniform Glivenko-Cantelli. We focus on parameterized truncated hinge loss function and we showed the Fisher consistency of this loss function. Finally, we established a guaranteed risk for our classifier using the bound of the Rademacher complexity.

Furthermore, in chapter 4 we have introduced a new kernel function which obtained by the transform fourier of the gaussian mixture where emphasis has been put on translation invariance. We have applied this kernel on several datasets using multiclass support vector machines package and the results are compared to those obtained with the Gaussian kernel. Then, the experimental results proved that this kernel is worth being used considering in particular that it does not involve much computation in addition to a simple kernel as the Gaussian kernel.

Perspectives

The work presented in this thesis has an interesting potential for future research:

In chapter 3, we have presented a new hyperboloïd kernel machine for multi-class pattern recognition. We introduced an optimization problem of the hyperbolïd kernel machine. We reformulate the Problem 1 to obtain optimization problem with a slack variables ξ_i .

Problem 2.

$$\min_{h \in \mathcal{H}, \xi} \left\{ \lambda \left\| \mathbf{R}_C \right\|_2^2 + \sum_{i=1}^{m} \xi_i \right\}$$

$$\begin{aligned} s.t.\ \forall k \in [\![1;C]\!],\ O_k \in conv\left(\{\kappa_{x_i} : y_i = k\}\right). \\ \forall i \in [\![1,m]\!],\ \left[\frac{1}{\gamma}\rho_h\left(z_i\right)\right]_+ \geq 1 - \xi_i, \\ \xi_i \geqslant 0. \end{aligned}$$

Indeed,

- if $\rho_h\left(z_i\right) \leqslant 0$, then $\xi_i \geqslant 1 \Longrightarrow \xi_i = 1$ and $\phi_{2,\gamma} \circ \rho_h\left(z_i\right) = 1$, therefore $\xi_i = \phi_{2,\gamma} \circ \rho_h\left(z_i\right)$.
- if $\rho_h\left(z_i\right) > \gamma$, then $\xi_i \geqslant 1 - \frac{\rho_h(z_i)}{\gamma}$, so $\xi_i = 0$, therefore $\xi_i = \phi_{2,\gamma} \circ \rho_h\left(z_i\right)$.
- if $\rho_h\left(z_i\right) \in (0, \gamma)$, then $\xi_i \geqslant 1 - \frac{\rho_h(z_i)}{\gamma}$, therefore $\xi_i = 1 - \frac{\rho_h(z_i)}{\gamma} = \phi_{2,\gamma} \circ \rho_h\left(z_i\right)$.

As a perspective, further studies will be needed to solve this optimization problem. We will use the method of Metropolis Hasting.

Furthermore, we will use the consolidation kernel of the chapter 4 on this optimization problem and compare this result with another kernel.