Topics in Statistical Learning Theory

Michael Nussbaum
Department of Mathematics, Cornell University

March 30, 2011

Abstract

These are course notes for MATH 7740, Statistical Learning Theory.

Contents

Primary References (a full list of references is at the end of these notes)**:**

[HTF] Hastie, T, Tibshirani, R. J. H. Friedman, R.H., *The Elements of Statistical Learning (Data Mining, Inference and Prediction),* Second Edition, Springer, 2009.

[DGL] Devroye, L, Gyorfi, L., Lugosi, G., *A Probabilistic Theory of Pattern Recognition*, Springer 1997

[Vap1] Vapnik, V. , *Statistical Learning Theory*, Wiley, 1998.

[Vap2] Vapnik, V. *The Nature of Statistical Learning Theory*, 2nd ed, Springer 1999

[ScS] Schölkopf, B. and Smola, A. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (Adaptive Computation and Machine Learning). MIT Press, 2002

[RW] Rasmussen, C. E. and Williams, C. K. I., *Gaussian Processes for Machine Learning* (Adaptive Computation and Machine Learning). MIT Press 2006

1 The Learning Problem

1.1 Supervised Learning as Classification

Example: handwritten digit recognition. In 1990, a U.S. Postal Service database contained 7300 training patterns from handwritten ZIP codes on envelopes from U.S. postal mail. Each image is a segment from a five digit ZIP code, isolating a single digit. The images are 16 x 16 eight-bit grayscale maps (with each pixel ranging in intensity from 1 to 255). Some sample images are shown in Figure 12 in the textbook.
The images have been normalized to have approximately the same size and orientation. The task is to predict, from the 16×16 matrix of pixel intensities, the identity of each image $\{0, 1, ..., 9\}$ quickly and accurately. If it is accurate enough, the resulting algorithm would be used as part of an automatic sorting procedure for envelopes. This is a classification problem for which the error rate needs to be kept very low to avoid misdirection of mail.
Thus each pattern is a matrix of dimension 16×16, or equivalently a vector of values in $16 \cdot 16 = 256$ dimensional space. The values are the intensities of each pixel, with possible values 1 to 255. This dataset can be written $X_1, \ldots, X_n$ where X_i is in $\mathbb{R}^d$ for $d = 256$. But this is a "training set", meaning that each X_i has already been "recognized" (apparently by eye), thus there are also given values $Y_1, \ldots, Y_n$ where each $Y_i \in \{0, 1, ..., 9\}$.
The training set together may be written $T^{(n)} = ((X_i, Y_i), i = 1, \ldots, n)$; a machine learning algorithm to be developed has to use the training set to **predict** the value of Y_i for a future or incoming X_i. In other words, the algorithm will **classify** all possible X_i into one of 10 possible categories (digits). Apparently prediction and classification are here used synonymously. Below we will make this notion precise, for an idealized setting where Y_i takes only two values 0 and 1.

The problem is called *supervised learning;* the "student" is identified with the algorithm and the "teacher" is the human eye which gave the correct digit Y_i for each X_i in the training set. The U.S. Postal Service database also contained an additional test set $((X_i, Y_i), i = 1, \ldots, n)$of $n = 2000$ patterns and corresponding digits. These are used to test the performance of the learning algorithm, e.g. to estimate its error rate. The division of data into training and test set is arbitrary. Training and test sets are routinely used in machine learning to "train" and test classifiers like neural networks, support vector machines (SVM) and others.

1.2 Unsupervised Learning

Example: human tumor microarray data. Data are a 6830×64 matrix of real numbers, each representing an expression level for a gene (row) and sample (column). Thus, one column (called a "sample" in this context) is a vector of 6830 gene expressions (ranging from -6 to 6) from one particular type of human tumor (breast cancer, melanoma and so on). Let x_i, $i = 1, \ldots, N$ be these colums ($N = 64$) in $\mathbb{R}^d$ for $d = 6830$. Each x_i carries a label, designating the type of cancer it came from. In an example of *unsupervised learning* one performs a cluster analysis, to find groups (clusters) within the x_i (ignoring the labels). Afterwards, the clustering obtained is compared with the labels, and one obtains a possible grouping of types of cancer, with respect to genetic similarity. Cluster analysis is an example of unsupervised learning since the analysis is performed without any "outcomes" Y_i indicating membership in a group. (In the human tumor microarray data, such Y_i are in fact present and are given by the cancer labels, but in the initial analysis they are not included. In this example, this reflects a choice how to perform an initial data analysis).

K-means clustering

(*follows [HTF], 460-62*) This algorithms requires a number of clusters K chosen. It is usually performed for various K and it is then judged which K gives the best compromise between data fit and complexity (i.e. number K of clusters).
Data are x_i, $i = 1, \ldots, N$ from $\mathbb{R}^d$. An underlying probabilistic assumption is that these are i.i.d random vectors from a density p on $\mathbb{R}^d$ which has multiple peaks, each corresponding to a "probability cluster". This may correspond to the density p being a mixture $p = \sum_{i=1}^{K} \lambda_i p_i$ where p_i are (unimodal) densities and $\sum_{i=1}^{K} \lambda_i = 1$. In the example, one may assume that each type of cancer appears in the sample with probability λ_i, and the type itself has a random gene expression vector with density p_i. The total sample x_i, $i = 1, \ldots, N$ may then be considered i.i.d. with density p. However, the clustering algorithms does not make use of a probability assumption about the sample. It can be described as follows:

- for each data point x_i, the closest cluster center (in Euclidean distance) is identified;
- each cluster center is replaced by the coordinatewise average of all data points that are closest to it.

To describe it in more detail, a prespecified number of clusters $K < N$ is postulated and each one is labeled by an integer $k \in \{1, \ldots, K\}$. Each observation is assigned to one and only one cluster. These assignments can be characterized by a many-to-one mapping, or encoder $k = C(i)$, that assigns the i-th observation to the k-th cluster. One seeks the particular encoder $C^*(i)$ that achieves the required goal (details below) , based on the dissimilarities $d(x_i, x_{i'})$ between every pair of observations. The most basic choice is squared Euclidean distance

$$d(x_i, x_{i'}) = \|x_i - x_{i'}\|^2 .$$

Generally, the encoder $C(i)$ is explicitly delineated by giving its value (cluster assignment) for each observation i. Thus, the "parameters" of the procedure are the individual cluster assignments for each of the N observations. These are adjusted so as to minimize a "loss" function that characterizes the degree to which the clustering goal is not met.
Since the goal is to assign close points to the same cluster, a natural loss (or "energy") function would be

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{i':C(i')=k} d(x_i, x_{i'}) \tag{1}$$

This criterion characterizes the extent to which observations assigned to the same cluster tend to be close to one another. It is sometimes referred to as the "within cluster" point scatter since

$$T = \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} d_{ii'} = \frac{1}{2} \sum_{k=1}^{K} \sum_{i:C(i)=k} \left(\sum_{i':C(i')=k} d_{ii'} + \sum_{i':C(i')\neq k} d_{ii'} \right)$$

or

$$T = W(C) + B(C)$$

where $d_{ii'} = d(x_i, x_{i'})$. Here T is the total point scatter, which is a constant given the data, independent of cluster assignment. The quantity

$$B(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{i':C(i')\neq k} d(x_i, x_{i'})$$

is the between-cluster point scatter. This will tend to be large when observations assigned to different clusters are far apart. Thus one has

$$W(C) = T - B(C)$$

and minimizing $W(C)$ is equivalent to maximizing $B(C)$.
Cluster analysis by combinatorial optimization is straightforward in principle. One simply minimizes W or equivalently maximizes B over all possible assignments of the N data points to K clusters. Unfortunately, such optimization by complete enumeration is feasible only for very small data sets. The number of distinct assignments is

$$S(N, K) = \frac{1}{K!} \sum_{k=1}^{K} (-1)^{K-k} \binom{K}{k} k^N$$

Comment. A "cluster" must have at least one element; the above formula gives the number of those assigments. To see this, consider first the number of assigments to one of K groups labelled $1, \dots, K$ when groups are allowed to have 0 elements. There are K^N such possible assigments (each data point is assigned to exactly one of the groups), and $K^N/K!$ is the number of assigments when group labels don't matter. To account for the restriction that each group has at least one element, subtract from $K!$ the number of assignments where at least one group has 0 elements. If one group has 0 elements then $(K-1)^N$ assigments remain, and summing over labels we get $K(K-1)^N$. Thus the "corrected" number of assigments is $\frac{1}{K!}\left(K^N - K(K-1)^N\right)$. But this correction needs to be corrected; indeed it counts twice those assignments where two groups have zero elements. We have to add these back, so we get $\frac{1}{K!}\left(K^N - K(K-1)^N + \binom{K}{K-2}(K-2)^N\right)$. Continuing in this manner, we obtain the formula.

For example, $S(10,4) = 34,105$ which is quite feasible. But $S(N,K)$ grows very rapidly with increasing values of its arguments. Already $S(19,4) \simeq 10^{10}$ and most clustering problems involve much larger data sets than $N = 19$. For this reason, practical clustering algorithms are able to examine only a very small fraction of all possible encoders $k = C(i)$. The goal is to identify a small subset that is likely to contain the optimal one, or at least a good suboptimal partition.
Such feasible strategies are based on iterative greedy descent. An initial partition is specified. At each iterative step, the cluster assignments are changed in such a way that the value of the criterion is improved from its previous value. When the prescription is unable to provide an improvement, the algorithm terminates with the current assignments as its solution.
In the K-means algorithm, the within-point scatter (1) can be written as

$$W(C) = \frac{1}{2}\sum_{k=1}^{K}\sum_{i:C(i)=k}\sum_{i':C(i')=k}\|x_i - x_{i'}\|^2 = \sum_{k=1}^{K} N_k \sum_{i:C(i)=k}\|x_i - \bar{x}_k\|^2 \tag{2}$$

where $\bar{x}_k = (\bar{x}_{1k}, \dots, \bar{x}_{dk})$ is the mean vector associated with the k-th cluster, and N_k is the cardinality of each cluster. Indeed, we have for given k and $C(i) = k$, $C(i') = k$

$$\|x_i - x_{i'}\|^2 = \|x_i - \bar{x}_k + \bar{x}_k - x_{i'}\|^2 = \|x_i - \bar{x}_k\|^2 + \langle x_i - \bar{x}_k, \bar{x}_k - x_{i'}\rangle + \|\bar{x}_k - x_{i'}\|^2,$$

$$\begin{aligned}\sum_{i':C(i')=k}\langle x_i - \bar{x}_k, \bar{x}_k - x_{i'}\rangle &= \left\langle x_i - \bar{x}_k, \sum_{i':C(i')=k}(\bar{x}_k - x_{i'})\right\rangle \\ &= \langle x_i - \bar{x}_k, N_k\bar{x}_k - N_k\bar{x}_k\rangle = 0\end{aligned}$$

hence

$$\begin{aligned}\sum_{i:C(i)=k}\sum_{i':C(i')=k}\|x_i - x_{i'}\|^2 &= N_k\sum_{i:C(i)=k}\|x_i - \bar{x}_k\|^2 + N_k\sum_{i':C(i')=k}\|x_{i'} - \bar{x}_k\|^2 \\ &= 2N_k\sum_{i:C(i)=k}\|x_i - \bar{x}_k\|^2\end{aligned}$$

which implies (2). Thus, the criterion is minimized by assigning the N observations to the K clusters in such a way that within each cluster the average dissimilarity of the observations from the cluster mean, as defined by the points in that cluster, is minimized.
An iterative descent algorithm for minimizing (2) over C can be obtained by noting that for any set of observations S

$$\bar{x}_S = \mathrm{argmin}_m \sum_{i \in S} \|x_i - m\|^2 . \tag{3}$$

Hence we can obtain a minimum by solving the enlarged optimization problem

$$\min_{C,\{m_k,k=1,\dots,K\}} \sum_{k=1}^{K} N_k \sum_{i:C(i)=k} \|x_i - m_k\|^2 \tag{4}$$

This can be minimized by an alternating optimization procedure as follows:
Algorithm.

1. For a given cluster assignment C, the total cluster variance (4) is minimized with respect to $\{m_k, k = 1, \dots, K\}$ yielding the means of the currently assigned clusters (3).

2. Given a current set of means $\{m_k, k = 1, \dots, K\}$, (4) is minimized by assigning each observation to the closest (current) cluster mean. That is,

$$C(i) = \mathrm{argmin}_{1 \le k \le K} \|x_i - m_k\|^2 .$$

3. Steps 1 and 2 are iterated until the assignments do not change.

One should start the algorithm with many different random choices for the starting means and compare results. .

2 The formal classification problem

2.1 Introduction

Suppose we observe a random variable X with values in $\mathbb{R}^d$. The distribution of X is given by one of two possible densities: f_0, f_1, and we have to decide which is the true one. The densities f_0, f_1 may be such that they have disjoint support S_0, S_1 respectively, or the support may be overlapping or even identical. In any case, each density is identified with a "class" or a category, and the decision: which density is the true one (where X came from), is called the **problem of classification.** Thus a **classifier** h is a measurable function: $h : \mathbb{R}^d \mapsto \{0, 1\}$, and "$h(X) = 1$" is interpreted as the decision "X came from density 1". In fact h is the same as a nonrandomized test function with critical region $\mathbf{1}_{\{h(X)=1\}}$.
This problem is just a special case of parameter estimation (or testing), when the parameter space is the set $\{0, 1\}$. It becomes different however when the two densities (or "classes") are unknown,

and instead there are only preliminary data available: $X_1, \ldots, X_n$ with information where each X_i came from: density 0 or density 1. This information can be written as a pair (X_i, Y_i) where Y_i takes values in the set $\{0,1\}$. The data (X_i, Y_i), $i = 1, \ldots, n$ are called the **training set.** From the data we have to "learn" the actual shape of the two densities f_0, f_1, and construct a decision function $\hat{h}$ accordingly. The decision function or classifier $\hat{h}$ is then supposed to act on a "new" incoming X and classify it as accurately as possible. Thus $\hat{h}$ depends on the training set $T^{(n)} = ((X_i, Y_i), i = 1, \ldots, n)$ and addditionally has argument X: thus $\hat{h}\left(T^{(n)}, X\right) = 1$ means that the classifier constructed on the training set $T^{(n)}$ classifies the new X as coming from density f_1.

A priori distributions. So far we have assumed that each X_i or X comes from one of the two densities f_0, f_1 in a nonrandom way (a special case of a "parametric model" $X \sim \{P_\vartheta, \vartheta \in \Theta\}$ where X is supposed to "come" from one of the probability measures P_ϑ). In classification theory one assumes however that there is an a priori distribution on the index: π is the probability $\pi = P(Y=1)$ and $1 - \pi = P(Y=0)$. Thus the index 0 or 1 is assumed random as well; giving rise to a joint distribution of (X, Y). The joint distribution is given by, for any measurable $A \subset \mathbb{R}^d$, $i \in \{0,1\}$

$$\begin{aligned} P(X \in A, Y = i) &= P(X \in A | Y = i) P(Y = i) \\ &= P(Y = i) \int_A f_i(x) dx \end{aligned}$$

where $P(Y=1) = \pi$ and $P(Y=0) = 1 - \pi$. The training set $T^{(n)} = ((X_i, Y_i), i = 1, \ldots, n)$ is assumed to consist of i.i.d. random variables having this joint distribution.

Error probabilities. Classifiers h should be optimal with respect to the error probability. Assume first that our classifier h does not depend on the training set (possibly it uses knowledge of f_0,f_1 and may be theoretical only, not available in reality). Initially, if there were no prior probability π, we could consider the error probability of first and second kinds:

$$P(h(X) = 1 | Y = 0) \text{ and } P(h(X) = 0 | Y = 1).$$

When the class index Y is also random it is natural to consider the total error probability:

$$\begin{aligned} P(h(X) \neq Y) &= P(h(X) = 1 | Y = 0)(1 - \pi) + P(h(X) = 0 | Y = 1)\pi \\ &= (1 - \pi) \int_{\{h(x)=1\}} f_0(x) dx + \pi \int_{\{h(x)=0\}} f_1(x) dx. \end{aligned} \tag{5}$$

2.2 Bayesian classification

The theoretical decision rule h^* which minimizes the above error probability is called the **Bayes rule** or Bayesian classifier. It is not based on any training set, but presupposes knowledge of the joint distribution of (X, Y). More specifically, it can be written in terms of the "true" **regression**

function

$$r(x) = E\left(Y|X=x\right) = P\left(Y=1|X=x\right) \tag{6}$$

$$= \frac{\pi f_1(x)}{\pi f_1(x) + (1-\pi) f_0(x)} \tag{7}$$

such that

$$h^*(x) = \begin{cases} 1 \text{ if } r(x) > 1/2 \\ 0 \text{ otherwise} \end{cases} \tag{8}$$

To comment, note that initially (a priori) Y is a Bernoulli r.v. with $P\left(Y=1\right)=\pi$. From the joint distribution of (X,Y) we then can find a conditional distribution of Y given $X=x$. This may also be called a **posterior distribution** of Y (after X has been observed). Since Y takes values in $\{0,1\}$, the posterior distribution is also Bernoulli, but now depending on x. Since for a Bernoulli, the expectation is the "success probability", we have

$$E\left(Y|X=x\right) = P\left(Y=1|X=x\right).$$

The formula (7) can easily be obtained from the formula for conditional densities. For this, we have to define a density p_Y of Y, with respect to counting measure on $\{0,1\}$ as

$$p_Y(0) = 1-\pi,\ p_Y(1) = \pi;$$

indeed a probability function for a discrete random variable can also be seen as a density. We then understand $f_i(x)$ as a conditional density of X given $Y=y$:

$$f_y(x) = p_X(x|y);$$

then

$$P\left(Y=y|X=x\right) = p_Y(y|x) = \frac{p_Y(y)p_X(x|y)}{p_X(x)}$$
$$= \frac{p_Y(y)p_X(x|y)}{p_X(x|0)p_Y(0) + p_X(x|1)p_Y(1)}$$

and for $y=1$ we obtain

$$P\left(Y=1|X=x\right) = \frac{f_1(x)\pi}{f_0(x)\left(1-\pi\right) + f_1(x)\pi}$$

which confirms our claim (7).
So far we have only defined a "Bayes rule" $h^*(x)$ in (8); before we prove the acutal optimality statement, let us note several equivalent forms:

$$h^*(x) = \begin{cases} 1 \text{ if } r(x) > 1/2 \\ 0 \text{ otherwise} \end{cases} = \begin{cases} 1 \text{ if } P\left(Y=1|X=x\right) > P\left(Y=0|X=x\right) \\ 0 \text{ otherwise} \end{cases}$$
$$= \begin{cases} 1 \text{ if } f_1(x)\pi > f_0(x)\left(1-\pi\right) \\ 0 \text{ otherwise} \end{cases}.$$

In the case $\pi = 1/2$ it coincides with the maximum likelihood rule:

$$h^*(x) = \begin{cases} 1 \text{ if } f_1(x) > f_0(x) \\ 0 \text{ otherwise} \end{cases}$$

Our performance criterion for any classification rule is the total error probability $P\left(h(X) \neq Y\right)$ given by (5). To express it in a different way, consider a statistical "loss function": $l(h(x), y)$ taken to be 0 for a correct decision $h(x)$, and 1 for an incorrect one. Thus

$$l(h(x), y) = \begin{cases} 0 \text{ if } h(x) = y \\ 1 \text{ otherwise} \end{cases}$$

or written as an indicator function $l(h(x), y) = \mathbf{1}_{\{h(x) \neq y\}}$. We then have

$$P\left(h(X) \neq Y\right) = E\mathbf{1}_{\{h(X) \neq Y\}} = El(h(X), Y)$$

i.e. the error probability is written as an expected loss, or as the **risk** of a decision rule.

2.1 Theorem *The Bayes rule h^* is optimal, that is, if h is any other classification rule then*

$$P\left(h^*(X) \neq Y\right) \leq P\left(h(X) \neq Y\right).$$

Comment. We are comparing h^* with other "nonpractical" classification rules which (possibly) utilize knowledge of f_0,f_1 and π. Intuitively, the "practical" rules where this knowledge is substituted by availability of training data T cannot be better. After the proof below we give a more formal reasoning on this.

Proof. This statement is exactly the same as the Bayes optimality of the maximizer of the posterior density in Bayesian inference. Assume we have densities $f_\vartheta(x)$ for observations X and an a priori density $\pi(\vartheta)$ for ϑ. (If the parameter space is finite, i.e. $\{0, 1\}$ then we take the density to be the probability function.). The total risk for any decision rule $h(x)$ with loss

$$l(h(x), \vartheta) = \begin{cases} 0 \text{ if } h(x) = \vartheta \\ 1 \text{ otherwise.} \end{cases}$$

(that is the zero-one loss: 0 loss for a correct decision, 1 for an incorrect one) is (writing $f(x) = \int f_\vartheta(x)\pi(\vartheta)d\vartheta$ for the marginal density of X)

$$\begin{aligned} EE_\vartheta l(h(X), \vartheta) &= \int\int l(h(x), \vartheta) f_\vartheta(x) dx \pi(\vartheta) d\vartheta \qquad (9) \\ &= \int\int l(h(x), \vartheta) f_\vartheta(x)\pi(\vartheta) dx d\vartheta \\ &= \int\int l(h(x), \vartheta) \frac{f_\vartheta(x)\pi(\vartheta)}{f(x)} d\vartheta f(x) dx \\ &= \int\int l(h(x), \vartheta) f(\vartheta|x) d\vartheta f(x) dx \end{aligned}$$

Thus for every x we need to minimize in h

$$\int_{\Theta} l(h(x),\vartheta) f(\vartheta|x)d\vartheta$$

that is, the a posteriori loss. What we can choose here is $h(x)$, that is for the given x we assign a value $h \in \{0,1\}$. Since ϑ takes values in $\{0,1\}$, the last integral is in fact

$$\begin{aligned} & l(h,0)P(\vartheta = 0|x) + l(h,1)P(\vartheta = 1|x) \\ & = l(h,0)P(\vartheta = 0|x) + (1 - l(h,0))\, P(\vartheta = 1|x)\,. \end{aligned}$$

since always $l(h,1) = 1 - l(h,0)$. To minimize this in h, we should select h such that the greater of the two probabilities gets factor 0 and the other gets 1. This gives

$$h(x) = \begin{cases} 0 \text{ if } P(\vartheta = 0|x) > P(\vartheta = 1|x) \\ 1 \text{ otherwise.} \end{cases}$$

This is the same as taking $h(x)$ to be the maximizer in ϑ of $P(\vartheta|x)$ (which is also $f(\vartheta|x)$), that is the maximizer of the posterior probability (or density). We have shown that this rule minimizes the total risk (9). ■

Recall that the Bayes rule presupposes knowledge of f_0 and f_1 (even if we assume π known, e.g. $\pi = 1/2$), it it usually not available in practice. The "practical" rules $\hat{h}$ which we will consider depend also on a training set $T^{(n)} = T$, they can be written $\hat{h}(T,X)$. But their absolute error probability $P\left(\hat{h}(T,X) \neq Y\right)$, *when P refers also to randomness in T*, cannot be better than $P(h^*(X) \neq Y)$.
Let us give a formal argument for that (almost obvious) claim. Recall that T and (X,Y) are assumed to be independent. Introduce $X^* := (T,X)$ and consider the classification (or prediction) problem with respect to the joint distribution (X^*,Y). Now we are trying to predict Y from $X^* = (T,X)$ but Y is independent of T. Using the Bayes rule in this context allows to dispense with T for the prediction.
Formally, we now admit classification rules $h(x^*)$ which are functions of x^*. The Bayes rule in this framework is

$$h^*(x^*) = \begin{cases} 0 \text{ if } P(Y = 0|x^*) > P(Y = 1|x^*) \\ 1 \text{ otherwise.} \end{cases} \tag{10}$$

We will show that this coincides with the previous Bayes rule, i.e. $h^*(x^*) = h^*(t,x)$ does not depend on t. For this, it suffices to show that $P(Y = 1|x^*) = P(Y = 1|t,x)$ does not depend on t. For this purpose, let $p_T(t)$ be the density of T with respect to an appropriate dominating measure (since $T = ((X_1,Y_1),\ldots,(X_n,Y_n))$, this one should be the n-fold product of ν, if ν is the product of Lebesgue measure with counting measure on $\{0,1\}$). Since T is independent of both X and Y, the joint density of (X^*,Y) is (for $x^* = (x,t)$

$$p_Y(y)p_{X^*}(x^*|y) = p_Y(y)p_X(x|y)p_T(t)$$

and hence

$$\begin{aligned} P\left(Y=y|X^*=x^*\right) &= p_Y(y|x^*) = \frac{p_Y(y)p_{X^*}(x^*|y)}{p_{X^*}(x^*)} \\ &= \frac{p_Y(y)p_X(x|y)p_T(t)}{p_X(x)p_T(t)} = \frac{p_Y(y)p_X(x|y)}{p_X(x)} \\ &= p_Y(y|x) = P\left(Y=y|X=x\right). \end{aligned}$$

The k classes case

Let us consider the case where there are possibly more than two classes, let us say K classes. The formal setup for Bayesian classification is very similar to the above. Suppose (X,Y) are random variables with a joint distribution such that $X=(X_1,\ldots,X_d)$ takes values in a set $\mathcal{X}\subset\mathbb{R}^d$ and Y takes values in a finite set $\mathcal{Y}=\{y_1,\ldots,y_K\}$. The y_j can be any labels but for simplicity we assume them to be $1,\ldots,K$. A classification rule h is a function $h:\mathcal{X}\mapsto\{1,\ldots,K\}$. The error rate of h is

$$L(h)=P\left(\{h(X)\neq Y\}\right)$$

where P refers to the joint distribution of (X,Y). Denote $p_k=P\left(Y=k\right)$ and assume that there are conditional densities of X given $Y=k$, denoted $f_k(x)$.
Define the Bayes rule again as

$$\begin{aligned} h^*(x) &= \text{argmax}_k P\left(Y=k|X=x\right) \\ &= \text{argmax}_k p_k f_k(x) \end{aligned}$$

where

$$P\left(Y=k|X=x\right)=\frac{p_k f_k(x)}{\sum_{r=1}^K p_r f_r(x)}$$

where argmax$_k$ means "the value of k that maximizes that expression". This classifier is also known as the "MAP" (maximum a posteriori) estimator. Indeed, in Bayesian terminology $P\left(Y=k|X=x\right)$ is the a posteriori probability of $Y=k$ (after $X=x$ has been observed).

2.2 Proposition *The statement of Theorem 2.1 is true in the k-class case.*

Proof. It follows verbatim the previous proof, where we write again ϑ for Y with a priori density $\pi(\vartheta)$ (or probability function) for ϑ given by $p_k=P\left(Y=k\right)=\pi(k)$ if $\vartheta=k$. Again we take a $0-1$ loss $l(h(x),\vartheta)=l(h(x),\vartheta)=\mathbf{1}_{\{h(x)\neq\vartheta\}}$. As above, for every x we then need to minimize in h

$$\int_\Theta l(h(x),\vartheta)f(\vartheta|x)d\vartheta=\sum_{k=1}^K l(h,k)P\left(Y=k|x\right).$$

To minimize this in h, we should select h such that the maximal probability $P\left(Y=k|x\right)$ gets factor 0 and the others all get factor 1. Thus for the maximal probability $P\left(Y=k|x\right)$ we have $l(h,k)=0$ which means $h=k$. This gives

$$h^*(x)=\text{argmax}_k P\left(Y=k|X=x\right)$$

as claimed. ∎

3 The perceptron algorithm

Suppose again that X is an $\mathbb{R}^d$-valued random vector and Y is a r.v. with two possible values, and (X, Y) have a joint probability distribution. For convenience we will assume now that these values are $\{-1, 1\}$. Consider a linear classification rule:

$$\phi(x) = \begin{cases} 1 \text{ if } a^\top x + a_0 \geq 0 \\ -1 \text{ otherwise.} \end{cases} = \operatorname{sgn}\left(a^\top x + a_0\right)$$

where $a \in \mathbb{R}^d$ and a_0 is a scalar. (Here $\operatorname{sgn}(t)$ is the sign of a scalar t; we take it to be 1 if $t \geq 0$ and -1 if $t < 0$.) In the history of artificial intelligence and neural network research, linear classifiers of this type were called **perceptrons.** (Fisher's linear discrimination analysis (LDA, 1936) is also given by a linear classifier). In this section we present a training algorithm for a perceptron, invented in 1957 at the Cornell Aeronautical Laboratory by Frank Rosenblatt. The algorithm created a great deal of interest when it was first introduced. It is guaranteed to converge if there exists a hyperplane that correctly classifies the training data.
The error probability of the rule is

$$L(\phi) = P(\phi(X) \neq Y)$$

Let a hyperplane in $\mathbb{R}^d$ be given by (w, b) where $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$, by the equation

$$\langle w, x \rangle + b = 0,\ x \in \mathbb{R}^d \tag{11}$$

where $\langle w, x \rangle = w^\top x$ is the scalar product. Note that for any scalar λ, the pairs (w, b) and $(\lambda w, b\lambda)$ define the same hyperplane. Under the assumption $\|w\| = 1$ the correspondence (w, b) to hyperplanes is one-to-one.
Suppose we have a training set $(x_1, y_1), \ldots, (x_n, y_n)$. It is called **linearly separable** if there is a hyperplane given by (w, b) such that all x_i with $y_i = 1$ are on one side of the hyperplane, while all x_i with $y_i = -1$ are on the other side. Let us make that notion precise.

3.1 Definition *Let (x_i, y_i) be an "example" (point in the training set $(x_1, y_1), \ldots, (x_n, y_n)$). The functional margin of (x_i, y_i) with respect to (w, b) is given by*

$$\gamma_i = y_i\left(\langle w, x_i \rangle + b\right).$$

With this definition, we say a training set is linarly separable if there exists a pair (w, b) defining a hyperplane such that all γ_i, $i = 1, \ldots, n$ have the same sign. Note that if we find a hyperplane such that all $\gamma_i \leq 0$, we can simply take $(-w, -b)$, and for this hyperplane we have all $\gamma_i \geq 0$.

3.2 Definition *The geometric margin of (x_i, y_i) with respect to (w, b) is*

$$\tilde{\gamma}_i = y_i\left(\left\langle \frac{w}{\|w\|}, x_i \right\rangle + \frac{b}{\|w\|}\right).$$

Here we have just $\tilde{\gamma}_i = \gamma_i / \|w\|$. Note that the equation

$$\left\langle \frac{w}{\|w\|}, x \right\rangle + \frac{b}{\|w\|} = 0, \; x \in \mathbb{R}^d$$

describes the same hyperplane as (11), but is a "canonical form" based on the unit vector $w/\|w\|$. Note that $|\tilde{\gamma}_i|$ measures the distance of x_i from the hyperplane given by (w, b) (picture here).

3.3 Definition *A hyperplane given by* (w, b) ***separates*** *the training set* $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ *if* $\min_{1 \le i \le n} \gamma_i > 0$. *In this case the geometric margin of* (w, b) *with respect to* S *is defined as* $\min_{1 \le i \le n} \tilde{\gamma}_i$.

A training set S is linearly separable if there exists (w, b) which separates S.

It is clear that the geometric margin of all separating hyperplanes is bounded from above. A hyperplane (w, b) which attains the largest possible value is called a *maximal margin hyperplane.* Its geometric margin is a characteristic of the training set S; it is called the *margin of the* (linearly separable) *training set* S.

Let $\eta > 0$ be some number, to be used as a "tuning parameter" in the algorithm.

———————————————————————————————————

Algorithm (Perceptron).

```
$w_0 \leftarrow \mathbf{0} \in \mathbb{R}^d$; $b_0 \leftarrow 0$, $k \leftarrow 0$
$R \leftarrow \max_{1 \le i \le n} \|x_i\|$

 repeat
        for $i = 1$ to $n$
                if      $y_i (\langle w_k, x_i \rangle + b_k) \le 0$ then
                        $w_{k+1} \leftarrow w_k + \eta y_i x_i$
                        $b_{k+1} \leftarrow b_k + \eta y_i R^2$
                        $k \leftarrow k + 1$
                endif
        endfor

until no mistakes made within the for loops
return $(w_k, b_k)$ where $k$ is the number of mistakes
```

———————————————————————————————————

A "mistake" occurs if $y_i (\langle w_k, x_i \rangle + b_k) \le 0$, i.e. a misclassification occurs (or a limiting case where x_i is exactly on the current hyperplane, $\gamma_i = 0$). Note that the starting value $(w, b) = (\mathbf{0}, 0)$ is not a hyperplane but it defines a trival linear classifier where $\phi(x) = 1$ for all x.

3.4 Theorem *(Novikoff, 1962). Let S be a nontrivial training set (not all y_i have the same sign), linearly separable. Let $R = \max_{1 \le i \le n} \|x_i\|$. Suppose there exist $(w_{\text{opt}}, b_{\text{opt}})$ such that $\|w_{\text{opt}}\| = 1$ and*

$$y_i \left(\langle w_{\text{opt}}, x_i \rangle + b_{\text{opt}} \right) \ge \gamma,\ i = 1, \ldots, n.$$

Then the perceptron algorithm stops after k_S steps, where

$$k_S \le \left(\frac{2R}{\gamma} \right)^2.$$

To comment, note that a large value of γ will cause the algorithm to stop earlier, while a large bound R has the opposite effect.

Proof. Augment the input vectors by an extra coordinate with value R. Denote the new vectors

$$\hat{x}_i = \begin{pmatrix} x_i \\ R \end{pmatrix}.$$

Similarly, add an extra coordinate to the weight vector w by setting

$$\hat{w} = \begin{pmatrix} w \\ b/R \end{pmatrix}.$$

The starting values are $\hat{w}_0 = \mathbf{0} \in \mathbb{R}^{d+1}$ and we denote the updates $\hat{w}_t$.

Let $\hat{w}_{t-1}$ be the augmented weight vector prior to the t-th mistake. The t-th update is performed when

$$y_i \langle \hat{w}_{t-1}, \hat{x}_i \rangle = y_i \left(\langle w_{t-1}, x_i \rangle + b_{t-1} \right) \le 0.$$

The update is

$$\begin{aligned} \hat{w}_t &= \begin{pmatrix} w_t \\ b_t/R \end{pmatrix} = \begin{pmatrix} w_{t-1} \\ b_{t-1}/R \end{pmatrix} + \eta y_i \begin{pmatrix} x_i \\ R \end{pmatrix} \\ &= \hat{w}_{t-1} + \eta y_i \hat{x}_i \end{aligned} \tag{12}$$

since $b_t = b_{t-1} + \eta y_i R^2$, hence $b_t/R = b_{t-1}/R + \eta y_i R$.

Now note

$$\begin{aligned} \langle \hat{w}_t, \hat{w}_{\text{opt}} \rangle &= \langle \hat{w}_{t-1}, \hat{w}_{\text{opt}} \rangle + \eta y_i \langle \hat{x}_i, \hat{w}_{\text{opt}} \rangle \\ &\ge \langle \hat{w}_{t-1}, \hat{w}_{\text{opt}} \rangle + \eta \gamma. \end{aligned}$$

By induction we obtain

$$\langle \hat{w}_t, \hat{w}_{\text{opt}} \rangle \ge t \eta \gamma. \tag{13}$$

Furthermore, from (12) we obtain

$$\begin{aligned} \|\hat{w}_t\|^2 &= \|\hat{w}_{t-1}\|^2 + 2\eta y_i \langle \hat{w}_{t-1}, \hat{x}_i \rangle + \eta^2 \|\hat{x}_i\|^2 \\ &\le \|\hat{w}_{t-1}\|^2 + \eta^2 \|\hat{x}_i\|^2 \\ &\le \|\hat{w}_{t-1}\|^2 + 2\eta^2 R^2 \end{aligned}$$

where we used $\eta y_i \langle \hat{w}_{t-1}, \hat{x}_i \rangle \leq 0$ for the first inequality (a "mistake" occurred in the t-th step), and $\|\hat{x}_i\|^2 = \|x_i\|^2 + R^2 \leq 2R^2$ for the second. By induction again

$$\|\hat{w}_t\|^2 \leq 2t\eta^2 R^2. \tag{14}$$

Now use Cauchy-Schwartz on (13) to obtain

$$\|\hat{w}_t\| \, \|\hat{w}_{\text{opt}}\| \geq t\eta\gamma$$

and write (14) as

$$\|\hat{w}_t\| \leq \eta R \sqrt{2t}.$$

Combine the last two inequalities to obtain

$$t\eta\gamma \leq \|\hat{w}_{\text{opt}}\| \, \eta R \sqrt{2t}$$

which implies

$$t^2\gamma^2 \leq \|\hat{w}_{\text{opt}}\|^2 R^2 2t$$

and hence

$$t \leq \|\hat{w}_{\text{opt}}\|^2 \frac{2R^2}{\gamma^2}.$$

Furthermore

$$\|\hat{w}_{\text{opt}}\|^2 = \|w_{\text{opt}}\|^2 + \frac{b_{\text{opt}}^2}{R^2}.$$

Now we have $b_{\text{opt}}^2 \leq R^2$ for nontrivial separation of S. (Indeed, $|b_{\text{opt}}|$ is the distance of the optimal hyperplane from the origin; any point x on the hyperplane x has $\|x\| \geq |b_{\text{opt}}|$. Hence (picture) there is a halfspace on one side of the hyperplane such that all points x in that halfspace also have $\|x\| \geq |b_{\text{opt}}|$, which implies $R \geq |b_{\text{opt}}|$). Hence

$$\|\hat{w}_{\text{opt}}\|^2 \leq \|w_{\text{opt}}\|^2 + 1 \leq 2$$

which implies the assertion

$$t \leq \left(\frac{2R}{\gamma}\right)^2.$$

■

However, if the training set is not linearly separable, the above online algorithm is not guaranteed to converge. Other training algorithms for linear classifiers are possible: e.g., support vector machine and logistic regression.

3.1 Intuitive background

Suppose for any training set, we try to find a hyperplane minimizing

$$J_n(w,b) = -\sum_{\gamma_i<0} y_i \left(\langle w, x_i\rangle + b\right) \tag{15}$$

under a restriction $\|w\| = 1$. Since $\gamma_i = y_i\left(\langle w, x_i\rangle + b)\right)$ and $\gamma_i < 0$ means that x_i is misclassified, we minimize the sum of the distances of x_i to the hyperplane for all misclassified x_i. Indeed, as we saw, $|y_i\left(\langle w, x_i\rangle + b\right)|$ is the distance of x_i to the hyperplane given by (w,b). The criterion $J_n(w,b)$ depends on the training set, and is defined for any training set $S = \{(x_1,y_1),\ldots,(x_n,y_n)\}$, separable or not.

The method of steepest descent (gradient method). Given a hyperplane $\hat{w}_t = (w_t, b_t)$, we try to find an improved value $\hat{w}_{t+1} = \hat{w}_t + \delta_t$ by adding a certain δ_t which points in the direction of "steepest descent". Since $J(\hat{w}_t) = J_n(w,b)$ is differentiable with respect to the $d+1$ variables $\hat{w}_t$, we can write for an increment vector v and a small ε

$$J(\hat{w}_t + \varepsilon v) \approx J(\hat{w}_t) + \varepsilon \langle \nabla J(\hat{w}_t), v\rangle \tag{16}$$

where $\nabla J(\hat{w}_t)$ is the gradient of the function $J(\cdot)$ at $\hat{w} = \hat{w}_t$:

$$\nabla J(\hat{w}) = \left(\frac{\partial J}{\partial \hat{w}^j}\right)_{j=1,\ldots,d+1}$$

where $\hat{w}^j$ denotes the components of the vector $\hat{w}$. Suppose we want to find a direction v (given by a unit vector $\|v\| = 1$) such that $J(\hat{w}_t + \varepsilon v) < J(\hat{w}_t)$ and the change is as large as possible. That means $\varepsilon\langle \nabla J(\hat{w}), v\rangle$ should be negative and $|\langle \nabla J(\hat{w}), v\rangle|$ should be maximal given $\|v\| = 1$. We have by Cauchy-Schwarz

$$|\langle \nabla J(\hat{w}), v\rangle| \le \|\nabla J(\hat{w})\| \, \|v\| = \|\nabla J(\hat{w})\|$$

with equality if v is a multiple of $\nabla J(\hat{w})$, i.e. $v = \nabla J(\hat{w}) / \|\nabla J(\hat{w})\|$. Thus the direction of steepest descent is given by $v = -\nabla J(\hat{w}) / \|\nabla J(\hat{w})\|$, and (16) becomes

$$J(\hat{w} + \varepsilon v) \approx J(\hat{w}) - \varepsilon \|\nabla J(\hat{w})\|.$$

This method (also called gradient method) is a commonly applied principle for iterative minimization of functions; in each successive step an "improvement" is found by taking a small step in the direction of steepest descent.

Applying this method to the target function (15), which is linear in $\hat{w} = (w,b)$ we find that the direction of steepest descent is given by the unit vector $v = v_0 / \|v_0\|$ where

$$v_0 = -\nabla J(\hat{w}) = \sum_{\gamma i<0} y_i \binom{x_i}{1}.$$

That could give rise to a stepwise minimization procedure of (15), sometimes called the "batch perceptron algorithm". However the algorithm we discussed is a modification, where each contribution from a false classification is treated separately:

$$J_{(i)}(w,b) = -y_i\left(\langle w, x_i\rangle + b\right)$$

with negative gradient

$$-\nabla J_{(i)}(\hat{w}) = y_i \begin{pmatrix} x_i \\ 1 \end{pmatrix}. \tag{17}$$

In each step in the "for $i=1$ to n" cycle, an improvement is found for one component $J_{(i)}(w,b)$ of the target function, corresponding to a particular misclassified x_i. The actual recipe $w_{k+1} \leftarrow w_k + \eta y_i x_i$, $b_{k+1} \leftarrow b_k + \eta y_i R^2$ represent a further modification ((17) would suggest $b_{k+1} \leftarrow b_k + \eta y_i$); the stepsize η is in some sense arbitrary.

Remark. The above discussion is somewhat flawed, since we treated $\hat{w} = (w,b)$ as $d+1$ free variables, i.e. we neglected the restriction $\|w\| = 1$. However the minimization of $J(\hat{w})$ should occur over the set $\{(w,b) : \|w\| = 1\}$; indeed otherwise (w,b) and $(\lambda w, \lambda b)$ represent the same hyperplane, while $(\lambda w, \lambda b)$ yields a smaller target function. So for a correct argument, all perturbations of a unit vector w should only be on the surface of the unit sphere, or (asymptotically) perpendicular to w.

3.2 Nonseparable training sets

We will give an illustration that the perceptron algorithm fails for nonseparable training sets. Consider two class-conditional laws on the real line given as follows: P_1 is uniform on the set $\{-1,2\}$ and P_{-1} is uniform on the set $\{-2,1\}$:

$$P_1(X=2) = P_1(X=-1) = 1/2,$$
$$P_{-1}(X=1) = P_{-1}(X=-2) = 1/2.$$

and $P(Y=1) = P(Y=-1) = 1/2$. Obviously P_1 and P_{-1} have disjoint support, so the Bayes risk is zero. A linear rule in this case takes the form

$$\phi(x) = \begin{cases} 1 \text{ if } wx+b \geq 0 \\ -1 \text{ otherwise.} \end{cases} = \operatorname{sgn}(wx+b)$$

where $w = \pm 1$ (a "threshold" rule). Since the present situation is symmetric in the two classes, it suffices to consider the case $w=1$. There is a linear rule with error probability $L(\phi) = P(\phi(X) \neq Y) = 1/4$: take $\phi(x) = 1$ if $x \geq 1.5$, $\phi(x) = -1$ otherwise. Indeed, if $Y=-1$ then $X=1$ or $X=-2$, i.e. $\phi(X) = -1$ always, and X is always classified correctly. If $Y=1$ then $X=-1$ or $X=2$, and X is misclassified only if $X=-1$, i.e. with probability $1/2$ conditionally on Y. Hence the overall probability of misclassification is $1/4$.

Now consider the linear rule which minimizes the perceptron criterion (15), which now becomes

$$J_n(b) = -\sum_{\gamma i<0} y_i\left(x_i + b\right)$$

Assume that n is large, so that we practically minimize an expectation. Since $\gamma_i = y_i\,(x_i + b)$, we can write

$$J_n(b) = \sum_{i=1}^{n} \left(-y_i\,(x_i + b)\right)_+$$

where $(z)_+ = z$ if $z \geq 0$, $(z)_+ = 0$ otherwise. Minimizing this is the same as minimizing

$$\frac{1}{n}\sum_{i=1}^{n} \left(-y_i\,(x_i + b)\right)_+ \approx E\left(-Y\,(X + b)\right)_+ =: J(b)$$

by the law of large numbers. Now (X, Y) has a uniform distribution on the 4 pairs $(2, 1)$, $(-1, 1)$, $(1, -1)$, $(-2, -1)$, i.e. each of these 4 points is taken with probability 1/4. Therefore

$$\begin{aligned} J(b) &= \frac{1}{4}\left(-\,(2 + b)\right)_+ + \frac{1}{4}\left(-\,(-1 + b)\right)_+ + \frac{1}{4}\left((1 + b)\right)_+ + \frac{1}{4}\left((-2 + b)\right)_+ \\ &= \frac{1}{4}\left(b - (-2)\right)_- + \frac{1}{4}\left(b - 1\right)_- + \frac{1}{4}\left(b - (-1)\right)_+ + \frac{1}{4}\left(b - 2\right)_+ \\ &= \frac{1}{4}\left(f_1(b) + f_2(b) + f_3(b) + f_4(b)\right) \end{aligned}$$

say, where we used notation $(z)_- = (-z)_+ = z$ if $z \leq 0$, $(z)_- = 0$ otherwise. Now note that inside the interval $-2 \leq b \leq 2$, we have $f_1(b) + f_4(b) = 0$ and outside the interval, the sum is positive. Furthermore, inside the interval $-1 \leq b \leq 1$ the we have $f_2(b) + f_3(b) = (1 - b) + (1 + b) = 2$ and outside this interval we have $f_2(b)+f_3(b) > 2$ (e.g. for $b > 1$ we have $f_2(b)+f_3(b) = f_3(b) = 1+b > 2$) so it is clear that $J(b)$ is minimzed for any $b \in [-1, 1]$, in particular $b = 0$ is a minimizer.

Now consider the error probability of the rule with $b = 0$, i.e.

$$\phi(x) = \left\{ \begin{array}{l} 1 \text{ if } x \geq 0 \\ -1 \text{ otherwise.} \end{array} \right. = \operatorname{sgn}(x)$$

Obviously it is

$$\begin{aligned} P\left(\phi(X) \neq Y\right) &= \frac{1}{2}P_1\left(X \leq 0\right) = \frac{1}{2}P_{-1}\left(X \geq 0\right) \\ &= \frac{1}{2}\cdot\frac{1}{2} + \frac{1}{2}\cdot\frac{1}{2} = \frac{1}{2}. \end{aligned}$$

This is very bad compared to the best linear rule having error probability 1/4. In fact error probability 1/2 is the worst possible in some sense; it can be achieved without a training set, by flipping a coin and classifying X according to the result.

4 Fisher's linear discriminant analysis

Fisher's linear discriminant analysis (LDA, 1936) has been derived from a certain empirical principle, involving sample means and covariances of the two classes. As almost always in statistics, such a "second moment" approach is connected to an intrinsic assumption that the data are normal. Assume we have a traing set (X_i, Y_i), $i = 1, \ldots, n$ where Y_i is uniform on $\{0,1\}$ and $X|Y = i$ is multivariate normal $N_d(m_i, \Sigma)$, $i = 0,1$, with the same covariance matrix Σ. Let us derive the Bayes rule for this case. We know that for equal prior probabilities $\pi = 1 - \pi = 1/2$, the Bayes rule coincides with the maximum likelihood rule:

$$h^*(x) = \begin{cases} 1 \text{ if } f_1(x) > f_0(x) \\ 0 \text{ otherwise} \end{cases}$$

where f_i is are the class-conditional densities, i. e. the densities of $N_d(m_i, \Sigma)$, $i = 0, 1$. We have

$$f_i(x) = \frac{1}{\sqrt{2\pi \det(\Sigma)}} \exp\left(-\frac{1}{2}(x - m_i)^\top \Sigma^{-1}(x - m_i)\right).$$

Hence the Bayes rule can be written

$$h^*(x) = \begin{cases} 1 \text{ if } r_1^2 < r_0^2 \\ 0 \text{ otherwise} \end{cases}$$

where

$$r_i^2 = (x - m_i)^\top \Sigma^{-1}(x - m_i),\ i = 0, 1$$

are the *squared Mahalanobis distances* of x from m_0 and from m_1. Hence the rule decides 1 if

$$(x - m_1)^\top \Sigma^{-1}(x - m_1) < (x - m_0)^\top \Sigma^{-1}(x - m_0)$$

which is equivalent to

$$-2x^\top \Sigma^{-1} m_1 + {m_1}^\top \Sigma^{-1} m_1 < -2x^\top \Sigma^{-1} m_0 + {m_0}^\top \Sigma^{-1} m_0$$

or to

$$x^\top \Sigma^{-1}(m_1 - m_0) > \frac{1}{2}\left({m_1}^\top \Sigma^{-1} m_1 - {m_0}^\top \Sigma^{-1} m_0\right)$$

Thus the **normal Bayes rule** is

$$h_N^*(x) = \begin{cases} 1 \text{ if } x^\top \Sigma^{-1}(m_1 - m_0) > \frac{1}{2}\left({m_1}^\top \Sigma^{-1} m_1 - {m_0}^\top \Sigma^{-1} m_0\right) \\ 0 \text{ otherwise} \end{cases} \tag{18}$$

Let us substitute m_i by estimates

$$\hat{m}_0 = \frac{1}{n_0} \sum_{j:Y_j=0} X_j,\ \hat{m}_1 = \frac{1}{n_1} \sum_{j:Y_j=1} X_j,$$
$$n_0 = \sum_{j:Y_j=0} 1,\ n_1 = \sum_{j:Y_j=1} 1$$

and Σ by an estimate

$$\hat{\Sigma} = (n-2)^{-1}(S_0 + S_1)$$

where S_i are the class scatter matrices

$$S_0 = \sum_{i:Y_i=0} (X_i - \hat{m}_0)(X_i - \hat{m}_0)^\top, \; S_1 = \sum_{i:Y_i=1} (X_i - \hat{m}_1)(X_i - \hat{m}_1)^\top.$$

The heuristics of the estimate $\hat{\Sigma}$ is obvious: both $\hat{\Sigma}_0 := S_0/(n_0-1)$ and $\hat{\Sigma}_1 := S_1/(n_1-1)$ are unbiased estimates of Σ, and $\hat{\Sigma}$ is the pooled estimate, i.e. an unbiased linear combination:

$$\hat{\Sigma} = \frac{n_0-1}{n-2}\hat{\Sigma}_0 + \frac{n_1-1}{n-2}\hat{\Sigma}_1.$$

This gives **Fisher's rule**

$$h_F(x) = \begin{cases} 1 \text{ if } x^\top \hat{\Sigma}^{-1}(\hat{m}_1 - \hat{m}_0) > \frac{1}{2}\left(\hat{m}_1{}^\top \hat{\Sigma}^{-1}\hat{m}_1 - \hat{m}_0{}^\top \hat{\Sigma}^{-1}\hat{m}_0\right) \\ 0 \text{ otherwise.} \end{cases} \tag{19}$$

4.1 Geometric interpretation

For this we drop the "hats" over $\hat{m}_i$ and $\hat{\Sigma}$, i.e. we refer to the normal Bayes rule (18). The whole reasoning is also valid for Fisher's rule, if the corresponding sample analogs are used. Assume $\Sigma = I$, otherwise we transform the data $\tilde{X}_i = \Sigma^{-1/2}X_i$ $i = 1, \dots, n$. Define

$$a = m_1 - m_0, \tag{20}$$

$$a_0 = \frac{1}{2}\left(\|m_0\|^2 - \|m_1\|^2\right), \tag{21}$$

then the normal Bayes rule uses a sample split

$$h_N^*(x) = \begin{cases} 1 \text{ if } x^\top a + a_0 > 0 \\ 0 \text{ otherwise.} \end{cases} \tag{22}$$

Thus it is a linear classifier, or a "perceptron" in an older terminology. Fisher's rule is also a linear classifier or "perceptron" where a, a_0 are estimated using the two sample means and the estimated covariance matrix from the training set. Thus the difference to Rosenblatt's perceptron algorithm lies only in how the a, a_0 are determined (the training method).

Let us describe another way of determining the constant a_0 according to the "Principle" below. Recall that if v, w are vectors and w is a unit vector ($\|w\| = 1$) then $v^\top w$ is the length of the projection of v onto the linear space spanned by w. Consider the unit vector corresponding to the optimal a above

$$\bar{a} = \frac{a}{\|a\|} = \frac{m_1 - m_0}{\|m_1 - m_0\|}.$$

Also, make m_0 the origin of the vector space by subtracting it from x, i.e. consider $x - m_0$. Now the vector $\bar{a}$ points from the origin in the direction of m_1 (since $m_0 + (m_1 - m_0) = m_1$).

Principle. *We want our classification rule such that if* $(x - m_0)^\top \bar{a}$ *is greater than one half the length* $\|m_1 - m_0\|$ *then we classify* 1*, otherwise we classify* 0.

This means we classify 1 if

$$(x - m_0)^\top \bar{a} > \frac{1}{2} \|m_1 - m_0\|$$

or equivalently

$$\begin{aligned}(x - m_0)^\top (m_1 - m_0) &> \frac{1}{2} \|m_1 - m_0\|^2 = \frac{1}{2} (m_1 - m_0)^\top (m_1 - m_0) \\ &= \frac{1}{2} m_1^\top (m_1 - m_0) - \frac{1}{2} m_0^\top (m_1 - m_0) .\end{aligned}$$

This is equivalent to

$$x^\top (m_1 - m_0) > \frac{1}{2} (m_1 + m_0)^\top (m_1 - m_0) = \frac{1}{2} \left(m_1^\top m_1 - m_0^\top m_0 \right)$$

which gives the same choice as (21) for a_0:

$$a_0 = -\frac{1}{2} \left(m_1^\top m_1 - m_0^\top m_0 \right) .$$

Thus the rule derived from the "Principle" coincides with the normal Bayes rule (18) in the case $\Sigma = I$. Fisher's LDA rule (19) was originally obtained from the empirical version of this idea: first $\hat{\Sigma}$ and $\hat{m}_0$, $\hat{m}_1$ are substituted for Σ, m_0, m_1, and then the "Principle" is applied to transformed data $\tilde{X}_i = \hat{\Sigma}^{-1/2} X_i$ $i = 1, \ldots, n$.

4.2 Linear inconsistency

Since Fisher's LDA is based on the first two moments of the training set, and thus on an underlying normality assumption, it is bound to encounter difficulties for nonnormal distributions. We will find an example where Fisher's rule is **linearly inconsistent**, i.e. for training set size $n \to \infty$ does not asymptotically attain the best linear risk. We cannot expect Fisher's rule to attain the Bayes risk in the general case, since it is a linear rule and the Bayes rule may be nonlinear. However we will show that even if the classes (distributions) can be separated by a linear rule, Fisher's LDA may not asymptotically find that rule.

Define class-conditional distributions as follows. Consider three points v_1,v_2,v_3 in $\mathbb{R}^2$ forming a equilateral triangle with side length l, i.e we have

$$\|v_1 - v_2\| = \|v_1 - v_3\| = \|v_2 - v_3\| = l.$$

Let P_0 be the uniform distribution on the vertices, i.e.

$$P_0 (X = v_i) = \frac{1}{3}, \; i = 1, 2, 3.$$

Consider another equilateral triangle with vertices w_1,w_2,w_3 in $\mathbb{R}^2$ and the same sidelength, such that the two triangles do not intersect. Let P_1 be the uniform distribution on the vertices w_1,w_2,w_3. We will show that a) l can be chosen such that P_0, P_1 have unit covariance matrix, b) the vertices can be chosen such that P_0, P_1 are linearly separable, c) the vertices can also be chosen such that Fisher's rule does not provide linear separation, i.e. it does not find a separating hyperplane even for training set size $n \to \infty$.

Description of a equilateral triangle. We start with side length $l = 2$ and vertices $v_1 = (1, 0)$, $v_2 = (-1, 0)$, and $v_3 = \left(0, \sqrt{3}\right)$. To check equal side length, use Pythagoras: we find $\|v_3 - v_2\| = \sqrt{1^2 + 3} = 2$. To find the center of this triangle (v_0, say), compute the expected value of P_0: if $X \sim P_0$ then

$$\begin{aligned} v_0 = EX = \frac{1}{3}\sum_{i=1}^{3} v_i &= \frac{1}{3}\begin{pmatrix}1\\0\end{pmatrix} + \frac{1}{3}\begin{pmatrix}-1\\0\end{pmatrix} + \frac{1}{3}\begin{pmatrix}0\\\sqrt{3}\end{pmatrix} \\ &= \begin{pmatrix}0\\1/\sqrt{3}\end{pmatrix}. \end{aligned}$$

For the covariance matrix $\Sigma := EXX^\top - (EX)(EX)^\top$, note

$$\begin{aligned} EXX^\top &= \\ &= \frac{1}{3}\begin{pmatrix}1&0\\0&0\end{pmatrix} + \frac{1}{3}\begin{pmatrix}1&0\\0&0\end{pmatrix} + \frac{1}{3}\begin{pmatrix}0&0\\0&3\end{pmatrix} \\ &= \begin{pmatrix}2/3&0\\0&1\end{pmatrix}, \\ (EX)(EX)^\top &= \begin{pmatrix}0&0\\0&1/3\end{pmatrix}, \end{aligned}$$

and hence

$$\Sigma := \begin{pmatrix}2/3&0\\0&2/3\end{pmatrix}.$$

Thus $\Sigma = (2/3)I$ where I is the unit matrix, and we see that the r.v. λX for $\lambda = \sqrt{3/2}$ has unit covariance matrix. Thus changing the side length of the equilateral triangle to $l = 2\lambda = \sqrt{6}$ gives a unit covariance matrix for the uniform distribution on the vertices. For our point about Fisher's rule, based on the geometry, the factor λ is irrelevant, and we will continue with $l = 2$.

Remark: It is easily seen that instead of the the triangle with vertices $\{v_1, v_2, v_3\}$, we could have taken $\{v_1, v_2, -v_3\}$ and obtained the same covariance matrix Σ, but center point $-v_0$.

Now we will change the position of the triangle $\{v_1, v_2, v_3\}$ by shifting the center (and thus all the vertices). Draw a horizontal line though the center point v_0 and find the intersection of this line with the left upper side of the triangle. i.e. with the line joining v_2 and v_3. Call this point $\tilde{v}$; we will shift the triangle so that $\tilde{v}$ comes to coincide with the origin. To find the x-component of

$\tilde{v} = (\tilde{v}_x, \tilde{v}_y)$, note that $\tilde{v}_y = 1/\sqrt{3}$ and by similarity of triangles

$$\frac{\sqrt{3}}{1} = \frac{\tilde{v}_y}{1 - |\tilde{v}_x|} = \frac{1/\sqrt{3}}{1 - |\tilde{v}_x|}$$

which gives $1 - |\tilde{v}_x| = 1/3$ and hence $\tilde{v}_x = -2/3$. Hence

$$\tilde{v} = \begin{pmatrix} -2/3 \\ 1/\sqrt{3} \end{pmatrix}.$$

Now subtract the vector $\tilde{v}$ from all three vertices, to obtain a shifted triangle with vertices

$$\begin{aligned} v_1^* = v_1 - \tilde{v} &= \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} -2/3 \\ 1/\sqrt{3} \end{pmatrix} = \begin{pmatrix} 5/3 \\ -1/\sqrt{3} \end{pmatrix}, \\ v_2^* = v_2 - \tilde{v} &= \begin{pmatrix} -1 \\ 0 \end{pmatrix} - \begin{pmatrix} -2/3 \\ 1/\sqrt{3} \end{pmatrix} = \begin{pmatrix} -1/3 \\ -1/\sqrt{3} \end{pmatrix} \\ v_3^* = v_3 - \tilde{v} &= \begin{pmatrix} 0 \\ \sqrt{3} \end{pmatrix} - \begin{pmatrix} -2/3 \\ 1/\sqrt{3} \end{pmatrix} = \begin{pmatrix} 2/3 \\ 2/\sqrt{3} \end{pmatrix} \end{aligned}$$

Now define $\tilde{P}_0$ to be the uniform distribution on $\{v_1^*, v_2^*, v_3^*\}$. It follows from the above that its covariance matrix is $\Sigma = (2/3)I$ (since the covariance matrix does not change under translations by a constant vector $\tilde{v}$).
Define another equilateral triangle as follows: it has vertices $\{-v_1^*, -v_2^*, -v_3^*\}$. This can be visualized as a double mirror image of $\{v_1^*, v_2^*, v_3^*\}$: first mirrored at the x-axis and then mirrored at the y-axis. Both triangles contain the origin in one of their sides, and in fact they have a section in common: the segment joining v_1^* and $-v_1^*$, i.e. the set $\{tv_1^*,\ -1 \leq t \leq 1\}$. *Now define $\tilde{P}_1$ to be the uniform distribution on $\{-v_1^*, -v_2^*, -v_3^*\}$.* Clearly that has the same covariance matrix Σ as $\tilde{P}_0$, (since it can also be obtained as a translation of the triangle given by $\{v_1, v_2, -v_3\}$, which gives the same Σ according to the remark above.
The distributions $\tilde{P}_0$, $\tilde{P}_1$ are separable (the Bayes risk is zero) , since their supports do not intersect. However they are not *linearly separable*: there is no straight line separating them; in fact the only candidate straight line $\{tv_1^*,\ t \in \mathbb{R}\}$ contains one vertex from each triangle.
However by "moving them apart" by an amount ε makes them linearly separable. Define the vector $u = (1, 0)$ and consider the two triangles $\{v_1^* + \varepsilon u, v_2^* + \varepsilon u, v_3^* + \varepsilon u\}$ and $\{-v_1^* - \varepsilon u, -v_2^* - \varepsilon u, -v_3^* - \varepsilon u\}$. Let P_0, P_1 be the respective uniform distributions on the vertices. If $\varepsilon > 0$, the distributions are now linearly separable (e.g. by the hyperplane $\{tv_1^*,\ t \in \mathbb{R}\}$). On the other hand, if $\varepsilon < 1/3$ then they are *not linearly separable by the y-axis*, and not by any hyperplane parallel to it. This means that Fisher's LDA fails for P_0, P_1, as we will discuss now

Behaviour of Fisher's LDA for distributions P_0, P_1. In a large training set, we may identify Fisher's rule with the normal Bayes rule (18) for separating laws $N(m_0, \Sigma)$, $N(m_1, \Sigma)$. Recall that our actual distributions are not normal now; data come from the two P_0, P_1 constructed above with mean vectors m_0, m_1 and covariance matrix Σ. When we use Fishers' rule, the sample means

$\hat{m}_i$ will converge to m_i and $\hat{\Sigma}$ will converge to Σ. So asymptotically we are using the normal Bayes rule (18) on actual distributions P_0, P_1. Since Σ is a multiple of the unit matrix, we have

$$h_N^*(x) = \begin{cases} 1 \text{ if } x^\top (m_1 - m_0) > c \\ 0 \text{ otherwise} \end{cases} \tag{23}$$

where $c = \frac{1}{2}\left(m_1{}^\top m_1 - m_0{}^\top m_0\right)$. In our construction of P_0, P_1, the two mean points were such that they both are on the x-axis:

$$m_0 = v_0 - \tilde{v} + \varepsilon u = \begin{pmatrix} 0 \\ 1/\sqrt{3} \end{pmatrix} - \begin{pmatrix} -2/3 \\ 1/\sqrt{3} \end{pmatrix} + \varepsilon \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 2/3 + \varepsilon \\ 0 \end{pmatrix},$$
$$m_1 = -v_0 + \tilde{v} - \varepsilon u = \begin{pmatrix} -2/3 - \varepsilon \\ 0 \end{pmatrix},$$

hence $m_1 - m_0$ is a vector parallel to the (or on the) x-axis. Thus any rule of type (23) projects the support of the two laws P_0, P_1 on the x-axis and attempts to separate them by a cut.

Now projected on the x-axis, P_0 will be uniform on the three points $\{5/3 + \varepsilon, -1/3 + \varepsilon, 2/3 + \varepsilon\}$ and P_1 will be uniform on the three points $\{-5/3 - \varepsilon, 1/3 - \varepsilon, -2/3 - \varepsilon\}$. More precisely, these are projections of laws: call them $P_{x,0}$ and $P_{x,1}$. Though $P_{x,0}$ tends to be more on the positive side and $P_{x,1}$ tends to be more on the negative side, as long as $0 < \varepsilon < 1/3$, we will have $-1/3 + \varepsilon < 0 < 1/3 - \varepsilon$, and hence one point from the support of $P_{x,0}$ will be negative and one point from the support of $P_{x,1}$ will be positive. Thus $P_{x,0}$ and $P_{x,1}$ cannot be separated by a linear rule (a split): at least one point is always misclassified, from either $P_{x,0}$ and $P_{x,1}$ (one may choose) so the best linear error probability is $(1/3) \cdot (1/2) = 1/6$. But Fisher's rule (23) uses $c = 0$ and hence has error probability $1/3$.

Recall that the original distributions P_0, P_1 are linearly separable by the hyperplane $\{tv_1^*,\ t \in \mathbb{R}\}$.

Remark. Problem 4.9., p. 58 in [DGL] claims that there is a distribution of (X, Y), $X \in \mathbb{R}^2$ such that Fisher's rule is even worse: every split rule using projection onto the vector $\hat{m}_1 - \hat{m}_0$ has error probability $1/2 - \varepsilon$ and Fisher's rule has error probability $1 - \varepsilon$, In our example above, these numbers are $1/6$ and $1/3$ respectively.

5 Kernelization

There is a trick called kernelization for improving a computationally simple classifier h. The idea is to map the covariate X which takes values in $\mathbb{R}^d$ into a higher dimensional space $\mathcal{Z}$ and apply the classifier in the bigger space $\mathcal{Z}$. This can yield a more flexible classifier while retaining computationally simplicity.
The standard example of this idea is the following. Let $d = 2$ and assume a training set (X_i, Y_i), $i = 1, \dots, n$ can be separated into two groups using an ellipse. Suppose we are using "ellipse classifiers" given by (where $x = (x_1, x_2)$)

$$h(x) = \begin{cases} 1 \text{ if } x^\top Ax > c \\ 0 \text{ otherwise} \end{cases}$$

where A is a symmetric positive definite 2×2 matrix and $c > 0$. The decision boundary here is an ellipse:

$$\Big\{x : x^\top Ax = c\Big\} = \big\{x : x_1^2 a_{11} + 2x_1x_2a_{12} + x_2^2 a_{22} = c\big\}$$

with shape given by the matrix A and "width" given by c. This does not give all ellipses in the plane; these would be given by

$$\Big\{x : (x-b)^\top A(x-b) = c\Big\},$$

with a vector $b \in \mathbb{R}^2$. But for the example we set $b = 0$ and consider only ellipses around the origin. Define a mapping ϕ by

$$z = (z_1, z_2, z_3) = \phi(x) = (x_1^2, \sqrt{2}x_1x_2, x_2^2).$$

Thus, ϕ maps $\mathbb{R}^2$ into $\mathcal{Z} = \mathbb{R}^3$. If the original data are separable by an elliptic decision boundary, then, in the higher-dimensional space $\mathcal{Z}$, the (X_i, Y_i) are separable by a linear decision boundary. In other words, a linear classifier in a higher-dimensional space corresponds to a nonlinear classifier in the original space.
The point is that to get a richer set of classifiers we do not need to give up the convenience of linear classifiers. We simply map the covariates to a higher-dimensional space. This is akin to making linear regression more flexible by using polynomials. The space $\mathcal{Z}$ of higher dimension is often called the **feature space** and the original $\mathbb{R}^d$ is called input space.
There is a potential drawback. If we significantly expand the dimension of the problem, we might increase the computational burden.

Example. Suppose x has dimension $d = 56$ and we wanted to use all fourth-order terms, i.e. all terms $x_i^a x_j^b x_k^c x_l^d$ where the sum of powers $a + b + c + d$ equals 4. These are all terms of type x_i^4, $x_i^3 x_j$, $x_i^2 x_j^2$ etc. For the number of these, consider only terms of type $x_i x_j x_k x_l$ where all 4 indices i, j, k, l are different. There are

$$\binom{56}{4} = \frac{56 \cdot 55 \cdot 54 \cdot 53}{1 \cdot 2 \cdot 3 \cdot 4} = 367,290$$

of these, so the dimension of $\mathcal{Z}$ exceeds this number.

We are spared this computational nightmare by the following two facts. First, many classifiers do not require that we know the values of the individual points but, rather, just the inner product between pairs of points. Second, notice in our example that the inner product in $\mathcal{Z}$ can be written

$$\begin{aligned}\langle z, \tilde{z}\rangle = \langle \phi(x), \phi(\tilde{x})\rangle &= x_1^2\tilde{x}_1^2 + 2x_1\tilde{x}_1x_2\tilde{x}_2 + x_2^2\tilde{x}_2^2 \\ &= (\langle x, \tilde{x}\rangle)^2 =: K(x, \tilde{x}).\end{aligned}$$

Thus we can compute $\langle z, \tilde{z}\rangle$ without ever computing $Z_i = \phi(X_i)$.
To summarize, kernelizalion involves finding a mapping $\phi : \mathbb{R}^d \mapsto \mathcal{Z}$ and a classifier such that:

1. $\mathcal{Z}$ has higher dimension than X and so leads a richer set of classifiers.
2. The classifier only requires computing inner products.
3. There is a function K, called a kernel such that $\langle \phi(x), \phi(\tilde{x})\rangle = K(x, \tilde{x})$.
4. Everywhere the term $\langle x, \tilde{x}\rangle$ appears in the algorithm, replace it with $K(x, \tilde{x})$.

In fact, we never need to construct the mapping ϕ at all. We only need to specify a kernel $K(x, \tilde{x})$ that corresponds to $\langle \phi(x), \phi(\tilde{x})\rangle$ for some ϕ. This raises an interesting question: given a function of two variables $K(x, y)$, does there exist a function $\phi(x)$ such that $K(x, y) = \langle \phi(x), \phi(y)\rangle$? The answer is provided by

Mercer's Theorem. *Let C be a compact subset of $\mathbb{R}^d$ and let $L_2(C)$ be the corresponding L_2-space, i..e. the Hilbert space of square integrable functions on C endowed with inner product*

$$\langle f, g\rangle = \int f(u)g(u)du, \ f, g \in L_2(C)$$

and associated norm $\|f\| = (\langle f, f\rangle)^{1/2}$. Suppose $K(u, v)$ is a continuous symmetric ($K(u, v) = K(v, u)$) and real valued function (kernel) on $C \times C$ which is positive definite, i.e. for all $f \in L_2(C)$ we have

$$\int_{C\times C} K(u, v)f(u)f(v)dudv \geq 0. \tag{24}$$

Then we can expand $K(u, v)$ in a uniformly convergent series (on $C \times C$) in terms of normalized eigenfunctions $\phi_j \in L_2(C)$ with $\|\phi_j\| = 1$ and nonnegative associated eigenvalues $\lambda_j \geq 0$

$$K(u, v) = \sum_{j=1}^{\infty} \lambda_j \phi_j(u)\phi_j(v). \tag{25}$$

Remark. The theorem is analogous to the spectral decomposition of real symmetric nonnegative definite matrices. Replace the functions f, g by finite dimensional vectors $f, g \in \mathbb{R}^k$ and assume we have a symmetric matrix $k \times k$ matrix $K = (K(i,j))$ where in analogy to (24) for all $f \in \mathbb{R}^k$ with i-th component $f(i)$

$$f^\top K f = \sum_{i,j=1}^{k} f(i)f(j)K(i,j) \geq 0.$$

Then we can expand K into a finite sum in terms of normalized eigenvectors $\phi_j \in \mathbb{R}^k$ with $\|\phi_j\| = 1$ (euclidean norm) and nonnegative associated eigenvalues $\lambda_j \geq 0$

$$K = \sum_{j=1}^{k} \lambda_j \phi_j \phi_j^\top$$

which can also be written, if $\phi_j(i)$ denotes the i-th component of vector ϕ_j

$$K(i,s) = \sum_{j=1}^{k} \lambda_j \phi_j(i) \phi_j(s)$$

which is analogous to (25).

Use of Mercer's Theorem with kernelization. Suppose we use a map ϕ: $\mathbb{R}^d \longmapsto \mathbb{R}^s$ from input space $\mathbb{R}^d$ into a higher dimensional feature space $\mathbb{R}^s$, and that the transformed training set $(\phi(X_1), Y_1), \ldots, (\phi(X_n), Y_n)$ is linearly separable (just for illustration; the kernelization method does not require this assumption). A separating hyperplane for the transformed training set can be expressed for $z \in \mathbb{R}^s$

$$(\langle a, z \rangle + a_0) = 0$$

thus in terms of the input space argument $x \in \mathbb{R}^d$

$$(\langle a, \phi(x) \rangle + a_0) = 0.$$

In all classifiers described so far, computation of the optimal vector vector a (for classification in feature space) requires only evaluation of scalar products $\langle Z_i, Z_j \rangle$ where $Z_j = \phi(X_j)$. Thus we have to compute all scalar products $\langle \phi(X_i), \phi(X_j) \rangle$. Define

$$K(x,y) = \langle \phi(x), \phi(y) \rangle$$

and assume that x, y are from a compact set $C \subset \mathbb{R}^d$ (the training set is finite and thus contained in a ball of finite radius in $\mathbb{R}^d$). We will show that the function $K(x,y)$ defines a positive definite kernel. Suppose $f \in L_2(C)$, then if $\phi_j(x)$, $j = 1, \ldots, s$ are the components of $\phi(x) \in \mathbb{R}^s$

$$\int_{C \times C} K(x,y) f(x) f(y) dx dy = \int_{C \times C} \langle \phi(x), \phi(y) \rangle f(x) f(y) dx dy$$

$$= \int_{C\times C}\left(\sum_{j=1}^{s}\phi_j(x)\phi_j(y)\right) f(x)f(y)dxdy = \sum_{j=1}^{s}\int_{C\times C}\phi_j(x)f(x)\phi_j(y)f(y)dxdy$$
$$= \sum_{j=1}^{s}\left(\int_C \phi_j(x)f(x)dx\right)^2 \geq 0.$$

Clearly $K(x,y)$ is also symmetric and continuous in x, y if the mapping $\phi(x)$ is continuous. Thus $K(x,y)$ is a kernel satisfying the conditions of Mercer's theorem.
The idea of kernelized classification now is to replace the special kernel $K(x,y) = \langle\phi(x), \phi(y)\rangle$ where $\phi(x)$ is a map into feature space, with a general kernel $K(x,y)$ satisfying the conditions of Mercer's theorem. The question arises: can the resulting classification procedure still be understood as linear classification in some feature space ?
The answer is given by Mercer's theorem. According to (25) we have

$$K(x,y) = \sum_{j=1}^{\infty}\lambda_j\phi_j(x)\phi_j(y).$$

We have to consider an infinite dimensional feature space l_2 (the space of sequences $(m_i)_{i=1}^{\infty}$ such that $\sum_{i=1}^{\infty} m_i^2 < \infty$) and define a mapping from $\mathbb{R}^d$ into this space by

$$\phi(x) = \left(\lambda_j^{1/2}\phi_j(x)\right)_{j=1}^{\infty}$$

Then, if the scalar product in l_2 is given by the obvious infinite series then

$$K(x,y) = \langle\phi(x), \phi(x)\rangle = \sum_{j=1}^{\infty}\lambda_j\phi_j(x)\phi_j(y).$$

5.1 Kernelization of Fisher's LDA

Let us first assume that we use a map ϕ: $\mathbb{R}^d \longmapsto \mathbb{R}^s$ from input space $\mathbb{R}^d$ into a higher dimensional feature space $\mathbb{R}^s$ $(s > d)$. We perform the LDA (linear discrimination analysis) on the basis of the transformed sample $Z_i = \phi(X_i)$ and we classify a "new" x from input space on the basis of the transformed value $\phi(x)$. We will write the method in such a way that it appears as a function of the scalar products $\langle\phi(X_i), \phi(X_j)\rangle$ and $\langle\phi(x), \phi(X_i)\rangle$ (for the x to be classified), and in a second step we will replace these expressions by $K(X_i, X_j)$ and $K(x, X_i)$ respectively, for a positive definite kernel K.
Thus, to be able to kernelize any classification method, we need two facts.

1. The method depends on the (transformed) training set $Z_i = \phi(X_i)$ only via their Gram matrix
$$G_Z := (\langle Z_i, Z_j\rangle)_{i,j=1}^{n}$$
and the vector $Y = (Y_1, \ldots, Y_n)$ of recorded Y_i's.

2. Then classification of z is a given function $\langle z, \hat{a}\rangle$ for a certain $\hat{a}$.

Fisher's rule in terms of $Z_i = \phi(X_i)$ and $z = \phi(x)$ is as follows. Let

$$n_0 = \sum_{i:Y_i=-1} 1, \;\; n_1 = \sum_{i:Y_i=1} 1$$

be the size of the respective group $(n_0 + n_1 = n)$ and let

$$\hat{m}_0 = \frac{1}{n_0} \sum_{i:Y_i=-1} Z_i, \;\; \hat{m}_1 = \frac{1}{n_1} \sum_{i:Y_i=1} Z_i$$

be the two respective group means. Let the two scatter matrices be

$$S_0 = \sum_{i:Y_i=-1} (Z_i - \hat{m}_0)(Z_i - \hat{m}_0)^\top, \;\; S_1 = \sum_{i:Y_i=1} (Z_i - \hat{m}_1)(Z_i - \hat{m}_1)^\top$$

and define $S = S_0 + S_1$. Then Fishers's rule is: classify $z \in \mathbb{R}^s$ according to

$$g(z) = \operatorname{sgn}(\langle \hat{a}, z\rangle + a_0) \tag{26}$$

where

$$\begin{aligned} \hat{a} &= S^{-1}(\hat{m}_1 - \hat{m}_0), \\ \hat{a}_0 &= \frac{1}{2}\left(\hat{m}_0^\top S^{-1} \hat{m}_0 - \hat{m}_1^\top S^{-1} \hat{m}_1\right). \end{aligned}$$

(Indeed, we derived Fisher's rule from the normal Bayes rule (18) by substituting estimates $\hat{m}_i$, $\hat{\Sigma}$ for m_i and Σ. But $\hat{\Sigma} = S/(n-1)$, and multiplying $\hat{a}$ and a_0 in (26) by a scalar gives the same rule).
We will have to show that $\hat{a}$, $\hat{a}_0$ depend on the training set only via its Gram matrix G_Z.

5.1 Lemma *For any vector $m \in \mathbb{R}^s$ and any nonsingular $s \times s$ matrix S*

$$m^\top S^{-1} m = \max_{u \in \mathbb{R}^s} \frac{(\langle u, m\rangle)^2}{u^\top S u}.$$

and a maximizing vector is

$$u^* = S^{-1} m$$

or any scalar multiple of u^*.

Proof. For the maximization problem on the right, write $v = S^{1/2} u$, then the problem becomes

$$\max_{v \in \mathbb{R}^s} \frac{\left(v^\top S^{-1/2} m\right)^2}{v^\top v}$$

Now apply Cauchy-Schwartz:

$$\left(v^\top S^{-1/2}m\right)^2 \le \left(v^\top v\right)\left\|S^{-1/2}m\right\|^2$$

and equality is attained if and only if v is a scalar multiple of $S^{-1/2}m$. Hence

$$\frac{\left(v^\top S^{-1/2}m\right)^2}{v^\top v} \le \left\|S^{-1/2}m\right\|^2 = m^\top S^{-1}m$$

and equality is attained if and only if $v = S^{1/2}u = \lambda S^{-1/2}m$ for some λ, which means $u = \lambda S^{-1}m$. ■

For the kernelized version, note that we made the implicit assumption that S is nonsingular. That is fulfilled if the $Z_i = \phi(X_i)$ are not concentrated on a linear subspace in $\mathbb{R}^s$, i.e. under suitable assumptions on ϕ, such that ϕ is not a linear mapping, and if n is large enough. Then we can claim that $\hat{a}$ is a linear combination of the Z_i. Indeed in this case the vectors Z_i, $i = 1, \ldots, n$ span the whole of $\mathbb{R}^s$, and hence $\hat{a} \in \mathbb{R}^s$ is a linear combination

$$\hat{a} = \sum_{i=1}^{n} \alpha_i Z_i.$$

We will now first obtain $\hat{a}$ as the solution of a maximization as in the above lemma for $m = \hat{m}_1 - \hat{m}_0$, and then reformulate the problem as a maximization problem in terms of $\alpha = (\alpha_1, \ldots, \alpha_n)^\top$. Similarly we assume $\hat{a}_0$ given by

$$\hat{a}_0 = \frac{1}{2}\left(\max_{u \in \mathbb{R}^s} \frac{(\langle u, \hat{m}_0\rangle)^2}{u^\top S u} - \max_{u \in \mathbb{R}^s} \frac{(\langle u, \hat{m}_1\rangle)^2}{u^\top S u}\right) \tag{27}$$

and will represent this in terms of maximization problems for α. For the numerators in these problems, write $\hat{m}_0 = \sum c_i Z_i$ for certain c_i depending only on Y and $u = \sum \alpha_i Z_i$, then

$$\langle u, \hat{m}_0\rangle = \sum_{i.j} \alpha_i c_j \langle Z_i, Z_j\rangle$$

and hence the numerators are expressions of type

$$\left(\sum_{i.j} \alpha_i c_j \langle Z_i, Z_j\rangle\right)^2$$

where the sum ranges over certain sets of indices i, j and c_j are certain coefficients depending only on Y . Consider now the denominators, which are of form $a^\top S a$ where $a = \sum_{i=1}^n \alpha_i Z_i$. For any j, k we have

$$Z_j^\top S Z_k = \sum_{i:Y_i=-1} Z_j^\top (Z_i - \hat{m}_0)(Z_i - \hat{m}_0)^\top Z_k + \sum_{i:Y_i=1} Z_j^\top (Z_i - \hat{m}_1)(Z_i - \hat{m}_1)^\top Z_k.$$

We see that this term again is a function of the vector Y and the scalar products $\langle Z_i, Z_j\rangle$ and $\langle Z_i, Z_k\rangle$. In the same way it is argued that $\hat{a}_0$ given by (27) is a function of Y and the Gram matrix G_Z. Thus condition 1.) above for kernelization is satisfied. Condition 2.) is satisfied in view of the form of Fisher's rule (26).

Explicit expressions in terms of the Gram matrix. Define a $s \times n_0$ matrix $\mathbf{Z}_0$ the colums of which are the Z_i with $Y_i = -1$ and a $s \times n_1$ matrix $\mathbf{Z}_1$ the the colums of which are the Z_i with $Y_i = 1$. Let $\mathbf{1}_0$ be the n_0-vector of 1's and $\mathbf{1}_1$ be the n_1-vector of 1's. Then

$$\hat{m}_0 = n_0^{-1}\mathbf{Z}_0\mathbf{1}_0,\ \hat{m}_1 = n_1^{-1}\mathbf{Z}_1\mathbf{1}_1,$$

$$\begin{aligned} S_0 &= \sum_{i:Y_i=-1} (Z_i - \hat{m}_0)(Z_i - \hat{m}_0)^\top = \sum_{i:Y_i=-1} Z_i Z_i^\top - n_0 \hat{m}_0 \hat{m}_0^\top \\ &= \mathbf{Z}_0\mathbf{Z}_0^\top - \mathbf{Z}_0 n_0^{-1}\mathbf{1}_0\mathbf{1}_0^\top\mathbf{Z}_0^\top = \mathbf{Z}_0 P_0 \mathbf{Z}_0^\top \end{aligned}$$

where

$$P_0 = I_{n_0} - n_0^{-1}\mathbf{1}_0\mathbf{1}_0^\top.$$

Similarly

$$S_1 = \mathbf{Z}_1 P_1 \mathbf{Z}_1^\top,\ P_1 = I_{n_1} - n_1^{-1}\mathbf{1}_1\mathbf{1}_1^\top.$$

Consequently

$$S = \mathbf{Z}_0 P_0 \mathbf{Z}_0^\top + \mathbf{Z}_1 P_1 \mathbf{Z}_1^\top.$$

Let the total $s \times n$ matrix of the Z_i be

$$\mathbf{Z} = (\mathbf{Z}_0\ \mathbf{Z}_1)$$

Then the matrix of scalar products $\langle Z_i, Z_j\rangle$ is

$$\tilde{G}_Z = \mathbf{Z}^\top\mathbf{Z} = \begin{pmatrix} \mathbf{Z}_0^\top\mathbf{Z}_0 & \mathbf{Z}_0^\top\mathbf{Z}_1 \\ \mathbf{Z}_1^\top\mathbf{Z}_0 & \mathbf{Z}_1^\top\mathbf{Z}_1 \end{pmatrix}.$$

The matrix $\tilde{G}_Z$ represents just a permutation of the Gram matrix G_Z.
As argued above, the vectors Z_i, $i = 1, \ldots, n$ span the whole of $\mathbb{R}^s$. Hence we can find s of these vectors which already span $\mathbb{R}^s$. Write these vectors as columns in an $s \times s$ matrix and call this matrix $\mathbf{Z}_s$. Then $\mathbf{Z}_s$ is nonsingular and we write $a = \mathbf{Z}_s\alpha$ for some $\alpha = (\alpha_1, \ldots, \alpha_s)^\top$ Then the target function in the maximization problem for $\hat{a}$ (according to Lemma 5.1) can be written

$$\begin{aligned} \frac{\left(a^\top(\hat{m}_1 - \hat{m}_0)\right)^2}{a^\top S a} &= \frac{\left(\alpha^\top\mathbf{Z}_s^\top\left(n_1^{-1}\mathbf{Z}_1\mathbf{1}_1 - n_0^{-1}\mathbf{Z}_0\mathbf{1}_0\right)\right)^2}{\alpha^\top\mathbf{Z}_s^\top S\mathbf{Z}_s\alpha} \\ &= \frac{\left(\alpha^\top\left(n_1^{-1}\mathbf{Z}_s^\top\mathbf{Z}_1\mathbf{1}_1 - n_0^{-1}\mathbf{Z}_s^\top\mathbf{Z}_0\mathbf{1}_0\right)\right)^2}{\alpha^\top\left(\mathbf{Z}_s^\top\mathbf{Z}_0P_0\mathbf{Z}_0^\top\mathbf{Z}_s + \mathbf{Z}_s^\top\mathbf{Z}_1P_1\mathbf{Z}_1^\top\mathbf{Z}_s\right)\alpha} \end{aligned}$$

Since the matrix $\mathbf{Z}_s$ was formed using s selected columns of the matrix $\mathbf{Z}$, we see that the minimization problem in terms of α now involves only the elements of the matrix $\tilde{G}_Z$. Furthermore, since S was assumed nonsingular, the matrix $\mathbf{Z}_s^\top S\mathbf{Z}_s$ is then also nonsingular. This allows us to write for the solution $\hat{\alpha}$, using Lemma 5.1

$$\begin{aligned}\hat{\alpha} &= \mathbf{M}^{-1}\left(n_1^{-1}\mathbf{Z}_s^\top\mathbf{Z}_1\mathbf{1}_1 - n_0^{-1}\mathbf{Z}_s^\top\mathbf{Z}_0\mathbf{1}_0\right),\\ \mathbf{M} &= \mathbf{Z}_s^\top\mathbf{Z}_0 P_0\mathbf{Z}_0^\top\mathbf{Z}_s + \mathbf{Z}_s^\top\mathbf{Z}_1 P_1\mathbf{Z}_1^\top\mathbf{Z}_s.\end{aligned}$$

A similar expression can be found for $\hat{a}_0$, namely

$$\begin{aligned}\hat{a}_0 &= \frac{1}{2}\left(\hat{m}_0^\top S^{-1}\hat{m}_0 - \hat{m}_1^\top S^{-1}\hat{m}_1\right)\\ &= \frac{1}{2}\left(n_1^{-2}\mathbf{1}_1^\top\mathbf{Z}_1^\top\mathbf{Z}_s\mathbf{M}^{-1}\mathbf{Z}_s^\top\mathbf{Z}_1\mathbf{1}_1 - n_0^{-2}\mathbf{1}_0^\top\mathbf{Z}_0^\top\mathbf{Z}_s\mathbf{M}^{-1}\mathbf{Z}_s^\top\mathbf{Z}_0\mathbf{1}_0\right).\end{aligned}$$

5.2 Kernelization of the perceptron algorithm

Recall that the perceptron is a linear classifier

$$h(x) = \operatorname{sgn}\left(\langle w, x\rangle + b\right)$$

where w and b are found from the perceptron training algorithm. This algorithm starts with $w = \mathbf{0}$ and successively adds expressions $\eta y_i x_i$ to it, and similarly the algorithm starts with $b = 0$ and adds expressions $\eta y_i R^2$. So when the algorithm stops, the final w is of form $w = \sum_{i=1}^n \alpha_i\ \eta y_i x_i$ for certain coefficients α_i and the final b is of form $b = \sum_{i=1}^n \beta_i\ \eta y_i R^2$. We see that the number η influences only the scaling of (w, b) in each step, not the hyperplane and the decisions in each step (i.e. $\operatorname{sgn}\left(\langle w, x\rangle + b\right) = \operatorname{sgn}\left(\langle \eta w, x\rangle + \eta b\right)$), so that we may equivalently set $\eta = 1$. Furthermore, let us represent all the successive w's in the form $w = \sum_{i=1}^n \alpha_i\ \eta y_i x_i$ where coefficients α_i may change in each step. Then we can write the perceptron training algorithm in the following "dual" form. Write $\alpha = (\alpha_1, \dots, \alpha_n)$ for the vector of coefficients. Recall that $(x_1, \dots, x_n)$ denotes the training set.

Algorithm.

```
$\alpha \leftarrow \mathbf{0} \in \mathbb{R}^d$; $b \leftarrow 0$, $k \leftarrow 0$
$R \leftarrow \max_{1 \le i \le n} \|x_i\|$

repeat
    for $i = 1$ to $n$
        if $y_i \left(\sum_{j=1}^n \alpha_j \langle x_j, x_i \rangle + b\right) \le 0$ then
            $\alpha_i \leftarrow \alpha_i + 1$
            $b \leftarrow b + y_i R^2$

        endif
    endfor

until no mistakes made within the for loops
return $(\alpha, b)$ to define a linear classifier
```

$$h(x) = \operatorname{sgn}\left(\sum_{j=1}^n \alpha_j y_j \langle x_j, x \rangle + b\right).$$

We see immediately that the perceptron algorithm and the resulting linear classifier are now written in terms of scalar products $\langle x_i, x_j \rangle$ and $\langle x_i, x \rangle$, i.e. they are ready to be kernelized.

5.3 Examples of kernels

Commonly used kernels are for $x, y \in \mathbb{R}^d$

$$\text{polynomial: } K(x,y) = (\langle x, y \rangle + b)^r, \text{ where } b \ge 0,\ r = 1, 2, \ldots \tag{28}$$

$$\text{Gaussian: } K(x,y) = \exp\left(-\|x - y\|^2 / c\right), \text{ where } c > 0. \tag{29}$$

$$\text{sigmoid: } K(x,y) = \tanh\left(a \langle x, y \rangle + a_0\right), \tag{30}$$

where $\tanh(x)$ means the hyperbolic tangent of x,

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

A kernel satisfying the conditions of Mercer's theorem is called a *Mercer kernel.* The sigmoid kernel is known not to be a Mercer kernel (not positive definite), but it has been sucessfully used in practice.

5.2 Lemma *Let C be a ball in $\mathbb{R}^d$: for some $M > 0$*

$$C = \left\{ x \in \mathbb{R}^d : \|x\| \leq M \right\}.$$

The polynomial kernel is a Mercer kernel on C.

Proof. Assume $b > 0$; the proof for $b = 0$ is similar. Without loss of generality then assume $b = 1$. For $v = (v_1, \ldots, v_d) \in \mathbb{R}^d$ note that

$$\left(\sum_{i=1}^{d} v_i + 1 \right)^r = \sum_{s_1+\ldots+s_d+s_{d+1}=r} c_{s_1,\ldots,s_d} \prod_{i=1}^{d} v_i^{s_i} 1^{s_{d+1}} = \sum_{0 \leq s_1+\ldots+s_d \leq r} c_{s_1,\ldots,s_d} \prod_{i=1}^{d} v_i^{s_i}.$$

where all coefficients $c_{s_1,\ldots,s_d}$ are nonnegative. (As an example consider $d = 2$, $r = 2$; then

$$(v_1 + v_2 + 1)^2 = v_1^2 + v_1^2 + 2v_1v_2 + 2v_1 + 2v_2 + 1).$$

We introduce notation for a multi-index $s = (s_1, \ldots, s_d)$ with integer nonnegative components, where $|s| = \sum_{i=1}^{d} s_i$ and

$$v^s = \prod_{i=1}^{d} v_i^{s_i}.$$

Then we have

$$\left(\sum_{i=1}^{d} v_i + 1 \right)^r = \sum_{|s| \leq r} c_s v^s, \tag{31}$$

$$(\langle x, y \rangle + 1)^r = \left(\sum_{i=1}^{d} x_i y_i + 1 \right)^r = \sum_{|s| \leq r} c_s x^s y^s.$$

Let f be any function $f \in L_2(C)$; then

$$\begin{aligned} \int_{C \times C} K(x, y) f(x) f(y) dx dy &= \sum_{|s| \leq r} \int_{C \times C} c_s x^s y^s f(x) f(y) dx dy \\ &= \sum_{|s| \leq r} c_s \left(\int_C x^s f(x) dx \right)^2 \geq 0 \end{aligned}$$

since all c_s are nonnegative. ∎

Note that the result holds true for more general sets C, not only balls. In (31) we effectively showed that a possible map ϕ into feature space is $\phi(x) = \left(c_s^{1/2} x^s \right)_{|s| \leq r}$ which takes values in a high dimensional space $\mathbb{R}^k$ with $k = \# \{ s : |s| \leq r \}$.

5.3 Lemma *Let C be as in the previous lemma. The Gaussian kernel is a Mercer kernel on C.*

Proof. Let $N_d(\mu, \Sigma)$ be the d-dimensional normal distribution with expectation μ and covariance matrix Σ. It is well known that the characteristic function of $N_d(0, I_d)$ is

$$E \exp\left(i \langle t, Z \rangle\right) = \exp\left(-\|t\|^2/2\right).$$

It follows that for $\sigma > 0$ the characteristic function of $N_d(0, \sigma^2 I_d)$ is, if Z is standard d-variate normal

$$E \exp\left(i \langle t, \sigma Z \rangle\right) = \exp\left(-\sigma^2 \|t\|^2/2\right).$$

Let $\sigma^2 = 2/c$ and let f be any function $f \in L_2(C)$; then

$$\begin{aligned}\int_{C\times C} K(x,y) f(x) f(y) dx dy &= \int_{C\times C} \exp\left(-\sigma^2 \|x-y\|^2/2\right) f(x) f(y) dx dy \\ &= \int_{C\times C} E \exp\left(i \langle x-y, \sigma Z \rangle\right) f(x) f(y) dx dy\end{aligned}$$

$$= E\left(\int_C \exp\left(i \langle x, \sigma Z \rangle\right) f(x) dx\right)\left(\int_C \exp\left(-i \langle y, \sigma Z \rangle\right) f(y) dy\right) \tag{32}$$

Let the complex valued random variable ξ be defined by

$$\xi = \int_C \exp\left(i \langle x, \sigma Z \rangle\right) f(x) dx$$

and let $\bar{\xi}$ be its complex conjugate; then in view of $\overline{\exp(iu)} = \exp(-iu)$ we have as a consequence of (32)

$$\int_{C\times C} K(x,y) f(x) f(y) dx dy = E\xi\bar{\xi} = E|\xi|^2 \geq 0.$$

■

In order to understand what is the corresponding map in to feature space $\phi(x)$ here, we would have to know the series expansion of the kernel. However the theory of reproducing kernels (see below) provides an alternative to finding a map $\phi(x)$.

5.4 Reproducing kernels

Suppose we have a symmetric positive definite kernel $K(x, y)$ defined on $C \times C$ (where C is a compact subset of $\mathbb{R}^d$) which serves as the basis of kernelization of a linear classifier. Such a kernel is also called a Mercer kernel. We will now discuss another way of assigning a feature space to the kernel, i.e. represent the kernel as

$$K(x,y) = \langle \Phi(x), \Phi(y) \rangle_* \tag{33}$$

where $\Phi : C \mapsto \mathcal{H}$ where $\mathcal{H}$ is a linear space endowed with a scalar product $\langle \cdot, \cdot \rangle_*$.

Above we saw that in the case of a Mercer kernel, the map given by

$$\tilde{\Phi}(x) = \left(\lambda_j^{1/2}\phi_j(x)\right)_{j=1}^{\infty}$$

which maps C into the space of sequences l_2 (which are square summable) Then, if the scalar product in l_2 is given by the obvious infinite series and denoted $\langle\cdot,\cdot\rangle_*$ then

$$K(x,y) = \left\langle \tilde{\Phi}(x), \tilde{\Phi}(y) \right\rangle_* = \sum_{j=1}^{\infty} \lambda_j \phi_j(x)\phi_j(y).$$

anf for $\mathcal{H}$ =l_2 we have found a corresponding feature space. However now we want the result of the feature map $\tilde{\Phi}(x)$ to be a function rather than a sequence.
Define the feature map as

$$\Phi(x) = K(\cdot, x)$$

where the dot $\cdot$ in the first argument signifies that the value of a map is the function $z \to K(z,x)$.
In this connection we need to develop some theory.
Let $L_2(C)$ be the usual Hilbert space of square integrable functions on C, with scalar product

$$\langle f, g\rangle = \int_C f(x)g(x)dx.$$

and associated norm $\|f\| = \sqrt{\langle f, f\rangle}$. If $\{\phi_j\}_{j=1}^{\infty}$ is an orthonormal basis (ONB) of $L_2(C)$, consisting of functions ϕ_j then every $f \in L_2(C)$ has a series representation $f = \sum_{j=1}^{\infty} f_j\phi_j$ where $f_j = \langle f, \phi_j\rangle$.
Moreower, for any two $f, g \in L_2(C)$ we have

$$\langle f, g\rangle = \sum_{j=1}^{\infty} f_j g_j$$

(Parseval's theorem), in particular

$$\|f\|^2 = \langle f, f\rangle = \sum_{j=1}^{\infty} f_j^2.$$

Now, given the kernel K, define a set $\mathcal{H}$ as follows: if $\{\phi_j\}_{j=1}^{\infty}$ is the ONB appearing in the Mercer theorem for K and λ_j are the corresponding numbers there ($\lambda_j > 0$) then

$$\mathcal{H} = \left\{ f \in L_2(C) : \sum_{j=1}^{\infty} \frac{1}{\lambda_j} f_j^2 < \infty \right\}.$$

Since in general we have $\lambda_j \to 0$ as $j \to \infty$, not all functions in $L_2(C)$ will be in $\mathcal{H}$. In fact if $\sum_{j=1}^{\infty} f_j^2$ is finite then there is no guarantee that $\sum_{j=1}^{\infty} f_j^2/\lambda_j$ converges, precisely because $\lambda_j \to 0$ as $j \to \infty$.

Thus $\mathcal{H}$ is a subset of $L_2(C)$. On $\mathcal{H}$ define a scalar product

$$\langle f, g\rangle_* = \sum_{j=1}^{\infty} \frac{1}{\lambda_j} f_j g_j.$$

It is easy to see that this is defined for any $f, g \in \mathcal{H}$: by the Cauchy-Schwartz inequality in the series space l_2, for any $k > 0$ the tail expression of the above series satisfies

$$\left|\sum_{j=k}^{\infty} \frac{1}{\lambda_j} f_j g_j\right| = \left|\sum_{j=k}^{\infty} \left(\frac{f_j}{\lambda_j^{1/2}}\right)\left(\frac{g_j}{\lambda_j^{1/2}}\right)\right| \le \left(\sum_{j=k}^{\infty} f_j^2/\lambda_j\right)^{1/2} \left(\sum_{j=k}^{\infty} g_j^2/\lambda_j\right)^{1/2}$$

and the expressions on the right tend to zero as $k \to \infty$ by assumption, hence the series expression $\langle f, g\rangle_*$ exists and is finite. A little more (standard) reasoning shows that $\mathcal{H}$ becomes a Hilbert space under the scalar product $\langle \cdot, \cdot\rangle_*$. Thus, as a set, $\mathcal{H} \subset L_2(C)$ but $\mathcal{H}$ is a Hilbert space with a different scalar product (not the scalar product of $L_2(C)$, i.e. $\langle \cdot, \cdot\rangle_*$). The Hilbert space $\mathcal{H}$ constructed is called the **reproducing kernel Hilbert space (RKHS)** of K.
To justify that name, for any given $x \in C$ consider the function $y \to K(y, x)$, written as $K(\cdot, x)$. Since $K(\cdot, x)$ is a continuous function on C, it must be in $L_2(C)$. Moreover, we have by the Mercer expansion

$$K(\cdot, x) = \sum_{j=1}^{\infty} \lambda_j \phi_j(\cdot)\phi_j(x)$$

so the function $K(\cdot, x)$ has a series expansion in terms of the ONB $\{\phi_j\}_{j=1}^{\infty}$ with coefficients $\lambda_j\phi_j(x)$. Let us check whether $K(\cdot, x) \in \mathcal{H}$:

$$\sum_{j=1}^{\infty} \frac{1}{\lambda_j} \lambda_j^2 \phi_j^2(x) = \sum_{j=1}^{\infty} \lambda_j \phi_j(x)\phi_j(x) = K(x, x) < \infty$$

since K was a continuous function with finite values everywhere on $C \times C$. Hence $K(\cdot, x) \in \mathcal{H}$. Now for any $f \in \mathcal{H}$ we have

$$\langle f, K(\cdot, x)\rangle_* = \sum_{j=1}^{\infty} \frac{1}{\lambda_j} f_j \lambda_j \phi_j(x) = \sum_{j=1}^{\infty} f_j \phi_j(x) = f(x). \tag{34}$$

The last equality defines the **reproducing property of the kernel** K, using the scalar product $\langle \cdot, \cdot\rangle_*$: if we take the function $K(\cdot, x)$ and scalarproduct it with any $f \in \mathcal{H}$ then we reproduce the function f, more specifically the value $f(x)$.
Recall that we wanted to use the map $\Phi(x) = K(\cdot, x)$ (which to $x \in C$ assigns a function in $\mathcal{H}$) as a feature map, i.e. we wanted to achieve (33). To this end apply the reproducing equality (34) for $f = K(\cdot, y)$, given some $y \in C$. In this case $f(x) = K(x, y)$, so (34) becomes

$$\langle K(\cdot, y), K(\cdot, x)\rangle_* = K(x, y)$$

which we now can write

$$\langle \Phi(x), \Phi(y) \rangle_* = \langle \Phi(y), \Phi(x) \rangle_* = K(x, y). \tag{35}$$

Thus we have succeeded in representing $K(x, y)$ as a scalar product of the values of feature maps.

5.4 Remark For a given Mercer kernel K we have obtained a map $\Phi(x)$ into some feature space H (the RKHS of K) endowed with a scalar product $\langle \cdot, \cdot \rangle_*$ such that (35) holds. Recall that this has been achieved earlier in another way: suppose the expansion of the kernel is

$$K(x, y) = \sum_{j=1}^{\infty} \lambda_j \phi_j(x) \phi_j(y)$$

and consider the map $\phi(x) = \left(\lambda_j^{1/2} \phi_j(x) \right)_{j=1}^{\infty}$ with values in l_2. Let $\langle \cdot, \cdot \rangle_{l_2}$ be the scalar product in l_2, i.e. for sequences $u, v \in l_2$

$$\langle u, v \rangle_{l_2} = \sum_{j=1}^{\infty} u_j v_j.$$

Then as we have seen before

$$K(x, y) = \langle \phi(x), \phi(y) \rangle_{l_2}.$$

Now the two approaches are basically equivalent: consider the function $\Phi(x) = K(\cdot, x)$ and its series expansion $K(\cdot, x) = \sum_{j=1}^{\infty} \lambda_j \phi_j(\cdot) \phi_j(x)$. The functions $\tilde{\phi}_j = \lambda_j^{1/2} \phi_j(\cdot)$, $j = 1, 2, \ldots$are an orthonormal basis (ONB) wrt the scalar product $\langle \cdot, \cdot \rangle_*$, and

$$\Phi(x) = K(\cdot, x) = \sum_{j=1}^{\infty} \lambda_j^{1/2} \tilde{\phi}_j(\cdot) \phi_j(x),$$

thus $\lambda_j^{1/2} \phi_j(x)$ are the coefficients of $\Phi(x)$ in that basis. A function in a Hilbert space can equivalently represented by is coefficients wrt an ONB, thus we a have a one-to-one map

$$\Phi(x) \leftrightarrow \left(\lambda_j^{1/2} \phi_j(x) \right)_{j=1}^{\infty} = \phi(x)$$

and by Parseval's theorem for Hilbert spaces

$$\langle \Phi(x), \Phi(y) \rangle_* = \langle \phi(x), \phi(y) \rangle_{l_2}$$

thus $\Phi(x)$ and $\phi(x)$ basically represent the same map, up to an isomorphism.

5.5 Remark *There is another definition of positive definiteness which is more common. Let us call a symmetric kernel $K(x, y)$ defined on $C \subset \mathbb{R}^d$ positive definite* if for all n and every n-tuple $x_1, \ldots, x_n$, the matrix $\tilde{K} = (K(x_i, x_j))_{i,j=1}^{n}$ is positive definite, i.e. for every $y \in \mathbb{R}^n$ we* *have* $y^\top \tilde{K} y \geq 0$. *It can easily be seen that every Mercer kernel is positive definite*: if* $K(u,v) = \sum_{j=1}^{\infty} \lambda_j \phi_j(u) \phi_j(v)$ *then*

$$\begin{aligned} y^\top \tilde{K} y &= \sum_{j=1}^{\infty} \lambda_j \sum_{i,k=1}^{n} y_i y_k \phi_j(x_i) \phi_j(x_k) \\ &= \sum_{j=1}^{\infty} \lambda_j \left(\sum_{i=1}^{n} y_i \phi_j(x_i) \right)^2 \geq 0. \end{aligned}$$

Example: the kernel $\min(x,y)$

Suppose $C = [0,1]$ and consider the kernel

$$K(x,y) = \min(x,y).$$

In the learning context, this is a "toy example" since the x_i are assumed to be on $[0,1]$. First let us demonstrate that the kernel is positive definite in the sense used in Mercer's theorem, i.e. in the sense of (24). We have

$$\min(x,y) = \int_0^1 \mathbf{1}_{[0,x]}(t) \mathbf{1}_{[0,y]}(t) dt$$

and thus

$$\begin{aligned} \int_{C \times C} K(x,y) f(x) f(y) dx dy &= \int_0^1 \int_{C \times C} \mathbf{1}_{[0,x]}(t) \mathbf{1}_{[0,y]}(t) f(x) f(y) dx dy dt \\ &= \int_0^1 \left(\int_t^1 f(x) dx \right)^2 dt \geq 0. \end{aligned}$$

To find the Mercer expansion of $K(x,y)$, we first note:

5.6 Lemma *(i) The set of functions*

$$\phi_j(x) = \sqrt{2} \sin\left((j - 1/2) \pi x \right), \; j = 1, 2, \ldots$$

is a complete orthonormal basis of $L_2(0,1)$.
(ii) Likewise, the set of functions

$$\psi_j(x) = \sqrt{2} \cos\left((j - 1/2) \pi x \right), j = 1, 2, \ldots$$

is a complete orthonormal basis of $L_2(0,1)$.

To comment, recall the standard trigonometric orthonormal basis in $L_2(0,1)$, consisting of functions

$$\begin{aligned} f_0 &= 1, \\ f_j(x) &= \sqrt{2} \cos(2\pi x), f_{-j}(x) = \sqrt{2} \sin(2\pi x), \; j = 1, 2, \ldots \end{aligned}$$

The two bases considered in the lemma are derived from this; they use different arguments in $\sin(\cdot)$ and $\cos(\cdot)$.

5.7 Proposition *The Mercer expansion of $K(x,y) = \min(x,y)$ on $C = [0,1]$ is*

$$K(x,y) = \sum_{j=1}^{\infty} \lambda_j \phi_j(x)\phi_j(y)$$

for the functions ϕ_j given in Lemma 5.6 and

$$\lambda_j = \frac{1}{(j-1/2)^2 \pi^2},\ j = 1, 2, \ldots.$$

Proof. We show that the ϕ_j are eigenfunctions, i.e. fulfill

$$\int_C K(x,y)\phi_j(y)dy = \lambda_j \phi_j(x).$$

Write $\gamma_j = (j-1/2)\pi$ and note that $\phi_j(y) = \sqrt{2}\sin(\gamma_j y)$, $\lambda_j = \gamma_j^{-2}$. The left side above fulfills

$$\int_C \min(x,y)\phi_j(y)dy = \int_0^x y\phi_j(y)dy + \int_x^1 x\phi_j(y)dy$$

and using partial integration on the first integral on the right,

$$= \sqrt{2}\left[y\frac{1}{\gamma_j}(-\cos(\gamma_j y))\right]_0^x + \sqrt{2}\int_0^x \frac{1}{\gamma_j}\cos(\gamma_j y)\,dy + \sqrt{2}x\left[\frac{1}{\gamma_j}(-\cos(\gamma_j y))\right]_x^1$$

$$\begin{aligned}
&= -\sqrt{2}\frac{x}{\gamma_j}\cos(\gamma_j x) + \sqrt{2}\frac{0}{\gamma_j}\cos(0) + \sqrt{2}\int_0^x \frac{1}{\gamma_j}\cos(\gamma_j y)\,dy \\
&- \sqrt{2}\frac{x}{\gamma_j}\cos\gamma_j + \sqrt{2}\frac{x}{\gamma_j}\cos(\gamma_j x) \\
&= -\sqrt{2}\frac{x}{\gamma_j}\cos\gamma_j + \sqrt{2}\int_0^x \frac{1}{\gamma_j}\cos(\gamma_j y)\,dy
\end{aligned}$$

and since $\cos\gamma_j = \cos((j-1/2)\pi) = 0$, this equals

$$\begin{aligned}
&= 0 + \sqrt{2}\left[\frac{1}{\gamma_j^2}\sin(\gamma_j y)\right]_0^x = \frac{1}{\gamma_j^2}\sqrt{2}(\sin(\gamma_j x) - \sin(0)) \\
&= \lambda_j \phi_j(x).
\end{aligned}$$

In the Mercer theorem, the set of eigenfunctions is unique if all λ_j are different. Since ϕ_j used here are a complete ONB, there can be no further eigenfunctions. ∎

We can now identify the reproducing kernel Hilbert space associated to kernel K and the associated scalar product $\langle \cdot, \cdot \rangle_*$. Consider functions $f = \sum f_j \phi_j$ with the property

$$\langle f, f \rangle_* = \sum_{j=1}^{\infty} \frac{1}{\lambda_j} f_j^2 < \infty. \tag{36}$$

To understand this, assume that f has a finite expansion in the basis ϕ_j: $f = \sum_{j=1}^{n} f_j \phi_j$. Then

$$\begin{aligned} f'(x) &= \sqrt{2} \sum_{j=1}^{n} f_j \frac{d}{dx} \sin(\gamma_j y) = \sqrt{2} \sum_{j=1}^{n} f_j \gamma_j \cos(\gamma_j y) \\ &= \sum_{j=1}^{n} f_j \gamma_j \psi_j(x). \end{aligned}$$

as noted in Lemma 5.6 the set of functions $\psi_j(x) = \sqrt{2} \cos(\gamma_j \cdot)$, $j = 1, 2, \ldots$ is orthonormal on $[0, 1]$. Therefore

$$\int \left(f'(x)\right)^2 dx = \sum_{j=1}^{n} f_j^2 \gamma_j^2 = \sum_{j=1}^{n} \frac{1}{\lambda_j} f_j^2. \tag{37}$$

Moreover, all f with a finite expansion $f = \sum_{j=1}^{n} f_j \phi_j$ satisfy $f(0) = 0$. A limiting argument for $n \to \infty$ (standard in functional analysis) shows that the set $\mathcal{H}$ of functions f fulfilling (36) can be identified with the set of all $f \in L_2(0, 1)$ which are absolutely continuous, with (generalized) derivative f' satisfying $\int_0^1 \left(f'(x)\right)^2 dx < \infty$ and additionally satisfying $f(0) = 0$. An argument entirely analogous to (37) shows that the scalar product in $\mathcal{H}$ must be

$$\langle f, g \rangle_* = \int_0^1 f'(x) g'(x) dx. \tag{38}$$

Finally let us check the reproducing property of K: for $f \in \mathcal{H}$: since

$$K(t, x) = \min(t, x) = \int_0^t \mathbf{1}_{[0.x]}(u) du,$$

we have

$$\begin{aligned} \langle f, K(\cdot, x) \rangle_* &= \int_0^1 f'(t) \left(\frac{d}{dt} K(t, x) \right) dt = \int_0^1 f'(t) \mathbf{1}_{[0.x]}(t) dt \\ &= \int_0^x f'(t) dt = [f(t)]_0^x = f(x). \end{aligned}$$

Summary. This example exhibits all the basic features of the general RKHS theory. Starting with a Mercer kernel $K(\cdot, \cdot)$ on $C \times C$, one finds an associated subspace of functions in the space $L_2(C)$. This space can be endowed with a scalar product $\langle \cdot, \cdot \rangle_*$ under which it becomes a Hilbert space $\mathcal{H}$. The scalar product is not the scalar product of $L_2(C)$ and it is not defined for all functions in $L_2(C)$. In fact the functions in $\mathcal{H}$ are usually smooth functions and the scalar product $\langle \cdot, \cdot \rangle_*$ is a scalar product for smooth functions, operating on the derivatives.
The scalar product (38) is of simple form, but the scalar products for general kernels on $C \subset \mathbb{R}^d$ can be involved and may use various kinds of derivatives. In analysis, one also asks the reverse question: given a scalar product on smooth functions in $L_2(C)$, and associated Hilbert space $\mathcal{H}$, find a kernel K such that $\mathcal{H}$ becomes the RKHS of $\mathcal{H}$.

Learning with kernel $\min(x, y)$. Recall that with kernelization, we replace the training set (x_i, y_i) with $(K(\cdot, x_i), y_i)$, i.e. our x_i are replaced by functions $K(\cdot, x_i)$. Let us see how a training set which is not linearly separable on $[0, 1]$ becomes so after kernelization. Assume that $x_1 < x_2 < \ldots < x_n$ and the y_i alternate, i.e. $y_1 = 1$, $y_2 = -1$ and so on. Such a set is not not linearly separable on $[0, 1]$, since a linear rule is just a cut of the interval: all x_i on one side of an h_0 are classified as 1 and all x_i on the other side are classified as -1. If n is large, the error of any such rule approaches 1/2.
Now what is a linear rule in the RKHS $\mathcal{H}$? Let $\tilde{h} \in \mathcal{H}$; then a rule would be

$$h(x_i) = \operatorname{sgn} \left\langle \tilde{h}, K(\cdot, x_i) \right\rangle_*$$

(we take a hyperplane through the origin as an example here). By the reproducing property, we have

$$\left\langle \tilde{h}, K(\cdot, x_i) \right\rangle_* = \tilde{h}(x_i),$$

so for separation of the training set we need a function $\tilde{h} \in \mathcal{H}$ such that $\tilde{h}(x_i) = y_i$ for all i. This is certainly possible for any training set: a function $\tilde{h}$ which is differentiable with $\tilde{h}(0) = 0$ and $\tilde{h}(x_i) = y_i$, $i = 1, \ldots,$, i.e. an interpolating function. Such a function always exists in the space $\mathcal{H}$ (it can be taken to be a continuously differentiable function).
For a special configuration of the x_i, the basis functions ϕ_n are suitable: note that

$$\phi_n\left(\frac{1}{2n-1}\right) = \sin\left((2n-1)\frac{\pi}{2}\frac{1}{2n-1}\right) = 1, \phi_n\left(\frac{2}{2n-1}\right) = 0,$$
$$\phi_n\left(\frac{3}{2n-1}\right) = -1, \phi_n\left(\frac{4}{2n-1}\right) = 0, \phi_n\left(\frac{5}{2n-1}\right) = 1, \ldots$$

so that if $x_i = \frac{2i-1}{2n-1}$, $i = 1, \ldots, n$ then $\tilde{h} = \phi_n$ separates.

6 Interpolation and Regularization

In the last example we have seen that with a high dimensional feature space, such as the function space (RKHS) $\mathcal{H}$ figuring there, every training becomes linearly separable. So the perceptron training algorithm would separate the training set; Fisher's LDA in feature space would not necessarily do this, but still it would work on a linearly separable training set. But we also saw that a completely separating rule is somewhat nonintuitive: the space where the x_i live is split up into many small regions; essentially every x_i has its own classification region around it. Such rules could be seen as erratic; for instance if one point x_i from the -1 class appears completely surrounded by many points of the $+1$ class , we should not give credence to this single point and classify the whole neighboring region as "+1".
A method to deal with this problem is called *regularization.* It essentially consists of imposing a penalty term on rules such the above interpolating rule $\tilde{h}$ which penalizes "wiggliness". In the above example, the interpolating rule $\tilde{h}$ was a differentiable function with $\tilde{h}(0) = 0$, $\tilde{h}(x_i) = y_i$,

$i = 1, \ldots, n$. This means that $\tilde{h}$ might be very wiggly, i. e. $\int_0^1 \left(\tilde{h}'(t)\right)^2 dt = \left\|\tilde{h}'\right\|_2^2$ might be large. If we seek a compromise between approximation of the training set and smallness of the smoothness term $\left\|\tilde{h}'\right\|_2^2$, we migh arrive at a reasonable classifier.
Approximation of the training set by a rule h might be measured by the empical risk, i.e the proportion of misclassified points:

$$R_{emp}(h) = n^{-1} \sum_{i=1}^{n} \mathbf{1}_{\{h(x_i) \neq y_i\}}.$$

However it is common to use a somewhat modified expression, called *soft margin risk*. For that recall that the classification problem can be understood in such a way that our object is the regression function (if $Y \in \{0,1\}$)

$$r(x) = E\left(Y|X = x\right) = P\left(Y = 1|X = x\right) \tag{39}$$

and the Bayes decision is such that

$$h^*(x) = \begin{cases} 1 \text{ if } r(x) > 1/2 \\ 0 \text{ otherwise} \end{cases} \tag{40}$$

So an classifier can be obtained by first estimating the regression function r by an estimator $\hat{r}$ and then deriving the classification rule from it according to (40). Presently we are assuming $Y \in \{-1, 1\}$; then again $r(x) = E\left(Y|X = x\right)$ and the classification rule derived from an $\hat{r}$ is

$$\hat{h}(x) = \operatorname{sgn} \hat{r}(x).$$

Recall that for a linear rule in the RKHS we had above

$$h(x) = \operatorname{sgn} \left\langle \tilde{h}, K\left(\cdot, x\right) \right\rangle_* . \tag{41}$$

or a more simple linear rule in the original space, if $x \in \mathbb{R}^d$, is

$$h(x) = \operatorname{sgn}\left(\langle a, x_i \rangle + a_0\right). \tag{42}$$

This suggests to regard $\langle a, x_i \rangle + a_0$ or $\left\langle \tilde{h}, K\left(\cdot, x\right) \right\rangle_* = \tilde{h}(x)$ as estimators of the regression function $r(x)$. We then use the following modification of the empirical risk: define

$$l_{sm}(y, r(x)) = \max\left(0, 1 - yr(x)\right) = \begin{cases} 0 \text{ if } yr(x) > 1 \\ 1 - yr(x) \text{ otherwise} \end{cases} . \tag{43}$$

The rationale is the following. If $yr(x_i) > 0$ then x_i is correctly classified, if $yr(x_i) < 0$ then there is an error. We regard $|r(x_i)|$ as the "strength" of "confidence" with which the sign of $r(x_i)$ is predicted. If there is a wrong classification of x_i, then the above loss is $1 + |r(x_i)|$, i.e. the loss increases with the expressed "confidence" for that wrong prediction. If there is correct

classification, $yr(x_i) > 0$, then the loss is set 0 only if $|r(x_i)| > 1$, i.e. for a high enough confidence. If $0 < |r(x_i)| < 1$ then there is still some penalty, even though x_i is correctly classified.
For a simple linear (hyperplane) rule (42), the expression $|\langle a, x_i\rangle + a_0|$ was seen to be the distance of x_i to the hyperplane $\langle a, x\rangle + a_0 = 0$. In this case the soft margin loss (43) penalized wrong classification, with increasing distance from the hyperplane, and it penalizes also correct classification, to a lesser extent, if the distance to the hyperplane is less than 1.
Consider any linear classifier (41) in a feature space which is an RKHS, and define the empirical soft margin risk

$$R^{sm}_{emp}(\tilde{h}) = n^{-1}\sum_{i=1}^{n} l_{sm}(y_i, \tilde{h}(x_i)).$$

Rather than achieving 0 by interpolation ($\tilde{h}(x_i) = y_i$) we now impose a roughness penalty term and try to minimize the *regularized risk*:

$$R_{reg}(\tilde{h}) = R^{sm}_{emp}(\tilde{h}) + \lambda \left\|\tilde{h}\right\|_*^2 \tag{44}$$

where $\|\cdot\|_*^2$ is the squared norm in the RKHS. Recall that in our example $\left\|\tilde{h}\right\|_*^2 = \left\|\tilde{h}'\right\|_2^2$, so the term penalizes indeed "wiggliness" or complexity of the function $\tilde{h}$. The number λ can be chosen: it expresses the weight of the penalty term relative to sample approximation. For $\lambda \to 0$, since we equivalently minimize $\lambda^{-1}R^{sm}_{emp}(\tilde{h}) + \left\|\tilde{h}\right\|_*^2$, in the limit we will enforce $R^{sm}_{emp}(\tilde{h}) = 0$, i.e. interpolation, while still minimizing $\left\|\tilde{h}\right\|_*^2$. This will amount to something like "minimum norm interpolation".

6.1 Theorem *(Representer Theorem). Each minimizer $\tilde{h}$ of the regularized risk functional (44) admits a representation of the form*

$$\tilde{h}(x) = \sum_{i=1}^{n} \alpha_i K(x, x_i).$$

where $\alpha_i \in \mathbb{R}$.

Proof. We may decompose any $f \in \mathcal{H}$ into a part contained in the span of the kernel functions $K(\cdot, x_1), \ldots, K(\cdot, x_n)$ and one in the orthogonal complement:

$$\tilde{h}(x) = h_{\|}(x) + h_{\perp}(x) = \sum_{i=1}^{n} \alpha_i K(x, x_i) + h_{\perp}(x).$$

Here $h_\perp \in \mathcal{H}$ and $\langle h_\perp, K(\cdot, x_i)\rangle_* = 0$ for all $i = 1, \ldots, n$. We also have

$$\tilde{h}(x_j) = \left\langle \tilde{h}, K(\cdot, x_j)\right\rangle_* = \sum_{i=1}^{n} \alpha_i K(x_i, x_j) + \langle h_\perp, K(\cdot, x_i)\rangle_* \tag{45}$$

$$= \sum_{i=1}^{n} \alpha_i K(x_i, x_j). \tag{46}$$

Furthermore, for all $h_\perp$

$$\left\|\tilde{h}\right\|_*^2 = \left\|\sum_{i=1}^n \alpha_i K(\cdot, x_i)\right\|_*^2 + \|h_\perp\|_*^2 \geq \left\|\sum_{i=1}^n \alpha_i K(\cdot, x_i)\right\|_*^2. \tag{47}$$

We may now minimize $R_{reg}(\tilde{h})$ in two steps: first fix α_i, $i = 1, \ldots, n$ and minimize over $h_\perp$, and then minimize over α_i. In the first step, in view of (46), all $\tilde{h}(x_j)$ are fixed too, hence the term $R_{emp}^{sm}(\tilde{h})$ is also fixed. Thus minimization in the first step concerns only the term $\left\|\tilde{h}\right\|_*^2$. From (47) we see that the optimal choice in the first step is $h_\perp = 0$. Thus the overall solution is always of the form $\tilde{h}(x) = h_\|(x) = \sum_{i=1}^n \alpha_i K(\cdot, x_i)$. ∎

The importance of this theorem lies in the fact that the infinite-dimensional minimization problem over $\tilde{h} \in \mathcal{H}$ is reduced to a finite dimensional one, over the linear span of $K(\cdot, x_1)$,..., $K(\cdot, x_n)$. Let us examine how these functions look like in our example. We have

$$K(t, x_i) = \min(t, x_i) = \begin{cases} t \text{ if } t \leq x_i \\ x_i \text{ otherwise} \end{cases}.$$

Such a function is a special *piecewiese linear spline on* $[0,1]$. It is linear for $t \leq x_i$ and constant for $t > x_i$ and it is continuous in the point x_i.

In general, a spline on Ω is a function which is polynomial on certain regions of Ω and the polynomials are "joined" in the borders of the regions such that the overall function has some continuity and smoothness properties.

6.1 Linear estimation of the regression function

Question: *Above it was claimed that if we use a simple linear discrimination rule in the original space ($x \in \mathbb{R}^d$)*

$$h(x) = \operatorname{sgn}(\langle a, x\rangle + a_0)$$

we can interpret the procedure as follows: the expression $\langle a, x\rangle + a_0$ is an estimator of the regression function $r(x) = E(Y|X = x)$ for $Y \in \{-1, 1\}$, and we then mimic the Bayes rule $h^*(x) = \operatorname{sgn}(r(x))$. However for $Y \in \{-1, 1\}$ the regression function takes values $|r(x)| < 1$, but $\langle a, x\rangle + a_0$ is linear in x and hence unbounded in value. How does this fit together ?

Answer. The expression $\langle a, x\rangle + a_0$ can be seen as a linear approximation to $r(x)$ in the following sense. Consider the case where the class-conditional densities f_i are both normal, i.e. the densities of $N_d(m_i, \Sigma)$, $i = 0, 1$. This case was treated in detail in the section on Fisher's LDA. For simplicity let us assume $\Sigma = I_d$. We now use a class indicator $Y \in \{-1, 1\}$ but we keep notation f_0, f_1 for the class-conditional densities. There is a one-to-one correspondence to the previously used class indicator 0 or 1 ($\tilde{Y} \in \{0, 1\}$, say) given by $Y = 2\tilde{Y} - 1$ and $\tilde{Y} = (Y + 1)/2$. We have

$$f_i(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\|x - m_i\|^2\right).$$

The Bayes rule was shown to be $h^*(x) = \operatorname{sgn}(f_1(x) - f_0(x))$ (recall we now formally decide between -1 and 1, not between 0 and 1) or equivalently

$$h^*(x) = \operatorname{sgn}(\langle a, x\rangle + a_0) \text{ for } a = m_1 - m_0,\ a_0 = \frac{1}{2}\left(\|m_0\|^2 - \|m_1\|^2\right) \tag{48}$$

(cf. (18). At the same time the Bayes rule can be stated in terms of the regression function $r(x) = E(Y|X = x)$ as $h^*(x) = \operatorname{sgn}(r(x))$. Indeed, when we used a class indicator $\tilde{Y} \in \{0, 1\}$ the regression function was

$$\tilde{r}(x) = E\left(\tilde{Y}|X = x\right) = (r(x) + 1)/2,$$

so $r(x) \geq 0$ if and only if $\tilde{r}(x) \geq 1/2$. Let us compute $r(x)$ for uniform prior probabilities ($P(Y = 1) = P(Y = -1) = 1/2$).

$$\begin{aligned} r(x) &= E(Y|X = x) = 1 \cdot P(Y = 1|X = x) + (-1)\ P(Y = -1|X = x) \\ &= \frac{\frac{1}{2}f_1(x)}{\frac{1}{2}f_0(x) + \frac{1}{2}f_1(x)} - \frac{\frac{1}{2}f_0(x)}{\frac{1}{2}f_0(x) + \frac{1}{2}f_1(x)} \\ &= \frac{f_1(x) - f_0(x)}{f_1(x) + f_0(x)}. \end{aligned}$$

By using the fact

$$f_i(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\|x\|^2\right) \exp\left(\langle m_i, x\rangle - \frac{1}{2}\|m_i\|^2\right),\ i = 0, 1$$

we obtain

$$\begin{aligned} r(x) &= \frac{\exp\left(\langle m_1, x\rangle - \frac{1}{2}\|m_1\|^2\right) - \exp\left(\langle m_0, x\rangle - \frac{1}{2}\|m_0\|^2\right)}{\exp\left(\langle m_1, x\rangle - \frac{1}{2}\|m_1\|^2\right) + \exp\left(\langle m_0, x\rangle - \frac{1}{2}\|m_0\|^2\right)} \\ &= \frac{\exp(z_1) - \exp(z_0)}{\exp(z_1) + \exp(z_0)}, \end{aligned}$$

where

$$z_i = \langle m_i, x\rangle - \frac{1}{2}\|m_i\|^2,\ i = 0, 1.$$

The z_i are linear functionals of x and $r(x)$ depends on x only via those linear functionals. Let us make a further linear transformation

$$u_1 = z_1 - z_0,\ u_2 = z_1 + z_0.$$

Then

$$z_0 = \frac{u_2 - u_1}{2},\ z_1 = \frac{u_2 + u_1}{2}$$

and

$$r(x) = \frac{\exp\left(\frac{u_2+u_1}{2}\right) - \exp\left(\frac{u_2-u_1}{2}\right)}{\exp\left(\frac{u_2+u_1}{2}\right) + \exp\left(\frac{u_2-u_1}{2}\right)} = \frac{\exp(u_1/2) - \exp(-u_1/2)}{\exp(u_1/2) + \exp(-u_1/2)}$$
$$= \tanh(u_1/2).$$

Here u_1 is the linear functional of x

$$u_1 = z_1 - z_0 = \langle a, x \rangle + a_0$$

where a, a_0 are given by (48). Thus

$$r(x) = \tanh\left(\frac{1}{2}\left(\langle a, x \rangle + a_0\right)\right). \tag{49}$$

The function $\tanh(x)$ is antisymmetric ($\tanh(x) = -\tanh(-x)$) and smooth. Therefore $\tanh(0) = 0$ and also $\tanh'(0) = 1$. Consider a Taylor expansion of the function $r(x)$, $x \in \mathbb{R}^d$ around any point x_0 on the decision boundary. i.e. $r(x_0) = 0$. The last relation is equivalent to $\langle a, x_0 \rangle + a_0 = 0$, i.e. x_0 is on the hyperplane given by a, a_0. Setting $z = x - x_0$ and letting Δ_{x_0} be the gradient of r at x_0 we obtain

$$r(x) = r(x_0 + z) = r(x_0) + \Delta_{x_0}^\top z + o(1) \text{ as } \|z\| \to 0 \tag{50}$$

and we find by the chain rule

$$\Delta_{x_0} = \tanh'(\frac{1}{2}\left(\langle a, x_0 \rangle + a_0\right))\frac{1}{2}a = \frac{1}{2}a.$$

Note that $\langle a, x_0 \rangle + a_0 = 0$ implies that $a_0 = -\langle a, x_0 \rangle$ does not depend on x_0, so we can write

$$r(x) = \Delta_{x_0}^\top (x - x_0) + o(1) = \frac{1}{2}\langle a, x \rangle + \frac{1}{2}a_0 + o(1) \text{ as } \|x - x_0\| \to 0. \tag{51}$$

We see that we can approximate the regression function in the normal case by *the same linear function* $\frac{1}{2}\left(\langle a, x \rangle + a_0\right)$ in every point x_0 on the decision boundary. In contrast, if r is an arbitrary regression function for the joint distribution (X, Y), assumed smooth, then we can approximate it near a point x_0 on the decision boundary, .according to (50) by a linear function $\Delta_{x_0}^\top x - \Delta_{x_0}^\top x_0$ which depends on x_0 in general. Then the decision boundary near x_0 is approximately a hyperplane $\left\{x : \Delta_{x_0}^\top x - \Delta_{x_0}^\top x_0 = 0\right\}$ which depends on x_0 in general.

$$\Delta_{x_0}^\top x - \Delta_{x_0}^\top x_0$$

In that sense, under the present normality assumption, the regression is approximately linear near the decision boundary.
Recall that if $\|a\| = 1$ then $|\langle a, x \rangle + a_0|$ is the distance of x from the hyperplane $\langle a, x \rangle + a_0 = 0$. In general, $|\langle a, x \rangle + a_0|$ is proportional to that distance (with factor $\|a\|$). Then, using only a Taylor expansion of $\tanh(x)$ at $x = 0$ we find from (49)

$$r(x) = \frac{1}{2}\left(\langle a, x \rangle + a_0\right) + o(1) \text{ as } |\langle a, x \rangle + a_0| \to 0$$

and we have obtained (51) again, but as an approximation directly in terms of the distance to the hyperplane.
In that sense, we can say that $\langle a, x\rangle + a_0$ approximates the regression function, up to a constant factor, for small values of the distance of x from the optimal hyperplane, in the Gaussian case with equal class-conditional covariance matrices $\Sigma = I_d$.

6.2 More on spline smoothing

So far we have considered the regularized risk (44) where the first term is the empirical soft margin risk $R^{sm}_{emp}(\tilde{h})$ and the second term is the regularization term or smoothness penalty $\lambda \left\|\tilde{h}\right\|_*^2$. In the representer theorem (Theor. 6.1) the minimization was over $\tilde{h} \in \mathcal{H}$. In our first example the RKHS $\mathcal{H}$ was

$$H_0^1(0,1) := \left\{f \in L_2(0,1) : f \text{ abs. continuous}, f(0) = 0, f' \in L_2(0,1)\right\}$$

with norm $\|f\|_*^2 = \|f'\|_2^2$. The decision rule was

$$h(x_i) = \operatorname{sgn}\left\langle \tilde{h}, K(\cdot, x_i)\right\rangle_* \tag{52}$$

where by the reproducing property, we have $\left\langle \tilde{h}, K(\cdot, x_i)\right\rangle_* = \tilde{h}(x_i)$, so in fact our rule is $h(x_i) = \operatorname{sgn}\tilde{h}(x_i)$. However the general form of a linear rule is $h(x_i) = \operatorname{sgn}\left(\langle a, x_i\rangle + a_0\right)$ where a constant a_0 is present. We might wish to include a constant a_0 in the linear rule in RKHS (52), i.e. to consider

$$h(x_i) = \operatorname{sgn}\left(\left\langle \tilde{h}, K(\cdot, x_i)\right\rangle_* + a_0\right) = \operatorname{sgn}\left(\tilde{h}(x_i) + a_0\right)$$

and then consider minimization of the regularized empirical risk functional. We then define $f = \tilde{h} + a$, $\tilde{h} \in \mathcal{H}$, $a \in \mathbb{R}$ and define

$$R^{sm}_{emp}(f) = n^{-1}\sum_{i=1}^{n} l_{sm}(y_i, f(x_i)).$$

and the regularized version by adding $\lambda \left\|\tilde{h}\right\|_*^2$. Note that in our example the expression $\|f'\|_2^2$ is defined since $f' = \tilde{h}'$.
A second extension we will consider is regression estimation, separate from classification but somewhat similar in methodology. We have i.i.d. observations (X_i, Y_i), $i = 1, \ldots, n$ where $X_i \in \mathbb{R}^d$, $Y_i \in \mathbb{R}$ and the purpose is to estimate $r(x) = E\left(Y|X = x\right)$. The empiricial risk is then just the squared risk:

$$R^{squ}_{emp}(f) = n^{-1}\sum_{i=1}^{n}(y_i - f(x_i))^2.$$

where $f = \tilde{h} + g$, $\tilde{h} \in \mathcal{H}$ (some RKHS) and $g \in \mathcal{G}$, where $\mathcal{G}$ is a *unisolvent set of functions* for the given $\{x_1, \ldots, x_n\}$:

$$\{g_1, g_2 \in \mathcal{G},\, g_1(x_i) = g_2(x_i),\, i = 1, \ldots, n\} \Longrightarrow g_1 = g_2.$$

For instance, if $\mathcal{G}$ is the set of constants then the values on one x_i uniquely determine $g \in \mathcal{G}$, if $\mathcal{G}$ is the set of linear functions then the values on two distinct x_i uniquely determine $g \in \mathcal{G}$ and so forth. Our primary example below will be linear functions and an RKHS $\mathcal{H}$ where the norm is given by the second derivative.

6.2 Theorem *(Generalized Representer Theorem). For functions $f = \tilde{h} + g$, $\tilde{h} \in \mathcal{H}$ (RKHS) and $g \in \mathcal{G}$ (unisolvent), consider a regularized risk functional*

$$R_{reg}(f) = R_{emp}(f) + \lambda \left\| \tilde{h} \right\|_*^2$$

where $R_{emp}(f)$ is either $R_{emp}^{sm}(f)$ or $R_{emp}^{squ}(f)$. Each minimizer f of $R_{reg}(f)$ admits a representation of the form

$$f(x) = \sum_{i=1}^{n} \alpha_i K\left(x, x_i\right) + g.$$

where $\alpha_i \in \mathbb{R}$ and $g \in \mathcal{G}$.

Proof. We may decompose any $\tilde{h} \in \mathcal{H}$ into a part contained in the span of the kernel functions $K\left(\cdot, x_1\right)$,..., $K\left(\cdot, x_n\right)$ and one in the orthogonal complement:

$$\tilde{h}(x) = h_{\|}(x) + h_{\perp}(x) = \sum_{i=1}^{n} \alpha_i K\left(x, x_i\right) + h_{\perp}(x).$$

Here $h_{\perp} \in \mathcal{H}$ and $\left\langle h_{\perp}, K\left(\cdot, x_i\right)\right\rangle_* = 0$ for all $i = 1, \ldots, n$. We also have

$$\tilde{h}(x_j) = \left\langle \tilde{h}, K\left(\cdot, x_j\right)\right\rangle_* = \sum_{i=1}^{n} \alpha_i K\left(x_i, x_j\right) + \left\langle h_{\perp}, K\left(\cdot, x_i\right)\right\rangle_* \tag{53}$$

$$= \sum_{i=1}^{n} \alpha_i K\left(x_i, x_j\right). \tag{54}$$

hence

$$f(x_j) = \tilde{h}(x_j) + g(x_j) = \sum_{i=1}^{n} \alpha_i K\left(x_i, x_j\right) + g(x_j) \tag{55}$$

Furthermore, for all $h_{\perp}$

$$\left\| \tilde{h} \right\|_*^2 = \left\| \sum_{i=1}^{n} \alpha_i K\left(\cdot, x_i\right) \right\|_*^2 + \left\| h_{\perp} \right\|_*^2 \geq \left\| \sum_{i=1}^{n} \alpha_i K\left(\cdot, x_i\right) \right\|_*^2. \tag{56}$$

We now may minimize $R_{reg}(f)$ in two steps: first fix α_i, $i = 1, \ldots, n$ and $g \in \mathcal{G}$, and minimize over $h_{\perp}$, and then minimize over α_i and g. In the first step, in view of (55), all $\tilde{h}(x_j)$ are fixed too, hence the term $R_{emp}^{sm}(\tilde{h})$ is also fixed. Thus minimization in the first step concerns only the term $\left\| \tilde{h} \right\|_*^2$.

From (47) we see that the optimal choice in the first step is $h_\perp = 0$. Thus the overall solution is always of the form claimed. ■

Remark. The theorem is true without the unisolvency condition on $\mathcal{G}$, but this one is useful for other purposes such as uniqueness of the solution.

Example 1: linear smoothing spline, special

The RKHS $\mathcal{H}$ is $H_0^1(0,1)$ with norm $\|f\|_*^2 = \|f'\|_2^2$, $\mathcal{G}$ is empty, $R_{emp}(f)$ is $R_{emp}^{squ}(f\ )$. We are in the general regression situation and minimizing

$$n^{-1}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda \left\|f'\right\|_2^2 \tag{57}$$

over $f \in H_0^1(0,1)$. We assume the points x_i ordered and all distinct: $0 < x_1 < \ldots < x_n < 1$. By the representer theorem, the result is a piecewise linear spline f with $f(0) = 0$ and $f(1) = f(x_n)$ (since any linear combination $\sum_{i=1}^n \alpha_i K(x_i, \cdot)$ is of this form). Write

$$y = (y_1, \ldots, y_n)^\top,\ a = (\alpha_1, \ldots, \alpha_n)^\top,\ K = (K(x_i, x_j))_{i,j=1}^n$$

According to Remark 5.5, the kernel $K(t,u)$ on $[0,1]\times[0,1]$ is positive definite*, which implies that the symmetric matrix K is nonnegative definite (meaning $c^\top Kc \geq 0$ for every $c \in \mathbb{R}^n$). Furthermore we have

6.3 Lemma *The matrix K is nonsingular.*

The proof is left as an exercise.
A symmetric matrix which is nonnegative definite and singular is *positive definite* (meaning meaning $c^\top Kc > 0$ for every $c \in \mathbb{R}^n$, $c \neq \mathbf{0}_n$)[1].

[1]For matrices and kernels we use a slightly different language here: a kernel was called positive definite (or p.d.*) if the relevant inequality for a function f involves " ≥ 0" whereas symmetric matrices with an inequality " ≥ 0"are called nonnegative definite. For symmetric matrices "positive definite" means the strict inequality " > 0" holds for any nonzero vectors.

Now our target function in the minimization problem (57) is (with $\|\cdot\|$ the euclidean norm)

$$\begin{aligned}
&n^{-1}\|y-Ka\|^2+\lambda\left\|\sum_{i=1}^{n}\alpha_i K(x_i,\cdot)\right\|_*^2\\
&=n^{-1}\|y-Ka\|^2+\lambda\sum_{i,j=1}^{n}\alpha_i\alpha_j\left\langle K(x_i,\cdot),K(x_j,\cdot)\right\rangle_*\\
&=n^{-1}\|y-Ka\|^2+\lambda\sum_{i,j=1}^{n}\alpha_i\alpha_j K(x_i,x_j)\\
&=n^{-1}\|y-Ka\|^2+\lambda a^\top K a.
\end{aligned}$$

We set $\hat{f}=Ka$ where $\hat{f}=(f(x_1),\ldots,f(x_n))^\top$; then our target function is

$$n^{-1}\left\|y-\hat{f}\right\|^2+\lambda\hat{f}^\top K^{-1}\hat{f} \tag{58}$$

To minimize this quadratic form, we differentiate and get a normal equation (using $K^\top=K$)

$$-2n^{-1}\left(y-\hat{f}\right)+2\lambda K^{-1}\hat{f}=0$$

with solution

$$\begin{aligned}
\left(n^{-1}I_n+\lambda K^{-1}\right)\hat{f}&=n^{-1}y,\\
\hat{f}=\hat{D}y \text{ for } \hat{D}&=\left(I_n+\lambda nK^{-1}\right)^{-1}
\end{aligned} \tag{59}$$

Shrinkage property of the solution. From the last equation we see that the solution, i.e. the linear smooting spline, is the result of a linear operator applied to the data y (if λ is chosen a priori, i.e. not depending on y). This operator is given by the matrix $\hat{D}=\left(I_n+\lambda nK^{-1}\right)^{-1}$, also called a *smoother matrix* (or *hat matrix*). For $\lambda=0$ we obtain $\hat{f}=y$, i.e. the linear spline which interpolates the data. For $\lambda>0$ the matrix λnK^{-1} is positive definite. Thus

$$I_n+\lambda nK^{-1}\geq I_n \tag{60}$$

(in the sense of the order of symmetric matrices, induced by positive definiteness: $A\geq B$ if $A-B$ is nonnegative definite, i.e. if $a^\top(A-B)a\geq 0$ for all $a\in\mathbb{R}^n$). From (60) we obtain by inverting

$$\hat{D}=\left(I_n+\lambda nK^{-1}\right)^{-1}\leq I_n$$

which means that the "hat matrix" $\hat{D}$ is smaller than the unit matrix, i.e. the data vector y is "shrunk" towards the origin by $\hat{f}=\hat{D}y$.

Smoothing property of the solution. Consider spectral decomposition of K

$$K = \sum_{j=1}^{n} \gamma_{j,n} k_j k_j^\top$$

where k_j, $j = 1, \ldots, n$ is an orthonormal set of eigenvectors in $\mathbb{R}^n$ ($\|k_j\| = 1$ for the euclidean norm) and $\gamma_{j,n} > 0$, $j = 1, \ldots, n$ are the eigenvalues. (Here of course the k_j also depend on n, but we emphasise this dependence on n in the eigenvalues). Then it is a well known fact that

$$K^{-1} = \sum_{j=1}^{n} \gamma_{j,n}^{-1} k_j k_j^\top$$

and the hat matrix is

$$\hat{D} = \left(I_n + \lambda n K^{-1}\right)^{-1} = \sum_{j=1}^{n} \frac{1}{1 + \lambda n \gamma_{j,n}^{-1}} k_j k_j^\top$$

Thus the operation $\hat{f} = \hat{D} y$ may be interpreted as follows: get new coordinates of the data y by taking $k_j^\top y$, multiply each coordinate by $\left(1 + \lambda n \gamma_{j,n}^{-1}\right)^{-1}$, change back to old coodinates. Since $\left(1 + \lambda n \gamma_{j,n}^{-1}\right)^{-1} < 1$ (if $\lambda > 0$), we again see the shrinkage character.
Now consider the Mercer decomposition of the kernel $K(x, y) = \min(x, y)$ given by Proposition 5.7: it is

$$K(x, y) = \sum_{j=1}^{\infty} \gamma_j \phi_j(x) \phi_j(y)$$

for the functions $\phi_j(x) = \sqrt{2} \sin\left((j - 1/2)\pi x\right)$, $j = 1, 2, \ldots$ given in Lemma 5.6 and

$$\gamma_j = \frac{1}{(j - 1/2)^2 \pi^2}, \; j = 1, 2, \ldots.$$

6.4 Claim *For the matrix $K = (K(x_i, x_j))_{i,j=1}^{n}$ we can expect that for large n, the eigenvectors k_j are close to $\hat{\phi}_j$ where*

$$\hat{\phi}_j := n^{-1/2} \left(\phi_j(x_1), \ldots, \phi_j(x_n)\right)^\top$$

and the eigenvalues $\gamma_{j,n}$ are approximately $n\gamma_j$.

Reasoning. Indeed approximating first the kernel K by a truncated series

$$K(x, y) \approx \sum_{j=1}^{n} \gamma_j \phi_j(x) \phi_j(y)$$

and then noting for the matrix K

$$K = (K(x_i, x_j))_{i,j=1}^{n} \approx \left(\sum_{s=1}^{n} \gamma_s \phi_s(x_i) \phi_s(x_j)\right)_{i,j=1}^{n} = \sum_{s=1}^{n} n \gamma_s \hat{\phi}_s \hat{\phi}_s^\top$$

where the vectors $\hat{\phi}_s = (\phi_s(x_1), \ldots, \phi_s(x_n))^\top$ are approximately orthogonal if the values $x_1, \ldots, x_n$ are nearly uniformly spaced within $[0,1]$:

$$\left\langle \hat{\phi}_r, \hat{\phi}_s \right\rangle = \hat{\phi}_r^\top \hat{\phi}_s = n^{-1} \sum_{i=1}^n \phi_r(x_i)\phi_s(x_i)$$
$$\approx \int_0^1 \phi_r(x)\phi_s(x)dx = \begin{cases} 1 \text{ if } r = s \\ 0 \text{ otherwise} \end{cases} .$$

Thus $K = \sum_{s=1}^n n\gamma_s \hat{\phi}_s \hat{\phi}_s^\top$ is an approximate spectral decomposition of the matrix K. ■

We obtain an approximation for $\hat{D}$: since

$$n\gamma_{j,n}^{-1} \approx \gamma_j^{-1} = (j - 1/2)^2 \pi^2 \approx \pi^2 j^2$$

(for large n), we get

$$\hat{D} = \sum_{j=1}^n \frac{1}{1 + \lambda n \gamma_{j,n}^{-1}} k_j k_j^\top \approx \sum_{j=1}^n \frac{1}{1 + \lambda \pi^2 j^2} \hat{\phi}_j \hat{\phi}_j^\top$$

Here we may see the action of the smoothing operator on the data. The j-th Fourier coefficient, corresponding to basis function $\sqrt{2}\sin((j - 1/2)\pi x)$, of the smoothed function $\hat{f}$ is approximately

$$\left(\int_0^1 \varphi_j(x)\hat{f}(x)dx\right) \approx n^{-1/2}\hat{\phi}_j^\top \hat{f} = \frac{1}{1 + \lambda\pi^2 j^2} n^{-1/2} \hat{\phi}_j^\top y$$

where $n^{-1/2}\hat{\phi}_j^\top y$ is the approximate Fourier coefficient of the data, corresponding to the same frequency. We see that smoothing means multiplication of the coefficient by the multiplier $\left(1 + \lambda\pi^2 j^2\right)^{-1}$. The higher the frequency j, the smaller the multiplier; *this means "dampening" or "filtering" of high frequencies.* Thus the smoothing spline is close to a tapered orthogonal series estimator.
The map $j \to \left(1 + \lambda\pi^2 j^2\right)^{-1}$ may be called the filter characteristic of the smoothing spline. Thus the smoothing spline is close to a "tapered" orthogonal series estimator. A *truncated* orthogonal series estimator would have filter characteristic $\mathbf{1}_{[0,s]}(j)$where s is the truncation index.

Example 2: linear smoothing spline, general

The setup is as in example 1, but now $\mathcal{G}$ is the set of constant functions on $[0,1]$. Thus $f = \tilde{h} + g$ where $\tilde{h} \in H_0^1(0,1)$ and g is constant. Consider a version of the space $H_0^1(0,1)$ without boundary conditions:

$$H^1(0,1) := \left\{ f \in L_2(0,1) : f \text{ abs. continuous, }, f' \in L_2(0,1) \right\}.$$

Obviously any function $f \in H^1(0,1)$ has a unique decomposition $f = \tilde{h} + g$, $\tilde{h} \in H_0^1(0,1)$ and g a constant: determine g by $g(0) = f(0)$, then it is clear that $\tilde{h} := f - g \in H_0^1(0,1)$. Since $\tilde{h}' = f'$

we may equivalently consider minimization of (57) over functions $f \in H^1(0,1)$. According to the generalized representer theorem (Theorem 6.2) any solution can be written

$$f = \sum_{i=1}^{n} \alpha_i K(\cdot, x_i) + g$$

where $K(t, x_i) = \min(t, x_i)$. This is a piecewise linear spline f with $f(1) = f(x_n)$. Note that the condition $f(0) = 0$ is no longer present since we are adding a constant g, but it can be shown that now $f(0) = f(x_1)$. Indeed, any other linear spline f with the same values of $f(x_i)$ would have a higher value of $\|f'\|_2^2$.

6.5 Corollary *Any solution of the problem of minimizing*

$$n^{-1} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int_0^1 \left(f'(t)\right)^2 dt$$

over functions $f \in H^1(0,1)$, for given $x_i \in [0,1]$ and y_i, is a continuous, piecewise linear spline function with knots in x_i, $i = 1, \ldots, n$.

This solution is called the linear smoothing spline, pertaining to data x_i, y_i, $i = 1, \ldots, n$.

Example 3: Cubic smoothing spline

The RKHS $\mathcal{H}$ is

$$H_0^2(0,1) := \left\{ f \in L_2(0,1) : f' \text{ exists, abs. continuous, }, f'' \in L_2(0,1),\ f(0) = f'(0) = 0 \right\}$$

with scalar product

$$\langle f_1, f_2 \rangle_* = \int_{[0,1]} f_1''(x) f_2''(x) dx$$

and norm $\|f\|_*^2 = \|f''\|_2^2$. The reproducing kernel [BTA], p. 287 [2] is

$$K(s,t) = \int_0^1 (t-w)_+ (s-w)_+ \, dw. \tag{61}$$

Let us evaluate the integral: for $s \le t$ we have

$$\begin{aligned} K(s,t) &= \int_0^s (t-w)(s-w)\, dw = \int_0^s \left(w^2 - (s+t)w + st\right) dw \\ &= s^3/3 - (s+t)s^2/2 + s^2 t = -s^3/6 + s^2 t/2. \end{aligned}$$

[2] Berlinet, A., Thomas-Agnan, C., Reproducing kernel Hilbert spaces in Probability and Statistics, Kluwer, 2004, p. 287

For $s \geq t$ the roles of s and t switch, thus we have

$$K(s,t) = \begin{cases} k(s,t) \text{ if } s \leq t \\ k(t,s) \text{ if } s \geq t \end{cases}$$
$$\text{for } k(s,t) = ts^2/2 - s^3/6.$$

The symmetry $K(s,t) = K(t,s)$ follows from this (or also directly from the integral representation (61). The function $K(\cdot,t)$ is a third degree polynomial on $[0,t]$ and a linear function on $[t,1]$, with common value $k(t,t) = t^3/3$ at $s = t$. Hence $K(\cdot,t)$ is continuous. The derivatives of the pieces are

$$\frac{\partial}{\partial s} K(s,t) = \begin{cases} ts - s^2/2 \text{ if } s \leq t \\ t^2/2 \text{ if } s \geq t \end{cases}$$

with common value $t^2/2$ at $s = t$, hence $K(\cdot,t)$ is continuously differentiable. The second derivatives of the pieces are

$$\frac{\partial^2}{\partial s^2} K(s,t) = \begin{cases} t - s \text{ if } s \leq t \\ 0 \text{ if } s \geq t \end{cases}$$

hence is $K(\cdot,t)$ twice continuously differentiable (i.e. is in C^2) and the second derivative is a continuous piecewise linear spline. The third derivative is the step function $-\mathbf{1}_{[0,t]}$. Functions like these (piecewise polynomial of 3rd degree) with smoothness properties are called *cubic splines*; thus $K(\cdot,t)$ is a cubic spline function which is in C^2. The points where the polynomials change are called *knots*; thus $K(\cdot,t)$ has one knot in t.
To check the reproducing property, note

$$\begin{aligned} \langle f, K(\cdot,x)\rangle_* &= \int_{[0,1]} f''(s) \frac{\partial^2}{\partial s^2} K\prime(s,x)\, ds = \int_0^x f''(s)(x-s)\, ds \\ &= \left[f'(s)(x-s)\right]_0^x + \int_0^x f'(s) ds = \\ &= f'(x)(x-x) - f'(0)x + f(x) - f(0) = f(x). \end{aligned}$$

Positive definiteness follows from the integral representation (61).
We may now consider the minimization problem

$$n^{-1} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int_0^1 \left(f''(t)\right)^2 dt$$

over $f = \tilde{h} + g$, $\tilde{h} \in H_0^2(0,1)$ and g a linear function $(g(x) = g_1 x + g_0)$. Here $f'' = \tilde{h}''$ so that the original penalty term $\int_0^1 \left(\tilde{h}''(t)\right)^2 dt$ can also be written in terms of f. According to the generalized representer theorem (Theorem 6.2) any solution can be written

$$f = \sum_{i=1}^{n} \alpha_i K(\cdot, x_i) + g.$$

Moreover, consider a version of the space $H_0^2(0,1)$ without boundary conditions:

$$H^2(0,1) := \left\{ f \in L_2(0,1) : f' \text{ exists, abs. continuous, } , f'' \in L_2(0,1) \right\}.$$

Obviously any function $f \in H^2(0,1)$ has a unique decomposition $f = \tilde{h} + g$, $\tilde{h} \in H_0^2(0,1)$ and g a linear function: determine g by $g(0) = f(0)$, $g'(0) = f'(0)$, then it is clear that $f - g \in H_0^2(0,1)$.

6.6 Corollary *Any solution of the problem of minimizing*

$$n^{-1} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int_0^1 \left(f''(t)\right)^2 dt$$

over functions $f \in H^2(0,1)$, for given $x_i \in [0,1]$ and y_i, is a cubic spline function in C^2 with knots in x_i, $i = 1, \ldots, n$.

This solution is called the cubic smoothing spline, pertaining to data x_i, y_i, $i = 1, \ldots, n$. For more insight on spline smoothing and RKHS see Wahba [Wah] and Berlinet, Thomas-Agnan [BTA]

6.3 Smoothing in classification

Above we discussed examples of smoothing splines for regression. Our main interest is classification however, and we originally introduced the regularization idea in RKHS in that context. In (43) we defined the soft margin loss for misclassification, we considered the corresponding soft margin risk and its penalized version (44). Our concrete examples of RKHS were for $x \in (0,1)$ however, and for classification we should certainly discuss some examples where $x \in \mathbb{R}^d$. Before that however, let us discuss another approach to make the smoothing idea work in classification. This approach is based on logistic regression.

Logistic regression

In the classification framework, assume that $Y \in \{0,1\}$ and $X \in \Omega \subset \mathbb{R}^d$, and (X,Y) have a joint distribution. We observe an i.i.d. training set (X_i, Y_i), $i = 1, \ldots, n$ all distributed like (X,Y) and we are required, for a "new" incoming pair (X,Y) of which only X is observed, to predict Y. Denote

$$p_x := P\left(Y = 1 | X = x\right)$$

and note that the joint law of (X,Y) is determined by p_x and the marginal law of X, say $\mathcal{L}(X)$. Indeed, given $X = x$, the conditional law of Y is a Bernoulli distribution with parameter p_x and $P\left(Y = 0 | X = x\right) = 1 - p_x$. In logistic regression it is assumed that

$$\log \frac{p_x}{1 - p_x} = \beta_0 + \langle \beta, x \rangle \tag{62}$$

for some $\beta \in \mathbb{R}^d$ and $\beta_0 \in \mathbb{R}$. The parameters β, β_0 are estimated from the training set as $\hat{\beta}, \hat{\beta}_0$, giving for each x an estimated $\hat{p}_x$ by solving (62). This $\hat{p}_x$ provides an obvious classifier h: $h(x) = 1$

if $\hat{p}_x > 1/2$. Recall that the Bayes classifier h^* is a function of the posterior probability p_x and classifies 1 if $p_x > 1/2$.
Since $\hat{p}_x > 1/2$ is equivalent to $\log\left(\hat{p}_x/\left(1-\hat{p}_x\right)\right) > 0$, the classifier ϕ is linear:

$$\phi(x) = \begin{cases} 1 \text{ if } \beta_0 + \langle \beta, x\rangle > 0 \\ 0 \text{ otherwise.} \end{cases}$$

Note that solving (62) for p_x gives

$$p_x = \frac{\exp\left(\beta_0 + \langle \beta, x\rangle\right)}{1+\exp\left(\beta_0 + \langle \beta, x\rangle\right)}.$$

The function

$$l(t) = \frac{\exp(t)}{1+\exp(t)}$$

is known as the *logistic function* and its inverse

$$\operatorname{logit}(u) = \log \frac{u}{1-u}$$

as the *logit function* (check $\operatorname{logit}(l(t)) = t$). Since p_x is also the expectation $E\left(Y|X=x\right)$, we have for the regression function in the logistic regression model

$$r(x) = E\left(Y|X=x\right) = p_x = l(\beta_0 + \langle \beta, x\rangle)$$

Relation to the normal model. When Fisher's LDA was discussed, a normal model for the class-conditional distributions $\mathcal{L}\left(X|Y=i\right)$ was assumed:

$$\mathcal{L}\left(X|Y=i\right) = N_d\left(m_i, \Sigma\right),\ i = 0,1 \tag{63}$$

with equal prior probabilities

$$\pi_i = P\left(Y=i\right) = 1/2, i = 0,1. \tag{64}$$

Together these two assumptions determine the joint distribution of (X,Y). In the logistic regression model, one proceeds the other way round: one specifies

$$\mathcal{L}\left(Y|X=x\right) = \text{Bernoulli}\left(p_x\right),\ p_x = l(\beta_0 + \langle \beta, x\rangle) \tag{65}$$

and one leaves $\mathcal{L}\left(X\right)$, the marginal law of X, unspecified. Thus, to build a joint distribution of (X,Y) here, one may assume *any* distribution for X.

6.7 Proposition *The class-conditional normal model (63), (64) is a logistic regression model (65) with*

$$\beta = \Sigma^{-1}\left(m_1 - m_0\right),$$
$$\beta_0 = \frac{1}{2}\left({m_0}^{\top}\Sigma^{-1}m_0 - {m_1}^{\top}\Sigma^{-1}m_1\right)$$

where $\mathcal{L}(X)$ is a mixture of normals: if ϕ_i is the density of $N_d(m_i, \Sigma)$, $i = 0, 1$ then the density ϕ of X is

$$\phi(x) = \frac{1}{2}\phi_1(x) + \frac{1}{2}\phi_2(x).$$

Proof. We assume $\Sigma = I_d$; the proof for general Σ is analogous with a transformation of X by $\Sigma^{-1/2}$ first. In the subsection "Linear estimation of the regression function" (in relation (49)) we proved that in the normal model (63), (64) when we use a class index $\tilde{Y} = 2Y - 1$ (taking values in $\{-1, 1\}$) then

$$\tilde{r}(x) := E\left(\tilde{Y}|X = x\right) = \tanh\left(\frac{1}{2}\left(\langle a, x\rangle + a_0\right)\right)$$

for $a = m_1 - m_0$, $a_0 = \frac{1}{2}\left(\|m_0\|^2 - \|m_1\|^2\right)$, where

$$\tanh(u) = \frac{\exp(u) - \exp(-u)}{\exp(u) + \exp(-u)}.$$

Since $Y = \left(\tilde{Y} + 1\right)/2$, we obtain for $u = \langle a, x\rangle + a_0$

$$\begin{aligned} r(x) &= E(Y|X = x) = \frac{1}{2}\left(\tanh(u/2) + 1\right) \\ &= \frac{1}{2}\frac{2\exp(u/2)}{\exp(u/2) + \exp(-u/2)} = \frac{\exp(u)}{\exp(u) + 1} = l(u) \end{aligned}$$

i.e.

$$r(x) = l(\langle a, x\rangle + a_0).$$

For $\Sigma = I_d$ we have $a = \beta$ and $a_0 = \beta_0$ as claimed. It remains to find the marginal distribution of X. This is immediate: for any measurable $A \subset \mathbb{R}^d$

$$\begin{aligned} P(X \in A) &= \frac{1}{2}P(X \in A|Y = 0) + \frac{1}{2}P(X \in A|Y = 1) \\ &= \frac{1}{2}\int_A \phi_0(x)dx + \frac{1}{2}\int_A \phi_1(x)dx = \int_A \phi(x)dx \end{aligned}$$

so that the density ϕ of X is as claimed. ■

Thus the claim (18) about the Bayes rule in the normal model

$$h_N^*(x) = \begin{cases} 1 \text{ if } x^\top a + a_0 > 0 \\ 0 \text{ otherwise} \end{cases}$$

is just a special case of the fact that in the logistic regression model, for any marginal distribution of X, the Bayes rule is

$$h^*(x) = \begin{cases} 1 \text{ if } r(x) > 1/2 \\ 0 \text{ otherwise} \end{cases} = \begin{cases} 1 \text{ if } \beta_0 + \langle \beta, x\rangle > 0 \\ 0 \text{ otherwise} \end{cases}.$$

Estimation of parameters. The parameters β, β_0 are estimated from the realized training set (x_i, y_i), $i = 1, \ldots, n$ by maximum likelihood, conditionally on $X_1, \ldots, X_n$. Taking this conditional point of view is logical, since we left $\mathcal{L}(X)$ unspecified and the parameters of interest concern only the conditional distribution $\mathcal{L}(Y_i|X_i)$. To obtain the likelihood function, note that if $Y \sim$ Bernoulli (p) then the probability function can be expressed (for $y \in \{0, 1\}$) as

$$p^y(1-p)^{1-y}$$

which takes value p if $y = 1$ and value $1-p$ if $y = 0$. The log-likelihood function for Y_i, $i = 1, \ldots, n$ thus is

$$\begin{aligned} L(\beta_0, \beta) &= \log \prod_{i=1}^{n} p_{x_i}^{y_i} \left(1 - p_{x_i}\right)^{1-y_i} \\ &= \sum_{i=1}^{n} y_i \log r(x_i) + (1-y_i) \log\left(1 - r(x_i)\right) \\ &= \sum_{i=1}^{n} y_i \log l(\beta_0 + \langle \beta, x \rangle) + (1-y_i) \log\left(1 - l(\beta_0 + \langle \beta, x \rangle)\right) \end{aligned}$$

where l is the logistic function. This can be slightly simplified, in view of the form of l. The maximization is performed numerically.

Nonparametric version

The above approach might be called *linear logistic regression* since in $r(x) = l(\beta_0 + \langle \beta, x \rangle)$, the term $\beta_0 + \langle \beta, x \rangle$ is linear (even though the regression function $r(x)$ is nonlinear). We may now substitute $\beta_0 + \langle \beta, x \rangle$ by a function $f(x)$ and build a regularized risk functional, analogously to (44)

$$R_{reg}(f) = R_{emp}^{lik}(f) + \lambda \|f\|_*^2$$

where

$$R_{emp}^{lik}(f) = -L(f) = -\left(\sum_{i=1}^{n} y_i \log l(f(x_i)) + (1-y_i) \log\left(1 - l(f(x_i))\right) \right).$$

Indeed taking the *negative* log-likelihood leads to a minimization problem, and the expression $R_{emp}^{lik}(f)$ is then analogous to the other risk criteria such as soft margin risk, misclassification risk and also quadratic risk for regression. The term $\lambda \|f\|_*^2$ is a penalty where $\|f\|_*^2$ is a smoothness measure and λ is chosen. In that framework, smoothing spline theory can now be applied to classification.

It is possible to obtain this framework formally from the feature space idea and RKHS theory. Recall that in this setting, (X_i, Y_i) is substituted by $\left(K(\cdot, X_i), Y_i\right)$ where K is a kernel. For linear logistic regression, the regression function then is

$$r(x) = l\left(\beta_0 + \langle g, K(\cdot, x) \rangle_*\right) = l\left(\beta_0 + g(x)\right)$$

so the classification rule obtained from estimating $f = \beta_0 + g$ is no longer linear.

6.4 Bayesian smoothing

Let us return to the regression (non-classification) setup discussed in example 1 above. We are in the general regression situation and minimizing

$$n^{-1}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda \left\|f'\right\|_2^2$$

over $f \in H_0^1(0,1)$. The result was a piecewise linear spline f with $f(0) = 0$ and $f(1) = f(x_n)$. We will show that this estimator of the regression function f can also be obtained in a *Bayesian setting* with a certain (Gaussian) prior on the regression function f. We will then develop the analog in the classification problem, via the logistic regression setup. This gives rise to the *Gaussian process (GP) classifiers*, closely related to the penalized kernel and RKHS methods discussed previously. GP classifiers are treated in the book [RW][3]

Regression with Gaussian prior

Let us specify the regression model as normal: given $X_i = x_i$, the Y_i are normal $N\left(f(x_i), \sigma^2\right)$ where σ^2 is known. With regard to the x_i, we will take the conditional point of view and assume they are nonrandom. Our parameter of interest is the function f. The model can thus equivalently be written

$$Y_i = f(x_i) + \varepsilon_i \tag{66}$$

where ε_i are i.i.d. $N(0,\sigma^2)$. It is well known that when we assume a parameter space $f \in \mathcal{F}$ and consider maximum likelihood estimation, the maximum likelihood criterion is equivalent to the least squares criterion. Indeed the likelihood function for f is

$$\frac{1}{(2\pi)^{n/2}\,\sigma^n}\exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - f(x_i))^2\right)$$

and taking the negative log-likelihood, recalling that σ is known, we see that maximizing the likelihood over $f \in \mathcal{F}$ is equivalent to minimizing

$$\sum_{i=1}^{n}(y_i - f(x_i))^2$$

over $f \in \mathcal{F}$, i.e. the least squares criterion. Let us now consider a Bayesian approach to estimation where we place a prior distribution on f, in the following form. Consider the kernel $K(x,y) = \min(x,y)$ and write its Mercer expansion on $[0,1]$ as

$$K(x,y) = \sum_{j=1}^{\infty}\lambda_j\phi_j(x)\phi_j(y) \tag{67}$$

[3]Rasmussen, C. E. and Wiliams, C.K. I., *Gaussian Processes for Machine Learning*, MIT Press, 2006.

for the functions $\phi_j(x) = \sqrt{2}\sin\left(\left(j - 1/2\right)\pi x\right)$, $j = 1, 2, \ldots$ and $\lambda_j = \left(j - 1/2\right)^{-2}\pi^{-2}$ (see Proposition 5.7). Assume f is square integrable on $[0,1]$ and write the expansion of f in terms of the basis $\{\phi_j, j = 1, 2, \ldots\}$ as

$$f = \sum_{j=1}^{\infty} f_j \phi_j \text{ where } f_j = \langle f, \phi_j \rangle$$

(where $\langle \cdot, \cdot \rangle$ is the scalar product in $L_2[0,1]$). Recall that we considered the function space as

$$H_0^1(0,1) := \left\{ f \in L_2(0,1) : f \text{ abs. continuous, } f(0) = 0, f' \in L_2(0,1) \right\}$$

as RKHS for K, with norm $\|f\|_*^2 = \|f'\|_2^2$. For any $f \in H_0^1(0,1)$ this norm is for $\gamma_j = \lambda_j^{-1/2} = \left(j - 1/2\right)\pi$

$$\|f'\|_2^2 = \sum_{j=1}^{\infty} \frac{1}{\lambda_j} f_j^2 = \sum_{j=1}^{\infty} f_j^2 \gamma_j^2.$$

Bayesian prior distribution on f:

$$\textit{assume } f \textit{ is random such that } f_j \textit{ are independent } N\left(0, \lambda_j\right). \tag{68}$$

With this prior, we are assuming that the Fourier coefficients of f are random, and since $\lambda_j \to 0$ as $j \to \infty$ and small variance means that the f_j is more likely to be close to 0, this prior can be interpreted as a *smoothness prior*. However it does not mean that f is in $H_0^1(0,1)$ almost surely: $\lambda^{-1/2} f_j$ are i.i.d. standard normal variables, ξ_j say, and

$$\|f\|_*^2 = \sum_{j=1}^{\infty} \frac{1}{\lambda_j} f_j^2 = \sum_{j=1}^{\infty} \xi_j^2 = \infty \text{ almost surely.}$$

6.8 Lemma *For a sequence ξ_j of i.i.d. standard normal random variables, $\sum_{j=1}^{\infty} \xi_j^2 = \infty$ almost surely.*

Proof. Let $S = \sum_{j=1}^{\infty} \xi_j^2$. Then, for $K > 0$ and $m > K$, since $E\xi_j^2 = 1$, and $\operatorname{Var}\left(\xi_j^2\right) = 2$,

$$\begin{aligned} P\left(S \le K\right) &\le P\left(\sum_{j=1}^{m} \xi_j^2 \le K\right) = P\left(\sum_{j=1}^{m} \xi_j^2 - m \le K - m\right) \\ &\le P\left(\left|\sum_{j=1}^{m} \left(\xi_j^2 - E\xi_j^2\right)\right| \ge m - K\right) \le \frac{m \operatorname{Var}\left(\xi_1^2\right)}{\left(m - K\right)^2} = \frac{2m}{\left(m - K\right)^2} \end{aligned}$$

according to Chebyshev's inequality, where the upper bound can be made arbitrarily small for fixed K, by letting $m \to \infty$. Thus $P(S \le K) = 0$ for all $K > 0$, hence $P(S = \infty) = 1$. ■
Now f itself can be represented

$$f(x) = \sum_{j=1}^{\infty} f_j \phi_j(x) = \sum_{j=1}^{\infty} \lambda_j^{1/2} \xi_j \phi_j(x)$$

where as above, ξ_j are i.i.d. standard normal. If this is a well-defined random variable for every x, i.e. the series converges almost surely, then $f(x)$ represents a stochastic process. Consider the partial sums $f_m(x) = \sum_{j=1}^{m} \lambda_j^{1/2} \xi_j \phi_j(x)$; this is a Gaussian stochastic process with

$$\begin{aligned} Ef_m(x) &= 0; \\ Ef_m(x)f_m(y) &= E\left(\sum_{i=1}^{m} \lambda_i^{1/2} \xi_i \phi_i(x)\right)\left(\sum_{j=1}^{m} \lambda_j^{1/2} \xi_j \phi_j(y)\right) = \sum_{j=1}^{m} \lambda_j E\xi_j^2 \phi_j(x)\phi_j(y) \\ &= \sum_{j=1}^{m} \lambda_j \phi_j(x)\phi_j(y) \to_{m\to\infty} K(x,y) = \min(x,y) \end{aligned}$$

in view of the Mercer expansion (67). Thus we have heuristic evidence for:

6.9 Proposition *The Bayesian prior on f given by (68) makes f a Gaussian stochastic process with zero expectation and covariance function*

$$Ef(t)f(u) = \min(t,u)$$

i. e. the Wiener process $W(t)$ (or Brownian motion) on $[0,1]$.

(A rigorous proof would argue that the r.v.'s $f_m(x)$ converge in quadratic mean to a r.v. which is finite a.s., then show that this is a Gaussian r.v. via the characteristic function, then show it jointly for every finite collection $f_m(x_1), \ldots, f_m(x_k)$ etc. This would be analogous to the construction of Brownian motion in [D][4])
Recall that the Brownian motion can also be characterized as zero mean Gaussian with independent increments and $\mathrm{Var}(W(t)) = t$. Let us compute the covariance function of $W(t)$ from there. For $t \le u$, the r.v. $W(t)$ and the increment $W(u) - W(t)$ are independent, hence the covariance function is

$$\begin{aligned} EW(u)W(t) &= E\left(W(t) + W(u) - W(t)\right)W(t) = EW^2(t) + E\left(W(u) - W(t)\right)EW(t) \\ &= EW^2(t) = \mathrm{Var}(W(t)) = t = \min(t,u). \end{aligned}$$

It is well known that the Brownian motion is a continuous random function with $W(0) = 0$ which is nowhere differentiable. Thus we can say that the prior distribution we put on f in (68) makes

[4] Durret, R., *Probability: Theory and Examples*, 2nd ed., chap. 7.1

f "somewhat" smooth (at least continuous), but certainly we do not have $f \in H_0^1(0,1)$, as also shown before. It is possible to achieve "genuine" or higher smoothness with Gaussian priors, e.g. by setting $f(t) \simeq \int_0^t W(u)du$ (integrated Brownian motion) or by taking other values $\lambda_j \to 0$ with faster rate.
Bayesian inference with Gaussian process prior. Our model is the regression model (66)

$$Y_i = f(x_i) + \varepsilon_i$$

where x_i are nonrandom and given x_i and f, the Y_i are independent $N(f(x_i), \sigma^2)$. In a Bayesian setting, f now has "become" random. But the distribution of the Y_i does not depend on the function f as a whole, but only on the values $f(x_1), \ldots, f(x_n)$. Thus effectively we have put a prior on the vector $\bar{f} = (f(x_1), \ldots, f(x_n))^\top$: it is Gaussian with 0 expectation and covariance matrix

$$Ef(x_i)f(x_j) = K(x_i, x_j) = \min(x_i, x_j), \; i, j = 1, \ldots, n.$$

Write

$$\bar{K} = (K(x_i, x_j))_{i,j=1}^n, \; y = (y_1, \ldots, y_n)^\top.$$

We showed before that the matrix $\bar{K}$ is nonsingular if all x_i are different (in connection with (57)). Consider the Bayesian formalism: we have a parametric family of densities for observed y: $p(y|\vartheta)$, $y \in \mathbb{R}^n$, $\vartheta \in \Theta$ where $\Theta = \mathbb{R}^n$. Here $\vartheta = \bar{f}$ and $p(y|\vartheta) \doteq N_n(\hat{f}, \sigma^2 I_n)$ (here is "$\doteq$" means "is the density of"). In a Bayesian statistical approach, assume that ϑ "becomes random" as well. Suppose we have a prior density $\pi(\vartheta)$; presently $\pi(\vartheta) \simeq N_n(\mathbf{0}, \bar{K})$ where $\mathbf{0}$ is the 0-vector. The joint density of y and ϑ is

$$p(y|\vartheta)\pi(\vartheta)$$

and the posterior density of ϑ is

$$p(\vartheta|y) = \frac{p(y|\vartheta)\pi(\vartheta)}{\int p_\vartheta(y)\pi(\vartheta)d\vartheta}.$$

It is well known that if both $p(y|\vartheta)$ and $\pi(\vartheta)$ are Gaussian densities, then $p(\vartheta|y)$ is also Gaussian. Common Bayesian estimators are the mode of the posterior density (MAP estimator, maximum a posteriori) and the posterior expectation, i.e. the expected value of $p(\vartheta|y)$. Since $p(\vartheta|y)$ is Gaussian, both coincide, and the posterior expectation is

$$\begin{aligned}\arg\max_\vartheta p(\vartheta|y) &= \arg\max_\vartheta p(y|\vartheta)\pi(\vartheta) = \arg\max_\vartheta (\log p(y|\vartheta) + \log \pi(\vartheta)) \\ &= \arg\min_\vartheta (-\log p(y|\vartheta) - \log \pi(\vartheta)) \\ &= \arg\min_{\hat{f}} \left(\frac{1}{2\sigma^2} \left\| y - \bar{f} \right\|^2 + \frac{1}{2} \bar{f}^\top \bar{K}^{-1} \bar{f}. \right)\end{aligned}$$

This is closely related to the minimization problem for the special linear smoothing spline obtained in (58), i.e where the target turned out to be

$$n^{-1} \left\| y - \bar{f} \right\|^2 + \lambda \bar{f}^\top K^{-1} \bar{f}.$$

For more flexibility, we may also consider a prior on f as a $f_j \simeq N(0, \delta\lambda_j)$ in (68) where $\delta > 0$.

6.10 Theorem *Suppose that in the regression model*

$$Y_i = f(x_i) + \varepsilon_i$$

where ε_i are i.i.d. $N(0, \sigma^2)$ and x_i are nonrandom, we assume a prior on f as $f \simeq \delta^{1/2} W$, where $\delta > 0$ and W is the standard Brownian motion on $[0, 1]$. Then the Bayes estimator of $\bar{f} = (f(x_1), \ldots, f(x_n))^\top$ coincides with the values at x_i of the minimizer of

$$n^{-1} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \left\| f' \right\|_2^2$$

over $f \in H_0^1(0,1)$, i.e. the special linear smoothing spline, for $\lambda = \sigma^2 / n\delta$.

With more reasoning, it can also be shown that for f itself, the Bayes estimator is the smoothing spline; one then has to take the expectation of the random pieces of f between the x_i conditionally on $f(x_i)$ (Brownian bridges).

Example b: Cubic smoothing spline

Consider a prior distribution on f as

$$f \simeq \delta^{1/2} V$$

where

$$V(t) = \int_0^t W(u) du$$

is the integrated Brownian motion on $[0, 1]$. Since W is continuous a.s., the integral is well defined and produces a Gaussian process. Let us compute the covariance function:

$$EV(s)V(t) = \int_0^s \int_0^t EW(u)W(v) dv du = \int_0^s \int_0^t \min(u, v)\, dv du.$$

When discussing the kernel $\min(u, v)$ we used the representation

$$\min(u, v) = \int_0^1 \mathbf{1}_{[0,u]}(w) \mathbf{1}_{[0,v]}(w) dw.$$

Plugging in we obtain

$$\begin{aligned} EV(s)V(t) &= \int_0^s \int_0^t \left(\int_0^1 \mathbf{1}_{[0,u]}(w) \mathbf{1}_{[0,v]}(w) dw \right) dv du \\ &= \int_0^1 \left(\int_0^s \mathbf{1}_{[0,u]}(w) du \right) \left(\int_0^t \mathbf{1}_{[0,v]}(w) dv \right) dw \\ &= \int_0^1 \left(\int_0^s \mathbf{1}_{\{u \geq w\}}(u) du \right) \left(\int_0^t \mathbf{1}_{\{v \geq w\}}(v) dv \right) dw \\ &= \int_0^1 (s - w)_+ (t - w)_+ dw. \end{aligned}$$

This is exactly the kernel $K(s,t)$ we found as reproducing for the space $H_0^2(0,1)$ when discussing the cubic smoothing spline, cf. (61):

$$K(s,t) = \begin{cases} k(s,t) \text{ if } s \leq t \\ k(t,s) \text{ if } s \geq t \end{cases}$$
$$\text{for } k(s,t) = ts^2/2 - s^3/6.$$

Thus we have shown:

6.11 Theorem *Suppose that in the regression model*

$$Y_i = f(x_i) + \varepsilon_i$$

where ε_i are i.i.d. $N(0,\sigma^2)$ and x_i are nonrandom, we assume a prior on f as $f \simeq \delta^{1/2} V$, where $\delta > 0$ and $V(t) = \int_0^t W(u)du$ is the integrated Wiener process on $[0,1]$. Then the Bayes estimator of $\bar{f} = (f(x_1),\ldots,f(x_n))^\top$ coincides with the values at x_i of the minimizer of

$$n^{-1}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda \left\|f''\right\|_2^2$$

over $f \in H_0^2(0,1)$, i.e. the (special) cubic linear smoothing spline, for $\lambda = \sigma^2/n\delta$.

Gaussian process classifiers

In the classification problem we assume a logistic link function:

$$P(Y=1|X=x)=l(f(x))$$

and we put a Gaussian process prior on the function f (also called latent function). The negative log-likelihood is, for $Y\in\{0,1\}$

$$R^{lik}_{emp}(f)=-L(f)=-\left(\sum_{i=1}^{n}y_i\log l(f(x_i))+(1-y_i)\log(1-l(f(x_i)))\right)=$$
$$=-\sum_{i=1}^{n}L_i(f) \text{ where } L_i(f)=\left\{\begin{array}{l}-\log l(f(x_i)) \text{ if } y_i=1\\ -\log(1-l(f(x_i))) \text{ if } y_i=0\end{array}\right. ..$$

Let us give an alternative expression in case we use $\tilde{Y}=2Y-1$ as a class index. The logistic function $l(t)=\frac{\exp(t)}{1+\exp(t)}$ has the property that $l(-t)=1-l(t)$. Hence $\log(1-l(f(x_i)))=\log l(-f(x_i))$ which is $\log l(\tilde{y}_i f(x_i))$ if $y_i=0$. Thus we may write

$$-L(f)=-\sum_{i=1}^{n}L_i(f) \text{ where } L_i(f)=\left\{\begin{array}{l}\log l(f(x_i)) \text{ if } \tilde{y}_i=1\\ \log l(-f(x_i)) \text{ if } \tilde{y}_i=-1\end{array}\right. =\log l(\tilde{y}_i f(x_i))$$

Then the Bayesian negative log-posterior density is (up to a fixed additive term)

$$-\sum_{i=1}^{n}\log l(\tilde{y}_i f(x_i))+\frac{1}{2}\bar{f}^\top\bar{K}^{-1}\bar{f}. \tag{69}$$

This is minimized numerically, to obtain Bayesian a estimate of $\bar{f}$. The classifier then produces 1 if $l\left(\bar{f}(x_i)\right)>1/2$.
The most commonly used covariance function is the squared exponential: for $x_1,x_2\in\mathbb{R}^d$

$$\operatorname{cov}(f(x_1),f(x_2))=K(x_1,x_2)=\sigma_f^2\exp\left(-\frac{1}{2\ell}\|x_1-x_2\|^2\right)$$

where σ_f^2, ℓ are hyperparameters of the prior distribution- they add flexibility to the Bayesian analysis. In Lemma 5.3 we have shown that this kernel is a Mercer kernel.

Predictions. Minimization of the expression (69) does not give the whole story for Bayesian classification, since it does not describe how to make a prediction for a test case x_*. From (69) we only obtain a Bayes (vector) estimate $\hat{f}=\left(\hat{f}(x_1),\dots,\hat{f}(x_n)\right)^\top$ which produces classifications of the points x_i, $i=1,\dots,n$ of the training set which can then be compared with the observed

y_i. However the purpose of classification is to predict y_* for a "new" test points x_*. To obtain the full Bayesian treatment of this problem, note that when we put a Gaussian process prior on f, all the values $f(x_1), \ldots, f(x_n), f(x_*)$ have a joint Gaussian distribution, with 0 expectation and covariance matrix induced by the kernel K. Let again $\bar{f} = (f(x_1), \ldots, f(x_n))^\top$ and $f_* = f(x_*)$, recall that all x_i, x_* are nonrandom and let the posterior density (given observed $y = (y_1, \ldots, y_n)^\top$) of $\bar{f}$ be

$$p\left(\bar{f}|y\right) = \frac{p\left(y|\bar{f}\right)\pi(\bar{f})}{\int p\left(y|\bar{f}\right)\pi(\bar{f})d\bar{f}} \tag{70}$$

where $\pi(\bar{f})$ is the prior density on $\bar{f}$. By taking $-\log p\left(\bar{f}|y\right)$ we obtain (69). Consider now the conditional density $\pi(f_*|\bar{f})$ derived from the joint prior distribution of $\bar{f}, f_*$. Then

$$p\left(\bar{f}, f_*|y\right) = \pi(f_*|\bar{f})p\left(\bar{f}|y\right)$$

gives the joint posterior density of $\bar{f}, f_*$.

Proof. We may carry out the formalism (70) jointly for $\bar{f}, f_*, y$: first build a joint density of $\bar{f}, f_*, y$ by noting that $p\left(y|\bar{f}, f_*\right) = p\left(y|\bar{f}\right)$ and hence the joint density of v, f_*, y is

$$p\left(y|\bar{f}, f_*\right)\pi\left(\bar{f}, f_*\right) = p\left(y|\bar{f}\right)\pi\left(\bar{f}, f_*\right) = p\left(y|\bar{f}\right)\pi(f_*|\bar{f})\pi(\bar{f}).$$

Then the joint posterior of f, f_* is

$$\begin{aligned} p\left(\bar{f}, f_*|y\right) &= \frac{p\left(y|\bar{f}, f_*\right)\pi\left(\bar{f}, f_*\right)}{\int\int p\left(y|\bar{f}, f_*\right)\pi\left(\bar{f}, f_*\right)df_* d\bar{f}} = \frac{p\left(y|\bar{f}\right)\pi(f_*|\bar{f})\pi(\bar{f})}{\int p\left(y|\bar{f}\right)\left(\int \pi(f_*|\bar{f})df_*\right)\pi(\bar{f})d\bar{f}} \\ &= \frac{p\left(y|\bar{f}\right)\pi(f_*|\bar{f})\pi(\bar{f})}{\int p\left(y|\bar{f}\right)\pi(\bar{f})d\bar{f}} = p\left(\bar{f}|y\right)\pi(f_*|\bar{f}). \end{aligned}$$

■

By integrating out $\bar{f}$ we obtain the (marginal) posterior density of f_*:

$$p\left(f_*|y\right) = \int p\left(\bar{f}|y\right)\pi(f_*|\bar{f})d\bar{f}.$$

Now $P\left(Y_* = 1|f(x_*), y\right) = l\left(f(x_*)\right) = l\left(f_*\right)$ is a deterministic function of f_* where l is the logistic function, so when f_* is random with its posterior density $p\left(f_*|y\right)$ then Y_* is Bernoulli with expectation

$$P\left(Y_* = 1|y\right) = \int l\left(f_*\right)p\left(f_*|y\right)df_*. \tag{71}$$

The natural prediction (classification of x_*) then is $h(x_*) = 1$ if $P\left(Y_* = 1|y\right) \geq 1/2$.

The expression (71) depends on x_* which was assumed fixed; for clarity we may write it $P(Y_* = 1|y, x_*)$. The book [RW] in sections 3.4, 3.6 describes two different computational approaches (Laplace approximation, EP algorithm) to obtaining $P(Y_* = 1|y, x_*)$ for an arbitrary x_*. It also describes experiments with the resulting Gaussian classifiers for handwritten digit recognition, on the US Postal service database of handwritten digits (p 63 ff. in [RW]).

Simple special case: linear logistic regression, Bayesian. A Gaussian process is a random function $f(x)$ such that all values $f(x_1), \ldots, f(x_m)$ have a joint normal distribution for arbitrary $x_1, \ldots, x_m$ (generally in $\mathbb{R}^d$). It is possible to give such a process as a linear function of x: set

$$f(x) = w^\top x, \; w \sim N_d(0, \Sigma)$$

where the $d \times d$ matrix Σ is positive definite. In this case our model is linear logistic regression

$$P(Y = 1|X = x) = l\left(w^\top x\right)$$

where a normal prior $N_d(0, \Sigma)$ is put on the weight vector $w \in \mathbb{R}^d$.
The book [RW] describes an instructive toy example for $d = 2$ and a given training set of 6 points in the plane (3 in each class). The prior on w is chosen as standard normal $N_2(0, I)$. The posterior of w is shown in the lower left figure (c). The maximum appears around the vector $(1/2, 1/2)^\top$ which agrees with the fact that the best classifying hyperplane through the origin appears to be perpendicular to this vector. (Note that in this model, the Bayesian classifier hyperplane is forced to go through the origin since we did not include a constant term in f, i.e did not set $P(Y = 1|X = x) = l\left(w^\top x + c\right)$). In this simple model, we automatically obtain the posterior of $f_* = f(x_*) = w^\top x_*$ from the posterior on w, i.e. we shortcut the reasoning leading to (71), but it nevertheless is a special case. Then (71) simplifies to

$$P(Y_* = 1|y, x_*) = \int l\left(w^\top x_*\right) p(w|y)\, dw. \tag{72}$$

As a function of x_*, this is called the predictive distribution; it is plotted in the lower right figure (d). The contour $P(Y_* = 1|y, x_*) = 1/2$ is linear and gives the Bayes classifier hyperplane through the origin.

Bayesian logistic regression (summary of example from book Rasmussen, Williams, [RW], p. 39))
$Y \in \{1, -1\}$, $X \in \mathbb{R}^2$, linear logistic regression

$$P(Y = 1|X = x) = l\left(w^\top x\right)$$

Gaussian process prior on function $f(x) = w^\top x$: vector w is normal $N_2(0, \Sigma)$ for $\Sigma = I$

implies bilinear covariance function for Gaussian process $f(x)$: for $x_1, x_2 \in \mathbb{R}^2$

$$\begin{aligned}\operatorname{cov}\left(f(x_1), f(x_2)\right) = K(x_1, x_2) &= E\left(w^\top x_1\right)\left(w^\top x_2\right) \\ &= x_1^\top \Sigma x_2 = x_1^\top x_2.\end{aligned}$$

Panel (a): plot of prior distribution $w \sim N_2(0, I)$
Panel (b): training set, 3 points in each class
Panel (c): plot of posterior distribution $w \sim p(w|y)$
Panel (d): predictive distribution $P(Y_* = 1|y, x_*)$ plotted as a function of x_*

Gaussian process classifier (summary of example from book Rasmussen, Williams [RW], p. 61))
$Y \in \{1, -1\}$, $X \in \mathbb{R}^2$, logistic link function:

$$P(Y = 1|X = x) = l(f(x))$$

Gaussian process prior on function $f : \mathbb{R}^2 \to \mathbb{R}$
covariance function: squared exponential kernel: for $x_1, x_2 \in \mathbb{R}^2$

$$\operatorname{cov}\left(f(x_1), f(x_2)\right) = K(x_1, x_2) = \sigma_f^2 \exp\left(-\frac{1}{2\ell}\left\|x_1 - x_2\right\|^2\right)$$

σ_f^2, ℓ hyperparameters
choices $\sigma_f^2 = 9$, ℓ varying (see below)
Panel (a): training set on $[0,1]^2$, 10 points in each class
Panel (b): predictive distribution $P(Y_* = 1|y, x_*)$ plotted as a function of x_*, for $\ell = 0,1$
Panel (c): same for $\ell = 0,2$
Panel (d): same for $\ell = 0,3$
Interpretation: higher values of ℓ mean more correlation (for fixed x_1, x_2, the value $K(x_1, x_2)$ increases in ℓ), hence less oscillation in f is assumed a priori. Thus less oscillation of f is also expressed in the posterior distribution for f, and the decision boundary is less wiggly.

7 Support vector machines

Suppose again that $S = \{(x_i, y_i), i = 1, \ldots, n\}$ is a training set where X_i are $\mathbb{R}^d$-valued and $Y_i \in \{-1, 1\}$. Assume the training set is linearly separable, i.e. separable by a hyperplane. Let a hyperplane in $\mathbb{R}^d$ be given by (b, b_0) where $b \in \mathbb{R}^d$ and $b_0 \in \mathbb{R}$, by the equation

$$\langle b, x\rangle + b_0 = 0,\ x \in \mathbb{R}^d \tag{73}$$

where $\langle b, x\rangle = b^\top x$ is the scalar product. Recall that for any scalar λ, the pairs (b, b_0) and $(\lambda b, \lambda b_0)$ define the same hyperplane. Under the assumption $\|b\| = 1$ the correspondence (b, b_0) to hyperplanes is one-to-one.

Recall that a training set $(x_1, y_1), \ldots, (x_n, y_n)$ is called *linearly separable* if there is a hyperplane given by (b, b_0) such that all x_i with $y_i = 1$ are on one side of the hyperplane, while all x_i with $y_i = -1$ are on the other side (in the strict sense, not *on* the hyperplane). Obviously we may assume $\|b\| = 1$. Thus a training set is linarly separable if there exists a pair (b, b_0) with $\|b\| = 1$ defining a hyperplane such that all

$$\gamma_i = y_i \left(\langle b, x_i \rangle + b_0\right)$$

are nonzero and have the same sign. Note that if we find a hyperplane such that all $\gamma_i < 0$, we can simply take $(-b, -b_0)$, and for this hyperplane we have all $\gamma_i > 0$. Then we defined the *geometric margin* of (x_i, y_i) with respect to (b, b_0) ($\|b\| = 1$) by γ_i, assuming all $\gamma_i > 0$. This is just the distance of a correctly classified x_i to the hyperplane. Then we call $\gamma = \min_{1 \le i \le n} \gamma_i$ the *geometric margin of the hyperplane* wrt the training set S.

Idea of the support vector classifier: *find a hyperplane which has maximal geometric margin for a given (linearly separable) training set S. For any hyperplane, the points x_i which attain $\gamma_i = \gamma$, i.e. which have minimal distance to the hyperplane are called the **support vectors**. The method to find a maximal margin hyperplane, to be described, centrally utilizes the concept of support vectors.*

Consider any separating hyperplane given by a pair (b, b_0) ($\|b\| = 1$). Consider $a = b/\gamma$ and $a_0 = b_0/\gamma$; the hyperplane given by the pair (a, a_0) is the same, but now $y_i \left(\langle a, x_i \rangle + a_0\right) \ge 1$ for all i. Thus we have shown

7.1 Lemma *The data $(x_1, y_1), \ldots, (x_n, y_n)$ are linearly separable if and only if there exists a hyperplane $\{x : H(x) = 0\}$ where*

$$H(x) = \langle a, x_i \rangle + a_0$$

such that

$$y_i H(x_i) \ge 1,\ i = 1, \ldots, n$$

7.2 Proposition *The hyperplane*

$$\hat{H}(x) = \langle \hat{a}, x_i \rangle + \hat{a}_0$$

that separates the data and maximizes the margin γ is given by solving the problem

$$\min \|a\|^2 \text{ subject to } y_i H(x_i) \ge 1,\ i = 1, \ldots, n.$$

Proof. For every separating hyperplane $H(x) = \langle b, x_i \rangle + b_0$ with $\|b\| = 1$ let γ_{b,b_0} be its margin. Let $a = b/\gamma_{b,b_0}$ and $a_0 = b_0/\gamma_{b,b_0}$; then the hyperplane given by $H(x) = \langle a, x_i \rangle + a_0$ is the same and fulfills $y_i H(x_i) \ge 1$, and $\|a\| = 1/\gamma_{b,b_0}$. Thus maximizing γ_{b,b_0} under $\|b\| = 1$ is equivalent to minimizing $\|a\|$ under $y_i H(x_i) \ge 1$, $i = 1, \ldots, n$. ■

The problem of finding a maximal margin hyperplane now turns out to be a **convex optimization problem.** In that connection, let us first discuss the *Kuhn-Tucker theorem* from convex optimization. Suppose we want to minimize a function f_0:

$$f_0(x) \to \text{minimum on } x \in A,\ A \in \mathbb{R}^d$$
$$\text{subject to constraints } f_k(x) \leq 0,\ k = 1, \dots, m.$$

where A is a convex set and all f_k are convex functions also. Consider a Lagrange function

$$L = L(x, \lambda_0, \lambda) = \sum_{k=0}^{m} \lambda_k f_k(x)$$

and denote $\lambda = (\lambda_1, \dots, \lambda_m)$.

7.3 Theorem *(Kuhn-Tucker) If x^* minimizes the function $f_0(x)$ under constaints $x \in A$ and $f_k(x) \leq 0$, $k = 1, \dots, m$ then there exist Lagrange multipliers λ_0^* and $\lambda^* = (\lambda_1^*, \dots, \lambda_m^*)$ (not all 0) such that*
(a) minimum principle:

$$\min_{x \in A} L(x, \lambda_0^*, \lambda^*) = L(x^*, \lambda_0^*, \lambda^*)$$

(b) Nonnegativity conditions

$$\lambda_k^* \geq 0,\ k = 0, \dots, m$$

(c) Kuhn-Tucker-conditions

$$\lambda_k^* f_k(x^*) = 0,\ k = 1, \dots, m.$$

If $\lambda_0^* \neq 0$ then (a), (b), (c) are also sufficient for x^* to be the solution of the optimization problem. For $\lambda_0^* \neq 0$ to hold it is sufficient that the Slater conditions *are satisfied: there exist $\bar{x}$ such that*

$$f_k(\bar{x}) < 0,\ k = 1, \dots, m.$$

7.4 Corollary a) *If the Slater condition is satisfied, then one can choose $\lambda_0^* = 1$ and rewrite the Langrangian*

$$L(x, 1, \lambda) = f_0(x) + \sum_{k=1}^{m} \lambda_k f_k(x)$$

b) *If $\lambda_0^* = 1$ then the Kuhn-Tucker theorem implies that (x^*, λ^*) defines a saddlepoint of the Lagrangian:*

$$\min_{x \in A} L(x, 1, \lambda^*) = L(x^*, 1, \lambda^*) = \max_{\lambda \geq 0} L(x^*, 1, \lambda) \tag{74}$$

(where $\lambda \geq 0$ means $\lambda_1 \geq 0, \dots, \lambda_n \geq 0$).

Proof. a) If the $\lambda_0^*, \lambda_1^*, \ldots, \lambda_m^*$ fulfill (a), (b), (c) of the Kuhn-Tucker theorem and $\lambda_0^* \neq 0$, then $1, \lambda_1^*/\lambda_0^*, \ldots, \lambda_m^*/\lambda_0^*$ also fulfill (a), (b), (c), in particular (a) which concerns minimization in x.
b) The first equality in (74) is claim (a) of the Kuhn-Tucker theorem. For the second equality, note

$$\begin{aligned} L(x^*, 1, \lambda^*) &= f_0(x^*) + \sum_{k=1}^m \lambda_k^* f_k(x^*) = f_0(x^*) \text{ (by Kuhn-Tucker conditions (c)} \\ &\geq f_0(x^*) + \sum_{k=1}^m \lambda_k f_k(x^*) \end{aligned}$$

since $\lambda_k \geq 0$ and $f_k(x^*) \leq 0$, $k = 1, \ldots, m$ (x^* maximizes $f_0(x)$ under the constraints, hence fulfills the constraints). ■

A trivial example. Consider minimization of the function $f_0(x) = x^2$ over $A = \mathbb{R}$ with constraint $f_1(x) = (x-a)^2 - 1 \leq 0$ where $a \geq 0$. By making a picture, we see immediately: if $a > 1$ then the solution is $x^* = a - 1$ and if $0 \leq a \leq 1$ then $x^* = 0$. Let us derive that solution from the Kuhn-Tucker theorem.
First, by a basic reasoning we find that a minimizer x^* under the constraint $f_1(x) \leq 0$ must exist. The Slater condition is satisfied: take $\bar{x} = a$; then $f_1(\bar{x}) = -1 < 0$. Hence there exists λ^* such that (x^*, λ^*) satisfy (a), (b), (c) with $\lambda_0^* = 1$. Now (a) means

$$\min_{x \in \mathbb{R}} \left(x^2 + \lambda^* \left((x-a)^2 - 1 \right) \right) = \left(x^{*2} + \lambda^* \left((x^* - a)^2 - 1 \right) \right),$$

hence by considering the behaviour of the target function at $x \to \pm\infty$ and taking a derivative, we see that x^* fulfills

$$2x^* + 2\lambda^* (x^* - a) = 0 \tag{75}$$

which implies

$$\lambda^* = \frac{x^*}{a - x^*}. \tag{76}$$

Case 1. $\lambda^* > 0$. Then by (c) we have $f_1(x^*) = 0 = (x^* - a)^2 - 1$, hence $x^* = a \pm 1$. If $x^* = a + 1 > 0$ then by (76) $\lambda^* = -1 - a < 0$ (since $a > 0$) which contradicts $\lambda^* \geq 0$ from (b). Hence we must have $x^* = a - 1$, and from (76) we obtain $\lambda^* = a - 1$. By assumption $\lambda^* > 0$ this implies $a > 1$.
Case 2. $\lambda^* = 0$. From (75) we obtain $x^* = 0$. Since x^* fulfills the constraint, we obtain

$$f_1(x^*) = a^2 - 1 \leq 0$$

hence $a \leq 1$.
We have thus reproduced the solution: if $a > 1$ then $x^* = a - 1$ (Case 1) and if $0 \leq a \leq 1$ then $x^* = 0$ (Case 2). ■

Let us return to the problem of finding a maximal margin hyperplane, i.e. to the problem formulated in Proposition 7.2. The problem is to minimize, in $a \in \mathbb{R}^d$ and real valued a_0 the function

$$f_0(a, a_0) = \frac{1}{2} \|a\|^2$$

subject to restrictions $y_i (\langle a, x_i \rangle + a_0) \geq 1$, $i = 1, \ldots, n$. Here (x_i, y_i), $i = 1, \ldots, n$ is the training set where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. The restrictions can be written

$$f_k(a, a_0) = 1 - y_k (\langle a, x_k \rangle + a_0) \leq 0, k = 1, \ldots, n.$$

7.5 Theorem *Let*

$$\hat{H}(x) = \langle \hat{a}, x \rangle + \hat{a}_0$$

be the optimal (largest margin) hyperplane. Then

$$\hat{a} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i$$

and $\hat{\alpha} = (\hat{\alpha}_1, \ldots, \hat{\alpha}_n)$ *is the vector that maximizes*

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle$$

subject to

$$\alpha_i \geq 0,\ i = 1, \ldots, n \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0.$$

The ***support vectors*** *(i.e. the* x_i *attaining* $y_i \hat{H}(x_i) = 1$*) are the points* x_i *where* $\hat{\alpha}_i \neq 0$. *The value* $\hat{a}_0$ *can be found by solving*

$$y_i (\langle \hat{a}, x_i \rangle + \hat{a}_0) = 1$$

for any support vector x_i.

Comments.

1. The α_i are Lagrange multipliers, the problem is a dual convex optimization problem where maximization is in α, not the original a, a_0 (the parameters of the hyperplane).

2. Moreover, vectors from the training set x_i which are known not to be support vectors can be excluded from the problem (since their α_i must be 0). Thus the dimensionality of the problem can be reduced considerably, and iterative algorithms can be devised.

3. The vectors x_i enter only through their scalar products $\langle x_i, x_j \rangle$. This allows computational simplification and also kernelization.

Exercise. Consider a linearly separable training set in the case $d = 1$. Then the support vector classifier should find the maximal margin split, i.e. the point where all x_i with $y_i = 1$ are one one side and all x_i with $x_i = -1$ are on the other, and which is exactly in the middle between the two adjacent points (these must be the support vectors). Specifically, assume $x_i \in \mathbb{R}$, $i = 1, \ldots, n$ and $y_i = -1$, $i = 1, \ldots, m$ while $y_i = 1$, $i = m+1, \ldots, n$. Then the support vector classifier should split at the point $(x_{m+1} + x_m)/2$, the maximal geometric margin is $\gamma = (x_{m+1} - x_m)/2$, and the support vectors are x_{m+1}, x_m (see below for a detailed treatment). We will treat this in detail in Example 1 below.

Proof of the theorem. The basis is the Kuhn-Tucker theorem, Theorem 7.3. The function $f_0(a, a_0)$ is convex in $(a, a_0) \in \mathbb{R}^{d+1}$ and all functions $f_k(a, a_0)$, $k = 1, \ldots, n$ are linear in (a, a_0), hence also convex. The convex set A is taken to be $\mathbb{R}^{d+1}$. The Slater conditions are satisfied: there exist $(\bar{a}, \bar{a}_0)$ such that

$$y_i \left(\langle \bar{a}, x_i \rangle + \bar{a}_0 \right) > 1, \; i = 1, \ldots, n$$

since for any separating hyperplane (b, b_0) with margin γ we can take $(\bar{a}, \bar{a}_0) = \frac{2}{\gamma}(b, b_0)$ so that $Y_i \left(\langle \bar{a}, X_i \rangle + \bar{a}_0 \right) \geq 2$. For the Lagange multipliers in Theorem 7.3 we now write $\lambda_i = \alpha_i$, $i = 1, \ldots, n$. The Lagrangian then is

$$L(a, a_0, \alpha) = f_0(a, a_0) + \sum_{k=1}^{n} \lambda_k f_k(a, a_0) =$$
$$= \frac{1}{2} \langle a, a \rangle - \sum_{i=1}^{n} \alpha_i \left(y_i \left(\langle a, x_i \rangle + a_0 \right) - 1 \right).$$

Theorem 7.3 now asserts that a necessary and sufficient condition for $\hat{a}, \hat{a}_0$ to be minimizers of $f_0(a, a_0)$ subject to the restrictions $f_k(a, a_0) \leq 0$ is the existence of $\hat{\alpha} = (\hat{\alpha}_1, \ldots, \hat{\alpha}_n)$ such that conditions (a), (b) and (c) are fulfilled. It is easy to see that a minimizer $\hat{a}, \hat{a}_0$ exists (i.e a maximum margin hyperplane; use standard results about minimizing a squared norm over a closed convex domain). Hence a corresponding $\hat{\alpha} = (\hat{\alpha}_1, \ldots, \hat{\alpha}_n)$ does exist; let us specify what conditions (a), (b) and (c) mean in this case.
Condition (a) of the Kuhn-Tucker theorem (minimum principle) implies that partial derivatives wrt to a and a_0 are 0, hence

$$\frac{\partial}{\partial a} L(a, a_0, \alpha) \left.\right|_{\hat{a}, \hat{a}_0, \hat{\alpha}} = 0 = \hat{a} - \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i$$

hence

$$\hat{a} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i. \tag{77}$$

Furthermore

$$\frac{\partial}{\partial a_0} L(a, a_0, \alpha) \left.\right|_{\hat{a}, \hat{a}_0, \hat{\alpha}} = 0 = \sum_{i=1}^{n} \hat{\alpha}_i y_i \tag{78}$$

Condition (b) (nonnegativity) means

$$\hat{\alpha}_i \geq 0,\ i = 1, \ldots, n$$

and condition (c) (Kuhn-Tucker-conditions)

$$\hat{\alpha}_i \left(y_i \left(\langle \hat{a}, x_i \rangle + \hat{a}_0 \right) - 1 \right) = 0,\ \ i = 1, \ldots, n$$

where $\hat{\alpha}_i > 0$ only for the support vectors x_i of the optimal hyperplane $\hat{a}, \hat{a}_0$ (i.e. for x_i fulfilling $y_i \left(\langle \hat{a}, x_i \rangle + \hat{a}_0 \right) = 1$). Now write the Lagrangian as

$$\begin{aligned} L\left(\hat{a}, \hat{a}_0, \hat{\alpha}\right) &= \frac{1}{2} \langle \hat{a}, \hat{a} \rangle - \sum_{i=1}^{n} \hat{\alpha}_i \left(y_i \left(\langle \hat{a}, x_i \rangle + \hat{a}_0 \right) - 1 \right) \\ &= \frac{1}{2} \langle \hat{a}, \hat{a} \rangle - \sum_{i=1}^{n} \hat{\alpha}_i y_i \langle \hat{a}, x_i \rangle - \hat{a}_0 \sum_{i=1}^{n} \hat{\alpha}_i y_i + \sum_{i=1}^{n} \hat{\alpha}_i. \end{aligned}$$

The third term on the right vanishes in view of (78). Using (77) to expand $\hat{a}$ in the first and second terms yields

$$L\left(\hat{a}, \hat{a}_0, \hat{\alpha}\right) =$$

$$\begin{aligned} &= \frac{1}{2} \sum_{i,j=1}^{n} \hat{\alpha}_i \hat{\alpha}_j y_i y_j \langle x_i, x_j \rangle - \sum_{i,j=1}^{n} \hat{\alpha}_i \hat{\alpha}_j y_i y_j \langle x_i, x_j \rangle + \sum_{i=1}^{n} \hat{\alpha}_i \\ &= \sum_{i=1}^{n} \hat{\alpha}_i - \frac{1}{2} \sum_{i,j=1}^{n} \hat{\alpha}_i \hat{\alpha}_j y_i y_j \langle x_i, x_j \rangle =: W(\hat{\alpha}) \end{aligned}$$

i. e. we denote $W(\alpha)$ the resulting functional of α. We claim

$$W(\hat{\alpha}) \geq W(\alpha) \text{ for all } \alpha \text{ satisfying } \sum_{i=1}^{n} \alpha_i y_i = 0 \text{ and } \alpha_i \geq 0,\ i = 1, \ldots, n. \tag{79}$$

Indeed, according to Corollary 7.4 to the Kuhn-Tucker theorem about the saddlepoint of the Lagrange function we have

$$W(\hat{\alpha}) = L\left(\hat{a}, \hat{a}_0, \hat{\alpha}\right) \geq L\left(\hat{a}, \hat{a}_0, \alpha\right)$$

for all α with nonnegative components. Define a function of $a \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}^n$

$$L_0\left(a, \alpha\right) := \frac{1}{2} \langle a, a \rangle - \sum_{i=1}^{n} \alpha_i y_i \langle a, x_i \rangle + \sum_{i=1}^{n} \alpha_i.$$

For all α satisfying $\sum_{i=1}^{n} \alpha_i y_i = 0$ we have

$$L\left(\hat{a}, \hat{a}_0, \alpha\right) = \frac{1}{2} \langle \hat{a}, \hat{a} \rangle - \sum_{i=1}^{n} \alpha_i y_i \langle \hat{a}, x_i \rangle + \sum_{i=1}^{n} \alpha_i = L_0\left(\hat{a}, \alpha\right)$$

Let $\tilde{a}_\alpha$ be the minimizer of $L_0(a,\alpha)$ in a for given α; since $L_0(a,\alpha)$ is quadratic in a, we obtain by taking a derivative, analogously to (77),

$$\tilde{a}_\alpha = \sum_{i=1}^n \alpha_i y_i x_i.$$

Hence

$$L_0(\hat{a},\alpha) \geq L_0(\tilde{a}_\alpha,\alpha) = W(\alpha)$$

so that (79) is proved. ■

Example 1. Consider a linearly separable training set in the case $d = 1$. Then the support vector classifier should find the maximal margin split, i.e. the point where all x_i with $y_i = 1$ are one one side and all x_i with $x_i = -1$ are on the other, and which is exactly in the middle between the two adjacent points (these must be the support vectors). Specifically, assume that the real numbers x_i are ordered: $x_1 < \ldots < x_n$ and $y_i = -1$, $i = 1,\ldots,m$ while $y_i = 1$, $i = m+1,\ldots,n$. Then the support vector classifier should split at the point $(x_{m+1} + x_m)/2$, the maximal geometric margin is $\gamma = (x_{m+1} - x_m)/2$, and the support vectors are x_{m+1}, x_m. Formally we may write that the best "separating hyperplane" with a "unit vector" $b = 1$ fulfills

$$y_i(bx_i + b_0) \geq \gamma \tag{80}$$

Indeed for $b_0 = -(x_{m+1} + x_m)/2$ we have

$$\begin{aligned} y_m(bx_m + b_0) &= (-1)(x_m - (x_{m+1} + x_m)/2) \\ &= (-1)((x_m - x_{m+1})/2) = \gamma \end{aligned}$$

and similarly

$$\begin{aligned} y_{m+1}(bx_{m+1} + b_0) &= (+1)(x_{m+1} - (x_{m+1} + x_m)/2) \\ &= (x_{m+1} - x_m)/2 = \gamma \end{aligned}$$

and for all other x_i we clearly have $y_i(bx_i + b_0) > \gamma$. Thus indeed x_{m+1}, x_m are the support vectors. Reformulating (80) using $a = b/\gamma$, $a_0 = b_0/\gamma$ such that

$$y_i(ax_i + a_0) \geq 1$$

we obtain

$$a = \frac{1}{\gamma} = \frac{2}{x_{m+1} - x_m},\ a_0 = \frac{b_0}{\gamma} = -\frac{x_{m+1} + x_m}{x_{m+1} - x_m}. \tag{81}$$

Let us check whether these values are also obtained as solutions of the dual optimization problem for SVM of Theorem 7.5. In the notation, we omit the hat from the solutions $\hat{a}, \hat{\alpha}_i$. First, the optimal a is $a = \sum_{i=1}^n \alpha_i y_i x_i$ for the solutions α_i. Since only x_m, x_{m+1} are support vectors, only

α_m, α_{m+1} are nonzero. Moreover, the optimal α_i fulfill $\sum_{i=1}^n \alpha_i y_i = 0$ hence $-\alpha_m + \alpha_{m+1} = 0$, which implies $\alpha_{m+1} = \alpha_m$. Let us denote α the common value of α_{m+1}, α_m. Then

$$a = \alpha \left(x_{m+1} - x_m\right). \tag{82}$$

Let us find α from maximizing

$$\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \left\langle x_i, x_j \right\rangle, \text{ subject to } \alpha_i \geq 0,\ i = 1, \ldots, n \text{ and } \sum_{i=1}^n \alpha_i y_i = 0. \tag{83}$$

This now becomes

$$2\alpha - \frac{1}{2}\alpha^2 \sum_{i,j=m}^{m+1} y_i y_j x_i x_j = 2\alpha - \frac{1}{2}\alpha^2 \left(x_{m+1} - x_m\right)^2 = 2\alpha - 2\alpha^2\gamma^2$$
$$\text{subject to } \alpha \geq 0.$$

The maximizing α without restrictions is given by taking a derivative:

$$2 - 4\alpha\gamma^2 = 0 \text{ hence } \alpha = \frac{1}{2\gamma^2}.$$

Since this solution fulfills $\alpha \geq 0$, it is the solution also for the restricted problem. Now plugging this α into (82) yields

$$a = \alpha \left(x_{m+1} - x_m\right) = \frac{\left(x_{m+1} - x_m\right)}{2\gamma^2} = \frac{\gamma}{\gamma^2} = \frac{1}{\gamma}$$

i.e. (81) for a is obtained. For a_0 we may solve $y_i \left(\langle a, x_i \rangle + a_0\right) = 1$ for any support vector x_i. Taking $i = m + 1$ we obtain

$$\begin{aligned} x_{m+1}/\gamma + a_0 = 1,& \\ a_0 &= 1 - \frac{x_{m+1}}{\gamma} = \frac{\gamma - x_{m+1}}{\gamma} = \frac{-x_{m+1}/2 - x_m/2}{\gamma} \\ &= -\frac{x_{m+1} + x_m}{x_{m+1} - x_m} \end{aligned}$$

i.e. we have reproduced (81) for a_0.

Example 2. Suppose in the general $\mathbb{R}^d$ space we have only two points: x_1 with $y_1 = -1$ and x_2 with $y_2 = 1$. Then the optimal a is given by $a = \sum_{i=1}^n \alpha_i y_i x_i = \alpha_2 x_2 - \alpha_1 x_1$. Since again $\sum_{i=1}^n \alpha_i y_i = 0$, we have $\alpha_1 = \alpha_2 =: \alpha$ and $a = \alpha \left(x_2 - x_1\right)$, which means that the maximal margin hyperplane is perpendicular to the vector $z := x_2 - x_1$. Similarly to the above reasoning we now have to maximize (83) which means for $\gamma = \|z\|/2$

$$2\alpha - \frac{1}{2}\alpha^2 \sum_{i,j=1}^2 y_i y_j \left\langle x_i, x_j \right\rangle = 2\alpha - \frac{1}{2}\alpha^2 \|z\|^2 = 2\alpha - 2\alpha^2\gamma^2.$$

We obtain $\alpha = 1/2\gamma^2$ as before, hence $a = \alpha z = z/2\gamma^2 = z/\|z\|\gamma = 2z/\|z\|^2$ and obtain a_0 from

$$\begin{aligned} y_2\left(\langle a, x_2\rangle + a_0\right) &= 1, \\ a_0 = 1 - 2\langle z, x_2\rangle / \|z\|^2 &= \frac{\langle z, z\rangle - 2\langle z, x_2\rangle}{\|z\|^2} \\ &= -\frac{\langle z, x_2 + x_1\rangle}{\|z\|^2} = -\frac{\langle x_2 - x_1, x_2 + x_1\rangle}{\|x_2 - x_1\|^2}. \end{aligned}$$

7.1 The nonseparable case

Recall that in the separable case, the margin of a separating hyperplane $\langle b, x\rangle + b_0 = 0$ with unit vector b ($\|b\| = 1$) is

$$\gamma = \min_{1\le i\le n} y_i\left(\langle b, x_i\rangle + b_0\right)$$

and the optimization problem for the maximum margin hyperplane can be written

$$\begin{gathered} \max_{b, b_0, \|b\|=1} \gamma \\ \text{subject to } y_i\left(\langle b, x_i\rangle + b_0\right) \ge \gamma,\ i = 1, \ldots, n. \end{gathered} \tag{84}$$

We saw (dividing the conditions by γ and replacing b by $a = b/\gamma$, b_0 by $a_0 = b_0/\gamma$) that this problem is equivalent to

$$\begin{gathered} \min_{a, a_0} \|a\| \\ \text{subject to } y_i\left(\langle a, x_i\rangle + a_0\right) \ge 1,\ i = 1, \ldots, n. \end{gathered}$$

Assume that we relax the condition (84) in such a way that for certain i, we allow

$$y_i\left(\langle b, x_i\rangle + b_0\right) \ge \gamma(1 - \xi_i)$$

for certain $\xi_i \ge 0$. In the separable case, and if $\xi_i < 1$, if that means that these x_i are allowed to be closer to the margin than γ, but still on the "right" side of the hyperplane. (In this case γ is no longer the margin, but can be seen as something like a "soft margin". Recall that $y_i\left(\langle b, x_i\rangle + b_0\right)$ for a unit vector b is the distance of x_i to the hyperplane). We may also allow $\xi_i \ge 1$, thus allowing misclassification. In this way we may be able to treat the linearly nonseparable case where some misclassification is inevitable.
In this approach it is natural to impose a restriction, in addition to $\xi_i \ge 0$

$$\sum_{i=1}^{n} \xi_i \le B \tag{85}$$

which represents a certain "budget" which one may spend in allowing x_i to be closer to the hyperplane or to be misclassified. The new variables ξ_i are called *slack variables.* It will be part of the optimization problem to determine which ξ_i are 0 and which are > 0.

In the same way as before we see that the new optimization problem can equivalently be formulated

$$\min_{a,a_0} \; \|a\|^2 \tag{86}$$
$$\text{subject to } y_i\left(\langle a, x_i\rangle + a_0\right) \geq (1-\xi_i),\; i=1,\ldots,n$$
$$\xi_i \geq 0, \quad i=1,\ldots,n$$
$$\sum_{i=1}^{n} \xi_i \leq B.$$

Let us consider a related problem: for a $\delta > 0$

$$\min_{a,a_0,\xi} \; \left(\frac{1}{2}\|a\|^2 + \delta\sum_{i=1}^{n}\xi_i\right) \tag{87}$$
$$\text{subject to } y_i\left(\langle a, x_i\rangle + a_0\right) \geq (1-\xi_i),\; i=1,\ldots,n$$
$$\xi_i \geq 0, \quad i=1,\ldots,n.$$

Here we omitted the constraint $\sum_{i=1}^{n}\xi_i \leq B$, but we added a "penalty term" to the objective function, which is weighted with a "weight" δ. On p. 420 (top) of the book Hastie et al. [HTF] it is claimed that the two previous problems are equivalent in some sense. This can be justified from the Kuhn-Tucker Theorem. For our optimal hyperplane problem, we note that the optimization problem (86) may not have a solution (i.e. there are no a, a_0, ξ satisfying the constraints), if the training set is "highly nonseparable" in the sense that the given bound B does not suffice to accomodate all the inevitable misclassifications. On the other hand, the problem (87) always has a solution for any $\delta > 0$; because for "highly nonseparable" data sets, the ξ_i are allowed to take large values as necessary, but large ξ_i are penalized by the term in the objective function $\delta\sum_{i=1}^{n}\xi_i$. Thus the optimization problem (87) is more flexible; then δ is a tuning parameter which specifies how much we penalize "slack" with regard to the margin or misclassification. The case $\delta = \infty$ corresponds to the hard margin case for a separable data set, where we try to prohibit any slack with regard to the margin and thus find the maximal margin hyperplane. In the initial problem (86) B would be such a tuning parameter which we have to choose first, but in the nonseparable case B would first have to be large enough.
Let us formulate the analog of the previous result on SVM for the soft margin case.

7.6 Theorem *Let $\hat{H}(x) = \hat{a}_0 + \langle \hat{a}, x \rangle$ denote the optimal hyperplane in the sense of solving optimization problem (87) for a given tuning parameter $\delta > 0$. Then*

$$\hat{a} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i$$

(sum of vectors), where $\hat{\alpha} = (\hat{\alpha}_1, \ldots, \hat{\alpha}_n)$ is the vector that maximizes

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle$$

subject to

$$0 \le \alpha_i \le \delta,\; i = 1, \ldots, n,$$
$$\sum_{i=1}^{n} \alpha_i y_i = 0.$$

The points x_i for which $\hat{\alpha}_i \neq 0$ are the ***support vectors****. Then the value $\hat{a}_0$ is determined from*

$$y_i \left(\langle \hat{a}, x_i \rangle + \hat{a}_0 \right) = 1$$

for any i such that $0 < \hat{\alpha}_i < \delta$, and the $\hat{\xi}_i$ are determined from

$$y_i \left(\langle \hat{a}, x_i \rangle + \hat{a}_0 \right) = 1 - \hat{\xi}_i$$

for all support vectors, and $\hat{\xi}_i = 0$ for other values of i.

Proof. The basis is again the Kuhn-Tucker theorem, Theorem 7.3. Let $\xi = (\xi_1, \ldots, \xi_n)$; the function $f_0(a, a_0, \xi)$

$$f_0(a, a_0, \xi) = \frac{1}{2} \|a\|^2 + \delta \sum_{i=1}^{n} \xi_i$$

is convex in $(a, a_0, \xi) \in \mathbb{R}^{d+1+n}$. Define functions

$$f_k(a, a_0, \xi) = (1 - \xi_k) - y_k \left(\langle a, x_k \rangle + a_0 \right),\; k = 1, \ldots, n,$$
$$f_{n+k}(a, a_0, \xi) = -\xi_k,\; k = 1, \ldots, n;$$

then all functions $f_k(a, a_0, \xi)$, $k = 1, \ldots, 2n$ are linear in (a, a_0, ξ), hence convex. The convex set A is taken to be $\mathbb{R}^{d+1+n}$.
For the Slater conditions, consider any $(\bar{a}, \bar{a}_0)$ and

$$\mu = \min \left\{ 0, y_i \left(\langle \bar{a}, x_i \rangle + \bar{a}_0 \right), i = 1, \ldots, n \right\} \tag{88}$$

then $\mu \leq 0$ with possibly large $|\mu|$ since the data may be highly nonseparable. Then take $\bar{\xi}_i = 1 - \mu + \varepsilon$, $\varepsilon > 0$, $i = 1, \ldots, n$; then $\mu > 1 - \bar{\xi}_i$ and $\bar{\xi}_i > 0$. This implies

$$\begin{aligned} f_k(\bar{a}, \bar{a}_0, \bar{\xi}) &= \left(1 - \bar{\xi}_k\right) - y_k\left(\langle a, x_k\rangle + a_0\right) \leq \left(1 - \bar{\xi}_k\right) - \mu < 0, \\ f_{k+n}(\bar{a}, \bar{a}_0, \bar{\xi}) &= -\bar{\xi}_k < 0, \end{aligned}$$

for $k = 1, \ldots, n$, thus the Slater conditions are satisfied.
Introduce Lagrange multipliers α_i, β_i, $i = 1, \ldots, n$ and write the Lagrangian

$$L\left(a, a_0, \xi, \alpha, \beta\right) =$$

$$= \frac{1}{2}\langle a, a\rangle + \delta \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i \left(y_i\left(\langle a, x_i\rangle + a_0\right) - (1 - \xi_i)\right) - \sum_{i=1}^{n} \beta_i \xi_i. \tag{89}$$

Let us specify what conditions (a), (b) and (c) of Theorem 7.3 mean in this case.
Condition (a) of the Kuhn-Tucker theorem (minimum principle) implies that partial derivatives wrt to a, a_0 and ξ are 0, hence

$$\frac{\partial}{\partial a} L\left(a, a_0, \xi, \alpha, \beta\right)|_{\hat{a}, \hat{a}_0, \hat{\xi}, \hat{\alpha}, \hat{\beta}} = 0 = \hat{a} - \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i$$

hence

$$\hat{a} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i. \tag{90}$$

Furthermore

$$\frac{\partial}{\partial a_0} L\left(a, a_0, \xi, \alpha, \beta\right)|_{\hat{a}, \hat{a}_0, \hat{\xi}, \hat{\alpha}, \hat{\beta}} = 0 = \sum_{i=1}^{n} \hat{\alpha}_i y_i \tag{91}$$

and

$$\frac{\partial}{\partial \xi_i} L\left(a, a_0, \xi, \alpha, \beta\right)|_{\hat{a}, \hat{a}_0, \hat{\xi}, \hat{\alpha}, \hat{\beta}} = 0 = \delta - \alpha_i - \beta_i = 0,\ i = 1, \ldots, n \tag{92}$$

Condition (b) (nonnegativity) means

$$\hat{\alpha}_i \geq 0,\ \hat{\beta}_i \geq 0,\ i = 1, \ldots, n$$

and condition (c) (Kuhn-Tucker-conditions)

$$\hat{\alpha}_i \left(y_i\left(\langle \hat{a}, x_i\rangle + \hat{a}_0\right) - (1 - \xi_i)\right) = 0, \tag{93}$$

$$\hat{\beta}_i \hat{\xi}_i = 0,\ \ i = 1, \ldots, n \tag{94}$$

where $\hat{\alpha}_i > 0$ only for the support vectors. We can eliminate $\hat{\beta}_i$ using (92) and $\hat{\beta}_i \geq 0$: this together yields

$$\hat{\alpha}_i + \hat{\beta}_i = \delta\ ,\ \beta_i \geq 0 \tag{95}$$

which implies a condition for α_i

$$0 \leq \hat{\alpha}_i \leq \delta.$$

The remainder of the reasoning is as before in the proof of Theorem 7.5. For the Lagrangian at the saddlepoint $\left(\hat{a}, \hat{a}_0, \hat{\xi}, \hat{\alpha}, \hat{\beta}\right)$ we find (first rearranging only the terms in (89))

$$L\left(\hat{a}, \hat{a}_0, \hat{\xi}, \hat{\alpha}, \hat{\beta}\right) = \frac{1}{2}\langle \hat{a}, \hat{a}\rangle - \sum_{i=1}^{n} \hat{\alpha}_i \left(y_i \left(\langle \hat{a}, x_i\rangle + \hat{a}_0\right) - 1\right) + \delta \sum_{i=1}^{n} \hat{\xi}_i - \sum_{i=1}^{n} \hat{\alpha}_i \hat{\xi}_i - \sum_{i=1}^{n} \hat{\beta}_i \hat{\xi}_i. \tag{96}$$

To simplify that, note relation (95) which eliminates all terms containing $\hat{\xi}_i$. Condition (91) eliminates $\hat{a}_0$, so that

$$L\left(\hat{a}, \hat{a}_0, \hat{\xi}, \hat{\alpha}, \hat{\beta}\right) = \frac{1}{2}\langle \hat{a}, \hat{a}\rangle - \sum_{i=1}^{n} \hat{\alpha}_i y_i \langle \hat{a}, x_i\rangle + \sum_{i=1}^{n} \hat{\alpha}_i. \tag{97}$$

Substituting $\hat{a}$ according to (90) yields

$$L\left(\hat{a}, \hat{a}_0, \hat{\xi}, \hat{\alpha}, \hat{\beta}\right) = \sum_{i=1}^{n} \hat{\alpha}_i - \frac{1}{2}\sum_{i,j=1}^{n} \hat{\alpha}_i \hat{\alpha}_j y_i y_j \langle x_i, x_j\rangle =: W(\hat{\alpha})$$

i. e. we denote $W(\alpha)$ the resulting functional of α. We claim

$$W(\hat{\alpha}) \geq W(\alpha) \text{ for all } \alpha \text{ satisfying } \sum_{i=1}^{n} \alpha_i y_i = 0 \text{ and } 0 \leq \alpha_i \leq \delta,\ i = 1, \ldots, n. \tag{98}$$

Indeed, according to Corollary 7.4 to the Kuhn-Tucker theorem about the saddlepoint of the Lagrange function we have

$$W(\hat{\alpha}) = L\left(\hat{a}, \hat{a}_0, \hat{\xi}, \hat{\alpha}, \hat{\beta}\right) \geq L\left(\hat{a}, \hat{a}_0, \hat{\xi}, \alpha, \beta\right)$$

for all α, β with nonnegative components; in particular for those satisfying additionally $\alpha_i + \beta_i = \delta$, $\sum_{i=1}^{n} \alpha_i y_i = 0$. For these we have (eliminating all terms containing $\hat{a}_0$ and $\hat{\xi}_i$, as we did in (96 obtaining (97))

$$L\left(\hat{a}, \hat{a}_0, \hat{\xi}, \alpha, \beta\right) = \frac{1}{2}\langle \hat{a}, \hat{a}\rangle - \sum_{i=1}^{n} \alpha_i y_i \langle \hat{a}, x_i\rangle + \sum_{i=1}^{n} \alpha_i.$$

Define a function of $a \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}^n$

$$L_0(a, \alpha) := \frac{1}{2}\langle a, a\rangle - \sum_{i=1}^{n} \alpha_i y_i \langle a, x_i\rangle + \sum_{i=1}^{n} \alpha_i.$$

For all α, β satisfying $\sum_{i=1}^{n} \alpha_i y_i = 0$ and $\alpha_i + \beta_i = \delta$ we have

$$L\left(\hat{a}, \hat{a}_0, \hat{\xi}, \alpha, \beta\right) = L_0(\hat{a}, \alpha).$$

Let $\tilde{a}_\alpha$ be the minimizer of $L_0(a,\alpha)$ in a for given α; since $L_0(a,\alpha)$ is quadratic in a, we obtain by taking a derivative, analogously to (90),

$$\tilde{a}_\alpha = \sum_{i=1}^n \alpha_i y_i x_i.$$

Hence

$$L_0(\hat{a},\alpha) \geq L_0(\tilde{a}_\alpha,\alpha) = W(\alpha)$$

so that (98) is proved.
If $0 < \hat{\alpha}_i < 1$ then by (95) we have $\hat{\beta}_i > 0$, hence by the Kuhn-Tucker condition (94) $\hat{\xi}_i = 0$. Also in this case, by (93)

$$(y_i(\langle \hat{a}, x_i\rangle + \hat{a}_0) - (1-\xi_i)) = 0$$

hence

$$y_i(\langle \hat{a}, x_i\rangle + \hat{a}_0) - 1 = 0$$

which determines $\hat{a}_0$. To determine $\hat{\xi}_i$, for the non-support vectors ($\hat{\alpha}_i = 0$) we have $\hat{\beta}_i = 1$ and hence by (94) $\hat{\xi}_i = 0$. Also for $0 < \hat{\alpha}_i < 1$ we have $\hat{\xi}_i = 0$, as argued above. For $\hat{\alpha}_i = 1$ we have by (93)

$$y_i(\langle \hat{a}, x_i\rangle + \hat{a}_0) - \left(1 - \hat{\xi}_i\right) = 0$$

which determines $\hat{\xi}_i$. ■

On p. 422 of [HTF] a simulation example for various values of γ is discussed.
Note that support vectors in this case are all the points on the wrong side of their margin or on their margin (as opposed to the hard margin case, where they are exactly *on* their margin). More specifically, if $0 < \hat{\alpha}_i < \delta$ then $\hat{\xi}_i = 0$, and x_i is exactly *on the* (soft) margin. If $\hat{\alpha}_i = \delta$ then $\hat{\xi}_i > 0$ and x_i is *inside the* (soft) margin. If $\hat{\xi}_i > 1$ then x_i is misclassified; all these x_i are support vectors.

7.2 Kernelization

Kernelization is now straightforward: for a positive definite kernel $K(\cdot,\cdot)$ and RKHS of functions with scalar product we may now replace $\langle x_i, x_j\rangle$ in the dual optimization problem of Theorem 7.6 by $\langle K(\cdot,x_i), K(\cdot,x_j)\rangle_* = K(x_i,x_j)$. Before we discuss that in detail let us give another perspective on the minimization problem (87): write it as

$$\min_{a,a_0,\xi} \left(\frac{1}{2}\|a\|^2 + \delta\sum_{i=1}^n \xi_i\right)$$
$$\text{subject to } \xi_i \geq 1 - y_i(\langle a, x_i\rangle + a_0),\ i = 1,\ldots,n$$
$$\xi_i \geq 0, \quad i = 1,\ldots,n.$$

Obviously, for fixed a, a_0, we try to make ξ_i as small as possible given the constraints. The two constraints may be written as one:

$$\xi_i \geq \max\{0, 1 - y_i(\langle a, x_i \rangle + a_0)\} = (1 - y_i(\langle a, x_i \rangle + a_0))_+$$

where $a_+ = \max\{0, a\}$. Clearly for fixed a, a_0 the minimizing ξ_i attains the bound:

$$\xi_i = (1 - y_i(\langle a, x_i \rangle + a_0))_+$$

so that, by plugging this into the target function, we obtain a problem

$$\min_{a,a_0} \left(\frac{1}{2} \|a\|^2 + \delta \sum_{i=1}^{n} (1 - y_i(\langle a, x_i \rangle + a_0))_+ \right).$$

Set $f(x) = (\langle a, x_i \rangle + a_0)$ and recall the definition of the *soft margin loss* l_{sm} : (also called *hinge loss*)

$$l_{sm}(y, f(x)) = (1 - yf(x))_+ = \left\{ \begin{array}{l} 0 \text{ if } yf(x) > 1 \\ 1 - yf(x) \text{ otherwise} \end{array} \right. . \tag{99}$$

If $yf(x_i) > 0$ then x_i is correctly classified, if $yf(x_i) < 0$ then there is an error. We regard $|f(x_i)|$ as the "strength" of "confidence" with which the sign of $f(x_i)$ is predicted. If there is a wrong classification of x_i, then the above loss is $1 + |f(x_i)|$, i.e. the loss increases with the expressed "confidence" for that wrong prediction. If there is correct classification, $yf(x_i) > 0$, then the loss is set 0 only if $|f(x_i)| > 1$, i.e. for a high enough confidence. If $0 < |f(x_i)| < 1$ then there is still some penalty, even though x_i is correctly classified.
Now our minimization problem may be formulated

$$\min_{a,a_0} \left(\frac{1}{2} \|a\|^2 + \delta \sum_{i=1}^{n} l_{sm}(y_i, \langle a, x_i \rangle + a_0) \right).$$

or equivalently, setting $\lambda = 1/2n\delta$

$$\min_{a,a_0} \left(n^{-1} \sum_{i=1}^{n} l_{sm}(y_i, \langle a, x_i \rangle + a_0) + \lambda \langle a, a \rangle \right).$$

In kernelization, we replace the scalar product $\langle \cdot, \cdot \rangle$ by $\langle \cdot, \cdot \rangle_*$, the vector x_j is replaced $K(\cdot, x_j)$ and the linear coefficient vector a is replaced by a function f in the RKHS. Then the target function becomes

$$\begin{aligned} & n^{-1} \sum_{i=1}^{n} l_{sm}(y_i, \langle f, K(\cdot, x_i) \rangle_* + a_0) + \lambda \langle f, f \rangle_* = \\ & = n^{-1} \sum_{i=1}^{n} l_{sm}(y_i, f(x_i) + a_0) + \lambda \langle f, f \rangle_* \end{aligned} \tag{100}$$

and minimization is over f, a_0. By the representer theorem Theorem (6.2) each minimizer f of the regularized risk functional (44) admits a representation of the form

$$f(x) = \sum_{i=1}^{n} \alpha_i K(x, x_i).$$

where $\alpha_i \in \mathbb{R}$. We see that the problem coincides with the *regularization and spline smoothing problems* discussed before, where $\lambda \langle f, f \rangle_*$ may be seen as a penalty term.
Recall the least squares analog for regression; there is an explicit solution: the linear smoothing spline or regularizer. For soft margin loss, we did not describe a solution in the section about regularization; we have now obtained one in the form of the SVM, i.e. by the dual optimization program.

7.3 Support vector regression

The standard regression analog, using squared error loss, of the problem of minimizing (100) over f, a_0 would be to minimize

$$n^{-1} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \langle f, f \rangle_*$$

over f. Consider an alternative to the squared errror loss, the so-called ε-insensitive loss function:

$$l_\varepsilon(y, f(x)) := (|y - f(x)| - \varepsilon)_+ .$$

This loss is 0 when ever $|y - f(x)|$ is below the threshold ε, and otherwise grows linearly with the distance $|y - f(x)|$. Thus it shares some properties with the soft margin loss $l_{sm}(y, f(x))$ for classification where $y \in \{-1, 1\}$. In support vector regression, one looks for a regression function estimator by minimizing

$$n^{-1} \sum_{i=1}^{n} l_\varepsilon(y_i, f(x_i)) + \lambda \langle f, f \rangle_* .$$

The solution can be described in the spirit of the SVM for classification. Support vectors would be those x_i for which $|y_i - f(x_i)| > \varepsilon$. See chap. 9 of Schoellkopf, Smola.

7.4 Summary and further developments

The soft margin loss function

$$l_{sm}(y, f) = (1 - yf)_+ = \begin{cases} 0 \text{ if } yf > 1 \\ 1 - yf \text{ otherwise} \end{cases} .$$

is also often called a *hinge loss*, due to the shape of its graph[5] as a function of yf. For any function $f(x)$ define the empirical hinge loss

$$L_n(f) = n^{-1} \sum_{i=1}^{n} l_{sm}(y_i, f(x_i))$$

[5]hinge: a jointed or flexible device on which a door, lid, or other swinging part turns

where in comparison to (100) the constant a_0 has been absorbed into f. Here the function $f : \mathbb{R}^d \to \mathbb{R}$ is considered a classifier; actually $h(x) = \text{sign}\,(f(x))$ is the derived classifier with values in $\{-1, 1\}$. The SVM is the minimizer in f over a class of functions $\mathcal{F}$ of

$$L_n(f) + \lambda J(f) \tag{101}$$

where $J(f)$ is some penalty functional. A motivation for using the hinge loss rather than the misclassification loss

$$l_0(y, f) = \text{sign}\,(yf) = \begin{cases} 0 \text{ if } yf(x) \geq 0 \\ 1 \text{ otherwise} \end{cases}$$

is that $l_0(y, f)$ poses computational problems for high dimensional data sets. In contrast, minimizing the penalized empirical soft margin loss (101) leads to a computationally tractable problem (Theorem 7.6). We have the following cases.

a Linear SVM**:** $\mathcal{F}$ is the class of linear functions $f(x) = \langle a, x\rangle + a_0$ where $a \in \mathbb{R}^d$, $a_0 \in \mathbb{R}$ and $J(f) = \|a\|^2$.

b kernelized SVM: $\mathcal{F}$ coincides with a RKHS $\mathcal{H}$, pertaining to kernel K, equipped with inner product $\langle f, g\rangle_*$ and $J(f) = \langle f, f\rangle_*$. By the representer theorem, the solution is of form

$$f_\alpha(x) = \sum_{i=1}^{n} \alpha_i K\,(x, x_i)$$

where $\alpha = (\alpha_1, \ldots, \alpha_n)$ and the problem becomes to minimize

$$L_n(f_\alpha) + \lambda \alpha^\top \bar{K} \alpha$$

over α.

For more insight on SVM and the hinge loss cf. Burges [Burg], Blanchard et al. [1]., Tarigan and van de Geer [5]

8 Neural Networks

Suppose again that X is an $\mathbb{R}^d$-valued random vector and Y is a r.v. with two possible values, and (X,Y) have a joint probability distribution. The values of Y are in $\{0,1\}$. Recall that for a hyperplane classifier, the decision was according to the sign of $c^\top x + \tilde{c}_0$ where $c \in \mathbb{R}^d$, $\tilde{c}_0 \in \mathbb{R}$. We may write this in the equivalent form

$$\phi(x) = \begin{cases} 0 \text{ if } \psi(x) \leq 1/2 \\ 1 \text{ otherwise} \end{cases} \tag{102}$$

where $\psi(x) = c^\top x + c_0$ and $c = (c_1, \ldots, c_n) \in \mathbb{R}^d$ and $c_0 = \tilde{c}_0 + 1/2$ is a scalar. Here $x = \left(x^{(1)}, \ldots, x^{(d)}\right)^\top \in \mathbb{R}^d$ are the inputs. This decision rule, historically called a perceptron, now defines a neural network without hidden layers.
In a feed-forward neural network with one hidden layer, one takes

$$\psi(x) = c_0 + \sum_{i=1}^{k} c_i \sigma\left(\psi_i\left(x\right)\right), \tag{103}$$

where the c_i are as before and each ψ_i is of the form

$$\psi_i\left(x\right) = a_{i0} + \sum_{j=1}^{d} a_{ij} x^{(j)}$$

where c_0, c_i, a_{i0}, a_{ij}, $i = 1, \ldots, k$, $j = 1, \ldots, d$ are coefficients, also called *weights*, σ is a sigmoid function (nondecreasing with $\sigma(x) \to -1$ as $x \to -\infty$ and $\sigma(x) \to 1$ as $x \to \infty$).
The principle underlying the neural network thus is: in a linear classifier (or perceptron) $\psi(x) = c^\top x + c_0$, replace the components $x^{(j)}$ by $\sigma\left(a_{j0} + \langle a_j, x\rangle\right)$. Thus we have introduced a "hidden layer" of additional transformation of the input x. This principle may be iterated, i.e we may again replace x by another another hidden layer etc. This gives rise to multilayer neural networks. The basic questions of designing a network then are: how many layers should be used, and which linear coefficients should be zero.
Sigmoid functions. The most commonly used sigmoid function (also called *activation function*) is the logistic

$$\sigma(x) = \frac{1 - \exp(-x)}{1 + \exp(-x)}.$$

also called the *standard sigmoid.* Training the neural network means choosing the weights as functions of a sample (the training set) (x_i, y_i), $i = 1, \ldots, n$ where $y_i \in \{0,1\}$. Other examples are the *threshold sigmoid*

$$\sigma(x) = \begin{cases} -1 \text{ if } x \leq 0 \\ 1 \text{ otherwise} \end{cases},$$

the arctan sigmoid

$$\sigma(x) = \frac{2}{\pi} \arctan(x)$$

and the gaussian sigmoid

$$\sigma(x) = 2\Phi(x) - 1$$

where Φ is the standard normal distribution function.

8.1 Remark *(Classification and regression).* Training a neural network is very similar to estimating coefficients in a nonlinear regression model. Suppose we have an i.i.d. sample (X_i, Y_i), $i = 1, \ldots, n$ where $X_i \in \mathbb{R}^d$ and Y_i are real valued instead of 0-1 valued. Suppose we are interested in estimating the regression function

$$r(x) = E\left(Y|X = x\right) \tag{104}$$

and we try to fit a family of functions $f_\vartheta(x)$, $\vartheta \in \Theta$ to the observed values (X_i, Y_i), $i = 1, \ldots, n$. Here ϑ is an s-dimensional parameter, thus $\Theta \subset \mathbb{R}^s$ and $f_\vartheta(x)$ may be nonlinear both in ϑ and x. The least squares principle may be used to obtain an estimate of ϑ

$$\min_\vartheta \sum_{i=1}^n \left(Y_i - f_\vartheta(X_i)\right)^2.$$

In the neural network described above in (103), we would have to set

$$\vartheta = (c_0, c_i, a_{i0}, a_{ij}, i = 1, \ldots, k, j = 1, \ldots, d)$$

thus $s = k(d+2) + 1$ and

$$f_\vartheta(x) = c_0 + \sum_{i=1}^k c_i \sigma\left(a_{i0} + \langle a_i, x\rangle\right)$$

for $a_i = (a_{i1}, \ldots, a_{id})^T$. In regression the main purpose may be estimation of $r(x)$, but it may also be prediction of Y (for a "new" X to be observed). In classification the goal is prediction of $Y \in \{0, 1\}$. We already observed several times that classification and prediction are closely related. In fact in classification we may also consider the regression function (104) ; then $r(x)$ is the Bernoulli parameter of the posterior distribution of Y. Recall that for classification, the function

$$r(x) = P\left(Y = 1|X = x\right)$$

is the basis for the optimal (Bayes) decision: the Bayes decision is

$$h^*(x) = \begin{cases} 0 \text{ if } r(x) \leq 1/2 \\ 1 \text{ otherwise} \end{cases}$$

Thus in a sense, classification also aims to estimate the regression function (since it always aims to get as close as possible to the Bayes decision $h^*(x)$). Therefore it may be justified to use the least squares principle also for classification, and indeed we shall see that it will play a role in training neural network classifiers.

Sigmoids between 0 and 1. If σ is a sigmoid as above, it is equivalent to use

$$\sigma^*(x) = \frac{1}{2}\left(\sigma(x)+1\right)$$

which is nondecreasing with $\sigma(x) \to 0$ as $x \to -\infty$ and $\sigma(x) \to 1$ as $x \to \infty$ (i.e. a probability distribution function). For the standard sigmoid we obtain

$$\sigma^*(x) = \frac{1}{2}\left(\frac{1-\exp(-x)}{1+\exp(-x)}+1\right) = \frac{1}{1+\exp(-x)} = \frac{\exp(x)}{1+\exp(x)}$$

i.e. the logistic distribution function with density

$$\frac{1+\exp(-x)}{\left(1+\exp(-x)\right)^2}.$$

Using σ^* and σ is equivalent since the linear transformation involved can be incorporated into the coefficients c_0, c_i in (103). ■

Logistic regression. Note that in the section on logistic regression, we used notation $l(x) = \exp(x)/\left(1+\exp(x)\right)$ for the logistic function; thus $l(x) = \sigma^*(x)$. One models the regression function $r(x)$ (for classification) as

$$r(x) = l(\beta_0 + \langle \beta, x\rangle)$$

and estimates the parameters β_0, β by maximum likelihood. Thus, for estimated $\hat{\beta}_0, \hat{\beta}$, the decision for x is

$$\phi(x) = \begin{cases} 1 \text{ if } \hat{r}(x) \leq 1/2 \\ 0 \text{ otherwise} \end{cases}$$

where $\hat{r}(x) = l(\hat{\beta}_0 + \left\langle \hat{\beta}, x\right\rangle)$. Thus linear logistic regression is formally a special of a neural network with one hidden layer (103) and only one neuron, that is with $k = 1$. However the power of logistic regression derives from the nonparametric version where we set $r(x) = l(g(x))$ where g is some function, which is then estimated using penalization. With neural networks, one adds complexity in a different way (multilayer networks). ■

Multilayer networks. In the neural network with one hidden layer, we say there are k hidden neurons- the output of the j-th hidden neuron is $u^{(j)} = \sigma(\psi_j(x))$. Thus (103) may be rewritten

$$\psi(x) = c_0 + \sum_{j=1}^{k} c_j u^{(j)}$$

which is similar in form to the simple linear classifier (perceptron). To obtain e.g. a two-hidden layer network we set

$$\begin{aligned} \psi(x) &= c_0 + \langle c, z\rangle, \qquad (105) \\ z^{(i)} &= \sigma\left(b_{i0} + \langle b_i, u\rangle\right), \; i = 1,\ldots,l, \\ u^{(j)} &= \sigma\left(a_{j0} + \langle a_j, x\rangle\right), \; j = 1,\ldots,k \\ u &= (u^{(1)},\ldots,u^{(k)})^\top, z = (z^{(1)},\ldots,z^{(l)})^\top, x = (x^{(1)},\ldots,x^{(d)})^\top \qquad (106) \end{aligned}$$

where c_0, a_{j0}, b_{i0} are constants and a_j are d-vectors, b_i are k-vectors and c is a l-vector. The first hidden layer has l hidden neurons while the second hidden layer has k hidden neurons. Here a *neuron* is an expression of form $\sigma\left(b + \langle a, u\rangle\right)$ for an input vector u and coefficients b, a.

K-class classification. In this case the first layer which consists of one linear function in (103) or (105) is replaced by K linear terms

$$\psi^{(s)}(x) = c_{0s} + \langle c_s, z\rangle\,,\ s = 1, \ldots, K$$

and the classification into one of K classes is obtained in the following way. Consider the *softmax functions* for argument $t = (t^{(1)}, \ldots, t^{(K)})^\top$

$$g_s(t) = \frac{\exp(t^{(s)})}{\sum_{l=1}^{K} \exp(t^{(l)})},\ s = 1, \ldots, K \tag{107}$$

then all values are positive with $\sum_{s=1}^{K} g_s(t) = 1$; and if we define a vector valued map $\tilde{\psi}(x) = (\psi^{(1)}(x), \ldots, \psi^{(K)}(x))^\top$ then the transformation

$$g_s(\tilde{\psi}(x)),\ s = 1, \ldots, K$$

is used as a basis for classification. The vector x will be classified into the class s for which $g_s(\tilde{\psi}(x))$ is largest.
In this approach, the values $g_s(\tilde{\psi}(x))$ are obviously considered estimates of the posterior probability of class s given $X = x$, in the same way as $\psi(x)$ in (102) is treated as an estimate of the posterior probability $r(x) = P\left(Y = 1|X = x\right)$ (though we have not required for the neural network that $\psi(x)$ takes values between 0 and 1). Consider the case of two classes $K = 2$ and rename the classes $0, 1$; then we decide

$$\phi(x) = \begin{cases} 0 \text{ if } g_0(\tilde{\psi}(x)) \leq 1/2 \\ 1 \text{ otherwise} \end{cases}$$

or equivalently we decide 0 if

$$\exp(\psi^{(0)}(x)) \leq \frac{1}{2}\left(\exp(\psi^{(0)}(x)) + \exp(\psi^{(1)}(x))\right)$$

which means $\exp(\psi^{(0)}(x)) \leq \exp(\psi^{(1)}(x))$ and is equivalent to

$$\psi^{(0)}(x) \leq \psi^{(1)}(x).$$

Now the difference $\psi^{(0)}(x) - \psi^{(1)}(x)$ is again of form $\psi(x)$ as in (102)

$$\psi(x) = \psi^{(0)}(x) - \psi^{(1)}(x) = c_{00} - c_{01} + \langle c_0 - c_1, z\rangle$$

so the decision is made according to $\psi(x) \leq 0$ or $\psi(x) > 0$, or by adding $1/2$ to both sides, is the same as in (102). Thus binary classification is a special case. ■

8.1 Fitting neural networks

Consider a single hidden layer neural network which is a classifier for the K-class problem. We write the neural network to be trained

$$\begin{aligned} z^{(i)} &= \sigma\left(a_{i0} + \langle a_i, x\rangle\right),\, i = 1, \dots, m, \\ t^{(k)} &= c_{0k} + \langle c_k, z\rangle,\, k = 1, \dots, K, \\ \psi_k(x) &= g_k(t), k = 1, \dots, K \end{aligned} \tag{108}$$

where $x = (x^{(1)}, \dots, x^{(d)})^\top$, $z = (z^{(1)}, \dots, z^{(m)})^\top$ and $t = (t^{(1)}, \dots, t^{(K)})^\top$. Here $g_k(t)$ is the k-th softmax function (107). Thus we consider a the network which is a classifier for the K-class problem. Also we consider the simplest case of only one hidden layer, represented by the z_i.
The neural network model has unknown parameters, often called weights, and we seek values for them that make the model fit the training data well. We denote the complete set of weights by θ, thus

$$\theta = (a_{10}, a_1, \dots, a_{m0}, a_m, c_{01}, c_1, \dots, c_{0K}, c_K)$$

which is $m(d+1)+K(m+1)$ dimensional. The components of θ are often called "weights"; thus we have $m(d+1)+K(m+1)$ weights in this neural network. However, implicitly additional constraints on θ are present, effectively reducing the dimensionality (i.e. the number of weights, see Remark 8.2) below).
For classification, we use sum-of-squared errors as our measure of fit (error function): for a training set (x_i, y_i) $i = 1, \dots, n$ where $x_i \in \mathbb{R}^d$ and $y_i = (y_i^{(1)}, \dots, y_i^{(K)})$ is an indicator vector (where $y_i^{(k)} = 1$ if x_i is in class k, 0 otherwise)

$$R(\theta) = \sum_{i=1}^{n}\sum_{k=1}^{K}\left(y_i^{(k)} - \psi_k(x_i)\right)^2 \tag{109}$$

and the corresponding classifier is $G(x) = \arg\max_{k\in\{1,\dots,K\}} \psi_k(x)$.
.
The generic approach to minimizing $R(\theta)$ is by gradient descent, called *back-propagation* in this setting. Because of the compositional form of the model, the gradient can be easily derived using the chain rule for differentiation. This can be computed by a forward and backward sweep over the network, keeping track only of quantities local to each unit.
An alternative measure of fit is *cross-entropy* (deviance):

$$R(\theta) = -\sum_{i=1}^{n}\sum_{k=1}^{K} y_i^{(k)} \log \psi_k(x_i) \tag{110}$$

With the softmax activation function and the cross-entropy error function, the neural network model is exactly a linear logistic regression model in the hidden units, and all the parameters are estimated by maximum likelihood.

Multinomial logistic regression. In linear logistic regression, cp (62), we set $p_x := P(Y = 1|X = x)$ and note that the joint law of (X, Y) is determined by p_x and the marginal law of X, say $\mathcal{L}(X)$. Indeed, given $X = x$, the conditional law of Y is a Bernoulli distribution with parameter p_x and $P(Y = 0|X = x) = 1 - p_x$. In logistic regression it is assumed that

$$p_x = l(a_0 + \langle a, x\rangle), \tag{111}$$

for some $a \in \mathbb{R}^d$ and $a_0 \in \mathbb{R}$, where $l(t) = \exp(t)/(1 + \exp(t))$ is the logistic function. The marginal distribution of X is not specified; thus we assume that our training set (x_i, y_i) has been generated from a model

$$Y_i \sim \text{Bernoulli}\left(l(a_0 + \langle a, x_i\rangle)\right) \tag{112}$$

The parameters a, a_0 are estimated from the training set, by maximum likelihood based (112) with fixed on $x_1, \ldots, x_n$, as $\hat{a}, \hat{a}_0$, giving for each x an estimated $\hat{p}_x = l(\hat{a}_0 + \langle \hat{a}, x\rangle)$. This $\hat{p}_x$ provides an obvious classifier h: $h(x) = 1$ if $\hat{p}_x > 1/2$. Recall that the (unknown) Bayes classifier h^* for a joint law of (X, Y) is a function of the (unkown) posterior probability $p_x = P(Y = 1|X = x)$ and classifies 1 if $p_x > 1/2$. The assumption (111) is true if the two class-conditional laws of X are normal: the law of (X, Y) is such that $\mathcal{L}(X|Y = i) = N_d(\mu_i, \Sigma)$, $i = 0, 1$ and $P(Y = 1) = 1/2$ (Proposition 6.7).
Let us consider a generalization of this setup to the K-classes case. Note that $Y \sim \text{Bernoulli}(p)$ holds if an only if the random vector $\mathbf{Y} = (1 - Y, Y)$ has a multinomial distribution

$$\mathfrak{M}_2(1, \mathbf{p}) \text{ where } \mathbf{p} = (1 - p, p)\,.$$

Write the constant a_0 in (112) as a difference $a_0 = a_{20} - a_{10}$ and similarly for the vector a: $a = a_2 - a_1$. Then the assumption (111) means for the probability vector $\mathbf{p}$ and $t^{(1)} = a_{10} + \langle a_1, x\rangle$, $t^{(2)} = a_{20} + \langle a_2, x\rangle$

$$\begin{aligned}
\mathbf{p} &= \left(1 - l(t^{(2)} - t^{(1)}), l(t^{(2)} - t^{(1)})\right) \\
&= \left(\frac{1}{1 + \exp(t^{(2)} - t^{(1)})}, \frac{\exp(t^{(2)} - t^{(1)})}{1 + \exp(t^{(2)} - t^{(1)})}\right) = \\
&= \left(\frac{\exp(t^{(1)})}{1 + \exp(t^{(2)} - t^{(1)})}, \frac{\exp(t^{(2)})}{1 + \exp(t^{(2)} - t^{(1)})}\right) \\
&= (g_1(\mathbf{t}), g_2(\mathbf{t}))
\end{aligned}$$

where g_1, g_2 are the softmax functions (107) for 2 dimensional input $\mathbf{t} = (t^{(1)}, t^{(2)})$.

8.2 Remark *The original logistic regression contained $d+1$ parameters a_0,a, whereas we are now using a $2(d + 1)$-dimensional parameter $\theta = (a_{10}, a_1, a_{20}, a_2)$ (related to a_0,a by $a_0 = a_{20} - a_{10}$, $a = a_2 - a_1$). Clearly the model is overparametrized now, that is θ is not identifiable (different values of θ give rise to the same law of $\mathbf{Y}$, for all possible values of x). To eliminate this nonidentifiability, we have to introduce $d + 1$ constraints on θ; an example would be $a_{20} = 0, a_2 = 0$. In general we* *write $H(\theta) = 0$ for such constraints, where H is a $\mathbb{R}^{d+1}$-valued map (linear or nonlinear). Note that such constraints should also be present on the c_{0k}, c_k, $k = 1, \ldots, K$ in the second line of (108), cf also below. .*

Thus the natural generalization of (112) to the K-class case is, for observed indicator vectors $\mathbf{Y}_i$ (K-vectors having a 1 in one position and a 0 in all others)

$$\mathbf{Y}_i \sim \mathfrak{M}_K(1, \mathbf{p}_{x_i}) \tag{113}$$

where $\mathbf{p}_x$ is a vector valued function of $x \in \mathbb{R}^d$, described as follows. Define the vector valued map $\mathbf{g}(\mathbf{t})$ for K- dimensional argument $\mathbf{t} = (t^{(1)}, \ldots, t^{(K)})$ as

$$\mathbf{g}(\mathbf{t}) = (g_1(\mathbf{t}), \ldots, g_K(\mathbf{t}))$$

where $g_s(\mathbf{t})$ is the s-th softmax function (107). Consider a_{k0} and d-vectors a_k, $k = 1, \ldots, K$; then the model assumption on $\mathbf{p}_x$ is

$$\mathbf{p}_x = \mathbf{g}\left((a_{10} + \langle a_1, x\rangle, \ldots, a_{K0} + \langle a_K, x\rangle)\right). \tag{114}$$

with $d+1$ additional constraints $H(\theta) = 0$ (where $\theta = (a_{10}, a_1, \ldots, a_{K0}, a_K)$), possibly given by $a_{10} = 0, a_1 = 0$. Together assumptions (113), (114) constitute the multinomial logistic regression model. Note that here $\mathbf{Y}_i$ are independent multinomial, but not identically distributed.
As in the univariate model, given a training set $(x_i, \mathbf{y}_i)$, $i = 1, \ldots, n$, one would estimate the parameters a_{k0}, a_k, $k = 1, \ldots, K$ by maximum likelihood. For shortness write for an observed $\mathbf{y} = \left(y^{(1)}, \ldots, y^{(K)}\right)$ and a probability vector $\mathbf{p} = (p_1, \ldots, p_K)$

$$\mathbf{p}^{\mathbf{y}} = \prod_{k=1}^{K} p_k^{y^{(k)}},$$

then the likelihood function (for fixed x_i) is

$$\prod_{i=1}^{n} \mathbf{p}_{x_i}^{\mathbf{y}_i}.$$

Thus the negative log-likelihood is

$$-\sum_{i=1}^{n} \log \mathbf{p}_{x_i}^{\mathbf{y}_i} = -\sum_{i=1}^{n}\sum_{k=1}^{K} y_i^{(k)} \log g_k\left((a_{10} + \langle a_1, x_i\rangle, \ldots, a_{K0} + \langle a_K, x_i\rangle)\right).$$

Comparing this with (110) we see that the expressions coincide if we identify $\psi_k(x_i)$ and $g_k\left((a_{10} + \langle a_1, x_i\rangle, \ldots, a_{K0} + \langle a_K, x_i\rangle)\right)$. However in (110) we actually have, according to the network structure (108),

$$\psi_k(x) = g_k(t) = g_k(c_{01} + \langle c_1, z\rangle, \ldots, c_{0K} + \langle c_K, z\rangle) \tag{115}$$

where $z = (z^{(1)}, \ldots, z^{(m)})$ are the hidden units, so (110) is indeed the multinomial logistic negloglikelihood in the hidden units. In the latter, we need $m+1$ additional constraints to make $c_{01}, c_1, \ldots, c_{0m}, c_m$ identifiable, such as $c_{01} = 0, c_1 = 0$.

8.2 An example: handwritten ZIP codes

We describe an experiment with neural networks by Le Cun (1989)[6], also discussed in [HTF], p. 404. A training set of $n = 320$ handwritten ZIP-code digits was considered; each handwritten digit came classified into one of $K = 10$ classes. Thus the training set was (x_i, y_i) where $y_i \in \{0, 1, \ldots, 9\}$ and $x_i \in \mathbb{R}^d$. (Equivalently, y_i could be a 10-dimensional indicator vector). Here $d = 256$, as each handwritten digit x_i came as a $16 \times 16 = 256$ pixel image, each pixel having an 8-bit certain grayscale value (that is a grayscale value from 0 to 255).
Along with the training set, there was a test set of 160 handwritten data $(\tilde{x}_i, \tilde{y}_i)$, $i = 1, \ldots, 160$. Five different neural networks were fit to the data (x_i, y_i) (trained on the training set) and then their preformance on the test set was evaluated. Table 1 (picture) in column 4 gives the performance (% of correct recognition) of each of the five networks on the test set. All 5 networks were fitted to the training set using the least squares error criterion (109). In each case, complete separation of the classes was achieved on the training set (no misclassifications). Note that already in Net-1 there are more parameters (weights) than observations, namely $n = 320$ observations and 2570 parameters. This explains the error-free performance of all networks on the training set.
We proceed to describe the 5 models used (Net-1 to Net-5).

Net-1. This is a single layer (no hidden layer) network, that is a network with no hidden units. Thus for $K = 10$

$$\psi_k(x) = g_k((a_{10} + \langle a_1, x\rangle, \ldots, a_{K0} + \langle a_K, x\rangle), k = 1, \ldots, K$$

Here the parameter is

$$\theta = (a_{10}, a_1, \ldots, a_{K0}, a_K)$$

which is of dimension $K(d+1) = 10 \cdot 257 = 2570$ (except for the $d+1 = 11$ additional constraints on θ). According to (113)-(115), this is equivalent ot fitting a multinomial logistic regression, except for the fact that maximum likelihood is not used here for training, but rather the least squares criterion. Line 1 of table 1 gives then number of links is 2570 (each of the 257 components of the augmented input $(x, 1)$ influences each of the 10 components of $\psi_k(x)$). However the effective number of weights is less, due to the $d + 1 = 257$ implicit constraints $H(\theta) = 0$. Net-1 correctly classifies 80% of the test set.

Net-2. This is a single hidden layer (two layer) network, fully connected. A network which has all possible links and no additional constraints on the parameter (except the necessary $H(\theta) = 0$ for the softmax function) may be called a "plain vanilla network". Here there are 12 hidden units; thus the model can be written as (108) with $m = 12$, $K = 10$, $d = 256$. Thus the dimension of θ (number of weights) is

$$m(d + 1) + K(m + 1) = 12 \cdot 257 + 10 \cdot 13 = 3214$$

[6] Le Cun, Y. (1989). Generalization and Network Design Strategies. Technical Report CRG-TR-89-4, Department of Computer Science, University of Toronto.

(the 3240 in the "number of weights" column for this network has been corrected to 3214 in [HTF]). Since this is a plain vanilla network, the number of links equals the number of weights. Net-2 correctly classifies 87% of the test set.

Net-3. Here we have two hidden layers (3 layers), that is the architecture given in (105) but not as a plain vanilla network but with "local connectivity". That means that not all possible links are present; rather each unit in the two hidden layers is influenced only by *some* (not all) units in the layer below. Suppose we write a two hidden layer network for K classes, analogously to (105) which was for $K = 2$ classes, as

$$u^{(i)} = \sigma\left(a_{i0} + \langle a_i, x\rangle\right), \; i = 1, \ldots, m, \tag{116}$$

$$z^{(j)} = \sigma\left(b_{j0} + \langle b_j, u\rangle\right), \; j = 1, \ldots, l \tag{117}$$

$$t^{(k)} = c_{0k} + \langle c_k, z\rangle, \; k = 1, \ldots, K,$$

$$\psi_k(x) = g_k(t), k = 1, \ldots, K$$

where $u = (u^{(1)}, \ldots, u^{(m)})^\top$, $z = (z^{(1)}, \ldots, z^{(l)})^\top$, $x = (x^{(1)}, \ldots, x^{(d)})^\top$. In Net-3 the first hidden layer is represented as a 8×8 pixel picture, hence there are $m = 64$ units $u^{(i)}$ in the first hidden layer and $l = 4 \times 4 = 16$ units $z^{(j)}$ in the second hidden layer. Local connectivity here means that each 256-vector a_i has only 9 nonzero components, and each 64-vector b_j has only 25 nonzero components. Thus Net-3, even though it has two hidden layers, has fewer weights and links than the plain vanilla Net-2, and achieves comparable performance 88.5 %.

Net-4. This adds "shared weights" to the two hidden layer architecture of Net-3. The first (lower) hidden layer is divided into two parts, represented as two 8×8 "pictures", and with in each of the two parts the weights are equal. That means that in (116), $m = 8 \times 8 \times 2 = 128$ and we introduce equality constraints

$$(a_{10}, a_1) = \ldots = (a_{64,0}, a_{64}),$$

$$(a_{65,0}, a_{65}) = \ldots = (a_{128,0}, a_{128}).$$

In addition, local connectivity is present, i.e. most of the components of a_i are zero. The second hiden layer is not split (no weight sharing there). This network has fewer weights than Net-3 but better performance (94 %).

Net -5. Here the second hidden layer is also split, into 4 parts now, and within each part there is weight sharing (the first hidden layer has the same structure of the first hidden layer as Net-4). This net has the best performance: 98.4 %.

9 Histogram and Tree Methods

9.1 Partitioning rules

We mostly follow [DGL], sec. 6.3 here. Many important classification rules partition $\mathbb{R}^d$ into disjoint cells $A_1, A_2, \ldots$and and classify in each cell according to the majority vote among the labels of the X_i's falling in the same cell. More precisely, if $A(x)$ denotes the cell into which $x \in \mathbb{R}^d$ is falling,

$$h_n(x) = \begin{cases} 0 \text{ if } \operatorname{card}\{(X_i, Y_i) : X_i \in A(x), Y_i = 1\} \leq \operatorname{card}\{(X_i, Y_i) : X_i \in A(x), Y_i = 0\} \\ 1 \text{ otherwise} \end{cases} \tag{118}$$

The decision is zero if the number of ones does not exceed the number of zeros in the cell where X falls, and vice versa. The partitions we consider in this section may change with n, and they may also depend on the points $X_1, \ldots, X_n$,, but we assume that they do not depend on the Y_i. We develop a general consistency result for such partitioning rules. It requires two properties of the partition: first, cells should be small enough so that local changes of the distribution can be detected. On the other hand, cells should be large enough to contain a large number of points so that averaging among the labels is effective. diam(A) denotes the diameter of a set A, that is,

$$\operatorname{diam}(A) = \sup_{x,y \in A} \|x - y\|$$

As usual, we assume that (X, Y), (X_i, Y_i), $i = 1, \ldots, n$ are i.i.d. distributed ($Y \in \{0, 1\}$), and let $\mu = \mathcal{L}(X)$ denote the marginal distribution of X. Let also μ_n denote the empirical measure of the X_i:

$$\mu_n(A) = n^{-1} \sum_{i=1}^{n} \mathbf{1}_A(X_i)$$

and denote

$$N(x) = n\mu_n(A(x)) = \sum_{i=1}^{n} \mathbf{1}_{A(x)}(X_i)$$

the number of X_i's falling in the same cell as x.

Error probabilities. Recall that a classification rule is a mapping $h : \mathbb{R}^d \to \{0, 1\}$, which also depends on the training set $T_n := ((X_1, Y_1), \ldots, (X_n, Y_n))$. So actually h is a (measurable) function

$$h : \mathbb{R}^d \times \left\{\mathbb{R}^d \times \{0, 1\}\right\}^{\times n} \to \{0, 1\}$$

and we occasionally (but not always) write $h(X, T_n)$. For a given decision $\kappa \in \{0, 1\}$ and a realized value y of the r.v. Y we write the loss (or prediction error)

$$L(\kappa, y) = \mathbf{1}\{\kappa \neq y\}$$

that is, the loss is 0 if κ and y coincide, and 1 otherwise. For a given (fixed) training set t_n, the performance of a rule h is measured by the probability of error when (X, Y) follow their joint distribution and $h(X, t_n)$ is used to predict Y

$$P\left(h(X, t_n) \neq Y\right) = EL\left(h(X, t_n), Y\right). \tag{119}$$

Following the notation in [DGL], we shall always write $L_n = L\left(h(X, T_n), Y\right)$ if it is clear from the context which rule h is being considered. Furthermore, the above error probability (119) for given training set t_n is written as a conditional probability:

$$E\left(L_n | T_n = t_n\right) := EL\left(h(X, t_n), Y\right)$$

$$= P\left(h_n \neq Y | T_n = t_n\right) = P\left(h(X, t_n) \neq Y\right)$$

where on the right side, the expectation refers to the random (X, Y) only. (Here we wrote $h_n = h(X, t_n)$.) The final measure of performance for the rule h_n is obtained by averaging over the training set T_n:

$$E\left(L_n\right) = E^T\left(E\left(L_n | T_n\right)\right), \tag{120}$$

$$= P\left(h_n \neq Y\right) = E^T\left(P\left(h(X, T_n) \neq Y\right)\right)$$

where on the right side, the expectation E^T refers to the random training set T_n.

Bayes rule. According to Theorem 2.1, the Bayes rule h^* is optimal among all rules $h : \mathbb{R}^d \to \{0, 1\}$, including those which depend on a training set T_n. The Bayes rule requires knowledge of $\mathcal{L}(X, Y)$ and is therefore not available in practice. Denoting

$$L^* := EL\left(h^*(X), Y\right) = P\left(h^* \neq Y\right)$$

where E and P refer only to $\mathcal{L}(X, Y)$, Theorem 2.1 implies for any rule $h_n = h(X, T_n)$

$$P\left(h_n \neq Y | T_n = t_n\right) = E\left(L_n | T_n = t_n\right) \geq L^* \tag{121}$$

and therefore also

$$P\left(h_n \neq Y\right) = E\left(L_n\right) \geq L^*.$$

.

Consistency. Recall that in (ordinary) statistical parameter estimation, an estimator $\hat{\theta}_n$ of a real valued parameter θ is called *consistent* if $\hat{\theta}_n \to_P \theta$ (convergence in probability). If θ is an element of a metric space and $d(\cdot, \cdot)$ the pertaining metric, then consistency at θ is defined as

$$d\left(\hat{\theta}_n, \theta\right) \to_P 0 \text{ as } n \to \infty. \tag{122}$$

In classification, the analog of the metric $d(\cdot, \cdot)$, is

$$P\left(h_n \neq Y | T_n\right) - L^* = E\left(L_n | T_n\right) - L^*$$

which in some sense measures the distance of h_n to the Bayes rule. Note that here the role of $\hat{\theta}_n$ is played by h_n and the role of θ (true parameter) is played by $\mathcal{L}(X, Y)$. Thus, classification fits very much into the framework of statistical decision theory: we try to capture a certain characteristic of the unknown law $\mathcal{L}(X, Y)$ by a decision function h_n, based on a sample $T_n = ((X_1, Y_1), \ldots, (X_n, Y_n))$.
Thus, in analogy to (122), a classification rule h_n may be called *consistent* at $\mathcal{L}(X, Y)$ if

$$E(L_n | T_n) - L^* \to_P 0.$$

Note that $\zeta_n := E(L_n | T_n) - L^*$ is a bounded random variable (as a function of T_n): in view of (121) we have $0 \le \zeta_n \le 1 - L^* \le 1$. If $0 \le \zeta_n \le 1$ then $\zeta_n \to_P 0$ is equivalent to $E\zeta_n \to 0$. (If $E\zeta_n \to 0$ then $\zeta_n \to_P 0$ by Chebyshev's inequality. For the other direction, cf. the topic "Integration to the Limit", Theorem 25.12 in Billingsley (1995)[7])
According to (120) we have $E\zeta_n = E(L_n) - L^*$. Thus we may equivalently define consistency of a classification rule h_n at $\mathcal{L}(X, Y)$ by

$$E(L_n) \to L^*$$

In (ordinary) statistical parameter estimation, consistency of an estimator $\hat{\theta}_n$ may not hold at all unknown values of θ or only under certain conditions. For instance a certain estimator for an expectation may be consistent only if the expectation exists, or if it exists and additionally a certain variance is finite etc. In classification, the lower bound L^* for the error probability (121) is always defined. Thus a rule h_n may be called *universally consistent* if

$$E(L_n) \to_P L^* \text{ for all unknown distributions } \mathcal{L}(X, Y).$$

Below we will establish universal consistency of the histogram rule.

Return first to the general partition rules (118). The conditions of the theorem below require that a random cell—selected according to the distribution of X- has a small diameter, and contains many points with large probability.

9.1 Theorem *Consider a partitioning classification rule as defined above in (118). Then* $E(L_n) \to L^*$ *if*
(i) $\operatorname{diam}(A(X)) \to 0$ *in probability*
(ii) $N(X) \to \infty$ *in probability.*

Proof. Define $r(x) = P(Y = 1 | X = x)$ (the regression function in $\mathcal{L}(X, Y)$). Define

$$\hat{r}_n(x) = \frac{\operatorname{card}\{(X_i, Y_i) : X_i \in A(x), Y_i = 1\}}{N(x)} = \frac{1}{N(x)} \sum_{i=1}^{n} Y_i \mathbf{1}_{A(x)}(X_i).$$

Then the rule h_n in (118) may equivalently be written

$$h_n(x) = \begin{cases} 0 \text{ if } \hat{r}_n(x) \le 1 - \hat{r}_n(x) \\ 1 \text{ otherwise} \end{cases}$$
$$= \begin{cases} 0 \text{ if } \hat{r}_n(x) \le 1/2 \\ 1 \text{ otherwise} \end{cases}. \quad (123)$$

Thus we may consider $\hat{r}_n(x)$ an estimator of $r(x)$ and $h_n(x)$ decides in analogy to the Bayes rule, with a threshold 1/2. In this situation we claim

$$E\left(L_n|T_n\right) - L^* \leq 2\int_{\mathbb{R}^d} |r(x) - \hat{r}_n(x)|\, d\mu(x) = 2E\left(|r(X) - \hat{r}_n(X)|\, |T_n\right) \tag{124}$$

To prove this, note that given $X = x$, (and also $T_n = t_n$, but we initially do not write possible dependence on t_n) the conditional error probability of any rule h may be expressed as

$$P\left(h(X) \neq Y|X = x\right) = 1 - P\left(h(X) = Y|X = x\right)$$

$$\begin{aligned}
&= 1 - P\left(Y = 1, h(X) = 1|X = x\right) - P\left(Y = 0, h(X) = 0|X = x\right)\\
&= 1 - \mathbf{1}_{\{h(x)=1\}}P\left(Y = 1|X = x\right) - \mathbf{1}_{\{h(x)=0\}}P\left(Y = 0|X = x\right)\\
&= 1 - \mathbf{1}_{\{h(x)=1\}}r(x) - \mathbf{1}_{\{h(x)=0\}}\left(1 - r(x)\right).
\end{aligned}$$

Thus for every $x \in \mathbb{R}^d$, if h^* is the Bayes rule

$$\begin{aligned}
&P\left(h(X) \neq Y|X = x\right) - P\left(h^*(X) \neq Y|X = x\right)\\
&= r(x)\left(\mathbf{1}_{\{h^*(x)=1\}} - \mathbf{1}_{\{h(x)=1\}}\right) + \left(1 - r(x)\right)\left(\mathbf{1}_{\{h^*(x)=0\}} - \mathbf{1}_{\{h(x)=0\}}\right)\\
&= r(x)\left(\mathbf{1}_{\{h^*(x)=1\}} - \mathbf{1}_{\{h(x)=1\}}\right) + \left(1 - r(x)\right)\left(-\mathbf{1}_{\{h^*(x)=1\}} + \mathbf{1}_{\{h(x)=1\}}\right)\\
&= \left(2r(x) - 1\right)\left(\mathbf{1}_{\{h^*(x)=1\}} - \mathbf{1}_{\{h(x)=1\}}\right).
\end{aligned}$$

Note that $\mathbf{1}_{\{h^*(x)=1\}} - \mathbf{1}_{\{h(x)=1\}}$ is nonzero if and only if $h^*(x) \neq h(x)$; if $\mathbf{1}_{\{h^*(x)=1\}} - \mathbf{1}_{\{h(x)=1\}} = -1$ then $h^*(x) = 0$ and hence (h^* being the Bayes rule) $r(x) \leq 1/2$, hence $2r(x) - 1 \leq 0$; if $\mathbf{1}_{\{h^*(x)=1\}} - \mathbf{1}_{\{h(x)=1\}} = 1$ then $h^*(x) = 1$ and hence $r(x) > 1/2$, consequently $2r(x) - 1 > 0$. This reasoning entails

$$P\left(h(X) \neq Y|X = x\right) - P\left(h^*(X) \neq Y|X = x\right) = 2\left|r(x) - \frac{1}{2}\right|\mathbf{1}_{\{h^*(x)\neq h(x)\}} \tag{125}$$

Now setting $h_n = h$ and using the description (123) in terms of $\hat{r}_n(x)$ we note: if $h^*(x) \neq h_n(x)$ then $r(x)$ and $\hat{r}_n(x)$ are at opposite sides of 1/2, hence

$$\left|r(x) - \frac{1}{2}\right| \leq \left|r(x) - \hat{r}_n(x)\right|. \tag{126}$$

Taking an expectation in (125) over X (with $T_n = t_n$ still fixed) and using (126) we obtain (124). Thus we need only show

$$E\left(|r(X) - \hat{r}_n(X)|\, |T_n\right) \to_P 0$$

or equivalently

$$E\left(|r(X) - \hat{r}_n(X)|\right) \to 0.$$

Introduce

$$\bar{r}(x) = E\left(r(X)|X \in A(x)\right) = P\left(Y = 1|X \in A(x)\right). \tag{127}$$

By the triangle inequality

$$E\left(|r(X)-\hat{r}_n(X)|\right) \le E\left(|\hat{r}_n(X)-\bar{r}(X)|\right)+E\left(|\bar{r}(X)-r(X)|\right).$$

By conditioning on the random variable $N(x)$, it is easy to see that $N(x)\hat{r}_n(x)=\sum_{i=1}^{n} Y_i \mathbf{1}_{A(x)}(X_i)$ is a binomial random variable $B\left(N(x),\bar{r}(x)\right)$ with parameters $N(x)$ and $\bar{r}(x)$. (That requires a short reasoning. Note that unconditionally, the r.v. $\sum_{i=1}^{n} Y_i \mathbf{1}_{A(x)}(X_i)$ is binomial $B\left(n, P\left(Y=1, X\in A(x)\right)\right)$). Thus

$$\begin{aligned}
&E\left(|\hat{r}_n(x)-\bar{r}(x)|\,|\mathbf{1}_{\{X_1\in A(x)\}},\ldots,\mathbf{1}_{\{X_n\in A(x)\}}\right)\\
&\le E\left(\left|\frac{B\left(N(x),\bar{r}(x)\right)}{N(x)}-\bar{r}(x)\right|\mathbf{1}_{\{N(x)>0\}}|\mathbf{1}_{\{X_1\in A(x)\}},\ldots,\mathbf{1}_{\{X_n\in A(x)\}}\right)+\\
&+E\left(\mathbf{1}_{\{N(x)=0\}}|\mathbf{1}_{\{X_1\in A(x)\}},\ldots,\mathbf{1}_{\{X_n\in A(x)\}}\right)
\end{aligned}$$

For a binomial $B(m,q)$, we have by Cauchy-Schwarz

$$E\left|\frac{B\left(m,q\right)}{m}-q)\right| \le \left(\frac{\mathrm{Var}\left(B\left(m,q\right)\right)}{m^2}\right)^{1/2}=\left(\frac{q(1-q)}{m}\right)^{1/2}$$

and applying this to the previous display we obtain

$$\begin{aligned}
&E\left(|\hat{r}_n(x)-\bar{r}(x)|\,|\mathbf{1}_{\{X_1\in A(x)\}},\ldots,\mathbf{1}_{\{X_n\in A(x)\}}\right)\\
&\le E\left(\left(\frac{\bar{r}(x)\left(1-\bar{r}(x)\right)}{N(x)}\right)^{1/2}\mathbf{1}_{\{N(x)>0\}}|\mathbf{1}_{\{X_1\in A(x)\}},\ldots,\mathbf{1}_{\{X_n\in A(x)\}}\right)\\
&+E\left(\mathbf{1}_{\{N(x)=0\}}|\mathbf{1}_{\{X_1\in A(x)\}},\ldots,\mathbf{1}_{\{X_n\in A(x)\}}\right).
\end{aligned}$$

Taking an expectation over $X=x$ and $X_1,\ldots,X_1$ we see that (noting $\bar{r}(x)\left(1-\bar{r}(x)\right)\le 1/4$)

$$\begin{aligned}
E\left(|\hat{r}_n(x)-\bar{r}(x)|\right) &\le E\left(\frac{1}{2\sqrt{N(X)}}\mathbf{1}_{\{N(X)>0\}}\right)+P\left(N(X)=0\right)\\
&\le \frac{1}{2}P\left(N(X)\le k\right)+\frac{1}{2\sqrt{k}}+P\left(N(X)=0\right)
\end{aligned}$$

for any k, and this can be made small, first by choosing k large enough and then by using condition (ii).

For $\varepsilon>0$, find a uniformly continuous $[0,1]$-valued function r_ε on a bounded set C and vanishing off C so that $E\left|r_\varepsilon(X)-r(X)\right|<\varepsilon$. (Indeed, $r(x)$ is measurable and bounded with values in $[0,1]$. Such an approximation is similar to an approximation of r by step functions, see basic texts about integration). Next, setting analogously to (127)

$$\bar{r}_\varepsilon(x)=E\left(r_\varepsilon(X)|X\in A(x)\right)$$

we employ the triangle inequality:

$$\begin{aligned}E\left|\bar{r}(X)-r(X)\right| &\leq E\left|\bar{r}(X)-\bar{r}_{\varepsilon}(X)\right| \\ &+E\left|\bar{r}_{\varepsilon}(X)-r_{\varepsilon}(X)\right| \\ &+E\left|r_{\varepsilon}(X)-r(X)\right| \\ &=I+II+III.\end{aligned}$$

Clearly, $III<\varepsilon$ by choice of r_ε. Since r_ε is uniformly continuous, and $\bar{r}_\varepsilon(x)$ represents an average of the function r over the set $A(x)$, we can find a $\theta=\theta\left(\varepsilon\right)>0$ such that

$$II\leq\varepsilon+P\left(\operatorname{diam}(A(X))>\theta\right)$$

There $II<2\varepsilon$ for n large enough, by condition (i). Finally, $I\leq III<\varepsilon$ since $\bar{r}$, $\bar{r}_\varepsilon$ represent averages of r, r_ε over sets $A(x)$. ■

9.2 Universal consistency

Histogram

The cubic histogram rule partitions $\mathbb{R}^d$ into cubes of the same size, and makes a decision according to the majority vote among the Y_i's such that the corresponding X_i falls in the same cube as X. Formally, let $\mathcal{P}_n=\{A_{n1,}A_{n2},\ldots\}$ be a partition of $\mathbb{R}^d$ into cubes of size $h_n>0$. More precisely, if $U=[0,1)^{\times d}$ is the (half-open) unit cube in $\mathbb{R}^d$ then the sets A_{n2} are of form $\mathbf{k}+h_nU$ where $\mathbf{k}=(k_1,\ldots,k_d)$ is a vector of integers. For every $x\in\mathbb{R}^d$ let $A_n(x)=A_{ni}$ if $x\in A_{ni}$. The histogram rule is defined as the pertaining special case of (118), i.e. by

$$h_n(x)=\left\{\begin{array}{l}0 \text{ if } \sum_{i=1}^{n}\mathbf{1}_{\{Y_i=1\}}\mathbf{1}_{A_n(x)}(X_i)\leq\sum_{i=1}^{n}\mathbf{1}_{\{Y_i=10\}}\mathbf{1}_{A_n(x)}(X_i) \\ 1 \text{ otherwise}\end{array}\right. .$$

The next theorem establishes universal consistency of certain cubic histogram rules (cf [DGL], p. 96).

9.2 Theorem *If $h_n\to 0$ and $nh_n^d\to\infty$ as $n\to\infty$ then the cubic histogram rule is universally consistent.*

Proof. We check the two simple conditions of of Theorem 9.1. Clearly, the diameter of each cell is $\sqrt{d}h^d$. Therefore condition (i) follows trivially:

$$\operatorname{diam}(A(X))\leq\sqrt{d}h^d\to 0.$$

To show condition (ii), we need to prove that for any $M<\infty$, $P\left(N(X)\leq M\right)\to 0$. Let S be an arbitrary ball centered at the origin. Then the number of cells intersecting S is not more than c_1+c_2/h^d for some positive constants c_1,c_2. Then

$$P\left(N(X)\leq M\right)\leq\sum_{j:A_{nj}\cap S\neq\emptyset}P\left(X\in A_{nj},N(X)\leq M\right)+P\left(X\in S^c\right)$$

$$\leq \sum_{j:A_{nj}\cap S\neq\emptyset,\ \mu(A_{nj})\leq 2M/n} \mu\left(A_{nj}\right) + \sum_{j:A_{nj}\cap S\neq\emptyset,\ \mu(A_{nj})>2M/n} \mu\left(A_{nj}\right) P\left(n\mu_n(A_{nj}) \leq M\right) + \mu(S^c)$$

(for the second term above, recall that X and $X_1,\ldots,X_n$ are independent),

$$\leq \frac{2M}{n}\left(c_1 + \frac{c_2}{h^d}\right) + \sum_{j:A_{nj}\cap S\neq\emptyset,\ \mu(A_{nj})>2M/n} \mu\left(A_{nj}\right) P\left(\mu_n(A_{nj}) - E\mu_n(A_{nj}) \leq M/n - \mu(A_{nj})\right) + \mu(S^c)$$

$$\leq \frac{2M}{n}\left(c_1 + \frac{c_2}{h^d}\right) + \sum_{j:A_{nj}\cap S\neq\emptyset,\ \mu(A_{nj})>2M/n} \mu\left(A_{nj}\right) P\left(\mu_n(A_{nj}) - E\mu_n(A_{nj}) \leq \frac{-\mu(A_{nj})}{2}\right) + \mu(S^c)$$

and applying Chebyshev's inequality

$$\leq \frac{2M}{n}\left(c_1 + \frac{c_2}{h^d}\right) + \sum_{j:A_{nj}\cap S\neq\emptyset,\ \mu(A_{nj})>2M/n} 4\mu\left(A_{nj}\right) \frac{\operatorname{Var}\left(\mu_n(A_{nj})\right)}{\left(\mu(A_{nj})\right)^2} + \mu(S^c)$$

$$\frac{2M}{n}\left(c_1 + \frac{c_2}{h^d}\right) + \sum_{j:A_{nj}\cap S\neq\emptyset,\ \mu(A_{nj})>2M/n} 4\mu\left(A_{nj}\right) \frac{1}{n\mu(A_{nj})} + \mu(S^c)$$

$$\leq \frac{2M+4}{n}\left(c_1 + \frac{c_2}{h^d}\right) + \mu(S^c) \to \mu(S^c)$$

by the condition $nh^d \to \infty$. Since S is arbitrary, the proof is complete. ■

k-nearest neighbor rule

This simple classifier is conceptually related to the histogram. Suppose a prediction of Y is to be made given $X = x$, on the basis of a training set $T_n = ((X_1, Y_1), \ldots, (X_n, Y_n))$. first the data are ordered according to increasing Euclidean distances of the X_j's to x:

$$\left((X_{(1)}(x), Y_{(1)}(x)), \ldots, (X_{(n)}(x), Y_{(n)}(x))\right),$$

that is, $X_{(i)}(x)$ is the i-th nearest neighbor of x among the points $X_1, \ldots, X_n$. Distance ties are broken by comparing indices, that is, in case of $\|X_i - x\| = \|X_j - x\|$, X_i is considered to be "closer" to x if $i < j$.
The k-nearest neighbor (k-NN) classification rule is defined as

$$h_n(x) = \begin{cases} 0 \text{ if } \sum_{i=1}^k \mathbf{1}_{\{Y_{(i)}(x)=1\}} \leq \sum_{i=1}^k \mathbf{1}_{\{Y_{(i)}(x)=0\}} \\ 1 \text{ otherwise} \end{cases}.$$

In other words, $h_n(x)$ is a majority vote among the labels of the k nearest neighbors of x.

9.3 Theorem *If $k \to \infty$ and $k/n \to 0$ then the $k-$NN rule is universally consistent.*

For a proof see Theorem 6.4 in [DGL], p. 101.

9.3 Classification and Regression Trees (CART)

Let us return to the similarity between classification and regression. Suppose we have an i.i.d. sample (X_i, Y_i), $i = 1, \ldots, n$ where $X_i \in \mathbb{R}^d$ and Y_i are real valued. In classification, the Y_i would be 0-1 valued for 2 classes. Suppose we are interested in estimating the regression function

$$r(x) = E\left(Y|X = x\right) \tag{128}$$

and we try to fit a family of functions $f_\vartheta(x)$, $\vartheta \in \Theta$ to the observed values (X_i, Y_i), $i = 1, \ldots, n$. As explained, in the classification case, for a ("new") test case x we would predict $Y = 1$ if the estimated $\hat{\vartheta}$ results in $f_{\hat{\vartheta}}(x) > 1/2$. Let us assume henceforth that X_i take values in a set $S \subset \mathbb{R}^d$, assumed compact.
In the histogram method for regression, we select a partition of the set S into subsets A_j, $j = 1, \ldots, m$ and the family of functions is piecewise constant on the sets A_j. Then ϑ may be taken as the vector of values $(\vartheta_j,\ j = 1, \ldots, m)$ of the function such that $f_\vartheta(x) = \vartheta_j$ for $x \in A_j$. The least squares principle may be used to obtain an estimate of ϑ

$$\min_\vartheta \sum_{i=1}^n \left(Y_i - f_\vartheta(X_i)\right)^2$$

It is clear that this yields the average of the $Y_i \in A_j$ as an estimate of ϑ_j:

$$\hat{\vartheta}_j := \bar{Y}_j := \frac{1}{n_j} \sum_{i=1}^n Y_i \mathbf{1}_{A_j}(X_i) \text{ where } n_j = \sum_{i=1}^n \mathbf{1}_{A_j}(X_i).$$

Thus the histogram method for regression also results from the least squares principle, once the partition A_j, $j = 1, \ldots, m$ is chosen. The main problem with histogram estimation is the choice of this partition.
The histogram rule for classification has been treated in section 9.2. Here, once the partition is chosen, each A_j is classified acording to whether $\bar{Y}_j \geq 1/2$, i.e. by a "majority vote" of the X_j in the given set A_j. Other nonparametric regression estimators also have their classification analogs, such as nearest neighbor methods.
Let us consider some algorithms in which the partition is chosen recursively, depending on the data.

Partition choice for regression. Assume first $S = [0, 1]$ and we consider a histogram with $m = 2$, where $A_1 = [0, s]$ and $A_2 = (s, 1]$, but where we try to select also the points s from the data. Now ϑ may be taken as the vector of values $(\vartheta_1, \vartheta_2, s)$ and the least squares principle gives (by minimizing first over ϑ_1, ϑ_2 for given s)

$$\min_\vartheta \sum_{i=1}^n \left(Y_i - f_\vartheta(X_i)\right)^2 = \min_s \left(\min_{c_1} \sum_{X_i \leq s} (Y_i - c_1)^2 + \min_{c_2} \sum_{X_i > s} (Y_i - c_2)^2 \right),$$

where the solutions of the minimum problem over c_j are $\bar{Y}_{j,s}$, $j = 1, 2$ i.e. the averages of Y_i over the sets $X_i \leq s$ and $X_i > s$. The function

$$Q(s) = \sum_{X_i \leq s} (Y_i - \bar{Y}_{1,s})^2 + \min_c \sum_{X_i > s} (Y_i - \bar{Y}_{2,s})^2 \tag{129}$$

is the criterion to be minimized over s, and can be seen as an *impurity measure* for a given split point s. For instance, it is 0 if all values Y_i to the right of s are equal and also all values Y_i to the left of s are equal.

Impurity for classification. Again, $S = [0, 1]$ but now the data Y_i are 0-1 valued. We have to choose a split point s for a histogram on a partition where $A_1 = [0, s]$ and $A_2 = (s, 1]$, such that each side is classified according to a majority vote. The essence of the CART method (classification and regression trees, Breiman et al. [BFOS])[8] to be discussed is the choice of a split point s using an impurity measure. Intuitively, an impurity function measures the "homogeneity" of data on each side of the split point s. Let $\hat{p}_{1,s}$ be the proportion of $1'^s$ in $\{Y_i : X_i \leq s\}$ and $\hat{p}_{2,s}$ be the proportion of $1'^s$ in $\{Y_i : X_i > s\}$. An impurity measure for the data left of the split point s (that is for $\{Y_i : X_i \leq s\}$) is a function ψ of $p = \hat{p}_{1,s}$ with the following properties:

1. $\psi\left(\frac{1}{2}, \frac{1}{2}\right) \geq \psi(p, 1-p)$ for any $p \in [0, 1]$
2. $\psi(0, 1) = \psi(1, 0) = 0$,
3. $\psi(p, 1-p)$ increases in p on on $[0, 1/2]$ and decreases in p on $[1/2, 1]$.

For a given function ψ to be used, the impurity function to be minimized for a possible split s is

$$I(s) = n_{1,s}\psi(\hat{p}_{1,s}, 1 - \hat{p}_{1,s}) + n_{2,s}\psi(\hat{p}_{2,s}, 1 - \hat{p}_{2,s}) \tag{130}$$

where $n_{1,s}$, $n_{2,s}$ are the sizes of the two classes:

$$n_{1,s} = \sum_{i=1}^{n} \mathbf{1}_{[0,s]}(X_i), \; n_{2,s} = \sum_{i=1}^{n} \mathbf{1}_{(s,1]}(X_i).$$

For a "perfect" split, where $\hat{p}_{1,s} = 1$ and $\hat{p}_{2,s} = 0$ we have $I(s) = 0$ and also for the case $\hat{p}_{1,s} = 0$ and $\hat{p}_{2,s} = 1$. If $\hat{p}_{1,s} = \hat{p}_{2,s} = 1/2$ (the worst case split) we have $I(s) = n/2$, the maximal value. Note that the weighting of $\psi(\hat{p}_{i,s}, 1 - \hat{p}_{i,s})$ by n_i , $i = 1, 2$ makes sense, as seen by the following example. Suppose there is an $s_0 \in (0, 1)$ such that $\hat{p}_{2,s_0} = 1/2$ and $\hat{p}_{1,s_0} = 0$, and n_1 is large while n_2 is small. Intuitively, a choice of split at $s = s_0$ does make sense, as the class to the left is completey "pure" and contains most of the data. If were to use the unweighted sum

$$I_0(s) = \psi(\hat{p}_{1,s}, 1 - \hat{p}_{1,s}) + \psi(\hat{p}_{2,s}, 1 - \hat{p}_{2,s})$$

[8]Breiman, L., Friedman, J., Olshen, R., Stone, C. (1984). *Classification and Regression Trees.* Wadsworth, Belmont, CA.

for a choice of split point s then then $I_0(s_0) = \psi(1/2, 1/2)$ and we could decrease $I_0(s)$ by choosing $s < s_0$ (then $\hat{p}_{2,s} < 1/2$ while $\hat{p}_{1,s}$ remains 0). We would arrive at a non-sensible split where s is very small. With the weighted criterion (130) however , a high impurity $\psi(\hat{p}_{2,s}, 1 - \hat{p}_{2,s})$ at $s = s_0$ is counterbalanced by a small $n_{2,s}$ so that a choice $s < s_0$ is not obviously favored.
Examples of functions ψ include:

1. The *entropy function* $\psi(p, 1-p) = -p \log p - (1-p) \log(1-p)$
2. The *Gini function* $\psi(p, 1-p) = 2p(1-p)$.
3. The *probability of misclassification* $\psi(p, 1-p) = \min(p, 1-p)$. Indeed, if on the left side we have $\hat{p}_{1,s} > 1/2$ then the left side is classified as "1". The proportion of 0's here is $1 - p_{1,s}$, and selecting a random point from the left side, the probability of a 0-point (that is a misclassified point) is $1 - p_{1,s}$. Analogously if $\hat{p}_{1,s} \leq 1/2$ then the left side is classified as "0" etc., so this measure is $\min(p, 1-p)$ (the proportion of points which are in the minority).

CART algorithm: forward stage (tree growing). To build a tree by successive, data-driven splitting of rectangles, for multivariate $X_i = (X_i^{(1)}, \ldots, X_i^{(d)}) \in \mathbb{R}^d$ one takes a split point s for the k-th component and a criterion analogous to (130), *applied to the k-th components $X_i^{(k)}$*, $i = 1, \ldots, n$. Successive splits on different components produce rectangles, which are split further. To give the a recursive formulation of the algorithm, suppose a rectangle $R \subset \mathbb{R}^d$ is considered for splitting along one of the coordinates, that is, only points $X_i \in R$ are taken into consideration. Define

$$n_{1,R,k,s} = \sum_{i=1}^{n} \mathbf{1}_{(-\infty,s]}(X_i^{(k)}) \mathbf{1}_R(X_i),\ n_{2,R,k,s} = \sum_{i=1}^{n} \mathbf{1}_{(s,\infty]}(X_i^{(k)}) \mathbf{1}_R(X_i), \tag{131}$$

$$\hat{p}_{1,R,k,s} = \frac{1}{n_{1,R,k,s}} \sum_{i=1}^{n} Y_i \mathbf{1}_{(-\infty,s]}(X_i^{(k)}) \mathbf{1}_R(X_i),\ \hat{p}_{2,R,k,s} = \frac{1}{n_{2,R,k,s}} \sum_{i=1}^{n} Y_i \mathbf{1}_{(s,\infty]}(X_i^{(k)}) \mathbf{1}_R(X_i) \tag{132}$$

and the impurity measure

$$I(R,k,s) = n_{1,R,k,s} \psi(\hat{p}_{1,R,k,s}, 1 - \hat{p}_{1,R,k,s}) + n_{2,R,k,s} \psi(\hat{p}_{2,R,k,s}, 1 - \hat{p}_{2,R,k,s}) \tag{133}$$

The algorithm proceeeds in a greedy manner, where, if at a given stage one has a collection of rectangles $R \in \mathcal{R}$, one seeks the minimum

$$\min_{R \in \mathcal{R}, k=1,\ldots,d,s} I(R,k,s)$$

where the split point s conforms to the rectangle R (that is, $r_l^{(k)} < s < r_u^{(k)}$ if $R = \prod_{j=1}^{d} \left[r_l^{(j)}, r_u^{(j)} \right)$).
If $\left(\hat{R}, \hat{k}, \hat{s} \right)$ is a minimizer then the rectangle $\hat{R}$ is split along coordinate $\hat{k}$ at point $\hat{s}$, and $\hat{R}$ gives rise to two new rectangles.

In this manner, a *binary classification tree* is obtained. The splits are continued until all sets contain less than C elements, eg. $C = 5$. One could imagine another stopping criterion, for instance one where the tree growing stops if no sensible gain in purity is obtained. However the CART algorithm does not use such a criterion; it continues regardless of purity gain until all rectangles contain few elements. Thus the resulting tree can be expected to have far too many leaves (overfit); it has to be cut back (pruned) in a second stage. The reason for this will be explained below.

CART algorithm: backward stage (pruning). We introduce some terminology for trees: the top of a binary tree is called the *root*. Each *node* has either no child (in that case it is called a *terminal node* or *leaf*), a left child, a right child or a left child and a right child. Each node is the root of a tree itself.
Let τ_0 be the binary tree obtained above in the first stage. We define a subtree $\tau \subset \tau_0$ to be any tree that can be obtained by pruning τ_0, that is collapsing of its internal (non-terminal) nodes (so that this node becomes terminal). We index terminal nodes by m, with node m region R_m. Let $|\tau|$ be the number of terminal nodes in τ. Letting

$$n_m = \sum_{i=1}^{n} \mathbf{1}_{R_m}(X_i),$$
$$\hat{p}_m = \frac{1}{n_m} \sum_{i=1}^{n} Y_i \mathbf{1}_{R_m}(X_i)$$

and for $\psi(p) = \min(p, 1-p)$ (misclassification rate)

$$\tilde{I}(m,\tau) = n_m \psi\left(\hat{p}_m, 1-\hat{p}_m\right)$$

we define the cost complexity criterion

$$C_\alpha(\tau) = \sum_{m=1}^{|\tau|} \tilde{I}(m,\tau) + \alpha\,|\tau|$$

The idea is to find, for each α, the subtree $\tau_\alpha \subset \tau_0$ to minimize $C_\alpha(\tau)$. The tuning parameter $\alpha \geq 0$ governs the tradeoff between tree size and its goodness of fit to the data. Large values of α result in smaller trees τ_α and conversely for smaller values of α. With $\alpha = 0$ the solution is the full tree τ_0 (or among all possible trees on the data, it would be a tree where each pair (X_i, Y_i) has its own region R_m, with zero classification error $\hat{p}_m$. The tree τ_0 stopped at 5 points per region intuitively comes close to that).
The degree of pruning (tuning parameter α for the penalty) is chosen by cross validation; the choice is an $\hat{\alpha}$. The final tree is the one pruned according to the cross validated $\hat{\alpha}$. For further discussion of cost complexity pruning and "greedy histograms", see Klemelä [Kl][9]. A justification of the pruning

[9] Klemelä, J. (2009). *Smoothing of Multivariate Data.*Wiley, New York.

algorithm as a model choice procedure (with oracle type inequalities) in the regression context is given by Sauvé [4].

Reason for the two stage procedure. There are configurations of data (that is laws $\mathcal{L}(X, Y)$) where initially, no split perpendicular to the axes seems reasonable, but there are splits below the first stage which are very good. Consider two class-conditional laws $P_0 = \mathcal{L}(X|Y = 0)$, $P_1 = \mathcal{L}(X|Y = 1)$ defined as follows. Divide the unit cube $U = [0, 1] \times [0, 1]$ into two regions: $U_0 = ([0, 1/2] \times [0, 1/2]) \cup ([1/2, 1] \times [1/2, 1])$ and $U_1 = ([1/2, 1] \times [0, 1/2]) \cup ([0, 1/2] \times [1/2, 1])$ and let P_i be the uniform law on U_i, $i = 1, 2$. Set $P(Y = 0) = 1/2$. Then, for large sample size, both $\hat{p}_{1,U,k,s}$ and $\hat{p}_{2,U,k,s}$ are close to $1/2$ for $k = 1, 2$ and all possible split points $s \in (0, 1)$. Thus the selection of the first split seems to be arbitrary; none of these appears really good. However after the first split has been made, the two resulting rectangles can be split in a meaningful way (picture).
For consistency results see Olshen [3][10] and references therein, as well as discussions in [DGL], sec. 20.9.
An example: spam data (HTF, p. 313).

[10] Olshen, R. (2007). Tree-structured regression and the differentiation of integrals. Ann. Statist. **35** 1-12

10 Boosting

Boosting is based on the observation that finding many rough prediction rules can be much easier than finding a single highly accurate predictor. In the boosting approach, we start with a " base" learning algorithm, capable of producing moderately accurate prediction rules (perhaps rules that do slightly better than random guessing). We call this algorithm on our training set, producing a rough prediction rule. We then tune the importance, or weight, of our training data, assigning higher weights to those data misclassified by our first rough predictor, and assigning lower weight to the data classified correctly. We repeat this procedure, producing many rough classifiers. Heuristically, each new classifier should perform well on the data misclassified by the previous predictors. At the end of this iteration, the algorithm must combine the set of weak classifiers into a single, and hopefully accurate, prediction rule for the entire data set.
There are two fundamental questions in this approach: first, how should the distribution on the training data be chosen on each round, and second, how should the weak rules be combined into a single rule? There is also the question of what to use for the base learning algorithm, which we will not address in this discussion. The goal here is to develop a general method for combining weak learners into a single good predictor.

10.1 Weighted training data

The AdaBoost algorithm is based on the notion of a training set equipped with weights w_i, $i = 1, \dots, n$. The algorithm uses a "base learner" (classifier) which is repeatedly applied to the same training set (x_i, y_i), $i = 1, \dots, n$ but with different weights each time. To understand that, we first have to clarify what does it mean to apply a classifier to a training set equipped with weights. We focus first on the tree method, and review the unweighted case first.
Unweighted impurity measure. Suppose for ease of notation all x_i are on the interval $[0, 1]$ and let $\psi(p, 1-p) = 2p(1-p)$ be the Gini index. The impurity function to be minimized for a possible split s is

$$I(s) = n_{1,s}\psi\left(\hat{p}_{1,s}, 1 - \hat{p}_{1,s}\right) + n_{2,s}\psi\left(\hat{p}_{2,s}, 1 - \hat{p}_{2,s}\right) \tag{134}$$

where $\hat{p}_{1,s}$ is the proportion of $1'^s$ in $\{Y_i : X_i \le s\}$ and $\hat{p}_{2,s}$ is the proportion of $1'^s$ in $\{Y_i : X_i > s\}$, and $n_{1,s}$, $n_{2,s}$ are the sizes of the two classes:

$$n_{1,s} = \sum_{i=1}^{n} \mathbf{1}_{[0,s]}(x_i),\ \hat{p}_{1,s} = \frac{1}{n_{1,s}} \sum_{i=1}^{n} y_i \mathbf{1}_{[0,s]}(x_i),$$
$$n_{2,s} = \sum_{i=1}^{n} \mathbf{1}_{(s,1]}(x_i),\ \hat{p}_{2,s} = \frac{1}{n_{2,s}} \sum_{i=1}^{n} y_i \mathbf{1}_{[0,s]}(x_i).$$

Weighted impurity measures. Let w_i, $i = 1, \dots, n$ be nonnegative with $\sum_{i=1}^{n} w_i = 1$. Define the weighted analogs of $\hat{p}_{1,s}$ and $n_{1,s}$ as

$$n_{1,s} = \sum_{i=1}^{n} w_i \mathbf{1}_{[0,s]}(x_i),\ \hat{p}_{1,s} = \frac{1}{n_{1,s}} \sum_{i=1}^{n} w_i y_i \mathbf{1}_{[0,s]}(x_i)$$

and analogously for $\hat{p}_{2,s}$ and $n_{2,s}$. The impurity function to be minimized for a possible split s is again (134) but with the modified $n_{i,s}, \hat{p}_{i,s}$, $i = 1, 2$. The same principle applies when for multivariate $x_i = (x_i^{(1)}, \ldots, x_i^{(d)}) \in \mathbb{R}^d$ one takes a split point s for the k-th component and a criterion analogous to (130), applied to the k-th components $x_i^{(k)}$, $i = 1, \ldots, n$. This means that when considering a split of a rectangle R along the k-th component, one defines the weighted quantities as $n_{1,R,k,s} = \sum_{i=1}^n w_i \mathbf{1}_{(-\infty,s]}(x_i^{(k)}) \mathbf{1}_R(x_i)$ etc. in analogy to (131), (132).
Neural network training. Suppose we have K classes and we use sum-of-squared errors as our measure of fit (error function): for a training set (x_i, y_i) $i = 1, \ldots, n$ where $x_i \in \mathbb{R}^d$ and $y_i = (y_i^{(1)}, \ldots, y_i^{(K)})$ is an indicator vector (where $y_i^{(k)} = 1$ if x_i is in class k, 0 otherwise)

$$R(\theta) = \sum_{i=1}^n \sum_{k=1}^K \left(y_i^{(k)} - \psi_k(x_i) \right)^2$$

(cf. (109)) and the corresponding classifier is $G(x) = \arg\max_{k \in \{1,\ldots,K\}} \psi_k(x)$. Here θ represents the parameters of the neural network (that is, of the the vector valued function $\psi_k(x)$, $k = 1, \ldots, K.$). The weighted variant of $R(\theta)$ is

$$R(\theta) = \sum_{i=1}^n \sum_{k=1}^K w_i \left(y_i^{(k)} - \psi_k(x_i) \right)^2$$

and the weighted variant of the cross-entropy (110) is

$$R(\theta) = -\sum_{i=1}^n \sum_{k=1}^K w_i y_i^{(k)} \log \psi_k(x_i).$$

10.2 The AdaBoost algorithm

Suppose we are given a training set $T_n = ((x_i, y_i), i = 1, \ldots, n)$ where the labels y_i are in $\{-1, 1\}$. The x_i can be in a general set $\mathcal{X}$. The following algorithm trains M weak learners, and combines them into a final prediction rule. We assume all classifiers $G(x)$ taskes values in $\{-1, 1\}$.

Algorithm AdaBoost.M1. (Freund & Schapire 1997)**.**

(1) Initialize the observation weights $w_i^{(1)} = 1/n$, $i = 1, 2, \ldots, n$.

(2) For $m = 1$ to M:

- **(a)** Fit a classifier $G_m(x)$ to the training data using weights $w_i^{(m)}$.
- **(b)** Compute.

$$\mathrm{err}_m = \frac{\sum_{i=1}^n w_i^{(m)} \mathbf{1}\left(y_i \neq G_m(x_i)\right)}{\sum_{i=1}^n w_i^{(m)}}$$

(c) Compute

$$\alpha_m = \log\left(\frac{1-\mathrm{err}_m}{\mathrm{err}_m}\right)$$

(d) Set $w_i^{(m+1)} := w_i^{(m)} \exp\left[\alpha_m \mathbf{1}\left(y_i \neq G_m(x_i)\right)\right],\ i = 1, 2, \dots, n$

(3) Output

$$G(x) = \mathrm{sign}\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right) \tag{135}$$

.

We assume that the initial classifier $G_1(x)$ has error $\mathrm{err}_1 < 1/2$. We can interpret this as meaning that $G_1(x)$ does slightly better than random guessing. From this assumption, we know that$(1 - \mathrm{err}_1)/\mathrm{err}_1 > 1$, so

1. $\alpha_1 = \log\left(\frac{1-\mathrm{err}_m}{\mathrm{err}_m}\right) > 0$, ie $G_1(x)$ has positive weight in the final classifier. In general, any weak learning that does slightly better than guessing will have positive weight in the final classifier.
2. $w_i^{(2)} \longleftarrow w_i^{(1)} \left(1 - \mathrm{err}_1\right)/\mathrm{err}_1$ for misclassified examples (x_i, y_i), and $w_i^{(2)} \longleftarrow w_i^{(1)}$ for correctly classified examples. This means that the weight of misclassified examples increases, and the weight of correctly classified examples remains the same.

Training error analysis

Let $\mathrm{err}_m = 1/2 - \gamma_m$, and suppose that each base learner is slightly better than random guessing, so that $\gamma_m > \gamma > 0$ for $m = 1, \dots, M$. In this case, we will show that the training error of the final classifier is exponentially small in the number of iterations of the AdaBoost algorithm. We will prove this result in a series of three steps.
First, define $f(x) = \sum_{m=1}^{M} \alpha_m G_m(x)$, so that $G(x) = \mathrm{sign}\left(f(x)\right)$, and define $W_m = \sum_{i=1}^{n} w_i^{(m)}$.

10.1 Lemma $W_m = \displaystyle\prod_{i=1}^{m-1} 2\left(1 - \mathrm{err}_i\right).$

Proof. This follows from the recursion

$$\begin{aligned}
W_{k+1} &= \sum_{i=1}^{n} w_i^{(k)} \exp\left(\alpha_k \mathbf{1}_{\{y_i \neq G_k(x_i)\}}\right) \\
&= \left(\frac{1-\mathrm{err}_k}{\mathrm{err}_k}\right) \sum_{i:\ y_i \neq G_k(x_i)} w_i^{(k)} + \sum_{i:\ y_i = G_k(x_i)} w_i^{(k)} \\
&= \left(\frac{1-\mathrm{err}_k}{\mathrm{err}_k}\right) \left(W_k \cdot \mathrm{err}_k\right) + \left(W_k - W_k \cdot \mathrm{err}_k\right) \\
&= 2\left(1 - \mathrm{err}_k\right) \cdot W_k
\end{aligned}$$

and the base $W_1 = 1$. ■

10.2 Lemma

$$w_i^{(M)} = \frac{1}{n} \exp\left(\frac{-y_i f(x_i)}{2}\right) \exp\left(\frac{1}{2}\sum_{m=1}^{M} \alpha_m\right)$$

Proof. Recall that $f(x) = \sum_{m=1}^{M} \alpha_m G_m(x)$ and the final classifier is $G(x) = \operatorname{sign}(f(x))$. Also note that for any classifier $G(x)$,

$$-yG(x) = 2\mathbf{1}_{\{y \neq G(x)\}} - 1.$$

This gives the relation

$$\mathbf{1}_{\{y \neq G(x)\}} = \frac{-yG(x) + 1}{2}.$$

Now by "unwrapping" the iterative update of $w_i^{(m)}$, we can use the above observation to calculate

$$\begin{aligned}
w_i^{(M)} &= \frac{1}{n} \exp\left(\sum_{m=1}^{M} \alpha_m \mathbf{1}_{\{y_i \neq G_m(x_i)\}}\right) \\
&= \frac{1}{n} \exp\left(\sum_{m=1}^{M} \alpha_m \left(\frac{-y_i G_m(x_i) + 1}{2}\right)\right) \\
&= \frac{1}{n} \exp\left(\frac{-y_i}{2} \sum_{m=1}^{M} \alpha_m G_m(x_i)\right) \exp\left(\frac{1}{2}\sum_{m=1}^{M} \alpha_m\right).
\end{aligned}$$

■

We can now combine these results to estimate the training error of the final classifier $G(x)$, defined as

$$\tau(G) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i \neq G(x_i)\}}.$$

10.3 Theorem

$$\tau(G) \leq \exp\left(-2(M-1)\gamma^2\right).$$

Proof. Note that if $G(x) \neq y$, then $yf(x) \leq 0$, so $-yf(x)/2 \geq 0$ and we have $\exp(-yf(x)/2) \geq 1$. We can use this to estimate the training error of the final classifier $G(x)$:

$$\tau(G) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i \neq G(x_i)\}}. \leq \frac{1}{n} \sum_{i=1}^{n} \exp\left(\frac{-y_i f(x_i)}{2}\right)$$

$$= \left(\sum_{i=1}^{n} w_i^{(M)}\right) \exp\left(-\frac{1}{2}\sum_{m=1}^{M} \alpha_m\right)$$

$$\leq \left(\prod_{i=1}^{M-1} 2\left(1-\mathrm{err}_m\right)\right)\left(\prod_{i=1}^{M}\sqrt{\frac{\mathrm{err}_m}{1-\mathrm{err}_m}}\right)$$

$$\leq \prod_{i=1}^{M-1} 2\sqrt{\mathrm{err}_m\left(1-\mathrm{err}_m\right)}$$

where the last inequality holds due to

$$\sqrt{\frac{\mathrm{err}_M}{1-\mathrm{err}_M}} = \sqrt{\frac{1/2-\gamma_M}{1/2+\gamma_M}} \leq 1.$$

Now using the assumption that $\mathrm{err}_m = 1/2 - \gamma_m$ and the fact that for all real numbers x one has $\exp\left(-2x^2\right) \geq 1 - 2x^2$, we obtain

$$\begin{aligned}\tau(G) &\leq \prod_{i=1}^{M-1} 2\sqrt{\mathrm{err}_m\left(1-\mathrm{err}_m\right)} \\ &= \prod_{i=1}^{M-1} \sqrt{1-4\gamma_m^2} \\ &\leq \exp\left(-2\sum_{m=1}^{M-1}\gamma_m^2\right) \leq \exp\left(-2(M-1)\gamma^2\right)\end{aligned}$$

since $\gamma_m > \gamma > 0$ for $m = 1, \ldots, M$ by assumption. ■

A simulation example

Let the distribution of $X = \left(X^{(1)}, \ldots, X^{(10)}\right)$ be $N_{10}(0, I)$ and let the conditional law $\mathcal{L}(Y|X)$ be given by

$$\mathcal{L}(Y|X=x) = \left\{\begin{array}{l} P(Y=1|X=x) = 1 \text{ if } \|X\|^2 > \chi^2_{10}(0.5) \\ P(Y=-1|X=x) = 1 \text{ otherwise} \end{array}\right.$$

where $\chi^2_{10}(0.5) = 9.34$ is the median (0.5-quantile) of the χ^2_{10}-distribution. Thus $P(Y=1) = P(Y=-1) = 1/2$ and the class-conditional laws $\mathcal{L}(X|Y=1)$, $\mathcal{L}(X|Y=-1)$ have disjoint support (on $\left\{X : \|X\|^2 > \chi^2_{10}(0.5)\right\}$ and $\left\{X : \|X\|^2 \leq \chi^2_{10}(0.5)\right\}$ respectively). There were $n = 2000$ training cases generated and also $10,000$ test observations. The base classifier $G_1(x)$ is a "stump": a rudimentary classification tree with two terminal nodes, such that for some $j \in \{1, \ldots, k\}$ and $s \in \mathbb{R}$ and $z \in \{-1, 1\}$

$$G_1(x) = \left\{\begin{array}{l} z \text{ if } x^{(j)} \leq s \\ -z \text{ if } x^{(j)} > s \end{array}\right. .$$

The test error rate of G_1 was 46%. We can easily compute the appriximate error rate of a stump with $s = 9.34$, $z = -1$ and $k = 1$ (or arbirtrary). Indeed, all X_i with $\|X_i\| \leq s$ are correctly classified as -1, and also all X_i with $X_i^{(1)} > s$ are correctly classified as 1. Thus the error rate is (for large n)

$$\begin{aligned} P\left(\|X\|^2 > s, X^{(1)} < s\right) &= 1 - P\left(\|X\|^2 < s\right) - P\left(X^{(1)} > 9.34\right) \\ &= 1/2 - \varepsilon \end{aligned}$$

for $\varepsilon = P\left(Z > 9.34\right)$ where Z is standard normal.
After $M = 400$ booosting iterations, the test error was 12.2%. For a comparison, a CART tree with backpruning achieved a 26% error rate. The boosted classifier

$$G(x) = \operatorname{sign}\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right)$$

should be constant on rectangles, and obviously the sum of the rectangles where -1 is classified approximately cover the ball $\left\{X : \|X\|^2 \le \chi^2_{10}(0.5)\right\}$.

10.3 Boosting as additive modeling

The goal of boosting is to express an accurate classifier $G(x)$ as a weighted sum of weak learners $G_m(x)$. We can view this as a special case of expressing an arbitrary function $f(x)$ as a weighted sum of elementary "basis" functions

$$f(x) = \sum_{m=1}^{M} \beta_m b(x, \gamma_m)$$

where the $b(x, \gamma_m)$ are the basis functions depending on some parameters γ_m and the β_m are the expansion coefficients. This is a common idea in many learning and nonparametric regression techniques, for example

- In single hidden layer neural networks ,$b(x; \gamma) = \sigma\left(\gamma_0 + \gamma_1^\top x\right)$ where $\sigma(\cdot)$ is a sigmoidal function and γ parameterizes a linear combination of the input variables.
- For wavelets, γ parametrizes the location and scale shifts of a "mother" wavelet. However in the standard orthogonal series estimator for regression, the parameters γ_m are fixed (not estimated), only the coefficients β_m are estimated in a linear or nonlinear way. Another example is classical Fourier series where $b(x; \gamma_m) = \phi_m(x)$ is the m-th element of an orthogonal basis in function space.
- MARS (not discussed in this course) uses truncated power spline basis functions where γ parameterizes the variables and values for the knots.
- For trees, γ parameterizes the split variables (k-th component) and split points at the internal nodes, and the predictions at the terminal nodes.

A typical approach to this problem is to minimize a loss function averaged over the training data

$$\min_{\beta_m,\gamma_m} \sum_{i=1}^{n} L\left(y_i, \sum_{m=1}^{M} \beta_m b(x_i,\gamma_m)\right) \tag{136}$$

. where the loss function L could be squared-error or a likelihood-based loss function. However, for many loss functions, this approach requires computationally intensive optimization techniques, so we consider the simpler subproblem of fitting just a single basis function

$$\min_{\beta,\gamma} \sum_{i=1}^{n} L\left(y_i, \beta b(x_i,\gamma)\right).$$

This is *Forward Stagewise Additive Modeling (FSAM)*, in which we approximate the solution to (136) by sequentially adding new basis functions to the expansion without adjusting the previously added basis functions or coefficients. In the algorithm below, at iteration m, we solve for the optimal basis function$b(x,\gamma_m)$ and coefficient β_m given the current expansion $f_{m-1}(x)$. We then update the current expansion and repeat the process.

Algorithm FSAM.

(1) Initialize $f_0(x) = 0$.

(2) For $m = 1$ to M:

- **(a)** Compute.

$$(\beta_m, \gamma_m) = \arg\min_{\beta,\gamma} \sum_{i=1}^{N} L\left(y_i, f_{m-1}(x_i) + \beta b(x_i;\gamma)\right)$$

- **(b)** Set $f_m(x) = f_{m-1}(x) + \beta_m b(x;\gamma_m)$

We can now see that the AdaBoost algorithm is a special case of FSAM, using the loss function

$$L(y, f(x)) = \exp\left(-yf(x)\right)$$

Note that if $f(x) = G(x)$ is a classifier taking values in $\{-1, 1\}$, then $L(y, f(x)) = e$ for misclassification and $L(y, 1(x)) = e^{-1}$ for correct classification.
So in the FSAM framework, we wish to solve

$$(\beta_m, G_m) = \arg\min_{\beta,G} \sum_{i=1}^{n} \exp\left[-y_i\left(f_{m-1}(x_i) + \beta G(x_i)\right)\right] \tag{137}$$

for the classifier G_m and corresponding coefficient β_m. Note that (137) is equivalent to

$$(\beta_m, G_m) = \arg\min_{\beta,G} \sum_{i=1}^{n} w_i^{(m)} \exp\left(-\beta y_i G(x_i)\right) \tag{138}$$

where $w_i^{(m)} = \exp\left(-y_i f_{m-1}(x_i)\right)$. Since each $w_i^{(m)}$ depends neither on β nor $G(x)$, it can be regarded as a weight applied to each observation. This weight depends on $f_{m-1}(x)$, so the weight values change with each iteration.

10.4 Lemma *For any value of $\beta > 0$, the solution to (138) for $G_m(x)$ is*

$$G_m = \arg\min_G \sum_{i=1}^N w_i^{(m)} \mathbf{1}_{\{y_i \neq G(x_i)\}}$$

which is the classifier that minimizes the weighted error rate in predicting y.

Proof. Since $y_i G(x_i) = -1$ for correct classification and $y_i G(x_i) = 1$ for misclassification, we can write the criterion in(138) as

$$\begin{aligned} & e^{-\beta} \sum_{y_i = G(x_i)} w_i^{(m)} + e^{\beta} \sum_{y_i \neq G(x_i)} w_i^{(m)} \qquad (139) \\ &= \left(e^{\beta} - e^{-\beta}\right) \sum_{i=1}^N w_i^{(m)} \mathbf{1}_{\{y_i \neq G(x_i)\}} + e^{-\beta} \sum_{i=1}^N w_i^{(m)}. \end{aligned}$$

and see that G_m satisfies (138) if and only if it satisfies (137). ■

Now fixing this G_m and minimizing the criterion in (137) with respect to β, we obtain the solution

$$\beta_m = \frac{1}{2} \log \frac{1 - \mathrm{err}_m}{\mathrm{err}_m}$$

(by differentiating (139) and setting to zero) where err_m is the minimized error rate

$$\mathrm{err}_m = \frac{\sum_{i=1}^n w_i^{(m)} \mathbf{1}_{\{y_i \neq G(x_i)\}}}{\sum_{i=1}^n w_i^{(m)}}$$

The approximation is then updated

$$f_m(x) = .f_{m-1}(x) + \beta_m G_m(x).$$

This causes the weight for the next iteration to be

$$w_i^{(m+1)} = w_i^{(m)} \exp\left(-\beta_m y_i G_m(x_i)\right). \qquad (140)$$

Since we have $-y_i G_m(x_i) = 2 \cdot \mathbf{1}_{\{y_i \neq G(x_i)\}} - 1$, the update step (140) becomes

$$w_i^{(m+1)} = w_i^{(m)} \exp\left(-\alpha_m \mathbf{1}_{\{y_i \neq G(x_i)\}}\right) e^{-\beta_m}$$

where $\alpha_m = 2\beta_m$ as in step 2 (c) of the algorithm AdaBoost. The above calculation is equivalent to the update step in the AdaBoost algorithm, up to a constant factor $e^{-\beta_m}$, so we can interpret AdaBoost as minimizing the exponential loss function via FSAM.

The use of exponential loss in the context of additive modeling is justified by the simple reweighting AdaBoost algorithm. However, it is of interest to inquire about its statistical properties. What does it estimate and how well is it being estimated? The first question is answered by seeking its population minimizer.
It is easy to show (Friedman et al., 2000)

10.5 Lemma *We have*

$$f^*(x) = \arg\min_{f(x)} E_{Y|x} \exp\left(-Yf(x)\right) = \frac{1}{2}\log\frac{\Pr\left(Y=1|x\right)}{\Pr\left(Y=-1|x\right)}$$

or equivalently

$$\Pr\left(Y=1|x\right) = \frac{1}{1+\exp\left(-2f^*(x)\right)}.$$

Thus, the additive expansion produced by AdaBoost is estimating one-half the log-odds of $\Pr\left(Y=1|x\right)$. This justifies using its sign as the classification rule in (135).

11 Statistical performance of classifiers

Consider again a training sample $T_n = \{(X_i, Y_i), i = 1, \ldots, n\}$ where $X_i \in \mathbb{R}^d$, $Y_i \in \{0,1\}$ and $Z_i = (X_i, Y_i)$ are i.i.d. random vectors with common distribution Q on $\mathbb{R}^d \times \{0,1\}$. A classifier is a (measurable) function

$$h : \mathbb{R}^d \times \left\{\mathbb{R}^d \times \{0,1\}\right\}^{\times n} \to \{0,1\}$$

Classifiers f should be optimal with respect to the error probability for a "new", incoming test case (X, Y) of which only X is observed. For a given (fixed) training set t_n, the performance of a rule h is measured by the probability of error for a test case (X, Y) having joint distribution Q and $h(X, t_n)$ is used to predict Y :

$$P_Q\left(h(X, t_n) \neq Y\right) = P_Q\left(h \neq Y | T_n = t_n\right). \tag{141}$$

(Note that now we index expectations and probabilities by the pertaining unknown law $Q = \mathcal{L}(X, Y)$). We denote the above conditional probability as a random variable

$$L_Q(h) := P_Q\left(h \neq Y | T_n\right).$$

and sometimes we omit the lower index, writing $L(h) = L_Q(h)$. The final measure of performance for the rule h_n is obtained by averaging over the training set T_n:

$$E_Q L(h) = E_Q P_Q\left(h \neq Y | T_n\right) = P_Q\left(h \neq Y\right). \tag{142}$$

The classifier minimizing (141) for given t_n *among all* $h : \mathbb{R}^d \mapsto \{0,1\}$, does not depend on t_n but depends on Q and is called the Bayes classifier h_Q^*:

$$L_Q^* := P_Q\left(h_Q^*(X) \neq Y\right) = \inf_{h:\mathbb{R}^d \mapsto \{0,1\}} P_Q\left(h(X) \neq Y\right);$$

Having a training set, $T_n = \{Z_i, i = 1, \ldots, n\}$, , one might consider it desirable to approximate the Bayes classifier as much as possible. In principle statistics gives the methods to do that: estimate Q or estimate the regression function

$$r(x) = E_Q\left(Y | X = x\right)$$

consistently from the data, and build a classifier from there. Heuristically, such a sequence of classifiers h_n should be *consistent*:

$$L(h_n) \to_P L_Q^* \text{ as } \mathcal{L}\left(T_n\right) = Q^n \text{ and } n \to \infty \tag{143}$$

where $L(h) = P_Q\left(h_n \neq Y | T_n\right)$ is random (as a function of T_n) and convergence is in probability. Consistency of a sequence h_n which holds for any Q is called *universal consistency.* There are simple classifiers which are universally consistent (histogram rule, nearest neighbor rule). Since $0 \leq L_Q(h) \leq 1$, the convergence in probability (143) is equivalent to

$$E_Q L(h_n) \to L_Q^* \text{ as } n \to \infty .$$

The consistency (143) may be written: for every $\alpha > 0$ there is a sequence $\varepsilon_n = \varepsilon(n, \alpha, Q) \to 0$ such that that for all n

$$P_Q \left(L(h_n) \le L_Q^* + \varepsilon_n \right) \ge 1 - \alpha.$$

(make a picture; recall that $L(h_n) \ge L_Q^*$). Generally $\varepsilon_n = \varepsilon(n, \alpha, Q)$ depends on Q which is unknown, and also L_Q^* is unknown. One may be interested in obtaining statements where ε_n is replaced by a known (or observable) quantity $\tilde{b}(n, \alpha)$, which does not depend on Q. But it will turn out that then the Bayes risk L_Q^* is too strong a benchmark; it will have to be replaced by the best risk over a restricted class of rules $\mathcal{C}$.

11.1 Vapnik-Chervonenkis Theory: Overview

Consider the following situation: Let $\mathcal{C}$ be a class of mappings of the form $h : \mathbb{R}^d \to \{0, 1\}$. Recall that a classifier was a map $h : \mathbb{R}^d \times \left\{ \mathbb{R}^d \times \{0, 1\} \right\}^{\times n} \to \{0, 1\}$, so $\mathcal{C}$ describes a class of h not depending on T_n. Assume however that T_n is present and that the error count or the empirical risk

$$L_{emp}(h) = \frac{1}{n} \sum_{j=1}^{n} \mathbf{1}_{\{h(X_j) \neq Y_j\}}$$

is used to estimate the error probability $L_Q(h) = P_Q \left(h(X) \neq Y \right)$ of each $h \in \mathcal{C}$. Denote by $\hat{h}$ the element of $\mathcal{C}$ that minimizes the estimated error probability over the class:

$$L_{emp}(\hat{h}) \le L_{emp}(h) \text{ for all } h \in \mathcal{C}.$$

This $\hat{h}$ now depends on T_n and is thus a classifier. With such an approach, we do not directly aim at the Bayes rule since we are selecting only among a restricted class $\mathcal{C}$ of rules. Of course, as $n \to \infty$, our class $\mathcal{C}$ may expand and in this way we may be able to "capture" the Bayes rule. But for now, suppose n fixed; then what we can hope for is to be as best as possible within the class of rules $\mathcal{C}$. Thus our target quantity is

$$\inf_{h \in \mathcal{C}} L_Q(h)$$

and since $L_Q(h)$ depends on the unknown Q, the best possible rule within $\mathcal{C}$ is also unknown. However for the (random) error probability

$$L_Q(\hat{h}) = P \left(\hat{h}(X) \neq Y | T_n \right)$$

of $\hat{h}$ we have:

11.1 Lemma

$$L_Q(\hat{h}) - \inf_{h \in \mathcal{C}} L_Q(h) \le 2 \sup_{h \in \mathcal{C}} \left| L_{emp}(h) - L_Q(h) \right|, \tag{144}$$

$$\left| L_{emp}(\hat{h}) - L_Q(\hat{h}) \right| \le \sup_{h \in \mathcal{C}} \left| L_{emp}(h) - L_Q(h) \right|. \tag{145}$$

Proof. In the proof we omit the subscript Q. We have

$$L(\hat{h}) - \inf_{h \in \mathcal{C}} L(h) \leq L(\hat{h}) - L_{emp}(\hat{h}) + L_{emp}(\hat{h}) - \inf_{h \in \mathcal{C}} L(h). \tag{146}$$

Let $\varepsilon > 0$ and let $h_\varepsilon \in \mathcal{C}$ be such that

$$\inf_{h \in \mathcal{C}} L(h) \geq L(h_\varepsilon) - \varepsilon$$

i.e. the classifier h_ε is within ε of attaining $\inf_{h \in \mathcal{C}} L(h)$ (h_ε always exists). Then

$$\begin{aligned} L_{emp}(\hat{h}) - \inf_{h \in \mathcal{C}} L(h) &\leq L_{emp}(\hat{h}) - L(h_\varepsilon) + \varepsilon \\ &\leq L_{emp}(h_\varepsilon) - L(h_\varepsilon) + \varepsilon \end{aligned}$$

(since $\hat{h}$ is the minimizer of R_{emp}), and hence from (146)

$$\begin{aligned} L(\hat{h}) - \inf_{h \in \mathcal{C}} L(h) &\leq L(\hat{h}) - L_{emp}(\hat{h}) + L_{emp}(h_\varepsilon) - L(h_\varepsilon) + \varepsilon \\ &\leq \left| L(\hat{h}) - L_{emp}(\hat{h}) \right| + \left| L_{emp}(h_\varepsilon) - L(h_\varepsilon) \right| + \varepsilon \\ &\leq 2 \sup_{h \in \mathcal{C}} \left| L_{emp}(h) - L(h) \right| + \varepsilon. \end{aligned}$$

Since $\varepsilon > 0$ was arbitrary, the first claim follows. The second inequality is trivially true. ■

The aim of Vapnik-Chervonenkis is to obtain probability inequalities of the type

$$P_Q\left(L_Q(\hat{h}) - \inf_{h \in \mathcal{C}} L_Q(h) > \varepsilon \right) \leq b_n(\varepsilon, \mathcal{C})$$

for some bounds $b(\varepsilon, \mathcal{C})$ independent of Q, but depending on the set of classifiers $\mathcal{C}$. By inequality (144) this reduces to

$$P_Q\left(\sup_{h \in \mathcal{C}} \left| L_{emp}(h) - L_Q(h) \right| > \varepsilon/2 \right) \leq b_n(\varepsilon, \mathcal{C}).$$

If $b_n(\varepsilon, \mathcal{C}) \to 0$ holds for fixed $\mathcal{C}$ then we obtain a type of consistency of the rule $\hat{h}$:

$$L_Q(\hat{h}) \to_P \inf_{h \in \mathcal{C}} L_Q(h) \text{ as } n \to \infty$$

analogous to (143). However the right hand side is a minimum within the class $\mathcal{C}$, thus $\hat{h}$ does not asymptotically attain the risk of the Bayes rule. However if $\mathcal{C}$ depends on n and $b_n(\varepsilon, \mathcal{C}_n) \to 0$ then we potentially obtain stronger statements on $\hat{h}$. Thus the specific form of the bound $b_n(\varepsilon, \mathcal{C})$ is of central interest.

Law of Large Numbers in function space

For any fixed $h \in \mathcal{C}$ we obtain trivially

$$L_{emp}(h) - L_Q(h) \to_P 0 \text{ as } n \to \infty.$$

Indeed

$$L_{emp}(h) = \frac{1}{n} \sum_{j=1}^{n} \mathbf{1}_{\{h(X_j) \neq Y_j\}}$$

is an average of n i.i.d. Bernoulli Bern(p) random variables with $p = E_Q \mathbf{1}_{\{h(X) \neq Y\}} = P_Q \left(h(X) \neq Y\right)$ $= L_Q(h)$, thus by the LLN

$$P_Q \left(\left|L_{emp}(h) - L_Q(h)\right| > \varepsilon\right) \to 0. \tag{147}$$

However the uniform version

$$P_Q \left(\sup_{h \in \mathcal{C}} \left|L_{emp}(h) - L_Q(h)\right| > \varepsilon\right) \to 0 \tag{148}$$

is not an automatic consequence; its validity depends on how large the class $\mathcal{C}$ is. The last relation may be interpreted as a uniform LLN over the set of classifiers $\mathcal{C}$. Thus a notion of *capacity* of the class $\mathcal{C}$ is likely to play a role in an upper bound $b_n(\varepsilon, \mathcal{C})$.
In fact we may write $L_Q(h)$ as the Q-probability of a set associated to h: if

$$A(h) = \{(x, y) : h(x) = 1, y = 0\} \cup \{(x, y) : h(x) = 0, y = 1\}$$

then

$$L_Q(h) = P_Q \left(h(X) \neq Y\right) = Q\left(A(h)\right).$$

If we define Q_n as the empirical measure associated to the training set T_n: for measurable $A \subset \mathbb{R}^d \times \{0, 1\}$

$$Q_n(A) = \frac{1}{n} \sum_{j=1}^{n} \mathbf{1}_A(X_j, Y_j)$$

then we have $L_{emp}(h) = Q_n(A(h))$ and defining a class of sets in $\mathbb{R}^d \times \{0, 1\}$

$$\mathcal{A} := \{A(h),\ h \in \mathcal{C}\} \tag{149}$$

(148) writes

$$P_Q \left(\sup_{A \in \mathcal{A}} \left|Q_n(A) - Q\left(A\right)\right| > \varepsilon\right) \to 0.$$

Thus the problem is reduced to the convergence of the emprical measure Q_n to Q, uniformly over the class of sets $\mathcal{A}$ in $\mathbb{R}^d \times \{0, 1\}$. Clearly, the size of the class $\mathcal{A}$ should be crucial for the validity of such a result.

Exponential inequalities

Recall Hoeffding's inequality:

11.2 Theorem *Let $Y_1, \ldots, Y_n$ be independent observations such that $EY_i = 0$ and $a \le Y_i \le b$. Let $\varepsilon > 0$. Then*

$$P\left(\left|n^{-1}\sum_{i=1}^{n} Y_i\right| \ge \varepsilon\right) \le 2\exp\left(-2n\varepsilon^2/(b-a)^2\right). \tag{150}$$

Let us apply this to estimating $L_{emp}(h) - L_Q(h)$ for one individual classification rule h. Set

$$V_i = \mathbf{1}_{\{h(X_i)\neq Y_i\}} - E_Q\mathbf{1}_{\{h(X)\neq Y\}} = \mathbf{1}_{\{h(X_i)\neq Y_i\}} - L_Q(h)$$

then V_i are i.i.d. with $-p = a \le V_i \le b = 1-p$ where $p = L_Q(h)$, and $n^{-1}\sum_{i=1}^{n} V_i = L_{emp}(h) - L_Q(h)$, hence $b - a = 1$ and (150) gives

$$P\left(|L_{emp}(h) - L_Q(h)| \ge \varepsilon\right) \le 2\exp\left(-2n\varepsilon^2\right).$$

We have thus "quantified" the simple Law of Large Numbers result (147). A uniform uniform version over $h \in \mathcal{C}$ (the VC Theorem for classifiers, Theorem 12.6 in [DGL] p. 199) takes the form

$$P\left(\sup_{h\in\mathcal{C}}|L_{emp}(h) - L_Q(h)| > \varepsilon\right) \le b_n(\varepsilon,\mathcal{C}) := 8s\left(\mathcal{C},n\right)\exp\left(-n\varepsilon^2/32\right) \tag{151}$$

where $s\left(\mathcal{C},n\right)$ is a capacity measure for the class $\mathcal{C}$ of classifiers, the so-called **shatter coefficient**. The shatter coefficient is originally a charactistic of a set of subsets of $\mathbb{R}^d$ and is then applied to classifiers via the correspondence (149).

11.3 Definition *Let $\mathcal{A}$ be a collection of measurable sets in $\mathbb{R}^d$. For $(z_1,\ldots,z_n) \in \left(\mathbb{R}^d\right)^n$, let $N_{\mathcal{A}}\left(z_1,\ldots,z_n\right)$ be the number of different sets in*

$$\left\{\{z_1,\ldots,z_n\}\cap A,\ A\in\mathcal{A}\right\}.$$

The n-th shatter coefficient of $\mathcal{A}$ is

$$s\left(\mathcal{A},n\right) := \max_{(z_1,\ldots,z_n)\in\left(\mathbb{R}^d\right)^n} N_{\mathcal{A}}\left(z_1,\ldots,z_n\right).$$

That is, the shatter coefficient is the maximal number of different subsets of n points that can be picked out by the class of sets $\mathcal{A}$. The shatter coefficient measures the richness or capacity of the class $\mathcal{A}$. We also state the VC theorem for general classes $\mathcal{A}$ ([DGL], Theorem 12.5, p. 197) of which (151) is a consequence.

11.4 Theorem *(Vapnik, Chervonenkis, 1971). Let Q be a probability measure on $\mathbb{R}^d$ and Q_n be the corresponding empirical measure based in an i.i.d. sample. For any collection $\mathcal{A}$ of measurable sets*

$$P_Q\left(\sup_{A\in\mathcal{A}}|Q_n(A) - Q\left(A\right)| > \varepsilon\right) \le b_n(\varepsilon,\mathcal{A}) := 8s\left(\mathcal{A},n\right)\exp\left(-n\varepsilon^2/32\right). \tag{152}$$

It turn out that, when the shatter coefficient $s(\mathcal{C}, n)$ of a class $\mathcal{C}$ of classifiers is computed via the correspondence (149) to sets, a simplification is possible. Namely, if we adopt the more obvious description of $h \in \mathcal{C}$ by the set $A_0(h) = \{x : h(x) = 1\}$ and $\mathcal{A}_0 = \{A_0(h), h \in \mathcal{C}\}$ then it can be shown that $s(\mathcal{C}, n) = s(\mathcal{A}_0, n)$ ([DGL] Theorem 13.1, p. 216). Therefore, in the examples below, when we discuss $s(\mathcal{A}, n)$ for some classes $\mathcal{A}$ of sets then this directly applies to the respective classifiers given by the indicator function of these sets.

Example 1. If $\mathcal{A}$ is the class of all subsets of $\mathbb{R}^d$ then from every $\{z_1, \ldots, z_n\}$, every subset can be picked out, and there are 2^n different subsets if all x_i are different. In this case $s(\mathcal{A}, n) = 2^n$ and inequality (152) loses its meaning:

$$b_n(\varepsilon, \mathcal{A}) = 8 \cdot (2^n) \exp\left(-n\varepsilon^2/32\right) = 8 \exp\left(n\left(\log 2 - \varepsilon^2/32\right)\right)$$

and for $\varepsilon < \sqrt{32 \log 2} = 4.71$ the upper bound $b_n(\varepsilon, \mathcal{C})$ does not converge to zero, i.e. (152) does not imply the uniform LLN over $\mathcal{C}$. $\square$

Example 2. Consider the case $d = 1$ and all half spaces in $\mathbb{R}$: $A = [a, \infty)$ or $A = (-\infty, a]$ for some $a \in \mathbb{R}$. Consider ordered points $x_1 < \ldots < x_n$; then sets $A = (-\infty, a]$ pick out all subsets $\{x_1, \ldots, x_k\}$, $k = 0, \ldots, n$ ($k = 0$ meaning the empty set); there are $n + 1$ of these. Also sets $A = [a, \infty)$ pick out all subsets $\{x_k, \ldots, x_n\}$, $k = 1, \ldots, n + 1$ ($k = n + 1$ meaning the empty set). Here we counted the empty set and the full set $\{x_1, \ldots, x_n\}$ twice, so $s(\mathcal{A}, n) = 2(n + 1) - 2 = 2n$. The upper bound $b_n(\varepsilon, \mathcal{A})$ becomes

$$b_n(\varepsilon, \mathcal{A}) = 8n \exp\left(-n\varepsilon^2/32\right) = 8 \exp\left(\log n - n\varepsilon^2/32\right)$$

This is still an exponential bound; since $\log n/n \to 0$, e.g. for sufficiently large n and a small $\delta > 0$ an upper bound is $8 \exp\left(-n\left(\varepsilon^2/32 - \delta\right)\right)$.$\square$

In these two examples the growth of the shatter coefficient $s(\mathcal{A}, n)$ was either exponential ($\exp(n \log 2)$ for the class of all sets) or linear ($2n$ for the half lines). The concept of **VC dimension** of $\mathcal{A}$, to be introduced, serves to describe classes $\mathcal{A}$ for which $s(\mathcal{A}, n)$ grows algebraically (that is like n^s), and for which (152) hence is a meaningful uniform LLN. Analogously, the VC dimension of $\mathcal{C}$ will provide a bound on learning capacity by (151) and (149).
If $s(\mathcal{A}, n) = 2^n$ then the class of sets $\mathcal{A}$ is said to *shatter* a set of n points (more precisely, *there is* a set of n distinct points which can be shattered). It is easy to see that if $s(\mathcal{A}, k) < 2^k$. that is . $\mathcal{A}$ does not shatter k points (there is no set of k points such that $\mathcal{A}$ picks out all its subsets) then $\mathcal{A}$ does not shatter n points for all $n > k$. The first time when this happens is important:

11.5 Definition *Let $\mathcal{A}$ be a collection of sets with $|\mathcal{A}| > 2$. The largest integer $k \geq 1$ for which $s(\mathcal{A}, k) = 2^k$ is denoted by $V_{\mathcal{A}}$ and is called the Vapnik-Chervonenkis dimension (or VC dimension) of the class $\mathcal{A}$. If $s(\mathcal{A}, n) = 2^n$ for all n then by definition $V_{\mathcal{A}} = \infty$.*

Again, we carry this over to classes $\mathcal{C}$ by (149). Thus, the VC dimension of $\mathcal{A}$ is the maximal number of points which can be arranged so that $\mathcal{A}$ can shatter them.
It turns out that the VC dimension plays a crucial role in the growth behaviour of $s(\mathcal{A}, n)$ as $n \to \infty$. Namely it can be shown [DGL, Theorem 13.3] that if $V_{\mathcal{A}} < \infty$ then

$$s(\mathcal{A}, n) \leq (n+1)^{V_{\mathcal{A}}}. \tag{153}$$

Example 2 (continued). Consider again the class $\mathcal{A}$ of half real lines $[a, \infty)$ and $(-\infty, a]$. It shatters any two points $x_1 < x_2$ and not does shatter any triple of points $x_1 < x_2 < x_3$ (the subset $\{x_2\}$ cannot be picked out). Thus $V_{\mathcal{C}} = 2$.
Remark. Then (153) yields a bound $s(\mathcal{A}, n) \leq (n+1)^2$, but above we already showed $s(\mathcal{A}, n) = 2n$. This shows that (153) is not sharp. □

Example 3. Consider the class $\mathcal{A}$ of closed intervals on the real line, i. e. $[a_1, a_2]$. It shatters any two points $x_1 < x_2$ and not does shatter any triple of points $x_1 < x_2 < x_3$ (the subset $\{x_1, x_3\}$ cannot be picked out). Thus $V_{\mathcal{C}} = 2$. On the other hand we can directly estimate $s(\mathcal{A}, n)$: suppose $x_1 < \ldots < x_n$; the nonempty subsets which can be picked out are $\{x_k, \ldots, x_{k+s}\}$ where $1 \leq k \leq n$ and $0 \leq s \leq n-k$. There are

$$\sum_{k=1}^{n}(n-k+1) = \sum_{k=1}^{n} k = n(n+1)/2$$

possibilities, thus, adding the empty set, we obtain $s(\mathcal{A}, n) = n(n+1)/2 + 1$ which is of order n^2, the same order as the upper bound in (153). □

Example 4. Let $\mathcal{A}$ be all linear half-spaces on the plane. Any 3-point set (not all on a line) can be shattered. No 4 point set can be shattered. Consider, for example, 4 points forming a diamond. Let T be the left and rightmost points. This can't be picked out; other configurations can also be seen to be unshatterable. So $V_{\mathcal{A}} = 3$. In general, halfspaces in $\mathbb{R}^d$ have VC dimension $d+1$ [DGL, Corollary 13.1, p. 223]. In example 2 we showed this for $d = 1$. □

Example 5.. Let $\mathcal{A}$ be all rectangles on the plane with sides parallel to the axes. There is a 4 point set that can be shattered; e.g. a diamond. Let S be a 5 point set. There is one point that is not leftmost, rightmost, uppermost, or lowermost. Let T be all points in S except this point.Then T can't be picked out. So $V_{\mathcal{A}} = 4$. In general, rectangles in $\mathbb{R}^d$ have VC dimension $2d$ [DGL, Theorem 13.8.] □

11.2 Confidence intervals

The combination of inequalities (153) and (151) yields

$$P_Q\left(\sup_{h\in\mathcal{C}}|L_{emp}(h)-L_Q(h)|>\varepsilon\right)\le b_n(\varepsilon,\mathcal{C}):=8\,(n+1)^{V_\mathcal{C}}\exp\left(-n\varepsilon^2/32\right)$$
$$=8\exp\left(V_\mathcal{C}\log(n+1)-n\varepsilon^2/32\right)$$

which is a valid exponential bound (since $\log(n+1)/n\to 0$). We may set the right side $b_n(\varepsilon,\mathcal{C})=\alpha$ and solve for ε; this yields $\varepsilon=\tilde{b}_n(\alpha,\mathcal{C})$ for

$$\tilde{b}_n(\alpha,\mathcal{C})=\sqrt{\frac{32}{n}\left(V_\mathcal{C}\log(n+1)+\log\frac{8}{\alpha}\right)}.$$

and an inequality

$$P_Q\left(\sup_{h\in\mathcal{C}}|L_Q(h)-L_{emp}(h)|\le\tilde{b}_n(\alpha,\mathcal{C})\right)\ge 1-\alpha\text{ for all }Q=\mathcal{L}\left(X,Y\right). \tag{154}$$

From this, two types of confidence statements can be derived, for the empirical risk minimizing classifier $\hat{h}$ treated in Lemma 11.1.

Oracle type inequalities

By inequality (144) we obtain

$$L_Q(\hat{h})-\inf_{h\in\mathcal{C}}L_Q(h)\le\tilde{b}_n(\alpha,\mathcal{C})$$

valid with probability at least $1-\alpha$, thus a statement

$$P_Q\left(L(\hat{h}_n)\le\inf_{h\in\mathcal{C}}L_Q(h)+\tilde{b}_n(\alpha,\mathcal{C})\right)\ge 1-\alpha\text{ for all }Q=\mathcal{L}\left(X,Y\right). \tag{155}$$

The name "oracle type" refers to the fact that with the classifier $\hat{h}_n$ we are almost as good (up to a term $\tilde{b}_n(\alpha,\mathcal{C})$) as if an oracle would have told us which $h\in\mathcal{C}$ is the best for the given unknown Q.

PAC bounds

In the inequality (155) have an upper bound for unknown $L(\hat{h}_n)$ which is itself unknown, indeed $\inf_{h\in\mathcal{C}}L_Q(h)$ depends on Q. We may want to replace it by a known or observable quantity, in order to asses the performance of $\hat{h}_n$ for a given class $\mathcal{C}$ and compare it with another class $\mathcal{C}'$. Inequality (145) of Lemma 11.1 can be used for that: in conjunction with (154) it yields

$$P_Q\left(L(\hat{h}_n) \leq L_{emp}(\hat{h}) + \tilde{b}_n(\alpha, \mathcal{C})\right) \geq 1-\alpha \text{ for all } Q = \mathcal{L}\left(X, Y\right). \tag{156}$$

Indeed here the upper bound $L_{emp}(\hat{h}) + \tilde{b}_n(\alpha, \mathcal{C})$ is known, and serves as an upper confidence bound of level $1-\alpha$ for $L(\hat{h}_n)$. This is sometimes referred to as a PAC statement ("**p**robably **a**pproximately **c**orrect"), meaning that with high probability $1-\alpha$ the empirical risk minimizer within the class $\mathcal{C}$ has $L(\hat{h}_n)$ bounded above by a known quantity, regardless of Q which is unknown. Some authors call this the "fundamental theorem of learning" (cf. [CST], Theorem 4.1, p. 56, for the case $L_{emp}(\hat{h}) = 0$). It should be noted that (156) is valid for any classifierv $\hat{h}$, not just the empirical risk minimizer, since (145) of Lemma 11.1 is holds for any $\hat{h}$.
The inequality (156) also gives rise to the idea of *structural risk minimization*, a model choice method, described as follows.

Structural risk minimization

Suppose there is a nested structure on sets of classifiers: $\mathcal{C}_r$, $r = 1, 2, \ldots$ such that $\mathcal{C}_{r_1} \subset \mathcal{C}_{r_2}$ if $r_1 < r_2$. One is then confronted with the problem of deciding what $\mathcal{C}$ to choose a priori. Problems of this kind are generally called *model selection problems*. Suppose that for each of these classes , there is an inequality

$$L(h) \leq L_{emp}(h) + \tilde{b}_n(\alpha, \mathcal{C}_r) \text{ for all } h \in \mathcal{C}_r \tag{157}$$

valid with probability at least $1-\alpha$, and a capacity bound $\tilde{b}_n(\alpha, \mathcal{C}_r)$ for the particular class $\mathcal{C}_r$. If $V_{\mathcal{C}_r}$ is the VC dimension of $\mathcal{C}_r$ then as shown above

$$\tilde{b}_n(\alpha, \mathcal{C}_r) = \sqrt{\frac{32}{n}\left(V_{\mathcal{C}_r}\log(n+1) + \log\frac{8}{\alpha}\right)}.$$

Then to choose an appropriate r, consider the best classifier $\hat{h}_r$ from each class $\mathcal{C}_r$, i.e. the one which minimizes the empirical risk:

$$L_{emp}(\hat{h}_r) = \inf_{h \in \mathcal{C}_r} L_{emp}(h).$$

Then according to (157) we have

$$L(\hat{h}_r) \leq L_{emp}(\hat{h}_r) + \tilde{b}_n(\alpha, \mathcal{C}_r) \tag{158}$$

The structural risk minimization (SRM) idea is as follows: select the r which minimizes the upper bound, i.e. the right side in (158). For larger classes $\mathcal{C}_r$, the empirical risk $L_{emp}(\hat{h}_r)$ will be smaller (fit to the data will be better) but the VC dimension will be higher, so $\tilde{b}_n(\alpha, \mathcal{C}_r)$ will be larger. Minimizing the sum of both promises to find the best $L(\hat{h}_r)$. That is a heuristical principle, akin to other procedures of model choice. Various justifications by rigourous results have been put forward in the literature, cf. chapter 18 in [DGL].

11.3 Bibliographic remarks

1. The tutorial [Burg] present a rigourous version of the structural risk minimization principle, not for SVM but for related classifiers, the so called gap tolerant classifiers.

2. In [Herb], section 4.2.3., p. 131 another mathematically rigorous theory is developed around the structural risk minimization (SRM) principle, using the concept of empirical VC dimension, p. 139, though the SRM is also not justified in its original form.

3. Other concepts of capacity of a set of classifiers, which are more suitable than the VC dimension for explaining the success of the kernelized SVM, are discussed in chap 12 of [ScS]. In particular, an exponential inequality for the kernelized SVM with radial basis (Gaussian) kernel is proved in Corollary 12.6 (p. 364) there.

4. For oracle type inequalities for SVM, and connections to the SRM principle, see Blanchard et al [1] and references there. Tarigan and van de Geer [5] give a result of this type for SVM with an ℓ_1 type penalty (as opposed to a smoothness penalty). For oracle inequalities in a regression (not classification) context, cf Sauvé [4].

5. Tsybakov [6] defines a concept of aggregation of classifiers and discusses efficiency, in terms of the behaviour of the regression function $r(x)$ near the decision boundary $r(x) = 1/2$.

A Appendix: Exercises

A.1 Homework Set 1

Problem 1.1 (10 points) Assume the points $x_i \in [0,1]$ ordered and all distinct: $0 < x_1 < \ldots <$

$x_n < 1$. Consider the matrix

$$\bar{K} = (K(x_i, x_j))_{i,j=1}^n$$

where $K(u,t) = \min(u,t)$. In the notes it is argued that the symmetric matrix $\bar{K}$ is nonnegative definite (meaning $c^\top \bar{K} c \geq 0$ for every $c \in \mathbb{R}^n$). Prove the following result:
Lemma. *The matrix $\bar{K}$ is nonsingular.*

Problem 1.2. (10 points) In equation (59), p. 53 it is argued that the solution of the minimization problem is

$$\hat{f} = \hat{D}y \text{ for } \hat{D} = \left(I_n + \lambda n \bar{K}^{-1}\right)^{-1}.$$

where $\hat{D}$ is the so-called "hat matrix". Show that $\hat{D}$ exists for any $\lambda > 0$, i.e. the matrix $I_n + \lambda n \bar{K}^{-1}$ is nonsingular. (Part of the reasoning is given in the paragraph below (59), p. 53. Nevertheless, write a proof as detailed as possible, including or not including some of the arguments in the paragraph below (59)).

A.2 Homework Set 2

Problem 2.1 (20 points) Assume the points $x_i \in [0,1]$ ordered and all distinct. Consider the matrix

$$\bar{K} = (K(x_i, x_j))_{i,j=1}^n \tag{159}$$

where $K(u,t) = \min(u,t)$. Use RKHS /spline theory to find the inverse of $\bar{K}$ for the case of equispaced design points: $x_i = i/n$, $i = 1, \ldots, n$.

Problem 2.2.
a) (20 points) Consider the general linear smoothing spline as treated in Corollary 6.5, p. 56, where the design points $x_i \in [0,1]$ are ordered and all distinct: $0 < x_1 < \ldots < x_n \leq 1$. Let $\hat{D}$ be the hat matrix for this problem, i.e. the matrix wich to any data vector $y = (y_1, \ldots, y_n)^\top$ assigns the vector of values in x_i of the resulting linear smoothing spline solution $\check{f} \in H^1(0,1)$

$$\hat{f} = \left(\check{f}(x_1), \ldots, \check{f}(x_n)\right)^\top$$

according to $\hat{f} = \hat{D}y$. Here $\check{f}$ is the minimizer over functions $f \in H^1(0,1)$ of

$$n^{-1}\sum_{i=1}^n (y_i - f(x_i))^2 + \lambda \int_0^1 \left(f'(t)\right)^2 dt. \tag{160}$$

Show that

$$\hat{D} = P + Q\left(Q + \lambda n \bar{K}^{-1}\right)^{-1} Q \tag{161}$$

where P is the projection operator (projection matrix) onto the linear subspace span($\mathbf{1}$) of $\mathbb{R}^n$ spanned by the vector $\mathbf{1} = (1, \ldots, 1)^\top$ and Q is the projection matrix onto the orthogonal complement of span($\mathbf{1}$). The matrix $\bar{K}$ is given by (159).

Hint. *Recall projection operators in* $\mathbb{R}^n$. Let V be a $n \times k$ matrix, $k \leq n$ with linearly independent columns, i.e. rank$(V) = k$. Let span(V) be the linear subspace of $\mathbb{R}^n$ spanned by the columns of V. Then $P := V\left(V^\top V\right)^{-1} V^\top$ is the projection operator onto span(V) and $Q := I_n - P$ is the projection operator onto the orthogonal complement of span(V). We have $PP = P$, $P = P^\top$, $PQ = 0$ and there is an orthogonal $n \times k$ matrix U (with $U^\top U = I_k$) such that $P = UU^\top$. For any $x \in \mathbb{R}^n$, the vector $Px \in$ span(V) is the minimizer of $\|x - z\|$ over $z \in$ span(V).

b) (optional) Show that $\hat{D}$ also has the shrinkage property as described on p. 53, that is all eigenvalues of $\hat{D}$ are between 0 and 1.

A.3 Homework Set 3

Problem 3.1. (15 points) Consider the linear logistic regression model for classification, discussed on p. 70. For the predictive distribution we obtained the expression (see (72) on p. 70)

$$P\left(Y_* = 1 | y, x_*\right) = \int l\left(w^\top x_*\right) p\left(w | y\right) dw.$$

using a normal prior $w \sim N_d\left(0, \Sigma\right)$ for the parameter $w \in \mathbb{R}^d$. Here one has a training set (x_i, y_i), $i = 1, \ldots, n$ where the y_i are assumed to be realizations of a random variables Y_i according to $P\left(Y = 1 | X = x\right) = l\left(w^\top x\right)$, with l the logistic function, and $x_1, \ldots, x_n, x_*$ (all in $\mathbb{R}^d$) are considered nonrandom.
It is evident that in this model, the resulting posterior density $p\left(w|y\right)$ is nongaussian (cp. also (69) p. 68). However in common approximation methods, a Gaussian approximation $\tilde{p}\left(w|y\right)$ for $p\left(w|y\right)$ is postulated. Define the approximate predictive distribution

$$\tilde{P}\left(Y_* = 1 | y, x_*\right) = \int l\left(w^\top x_*\right) \tilde{p}\left(w | y\right) dw.$$

Show that if $\tilde{p}$ is any multivariate normal density on $\mathbb{R}^d$ then the decision boundary

$$\left\{x_* : \tilde{P}\left(Y_* = 1 | y, x_*\right) = 1/2\right\}$$

is a hyperplane in $\mathbb{R}^d$ passing through the origin.
Hint: use symmetries of the logistic function and normal densities.

Problem 3.2. (optional) Suppose that for Bayesian inference in the above model, one adds the assumption that x_i, $i = 1, \ldots, n$ are also realizations of random variables with a specific distribution. Thus, each (x_i, y_i), $i = 1, \ldots, n$ is assumed to be an independent realization of a random variable (X, Y) where $P(Y = 1|X = x) = l\left(w^\top x\right)$ and X has a density

$$q(x) = \frac{1}{2}\varphi(x - w/2) + \frac{1}{2}\varphi(x + w/2),\ x \in \mathbb{R}^d$$

where φ is the standard normal density on $\mathbb{R}^d$ (the density of $N_d(0, I_d)$). Again, assume a normal prior $w \sim N_d(0, \Sigma)$ for the parameter $w \in \mathbb{R}^d$ and let $p(w|x, y)$ be the posterior density of w given $(x, y) = ((x_i, y_i), i = 1, \ldots, n)$. Show that $p(w|x, y)$ is normal.
(Note that the randomness assumption has not been extended to the value x_* from above.)

B References

References

[Bish] Bishop, C. M. *Pattern Recognition and Machine Learning* (Information Science and Statistics). Springer 2006

[BTA] Berlinet, A., Thomas-Agnan, C., *Reproducing kernel Hilbert spaces in Probability and Statistics,* Kluwer, 2004

[BFOS] Breiman, L., Friedman, J., Olshen, R., Stone, C. *Classification and Regression Trees.* Wadsworth, Belmont, CA., 1984.

[Burg] Burges, C. J. C., A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery, 2, 121-167 (1998)

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

[D] Durret, R., *Probability: Theory and Examples,* 2nd Ed., Duxbury Press, 1996.

[DGL] Devroye, L, Gyorfi, L., Lugosi, G., *A Probabilistic Theory of Pattern Recognition*, Springer 1997

[Fine] Fine, T. L., *Feedforward Neural Network Methodology* (Springer Series in Statistics). Springer 1999

[HTF] Hastie, T, Tibshirani, R. J. H. Friedman, R.H., *The Elements of Statistical Learning (Data Mining, Inference and Prediction),* Second Edition, Springer, 2009.

[Herb] Herbrich, R. *Learning Kernel Classifiers: Theory and Algorithms* (Adaptive Computation and Machine Learning). MIT Press 2002

[Kl] Klemelä, J., *Smoothing of Multivariate Data.*Wiley, New York, 2009

[RW] Rasmussen, C. E. and Williams, C. K. I., *Gaussian Processes for Machine Learning* (Adaptive Computation and Machine Learning). MIT Press 2006

[ScS] Schölkopf, B. and Smola, A. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (Adaptive Computation and Machine Learning). MIT Press, 2002

[Vap1] Vapnik, V. , *Statistical Learning Theory*, Wiley, 1998..

[Vap2] Vapnik, V. *The Nature of Statistical Learning Theory*, 2nd ed, Springer 1999

[Wah] Wahba, Grace, *Spline models for observational data*. SIAM [Society for Industrial and Applied Mathematics] (Philadelphia) 1990

[W] Wasserman, L., (2004). *All of Statistics. A Concise Course in Statistical Inference*. Springer Verlag

[1] Blanchard, G., Bousquet, O., Massart, P., (2008). Statistical performance of support vector machines, Ann. Statist. **36** 489-531.

[2] Boucheron, S, Bousquet, O. and Lugosi, G. (2005). Theory of classification: A survey of some recent advances. ESAIM P&S: Probability and Statistics (online), 9, 323-375

[3] Olshen, R. (2007). Tree-structured regression and the differentiation of integrals. Ann. Statist. **35** 1-12

[4] Sauvé, M. (2009). Histogram selection in non Gaussian regression. ESAIM: P&S: Probability and Statistics (online), **13** 70–86

[5] Tarigan, B. and van de Geer, S. A. (2006). Classifiers of support vector machine type with ℓ_1 complexity regularization. Bernoulli **12** 1045–1076

[6] Tsybakov, A. B. (2004). Optimal aggregation of classifiers in statistical learning, Ann. Statist. **32** 135-166.