Journal of Machine Learning Research 7 (2003) 501-528 Submitted 07/03; Published ??/0?

Logic, Trees and Kernels

Adam Kowalczyk ADAM@KERNEL-MACHINES.ORG
Telstra Research Laboratories
770 Blackburn Road
Clayton, VIC 3168, Australia

Alexander J. Smola ALEX.SMOLA@ANU.EDU.AU
Machine Learning Group, RSISE
Australian National University
Canberra, ACT 0200, Australia

Robert C. Williamson BOB.WILLIAMSON@ANU.EDU.AU
Research School of Information Sciences and Engineering
Australian National University
Canberra, ACT 0200, Australia

Editor: U.N. Known (joint publication with WWW.KERNEL-MACHINES.ORG)

Abstract

Kernel based methods achieved much of their initial success on problems with real valued attributes. There are many problems with discrete attributes (including Boolean) and in this paper we present a number of results concerning the kernelisation of Boolean and discrete problems.

We give results about the learnability and required complexity of logical formulae to solve classification problems. These results are obtained by linking propositional logic with kernel machines. In particular we show that decision trees and disjunctive normal forms (DNF) can be represented via of a special kernel, which connects the regularised risk to the margin of separation.

Subsequently we derive a number of lower bounds on the required complexity of logical formulae using properties of algorithms for generation of linear machines machines. An interesting side effect of the development is a number of connections between machine learning algorithms that utilize discrete structures (such as a trees) and kernel machines.

We also present some more general kernel constructions on discrete sets using the machinery of frames. These can be used to progressively penalize higher order interactions by explicitly constructing reproducing kernel Hilbert spaces, their associated kernels and the concomitant use of norm-based used regularization.

Keywords: ANOVA Decomposition, Boolean Functions, Decision Trees, Disjunctive Normal Forms, Kernel methods, Fourier Transforms, Frames, Perceptron Learning Algorithm, Regularization, Walsh Functions.

Contents

1. Introduction

Many classical approaches to machine learning restrict the hypotheses used to some specific class a priori for reasons of interpretability. Several questions arise naturally from this practice. For example, when aiming to infer a boolean function from data, the question arises of how many Boolean primitives are needed. Answering this question is computationally hard, especially when the data is noisy. Similarly, when utilizing decision trees, the question of what depth and complexity of a tree is required to learn a certain mapping is also difficult.

In this paper we address the above two questions. We give *lower* bounds on the number of Boolean functions required to learn a mapping. This is achieved by a constructive algorithm which can be carried out in polynomial time. Our tools for this purpose are a Support Vector learning algorithm and a special ANOVA kernel which constructs dot products in the space of Boolean functions. Similarly, we show how to incorporate decision trees directly into the standard kernel machinery thereby further broadening the applicability of kernel methods.

In Section 2 we will define the classes of functions to be studied, and we show that we can treat propositional logic and decision trees within the same framework. Next we prove that in the limit boosted decision trees correspond to polynomial classifiers built directly on the data. Section 3 introduces Boolean kernels in an axiomatic way and Section 4 contains our main result linking the margin of separation to a complexity measure on the class of logical formulae which enables us to devise test procedures concerning the complexity of logical formulae capable of learning a certain datasets. In Section 5 we show how to learn decision trees of various types using kernels and relate the resulting algorithms to the classical CART algorithm.

Part II of the paper considers more general structured data (not just Boolean). In Section 6 we show how the idea of ANOVA decompositions can be extended in order to provide a general regularization framework for discrete attribute learning problems (Section 7). The machinery of frames will allow us to construct kernels efficiently. We show how this allows us to compute kernels on wavelets efficiently. Section 8 concludes.

Previous work on Boolean kernels (Khardon et al., 2002) has focussed on the use of Boolean kernels in online algorithms and in particular upper bounds on the computational complexity.

Part I
Logic and Boolean Kernels

2. Polynomial Representation of Boolean Formulae

We consider the standard supervised learning problem formulation. We have a training set $Z := \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\} \subseteq \mathbf{X} \times Y$. Based on these observations we attempt to find a function $f : \mathbf{X} \to Y$ which "learns" the information given by the training set. The goodness of fit of f to Z is measured with respect to some predefined loss function (Schölkopf and Smola, 2002) or a probabilistic model (Williams, 1998).

In this first part we consider the special case that $\mathbf{X} \subseteq \mathbb{B}^n$ where $\mathbb{B} := \{0, 1\}$ and $Y = \{\pm 1\}$. In other words, we will be considering the learning of a binary function on Boolean variables. To simplify notation we use the shorthand $[a : b] := [a, b] \cap \mathbb{Z}$ to denote the set of integers contained in the closed segment $[a, b]$, i.e. all integers $\geq a$ and $\leq b$.

2.1 Definitions

Definition 1 (Binary Polynomials of degree d: Pol_d) *For* $\mathbf{x}$ *and* $\mathbf{i} = (i_1, ..., i_n) \in \mathbb{B}^n$, *let* $\mathbf{x}^{\mathbf{i}} := x_1^{i_1} \cdots x_n^{i_n}$, *with the usual convention that* $x^0 := 1$ *for every* $x \geq 0$. *Let* $|\mathbf{i}| := i_1 + \cdots + i_n$ *and let* $\mathbb{B}_d^n := \{\mathbf{i} \colon \mathbf{i} \in \mathbb{B}^n \text{ and } |\mathbf{i}| \leq d\}$ *denote the subset of* $\mathbb{B}^n$ *of sequences with at most* d *non-zero entries. The set of all polynomials of degree* $p \leq d \in \mathbb{N}$ *on* $\mathbb{B}^n$ *is*

$$\mathrm{Pol}_d^n := \left\{ \mathbf{x} \mapsto \sum_{\mathbf{i} \in \mathbf{I}} c_{\mathbf{i}} \mathbf{x}^{\mathbf{i}} \colon \mathbf{I} \subset \mathbb{B}_d^n,\ c_{\mathbf{i}} \in \mathbb{R} \right\} \subseteq \mathbb{R}^{\mathbb{B}^n}. \tag{1}$$

In order to avoid further notation we always assume in such expansions that $c_{\mathbf{i}} \neq 0$ for all $\mathbf{i} \in \mathbf{I}$ unless we write explicitly $\mathbf{i} \in \mathbb{B}_d^n$ or $\mathbf{i} \in \mathbb{B}^n$. Note that such binary polynomials are widely used under the name of score tables, e.g. typically loan applications are assesed by financial insitutions by an evaluation of such score tables.

Definition 2 (Disjunctive Normal Forms of degree d: $\mathrm{DNF}_d(X)$) *The subset* $\mathrm{DNF}_d(X) \subset \mathrm{Pol}_d$ *of all polynomials on* X *of the form*

$$f(\mathbf{x}) = -1 + 2 \sum_{\mathbf{i} \in \mathbf{I}} \mathbf{x}^{\mathbf{i}} \text{ where } \mathbf{I} \subset \mathbb{B}_d^n \tag{2}$$

will be called disjunctive normal forms.

An example of such a DNF is

$$f(\mathbf{x}) = -1 + 2x_1x_2x_{42} + 2x_1x_{10} + 2x_{13} \tag{3}$$

which is clearly in $\mathrm{DNF}_3(\mathbb{B}^n)$, where $n \geq 42$. We can see that only for $x_1 = x_2 = x_{42} = 1$, for $x_1 = x_{10} = 1$, or for $x_{13} = 1$ the function $f(\mathbf{x})$ will assume positive values.

For the rest of this chapter we will make the assumption that the subset $X \subset \mathbb{B}^n$ is such that to each $i \in [1 : n]$ there is a uniquely allocated index $i' \in [1 : n]$ such that

$$x_{i'} = \bar{x_i} := 1 - x_i \text{ for all } \mathbf{x} = (x_1, ..., x_n) \in X. \tag{4}$$

This condition is easily satisfied by doubling the dimensionality of $\mathbf{x}$ to include $1 - x_i$ with every x_i. This technical assumption allows for more concise proofs.

Now we recall the standard connection between $\mathrm{DNF}(\mathbb{B}^n)$ and Disjunctive Normal Forms as commonly known in logic, $\mathrm{DNF}_{d,\mathrm{logic}}^n$. The latter consist of all clauses $\phi : \mathbb{B}^n \to \{\mathrm{FALSE}, \mathrm{TRUE}\}$ which can be expressed by disjunctions of terms, each being a conjunctions of up to d logical primitives of the form $[x_i = 1]$ and $[x_i = 0]$.

Lemma 3 *Under assumption (4), for every* $f \in \mathrm{DNF}_d(\mathbb{B}^n)$ *there exists* $\phi \in \mathrm{DNF}_{d,\mathrm{logic}}^n$ *such that*

$$f(\mathbf{x}) = 1 \text{ if and only if } \phi(\mathbf{x}) = \mathrm{TRUE} \quad \text{for every } \mathbf{x} \in X \tag{5}$$

and for every $\phi \in \mathrm{DNF}_{d,\mathrm{logic}}^n$ *there exists* $f \in \mathrm{DNF}_d^n$ *satisfying (5).*

Note the the above correspondence is not necessarily unique, i.e. for every ϕ may exist multiple f satisfying the above requirement, and vice versa.

Proof Let us consider disjunctions

$$\phi(\mathbf{x}) \quad := \quad \bigvee_{(\alpha_i,\beta_j)} [x_{\alpha_1} = 1] \wedge \cdots \wedge [x_{\alpha_s} = 1] \wedge [x_{\beta_1} = 0] \wedge \cdots \wedge [x_{\beta_r} = 0] \tag{6}$$

where the disjunction is over a set of $(s+r)$-tuples of indices $\alpha_1, ..., \alpha_s, \beta_1, ..., \beta_r \in [1:n]$ with $s + r \leq d$. Under assumption (4),

$$\phi(\mathbf{x}) \quad \equiv \quad \bigvee_{(\alpha_i,\beta_j)} [x_{\alpha_1} = 1] \wedge \cdots \wedge [x_{\alpha_s} = 1] \wedge [x_{\beta'_1} = 1] \wedge \cdots \wedge [x_{\beta'_r} = 1]) \tag{7}$$

for every $\mathbf{x} \in X$. For each $(s+r)$-tuple (α_i, β_j) let us allocate a pair $\mathbf{a} = (a_i)_i, \mathbf{b}' = (b'_i)_i$ of elements of $\mathbb{B}^n$ such that $a_i = 1$ iff $i \in \{\alpha_1, ..., \alpha_s\}$ and $b'_j = 1$ iff $j \in \{\beta'_1, ..., \beta'_r\}$. Consider the polynomial $f(\mathbf{x}) := 2\sum_{(\mathbf{a},\mathbf{b})} \mathbf{x}^{\mathbf{a}}\mathbf{x}^{\mathbf{b}'} - 1$. This polynomial belongs to DNF_d since $|\mathbf{a}| + |\mathbf{b}'| \leq d$ and due to the above construction it satisfies (4). Reversing the above construction, for every $f \in \mathrm{DNF}_d$, we find $\phi \in \mathrm{DNF}_{d,\mathrm{logic}}$ satisfying (4). ■

Now we consider another special subclass of polynomials — *decision trees*. They are equivalent to a special class of disjunctive normal forms $\mathrm{DNF}_{d,\mathrm{logic}}$ of the form (6) such that for any given input, exactly one of the conjunctions will be true. It is easy to see that a binary decision tree as used in (Breiman et al., 1984, Quinlan, 1993) is equivalent to an element of $\mathrm{DNF}_{d,\mathrm{logic}}$ with each leaf of the tree corresponding to a monomial in (6), and the depth of the leaf equal to the degree of that monomial.

Definition 4 (Decision Trees of Depth d: $\mathrm{DT}_d(X)$) *Decision trees f of depth d on $X \subseteq \mathbb{B}^n$ are polynomials*

$$f\colon \mathbf{x} \mapsto \sum_{\mathbf{i}\in\mathbf{I}} c_{\mathbf{i}}\mathbf{x}^{\mathbf{i}} \in \mathrm{Pol}^n_d \tag{8}$$

which satisfy

$$|c_{\mathbf{i}}| = 1 \text{ for all } \mathbf{i} \in I \subseteq \mathbb{B}^n_d \text{ and } \sum_{\mathbf{i}\in I} \mathbf{x}^{\mathbf{i}} = 1 \text{ for all } \mathbf{x} \in X. \tag{9}$$

The set of decision trees of depth d on X is denoted $\mathrm{DT}_d(X)$.

It is easy to see that above definition implies $|f(\mathbf{x})| = 1$ for all $f \in \mathrm{DT}_d(X)$ and for all $\mathbf{x} \in X$. Note that we can associate a DNF_d

$$f'(\mathbf{x}) := -1 + 2 \sum_{i\in I, c_{\mathbf{i}}=1} \mathbf{x}^{\mathbf{i}}$$

such that $f(\mathbf{x}) = f'(\mathbf{x})$ for every $\mathbf{x} \in X$ with each decision tree f.

Note that only trivial, i.e. constant, decision trees can exist on $X = \mathbb{B}^n$. Indeed, assume that $\mathbf{x}_1 := (1, 1, \ldots, 1) \in X$. In this case, for all $\mathbf{i}$ the monomials $\mathbf{x}^{\mathbf{i}}_1 = 1$. This contradicts the assumption that of monomials of $\mathrm{DT}_d(X)$ only one of them may fire at the same time, unless $\mathrm{DT}_d(X)$ consists only of the constant functions.

data point	coordinates						labels	
	x_1	x_2	x_3	x_4	x_5	x_6		
1	0	0	1	1	1	0	-1	(FALSE)
2	0	1	1	1	0	0	+1	(TRUE)
3	1	1	0	0	0	1	-1	(FALSE)
4	1	0	1	0	1	0	-1	(FALSE)
5	1	1	1	0	0	0	+1	(TRUE)

Table 1: Truth table

2.2 An Example: Disjunctive Normal Forms

We illustrate the representation of decision trees via Boolean polynomials using an example. Let us consider the set $X \subset \mathbb{B}^6$ consisting of five points as described in the truth table of Table 1.

It is easy to check that X satisfies property (4): $\bar{x_1} = 1 - x_1 = x_4$, $\bar{x_2} = x_5$, $\bar{x_3} = x_6$ (in other words $1' = 4$, $4' = 1$, $2' = 5$, $5' = 2$, $3' = 6$ and $6' = 3$). The last column of Table 1 shows the corresponding allocation of labels.

Two examples of DNFs consistent with this table are

$$\begin{aligned}\phi_1(\mathbf{x}) &:= (x_1 \wedge x_2 \wedge x_3) \vee (\bar{x_1} \wedge x_2 \wedge x_3) \\ &= (x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_2 \wedge x_3) \\ &= \bigvee_{(i_1,\ldots,i_6)\in \mathbf{I}_1} x_1^{i_1} \wedge x_2^{i_2} \wedge \cdots \wedge x_6^{i_6},\end{aligned}$$

where $\mathbf{I}_1 := \{(1,1,1,0,0,0),(0,1,1,1,0,0)\}$ and

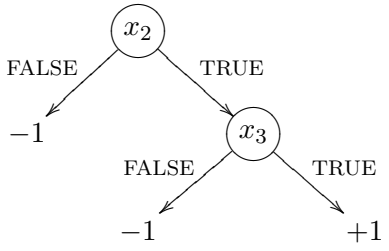
$$\phi_2(\mathbf{x}) := (x_2 \wedge x_3) = \bigvee_{(i_1,\ldots,i_6)\in \mathbf{I}_2} x_1^{i_1} \wedge x_2^{i_2} \wedge \cdots \wedge x_6^{i_6},$$

where $\mathbf{I}_2 := \{(0,1,1,0,0,0)\}$. Corresponding polynomials from DNF_3 are

$$\begin{aligned}f_1(\mathbf{x}) &= 2(x_1x_2x_3 + x_4x_2x_3) - 1 \\ &= 2\mathbf{x}^{(1,1,1,0,0,0)} + 2\mathbf{x}^{(0,1,1,1,0,0)} - 1 = 2\sum_{\mathbf{i}\in\mathbf{I}_1} \mathbf{x}^{\mathbf{i}} - 1\end{aligned}$$

$$f_2(\mathbf{x}) = 2x_2x_3 - 1 = 2\mathbf{x}^{(0,1,1,0,0,0)} - 1 = 2\sum_{\mathbf{i}\in\mathbf{I}_2} \mathbf{x}^{\mathbf{i}} - 1.$$

A binary decision tree $f_3 \in \mathrm{DT}_3(\mathbf{B}^6)$ consistent with the truth table in Table 1 is



It can be represented symbolically as

$$\begin{aligned} f_3(\mathbf{x}) &:= -(1-x_2) - x_2(1-x_3) + x_2x_3 = -x_5 - x_2x_6 + x_2x_3 \\ &= -\mathbf{x}^{(0,0,0,0,1,0)} - \mathbf{x}^{(0,1,0,0,0,1)} + \mathbf{x}^{(0,1,1,0,0,0)}. \end{aligned} \tag{10}$$

2.3 Connections between Propositional Logic and Pol_d^n

In order to use kernels and linear techniques for Boolean problems we need to determine the connection between the sets of boolean functions defined above and the space Pol_d^n of polynomials of degree d on $\mathbb{B}^n$.

Lemma 5 *The set Pol_d^n is a real vector space of dimension*

$$N := \dim(\mathrm{Pol}_d^n) = \sum_{i=0}^{d} \binom{n}{i}. \tag{11}$$

We include a straightforward proof for completeness.

Proof Since every $f \in \mathrm{Pol}_d$ has an expansion (1) in terms of N monomials $\mathbf{x_i}$, $\mathbf{i} \in \mathbb{B}_d^n$, it is sufficient to show that they are linearly independent on $\mathbb{B}^n$. To that end, suppose that

$$f(\mathbf{x}) = \sum_{\mathbf{i}\in\mathbb{B}_d^n} f_{\mathbf{i}}\mathbf{x}^{\mathbf{i}} = 0 \text{ for every } \mathbf{x} \in \mathbb{B}^n \tag{12}$$

and that some coefficients $f_{\mathbf{i}} \neq 0$. Let $\mathbf{i}^*$ be such that $f_{\mathbf{i}^*} \neq 0$ and $|\mathbf{i}^*| = \min\{|\mathbf{i}|\colon f_{\mathbf{i}} \neq 0\}$. However, evaluating f at $\mathbf{x} = \mathbf{i}^* \in \mathbb{B}^n$ and using (12) we get a contradiction: $f_{\mathbf{i}^*} = f(\mathbf{i}^*) = 0$. This implies that if (12) holds, then all coefficients $f_{\mathbf{i}} = 0$, hence the monomials $\mathbf{x}^{\mathbf{i}}$ on $\mathbb{B}^n$ are linearly independent. ■

The set $\mathrm{Pol}_{d|X}^n := \{f_{|X}\colon f \in \mathrm{Pol}_d\}$ of all polynomials restricted to X also has the natural structure of vector space. However, if $X \neq \mathbb{B}^n$, then in general we will have $\dim(\mathrm{Pol}_{d|X}^n) \leq \dim(\mathrm{Pol}_d^n)$. The last inequality is equivalent to the statement that there may exist different polynomials of degree $\leq d$ on $\mathbb{B}^n$ having identical restrictions to X (recall the example of Section 2.2).

When we need to refer to a unique polynomial representation, we shall refer to Pol_d^n rather than $\mathrm{Pol}_{d|X}^n$. Next we show that decision trees span Pol_d^n.

Lemma 6 (Decision Trees span Pol_d^n) *Assume that the domain $X \subseteq \mathbb{B}^n$ has the property (4). Then*

$$\mathrm{span}\left\{f_{|X}\colon f \in \mathrm{DT}_d(X)\right\} = \mathrm{Pol}_{d|X}. \tag{13}$$

Proof It is sufficient to show that every monomial $\mathbf{x}|_X \in$, $\mathbf{i} \in \mathbb{B}_d^n$, is a linear combination of decision trees from $\mathrm{DT}_d(X)$. We begin with a partition of unity into

$$1 = \sum_{\mathbf{a}+\mathbf{b}=\mathbf{i}} \mathbf{x}^{\mathbf{a}}\bar{\mathbf{x}}^{\mathbf{b}} \text{ where } \mathbf{a}, \mathbf{b} \in \mathbb{B}^n, \tag{14}$$

and $\bar{\mathbf{x}}^{\mathbf{b}} := (1-\mathbf{x})^{\mathbf{b}} = \mathbf{x}^{\mathbf{b}'}$ on X for certain $\mathbf{b}' \in \mathbb{B}_d^n$ by virtue of the standing assumption (4). Since for any $\mathbf{x}$ only one of the terms in the sum can be nonzero (for every dimension

either the corresponding x_i or $\bar{x}_i = 1 - x_i$ appears in the product) we see that (14) holds. Futhermore, (4) implies that each of the terms $\mathbf{x}^{\mathbf{a}}\bar{\mathbf{x}}^{\mathbf{b}}$ can be written as $\mathbf{x}^{\mathbf{c}}$ for some suitable $\mathbf{c} \in \{0,1\}^n$. From (14) it follows that

$$p_{\mathbf{i}}(\mathbf{x}) := \mathbf{x}^{\mathbf{i}} - \sum_{\mathbf{a}+\mathbf{b}=\mathbf{i}, \mathbf{b}\neq 0} \mathbf{x}^{\mathbf{a}}\bar{\mathbf{x}}^{\mathbf{b}} = 2\mathbf{x}^{\mathbf{i}} - 1 \tag{15}$$

and therefore $\mathbf{x}^{\mathbf{i}} = \frac{1}{2}(1+p_{\mathbf{i}}(\mathbf{x}))$. Finally note that for any $\mathbf{i} \in \mathbb{B}_d^n$, $p_{\mathbf{i}}(\mathbf{x}) = \mathbf{x}^{\mathbf{i}} - \sum_{\mathbf{a}+\mathbf{b}=\mathbf{i}, \mathbf{b}\neq 0} \mathbf{x}^{\mathbf{a}}\mathbf{x}^{\mathbf{b}'}$ is a decision tree of depth d satisfying all conditions of Definition 4. Since all the monomials $\{\mathbf{x}^{\mathbf{i}} \colon \mathbf{i} \in \mathbb{B}_d^n\}$ are linearly independent, we can conclude that (13) holds. ■

A consequence of this result is that a minimization of a risk functional over the space of boosted decision trees of depth d on X is equivalent to the minimization over the space of all polynomials of degree d on X. This is because the hypotheses obtained by boosting decision trees will be of the form $\operatorname{sgn}\left(\sum_{\mathbf{i}} c_{\mathbf{i}} f_{\mathbf{i}}(\mathbf{x})\right)$. This suggests that in the limit, boosting by using decision trees cannot lead to solutions other than those obtained in Pol_d by means of a linear classifier. See Section 5 below for a more detailed discussion of this matter.

3. Boolean Kernels

In this section we will introduce Boolean kernels in an axiomatic way, which means that we will posit the existence of such kernels and describe some useful properties following from these definitions. The explicit construction will be achieved using combinatorial reasoning.

3.1 Basic Properties

We begin by introducing a norm on Pol_d^n.

Definition 7 (Norm on Pol_d^n) *For $f \in \mathrm{Pol}_d^n$ and "weights" $K := (K_0, K_1, \ldots, K_d)$, $K_i > 0$ for $i \in [0:d]$ define*

$$\|f\|_K := \left(\sum_{\mathbf{i}\in\mathbf{I}} K_{|\mathbf{i}|} f_{\mathbf{i}}^2\right)^{\frac{1}{2}} \quad \text{where } f = \sum_{\mathbf{i}\in\mathbf{I}} f_{\mathbf{i}} x^{\mathbf{i}}, \qquad \mathbf{I} \subseteq \mathbb{B}_d^n \tag{16}$$

It is easy to see that $\|f\|_K$ is indeed a norm, since the representation of f as a linear combination of monomials is unique. We shall shortly see how (16) can become a Reproducing Kernel Hilbert Space norm shortly.

Remark 8 (Induced Hilbert Space) *By exploiting the polarization equation, which connects quadratic and bilinear forms, the norm given by (16) induces a Hilbert space structure on Pol_d via the dot product*

$$\langle f, g\rangle_K = \sum_{\mathbf{i}\in\mathbb{B}_d^n} K_{|\mathbf{i}|} f_{\mathbf{i}} g_{\mathbf{i}} \text{ where } f = \sum_{\mathbf{i}\in\mathbb{B}_d^n} f_{\mathbf{i}}\mathbf{x}^{\mathbf{i}} \text{ and } g = \sum_{\mathbf{i}\in\mathbb{B}_d^n} g_{\mathbf{i}}\mathbf{x}^{\mathbf{i}}. \tag{17}$$

Next we apply $\|\cdot\|_K^2$ to various polynomials we encountered earlier. For disjunctive normal forms (2) and decision trees (8) we have

$$\|f\|_K^2 = K_0 + 4\sum_{\mathbf{i}\in I-\{0\}} K_{|\mathbf{i}|}, \quad \text{where } f(\mathbf{x}) = 2\sum_{\mathbf{i}\in I}\mathbf{x}^{\mathbf{i}} - 1 \in \mathrm{DNF}_d^n, \tag{18}$$

$$\|f\|_K^2 = \sum_{\mathbf{i}\in I} K_{|\mathbf{i}|}, \qquad \text{where } f(\mathbf{x}) = \sum_{\mathbf{i}\in I} f_{\mathbf{i}}\mathbf{x}^{\mathbf{i}} \in \mathrm{DT}_d(X),\ f_{\mathbf{i}} = \pm 1. \tag{19}$$

In the latter case this means that $\|f\|_K^2$ counts the leaves of the decision tree, weighted by their corresponding depths.

3.2 The Kernels Associated with the Hilbert Space

Now we are able to introduce the kernel corresponding to (16) and provide efficient means of computing it. Consider the following feature map from $\mathbb{B}^n$ to the Hilbert space $H = (\mathrm{Pol}_d^n, \langle\ \rangle_K)$:

$$\Phi\colon \mathbf{x} \mapsto \sum_{\mathbf{i}\in\mathbb{B}_d^n} K_{|\mathbf{i}|}^{-1}\mathbf{x}^{\mathbf{i}}\mathbf{e}_{\mathbf{i}} \tag{20}$$

where $(\mathbf{e}_{\mathbf{i}})_{\mathbf{i}\in\mathbb{B}_d^n}$ are the orthonormal canonical basis vectors in Pol_d^n corresponding to the monomials $\mathbf{x}_{\mathbf{i}}$, $\mathbf{i}\in\mathbb{B}_d^n$. Given a polynomial $f(\mathbf{x}) = \sum_{\mathbf{i}\in\mathbf{I}} f_{\mathbf{i}}\mathbf{x}^{\mathbf{i}} \in \mathrm{Pol}_d^n$, denote the weight vector $\mathbf{w} = \sum_{\mathbf{i}\in\mathbb{B}_d^n} w_{\mathbf{i}}\mathbf{e}_{\mathbf{i}}$, where $w_{\mathbf{i}} = f_{\mathbf{i}}$ for $\mathbf{i}\in\mathbf{I}$ and $w_{\mathbf{i}} = 0$, otherwise. f and $\mathbf{w}$ then satisfy the conditions

$$f(\mathbf{x}) = \langle\Phi(\mathbf{x}), \mathbf{w}\rangle_K \text{ for all } \mathbf{x}\in\mathbb{B}^n, \tag{21}$$

$$||f||_K = ||\mathbf{w}||_K. \tag{22}$$

Now observe that the kernel

$$k(\mathbf{x},\mathbf{x}') := \langle\Phi(\mathbf{x}),\Phi(\mathbf{x}')\rangle_K = \sum_{\mathbf{i}\in\mathbb{B}_d^n} K_{|\mathbf{i}|}^{-1}\mathbf{x}^{\mathbf{i}}\mathbf{x}'^{\mathbf{i}} \tag{23}$$

satisfies the reproducing property with respect to the dot product $\langle\ \rangle_K$; that is:

$$\langle f(\cdot), k(\mathbf{x}',\cdot)\rangle_K = f(\mathbf{x}') \tag{24}$$

for every $f(\mathbf{x}) = \sum_{\mathbf{i}\in\mathbf{I}} f_{\mathbf{i}}\mathbf{x}^{\mathbf{i}} \in \mathrm{Pol}_d^n \in \mathrm{Pol}_d$ and every $\mathbf{x}'\in\mathbb{B}^n$. This can be verified explicitly by writing out (24):

$$\langle f(\cdot), k(\mathbf{x}',\cdot)\rangle_K = \left\langle \sum_{\mathbf{i}\in\mathbf{I}} f_{\mathbf{i}}\mathbf{e}_{\mathbf{i}}, \sum_{\mathbf{i}\in\mathbb{B}_d^n} K_{|\mathbf{i}|}^{-1}\mathbf{x}'^{\mathbf{i}}\mathbf{e}_{\mathbf{i}} \right\rangle_K \tag{25}$$

$$= \sum_{\mathbf{i}\in\mathbf{I}} K_{|\mathbf{i}|}K_{|\mathbf{i}|}^{-1} f_{\mathbf{i}}\mathbf{x}'^{\mathbf{i}} = f(\mathbf{x}'). \tag{26}$$

We shall call the kernel (23) *the Boolean kernel of degree d and weight sequence K.*

3.3 Efficient Computation of Boolean Kernels

An obvious practical problem with (23) is that it involves summing $|\mathbb{B}_d^n| = O(n^d)$ terms. We will now present a computationally efficient way of computing (23). The key observation to an efficient formula for (23) is that for a given $\mathbf{i} = (i_1, ..., i_n) \in \mathbb{B}^n$, $|\mathbf{i}| = j$, the $\mathbf{i}$th term will contribute to the sum iff $x_\alpha x'_\alpha = 1$ for every $1 \leq \alpha \leq n$ such that $\mathbf{x}_\alpha = 1$. What this means is that if there are exactly p nonzero terms $x_\alpha x'_\alpha$, we can form only $\binom{p}{j}$ products of the form $\mathbf{x}^\mathbf{i}\mathbf{x}'^\mathbf{i} = x_{\alpha_1} x'_{\alpha_1} \cdots x_{\alpha_j} x'_{\alpha_j}$ which do not vanish, regardless of the dimensionality of $\mathbb{B}^n$. Since $\mathbf{x}, \mathbf{x}' \in \mathbb{B}_d^n$ are vectors of Boolean variables, we have that $p = \langle \mathbf{x}, \mathbf{x}' \rangle$. This leads to

$$\sum_{\mathbf{i}:\ |\mathbf{i}|=j} \mathbf{x}^\mathbf{i}\mathbf{x}'^\mathbf{i} = \binom{\langle \mathbf{x}, \mathbf{x}' \rangle}{j} \tag{27}$$

and thus

$$k(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{i} \in \mathbb{B}_d^n} K_{|\mathbf{i}|}^{-1} \mathbf{x}^\mathbf{i}\mathbf{x}'^\mathbf{i} = \sum_{j=0}^{d} K_j^{-1} \binom{\langle \mathbf{x}, \mathbf{x}' \rangle}{j}. \tag{28}$$

Note that for the special case where $K_j = \epsilon^{-j}$ with $\epsilon > 0$ and $d \geq \langle \mathbf{x}, \mathbf{x}' \rangle$ (27) leads to a binomial expansion and we obtain

$$k(\mathbf{x}, \mathbf{x}') = \sum_{j=0}^{d} \epsilon^j \binom{\langle \mathbf{x}, \mathbf{x}' \rangle}{j} = (1 + \epsilon)^{\langle \mathbf{x}, \mathbf{x}' \rangle}. \tag{29}$$

The smaller ϵ, by (16) the more severely will penalize higher order polynomials, which provides us with an effective means of controlling the complexity of the class of functions used. Note that this trick is applicable only to the case when $d \geq \max_{\mathbf{x} \in X} |\mathbf{x}|$ which always holds for $d = n$.

We will discuss the properties of kernels in more detail in Section 7, where we make the connection between regularization operators, corresponding expansions in terms of parity functions, and kernels on discrete objects.

For general $K_{|\mathbf{i}|}$ the kernel k cannot be computed in a closed form. However, for many practical cases we can resort to table lookup in order to find an equally efficient way ($O(n)$ time) to compute values of k:

Remark 9 (Table Lookup) *Boolean kernels on $\mathbb{B}^n$ can be computed in $O(n)$ time by table lookup. Since $k(\mathbf{x}, \mathbf{x}')$ is a function of $\langle \mathbf{x}, \mathbf{x}' \rangle$, and the latter may only take on values in $\{0, \ldots, n'\}$, where $n' := \max_{\mathbf{x} \in X} |\mathbf{x}| \leq n$, we can pre-compute all outcomes of $k(\langle \mathbf{x}, \mathbf{x}' \rangle)$ by storing $n'+1$ values and performing a table-lookup to obtain $k(\mathbf{x}, \mathbf{x}')$ by counting the number of variables which are true for both $\mathbf{x}$ and $\mathbf{x}'$. Counting this number costs $O(n)$ operations. This holds regardless of the order of d.*

4. Complexity Bounds for Propositional Logic

In this section we will establish lower bounds on the complexity of decision trees and disjunctive normal forms required in order to learn a dataset with a certain accuracy. We will also derive tools for constructing algorithms to compute such bounds.

4.1 The Regularized Risk Functional

We begin with a criterion to trade off complexity of the solution with the number of errors committed on the training set. For convenience we choose the regularized risk, i.e., goodness of fit plus complexity penalty, as commonly used with kernel methods (Schölkopf and Smola, 2002, Vapnik, 1998).

Given a labelled training sequence $(\mathbf{x}_i, y_i) \in X \times \{\pm 1\}$ for $i \in [1:m]$, a regularization constant $\lambda > 0$ and a weight sequence $K = (K_0, K_1, \ldots, K_d)$ of numbers > 0 the *regularised risk* is

$$R_{\mathrm{reg}}[f, \lambda] \quad := \quad \lambda \|f\|_K^2 + \sum_{i=1}^{m} [1 - y_i f(\mathbf{x}_i)]_+^2, \tag{30}$$

where $[\xi]_+ := \max(0, \xi)$ for $\xi \in \mathbb{R}$. This risk functional is common to support vector machines with quadratic penalty (Cristianini and Shawe-Taylor, 2000). Often we will need a second quantity, called the empirical error. It is given by

$$R_{\mathrm{emp}}[f] := |\{\mathrm{sgn}(f(x_i)) \neq y_i \colon i \in [1:m]\}|; \tag{31}$$

that is, by the number of disagreements between $\mathrm{sgn}(f(x))$ and the corresponding label y.

Our strategy for computing lower bounds on the complexity of logical formulae is to compute the value of $R_{\mathrm{reg}}[f, \lambda]$ for $\mathrm{DNF}_d(X)$ and $\mathrm{DT}_d(X)$ and compare them with the lowest value obtainable by minimizing $R_{\mathrm{reg}}[f, \lambda]$ over the whole space Pol_d^n or a linear subspace thereof.

4.2 Lower Bounds

We begin by computing the value of $R_{\mathrm{reg}}[f, \lambda]$ for decision trees. If

$$f = \mathbf{x} \mapsto \sum_{\mathbf{i} \in \mathbf{I}} f_{\mathbf{i}} \mathbf{x}^{\mathbf{i}} \in \mathrm{DT}_d(X),$$

then all nonzero coefficients $f_{\mathbf{i}}$ have magnitude 1 and $|f(\mathbf{x})| = 1$ for all $\mathbf{x} \in X$. Consequently the data dependent terms $1 - y_i f(\mathbf{x}_i)$ in (30) are either 0 or 2. This means that

$$\sum_{i=1}^{m} [1 - y_i f(\mathbf{x}_i)]_+^2 = 4 R_{\mathrm{emp}}[f], \tag{32}$$

and by using (19) we obtain

$$R_{\mathrm{reg}}[f, \lambda] = \lambda \sum_{\mathbf{i} \in \mathbf{I}} K_{|\mathbf{i}|} + 4 R_{\mathrm{emp}}[f]. \tag{33}$$

In other words, the risks are fully determined by the number and depths of the leaves of the decision tree and the number of errors. Finally, if the decision tree is consistent with the labels, i.e., if $R_{\mathrm{emp}}[f] = 0$, we have $\lambda \|f\|_K^2 = R_{\mathrm{reg}}[f, \lambda] = \lambda \sum_{\mathbf{i} \in \mathbf{I}} K_{|\mathbf{i}|}$. Likewise, in the case of DNFs consistent with the data:

$$\lambda \|f\|_K^2 = R_{\mathrm{reg}}[f, \lambda] = \lambda \left[K_0 + 4 \sum_{\mathbf{i} \in I \setminus \{0\}} K_{|\mathbf{i}|} \right]$$

We have proved the following theorem:

Theorem 10 (Lower Bounds for Decision Trees and DNFs) *Given an arbitrary data sequence $((\mathbf{x}_i, y_i))_{i=1}^m$ Denote by R^* the minimum of the regularized risk functional with respect to $f \in \mathrm{Pol}_d^n$, as defined in (30). Then*

$$R_{\mathrm{emp}}[f] \geq \frac{1}{4}R^* - \frac{\lambda}{4}\sum_{\mathbf{i}\in\mathbf{I}} K_{|\mathbf{i}|}$$

for all $f \in \mathrm{DT}_d(X)$ and

$$\|f\|_K^2 = \sum_{\mathbf{i}\in\mathbf{I}} K_{|\mathbf{i}|} \geq \frac{1}{\lambda}R^*$$

for every consitent decision tree $f \in \mathrm{DT}_d(X)$. Similarily,

$$\|f\|_K^2 = K_0 + 4\sum_{\mathbf{i}\in I\setminus\{0\}} K_{|\mathbf{i}|} \geq \frac{1}{\lambda}R^*$$

for every consitent $f \in \mathrm{DNF}_d(X)$.

The theorem holds for any choice of λ and K and one could optimise over choices of these.

The practical use of Theorem 10 is that we can now use several algorithms to find a (possibly tight) lower bound on R^*, e.g. Kowalczyk et al. (2002). This will allow us to give lower bounds on the error of propositional logic in polynomial time.

If we are willing to afford $O(m^3)$ time, we can obtain the value of R^* straightforwardly by solving a quadratic program arising from the minimization of $R_{\mathrm{reg}}^+[f, \lambda]$. In many cases, however, we may be content with a slightly worse lower bound while reaping the benefits of a much faster algorithm.

4.3 Extended Hilbert Space

The approximate algorithms that we will use to determine lower bounds on the complexity of admissible hypotheses rely on data which can be classified without error, i.e., $R_{\mathrm{emp}}[f] = 0$. Since this is unlikely to be the case for arbitrary data (nor is it desirable to achieve such a goal in all cases), we need to reformulate the learning problem such that it behaves *as if it was solvable* with zero training error. The idea is to shift the training task to an extended Hilbert Space. The construction to be introduced now is similar to the methods in (Cristianini and Shawe-Taylor, 2000, Freund and Schapire, 1998), with the slight twist that we will formulate everything in terms of kernels and norms rather than feature maps.

Accordingly we begin with the introduction of an extension of the Hilbert space H into $\tilde{H}$, which is defined on pairs of observations $\mathbf{x}$ and indices i. Given $\lambda > 0$, we define $\tilde{H}$ via its corresponding kernel

$$\tilde{k}((\mathbf{x}, i), (\mathbf{x}', j)) := k(\mathbf{x}, \mathbf{x}') + \lambda\delta_{ij}. \tag{34}$$

$\tilde{k}$ is clearly a Mercer kernel since whenever $[k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^m$ is positive definite $[k(\mathbf{x}_i, \mathbf{x}_j) + \lambda\delta_{ij}]_{i,j=1}^m$ is too. By the reproducing property we know that $\tilde{f} \in \tilde{H}$ has the form $\tilde{f}((\mathbf{x}, i)) = f(\mathbf{x}) + \tilde{f}_i = f(\mathbf{x}) + \sum_j \tilde{f}_j\delta_{j,i}$, where $f \in H$ and $\tilde{f}_i \in \mathbb{R}$, and we have

$$\langle \tilde{f}, \tilde{f}' \rangle_{\tilde{H}} = \langle f, f' \rangle_H + \frac{1}{\lambda}\sum_{j=1}^m \tilde{f}_j\tilde{f}'_j$$

for every $\tilde{f} = f + \sum_j \tilde{f}_j \delta_{j,\cdot}$ and $\tilde{f}' = f' + \sum_j \tilde{f}'_j \delta_{j,\cdot}$ from $\tilde{H}$. Hence

$$\langle \tilde{f}, \tilde{k}((\mathbf{x}, i), \cdot) \rangle_{\tilde{H}} = \langle f, k(\mathbf{x}, \cdot) \rangle_H + \frac{1}{\lambda} \sum_{j=1}^{m} \tilde{f}_j \lambda \delta_{i,j} = f(\mathbf{x}) + \tilde{f}_i. \tag{35}$$

This yields an explicit form of the RKHS norm of $\tilde{H}$

$$\|\tilde{f}\|^2 = \|f\|_H^2 + \frac{1}{\lambda} \sum_i \tilde{f}_i^2.$$

which allows us to state the following theorem:

Theorem 11 (Extended Hilbert Space) *Given K, let $f^* = \arg\min_f R_{\mathrm{reg}}[f, \lambda]$. Furthermore denote by $\tilde{f}^*$ the minimizer of the following problem:*

$$\textit{minimize } \|\tilde{f}\|^2 \textit{ subject to } 1 - y_i \tilde{f}(\mathbf{x}_i, i) \leq 0 \textit{ for all } i \in [1 : m] \tag{36}$$

Then the following equality holds:

$$R_{\mathrm{reg}}[f^*, \lambda] = \lambda \|\tilde{f}^*\|^2 = \frac{\lambda}{\rho^2},$$

where ρ is the margin achievable by the corresponding standard hard-margin classification problem.

Proof By construction we have $1 - y_i \tilde{f}(\mathbf{x}_i, i) = 1 - y_i f(\mathbf{x}_i) - \tilde{f}_i y_i$. Therefore we may rewrite (36) as follows:

$$\text{minimize } \|f\|_K^2 + \lambda^{-1} \sum_i \tilde{f}_i^2 \text{ subject to } 1 - y_i f(x_i) - \tilde{f}_i y_i \leq 0 \text{ for all } i \in [1 : m]. \tag{37}$$

Here the minimum is taken with respect to $f \in H$ and $\{\tilde{f}_1, \ldots, \tilde{f}_m\} \in \mathbb{R}^m$ or equivalently, $\tilde{f} \in \tilde{H}$. Multiplying the objective function by λ, and noticing that the optimal choice of $\tilde{f}_i$ is $y_i \min(0, 1 - y_i f(x_i))$ we can rewrite (37) as

$$\text{minimize } \lambda \|f\|_K^2 + \sum_i \tilde{f}_i^2 \text{ where } y_i \tilde{f}_i = \min(0, 1 - y_i f(x_i)) \text{ for all } i \in [1 : m]. \tag{38}$$

Finally, substituting $\min(0, 1 - y_i f(x_i))$ into the objective function and dropping the constraints shows that the optimization problem is identical to the minimisation of the regularized risk functional (30). The last equality follows from a result in (Boser et al., 1992) stating that $\|\mathbf{w}\|^2 = \rho^{-2}$, where in our case $\mathbf{w}$ corresponds to $\tilde{f}^*$. ■

Algorithm 1 Regularized Kernel Perceptron (k, λ-perceptron)

Given : data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$, a Mercer kernel k and $\lambda \geq 0$.
Initialise : $t = 0$ and $\alpha_i = 0$ for $i = 1, \ldots, m$.

while There exist some i with $y_i \left(\sum_j \alpha_j y_j k(\mathbf{x}_j, \mathbf{x}_i) + \lambda \alpha_i) \right) < 0$
update $\alpha_i \leftarrow \alpha_i + 1$ and $t \leftarrow t + 1$.
end while

4.4 Kernel Perceptron Test

Now we are in a position to give an example of an explicit algorithm for determining a lower bound on the complexity of decision trees and DNF using the perceptron algorithm.

Proposition 12 *Assume that there exists some R with $R^2 \geq k(\mathbf{x}_i, \mathbf{x}_i)$ for all $i \in [1 : m]$ and let $\lambda > 0$. If Algorithm 1 converges after t steps, the regularized risk functional $R_{\mathrm{reg}}[f, \lambda]$ is bounded from below by $R_{\mathrm{reg}}[f, \lambda] \geq \frac{\lambda t}{R^2 + \lambda}$.*

Proof From Novikoff (1962) we know that a perceptron with achievable margin ρ, whose data is bounded in a ball of radius r will converge after at most $t \leq \frac{r^2}{\rho^2}$ steps. In other words, we can bound $\frac{t}{r^2} \leq \rho^{-2}$.

Recall that the regularized kernel perceptron (Algorithm 1) is nothing but the ordinary perceptron in feature space, where we used $\tilde{k}$ as a kernel instead of k. The radius r for the data is then given by $r^2 = \max_i \tilde{k}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_i) = \max_i k(\mathbf{x}_i, \mathbf{x}_i) + \lambda \leq R^2 + \lambda$. To conclude the proof, we apply Theorem 11 to obtain the inequality $R_{\mathrm{reg}}[f^*, \lambda] = \frac{\lambda}{\rho^2} \geq \frac{\lambda t}{R^2 + \lambda}$. ■

This holds, in particular, for $f \in \mathrm{DT}_d$ or $f \in \mathrm{DNF}_d$. Therefore, application of Theorem 10 allows us to bound the classification error and complexity in terms of the runtime of the perceptron algorithm.

5. Kernels and Decision Trees

Decision trees are a widely used method for classification and regression (Breiman et al., 1984, Quinlan, 1993) and have experienced an increase in performance by taking convex combinations of them (Freund and Schapire, 1996). What we will show below is that in all popular cases one can construct kernels which are able to perform all possible classifications that also decition trees are able to find. In particular, such kernels will find the *optimal*, i.e., maximum margin solution among the class of convex combinations of such decision trees.

More specifically, we will deal with three strategies commonly used in decision trees:

1. splits parallel to one of the axes, i.e., threshold functions of the type $\mathbf{x} \mapsto \chi_{(-\infty, c]}(x_i)$ and $\mathbf{x} \mapsto \chi_{[c, -\infty)}(x_i)$ for $c \in \mathbb{R}$ and $i \in \{1, \ldots, n\}$ (where χ denotes the indicator function),

2. general linear separators of the type $\frac{1}{2}(\mathrm{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b) + 1)$, and

3. power sets, i.e., $\chi_S(x_i)$ where $S \subseteq X_i$ and $x_i \in X_i$.

The advantage that kernels offer is that optimization over such sets, which is typically a combinatorial problem, is transformed into an easily tractable problem by taking the convex combination of all solutions (something that is bound to occur in boosting anyway) and finding the optimal solution in such a space. We use a technique similar to the one which led to spline kernels with "infinite numbers of knots" (Smola, 1996).

5.1 Pruning of Decision Trees

We begin by setting the weighting $K = (1, 1, \ldots, 1)$ in the definition of the RKHS norm (16). This means that the RKHS norm of a decision tree will simply be a multiple of the number of leaves of the tree. Consequently the regularized risk $R_{\mathrm{reg}}[f, \lambda]$ reduces to the "cost complexity" used by Breiman et al. (1984) in the pruning of decision trees via the CART algorithm.

The pruning proceeds as follows: first the maximal decision tree $\hat{f}$ is generated, achieving $R_{\mathrm{emp}}[f] = 0$. Then for each $\lambda \geq 0$, a (unique) subtree f_λ minimising $R_{\mathrm{reg}}[f, \lambda]$ is found. It turns out that there exists a finite sequence of numbers $0 = \lambda_1 < \lambda_2 < \cdots < \lambda_s < \lambda_{s+1} = \infty$ such that

- $f_0 = \hat{f}$ and $f_\infty \equiv 1$.
- For all $\lambda \in (\lambda_i, \lambda_{i+1})$ with $i \in [1 : s]$ we have $f_\lambda = f_{\lambda_i}$.
- $f_{\lambda_{i+1}}$ is a proper subtree of f_{λ_i} for all $i \in [1 : s]$.

To find a suitable value of λ a validation set is used. In particular, one seeks the largest λ for which the misclassification error is smaller than the sum of the *minimal validation error* plus the *standard error*. More formally these quantities are defined as follows:

$$R^*_{\mathrm{val}} := \min_\lambda \frac{1}{m_{\mathrm{val}}} |\{i \in [1 : m_{\mathrm{val}}] \colon y_i \neq f_\lambda(x_i)\}| \tag{39}$$

$$R_{\mathrm{StdErr}} := \sqrt{R^*_{\mathrm{val}}(1 - R^*_{\mathrm{val}})/m_{\mathrm{val}}} \tag{40}$$

$$\lambda_{\mathrm{opt}} := \max\{\lambda \colon R_{\mathrm{val}}[f_\lambda] \leq R^*_{\mathrm{val}} + R_{\mathrm{StdErr}}\}. \tag{41}$$

Here m_{val} is the number of instances on the validation set, $R_{\mathrm{val}}[f]$ is the validation error of f, and λ_{opt} is the optimal parameter according to the validation procedure.

In other words, one chooses a value of λ which is close to being optimal on the validation set, while taking the inherent variation on the validation set via R_{StdErr} into account. In this sense the CART algorithm implements minimisation of the regularised risk in the class of subtrees of the maximal tree, with the regularisation constant λ selected by a heuristic applied to a validation set.

5.2 Axis Parallel Splits

We begin with splits along a coordinate i of the data. That is, we will study the class of all functions $\psi_c(x_i)$ and $\bar{\psi}_c(x_i) = 1 - \psi_c(x_i)$ with

$$\psi_c(x_i) = \chi_{(-\infty, c]}(x_i) \text{ and } \bar{\psi}_c(x_i) = \chi_{[c, \infty)}(x_i). \tag{42}$$

Assuming that we use a set of pre-defined splits $\{c_j\}_j$, the corresponding kernel $k(x_i, x_i')$ becomes

$$k(x_i, x_i') = \sum_j \psi_{c_j}(x_i)\psi_{c_j}(x_i') + \bar{\psi}_{c_j}(x_i)\bar{\psi}_{c_j}(x_i'). \tag{43}$$

We now have several options how to choose such c_j. One option is to use the empirical marginal distribution on x_i. This will provide us with the class of all possible splits that differ on the training data, since for all other splits the values of ψ_c will not change on the training set. We obtain

$$\begin{aligned} k(x_i, x_i') &= \sum_{j: c_j \leq \min(x_i, x_i')} 1 + \sum_{j: c_j \geq \max(x_i, x_i')} 1 && (44) \\ &= m(F_i(\min(x_i, x_i')) + (1 - F_i(\max(x_i, x_i')))) && (45) \\ &= m(1 - |F_i(x_i) - F_i(x_i')|). && (46) \end{aligned}$$

Here F_i denotes the empirical cumulative distribution function of the marginal distribution of the coordinate x_i. Clearly we may drop the normalization constant m without further problems. Secondly, for any other (arbitrary) choice of c_i we will obtain the cumulative distribution function of c_i. Finally, in the limit of an infinite number of c_i, we may use any monotonic function G_i mapping x_i into $[0, 1]$. In summary, the kernel

$$k(x_i, x_i') = 1 - |G_i(x_i) - G_i(x_i')| \tag{47}$$

will lead to a classifier taken from the convex combination of splits along the coordinate x_i and the derivative $g(x) = G'(x)$ determines the weight given to each of the splits.

Remark 13 (Probability Integral Transform) *Via the transformation $x \to F(x)$ any kernel $k(F(x), F(x'))$ is* scale invariant. *Note that this transformation is useful also for kernels other than those introduced in (45). This shows that kernel methods can be made invariant under rescaling just like trees.*

5.3 Halfspaces

We proceed to the class of functions defined by halfspaces on $\mathbb{R}^n$, i.e. functions given by

$$\psi_{\mathbf{w},b}(\mathbf{x}) := \frac{1}{2}(1 + \operatorname{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)) \text{ and } \bar{\psi}_{\mathbf{w},b}(\mathbf{x}) := \frac{1}{2}(1 - \operatorname{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)). \tag{48}$$

One might be tempted to choose a finite set of $(\mathbf{w}_i, b_i)$ and then compute k in analogy to (43). This approach, however, is fraught with problems since the number of possible splits of the training data into half spaces can be exponential (Minsky and Papert, 1969) in the sample size and we are unaware of an efficient means of enumerating these half spaces, not even symbolically.[1] We can, however, resort to integration over the domain of $\mathbf{w}, b$ and obtain a kernel via

$$k(\mathbf{x}, \mathbf{x}') := \int \left(\psi_{\mathbf{w},b}(\mathbf{x})\psi_{\mathbf{w},b}(\mathbf{x}') + \bar{\psi}_{\mathbf{w},b}(\mathbf{x})\bar{\psi}_{\mathbf{w},b}(\mathbf{x}')\right) d\mu(\mathbf{w}, b) \tag{49}$$

1. This does not imply that such a kernel might not exist and there are problems where such an enumeration approach is successful, such as the one described in Section 6.4.

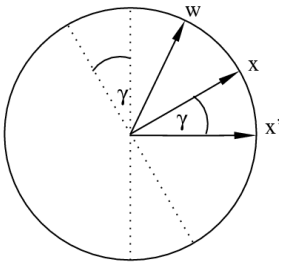


Figure 1: The dot products $\langle \tilde{\mathbf{w}}, \tilde{\mathbf{x}} \rangle$ and $\langle \tilde{\mathbf{w}}, \tilde{\mathbf{x}}' \rangle$ only disagree in the area between the dotted lines. Hence the integral over the arc where $\psi_{\mathbf{w}}(\mathbf{x})$ and $\psi_{\mathbf{w}}(\mathbf{x}')$ agree is given by $\pi - \gamma$.

where the measure $\mu(\mathbf{w}, b)$ is appropriately chosen. In other words, μ determines how much weight or importance we may assign to individual splits. This is perfectly in order, since also in decision trees one has to find suitable priorities when searching the space of possible halfspaces (Breiman et al., 1984).

First note that we have a scaling freedom in the choice of $\mathbf{w}, b$ to the extent that $\psi_{\alpha\mathbf{w},\alpha b} = \psi_{\mathbf{w},b}$ for $\alpha > 0$. Furthermore we can merge $\mathbf{w}$ and b into $\tilde{\mathbf{w}} := (\mathbf{w}, b)$ by defining $\tilde{\mathbf{x}} := (\mathbf{x}, 1)$ and consider

$$\psi_{\tilde{\mathbf{w}}}(\tilde{\mathbf{x}}) := \frac{1}{2}(1 + \operatorname{sgn}\langle \mathbf{w}, \mathbf{x} \rangle) \text{ and } \bar{\psi}_{\tilde{\mathbf{w}}}(\tilde{\mathbf{x}}) := \frac{1}{2}(1 - \operatorname{sgn}\langle \mathbf{w}, \mathbf{x} \rangle) \tag{50}$$

with $\|\tilde{\mathbf{w}}\| = 1$ instead. Since no direction should be considered to be preferred, we choose the unit sphere $S_n \subset \mathbb{R}^{n+1}$ with $(\mathbf{x}, 1) \in \mathbb{R}^{n+1}$ as the domain of integration. After rearranging terms we obtain

$$k(\mathbf{x}, \mathbf{x}') := \int_{S_n} \chi_{\{\langle \tilde{\mathbf{w}}, \tilde{\mathbf{x}} \rangle \langle \tilde{\mathbf{w}}, \tilde{\mathbf{x}}' \rangle > 0\}} d\mu(\tilde{\mathbf{w}}). \tag{51}$$

Next note that the integral depends only on $\mathbf{x}, \mathbf{x}'$, or more precisely, on the two dimensional plane passing through them and the origin. Therefore, we may simply study the case of S_1 and ignore the remaining dimensions, which would only result in a multiplicative constant. Geometrical considerations (see Figure 1) lead to

$$k(\mathbf{x}, \mathbf{x}') \propto \pi - \gamma = \pi - \arccos \frac{\langle \tilde{\mathbf{x}}, \tilde{\mathbf{x}}' \rangle}{\|\tilde{\mathbf{x}}\| \|\tilde{\mathbf{x}}'\|} = \pi - \arccos \frac{\langle \mathbf{x}, \mathbf{x}' \rangle + 1}{\sqrt{(\|\mathbf{x}\|^2 + 1)(\|\mathbf{x}'\|^2 + 1)}}. \tag{52}$$

A standard argument yields that in analogy to polynomial kernels, taking powers of k means that we are considering concatenations of splits over various halfspaces.

Part II
Hilbert Spaces on Structured Data

We will now consider more general discrete input spaces X and develop techniques for controlling the complexity of hypotheses in terms of the degree on interaction between different dimensions. In doing so we will extend some existing results on ANOVA decompositions.

6. Hilbert Spaces, Frames, and ANOVA Decompositions

In this section we will exploit ANOVA decompositions to provide an orthogonal decomposition of the space of functions on discrete variables. This will allow us to extend the methods

from regularization of functions of continuous variables to those on discrete variables in a natural way.

We drop the assumption (4) that for each i there exists a j with $x_i = (1 - x_j)$. Furthermore we extend the notion of binary observations to the one of observations on *discrete* sets of variables. We formalize this as follows:

$$X = \bigotimes_{i=1}^{n} X_i \text{ where } X_i = [1 : p_i]. \tag{53}$$

To avoid pathological cases we assume that $p_i > 1$. In other words, X is the cartesian product of finite sets X_i of cardinality p_i. The choice of integers was made purely for the sake of convenience (since we will not exploit the ordering on X_i). Clearly we could modify our definition to include sets such as $\{\text{German}, \text{Polish}, \text{Australian}\}$ or $\{\text{True}, \text{False}\}$.

6.1 A Hilbert Space on X

We begin by defining a Hilbert Space H on X, given by

$$H = L_2(X) = \bigotimes_{i=1}^{n} H_i \text{ where } H_i := L_2(X_i). \tag{54}$$

It is well known that the Kronecker functions $\delta_{\mathbf{x}'}(\mathbf{x})$ form a basis of H.

Our aim is to construct a decomposition of H into the direct sum of $A_0, \dots A_n$, i.e. $H = \bigoplus_i A_i$, where A_i are Hilbert Spaces containing functions that depend on exactly i variables. This is a useful strategy, since we can expect that higher order interactions will correspond to more complex (and thus less desirable) hypotheses in an estimator, which, in turn, is something one would like to control.

6.2 Frames and Kernels

While it is often rather difficult to find bases for subspaces (and the subspaces A_i will be no exception to this), it is much easier to provide tight frames which have been obtained by projection. Frames are defined as follows:

Definition 14 (Frame) *Denote by H a Hilbert space and by $\Phi \subset H$ a countable set. Then Φ is a frame if there exist numbers $0 < c \leq C < \infty$ such that*

$$c^2\|x\|^2 \leq \sum_{\phi \in \Phi} \langle \phi, x \rangle^2 \leq C^2\|x\|^2 \text{ for all } x \in H. \tag{55}$$

We call a frame tight *if $c = C$.*

The benefit of using frames is that given a tight frame on a subspace A_i of H we can form a kernel on A_i as follows.

Lemma 15 (Kernels from Tight Frames) *Denote by Φ a tight frame of a Hilbert space H with frame constant c. Then the reproducing kernel of H can be written as follows:*

$$k(x, x') := c^{-2} \sum_i \phi_i(x)\phi_i(x'). \tag{56}$$

Proof We begin by rewriting the dot product on H in terms of Φ. Since Φ is tight we have $\|f\|^2 = c^{-2}\sum_{\phi\in\Phi}\langle\phi, f\rangle^2$. The polarization equation allows us to recover the dot product as

$$\langle f, g\rangle = c^{-2}\sum_{\phi\in\Phi}\langle f, \phi\rangle\langle\phi, g\rangle \text{ for all } f, g \in H. \tag{57}$$

For k to be a reproducing kernel it has to satisfy $\langle f, k(x,\cdot)\rangle = f(x)$ for all $f \in H$, or equivalently $\langle\langle f, k\rangle, g\rangle = \langle f, g\rangle$. The latter can be seen by

$$\left\langle\left\langle f, c^{-2}\sum_{\phi\in\Phi}\phi\phi\right\rangle, g\right\rangle = c^{-2}\sum_{\phi\in\Phi}\langle f, \phi\rangle\langle g, \phi\rangle = \langle f, g\rangle. \tag{58}$$

■

6.3 Frames and ANOVA Decompositions

The next step in our analysis is to construct the frames corresponding to A_i, i.e., the spaces of bounded interaction, from decompositions of the Hilbert spaces H_i on the coefficients into constant and variable parts. We will decompose H_i into

$$\phi_{i0}(x) = p_i^{-\frac{1}{2}} \text{ and } \phi_{ij}(x) = \delta_{j,x} - p_i^{-1} \text{ for } j \in [1 : p_i]. \tag{59}$$

One may check that $\|\phi_{i0}\|^2 = 1$, $\|\phi_{ij}\|^2 = (p_i - 1)/p_i$, and $\langle\phi_{i0}, \phi_{ij}\rangle = 0$ for all $j \in [1 : p_i]$. Finally, $\langle\phi_{ij}, \phi_{il}\rangle = -1/p_i$ for $j \neq l$ and $i, j > 0$. Clearly the functions ϕ_{ij} (with $i, j > 0$) are linearly dependent and span a $p_i - 1$ dimensional space orthogonal to the constant function p_{i0}. To check that the sets

$$\Phi_i := \{\phi_{ij} \colon j \in [1 : p_i]\} \tag{60}$$

form frames we have to show that for any $f \in H_i$ with $\langle\phi_{i0}, f\rangle = 0$ eq. (55) holds. Observe

$$\sum_{\phi\in\Phi_i}\langle f, \phi\rangle^2 = \sum_{\phi\in\Phi_i}\langle f, \phi + p_i^{-\frac{1}{2}}\phi_{i0}\rangle^2 = \sum_{j\in[1:p_i]}\langle f, \delta_j\rangle^2 = \|f\|^2. \tag{61}$$

Using (56), the kernel k_i on span Φ_i evaluates to

$$k_i(x, x') = \left(\delta_{x,x'} - 1/p_i\right). \tag{62}$$

Next we need to prove that also cartesian products of Φ_i lead to tight frames, and, in fact span the spaces of bounded interactions A_j.

Lemma 16 (Tight Frames on Product Spaces) *Denote by H_i with $i \in [1 : n]$ Hilbert spaces with corresponding tight frames Φ_i and frame constant $c = 1$. Then the set*

$$\Phi := \Phi_1 \times \cdots \times \Phi_n \tag{63}$$

is a tight frame on $H := H_1 \otimes \cdots \otimes H_n$ with frame constant 1.

Proof We prove the statement for $n = 2$, all other cases follow by induction. Since H_i is a Hilbert space ($i = 1, 2$), there exists an orthonormal basis Ψ_i spanning H_i. Clearly, span $(\Psi_1 \times \Psi_2) = H$. We only need to show that (55) holds with $c = C = 1$ for all $\psi = \psi_1\psi_2 \in \Psi_1 \times \Psi_2$, since the rest follows from the fact that (55) contains only homogeneous quadratic forms. We have

$$\sum_{\phi_1 \in \Phi_1, \phi_2 \in \Phi_2} \langle \psi_1\psi_2, \phi_1\phi_2 \rangle^2 = \sum_{\phi_1 \in \Phi_1, \phi_2 \in \Phi_2} \langle \psi_1, \phi_1 \rangle \langle \psi_2, \phi_2 \rangle = \|\psi_1\|_{H_1}^2 \|\psi_2\|_{H_2}^2 = \|\psi\|_H^2. \quad (64)$$

■

Lemma 17 (Tight Frames on Direct Sums) *Denote by H_i Hilbert spaces and by $H = \bigoplus_i H_i$ the space formed by the direct sum of the H_i. Furthermore denote by Φ_i tight frames with frame constant 1 corresponding to each of the spaces H_i. Then the set $\Phi := \cup_i \Phi_i$ is a tight frame with frame constant 1 on $H_+ := H_1 \oplus \cdots \oplus H_n$.*

Proof Any $f \in H_+$ can be decomposed into $f = \sum_{i \in [1:n]} f_i$ with $f_i \in H_i$. Due to orthogonality $\sum_{\phi \in \Phi} \langle \phi, f \rangle^2 = \sum_{i \in [1:n]} \sum_{\phi_i \in \Phi_i} \langle \phi_i, f_i \rangle^2 = \sum_{i \in [1:n]} \|f_i\|^2 = \|f\|^2$. ■

Now we may state the key result which allows us to perform ANOVA decompositions with tight frames.

Theorem 18 (ANOVA Decompositions and Tight Frames) *Denote by $H_1, \ldots, H_n$ Hilbert spaces which are each of them decomposed into $H_i = H_i^c \oplus H_i^v$ where H_i^c is the space of all constant functions and H_i^v is the orthogonal complement thereof on H_i. Furthermore denote by Φ_i tight frames for H_i^v with frame constant $c = 1$. Then the set*

$$\Psi_d := \left\{ \psi \colon \psi = \|1\|_H^{-1} \pi^{\mathbf{i}} \phi^{\mathbf{i}} \text{ where } \mathbf{i} \in \mathbb{B}^n, |\mathbf{i}| = d \text{ and } \phi \in \Phi_1 \times \cdots \times \Phi_n \right\} \quad (65)$$

is a tight frame with frame constant 1. Here $\pi := (\pi_1, \ldots, \pi_n)$ with $\pi_i := \|1\|_{H_i}$ and $\phi^{\mathbf{i}} := \phi_1^{i_1} \cdot \cdots \cdot \phi_n^{i_n}$. Furthermore Ψ_d spans the Hilbert space $\underline{H}_d$ of all interactions of order d between functions on the spaces H_i.

Proof The Hilbert space of all interactions of order d is given by the product of d spaces H_i^v and $n - d$ spaces H_i^c. Since $H_i^v \perp H_i^c$, also all product spaces are mutually orthogonal. There exist exactly $\binom{n}{d}$ such spaces, indexed by $\mathbf{i}$ satisfying the conditions in (65). Ψ_d contains all of them, hence it spans $\underline{H}_d$. Without loss of generality we now focus on a fixed $\mathbf{i}$ and corresponding Hilbert space $H_{\mathbf{i}} := \bigotimes_{j \colon i_j = 1} H_j^v \bigotimes_{j \colon i_j = 0} H_j^c$ with the corresponding set $\Psi_{\mathbf{i}} := \left\{ \psi \,\middle|\, \psi = \pi^{1-\mathbf{i}} \phi^{\mathbf{i}} \right\}$ and apply Lemma 17 to obtain a tight frame on the sum of all $H_{\mathbf{i}}$ subsequently.

Recall that all Φ_i are tight frames on H_i^v with frame constant 1. In order to apply Lemma 16 we need to get rid of the normalization constants π_i induced by the padding of $H_{\text{short}} = \bigotimes_{j \colon i_j = 1} H_j^v$ by $n - d$ spaces H_i^c containing only the constant function.

For this purpose note that the dot products on $H_{\mathbf{i}}$ equal those on H_{short}, since the normalization terms π_i and the corresponding norms of the constant function on H_i^c cancel

out. Therefore we can apply Lemma 16 for H_{short} and then extend the conclusions to $\tilde{H}$ by noting that the map from H_{short} into $\tilde{H}$ is an isometry. ■

Now that we know how tight frames on product spaces can be built from products of frames on the underlying spaces, we proceed to a strategy to compute such sums efficiently.

6.4 Power Sets (Groupings)

Eq. (62) can also be derived directly by the notion of power sets on decision trees. This is instructive, since power sets (often also referred to as groupings) are a staple to the construction of splitting rules in decision trees. A commonly used strategy is to construct splits by the function $\chi_{I_i}(x_i)$ where $I_i \subseteq [1 : p_i]$.

Search over the set of all I_i is very expensive, since there are 2^{p_i} possibilities to find such a set, that is, if we include $\emptyset$ and $[1 : p_i]$ in our considerations.[2] While finding the optimal set is intractable, finding the optimal function, given as a convex combination of indicator functions on such sets, is a very tractable problem. It can be achieved by defining the following kernel:

$$k(x_i, x_i') := \sum_{I_i \subseteq \{1,\ldots,p_i\}} \chi_{I_i}(x_i)\chi_{I_i}(x_i') \tag{66}$$

To compute (66) note that for the product of the indicator functions to be 1 the condition is that $x_i, x_i' \in I_i$. If $x_i = x_i'$, there are exactly 2^{p_i-1} different possibilities, whereas for $x_i \neq x_i'$, we are limited to 2^{p_i-2} different choices. Therefore we obtain

$$k(x_i, x_i') = \left\{ \begin{array}{ll} 2^{p_i-1} & \text{if } x_i = x_i' \\ 2^{p_i-2} & \text{if } x_i \neq x_i' \end{array} \right\} = 2^{p_i-2}(\delta_{x_i,x_i'} + 1). \tag{67}$$

Up to scaling factors and a constant offset this kernel is identical to (62).

6.5 Efficient Computation of Kernels from Frames

In this section we extend the results from Section 3.3 to arbitrary kernels from frames.

Proposition 19 (Burges and Vapnik (1995)) *The ANOVA kernel k_d of order d on $\mathbb{R}^n$*

$$k_d(\mathbf{x}, \mathbf{x}') := \sum_{\mathbf{i} \in \mathbb{B}^n, |\mathbf{i}| = d} \mathbf{x}^{\mathbf{i}} \mathbf{x}'^{\mathbf{i}} \tag{68}$$

can be computed efficiently by the following recursion:

$$k_d(\mathbf{x}, \mathbf{x}') = \frac{1}{d} \sum_{s=1}^{d} (-1)^{s+1} k_{d-s}(\mathbf{x}, \mathbf{x}'), \bar{k}_s(\mathbf{x}, \mathbf{x}'). \tag{69}$$

where $k_0(\mathbf{x}, \mathbf{x}') = 1$ and $\bar{k}_s(\mathbf{x}, \mathbf{x}') := \sum_{i=1}^{n} (x_i x_i')^s$.

Below we extend this result to the case where rather than individual coordinates x_i we will have whole subspaces appearing in the ANOVA decomposition. While this extension is rather straightforward, it opens the door to efficient computational representations of frames and kernels on subspaces of bounded interaction.

2. This is done merely for notational convenience, since we are including only the constant functions 0 and 1. Otherwise we simply would obtain a small constant offset to k.

Proposition 20 (Kernel Recursion) *With the assumptions of Theorem 18 denote by k_d the kernel induced by the tight frame Φ_d. Then*

$$\underline{k}_d(\mathbf{x}, \mathbf{x}') := \sum_{\mathbf{i} \in \mathbb{B}^n, |\mathbf{i}| = d} \prod_{j \in [1:n]} (\pi_i^2 k_d(x_i, x_i'))^{i_j} \tag{70}$$

is the kernel given by the tight frame $\|1\|_H \Psi_d$ on $\underline{H}_d$. It can be computed by the recurrence

$$\underline{k}_d(\mathbf{x}, \mathbf{x}') = \frac{1}{d} \sum_{s=1}^{d} (-1)^{s+1} k_{d-s}(\mathbf{x}, \mathbf{x}') \bar{k}_s(\mathbf{x}, \mathbf{x}'), \tag{71}$$

where $k_0(\mathbf{x}, \mathbf{x}') = 1$ and $\bar{k}_s(\mathbf{x}, \mathbf{x}') := \sum_{i=1}^n \pi_i^s k_i(x_i, x_i')^s$.

The recursion (71) means that k_d can be computed at $O(d^2 + n)$ cost, even though the number of terms involved (see (70)) can be much larger.

Proof All we need to do is observe that x_i and x_i' only appear as products $x_i x_i'$ in Proposition 19. Therefore we may replace these terms by the kernels given on each of the H_i^v and we will see terms $\prod_{j=1}^n k_i^{i_j}(x_i, x_i')$ appearing in (68) instead of $\mathbf{x}^{\mathbf{i}} \mathbf{x}'^{\mathbf{i}}$. Then, the same argument as applied in the construction of polynomial kernels (Poggio, 1975) yields the desired result. ■

6.6 Efficient Computation of ANOVA Kernels

In the current section we assume that $\mathcal{X} = [1 : n_1] \times \cdots \times [1 : n_d]$ where $n_i \in \mathbb{N} \backslash \{1\}$ for all i. This situation is analogous to the one of Section 3.3, where we showed by a combinatorial reasoning that for Boolean variables $\sum_{|\mathbf{i}|=j} \mathbf{x}^{\mathbf{i}} \mathbf{x}'^{\mathbf{i}} = \binom{\langle \mathbf{x}, \mathbf{x}' \rangle}{j}$.

We now extend the reasoning of Section 3.3 to $\mathcal{X}$, as defined above. Note that the space of all functions depending on exactly j out of n coordinates of $\mathbf{x}$ is spanned by the indicator functions on j out of n coordinates. Therefore, a kernel formed by the dot product on the values of all such indicator functions will correspond to a Hilbert space spanning all j-coordinate interactions.

Since indicator functions only assume 0 and 1 as their values, the kernel on the dot product of the values of such indicator functions will only obtain nonzero contributions where the indicator functions on both $\mathbf{x}$ and $\mathbf{x}'$ yield 1. Denote by $a(\mathbf{x}, \mathbf{x}')$ the agreement between $\mathbf{x}$ and $\mathbf{x}'$:

$$a(\mathbf{x}, \mathbf{x}') := |\{\mathbf{x}_i = \mathbf{x}_i' \colon i \in [1 : n]\}| \tag{72}$$

and let

$$d(\mathbf{x}, \mathbf{x}') := n - a(\mathbf{x}, \mathbf{x}')$$

be the corresponding number of disagreements. Then the corresponding kernel can be found as $k_j(\mathbf{x}, \mathbf{x}') = \binom{a(\mathbf{x}, \mathbf{x}')}{j}$, since the latter is exactly the number of indicator functions that are nonzero simultaneously on $\mathbf{x}$ and $\mathbf{x}'$. While appealing in practice, due to its small number of computations, this kernel has an important downside, namely that it is *not* orthogonal to the spaces spanned by functions of interaction smaller than j.

If we cast these concerns aside, we may define a kernel by taking linear combinations of k_j, which leads to

$$k(\mathbf{x},\mathbf{x}') := \sum_{j=0}^{\infty} c_j k_j(\mathbf{x},\mathbf{x}') = \sum_{j=0}^{\infty} c_j \binom{a(\mathbf{x},\mathbf{x}')}{j} = \sum_{j=0}^{a(\mathbf{x},\mathbf{x}')} c_j \binom{a(\mathbf{x},\mathbf{x}')}{j}. \tag{73}$$

Finally, a suitable choice of c_j as ϵ^j leads to $k(\mathbf{x},\mathbf{x}') = (1+\epsilon)^{a(\mathbf{x},\mathbf{x}')}$. Note that this is identical to the diffusion kernel on discrete variables, as introduced by Kondor and Lafferty (2002). In other words, our reasoning provided us with an explicit feature-space representation of the diffusion kernel.

Going one step further, we may consider normalizing such a kernel, as suggested by Haussler (1999), This leads to

$$\begin{aligned} k_{\text{normalized}}(\mathbf{x},\mathbf{x}') &= \frac{k(\mathbf{x},\mathbf{x}')}{\sqrt{k(\mathbf{x},\mathbf{x})k(\mathbf{x}'\mathbf{x}')}} & (74)\\ &= (1+\epsilon)^{a(\mathbf{x},\mathbf{x}') - \overbrace{\frac{1}{2}(a(\mathbf{x},\mathbf{x}) + a(\mathbf{x}',\mathbf{x}'))}^{=n}} \\ &= (1+\epsilon)^{-d(\mathbf{x},\mathbf{x}')}. & (75) \end{aligned}$$

We now map $\mathbf{x}$ into a space of dummy variables. That is we map each coordinate $\mathbf{x}_i$ into the unit vector $\mathbf{e}_{\mathbf{x}_i} \in \mathbb{R}^{n_i}$:

$$\mathbf{x} \to E(\mathbf{x}) := (\mathbf{e}_{\mathbf{x}_1}, \mathbf{e}_{\mathbf{x}_2}, \ldots, \mathbf{e}_{\mathbf{x}_n})$$

An easy calculation then shows that

$$k_{\text{normalized}}(\mathbf{x},\mathbf{x}') = \exp(-\sigma\|E(\mathbf{x}) - E(\mathbf{x}')\|^2), \qquad \text{where } \sigma = \log(1+\epsilon)/2. \tag{76}$$

This reasoning explains to some extent why the commonly used heuristic of mapping discrete data into dummy variables and computing Gaussian RBF-kernels on them tends to work in practice better than one might expect at first glance.

6.7 Frames and Wavelets

Returning to Lemma 16 we can draw the following two conclusions:

Lemma 21 (Hilbert Space via Frames) *Any set of functions ψ_i in a Hilbert space H for which there exists at least one $\bar{f} \in H\backslash\{0\}$ for which $\sum_i \langle \bar{f}, \psi_i\rangle^2$ is finite, induces a Hilbert space $\mathcal{H}$ via*

$$\langle f, g\rangle_{\mathcal{H}} := \sum_i \langle f, \psi_i\rangle_H \langle \psi_i, g\rangle_H. \tag{77}$$

Here we identify all f, f' for which $\langle f, g\rangle = \langle f', g\rangle$ for all g and we assume the canonical completion for $\mathcal{H}$. Furthermore, if the ψ_i form a frame on H, $\mathcal{H}$ is automatically complete.

Proof Clearly (77) is a positive symmetric bilinear form in f and g, that is, $\mathcal{H}$ is a dot product space. Furthermore, $\mathcal{H}$ is nonempty since by assumption there exists at least one family of elements $\{\alpha\bar{f}\colon \alpha \in \mathbb{R}\}$, denoted by $\alpha\bar{f}$ with $\alpha \in \mathbb{R}$ for which (77) is well defined.

If ψ_i is a frame, it is is clear that each Cauchy sequence in $\mathcal{H}$ also has to be a Cauchy sequence in H and vice versa. Hence, also the limits have to agree which implies completeness. ■

Also note that by construction, ψ_i is a tight frame on $\mathcal{H}$. A second conclusion is that any Wavelet basis may also be used to build kernels via Lemma 15. This gives rise to the question whether $k(\mathbf{x}, \mathbf{x}')$, as defined in (56), can be computed efficiently.

Remark 22 (Kernels from Bounded-Support Wavelets) *Denote by ψ a wavelet with support* $\operatorname{supp}\psi \subset [0, a]$. *Then the kernel*

$$k(x, x') = \sum_{i\in\mathbb{N}, j\in\mathbb{Z}} c_i \psi(2^i x + j)\psi(2^i x' + j) \tag{78}$$

can be computed in less than $\lfloor a + 1\rfloor \hat{n}$ steps, where

$$\hat{n}(x, x') := \max\left\{n\colon |\lfloor 2^n x\rfloor - \lfloor 2^n x'\rfloor| \leq a\right\}.$$

The reason is that the sum over i contains only zero terms if $2^i x$ and $2^i x'$ are more than a apart with respect to $\mathbb{Z}$. Furthermore, for each i only $\lfloor a + 1\rfloor$ terms may be nonzero (and fewer for large i). Therefore $\lfloor a + 1\rfloor \hat{n}$ is an upper bound on the number of computations needed.

Computations are particularly fast in the case of the Haar wavelet basis. The latter is given by $\psi(x) = \chi_{[\frac{1}{2},1)}(x) - \chi_{[0,\frac{1}{2})}(x)$. It is commonly used in fast wavelet decompositions (Daubechies, 1992). Being an orthonormal basis, it clearly also satisfies the frame property. Using Remark 22 we can see that (78) can be computed efficiently for the Haar wavelet basis. Clearly the support of ψ is $[0, 1]$ and we can split the sum in (78) as follows:

$$\begin{aligned} k(x, x') &= \sum_{i\in\mathbb{N}, j\in\mathbb{Z}} c_i \psi(2^i x + j)\psi(2^i x' + j) \\ &= \sum_{i=0}^{\hat{n}(x,x')} \psi(x - \lfloor 2^i x\rfloor)c_i\psi(x' - \lfloor 2^i x'\rfloor) + \sum_{i=\hat{n}(x,x')+1} \psi(x - \lfloor 2^i x\rfloor)c_i\psi(x' - \lfloor 2^i x'\rfloor) \\ &= \sum_{i=0}^{\hat{n}} c_i - c_{\hat{n}} =: d(\hat{n}(x, x')). \end{aligned} \tag{79}$$

The values of $d(\hat{n})$ can be precomputed to facilitate efficient kernel evaluation. Note that the computational complexity increases only linearly with the number of dimensions, since the computation of the smallest common support can be done on a per-dimension basis.

A similar reasoning as the one employed for the Haar basis holds for wavelets which can attain only a finite number of different values. However, one needs to take care of the smallest common support set and ensure that positions in the support are computed properly.

7. Regularization Operators and Kernels on Discrete Objects

One of the considerations in the previous section was to find efficient means of computing kernels on spaces of bounded interaction. The latter is important, since it allows us to trade off hypotheses of varying degree of complexity via the regularization framework. In a nutshell, the functions of bounded interaction will play a role similar to that of functions with a small number of nonzero Fourier coefficients when considering translation invariant kernels on $\mathbb{R}$.

7.1 Fourier Transforms and Walsh Functions

We begin with a very simple, yet important special case: $X = \mathbb{B}^n$, i.e., the class of functions on Boolean variables. We will use the notation of Section 6.3. Here the kernel corresponding to the non-constant functions on $L_2(\mathbb{B}^n)$ is

$$\kappa(x_i, x_i') = (2\delta_{x_i, x_i'} - 1) = \begin{cases} 1 & \text{if } x_i = x_i' \\ -1 & \text{otherwise} \end{cases} \tag{80}$$

In other words, $\kappa(x_i, x_i')$ is the parity function. Also note that for $p_i = 2$ we have $\phi_{i1} = -\phi_{i2}$, as can be seen from (59). In this special case, the functions ϕ_{i0} and $\sqrt{2}\phi_{i1}$ form an orthonormal basis of the spaces $H_i = L_2(\{-1, 1\})$.

Finally, the space of all interactions of d terms is the space of all terms $\mathbf{x}^{\mathbf{i}}$ with $\mathbf{i} \in \{0, 1\}^n$ and $|\mathbf{i}| = d$. These are precisely all parity functions of order d that can be formed from n variables. It is well known (see e.g. Wickerhauser (1991) that these functions, sometimes also referred to as *Walsh*-functions, form an orthonormal basis on $L_2(\mathbb{B}^n)$. They play a role similar to that of the Fourier basis on $\mathbb{R}^n$. Likewise, the kernels on $[1 : p]$, as defined by (62), lead to a Fourier-type expansion on domains other than $\mathbb{B}^n$.

The main advantage of kernels like k_d is that we now may trade off functions of different complexity more easily, since k_d provides us with an efficient means of computing norms on the subspace of all parity functions of order d. This allows us to custom-tailor kernels via

$$k(\mathbf{x}, \mathbf{x}') := \sum_{i=0}^{n} c_i k_i(\mathbf{x}, \mathbf{x}') \text{ where } c_i \geq 0 \tag{81}$$

which has an easily understandable interpretation in terms of Hilbert spaces:

Proposition 23 *Denote by H_i the Hilbert spaces given by the subspaces of $L_2(\{-1, 1\}^n)$ corresponding to functions of d-term interactions. Then the kernel (81) is a reproducing Kernel to the Hilbert space H with the norm*

$$\|f\|_H^2 = \sum_{i=0}^{n} c_i^* \|f_i\|_{H_i}^2 \tag{82}$$

where f_i is the projection of f onto H_i and $c_i^* = c_i^{-1}$ if $c_i > 0$ and $c_i^* = 0$ otherwise.

Proof Without loss of generality we assume that all $c_i > 0$ (otherwise we simply drop the terms from the summation). Clearly (82) defines a norm and it follows from the polarization

equation that the corresponding dot product can be written as

$$\langle f, g\rangle_H = \sum_{i=0}^{n} c_i^* \langle f_i, g_i\rangle_{H_i} \tag{83}$$

where the f_i and g_i are the projections of f and g onto the subspaces H_i respectively. For k to be a reproducing kernel it has to satisfy the reproducing property. Since each $k_i(\mathbf{x}, \cdot) \in H_i$, we obtain

$$\langle f, k(\mathbf{x}, \cdot)\rangle_H = \sum_{i=1}^{n} \langle f_i, k_i(\mathbf{x}, \cdot)\rangle c_i^* c_i = \sum_{i=1}^{n} f_i(\mathbf{x}) c_i^* c_i = f(\mathbf{x}). \tag{84}$$

Here we assumed that f has a nontrivial projection only onto those spaces where $c_i > 0$. However this is easily satisfied since in an RKHS $k(\mathbf{x}, \cdot)$ spans the space. ■

Note that nowhere in the proof we needed the specific form of $\oplus_i H_i$ or of the individual spaces H_i, hence such compositions of Hilbert spaces are completely generic and can be used to join functions on completely different domains together into one RKHS.

7.2 Generic ANOVA Kernels

Now we can, in complete analogy to kernels defined on translation invariant data (see e.g., Smola et al. (1998), Schölkopf and Smola (2002) for a detailed overview), provide customized scaling coefficients c_i. If we use for instance the coefficients suggested by a Gaussian RBF kernel, i.e., $c_i \propto \exp(-\sigma i^2)$ one obtains

$$k_{\text{Gauss}-\text{Walsh}}(\mathbf{x}, \mathbf{x}') = \sum_{i=0}^{n} e^{-\sigma i^2} k_i(\mathbf{x}, \mathbf{x}'). \tag{85}$$

Note that due to the recurrence relation of Proposition 20, $k_{\text{Gauss}-\text{Walsh}}(\mathbf{x}, \mathbf{x}')$ is no more expensive to compute than a kernel taking only interactions of a fixed order, say d, into account.

Note that the reasoning put forward in Lemma 16 and 17, and Theorem 18 has been completely generic in the sense that we made no specific assumptions about the spaces H_i. This suggests that we might use just about any arbitrary Hilbert space H_i to build $H = \otimes_i H_i$ and after decomposing H_i into a constant subspace H_i^c and the orthogonal complement H_i^v, provide a decomposition of H into a direct sum of H_p where the Hilbert spaces H_p are spaces built by the interaction of p terms from the various H_i.

Furthermore, even if H_i is not a finite dimensional Hilbert space, such recurrence relations will allow one to find models of bounded order efficiently. The only restriction is that one needs to be able to decompose H_i into the space of constant functions and its orthogonal complement.

Finally, expansions such as (81) may allow one to determine appropriate model orders of an estimator rather efficiently: there is nothing to keep us from optimizing over Hilbert spaces of bounded interaction separately and readjusting terms or ignoring them if the corresponding RKHS norm is too small. Optimization over subspaces is guaranteed to converge exponentially, provided all subspaces are regularly visited.

8. Conclusions

We have developed new methods for dealing with instrinsically discrete data using kernels. We have shown how one can essentially implement decision trees using kernels and that the penalty regularization used in CART can be related to classical norm based regularization aring naturally in kernel methods. We have also indicated how the new kernels can be implemented efficiently. A consequence of the Boolean kernels we developed is that given a data set we can provide a lower bound on the complexity of a DNF representation that completely agrees with the data.

Acknowledgement

This work was supported by the Australian Research Council and by DFG Sm 62/1-1.

References

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the Annual Conference on Computational Learning Theory*, pages 144–152, Pittsburgh, PA, July 1992. ACM Press.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA, 1984.

C. J. C. Burges and V. Vapnik. A new method for constructing artificial neural networks. Interim technical report, ONR contract N00014-94-c-0186, AT&T Bell Laboratories, 1995.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, Cambridge, UK, 2000.

I. Daubechies. *Ten Lectures on Wavelets*. CBMS-NSF Regional Conferences Series in Applied Mathematics. SIAM, Philadelphia, PA, 1992. Notes from the 1990 CBMS-NSF Conference on Wavelets and Applications at Lowell, MA.

Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the International Conference on Machine Learing*, pages 148–146. Morgan Kaufmann Publishers, 1996.

Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. In J. Shavlik, editor, *Proceedings of the International Conference on Machine Learning*, pages 209–217, San Francisco, CA, 1998. Morgan Kaufmann Publishers.

D. Haussler. Convolutional kernels on discrete structures. Technical Report UCSC-CRL-99-10, Computer Science Department, UC Santa Cruz, 1999.

R. Khardon, D. Roth, and R. Servedio. Efficiency versus convergence o boolean kernels for on-line learning algorithms. In *Advances in Neural Information Processing Systems 14*, 2002.

R. S. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proceedings of the ICML*, 2002.

A. Kowalczyk, A. J. Smola, and R. C. Williamson. Kernel machines and boolean functions. In *Neural Information Processing Systems*, 2002.

M. Minsky and S. Papert. *Perceptrons: An Introduction To Computational Geometry*. MIT Press, Cambridge, MA, 1969.

A. B. J. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume 12, pages 615–622. Polytechnic Institute of Brooklyn, 1962.

T. Poggio. On optimal nonlinear associative recall. *Biological Cybernetics*, 19:201–209, 1975.

J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

A. Smola, B. Schölkopf, and K.-R. Müller. General cost functions for support vector regression. In T. Downs, M. Frean, and M. Gallagher, editors, *Proc. of the Ninth Australian Conf. on Neural Networks*, pages 79–83, Brisbane, Australia, 1998. University of Queensland.

A. J. Smola. Regression estimation with support vector learning machines. Diplomarbeit, Technische Universität München, 1996.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

M. V. Wickerhauser. Lectures on wavelet packet algorithms. 1991.

C. K. I. Williams. Prediction with Gaussian processes: From linear regression to linear prediction and beyond. In M. I. Jordan, editor, *Learning and Inference in Graphical Models*, pages 599–621. Kluwer Academic, 1998.