Introduction to statistical learning (lecture notes)

Jüri Lember

Aalto, Spring 2012

http://math.tkk.fi/en/research/stochastics

Literature:

- "The elements of statistical learning"
  T. Hastie, R. Tibshirani, T. Friedman.
  Springer, 2001.

- "Statistical learning theory"
  V. N. Vapnik.
  Wiley, 1998.

- "A probabilistic theory of pattern recognition"
  L. Devroye, L. Györfi, G. Lugosi.
  Springer, 1996.

- "An introduction to support vector machines and other kernel-based learning methods"
  N. Cristianini, J. Shawe-Taylor.
  Cambridge University Press, 2003.

- "Kernel methods for pattern analysis"
  J. Shawe-Taylor, N. Cristianini.
  Cambridge University Press, 2004.

- "Learning with kernels : support vector machines, regularization, optimization, and beyond"
  B. Schölkopf ; A. J. Smola.
  MIT Press, 2002.

- "Pattern recognition and machine learning"
  C. Bishop
  Springer 2006.

- "Discriminant analysis and statistical pattern recognition"
  G. McLachlan
  Wiley, 1992.

- "Introduction to machine learning"
  E. Alpaydin
  MIT, 2004.

- "Statistical pattern recognition" (2nd edition)
  A. Webb
  Wiley, 2002

- "Pattern classification" (2nd edition)
  R. Duda, P. Hart, D. Stork
  Wiley 2000

- "Pattern recognition"
  S. Theodoridis
  Academic Press, 2003
- $\cdots$

Homepages (lecture notes, readings, tutorials, cites):

- Gabor Lugosi (Pompeu Fabra):
  http://www.econ.upf.es/ lugosi/surveys.html
- Peter Bartlett (Berkley):
  http://www.cs.berkeley.edu/ bartlett
- Rober Schapire (Princenton):
  http://www.cs.princeton.edu/ schapire
- M. Jordan (Berkley), T. Jaakola, (MIT), R. Nowak (Wisconsin) etc.

Contents

Chapter 1

Introduction to pattern recognition

1.1 The problem

Pattern recognition or classification or discrimination is the assignment of a label or class to a given input.

Examples of classification are (among many others):

- In banking, calculating the credit risk given the information about the customer. The classes are: "high risk" and "low risk".
- Determining whether a given e-mail is spam or not. The classes are: "spam" and "not spam".
- Crediting students in a test. The classes are: "A", "B", "C", "D", "E", "F".
- Character recognition (handwritten or, for example, from licence plate). The classes are "A", "B", "C",..., "Y".
- The digit recognition. The classes are "0", "1",..., "9".
- Medical diagnosis. The input is medical information and the classes are the illnesses.
- Face recognition. The input is an image and the classes people to be recognized.

In all these examples the characteristic features are: the set of classes (labels) is given, finite and "relatively small" (often there are only two classes). Thus, we can identify every class with a number from a set $\mathcal{Y} = \{0, \ldots, k-1\}$ (altogether k classes). How the classes are coded, does obviously not matter. Often, the two classes are coded as $\{-1, 1\}$.
The classifier (machine, human being, computer program) assigns a class to every object given the information about the classified object (input). Without loss of generality, we can assume that the information or input is given as a d-dimensional **feature** or pattern or observation vector $x \in \mathbb{R}^d$. Indeed, every information can be presented as such a vector where the dimension d possibly very large. An important part of the classification

or recognition procedure is **knowledge or feature extraction** that consists of finding these features or variables that explain the data and are most useful in classification. These features are then presented in the feature vector x.

Formally, a **classifier** is a function

$$g : \mathbb{R}^d \mapsto \mathcal{Y}. \tag{1.1.1}$$

Note that (1.1.1) can be written as

$$g = \sum_{i=0}^{k-1} i I_{C_i}(x), \tag{1.1.2}$$

where I_C is the indicator of the set C, i.e.

$$I_C(x) := \begin{cases} 1, & \text{if } x \in C; \\ 0, & \text{if } x \notin C. \end{cases}$$

Hence g defines a partition $\{C_0, \ldots, C_{k-1}\}$ of the set $\mathbb{R}^d$, where class i will be assigned to an object if and only if the feature vector belongs to the set C_i.

1.2 Bayesian decision theory

How to measure the goodness of g? Since the input vectors x are not known for sure, they are considered as random. Every random vector has a distribution or law that is uniquely characterized by its distribution function. In the following, let F be the distribution function of feature vector. Recall that for every function $h : \mathbb{R}^d \to \mathbb{R}$, the expected value of $h(x)$ over all features is the Lebesgue integral

$$\int h(x) dF(x).$$

Example. Suppose we aim to determine the sex of a human being based on his or her length and weight, both given as the elements of $\mathbb{N}$. Hence, $d = 2$ and $x = (x_1, x_2)$, where $x_1, x_2 \in \mathbb{N}$. Let $N(x_1, x_2)$ be the number of individuals in the population with length and weight (x_1, x_2). Assuming that every individual is chosen with equal probability, we obtain the probability that an individual with feature vector $x = (x_1, x_2)$ is chosen is

$$p(x) = p(x_1, x_2) = \frac{N(x_1, x_2)}{N},$$

where N is the size of population. The distribution of the feature vector is given by the vectors and their probabilities:

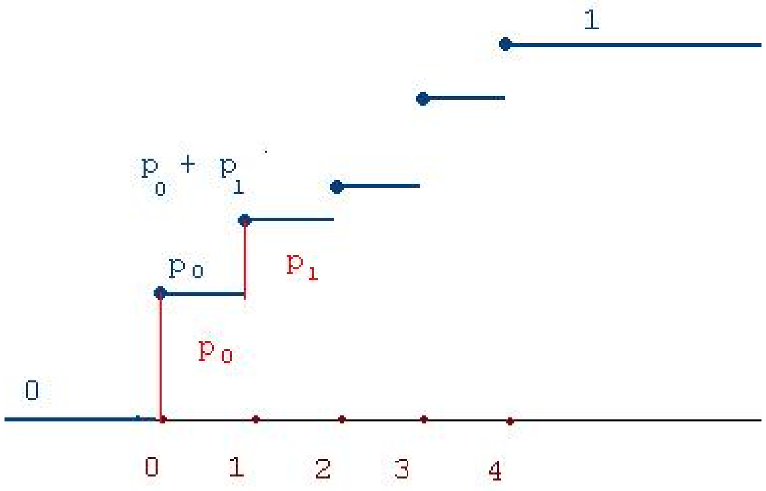
$$\{(x_1, x_2), p(x_1, x_2) : x_1, x_2 \in \mathbb{N}\}.$$

The distribution function $F : \mathbb{R}^2 \mapsto [0,1]$ is then

$$F(x) = F(x_1, x_2) = \sum_{x_1' \leq x_1, x_2' \leq x_2} p(x_1', x_2').$$

Although every object belongs to one class, in practice a feature vector x can belong to the objects from different classes. Hence, the class of the object with features x can be considered random as well. Let $p(j|x)$ be the probability that an object with features x belongs to class $j = 0, \dots k-1$. Let $F(y|x)$ be the corresponding distribution function:

$$F(y|x) := \sum_{i \leq y} p(i|x).$$



Recall the (conditional) expectation of a function h over the measure $F(y|x)$ is the sum

$$\int h(y) dF(y|x) = \sum_i h(i) p(i|x). \tag{1.2.1}$$

Example. Let us continue with the example. Suppose that $\frac{2}{3}$ of the people with length 175 and weight 67 are men. Then $p(0|(175,65)) = \frac{2}{3}$ and $p(1|(175,65)) = \frac{1}{3}$, where 1 is female and 0 male.

The feature x is a random vector with distribution function F. Given the feature vector x, also the class of the object is random with conditional distribution function $F(y|x)$. Thus the pair – feature vector and the corresponding class – is $d+1$-dimensional random vector, let $F(x,y)$ be its distribution function. Given a function

$$h : \mathbb{R}^d \times \mathcal{Y} \mapsto \mathbb{R},$$

the expected value of $h(x,y)$ over the vectors (x,y) is the following integral

$$\int_{\mathbb{R}^d\times\mathcal{Y}} h(x,y)dF(x,y) = \int_{\mathbb{R}^d}\int_{\mathcal{Y}} h(x,y)dF(y|x)dF(x) = \int_{\mathbb{R}^d}\sum_{i=0}^{k-1} h(x,i)p(i|x)dF(x). \tag{1.2.2}$$

Example. Let us continue with the example. The probability $p(0|(x_1,x_2))$ is the proportion of men amongst the people with length x_1 and weight x_2; $p(x_1,x_2)$ is the proportion of such people in population and $p(0|(x_1,x_2))p(x_1,x_2) =: p(x_1,x_2,0)$ is the proportion of men with features x_1 and x_2 in the population. In other words, $p(x_1,x_2,0)$ is the probability that the three-dimensional random vector – length, weight and sex – takes the value $(x_1,x_2,0)$. The distribution function of that three-dimensional random vector is

$$F:\mathbb{R}^3\mapsto[0,1],\quad F(y,x_1,x_2)=\sum_{i\le y}\sum_{x_1'\le x_1}\sum_{x_2'\le x_2} p(x_1',x_2',i).$$

Let π_0 and π_1 be the probability that randomly chosen individual is man and woman, respectively. Clearly

$$\pi_0=\sum_{x_1,x_2} p(x_1,x_2,0),\quad \pi_1=\sum_{x_1,x_2} p(x_1,x_2,1).$$

Exercise: Suppose that one-dimensional random variable (feature) takes five possible values: 1,2,3,4,5 with equal probability. Let $k=2$, and $p(1|x=i)=\frac{i}{5}$, $i=1,\ldots,5$. Find the joint distribution of class and feature. Find $F(y,x)$, π_0 and π_1.

Note that (1.1.1) is non-random. i.e. g classifies every feature x uniquely. Since, in general, that is not so, we have that misclassification is inevitable!

1.2.1 Loss and risk

In reality, the misclassification causes loss that often quantify.

Definition 1.2.1 **The loss function**

$$L:\mathcal{Y}\times\mathcal{Y}\to\mathbb{R}^+$$

assigns to every pair (i,j) loss that occur when the object belonging to class i is classified as belonging to class j.

It is natural to take $L(i,i)=0$ – classifying correctly causes no loss.

The loss-function depends on the concrete task and it should be taken into account in measuring the goodness of g. Clearly good classifier g is such that the loss $L(y,g(x))$ is small. However, since (x,y) is random, so is the loss $L(y,g(x))$. Therefore, it is natural measure the goodness of g using the expected loss, where expectation is taken over pairs (x,y). The expected loss of g measures the average loss when g is applied repeatedly.

Definition 1.2.2 **The risk** *of classifier g is the average loss over the joint distribution $F(x,y)$:*

$$R(g) := \int L(y, g(x))dF(x,y). \tag{1.2.3}$$

From (1.2.2), we get

$$R(g) = \int_{\mathbb{R}^d} \int_{\mathcal{Y}} L(y, g(x))dF(y|x)dF(x) = \int_{\mathbb{R}^d} \sum_{j=0}^{k-1} L(j, g(x))p(j|x)dF(x). \tag{1.2.4}$$

1.2.2 Bayes classifier and Bayes risk

Let us find the best possible classifier, i.e. classifier that minimizes (1.2.3) over all possible classifiers. Let x be a feature vector and g a classifier. The **conditional risk** of g given x is the average loss when classifying object with features x:

$$R(g(x)|x) := \int_{\mathcal{Y}} L(y, g(x))dF(y|x) = \sum_{j=0}^{k-1} L(j, g(x))p(j|x).$$

In other words, the conditional risk $R(i|x)$ is average risk when classifying an object with features x to the class i (i.e. when $g(x) = i$). Averaging the conditional risk over all features, we obtain the risk $R(g)$ (recall (1.2.4))

$$\int_{\mathbb{R}^d} R(g(x)|x)dF(x) = \int_{\mathbb{R}^d} \int_{\mathcal{Y}} L(y, g(x))dF(y|x)dF(x) = R(g).$$

From the equality above, it follows that to minimize $R(g)$ it suffices to minimize $R(g(x)|x)$ for every x. Indeed, let the classifier $g^*(x)$ be defined as follows:

$$g^*(x) = \arg\min_{g(x)} R(g(x)|x) = \arg\min_{i \in \mathcal{Y}} R(i|x) = \arg\min_{i \in \mathcal{Y}} \sum_{j=0}^{k-1} L(j, i)p(j|x). \tag{1.2.5}$$

From the definition above, it follows that

$$R(g^*(x)|x) = \min_{i \in \mathcal{Y}} R(i|x)$$

so that for any other classifier g it holds $R(g^*(x)|x) \leq R(g(x)|x)$. By the monotonicity of integration

$$R(g^*) = \int R(g^*(x)|x)dF(x) \leq \int R(g(x)|x)dF(x) = R(g). \tag{1.2.6}$$

Hence, $R(g^*)$ has the smallest possible risk over all possible classifiers:

$$R(g^*) = \inf_g R(g).$$

Definition 1.2.3 *The classifier g^*, where $g^*(x)$ is defined as in (1.2.5) is* **Bayes classifier** *and its risk*

$$R^* := R(g^*) = \int L(y, g^*(x))dF(y,x)$$

is called **Bayes risk**.

From the equation $R(g^*(x)|x) = \min_{i \in \mathcal{Y}} R(i|x)$, we get

$$R^* = \int \min_{i \in \mathcal{Y}} R(i|x)dF(x) = \int \min_{i \in \mathcal{Y}} \sum_{j=0}^{k-1} L(j,i)p(j|x)dF(x).$$

To recapitulate: Bayes classifier is the best possible classifier and Bayes risk is the smallest possible risk.

When $L(i,i) = 0$ and $k = 2$, then for any x, the conditional risk is

$$R(g(x)|x) = \begin{cases} L(1,0)p(1|x) & \text{when } g(x) = 0, \\ L(0,1)p(0|x) & \text{when } g(x) = 1. \end{cases}$$

Bayes classifier is then

$$g^*(x) = \begin{cases} 0 & \text{when } L(1,0)p(1|x) \leq L(0,1)p(0|x), \\ 1 & \text{when } L(1,0)p(1|x) > L(0,1)p(0|x). \end{cases} \tag{1.2.7}$$

Example. Let us continue with our example. Let g be a classifier. The risk of g is

$$\begin{aligned} R(g) &= \int L(y, g(x))dF(x,y) = \sum_{x_1,x_2,i} L(i, g(x_1,x_2))p(x_1,x_2,i) \\ &= \sum_{x_1,x_2} \Big(\sum_{i=0}^{1} L(i, g(x_1,x_2))p(i|x_1,x_2) \Big) p(x_1,x_2) = \sum_{x_1,x_2} R(g(x_1,x_2)|(x_1,x_2))p(x_1,x_2), \end{aligned} \tag{1.2.8}$$

where the conditional risk

$$R(g(x_1,x_2)|(x_1,x_2)) = L(0, g(x_1,x_2))p(0|x_1,x_2) + L(1, g(x_1,x_2))p(1|x_1,x_2)$$

is the average loss that g causes when classifying the objects with features (x_1,x_2). For calculating risk, we multiply that loss with the proportion of those people $p(x_1,x_2)$. When there are few such individuals, i.e. $p(x_1,x_2) \approx 0$, then such people do not influence much the risk of g. If $p(x_1,x_2)$ is big, then classifying (x_1,x_2) influences the whole risk remarkably.

1.2.3 Symmetric loss

The Bayes classifier g^* depends on loss function L. The most common loss function is **symmetric or 0-1 loss**. For symmetric loss $L(i,i)=0$ and the loss of misclassification is always the same. Without loss of generality, it can be one. Hence, the symmetric loss is

$$L(j,i)=\begin{cases}0 & \text{when } i=j,\\ 1 & \text{when } i\neq j.\end{cases} \tag{1.2.9}$$

With this loss, the conditional risk is

$$R(i|x)=\sum_{j=0}^{k-1}L(j,i)p(j|x)=\sum_{j\neq i}p(j|x)=1-p(i|x). \tag{1.2.10}$$

With other words, $R(i|x)$ is the probability that x does not belong to the class i. From (1.2.5)) we get the Bayes classifier

$$g^*(x)=\arg\min_i R(i|x)=\arg\min_i(1-p(i|x))=\arg\max_i p(i|x). \tag{1.2.11}$$

When loss function is symmetric, then Bayes classifier assigns to every feature vector the class with biggest conditional probability.

For every g, the conditional risk with symmetric loss function is

$$R(g(x)|x)=1-p(g(x)|x),$$

i.e. $R(g(x)|x)$ is the conditional probability of misclassification. Averaging the conditional probability of misclassification over the feature distribution $F(x)$, we get the (overall) **misclassification probability**. Thus when loss-function is symmetric, then the risk of a classifier is the probability of misclassification and the Bayes classifier has the lowest (overall) misclassification probability.

Note that (by symmetric loss-function) the Bayes risk or misclassification probability can be calculated as

$$R^*=\int\min_i R(i|x)dF(x)=\int\min_i(1-p(i|x))dF(x)=1-\int\max_i p(i|x)dF(x). \tag{1.2.12}$$

The special case: $k=2$. In this case $p(1|x)+p(0|x)=1$, hence $p(1|x)\geq p(0|x)$ iff $p(1|x)\geq 0.5$. Thus, in this case (by symmetric loss)

$$g^*(x)=\begin{cases}1 & \text{if } p(1|x)\geq 0.5,\\ 0 & \text{if } p(1|x)<0.5.\end{cases} \tag{1.2.13}$$

Note: when $p(1|x)=0.5$, then $R(g(x)|x)=0.5$ for every classifier and for such features, the Bayes classifier can also assign class 0. For those features Bayes classifier is not unique.

Since $k = 2$, the conditional risk of Bayes classifier at x is

$$R(g^*(x)|x) = \min\{1 - p(0|x), 1 - p(1|x)\} = \min\{p(0|x), p(1|x)\},$$

so that Bayes risk can be found as

$$R^* = \int \min\{p(0|x), p(1|x)\} dF(x). \tag{1.2.14}$$

Exercise: Let $k = 2$ and the loss function L be defined as follows:

$$L(i,j) = \begin{cases} 0, & \text{if } i = j; \\ a, & \text{if } i = 1, j = 0; \\ b, & \text{if } i = 0, j = 1. \end{cases}$$

Find Bayes classifier via $p(1|x)$ (i.e. generalize (1.2.13)). Find Bayes risk and generalize (1.2.14). How these formulas look like when $a = b$?

Exercise: Let $d = 1$, $k = 2$. For every function $h : \mathbb{R} \to \mathbb{R}$, let us define the loss function

$$J_p(h) = \int_{\mathbb{R}\times\mathcal{Y}} |h(x) - y|^p dF(x, y).$$

Find

$$h_p^* = \arg\inf_h J_p(h), \quad p = 1, 2.$$

Show that $J_1(h_1^*) = R^*$.

Exercise: Prove that (1.2.14) is equivalent to

$$R^* = \frac{1}{2} - \frac{1}{2}\int |2p(1|x) - 1| dF(x).$$

1.2.4 Densities

Let $F(x|i)$, $i = 0, \dots, k-1$ be the conditional distribution (functions) of the feature vector given the class is i. Thus, by the law of total probability

$$F(x) = \sum_{i=0}^{k-1} F(x|i)\pi_i,$$

where the probability that randomly chosen object belongs to class i is

$$\pi_i := \int_{\mathbb{R}^d} p(i|x) dF(x), \quad i = 0, \dots, k-1.$$

Let f_i be the density of $F(x|i)$, $i = 0, \dots, k-1$ with respect to a reference measure. If the reference measure is Lebesgue'i measure, then all densities are absolutely continuous and f_i is then the density as known from the basic probability course. When, for every i, the distributions $F(x|i)$ are discrete, then the reference measure is counting measure and the functions f_i are probability mass functions. Of course, it might be that some distributions $F(x|i)$ are discrete and some other continuous. Also, in this case there exists a reference measures so that all distributions are absolutely continuous with respect to. One such a reference measure can be $F(x|0) + \cdots + F(x|k-1)$.

<u>To summarize:</u> the existence of f_i w.r.t. some reference measure is not restrictive. In the following, we denote the reference measure all conditional densities are absolutely continuous with as dx. Hence, for every measurable function h

$$\int_{\mathbb{R}^d} h(x)dF(x|i) = \int_{\mathbb{R}^d} h(x)f_i(x)dx.$$

In this case also the distribution $F(x)$ has the density $f(x) = \sum_i f_i(x)\pi_i$.

With conditional densities the probability $p(i|x)$ is (the Bayes formula)

$$p(i|x) = \frac{f_i(x)\pi_i}{f(x)}.$$

Hence, the Bayes classifier (1.2.5) is

$$g^*(x) = \arg\min_{i\in\mathcal{Y}} \sum_{j=0}^{k-1} L(j,i)\frac{\pi_i f_i(x)}{f(x)} = \arg\min_{i\in\mathcal{Y}} \sum_{j=0}^{k-1} L(j,i)\pi_i f_i(x). \tag{1.2.15}$$

Symmetric loss. For 0-1 loss, (1.2.15) is (recall also (1.2.11))

$$g^*(x) = \arg\max_{i\in\mathcal{Y}} \pi_i f_i(x). \tag{1.2.16}$$

Since $R(i|x) = 1 - p(i|x)$, the function $R(i|x)f(x)$ is then

$$R(i|x)f(x) = f(x) - \pi_i f_i(x) = \sum_{j,j\neq i} \pi_j f_j(x).$$

Therefore, the Bayes risk is (recall (1.2.12)):

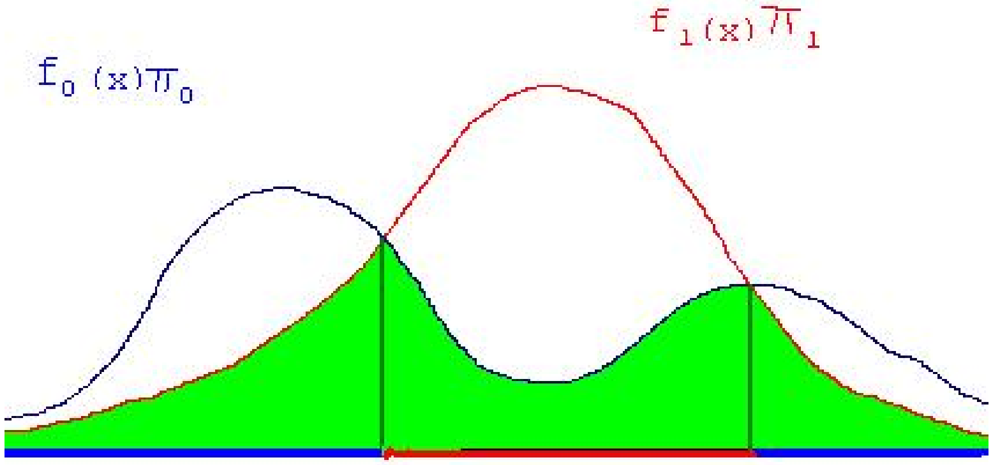
$$\begin{aligned} R^* &= \int \min_i R(i|x)dF(x) = \int \min_i R(i|x)f(x)dx = \int \min_i \big(\sum_{j,j\neq i} \pi_j f_j(x)\big)dx \\ &= \int \big(f(x) - \max_i \pi_i f_i(x)\big)dx = 1 - \int \max_i \pi_i f_i(x)dx. \end{aligned}$$

When $k = 2$ and L is symmetric, then the Bayes classifier (1.2.16) can be defined via **likelihood ratio** as follows

$$g^*(x) = \begin{cases} 1, & \text{if } \frac{f_1(x)}{f_0(x)} \geq \frac{\pi_0}{\pi_1}; \\ 0, & \text{else.} \end{cases} \tag{1.2.17}$$

The Bayesi risk in this case is (recall (1.2.14))

$$R^* = \int \min \{\pi_1 f_1(x), \pi_0 f_0(x)\} \, dx. \tag{1.2.18}$$



Example. In our example, π_0 and π_1 are the proportions of male and female individuals in the population. Let $p(x|0)$ and $p(x|1)$ be the distribution of the features in different classes:

$$p(x|0) = p(x_1, x_2|0) = \frac{\text{the number of men with the length } x_1 \text{ and weight } x_2}{\text{number of men}}.$$

according to Bayes formula

$$p(i|x) = \frac{p(x,i)}{p(x)} = \frac{p(x|i)\pi_i}{p(x)}, \quad i = 0, 1.$$

Hence $p(0|x) < p(1|x)$ iff $p(x|0)\pi_0 < p(x|1)\pi_1$ and (symmetric loss) the Bayes classifier is

$$g^*(x) = \begin{cases} 0 & \text{if } \pi_0 p(x|0) < \pi_1 p(x|1), \\ 1 & \text{else.} \end{cases}$$

This is the formula (1.2.16). Bayes risk (1.2.14) is

$$\begin{aligned} R^* &= \int \min\{p(0|x), p(1|x)\} dF(x) = \sum_x \min\{p(0|x), p(1|x)\} p(x) \\ &= \sum_x \min\{\frac{p(x|0)\pi_0}{p(x)}, \frac{p(x|1)\pi_1}{p(x)}\} p(x) = \sum_x \min\{p(x|0)\pi_0, p(x|1)\pi_1\}. \end{aligned}$$

The right hand side of previous equalities is (1.2.18).

Exercise: Generalize the formulas (1.2.17) and (1.2.18) for the following loss function

$$L(i,j) = \begin{cases} 0, & \text{if } i = j; \\ a, & \text{if } i = 1, j = 0; \\ b, & \text{if } i = 0, j = 1. \end{cases}$$

Exercise: Let x the number of hours a student spends for studying. Let $p(1|x)$ the probability that a student passes the test given that he or she studies x hours. Assume

$$p(1|x) = \frac{x}{c+x}, \quad c > 0.$$

Find the Bayes classifier to decide whether a student passes the test or not based on x. Suppose that x is uniformly distributed from 0 to $4c$, i.e

$$f(x) = \frac{1}{4c} I_{[0,4c]}.$$

Find π_i, f_i and Bayes risk ($R^* = \frac{1}{4} \ln \frac{5e}{4}$, $\pi_1 = 1 - \frac{1}{4} \ln 5$).

Exercise: Prove that when $\pi_0 = \pi_1$, then (1.2.18) is

$$R^* = \frac{1}{2} - \frac{1}{4} \int |f_1(x) - f_0(x)| dx.$$

1.3 Reject option

The misclassification probability could be reduced by a possibility to reject the decision. Formally, for a classifier, there is an additional output "r" that means that no decision has been made. Hence, with reject option, the classifier is a function

$$g : \mathbb{R}^d \to \mathcal{Y} \cup \{r\}. \tag{1.3.1}$$

Hence

$$g = \sum_{i=0}^{k-1} i I_{C_i} + r I_R,$$

where $\{C_0, \ldots, C_{k-1}, R\}$ is a partition of $\mathbb{R}^d$; the part R is so-called **reject region** and its complement $A := \cup_{i=0}^{k-1} C_i$ is **acceptance region**.

Recall that Bayes classifier $g^*(x)$ minimizes the conditional risk: $R(g^*(x)|x) = \min_i R(i|x)$. Hence, Bayes classifier is relatively risk-free when $\min_i R(i|x)$ is small. Hence, it is natural to consider the region R in the form

$$R(t) = \{x : \min_i R(i|x) > t\}, \tag{1.3.2}$$

where $t \geq 0$ is a threshold. In this case, the acceptance region is

$$A(t) = \{x : \min_i R(i|x) \leq t\}$$

and on this region, it is natural to use Bayes classifier. Thus, we get a new classifier with reject option as follows

$$g_t(x) := \begin{cases} \arg\min_i R(i|x), & \text{if } \min_i R(i|x) \leq t\ ; \\ r, & \text{if } \min_i R(i|x) > t. \end{cases}$$

The reject region should be as small as possible. Here the "smallness" of a set A is measured in terms of F-probability:

$$P(A) := \int_A f(x)dx = \int_A dF,$$

where f is the density of the distribution F. In other words, $P(A)$ is the probability that random feature vector takes a value in the set A. The following lemma shows that the risk of g_t is smallest amongst all classifiers with reject option and $P(A)$ at least as big as that of g_t.
Recall that for any function $h : \mathbb{R}^d \to \mathbb{R}$,

$$\int h(x)dF(x) = \int h(x)f(x)dx.$$

Lemma 1.3.1 *Let g a classifier with reject and acceptance region R and A, respectively. When $P(R) \leq P(R(t))$, then*

$$\int_A R(g(x)|x)dF(x) \geq \int_{A(t)} R(g_t(x))dF(x) = \int_{A(t)} \min_i R(i|x)dF(x). \tag{1.3.3}$$

Proof. Clearly,

$$\int_A R(g(x)|x)dF(x) \geq \int_A \min_i R(i|x)dF(x).$$

Hence, it suffices to show that

$$\int_A \min_i R(i|x)dF(x) \geq \int_{A(t)} \min_i R(i|x)dF(x). \tag{1.3.4}$$

The proof of (1.3.4) is left as the exercise. ■

Exercise: Prove the following inequality

$$\int_A \min_i R(i|x)dF(x) - \int_{A(t)} \min_i R(i|x)dF(x) \geq t(P(A) - P(A(t)) \geq 0.$$

Symmetric loss-function and reject option. For symmetric loss-function, we have $R(i|x) = 1 - p(i|x)$, hence

$$g_t(x) := \begin{cases} \arg\max_i p(i|x), & \text{kui } \max_i p(i|x) \geq 1 - t \ ; \\ r, & \text{kui } \max_i p(i|x) < 1 - t. \end{cases} \tag{1.3.5}$$

The region is decreasing as t increases $R(t)$. Note that when $t \geq \frac{k-1}{k}$, then $R(t) = \emptyset$. Hence, for $k = 2$, the reject option is only for $t < 0.5$ and denoting $0.5 - t =: c$ we get that g_t is in the following form

$$g_t(x) := \begin{cases} 1, & \text{if } p(1|x) \geq \frac{1}{2} + c \ ; \\ 0, & \text{if } p(1|x) \leq \frac{1}{2} - c \ ; \\ r, & \text{if } |\frac{1}{2} - p(1|x)| < c. \end{cases} \tag{1.3.6}$$

Chapter 2

Introduction to VC-theory

2.1 Supervised learning from data

We saw that when the conditional probabilities $p(i|x)$ or, equivalently, the conditional distribution $F(y|x)$ is known for every x, then finding the best possible classifier g^* is easy (recall (1.2.5)). We also know that the probabilities $p(i|x)$, and therefore the Bayes classifier g^* can easily be found if the class-conditional distributions F_i (or, equivalently, the densities f_i) and the probabilities π_i are known for every $i = 0, \dots k-1$ (recall (1.2.15)). Knowing F_i and π_i is equivalent to knowing the joint distribution $F(x, y)$, since

$$F(x,y) = \sum_{i=1}^{y} F_i(x)\pi_i.$$

However, in practice usually neither the conditional distribution $F(y|x)$ nor the full distribution $F(x, y)$ is exactly known. Instead we have the **(training) sample** or **(training) data**:

$$\mathcal{D}_n := \{(x_1, y_1), \dots, (x_n, y_n)\}. \tag{2.1.1}$$

This sample consists of n objects with known feature vectors $x_1, \dots, x_n$ and corresponding classes $y_1, \dots, y_n$. Typically, it is assumed that all pairs (x_i, y_i) are obtained independently from the same unknown distribution $F(x, y)$. Such a sample is called **iid** (independent and identically distributed) random sample. As usually in statistics, these assumptions are not always justified and can be wrong. However, iid sample is a natural assumption to start with, hence during the supervised learning part of the course, we consider that case. The objective of **statistical pattern recognition** is to find a good classifier based on the iid sample $\mathcal{D}_n$. In statistical learning terminology, the procedure of finding a good classifier based on $\mathcal{D}_n$ is called **supervised learning or training**. The word "supervised" reflects the fact that the training data are given with labels, i.e. for every x_i in the sample, the correct label y_i is known.
The obtained classifier is thus a function that depends on data

$$g_n : (\mathbb{R}^d \times \mathcal{Y})^n \times \mathbb{R}^d \to \mathcal{Y} \tag{2.1.2}$$

and its risk

$$R_n := R(g_n) = \int L(y, g(\mathcal{D}_n, x))dF(x,y) \tag{2.1.3}$$

depends on data as well. Hence R_n is a random variable and it is reasonable to design g_n such that it expected over all possible samples $\mathcal{D}_n$ risk were as small as possible. This does obviously not guarantee that for given $\mathcal{D}_n$ the risk R_n is small as well.

When designing a classifier, it is natural to expect that it is an outcome of a rule that applies for any n and any data. We shall see several rules in following sections, let us now mathematically formulate a **(classification or discrimination) rule** as a sequence of classifiers $g_1, g_2, \ldots, g_k, \ldots$ Essentially, a rule is an algorithm or principle that, in general, does not depend on the size or configuration of the sample. However, applying a rule for the data, we get a concrete classifier g_n as in (2.1.2) that depends on the data.

For example, if $k = 2$, then a rule might be: if the sample $y_1, \ldots, y_n$ contains strictly more ones as zeros, classify any unknown object as one, otherwise as zero. Such a rule applies for any sample, but the outcome depends on the sample. Obviously, this is not very smart rule, since it does not take into account the feature vectors.

2.2 Mathematical formulation

Let us formulate the introduced concepts in terms of probability theory. Let $(\Omega, \mathcal{F}, \mathbf{P})$ be a probability space and (X, Y) a random vector, where X is $\mathbb{R}^d$-dimensional and Y is $\mathcal{Y}$-valued random variable. Let, $\forall x \in \mathbb{R}^d$ and $\forall y \in \mathbb{R}$

$$F(y,x) := \mathbf{P}(X \leq x, Y \leq y), \quad F(y|x) := \mathbf{P}(Y \leq y|X = x), \quad F(x|i) = \mathbf{P}(X \leq x|Y = i).$$

Here, for $x, x' \in \mathbb{R}^d$, the inequality $x \leq x'$ is defined componentwise). Clearly (X, Y) models the feature vector and its class.
For every classifier g, its risk is

$$R(g) = EL(Y, g(X)) = \int L(Y, g(X))d\mathbf{P}.$$

If L is symmetric loss-function (1.2.9), then

$$R(g) = \int I_{\{Y \neq g(X)\}} d\mathbf{P} = \mathbf{P}(Y \neq g(X))$$

so that (as we know) for symmetric loss the risk is the misclassification probability.

Let

$$D_n := ((X_1, Y_1), \ldots, (X_n, Y_n)) \tag{2.2.1}$$

iid copies of (X, Y); D_n models the training data. The classifier g_n depends on the vectors (2.2.1), formally thus

$$g_n(D_n, \cdot) : \mathbb{R}^d \mapsto \mathcal{Y}. \tag{2.2.2}$$

Given the data, i.e. given the values $(X_i, Y_i) = (x_i, y_i)$, $i = 1, \ldots, n$, (2.2.2) is a fixed classifier with fixed risk

$$R_n(\mathcal{D}_n) = EL(Y, g_n(\mathcal{D}_n; X)),$$

where (X, Y) is independent of D_n. Considering $\mathcal{D}_n$ random, we get that the risk R_n depends on random sample D_n and, therefore, is a random variable.

2.3 Consistency

Since $R_n = R(g_n)$ depends on data, how to measure the goodness of g_n? As mentioned earlier, a possibility is to consider the expected risk ER_n, where the expectation is taken over all samples. Note that for every sample, $R_n \geq R^*$. Hence $ER_n = R^*$ iff $R_n = R^*$, a.s. implying that expected risk equals to Bayes risk iff the classifier $g_n(\mathcal{D}_n, \cdot)$ is the Bayes classifier for almost every sample $\mathcal{D}_n$. This is impossible for most cases in practice. Hence, in general, $ER_n > R^*$ for every n.

On the other hand, when dataset is big (n is large), then we have so much information about unknown distribution $F(x, y)$ that it should be possible to design a classifier g_n such that $ER(g_n)$ is close to R^*. That consideration brings us to consistent rules. Recall that a classification rule is just a sequence of classifiers $\{g_n\}$, where g_n is as in (2.2.2).

Definition 2.3.1 *A classification rule $\{g_n\}$ is* **consistent** *for a certain distribution $F(x, y)$, if*

$$ER(g_n) \to R^* \tag{2.3.1}$$

and **strongly consistent** *, if*

$$R(g_n) \to R^* \quad a.s.$$

If a rule is (strongly) consistent for every distribution $F(x, y)$, it is called **universally (strongly) consistent** *.*

Consistency characterizes a rule not a single classifier. Since $0 \leq R_n \leq \max_{i,j} L(i, j)$ the convergence (2.3.1) is equivalent to the convergence in probability , i.e. $\forall \epsilon > 0$,

$$\lim_n \mathbf{P}(R(g_n) - R^* > \epsilon) = 0.$$

Consistency is a good property. A consistent rule guarantees that by increasing the amount of data the probability that the risk is within a very small distance of the optimal achievable gets arbitrarily close to one. Strong consistency means that by using more data the error probability gets arbitrarily close to R^* for every training sequence except for a set of sequences that has zero probability altogether. Since a.s. convergence implies the convergence in probability, the strong consistency is indeed stronger property than consistency.

It is important that for a consistent rule, even for big n, the classifier g_n need not to be similar or (in some sense) close to Bayes classifier g^*, i.e. the fact that $\{g_n\}$ is consistent does not necessarily imply $g_n \to g^*$ (in some sense), but the classification properties of g_n and g^* are close. In the terminology of Vapnik consistent rule imitates but does not identify the Bayes classifier. But what would we expect from a good machine? To work almost as good as the best possible or to look almost like the best possible?

A rule can be consistent for a certain class of distributions $F(x, y)$, but may not be consistent for others. However, in most of the cases, we do not have any idea about $F(x, y)$. Therefore, it is very desirable to have a rule that is consistent for all distributions. Such rules are universally consistent. The universal consistency is a very strong requirement and for a while it was not clear whether universally consistent rules exists.

No super-rules. The existence of universally consistent rules might arise question, whether there exists an universally consistent rule $\{g_n\}$ such that the speed of convergence $ER(g_n)$ were fastest over all other rules for every distribution $F(x, y)$. The following theorem ([1], Thm 7.1) shows that such best rule does not exists.

Theorem 2.3.2 (No Free Lunch) *Let $k = 2$ and L symmetric. For every $\epsilon > 0$, n and classifier g_n, there exists a distribution $F(x, y)$ such that $R^* = 0$ and $ER(g_n) > \frac{1}{2} - \epsilon$.*

Note that the theorem does not contradict the universal consistency.

Another question that might arise: whether there exists universally best super-rule, i.e. rule that for every $F(x, y)$ minimizes the risk over all other classifiers at least for big n. This is not so – it can be shown that for every rule $\{g_n\}$, there exists the distribution $F(x, y)$ (that depends on the rule) and another rule $\{g'_n\}$ such that

$$ER(g'_n) < ER(g_n), \quad \forall n.$$

In the light of of Theorem 2.3.2, the non-existence of super-rule is not that surprising.

2.4 Empirical risk minimization

In general, with the help of data, a good classifier is searched from a class of classifiers $\mathcal{G}$. Sometimes, the set $\mathcal{G}$ is also referred to as the **model**. Usually $\mathcal{G}$ is much smaller than the class of all classifiers. If the distribution $F(y,x)$ were known, we would choose the classifier that minimizes the risk over $\mathcal{G}$. In practice, however, instead of the distribution $F(y,x)$, we have the data $\mathcal{D}_n$. Recall that every sample can be considered as the empirical distribution with empirical distribution function F_n. Recall that for every $(x,y) \in \mathbb{R}^{d+1}$

$$F_n(x,y) = \frac{1}{n}\sum_{i=1}^n I_{\{x_i \le x, y_i \le y\}},$$

and for any function $h : \mathbb{R}^d \times \mathcal{Y} \to \mathbb{R}$,

$$\int h(x,y)dF_n(x,y) = \frac{1}{n}\sum_{i=1}^n h(x_i,y_i). \tag{2.4.1}$$

The empirical distribution (function) F_n is an estimate to unknown distribution (function) F. Recall that F_n is a good estimate of F, because of the **Glivenko-Cantelli theorem:**

$$\sup_{x,y} |F_n(x,y) - F(x,y)| \to 0, \quad a.s..$$

This gives the idea: when calculating the risk $R(g)$, replace the unknown $F(x,y)$ by its empirical version $F_n(x,y)$ and use (2.4.1) to obtain the **empirical risk**

$$R_n(g) := \int L(y,g(x))dF_n(y,x) = \frac{1}{n}\sum_{i=1}^n L(y_i, g(x_i)). \tag{2.4.2}$$

The classifier g_n is now the one that minimizes empirical risk over $\mathcal{G}$, i.e.

$$g_n = \arg\inf_{g\in\mathcal{G}} R_n(g).$$

This method of finding g_n is called as **empirical risk minimization (ERM) principle.** When L is symmetric, then empirical risk, in this case also called **empirical error** is

$$R_n(g) = \frac{1}{n}\sum_{i=1}^n I_{\{y_i \neq g(x_i)\}}.$$

Thus $R_n(g)$ is proportional to the misclassified objects. When L is symmetric, then the empirical risk minimization principle chooses such classifier(s) that minimizes the number of misclassified objects (empirical or training error) in training sample.

2.4.1 Empirical risk minimization in general

ERM is a simple yet powerful principle that applies in many fields of statistics. Let us consider this principle in a broader context. Suppose we have an iid sample from $F(x,y)$

$$\mathcal{D}_n := (x_1,y_1),\ldots,(x_n,y_n), \tag{2.4.3}$$

where $(x_i,y_i)\in\mathcal{X}\times\mathcal{Y}$ and $\mathcal{X}$, $\mathcal{Y}$ are arbitrary. We interprate $x_1,\ldots,x_n$ as inputs and $y_1,\ldots,y_n$ as corresponding outputs. The conditional distribution (function) of output given the input x is $F(y|x)$. The objective is to predict the output to an input x.

Let $\mathcal{G}$ be a class of functions $\mathcal{X}\to\mathcal{Y}$, $L(x,y)$ a loss function (depends on task) and for every $g\in\mathcal{G}$, the risk $R(g)$ is (as previously)

$$R(g) := \int L(y,g(x))dF(y,x). \tag{2.4.4}$$

Examples of statistical problems

Classification (pattern recognition). Here $\mathcal{Y}=\{0,\ldots,k-1\}$ (classes) and g has values in $\mathcal{Y}$. When L is symmetric, then the risk is the misclassification probability and the best classifier over all possible functions is Bayes classifier:

$$g^*(x)=\arg\max_i p(i|x).$$

Regression. Here $\mathcal{Y}=\mathbb{R}$ and loss-function L is typically $L(y,g)=(y-g)^2$. Risk is

$$R(g)=\int (y-g(x))^2 dF(x,y)$$

and the best possible prediction over all possible function is conditional mean:

$$g^*(x)=\int y dF(y|x).$$

Density estimation. Here sample consists of imputs, only $x_1,\ldots,x_n$ (formally $\mathcal{Y}=\emptyset$) and $\mathcal{G}$ is a class of densities. Let $L(g)=-\ln g$. Thus the risk of g is **likelihood contrast**:

$$R(g)=-\int \ln g(x)dF(x).$$

In case F has density $f(x)$, then

$$R(g)=-\int \ln g(x)f(x)dx$$

and the function that minimizes $R(g)$ over all possible densities is $f(x)$, i.e. $g^*(x)=f(x)$.

ERM-principle: Replace risk (2.4.4) by empirical risk

$$R_n(g) := \int L(y,g(x))dF_n(y,x)=\frac{1}{n}\sum_i L(y_i,g(x_i)). \tag{2.4.5}$$

Examples of ERM-principles

Classification (pattern recognition) For symmetric L, the ERM-principle is minimizing empirical error:

$$R_n(g) = \frac{1}{n}\sum_{i=1}^n L(y_i, g(x_i)) = \frac{1}{n}\sum_{i=1}^n I_{\{(x,y):g(x)\neq y\}}.$$

Regression. ERM-principle is the principle of least squares: minimize

$$R_n(g) = \frac{1}{n}\sum_{i=1}^n L(y_i, g(x_i)) = \frac{1}{n}\sum_{i=1}^n (y_i - g(x_i))^2.$$

Density estimation. ERM-principle is maximum likelihood principle: minimize

$$R_n(g) = -\frac{1}{n}\sum_{i=1}^n \ln g(x_i).$$

This equals to maximizing the log-likelihood.

Thus, ERM has been successfully exploited in statistics under various names more than 100 years. Vapnik-Chervonenkis theory that studies ERM principle in general, provides an unified approach all these (and many other) methods.

2.5 Approximation error and estimation error

In general, with the help of data, a good classifier is searched from a class of classifiers $\mathcal{G}$. Let g_n be a classifier selected from $\mathcal{G}$ by some rule using data. The difference $R(g_n) - R^* \geq 0$ can split into two parts:

$$R(g_n) - R^* = \Big(R(g_n) - \inf_{g\in\mathcal{G}} R(g)\Big) + \Big(\inf_{g\in\mathcal{G}} R(g) - R^*\Big). \tag{2.5.1}$$

The first part – **estimation error** – depends on the sample and the method of choosing g_n. Since $R(g_n)$ depends on sample, so does the estimation error. The larger is $\mathcal{G}$, the larger estimation error.
The second part – **approximation error** – depends on $\mathcal{G}$ but not on sample. The bigger $\mathcal{G}$, the smaller approximation error. If $\mathcal{G}$ is the class of all classifiers, then the approximation error is zero. Then the estimation error is typically very big.

To get a consistent rule by selecting the classifiers from $\mathcal{G}$, it has to depend on n and $\mathcal{G}_n$ has to grow so that the approximation error tends to zero. On the other hand, the growth must be slow enough to control the estimation error. The question of the right size – so-called *complexity* of the class $\mathcal{G}$ – is central in the theory of statistical learning.

As an example of too large $\mathcal{G}$, let it be the class of all classifiers, and let g_n be chosen using empirical risk minimization. When the distribution of X is absolutely continuous, then with probability one, all sample points x_i are different. Then, clearly, the class $\mathcal{G}$ contains classifiers that classify all training sample correctly, i.e. the empirical error is zero. One such a classifier is, for instance, the following:

$$g_n(x) = \begin{cases} y_i & \text{if } x = x_i, \\ 0 & \text{otherwise.} \end{cases}$$

Note that $g_n(X) = 0$, a.s.. If L is symmetric, then

$$R(g_n) = \mathbf{P}(Y \neq g(X)) = \mathbf{P}(Y \neq 0).$$

Thus $R(g_n)$ does not depend on the training sample, and it is, in general, strictly positive. Typically $R^* = \mathbf{P}(g^*(X) \neq Y) < \mathbf{P}(Y \neq 0)$, hence the estimation error $\mathbf{P}(Y \neq 0) - R^* > 0$ is positive constant for any n. Obviously such a g_n is not what we expect from a good method, a reason for bad g_n is too large $\mathcal{G}$. This phenomenon is called **overfitting**. Note that when $\mathcal{G}$ is maximal, then $\inf_{g\in\mathcal{G}} R(g) = R^*$ and the approximation error equals to zero.

As an example of too little $\mathcal{G}$, let it consists of one function, only, i.e. $\mathcal{G} = \{g\}$.Then $g_n = g$ and $R(g_n) = R(g) = \inf_{\mathcal{G}} R(g)$, i.e. estimation error is zero. However, the approximation error is now $R(g) - R^*$, that can be very big.

Hence, a good classifier g_n should be chosen from a class that in some sense has optimal size. Instead of the physical size, what matters is how many different classifiers the class has. This is called **complexity**.

Estimation and generalization error

Let $\mathcal{G}$ be a class of classifiers, and let $g_n \in \mathcal{G}$ chosen by some method (not necessarily ERM) from that class using training sample. Recall that sample is random, so is g_n and its risk $R(g_n)$. Also recall that $R_n(g_n)$ is the empirical risk of g_n that, obviously, is random as well. The central objects of VC- theory are:

- estimation error (*excess risk*):
$$R(g_n) - \inf_{g\in\mathcal{G}} R(g)$$
- *generalization error*:
$$|R(g_n) - R_n(g_n)|.$$

The theory aims to find upper bounds to these random variables. Typical bounds are in form

$$\mathbf{P}\big(R(g_n) - \inf_{g\in\mathcal{G}} R(g) > \epsilon\big) \leq \delta_1(\epsilon, n, \mathcal{G}) \tag{2.5.2}$$

$$\mathbf{P}\big(|R(g_n) - R_n(g_n)| > \epsilon\big) \leq \delta_2(\epsilon, n, \mathcal{G}), \tag{2.5.3}$$

where $\delta_i(\epsilon, n, \mathcal{G})$ are functions that presumably converge to zero as n grows. The bounds (2.5.2) are important for consistency, the bounds (2.5.3) allow to estimate the unknown risk $R(g_n)$ and unknown quantity $\inf_{g\in\mathcal{G}} R(g)$ using the (known) empirical risk $R_n(g_n)$:

$$\mathbf{P}(\inf_{g\in\mathcal{G}} R(g) \leq R_n(g_n) + \epsilon) \geq \mathbf{P}\big(R(g_n) \leq R_n(g_n) + \epsilon\big) \geq 1 - \delta_2(\epsilon, n, \mathcal{G}). \tag{2.5.4}$$

In statistical learning, ϵ is usually considered as a function of δ, so that (2.5.2) and (2.5.4) can be reformulated as

$$\text{with probability } 1-\delta \text{ (over samples): } R(g_n) - \inf_{g\in\mathcal{G}} R(g) \leq \epsilon_1(\delta, n, \mathcal{G}) \tag{2.5.5}$$

$$\text{with probability } 1-\delta \text{ (over samples): } \inf_{g\in\mathcal{G}} R(g) \leq R(g_n) \leq R_n(g_n) + \epsilon_2(\delta, n, \mathcal{G}). \tag{2.5.6}$$

The inequalities (2.5.5) and (2.5.6) are sometimes called **PAC** (*probably almost correct*) bounds.
Thus, important inequalities are (2.5.2) and (2.5.3). Following simple but yet powerful lemma shows that when g_n is obtained by ERM-principle, then both inequalities follow from the upper bound to the quantity $\sup_{g\in\mathcal{G}} |R_n(g) - R(g)|$.

Lemma 2.5.1 (Vapnik, Chervonenkis, 1974) *Let $g_n \in \mathcal{G}$ be arbitrary classifier and let, $\hat{g} \in \mathcal{G}$ be obtained by empirical risk minimization, i.e.*

$$\hat{g} = \arg\inf_{g\in\mathcal{G}} R_n(g).$$

Then

$$|R_n(g_n) - R(g_n)| \leq \sup_{g\in\mathcal{G}} |R_n(g) - R(g)| \tag{2.5.7}$$

$$R(\hat{g}) - \inf_{g\in\mathcal{G}} R(g) \leq 2\sup_{g\in\mathcal{G}} |R_n(g) - R(g)|. \tag{2.5.8}$$

Proof. The first inequality is obvious.
The second inequality: Let $g^* \in \mathcal{G}$ be such that

$$R(g^*) = \inf_{g\in\mathcal{G}} R(g).$$

For time being, let us suppose that g^* exists. Then

$$\begin{aligned} R(\hat{g}) - \inf_{g\in\mathcal{G}} R(g) = R(\hat{g}) - R(g^*) &= R(\hat{g}) - R_n(\hat{g}) + R_n(\hat{g}) - R_n(g^*) + R_n(g^*) - R(g^*) \\ &\leq R(\hat{g}) - R_n(\hat{g}) + R_n(g^*) - R(g^*) \\ &\leq 2\sup_{g\in\mathcal{G}} |R_n(g) - R(g)|. \end{aligned}$$

If the optimal g^* does not exists, then $\forall\ \epsilon > 0\ \exists\ g_\epsilon \in \mathcal{G}$ such that $R(g_\epsilon) \leq \inf_{g\in\mathcal{G}} R(g) + \epsilon$. Then

$$\begin{aligned}
R(\hat{g}) - \inf_{g\in\mathcal{G}} R(g) &\leq R(\hat{g}) - R(g_\epsilon) + \epsilon \\
&\leq 2\sup_{g\in\mathcal{G}} |R_n(g) - R(g)| + R_n(\hat{g}) - R_n(g_\epsilon) + \epsilon \\
&\leq 2\sup_{g\in\mathcal{G}} |R_n(g) - R(g)| + \epsilon.
\end{aligned}$$

Since ϵ was arbitrary, we obtain (2.5.8). ■

Thus, the inequalities like

$$\mathbf{P}\Big(\sup_{g\in\mathcal{G}} |R_n(g) - R(g)| > \epsilon\Big) \leq \delta(\epsilon, n, \mathcal{G}). \tag{2.5.9}$$

are central in VC-theory. These inequalities are based on so-called **large deviation** or **concentration inequalities** that are an important subject of probability theory. It turns out that the probability

$$\mathbf{P}\Big(\sup_{g\in\mathcal{G}} |R_n(g) - R(g)| > \epsilon\Big) \tag{2.5.10}$$

depends on the complexity of the class $\mathcal{G}$. In an example above, we saw that when $\mathcal{G}$ is too large (too complex) and X absolutely continuous, then for (almost) any sample there exists $g \in \mathcal{G}$ such that $R_n(g) = 0$; in this case for any $\epsilon < R(g)$ and for any n, the probability (2.5.10) equals to one.

2.6 Shatter coefficient and VC-dimension

In this subsection, we consider the most common measure of complexity: VC dimension. Let $\mathcal{A}$ be a class of sets on $\mathbb{R}^d$.

Definition 2.6.1 *For any $n \in \mathbb{N}$, the* **shatter coefficient** *of $\mathcal{A}$ is*

$$\mathbb{S}_{\mathcal{A}}(n) := \max_{x_1,\dots,x_n} |\{x_1, \dots, x_n\} \cap A : A \in \mathcal{A}\}|.$$

Thus, $\mathbb{S}_{\mathcal{A}}(n)$ is the maximal number of different subsets of a set of n points which can ne obtained by intersecting it with elements of $\mathcal{A}$. Clearly $\mathbb{S}_{\mathcal{A}}(n) \leq 2^n$ and the equality holds iff there exist n points $\{x_1, \dots, x_n\}$ so that all possible subsets can be obtained by intersecting it with elements of $\mathcal{A}$. If this happens, we say that $\mathcal{A}$ **shatters** the set $\{x_1, \dots, x_n\}$.

Examples:

1. Let $d = 1$ and $\mathcal{A} := \{(-\infty, a] : a \in \mathbb{R}\}$. Then $\mathbb{S}_{\mathcal{A}}(n) = n + 1$.
2. Let $d = 2$ and $\mathcal{A} = \{(-\infty, a^1] \times (-\infty, a^2] : a = (a^1, a^2) \in \mathbb{R}^2\}$. Then (Exercise)
$$\mathbb{S}_{\mathcal{A}}(n) = 1 + \sum_{k=1}^{n}(n - k + 1) = 1 + \frac{n(n+1)}{2}.$$
3. Let $d = 1$ and $\mathcal{A} := \{[a, b] : a, b \in \mathbb{R}\}$. Then (Exercise)
$$\mathbb{S}_{\mathcal{A}}(n) = 1 + \sum_{k=1}^{n}(n - k + 1) = 1 + \frac{n(n+1)}{2}.$$
4. Let $d = 2$ and $\mathcal{A} = \{x : w'x \geq b : w \in \mathbb{R}^2, b \in \mathbb{R}\}$. Thus $\mathcal{A}$ is the class of all halfspaces. Then (Exercise)
$$\mathbb{S}_{\mathcal{A}}(2) = 4, \quad \mathbb{S}_{\mathcal{A}}(3) = 8, \quad \mathbb{S}_{\mathcal{A}}(4) = 14.$$
5. The example above can be generalized. Let $\mathcal{A}$ be the set of halfspaces in $\mathbb{R}^d$. Then, for $n \geq d + 1$
$$\mathbb{S}_{\mathcal{A}}(n) = 2\sum_{i=0}^{d}\binom{n-1}{i} \leq 2(n-1)^d + 2$$
([1], Cor. 13.1)

The properties of shatter coefficient:

Theorem 2.6.2 *Let $\mathcal{A}$ and $\mathcal{B}$ be classes of subsets of $\mathbb{R}^d$, and let $n, m \geq 1$ be integers. Then*

1. $\mathbb{S}_{\mathcal{A}}(n) \leq |\mathcal{A}|$;
2. $\mathbb{S}_{\mathcal{A}}(n + m) \leq \mathbb{S}_{\mathcal{A}}(n)\mathbb{S}_{\mathcal{A}}(m)$;
3. *If* $\mathcal{C} = \mathcal{A} \cup \mathcal{B}$, *then* $\mathbb{S}_{\mathcal{C}}(n) \leq \mathbb{S}_{\mathcal{A}}(n) + \mathbb{S}_{\mathcal{B}}(n)$;
4. *If* $\mathcal{C} = \{A^c : A \in \mathcal{A}\}$, *then* $\mathbb{S}_{\mathcal{A}}(n) = \mathbb{S}_{\mathcal{C}}(n)$;
5. *If* $\mathcal{C} = \{A \cap B : A, \in \mathcal{A}$ *and* $B \in \mathcal{B}\}$, *then* $\mathbb{S}_{\mathcal{C}}(n) \leq \mathbb{S}_{\mathcal{A}}(n)\mathbb{S}_{\mathcal{B}}(n)$;
6. *If* $\mathcal{C} = \{A \cup B : A, \in \mathcal{A}$ *and* $B \in \mathcal{B}\}$, *then* $\mathbb{S}_{\mathcal{C}}(n) \leq \mathbb{S}_{\mathcal{A}}(n)\mathbb{S}_{\mathcal{B}}(n)$;
7. *If* $\mathcal{C} = \{A \times B : A, \in \mathcal{A}$ *and* $B \in \mathcal{B}\}$, *then* $\mathbb{S}_{\mathcal{C}}(n) \leq \mathbb{S}_{\mathcal{A}}(n)\mathbb{S}_{\mathcal{B}}(n)$.

Proof. Exercise . ■

Definition 2.6.3 *The* **Vapnik-Cervonenkis (VC) dimension V of a class $\mathcal{A}$** *is defined as the largest integer n such that $\mathbb{S}_{\mathcal{A}}(n) = 2^n$. If $\mathbb{S}_{\mathcal{A}}(n) = 2^n$ for all n, then $V = \infty$.*

The definition is correct, because if $\mathbb{S}_{\mathcal{A}}(n) < 2^n$, then for any $m > n$, $\mathbb{S}_{\mathcal{A}}(m) < 2^m$.

Thus, VC dimension of the class $\mathcal{A}$ is the capacity of the largest set that the class $\mathcal{A}$ shatters. In other words, if VC dimension is V then for any $n > V$, there exists no set $\{x_1, \dots, x_n\}$ that class $\mathcal{A}$ shatters.

Examples:

1. Let $d = 1$ and $\mathcal{A} := \{(-\infty, a] : a \in \mathbb{R}\}$. Then $V = 1$. No set of two elements can be shattered.
2. Let $d = 2$ and $\mathcal{A} = \{(-\infty, a^1] \times (-\infty, a^2] : a = (a^1, a^2) \in \mathbb{R}^2\}$. Then $\mathbb{S}_{\mathcal{A}}(2) = 4 = 2^2$ and $\mathbb{S}_{\mathcal{A}}(3) = 1 + 6 = 7 < 2^3$. Hence $V = 2$.
3. The previous example can be generalized: if $\mathcal{A} = \{(-\infty, a^1] \times \cdots \times (-\infty, a^d] : (a^1, \dots, a^d) \in \mathbb{R}^d\}$, then $V = d$ (Exercise).
4. Let $d = 1$ and $\mathcal{A} := \{[a, b] : a, b \in \mathbb{R}\}$. Then $\mathbb{S}_{\mathcal{A}}(2) = 4 = 2^2$ and $\mathbb{S}_{\mathcal{A}}(3) = 1 + 6 = 7 < 2^3$. Hence $V = 2$.
5. The previous example can be generalized: if $\mathcal{A} = \{[a^1, b^1] \times \cdots \times [a^d, b^d] : a, b \in \mathbb{R}^d\}$, i.e. $\mathcal{A}$ is the class of all retangles, then $V = 2d$ ([1], Thm. 13.8).
6. Let $\mathcal{A}$ be the set of halfspaces in $\mathbb{R}^d$. Then, taking account that
$$\sum_{i=0}^{k} \binom{k}{i} = 2^k,$$
we get
$$S_{\mathcal{A}}(d+1) = 2\sum_{i=0}^{d} \binom{d}{i} = 2 \cdot 2^d = 2^{d+1}$$
$$S_{\mathcal{A}}(d+2) = 2\sum_{i=0}^{d} \binom{d+1}{i} = 2(2^{d+1} - 1) < 2^{d+1},$$
so that $V = d + 1$.
7. Let $\mathcal{A} = \{x \in \mathbb{R}^d : \|x - a\| \leq r, a \in \mathbb{R}^d, r \geq 0\}$ i.e. $\mathcal{A}$ is the class of all closed balls in $\mathbb{R}^d$. Then $V \leq d + 2$ ([1], Cor. 13.2).
8. In $\mathcal{A}$ is the class of convex polygons in $\mathbb{R}^2$, then $V = \infty$. Indeed, putting, for any n, the points $x_1, \dots, x_n$ in a circle, it is easy to see that any subset can be picked by a convex polygon.

By definition, if for a class $\mathcal{A}$, the VC dimension V is finite, then for any $n > V$, $\mathbb{S}_{\mathcal{A}}(n) < 2^n$. The following important lemma shows that in this case the growth is polynomial with power V. For the proof of the following lemma, see [1], Thm 13.2 and Thm 13.3, also [2], Corollary 1.3.

Lemma 2.6.1 (Sauer's lemma) *Let $\mathcal{A}$ be a class of sets with VC dimension $V < \infty$. Then, for all n.*

$$\mathbb{S}_{\mathcal{A}} \leq \sum_{i=0}^{V} \binom{n}{i} \leq (n+1)^V$$

and for all $n \geq V$,

$$\mathbb{S}_{\mathcal{A}} \leq \left(\frac{ne}{V}\right)^V$$

Hence, if $V \geq 3$, then $\mathbb{S}_{\mathcal{A}} \leq n^V$.

2.7 Vapnik-Cervonenkis inequality and risk bounds

Empirical measures. Let $Z_1, \ldots, Z_n$ be iid random vectors in $\mathbb{R}^d$. Recall that for any (measurable) set $A \subset \mathbb{R}^d$, the **empirical measure** $P_n(A)$ is the proportion of Z_i's in the set A:

$$P_n(A) := \frac{1}{n}\sum_{i=1}^{n} I_{\{Z_i \in A\}}.$$

Given A, $P_n(A)$ is a random variable with expectation

$$EP_n(A) = \mathbf{P}(Z_1 \in A) =: P(A)$$

Note that $nP_n(A) \sim B(n, P(A))$. By SLLN, $P_n(A) \to P(A)$, a.s.. Moreover, it is also known that by so-called **Höffding's inequality** for every $\epsilon > 0$

$$\mathbf{P}(|P_n(A) - P(A)| > \epsilon) \leq 2\exp[-2n\epsilon^2]. \tag{2.7.1}$$

Note that by Borel-Cantelli lemma, from (2.7.1) the convergence $P_n(A) \to P(A)$, a.s. follows.

Exercise: Let $\mathcal{A}$ be a class of measurable sets. Prove that

$$\mathbf{P}(\sup_{A\in\mathcal{A}} |P_n(A) - P(A)| > \epsilon) \leq 2|\mathcal{A}| \exp[-2n\epsilon^2]. \tag{2.7.2}$$

VC inequality. VC inequality strengthes (2.7.2) for infinite classes with finite VC-dimension.

Theorem 2.7.1 (Vapnik and Cervonenkis' 71) *For any class of sets $\mathcal{A}$, for any n and $\epsilon > 0$,*

$$\mathbf{P}\Big(\sup_{A\in\mathcal{A}} |P_n(A) - P(A)| > \epsilon\Big) \leq 8\mathbb{S}_{\mathcal{A}}(n)\exp[-\frac{\epsilon^2}{32}n] \tag{2.7.3}$$

$$E\Big(\sup_{A\in\mathcal{A}} |P_n(A) - P(A)|\Big) \leq 2\sqrt{\frac{\ln \mathbb{S}_{\mathcal{A}}(n) + \ln 2}{n}}. \tag{2.7.4}$$

For the proof of (2.7.3), see [1], Thm. 12.5; for the proof of (2.7.4), see [2], Thm 1.9.

The first inequality (2.7.3) is important and not trivial. To get from (2.7.3) an inequality like the second one (2.7.4) (with possible bigger constants) is rather easy. Indeed, it is not difficult to see that the following proposition holds.

Proposition 2.7.1 *Let Z nonnegative random variable so that for every $\epsilon > 0$*

$$\mathbf{P}(Z > \epsilon) \leq C(n)\exp[-A(n)\epsilon^2], \tag{2.7.5}$$

where $A(n) > 0$ and $C(n)$ are independent of ϵ. Then

$$EZ \leq \sqrt{\frac{\ln C(n) + 1}{A(n)}}. \tag{2.7.6}$$

Exercise: Prove Proposition. Hint: Use

$$(EZ)^2 \leq EZ^2 = \int_0^u \mathbf{P}(Z^2 > t)dt + \int_u^\infty \mathbf{P}(Z^2 > t)dt \leq u + \int_u^\infty \mathbf{P}(Z^2 > t)dt.$$

Find u that minimizes the upper bound.

Corollary 2.7.1 *If $\mathcal{A}$ has finite VC-dimension V, then*

$$\mathbf{P}\Big(\sup_{A\in\mathcal{A}} |P_n(A) - P(A)| > \epsilon\Big) \leq 8(n+1)^V\exp[-\frac{\epsilon^2}{32}n] \tag{2.7.7}$$

$$E\Big(\sup_{A\in\mathcal{A}} |P_n(A) - P(A)|\Big) \leq 2\sqrt{\frac{V\ln(n+1) + \ln 2}{n}}. \tag{2.7.8}$$

Proof. Just use the inequality $\mathbb{S}_{\mathcal{A}} \leq (n+1)^V$ in (2.7.3) and (2.7.4). ■

Exercise: Show how (2.7.7) implies the Clivenko-Cantelli theorem:

$$\sup_x |F_n(x) - F(x)| \to 0, \text{ a.s..}$$

Risk bounds for symmetric (0-1) loss. Let us consider the symmetric loss function, i.e.

$$L(g(x),y) = I_{A_g}, \quad A_g := \{(x,y) : g(x) \neq y\} \subset \mathbb{R}^d \times \{0,\dots,k-1\}.$$

Then, with $Z_i = (X_i, Y_i)$

$$R(g) = P(A_g), \quad R_n(g) = P_n(A_g).$$

If these equalities are not obvious, recall that

$$R(g) = \mathbf{P}(g(X) \neq Y) = \mathbf{P}((X_1,Y_1) \in A_g) = P(A_g),$$
$$R_n(g) = \frac{1}{n}\sum_{i=1}^n I_{\{g(X_i)\neq Y_i\}} = \frac{1}{n}\sum_{i=1}^n I_{\{(X_i,Y_i)\in A_g\}} = P_n(A_g).$$

Hence (2.7.3) and (2.7.4) are now

$$\mathbf{P}\Big(\sup_{g\in\mathcal{G}} |R_n(g) - R(g)| > \epsilon\Big) \leq 8\mathbb{S}_{\mathcal{A}_\mathcal{G}}(n)\exp[-\frac{n\epsilon^2}{32}] \tag{2.7.9}$$

$$E\Big(\sup_{g\in\mathcal{G}} |R_n(g) - R(g)|\Big) \leq 2\sqrt{\frac{\ln \mathbb{S}_{\mathcal{A}_\mathcal{G}}(n) + \ln 2}{n}}, \tag{2.7.10}$$

where

$$\mathcal{A}_\mathcal{G} = \{A_g : g \in \mathcal{G}\}.$$

When VC-dimension of class $\mathcal{A}_\mathcal{G}$, say V, is finite, then this can be used by estimating $\mathbb{S}_{\mathcal{A}_\mathcal{G}}(n)$ to obtain the bounds (2.7.7) and (2.7.8). The VC-dimension of class $\mathcal{A}_\mathcal{G}$ is sometimes called the **graph-dimension of $\mathcal{G}$**.

Risk bounds for symmetric (0-1) loss and binary classification. Note that for $k = 2$ (binary classification) then any classifier g is in the form of I_A so that the class of classifiers $\mathcal{G}$ is uniquely determined by the class of sets $\mathcal{A}$. Therefore, in this case the shattering coefficient and VC-dimension of $\mathcal{A}$ are also called the **shattering coefficient of $\mathcal{G}$** and **VC-dimension of $\mathcal{G}$**. Now for any binary classifier $g = I_A$, set A_g is then

$$A_g = \{A \times \{0\} \cup A^c \times \{1\}\}.$$

Again, recall that $A_g \in \mathbb{R}^d \times \{0,1\}$, but $A \subset \mathbb{R}^d$. It is rather easy to see ([1], Thm. 13.1) that the shattering coefficient of $\mathcal{A}_\mathcal{G}$ equals to that one of $\mathcal{A}$, so that for binary classification the graph dimension of $\mathcal{G}$ equals to VC dimension of $\mathcal{G}$ (equivalently, to VC dimension of $\mathcal{A}$). Hence, for binary classification and symmetric risk, the risk bounds (2.7.9) and (2.7.10) are

$$\mathbf{P}\Big(\sup_{g\in\mathcal{G}} |R_n(g) - R(g)| > \epsilon\Big) \leq 8\mathbb{S}_{\mathcal{A}}(n)\exp[-\frac{n\epsilon^2}{32}] \leq 8(n+1)^V\exp[-\frac{n\epsilon^2}{32}] \tag{2.7.11}$$

$$E\Big(\sup_{g\in\mathcal{G}} |R_n(g) - R(g)|\Big) \leq 2\sqrt{\frac{\ln \mathbb{S}_{\mathcal{A}}(n) + \ln 2}{n}} \leq 2\sqrt{\frac{V\ln(n+1) + \ln 2}{n}}, \tag{2.7.12}$$

where V is the VC-dimension of $\mathcal{G}$. Hence, if $V < \infty$, then $\delta(\epsilon, n, \mathcal{G})$ converges towards zero exponentially fast and sample size n grows.

Exercise: Prove the following inequalities: (2.7.13), (2.7.14) and (2.7.15):

$$ER(\hat{g}_n) - \inf_{g\in\mathcal{G}} R(g) \leq 4\sqrt{\frac{V\ln(n+1)+\ln 2}{n}} \tag{2.7.13}$$

$$\mathbf{P}\Big(R(\hat{g}_n) - \inf_{g\in\mathcal{G}} R(g) > \epsilon\Big) \leq 8(n+1)^V \exp[-\frac{n\epsilon^2}{128}], \tag{2.7.14}$$

and with probability $1-\delta$

$$R(g_n) \leq R_n(g_n) + 2\sqrt{\frac{8(V\ln(n+1) - \ln\delta + \ln 8)}{n}}. \tag{2.7.15}$$

where $\hat{g} \in \mathcal{G}$ is the empirical risk minimizer and $g_n \in \mathcal{G}$ is arbitrary classifier.

Exercise: Prove that for $V < \infty$,

$$R(\hat{g}_n) \to \inf_{g\in\mathcal{G}} R(g) \quad \text{a.s.}, \quad ER(\hat{g}_n) \to \inf_{g\in\mathcal{G}} R(g). \tag{2.7.16}$$

Remarks: 1. The inequalities (2.7.3) and (2.7.4) are not the sharpest possible. In the original paper of Vapnik and Cervonenkis, the better exponent $\frac{-n\epsilon^2}{8}$ was instead of $\frac{-n\epsilon^2}{32}$. For some better exponents see [1], 12.8 also [2]. In the book of Vapnik ([5], Thm 4.1) the VC inequality (2.7.3) is given in the form

$$\mathbf{P}\Big(\sup_{A\in\mathcal{A}} |P_n(A) - P(A)| > \epsilon\Big) \leq 4\mathbb{S}_{\mathcal{A}}(2n)\exp[-(\epsilon - \frac{1}{n})^2 n] \tag{2.7.17}$$

The sharper VC inequalities yield to the sharper bounds also in (2.7.14). For all these improvements, the bounds are in general form $A\exp[-c\epsilon^2 n]$, where A and c are some constants and this cannot in general be improved. An exception is the case when $\inf_{g\in\mathcal{G}} R(g) = 0$ and then the risk bound on (2.7.14) is $A\exp[-c\epsilon n]$ i.e. ϵ instead of ϵ^2 that for small ϵ in a substantial improvement (see [1], 12.7).
It turns out that with additional work, the ln-factor ban be removed from the inequality (2.7.13), i.e. it holds ([2], 1.4.6)

$$ER(\hat{g}_n) - \inf_{g\in\mathcal{G}} R(g) \leq c\sqrt{\frac{V}{n}}, \tag{2.7.18}$$

where c on konstant. For n big enough (2.7.18) is better than (2.7.13).

2. When L is symmetric but the number of classes is bigger then two, then the inequalities obviously hold with V being the graph-dimension of $\mathcal{G}$.

2.8 Complexity regularization

Recall that for symmetric loss the finite graph-dimension (VC-dimension for binary classification) V implies that the following convergences (2.7.16) hold:

$$R(\hat{g}_n) \to \inf_{g \in \mathcal{G}} R(g) \quad \text{a.s.}, \qquad ER(\hat{g}_n) \to \inf_{g \in \mathcal{G}} R(g).$$

If the approximation error of $\mathcal{G}$ equals to zero, i.e. $\mathcal{G}$ is such that

$$\inf_{g \in \mathcal{G}} R(g) = R^*, \tag{2.8.1}$$

then these two convergences above would be the (strong) consisteny of ERM rule. However, usually (that depends on $F(x, y)$), if (2.8.1) holds, then $\mathcal{G}$ is so big (complex) that $V = \infty$. Then the convergences above would not hold true any more. A way out is to consider the sequence of classes models $\mathcal{G}_n$ so that the complexity of $\mathcal{G}_n$ increases with n so that the approximation error goes to zero. On the other hand, the convergence cannot be too fast to control the estimation error and avoid overfitting. To find the suitable classes (models) $\mathcal{G}_n$ and a method for selecting the classifier from the set of models is the issue of **complexity regularization** .

Sieves. Perhaps the simplest complexity regularization method is the following. Let

$$\mathcal{G}_1 \subset \mathcal{G}_2 \subset \cdots$$

be such that the approximation error goes to zero as k grows:

$$\lim_k \inf_{g \in \mathcal{G}_k} R(g) = R^*. \tag{2.8.2}$$

Since the classes are increasing, their complexity (e.g. graph-dimension) increases when k grows. Let $n \mapsto k(n)$ be a function such that $k(n) \to \infty$ as $n \to \infty$. Typically k is increasing. For every n, we now choose a classifier g_n with help of data from $\mathcal{G}_{k(n)}$ (for instance, by ERM). Then

$$R(g_n) - R^* = \big(R(g_n) - \inf_{g \in \mathcal{G}_{k(n)}} R(g)\big) + \big(\inf_{g \in \mathcal{G}_{k(n)}} R(g) - R^*\big).$$

Since $k(n) \to \infty$ as $n \to \infty$, then the approximation error goes to zero. On the other hand, the complexity of $\mathcal{G}_{k(n)}$ increases. If this increase is not too large and the estimation error can be controlled, then the consistency can achieved. As an example of such result let us consider the following theorem ([1], Thm. 18.1). In this theorem, let L be symmetric and, for every n, $g_n = \hat{g}_{k(n)}$ is obtained by ERM, i.e.

$$g_n = \hat{g}_{k(n)} = \arg \inf_{g \in \mathcal{G}_{k(n)}} R_n(g). \tag{2.8.3}$$

Theorem 2.8.1 *Let $\mathcal{G}_k$ be such that for every distribution $F(x,y)$ (2.8.2) holds. Let $k(n)$be such that*

$$\lim_n k(n) = \infty, \quad \lim_n \frac{V_{k(n)} \ln n}{n} = 0, \tag{2.8.4}$$

where V_k is the graph-dimension of $\mathcal{G}_k$. Then the rule $\{g_n\}$ as in (2.8.3) is strongly universally consistent.

Exercise: Prove the theorem.

Remark. The theorem does not require that $\mathcal{G}_1 \subset \mathcal{G}_2 \subset \cdots$ (sieves), but in most cases in practice it is so.

Structural risk minimization (SRM). SRM is applying ERM simultaneously to several classes. Let again $\mathcal{G}_k$ be the classes with increasing complexity (usually sieves); let V_k measure the complexity of $\mathcal{G}_k$ (say graph-dimension). Then, for every k and n the positive **penalty term** $C(V_k, n)$ is defined. Typically $C(V,n)$ increases in V and decreases in n, for instance

$$C(V,n) = a\sqrt{\frac{V \ln n + b}{n}},$$

where a, b are some constants.

SRM: for every k and n, find $\hat{g}_{k,n}$ that minimizes empirical risk over $\mathcal{G}_k$:

$$\hat{g}_{k,n} = \arg \inf_{g \in \mathcal{G}_n} R_k(g).$$

Now choose from the sequence $\hat{g}_{1,n}, \hat{g}_{2,n}, \ldots$ the one that minimizes

$$R_n(\hat{g}_{k,n}) + C(V_k, n).$$

over $k \geq 1$. Formally,

$$g_n = \hat{g}_{k(n),n}, \quad \text{where} \quad k(n) = \arg \inf_k \Big(R_n(\hat{g}_{k,n}) + C(V_k, n) \Big). \tag{2.8.5}$$

The difference with the method described above is that $k(n)$ is chosen automatically by data. Note that when $C(V_k, n)$ is very small in comparison with $R_n(\hat{g}_{k,n})$, then minimizing $R_n(\hat{g}_{k,n}) + C(V_k, n)$ were almost the same as minimizing $R_n(\hat{g}_{k,n})$ so that then $k(n)$ were typically very large – overfitting. On the other hand, when $C(V_k, n)$ is very big in comparison with $R_n(\hat{g}_{k,n})$, then minimizing $R_n(\hat{g}_{k,n}) + C(V_k, n)$ were almost the same as minimizing $C(V_k, n)$, and since it increases with k, we would have that $k(n)$ were typically very small – underfitting. Hence, choosing correct penalty is crucial.

Exercise: Suppose $\mathcal{G}_1 \subset \mathcal{G}_2 \subset \cdots$. Prove that SRM classifier g_n as in (2.8.5) can be defined as

$$g_n = \arg \inf_{\cup_k \mathcal{G}_k} \big(R_n(g) + C(g, n) \big), \tag{2.8.6}$$

where $C(g,n)=C(V_k,n)$, if $g\in\mathcal{G}_k/\mathcal{G}_{k-1}$.

Let to the end of this paragraph, L be symmetric. Then, from (2.7.13) we obtain:

$$\begin{aligned} ER(\hat{g}_{k,n})-R^* &= \Big(ER(\hat{g}_{k,n})-\inf_{g\in\mathcal{G}_k}R(g)\Big)+\Big(\inf_{g\in\mathcal{G}_k}R(g)-R^*\Big) \\ &\leq 4\sqrt{\frac{V_k\ln(n+1)+\ln 2}{n}}+\Big(\inf_{g\in\mathcal{G}_k}R(g)-R^*\Big), \end{aligned} \tag{2.8.7}$$

where V_k is graph-dimension of $\mathcal{G}_k$. We aim to minimize $ER(\hat{g}_{k,n})$ over k. If we knew $\inf_{g\in\mathcal{G}_k}R(g)$, we would minimize the right side of (2.8.7). It turns out that SRM almost does the job. As an example, consider the following result ([2] Thm. 1.20 ja 1.6.3), where the penalty is as follows

$$C(V_k,n)=2\sqrt{\frac{V_k\ln(n+1)+\ln 2}{n}}+\sqrt{\frac{\ln k}{n}}. \tag{2.8.8}$$

Theorem 2.8.2 *Let, for any k, the graph-dimension V_k finite. Let g_n be obtained by SRM with penalty (2.8.8). Then*

$$ER(g_n)-R^*\leq \min_k\Big[\sqrt{\frac{V_k\ln(n+1)+\ln 2}{n}}+\Big(\inf_{g\in\mathcal{G}_k}R(g)-R^*\Big)+\sqrt{\frac{\ln k}{n}}\Big]+\sqrt{\frac{1}{2n}}. \tag{2.8.9}$$

Corollary 2.8.1 *Let $\mathcal{G}_k$ be such that for every distribution $F(x,y)$ (2.8.2) holds. Then $\{g_n\}$ is universally consistent.*

Exercise: Prove corollary.

The following theorem ([1], Thm. 18.2) claims that $\{g_n\}$ is also strongly universally consistent. The theorem is proven for binary classification, so V_k is VC-dimension of $\mathcal{G}_k$. The penalty is

$$C(V_k,n)=\sqrt{\frac{32V_k\ln n+1}{n}}. \tag{2.8.10}$$

Theorem 2.8.3 *Let $\mathcal{G}_k$ be such that for every distribution $F(x,y)$ (2.8.2) holds. Assume that VC dimensions V_k satisfy*

$$\sum_{k=1}^{\infty}e^{-V_k}<\infty. \tag{2.8.11}$$

Then $\{g_n\}$ based on SRM with penalty (2.8.10) is strongly universally consistent.

Note: when $\sup_k V_k<\infty$, then the assumption (2.8.11) holds. If $V_k\to\infty$, then without loss of generality we can assume that for every k $V_k<V_{k+1}$ (just choose a subsequence) and then (2.8.11) holds, because V_k is always an integer.

Finally note: suppose the distribution is such that the Bayes rule $g^* \in \cup_k \mathcal{G}_k$, i.e. for a k_o, $g^* \in \mathcal{G}_{k_o}$. The corresponding ERM-classifier $g_{k_o,n}$ satisfy (2.7.18)

$$ER(\hat{g}_{k_o,n}) - R^* \leq C(k_o)\sqrt{\frac{1}{n}},$$

where $C(k_o)$ is a constant depends on k_o. Hence, if we knew the right class $\mathcal{G}_{k_o}$, we would use ERM using this class, only. As a result, we would get that that our ERM classifier $g_n = \hat{g}_{k_o,n}$ has the property

$$ER(g_n) - R^* = O\Big(\sqrt{\frac{1}{n}}\Big).$$

It is also known that this is the best possible rate. On the other hand, from Theorem 2.8.2, it follows that in this case g_n obtained by SRM gives the rate:

$$ER(g_n) - R^* \leq \sqrt{\frac{V_{k_o}\ln(n+1) + \ln 2}{n}} + \sqrt{\frac{\ln k_o}{n}} + \sqrt{\frac{1}{2n}} \leq B(k_o)\sqrt{\frac{\ln(n+1) + \ln 2}{n}},$$

where $B(k_o)$, again, is a constant, depending on k_o. Hence, without knowing the correct class $\mathcal{G}_{k_o}$ (true model), SRM-principle gives us classifier g_n so that $ER(g_n) - R^*$ converges to zero with almost the same speed with some additional $\sqrt{\ln n}$ factor, i.e.

$$ER(g_n) - R^* = O\Big(\sqrt{\frac{\ln n}{n}}\Big).$$

This is not so by the sieves method described above, because the speed depends on the prescribed sequence $k(n)$.

General regularization. Let $\mathcal{G}$ be a (complex) class of functions (with infinite graph-dimension, for instance), but to every function we define individual penalty $C(g,n)$. The overall classifier g_n is now

$$g_n = \arg\inf_{g\in\mathcal{G}}\Big(R_n(g) + C(n,g)\Big). \tag{2.8.12}$$

Typically $C(g,n)$ depends on g via some complexity measurement $v(g)$ and $C(v,n)$ is increasing in v and decreasing in n. The measurement v depends on the class of functions and task, it can be its norm, description length, degree etc. Note that SRM (2.8.6)is a special case of (2.8.12), where $\mathcal{G} = \cup_k \mathcal{G}_k$ and $g \mapsto C(g,n)$ is constant on $\mathcal{G}_1, \mathcal{G}_k/\mathcal{G}_{k-1}$, $k = 2, 3, \ldots$.. When $g \in \mathcal{G}_k/\mathcal{G}_{k-1}$, then $v(g) = V_k$.

Chapter 3

Linear discrimination

3.1 Preliminaries

3.1.1 Hyperplane and the distance from it

Let $\mathbb{X}$ be a Hilbert space. A **hyperplane** is the set

$$H := \{x : \langle w, x\rangle + w_0 = 0\},$$

where $w \in \mathbb{X}$. The distance of a point $y \in \mathbb{X}$ from H is

$$d(y, H) = \frac{1}{\|w\|}\big|\langle w, y\rangle + w_0\big|.$$

Without loss of generality w can be such that $\|w\| = 1$. Then defining

$$h(x) := \langle w, x\rangle + w_0,$$

we get that the distance of y from the hyperplane $\{x \in \mathbb{X} : h(x) = 0\}$ equals to $|h(y)|$.

If $\mathbb{X} = \mathbb{R}^d$, then $\langle w, x\rangle = w'x$. Hence the distance of y from

$$H := \{x \in \mathbb{X} : w'x + w_0 = 0\}$$

is

$$d(y, H) = \frac{|w'y + w_0|}{\|w\|} = \frac{|h(y)|}{\|w\|}, \text{ where } h(x) = w'x + w_0.$$

3.1.2 Decomposition of covariance matrix

Let X, Y random variables. Recall that

$$\text{Var}(X) = E(\text{Var}[X|Y]) + \text{Var}(E[X|Y]).$$

Let X d-dimensional random vector, $m = EX \in \mathbb{R}^d$. The covariance matrix of X is

$$\text{Cov}(X) = E(X-m)(X-m)' = E(XX') - mm'. \tag{3.1.1}$$

Proposition 3.1.1 *Let Y be a random variable. Then*

$$\text{Cov}(X) = E\big(E[(X - E(X|Y))(X - E(X|Y))'|Y]\big) + E\big((E(X|Y) - m)(E(X|Y) - m)'\big)$$

or, equivalently,

$$\text{Cov}(X) = E\big(\text{Cov}[X|Y]\big) + \text{Cov}\big(E[X|Y]\big). \tag{3.1.2}$$

Proof. To see this, note that

$$E(X-m)(X-m)' = E\big(E[(X-m)(X-m)'|Y]\big) =$$
$$E\Big(E\big[\big(X - E(X|Y) + E(X|Y) - m\big)\big(X - E(X|Y) + E(X|Y) - m\big)'|Y\big]\Big).$$

Conditional expectation is linear, i.e.

$$E[(X - E(Y|X))(E(X|Y) - m)'|Y] = E[(E(X|Y) - m)(X - E(X|Y))'|Y] = 0.$$

Hence

$$E\big[\big(X - E(X|Y) + E(X|Y) - m\big)\big(X - E(X|Y) + E(X|Y) - m\big)'|Y\big] =$$
$$E\big[\big(X - E(X|Y)\big)\big(X - E(X|Y)\big)'|Y\big] + E[(E(X|Y) - m)(E(X|Y) - m)'|Y].$$

■

Let now Y be $\{0, 1, \ldots, k-1\}$ - valued. Denote

$$m_i := E[X|Y = i], \quad \pi_i = \mathbf{P}(Y = i), \quad \Sigma_i := \text{Cov}[X|Y = i], \quad \Sigma_X := \text{Cov(X)}.$$

Then (3.1.2) is

$$\Sigma_X = \sum_{i=0}^{k-1} \pi_i \Sigma_i + \sum_{i=0}^{k-1} \pi_i (m_i - m)(m_i - m)' \tag{3.1.3}$$

Denoting

$$\Sigma_W := \sum_{i=0}^{k-1} \pi_i \Sigma_i, \quad \Sigma_B := \sum_{i=0}^{k-1} \pi_i (m_i - m)(m_i - m)'$$

and then (3.1.3) is

$$\Sigma_X = \Sigma_W + \Sigma_B.$$

The matrices Σ_W and Σ_B are interpreted as **within-classes** and **between-classes** covariances, respectively.

Exercise: Prove that for $k = 2$,

$$\Sigma_B = \pi_1 \pi_0 (m_1 - m_0)(m_1 - m_0)' \tag{3.1.4}$$

Sample covariance matrix Given sample $(x_1, y_1), \ldots, (x_n, y_n)$, the quantities π_i, Σ_X, Σ_i, m_i are estimated as follows:

$$\hat{\Sigma}_X := \frac{1}{n}\sum_{j=1}^{n}(x_j - \hat{m})(x_j - \hat{m})', \quad \hat{m} := \frac{1}{n}\sum_{j=1}^{n} x_j,$$
$$\hat{\Sigma}_i := \frac{1}{n_i}\sum_{j:y_j=i}(x_j - \hat{m}_i)(x_j - \hat{m}_i)', \quad \hat{\pi}_i := \frac{n_i}{n}, \quad \hat{m}_i = \frac{1}{n_i}\sum_{j:y_j=i} x_j, \quad i = 0, \ldots, k-1$$

Decomposition (3.1.2) is now

$$\begin{aligned}\hat{\Sigma}_X &= \sum_{i=0}^{k-1}\hat{\pi}_i\hat{\Sigma}_i + \sum_{i=0}^{k-1}\hat{\pi}_i(\hat{m}_i - \hat{m})(\hat{m}_i - \hat{m})' \\ &= \frac{1}{n}\sum_{i=0}^{k-1}\sum_{j:y_j=i}(x_j - \hat{m}_i)(x_j - \hat{m}_i)' + \frac{1}{n}\sum_{i=0}^{k-1} n_i(\hat{m}_i - \hat{m})(\hat{m}_i - \hat{m})'.\end{aligned}$$

Multiplying both sides by n, we get

$$\begin{aligned}S := \sum_{j=1}^{n}(x_j - \hat{m})(x_j - \hat{m})' &= \sum_{i=0}^{k-1}\sum_{j:y_j=i}(x_j - \hat{m}_i)(x_j - \hat{m}_i)' + \sum_{i=0}^{k-1} n_i(\hat{m}_i - \hat{m})(\hat{m}_i - \hat{m})' \\ &= \sum_{i=0}^{k-1} S_i + \sum_{i=0}^{k-1} n_i(\hat{m}_i - m)(\hat{m}_i - m)' = S_W + S_B,\end{aligned}$$

where

$$S_i := \sum_{j:y_j=i}(x_j - \hat{m}_i)(x_j - \hat{m}_i)', \quad S_W := \sum_{i=0}^{k-1} S_i, \quad S_B := \sum_{i=0}^{k-1} n_i(\hat{m}_i - \hat{m})(\hat{m}_i - \hat{m})'.$$

are within-classes and between-classes **scatter matrixes**. For two classes

$$S_B = \frac{n_1 n_0}{n}(\hat{m}_1 - \hat{m}_0)(\hat{m}_1 - \hat{m}_0)'. \tag{3.1.5}$$

3.2 Linear discriminant

Assumption: In this chapter, when not explicitly specified, we consider binary classification with symmetric loss. Hence, we have training data $(x_1, y_1), \ldots, (x_n, y_n)$, $x_i \in \mathbb{R}^d$, $y_i \in \{0, 1\}$.

Recall that in this case the Bayes classifier and Bayes risk are

$$g^*(x) = \begin{cases} 1 & \text{if } p(1|x) \geq 0.5, \\ 0 & \text{if } p(1|x) < 0.5. \end{cases} \tag{3.2.1}$$

$$R^* = R(g^*) = \mathbf{P}(g^*(X) \neq Y) = \inf_{g:\mathbb{R}^d \to \{0,1\}} \mathbf{P}(g(X) \neq Y) = E\big(\min\{p(1|X), 1 - p(1|X)\}\big),$$

where $p(1|x) = \mathbf{P}(Y = 1|X = x)$.

Linear discriminant (linear classifier) uses a hyperplane in $\mathbb{R}^d$ as a decision boundary. Formally:

$$g(x) = \begin{cases} 1 & \text{if } w'x + w_0 = \sum_{i=1}^d w^i x^i + w_0 \geq 0, \\ 0 & \text{otherwise}, \end{cases} \tag{3.2.2}$$

where $x' = (x^1, \ldots, x^d)$ and $w' = (w^1, \ldots, w^d)$ are in $\mathbb{R}^d$ and $w_0 \in \mathbb{R}$. Without loss of generality, w can be taken with unit length, $\|w\| = 1$. The decision boundary is then a hyperplane

$$H = \{x \in \mathbb{R}^d : w'x + w_0 = 0\}.$$

Remarks: 1. In pattern recognition theory, the classes are often coded as $+1$ and -1 (obtained from $\{0, 1\}$ coding via transformation $2Y - 1$) and is this case linear discriminant is

$$g(x) = \operatorname{sgn}(w'x + w_0).$$

2. In statistical learning (especially in connections with neural networks), the linear discriminant is often called as **perceptron**.

3.3 Risk bound via covariation matrixes

3.3.1 One-dimensional space

In one-dimensional space, a hyperplane is a point, also called **a split**. Hence, when $d = 1$, the linear discriminant is

$$g(x) = \begin{cases} y', & \text{kui } x \leq x'; \\ 1 - y', & \text{kui } x > x'. \end{cases}, \tag{3.3.1}$$

where $x' \in \mathbb{R}$ and $y' \in \{0, 1\}$.

Let $\mathcal{G}$ be the set of all linear discriminants and let us investigate the risk of the best linear discriminant

$$R := \inf_{\mathcal{G}} R(g).$$

The smaller is R, the better the classes are linearly separable.

Exercise: Prove that in one-dimensional case

$$R = \inf_{x',y'} \Big[I_{\{y'=0\}} \big(\pi_1 F_1(x') + \pi_0(1 - F_0(x'))\big) + I_{\{y'=1\}} \big(\pi_0 F_0(x') + \pi_1(1 - F_1(x'))\big) \Big] \leq \pi_1 \wedge \pi_0 \leq \frac{1}{2}, \tag{3.3.2}$$

where

$$F_i(x) = \mathbf{P}(X \leq x | Y = i), \quad \pi_i = \mathbf{P}(Y = i), \quad i = 0, 1.$$

From (3.3.2) it follows that in general $R^* \leq R \leq \frac{1}{2}$. Moreover, it can be shown ([1], Lemma 4.1) that $R = \frac{1}{2}$ iff $R^* = \frac{1}{2}$.

The class-conditional distributions F_0 and F_1 are, in general, unknown. The next lemma bounds R via class-conditional means and covariances that are easy to estimate:

$$m_i = E(X|Y = i), \quad \sigma_i^2 = \text{Var}(X|Y = i), \quad i = 1, 0.$$

Recall **Chebysev-Cantelli's inequality**: for and random variable X with finite variance,

$$\mathbf{P}(X - EX \geq c) \leq \frac{\text{Var}X}{c^2 + \text{Var}X}.$$

Lemma 3.3.1

$$R \leq \Big(1 + \frac{(m_0 - m_1)^2}{(\sigma_0 + \sigma_1)^2}\Big)^{-1}. \tag{3.3.3}$$

Proof. Without loss of generality, let $m_0 < m_1$. Let $\Delta_1 > 0$, $\Delta_0 > 0$ such that

$$m_1 - m_0 = \Delta_1 + \Delta_0.$$

Consider the discriminant

$$g(x) = \begin{cases} 0, & \text{if } x \le m_0 + \Delta_0 = m_1 - \Delta_1; \\ 1, & \text{if } x > m_0 + \Delta_0 = m_1 - \Delta_1. \end{cases}$$

Obviously

$$R \le R(g) = \mathbf{P}(Y=1)\mathbf{P}(X \le m_1 - \Delta_1 | Y=1) + \mathbf{P}(Y=0)\mathbf{P}(X > m_0 + \Delta_0 | Y=0).$$

From Chebysev-Cantelli

$$\mathbf{P}(X > m_0 + \Delta_0 | Y=0) \le \frac{\sigma_0^2}{\sigma_0^2 + \Delta_0^2}, \quad \mathbf{P}(X \le m_1 - \Delta_1 | Y=1) \le \frac{\sigma_1^2}{\sigma_1^2 + \Delta_1^2}.$$

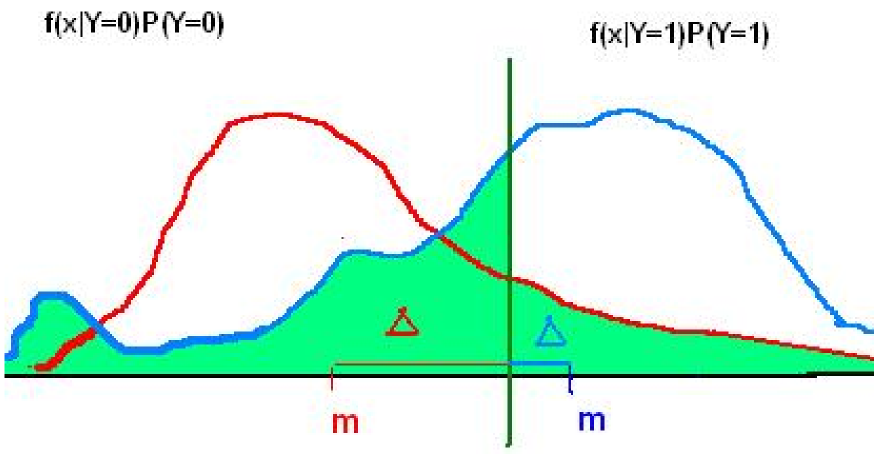
Therefore

$$R \le \frac{\pi_1 \sigma_1^2}{\sigma_1^2 + \Delta_1^2} + \frac{(1-\pi_1)\sigma_0^2}{\sigma_0^2 + \Delta_0^2},$$

where $\pi_1 = \mathbf{P}(Y=1)$. Take now

$$\Delta_0 = \frac{(m_1 - m_0)\sigma_0}{\sigma_0 + \sigma_1}, \quad \Delta_1 = \frac{\sigma_1}{\sigma_0}\Delta_0,$$

and use the last inequality to get (3.3.3). ■



Hence, the bigger is

$$\frac{|m_1 - m_0|}{\sigma_0 + \sigma_1},$$

the smaller is R (and also R^*) and the best linear discriminant performs well.

Exercise: Let $X \sim U[0,1]$ and

$$Y = \begin{cases} 0, & \text{if } X \in [\frac{1}{3}, \frac{2}{3}]; \\ 1, & \text{else.} \end{cases}$$

Show that $R^* = 0$, $R = \frac{1}{3}$. Find the bound (3.3.3) and best linear classifier.

3.3.2 Space $\mathbb{R}^d$

In $\mathbb{R}^d$, every linear discriminant

$$g(x) = \begin{cases} 1 & \text{if } w'x + w_0 \geq 0, \\ 0 & \text{otherwise,} \end{cases}$$

is uniquely determined by the d-dimensional vector w and scalar w_0. Without loss of generality, w can always be taken with such that $\|w\| = 1$. Again, let $R := \inf_{\mathcal{G}} R(g)$, where $\mathcal{G}$ is the set of all linear discriminants. Recall

$$m_i = E[X|Y = i], \quad \Sigma_i = E[(X - m_i)(X - m_i)'|Y = i]$$

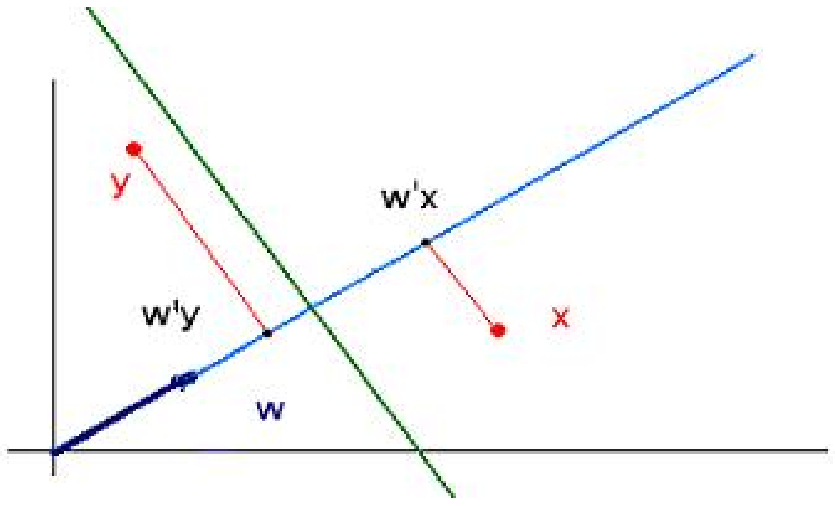
Theorem 3.3.1

$$R^* \leq R \leq \inf_{w \in \mathbb{R}^d} \Big(1 + \frac{\big(w'(m_1 - m_0)\big)^2}{\big((w'\Sigma_1 w)^{\frac{1}{2}} + (w'\Sigma_0 w)^{\frac{1}{2}}\big)^2}\Big)^{-1}. \tag{3.3.4}$$

Proof. Fix a $w \in \mathbb{R}^d$ of unit length and $w_0 \in \mathbb{R}$. Let g be the corresponding discriminant. The classifier g projects every x onto one-dimensional subspace spanned by w and compares the projection with w_0. Hence, one dimensional discriminant applied to the projection $w'x$. The class-conditional means and variances of $w'X$ are $E[w'X|Y = i] = w'm_i$ and $\text{Var}[(w'X)|Y = i] = w'\Sigma_i w$. From Lemma 3.3.1, thus

$$R(g) \leq \Big(1 + \frac{\big(w'(m_1 - m_0)\big)^2}{\big((w'\Sigma_1 w)^{\frac{1}{2}} + (w'\Sigma_0 w)^{\frac{1}{2}}\big)^2}\Big)^{-1}.$$

■



Functions J and J'. The inequality (3.3.4) says: R is small if $\exists\ w \in \mathbb{R}^d$ such that $J(w)$ is big, where

$$J(w) := \frac{w'(m_1 - m_0)}{(w'\Sigma_0 w)^{\frac{1}{2}} + (w'\Sigma_1 w)^{\frac{1}{2}}} \tag{3.3.5}$$

In other words: R is small if $\exists\ w \in \mathbb{R}^d$ such the projected feature feature vector $w'X$ has class conditional means far from each other and the class-conditional variances small.

In practice, instead of J, the related quantity J' is often considered

$$\begin{aligned} J'(w) &:= \frac{(w'(m_1 - m_0))^2}{\pi_0 w'\Sigma_0 w + \pi_1 w'\Sigma_1 w} = \frac{(w'(m_1 - m_0))^2}{w'(\pi_0\Sigma_0 + \pi_1\Sigma_1)w} \\ &= \frac{w'(m_1 - m_0)(m_1 - m_0)'w}{w'\Sigma_W w} = (\frac{1}{\pi_1\pi_0})\frac{w'\Sigma_B w}{w'\Sigma_W w}. \end{aligned}$$

To obtain the last equation, use (3.1.4): $\Sigma_B = \pi_1\pi_0(m_1 - m_0)(m_1 - m_0)'$.

Note that when $\Sigma_0 = \Sigma_1 =: \Sigma$, then $\Sigma_W = \Sigma$ and $4J^2(w) = J'(w)$.

One reason for considering J' is that maximizing J' is easier then maximizing J. Indeed maximizing J' is equivalent to the following problem

$$\max_{w:\|w\|=1} \frac{w'\Sigma_B w}{w'\Sigma_W w}. \tag{3.3.6}$$

and the solution of (3.3.6) is

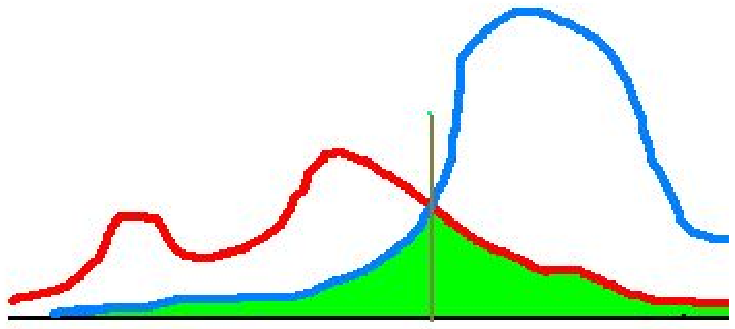
$$w \propto \Sigma_W^{-1}(m_1 - m_0). \tag{3.3.7}$$

When $\Sigma_0 = \Sigma_1 =: \Sigma$, then (3.3.7) (the vector that in this special case maximizes J' as well as J) is

$$w \propto \Sigma^{-1}(m_1 - m_0) =: \underline{w}.$$

3.4 When Bayes classifier is a linear discriminant?

When $d = 1$, then linear discriminant is the separation by one point and this happens



For $d > 1$, a linear Bayes discriminant is rather an exception than a rule. An important special case is when class-conditional densities are multivariate normal. Recall that with f_0 and f_1 being the class conditional densities and $\pi := \pi_1$, the Bayes rule is

$$g^*(x) = \begin{cases} 1 & \text{if } \pi_1 f_1(x) > (1-\pi_1) f_0(x), \\ 0 & \text{else.} \end{cases} \tag{3.4.1}$$

When

$$f_i(x) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_i|}} \exp[-\frac{1}{2}(x-m_i)'\Sigma_i^{-1}(x-m_i)],$$

we obtain that $g^*(x) = 1$ iff

$$(x-m_1)'\Sigma_1^{-1}(x-m_1) - 2\ln \pi_1 + \ln|\Sigma_1| < (x-m_0)'\Sigma_0^{-1}(x-m_0) - 2\ln(1-\pi_1) + \ln|\Sigma_0|. \tag{3.4.2}$$

A special case, when $\Sigma_1 = \Sigma_0 = \Sigma$, the inequality(3.4.2) is

$$(x-m_1)'\Sigma^{-1}(x-m_1) < (x-m_0)'\Sigma^{-1}(x-m_0) - 2\ln\frac{\pi_1}{1-\pi_1}. \tag{3.4.3}$$

Equivalently

$$-2x'\Sigma^{-1}(m_1 - m_0) < 2\ln\frac{\pi_1}{1-\pi_1} + m_0'\Sigma^{-1}m_0 - m_1'\Sigma^{-1}m_1$$

so that the Byes classifier is a linear discriminant:

$$g^*(x) = \begin{cases} 1, & \text{when } \underline{w}'x + w_0 > 0; \\ 0, & \text{else,} \end{cases}$$

where

$$\underline{w} := \Sigma^{-1}(m_1 - m_0) \tag{3.4.4}$$

and

$$\begin{aligned} w_0 &= \ln\frac{\pi_1}{1-\pi_1} + \frac{1}{2}\Big(m_0'\Sigma^{-1}m_0 - m_1'\Sigma^{-1}m_1\Big) = \ln\frac{\pi_1}{1-\pi_1} - \frac{(m_0+m_1)'}{2}\Sigma^{-1}(m_1-m_0) \\ &= \ln\frac{\pi_1}{1-\pi_1} - \frac{(m_0+m_1)'}{2}\underline{w}. \end{aligned}$$

Equivalently $g^*(x) = 1$ iff

$$\underline{w}'x > \frac{(m_0+m_1)'}{2}\underline{w} - \ln\frac{\pi_1}{1-\pi_1} = \frac{\underline{w}'m_0 + \underline{w}'m_1}{2} - \ln\frac{\pi}{1-\pi}$$

or, since

$$\underline{w}'(m_1 - m_0) = (m_1-m_0)'\Sigma^{-1}(m_1-m_0) \geq 0,$$

$$|\underline{w}'x - \underline{w}'m_1| < |\underline{w}'x - \underline{w}'m_0| - 2\ln\frac{\pi_1}{1-\pi_1}. \tag{3.4.5}$$

In a very special case, when $\pi = \frac{1}{2}$ (and $\Sigma_1 = \Sigma_0 = \Sigma$), we have

$$\frac{(m_0 + m_1)}{2} = EX =: m$$

and the Bayes classifier in that case is

$$g^*(x) = 1 \Leftrightarrow \underline{w}'x > \underline{w}'m.$$

When the feature vector X is normally distributed in both classes and the covariation matrixes are equal, the best classifier is a linear discriminant.

3.5 Risk bounds and ERM principle for linear classifers

Recall that in this chapter, $\mathcal{G}$ stands for the set of linear classifiers $\mathcal{G}$ and R is the risk of best linear classifier

$$R = \inf_{g \in \mathcal{G}} R(g).$$

Given a data-based classifier $g_n \in \mathcal{G}$, we are interested in its risk $R(g_n)$. In Section 2.7, the PAC-type of inequalities, based on VC-dimension of $\mathcal{G}$ were for estimating $R(g_n)$ were introduced. The set of linear classifiers $\mathcal{G}$ is not complex, its VC-dimension is $d+1$. Hence, the inequality (2.7.11) in that case is

$$\mathbf{P}\Big(\sup_{g \in \mathcal{G}} |R_n(g) - R(g)| > \epsilon\Big) \leq 8(n+1)^{d+1} \exp[-\frac{n\epsilon^2}{32}] \tag{3.5.1}$$

and the corresponding PAC-inequality (2.7.15) in this case is: with probability $1-\delta$

$$R(g_n) \leq R_n(g_n) + 2\sqrt{\frac{8((d+1)\ln(n+1) - \ln\delta + \ln 8)}{n}} = R_n(g_n) + 8\sqrt{\frac{(d+1)\ln(n+1) + \ln\frac{8}{\delta}}{2n}}. \tag{3.5.2}$$

Let us recall once again, that (3.5.2) holds for any method.

ERM-principle. For symmetric loss function, ERM-principle is minimizing the training errors. We used to denote the classifier obtained by ERM-principle as $\hat{g}_n$. Hence

$$\hat{g}_n = \arg\min_{g \in \mathcal{G}} R_n(g) = \arg\min_{g \in \mathcal{G}} \sum_{i=1}^{n} I_{\{y_i \neq g(x_i)\}}.$$

The PAC bound (3.5.2) holds obviously for $\hat{g}_n$ as well, but from (2.7.13) and (2.7.14), we get the additional bounds

$$ER(\hat{g}_n) - R \leq 4\sqrt{\frac{(d+1)\ln(n+1) + \ln 2}{n}}$$

$$\mathbf{P}\Big(R(\hat{g}_n) - R > \epsilon\Big) \leq 8(n+1)^{(d+1)} \exp[-\frac{n\epsilon^2}{128}],$$

implying that (how?)

$$R(\hat{g}_n) \to R, \quad \text{a.s.}, \quad R_n(\hat{g}_n) \to R.$$

Hence $\hat{g}_n$ is a classifier with good theoretical properties. If $F(x,y)$ is such that Bayes classifier is linear, then ERM is a consistent rule.

The problem: $\hat{g}_n$ is hard to find. The gradients of empirical risk function $R_n(g)$ are almost everywhere equal to zero, so gradient-methods will not work. Moreover, it can be shown that finding $\hat{g}_n$ is NP-hard. Therefore, several alternatives are used. Note that the empirical equals to

$$\frac{1}{n}\sum_{i=1}^{n}\big|y_i - I_{(0,\infty)}(w^T x_i + w_o)\big|^p, \tag{3.5.3}$$

where $p \geq 1$. A possibility to smooth the function is to replace the indicator $I_{(0,\infty)}$ with some similar smooth function σ, usually called as the **sigmoid**. Hence the objective is the following differentiable function:

$$\frac{1}{n}\sum_{i=1}^{n}\big|y_i - \sigma(w^T x_i + w_o)\big|^p. \tag{3.5.4}$$

Often the logistic function

$$\sigma(t) = \frac{\exp[t]}{1+\exp[t]}$$

is used. Then (3.5.3) is

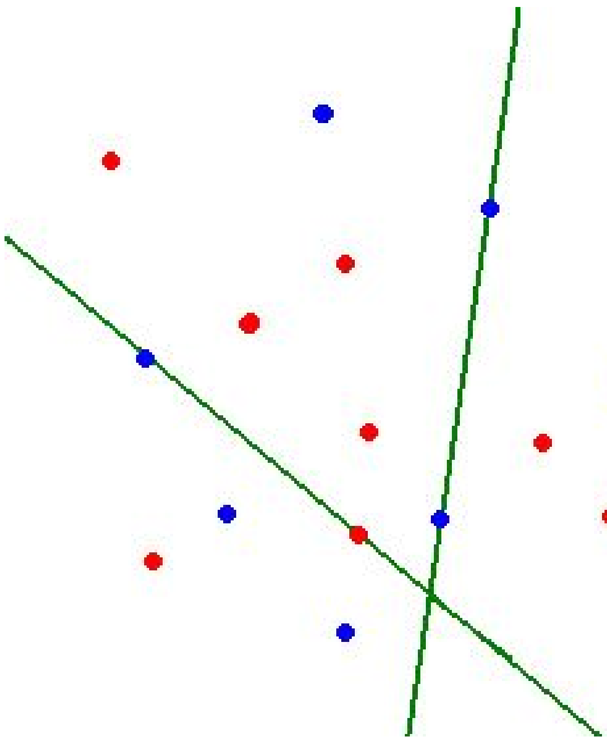
$$\frac{1}{n}\sum_{i=1}^{n}|y_i - \frac{\exp[w^T x_i + w_o]}{1+\exp[w^T x_i + w_o]}|^p.$$

If $p = 2$, we get the logistic regression problem. About logistic discrimination read Section 3.6.2.

Fingering. A way to restrict the search space is to consider only those hyperplanes that go through d sample points. Every such a hyperplane defines two classifiers. If X is absolutely continuous, then in every hyperplane, there are at most d sample points, hence in this case $\binom{n}{d}$ hyperplanes (much less than the set of all hyperplanes) should be considered, amongst them the one with smaller risk is chosen. This method is called **fingering**. The classifier g_n obtained in this way, does not minimize empirical risk, since the points on the hyperplane are not distinguished. However, since (for continuous X) in every hyperplane, there are d points, clearly

$$R_n(g_n) - R_n(\hat{g}_n) \leq \frac{d}{n},$$

so that for big n the difference is small and all the convergences above hold as well.



3.6 Training linear classifier: some classical methods

Despite the fact that Bayes classifier is usually not linear, due to its simplicity, they are still often used. We know that a good way to train a linear classifier is ERM-principle, but this is hard to apply. In the following, we shall consider some most popular methods of choosing a linear classifier from the set $\mathcal{G}$ of all linear classifiers using data.

3.6.1 Linear regression

The idea: approximate $x \mapsto p(1|x)$ with some linear function $f(x) := w'x + a$, where $w \in \mathbb{R}^d$, $a \in \mathbb{R}$. Hence, from (3.2.1), we get the classifier

$$g(x) = \begin{cases} 1 & \text{when } w'x + a \geq 0.5, \\ 0 & \text{when } w'x + a < 0.5. \end{cases} \tag{3.6.1}$$

Taking $w_o := a - 0.5$, we get that (3.6.1) is linear classifier in the form (3.3.2):

$$g(x) = 1 \quad \Leftrightarrow \quad w'x + w_o > 0, \quad \text{elsewhere } g(x) = 0.$$

Ordinary least squares: Find $\hat{w}$ and $\hat{a}$ by solving

$$\min_{w,a} \frac{1}{n} \sum_{i=1}^{n} (y_i - w'x_i - a)^2. \tag{3.6.2}$$

With $\hat{w}$ and $\hat{a}$ construct the classifier g_n as in (3.6.1).

Some properties. The OLS-problem is the empirical (data-based) version of the following optimization problem:

$$\min_{w,a} E\big(Y - (w'X + a)\big)^2. \tag{3.6.3}$$

Let us briefly study the properties of the corresponding classifier g [as in (3.6.1), where w and a are the solutions of (3.6.3)]. If g has good properties, then it is natural to hope that then the corresponding empirical classifier g_n has good properties as well, at least when n is sufficiently big.
Note that the following propositions hold for arbitrary $F(x, y)$, hence they also hold for empirical measure $F_n(x, y)$, thus for sum (3.6.1).

Proposition 3.6.1 *Let $f : \mathbb{R}^d \to \mathbb{R}$ be an arbitrary function. Then*

$$E\big(Y - f(X)\big)^2 = E\big(Y - p(1|X)\big)^2 + E\big(p(1|X) - f(X)\big)^2.$$

Exercise: Prove Proposition 3.6.1.

Hence, for any class of functions $\mathcal{F}$, minimizing $E\big(Y - f(X)\big)^2$ is the same as minimizing $E\big(p(1|X) - f(X)\big)^2$. In particular, the following minimization problems are equivalent:

$$\min_{w,a} E\big(p(1|X) - (w'X + a)\big)^2 \quad \Leftrightarrow \quad \min_{w,a} E\big(Y - (w'X + a)\big)^2 \tag{3.6.4}$$

and the minimizer $f(x) = w'x + a$ is then, in the sense of the least squares, the best approximation of $p(1|x)$.

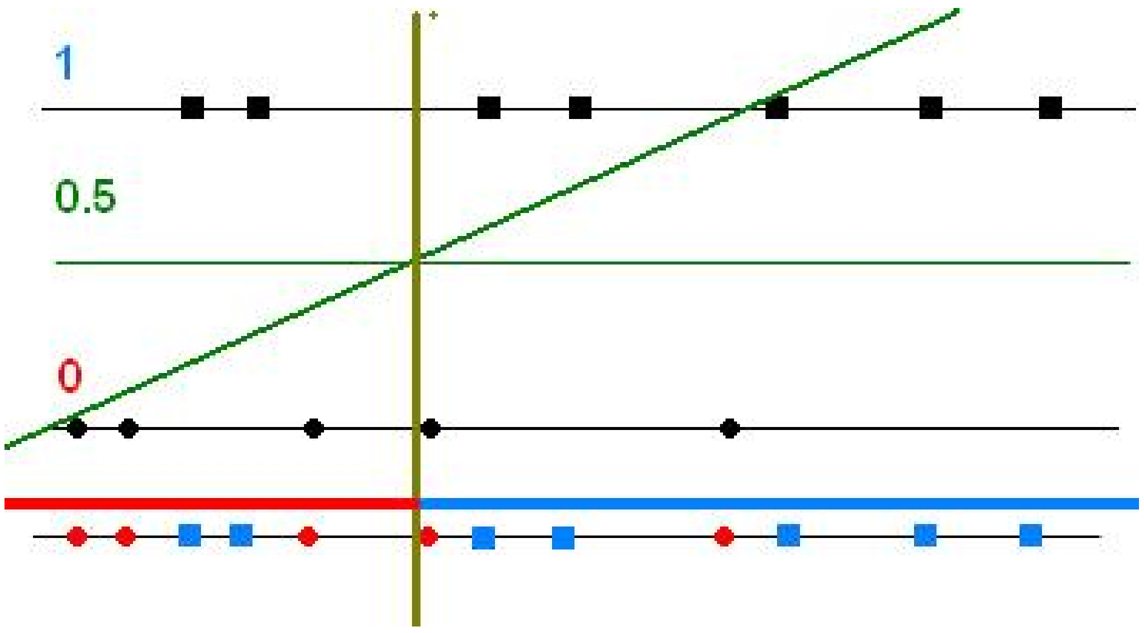
Proposition 3.6.2 *Let $\gamma, \beta \in \mathbb{R}$. Let w^* and a^* be the solutions of (3.6.3). Then γw^* and $\gamma a^* + \beta$ are the solutions of the following optimization problem*

$$\min_{w,a} \big((\gamma Y + \beta) - (w'X + a)\big)^2 = \min_{w,a} \big(\tilde{Y} - (w'X + a)\big)^2, \tag{3.6.5}$$

where $\tilde{Y} = \gamma Y + \beta$

Exercise: Prove Proposition 3.6.2.

From Proposition 3.6.2, it follows that the classifier g remains unchanged when decoding the labels by changing $0 \leftrightarrow 1$, i.e. $\gamma = -1$ and $\beta = 1$. Indeed, let $\tilde{w} := \gamma w$ and $\tilde{a} := \gamma a + \beta$ be the solutions of (3.6.5) and let $\tilde{f} := \tilde{w}'x + \tilde{a}$ be the corresponding regression function. Now, with $\gamma = -1$ and $\beta = 1$, $\tilde{f}(x) = 1 - f(x)$, where $f(x) = w'x + a$ is the original regression function and, therefore $f(x) > 0.5$ iff $\tilde{f}(x) < 0.5$.



Hence, the classifier does not change (unless, for those x that satisfy $f(x) = 0.5$) when in the picture above, the red dots are coded as one and blue dots as zero.

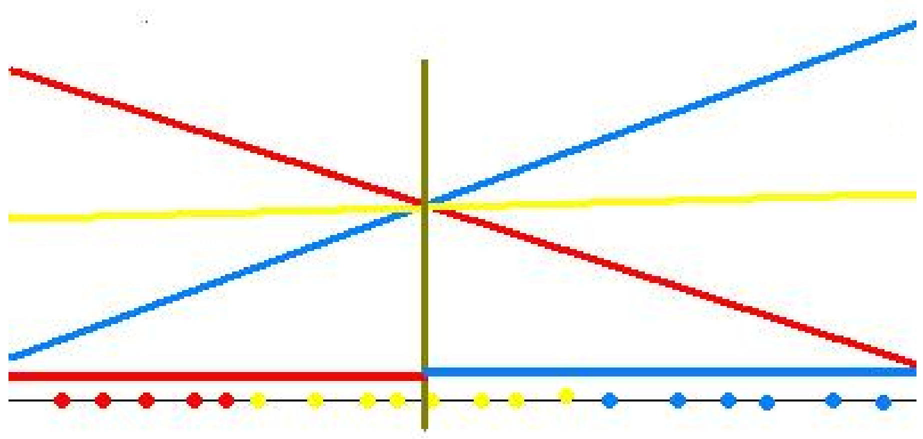
Many classes. Recall that by changing the labels, the regression functions $f(x)$ and $\tilde{f}(x)$ are such that $\tilde{f}(x) + f(x) = 1$, Thus,

$$f(x) > 0.5 \Leftrightarrow f(x) > \tilde{f}(x).$$

This is in full correspondence with Bayesian decision theory: $f(x)$ approximates $p(1|x)$ and $\tilde{f}(x)$ approximates $p(0|x)$ (after decoding), and to be consistent with Bayesian decision theory, we have to classify x into the class with bigger probability.
The genralization for more than two classes is now obvious: for any class $j = 0, \ldots, k-1$, relabel the data so that $\tilde{y}_i = 1$ iff $y_i = j$, otherwise $y_i = 0$, find the corresponding regression function $f_j(x)$ by solving (3.6.1), then define the classifier as follows

$$g_n(x) = \arg \max_{i=0,\ldots,k-1} f_i(x). \tag{3.6.6}$$

A problem for more than two classes might be *"masking"*.



Minimizing (3.6.3). Recall (3.6.3):

$$\min_{w,a} E\big(Y - (w'X + a)\big)^2,$$

Taking the partial derivatives, setting them to zero, we get the system of equation for finding the solutions of (3.6.3):

$$\begin{aligned} E(XX')w + aEX &= E(XY) \\ w'EX + a &= EY. \end{aligned}$$

Thus

$$a = EY - w'EX$$

and

$$E(XX')w + EYEX - EX(EX)'w = E(XY). \tag{3.6.7}$$

since $E(XX') - EX(EX)' = \Sigma_X$, we get

$$\Sigma_X w = E(XY) - EYEX, \tag{3.6.8}$$

so that

$$w = \Sigma_X^{-1}\Big(E(XY) - EYEX\Big).$$

Exercise: Let there be k-classes. Define $Y_i := I_{\{i\}}(Y)$, i.e. $Y_i = 1$ iff $Y = i$. Let

$$(w_i, a_i) = \arg\min_{w,a} E\big(Y_i - (w'X + a)\big)^2.$$

Show that

$$\sum_{i=0}^{k-1} w_i = 0, \quad \sum_{i=0}^{k-1} a_i = 1. \tag{3.6.9}$$

(Show that without loss of generality, you can assume $EX = 0$).

<u>Two classes.</u> The solutions

$$w = \Sigma_X^{-1}\big(E(XY) - EYEX\big), \quad a = EY - w'EX$$

hold for any Y. For binary classification, these solutions can be written as follows. Recall that $m_i := E[X|Y=i]$, $i = 0, 1$ and

$$m := EX = \pi_1 m_1 + \pi_0 m_0.$$

Then

$$E(XY) - EYEX = \pi_1 m_1 - \pi_1 m = \pi_0\pi_1(m_1 - m_0)$$

so that

$$w = \pi_0\pi_1\Sigma_X^{-1}(m_1 - m_0). \tag{3.6.10}$$

Recall the decomposition

$$\Sigma_X = \Sigma_W + \pi_0\pi_1(m_1 - m_0)(m_1 - m_0)'.$$

Plugging this into (3.6.8), after some algebra, we get

$$w = \alpha\Sigma_W^{-1}(m_1 - m_0), \quad a = \pi_1 - w'm, \tag{3.6.11}$$

where

$$\alpha = \pi_0\pi_1\big(1 - (m_1 - m_0)'w\big) = \pi_0\pi_1\big(1 - \pi_0\pi_1(m_1 - m_0)'\Sigma^{-1}(m_1 - m_0)\big).$$

It can be shown that $\alpha > 0$. Hence, up to a positive constant, the solution w is the same as the one minimizing J'.

Relation with linear Bayes classifier. Suppose, for moment that class conditional distributions of X are multivariate normal with equal covariances: $\Sigma_1 = \Sigma_0 = \Sigma$. The Bayes classifier in that case is

$$g^*(x) = 1 \quad \Leftrightarrow \quad \underline{w}'x + w_0 \geq 0,$$

where

$$\underline{w} = \Sigma^{-1}(m_1 - m_0) \quad \text{and} \quad w_0 = \ln\frac{\pi_1}{\pi_0} - \frac{(m_1 + m_0)'}{2}\underline{w}.$$

Then $w = \alpha\underline{w}$, $\alpha > 0$, so that the classifier $g(x)$ in this case is

$$g(x) = 1 \quad \Leftrightarrow \quad \alpha\underline{w}'x + a - 0.5 \geq 0 \quad \Leftrightarrow \quad \underline{w}'x + \alpha^{-1}(\pi_1 - \alpha\underline{w}'m - 0.5) \geq 0. \tag{3.6.12}$$

When $\pi_1 = \pi_0 = 0.5$, both rules are the same, hence g is Bayes rule, when $\pi_1 \neq \pi_0$, then the vectors w are the same, but the constants are different.

How good is g? Recall that g is a linear classifier such that $g(x) = 1$ iff $w'x + a > 0.5$, when w and a are solutions of (3.6.3). We saw that in a very special case, g is the best possible – Bayes classifier, but in general it might not be even the best from all linear classifiers (even when class conditional distributions are multivariate normal and Bayes classifier is linear) so that in general $R(g) - R > 0$, where $R = \inf_{g\in\mathcal{G}} R(g)$ with $\mathcal{G}$ being the class of linear classifiers. How big can the difference $R(g) - R$ be? It depends on the distribution of $F(x, y)$ and can be very big, namely

$$\sup\bigl(R(g) - R\bigr) = 1, \tag{3.6.13}$$

where supremum is taken over all possible distributions $F(x, y)$. The proof is the following example.

Exercise: Let

$$\mathbf{P}((X,Y) = (-m,1)) = \mathbf{P}((X,Y) = (m,0)) = \epsilon,$$
$$\mathbf{P}((X,Y) = (1,1)) = \mathbf{P}((X,Y) = (-1,0)) = \frac{1}{2} - \epsilon, \quad m > 0.$$

Find R^* and R. Prove that

$$w = \frac{1 - 2\epsilon(1+m)}{4(m^2\epsilon + (\frac{1}{2} - \epsilon))}, \quad a = 0.5.$$

Show that for every ϵ there exists m such that $w < 0$. Prove (3.6.13).

The classifier g_n. Replacing the theoretical expectations with sample averages, we get the solutions of

$$\min_{w,a} \frac{1}{n}\sum_{i=1}^{n}(y_i - w'x_i - a)^2.$$

as follows:

$$\hat{w} = \hat{\pi}_0\hat{\pi}_1\hat{\Sigma}_X^{-1}(\hat{m}_1 - \hat{m}_0) = \hat{\alpha}\hat{\Sigma}_W^{-1}(\hat{m}_1 - \hat{m}_0) = (\hat{\alpha}n)S_W^{-1}(\hat{m}_1 - \hat{m}_0)$$
$$\hat{a} = \hat{\pi}_1 - \hat{w}'\hat{m},$$
$$\hat{\alpha} = \hat{\pi}_1\hat{\pi}_0\Bigl(1 - \frac{\hat{\pi}_1\hat{\pi}_0(\hat{m}_1 - \hat{m}_0)'\hat{\Sigma}_W^{-1}(\hat{m}_1 - \hat{m}_0)}{1 + \hat{\pi}_1\hat{\pi}_0(\hat{m}_1 - \hat{m}_0)'\hat{\Sigma}_W^{-1}(\hat{m}_1 - \hat{m}_0)}\Bigr).$$

Of course, with $x_i = (x_{i1}, \dots, x_{id})'$ $i = 1, \dots, n$ and

$$\mathbf{Z} = \begin{pmatrix} x_{11} & \cdots & x_{1d} & 1 \\ x_{21} & \cdots & x_{2d} & 1 \\ \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nd} & 1 \end{pmatrix}, \quad y = \begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{pmatrix}, \quad \beta = \begin{pmatrix} w_1 \\ \cdots \\ w_d \\ a \end{pmatrix}$$

we get the solution in usual form

$$\begin{pmatrix} \hat{w} \\ \hat{a} \end{pmatrix} = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'y.$$

When n grows, then

$$\hat{m}_i \to m_i \quad \text{a.s.}, \quad \hat{\Sigma}_W^{-1} \to \Sigma_W^{-1} \quad \text{a.s.}, \quad \hat{\pi}_{\mathrm{i}} \to \pi_{\mathrm{i}} \quad \text{a.s.}$$

so that

$$\hat{w} \to w, \quad \text{a.s.}, \quad \hat{a} \to a, \quad \text{a.s.}.$$

When $\mathbf{P}(w'X + a = 0.5) = 0$ (e.g. X is absolutely continuous), then these convergences imply (just dominated convergence)

$$R(g_n) \to R(g), \quad \text{a.s.}. \tag{3.6.14}$$

Hence, for many cases $R(g_n) \to R(g)$, but, as we saw, the limit $R(g)$ can be rather bad. However in a very special case, when X has multivariate normal distribution in both classes and $\pi_i = 0.5$, then (3.6.14) implies that g_n is consistent.

To summarize: Depending on $F(x, y)$, the classifier based on linear regression can behave very badly even for very big n.

References: About linear regression read [8] (sec 4.2.4 and 4.3.4), [9] (sec 4.6), [6] (sec 5.8), [7] (sec 4.2 and Ch 3).

3.6.2 Logistic discrimination (regression)

In logistic discrimination, instead the conditional probabilities , the log-ratio or log-odds:

$$\ln \frac{p(1|x)}{p(0|x)}.$$

is approximated by some linear function. Suppose, for a moment that

$$\ln \frac{p(1|x)}{p(0|x)} = w'x + w_o. \tag{3.6.15}$$

Since $p(0|x) = 1 - p(1|x)$ we get that under (3.6.15)

$$\frac{p(1|x)}{1 - p(1|x)} = \exp[w'x + w_o],$$

so that

$$p(1|x) = \frac{\exp[w'x + w_o]}{1 + \exp[w'x + w_o]}, \quad p(0|x) = \frac{1}{1 + \exp[w'x + w_o]}.$$

Hence, in logistic discrimination the parameters $(w', w_o)' \in \mathbb{R}^{d+1}$ are searched so that the functions

$$\eta_1(x) := \frac{\exp[w'x + w_o]}{1 + \exp[w'x + w_o]}, \quad \eta_0(x) := \frac{1}{1 + \exp[w'x + w_o]}$$

(in a sense) are the best fits to the conditional probabilities $p(1|x)$ and $p(0|x)$. These functions are more realistic than linear: always positive and they sum up to one. After finding the functions η_0 and η_1, the classifier is, obviously, as follows

$$g(x) = \begin{cases} 1 & \text{if } \eta_1(x) \geq \eta_0(x), \\ 0 & \text{else.} \end{cases} \tag{3.6.16}$$

Since $\eta_1(x) = \eta_0(x)$ iff

$$\ln \frac{\eta_1(x)}{\eta_0(x)} = w'x + w_o = 0,$$

we see that (3.6.16) is a linear discriminant:

$$g(x) = \begin{cases} 1 & \text{if } w'x + w_o \geq 0, \\ 0 & \text{else.} \end{cases} \tag{3.6.17}$$

Note that there is no difference, whether we model linearly the ratio $\frac{p(1|x)}{p(0|x)}$ or $\frac{p(0|x)}{p(1|x)}$.

Estimating the parameters: conditional max-likelihood. Estimating the parameters w and w_o is done with maximum likelihood estimation which entails finding the set of parameters for which the probability of the observed data is greatest. The maximum likelihood equation is derived from the probability distribution of the *labels*, only. Since our sample is iid, then given features $x_1, \ldots, x_n$, the (conditional) probability to obtain the classes $y_1, \ldots, y_n$ is

$$L := \prod_{i:y_i=1} p(1|x_i) \prod_{i:y_i=0} p(0|x_i).$$

Replacing $p(i|x)$ with the functions $\eta_i(x; w, w_o)$, we get conditional likelihood function:

$$L(w, w_o) := \prod_{i:y_i=1} \eta_1(x_i; w, w_o) \prod_{i:y_i=0} \eta_0(x_i; w, w_o).$$

The aim is to find the parameters w, w_o (equivalently, the functions η_i) so that $L(w, w_o)$ is maximal. For that the conditional log-likelihood function is used:

$$\begin{aligned} l(w, w_o) &:= \ln\Big(\prod_{i:y_i=1} \eta_1(x_i; w, w_o) \prod_{i:y_i=0} \eta_0(x_i; w, w_o) \Big) \\ &= \sum_{i:y_i=1} \ln \eta_1(x_i; w, w_o) + \sum_{i:y_i=0} \ln \eta_0(x_i; w, w_o) \\ &= \sum_{i:y_i=1} (w'x_i + w_o) - \sum_{i=1}^{n} \ln(1 + \exp[w'x_i + w_o]) \\ &= \sum_{i=1}^{n} \Big(y_i(w'x_i + w_o) - \ln(1 + \exp[w'x_i + w_o]) \Big). \end{aligned}$$

To maximize $l(w, w_o)$, the vector of partial derivatives – gradient – is set to zero. In statistics, the gradient of log-likelihood function is called score and in our case it is as follows:

$$\begin{pmatrix} \frac{\partial l(w,w_o)}{\partial w_o} \\ \frac{\partial l(w,w_o)}{\partial w_1} \\ \cdots \\ \frac{\partial l(w,w_o)}{\partial w_d} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n \Big(y_i - \frac{\exp[w'x_i+w_o]}{1+\exp[w'x_i+w_o]} \Big) \\ \sum_{i=1}^n x_i^1 \Big(y_i - \frac{\exp[w'x_i+w_o]}{1+\exp[w'x_i+w_o]} \Big) \\ \cdots \\ \sum_{i=1}^n x_i^d \Big(y_i - \frac{\exp[w'x_i+w_o]}{1+\exp[w'x_i+w_o]} \Big) \end{pmatrix} = \sum_{i=1}^{n} \begin{pmatrix} y_i - \eta_1(x_i; w, w_o) \\ x_i^1 \big(y_i - \eta_1(x_i; w, w_o) \big) \\ \cdots \\ x_i^d \big(y_i - \eta_1(x_i; w, w_o) \big) \end{pmatrix}.$$

We end up with a system with $d+1$ non-linear equation that are solved via several numerical methods. In the book [7], you can find an algorithm for solving the set of equations via Newton-Raphson method.

With obtained estimates $\hat{w}$ and $\hat{w}_o$ the classifier, as in (3.6.17), is constructed:

$$g_n(x) = \begin{cases} 1 & \text{if } \hat{w}'x + \hat{w}_o \geq 0, \\ 0 & \text{else.} \end{cases}$$

More than two classes. When the number of classes are bigger than two, the probabilities are modeled as follows

$$\begin{aligned} p(i|x) &= \frac{\exp[w_i'x + w_{i0}]}{1 + \sum_{i=0}^{k-2} \exp[w_i'x + w_{i0}]}, \quad i = 0, \dots, k-2 \\ p(k-1|x) &= \frac{1}{1 + \sum_{i=0}^{k-2} \exp[w_i'x + w_{i0}]}. \end{aligned} \tag{3.6.18}$$

Hence, for k classes, there are $(k-1)(d+1)$ parameters, all estimated via conditional likelihood method. With these estimates $\hat{w}_i, \hat{w}_{i0}$, $i = 0, \dots, k-2$ the conditional probabilities are estimated as follows

$$\begin{aligned} \hat{p}(i|x) &= \frac{\exp[\hat{w}_i'x + \hat{w}_{i0}]}{1 + \sum_{i=0}^{k-2} \exp[\hat{w}_i'x + \hat{w}_{i0}]}, \quad i = 0, \ldots, k-2 \\ \hat{p}(k-1|x) &= \frac{1}{1 + \sum_{i=0}^{k-2} \exp[\hat{w}_i'x + \hat{w}_{i0}]} \end{aligned} \tag{3.6.19}$$

and the classifier is

$$g_n(x) = \arg \max_{0=1,\ldots,k-1} \hat{p}(i|x).$$

NB! The conditional likelihood method described here does not assume anything about $F(x)$.

On the consistency of (conditional) max-likelihood classifier

Logistic regression is classifying with the help of (conditional) max-likelihood method: the probability $p(1|x)$ is assumed to belong to a class (model) $\mathcal{P}$. Hence $\mathcal{P}$ is a subset of functions

$$\eta : \mathbb{R}^d \to [0, 1].$$

and the data-based estimate of it, let it be $\hat{\eta}_n$ is picked via maximum likelihood method. With $\hat{\eta}_n$, the classifier g_n, as usually, is the following: $g_n(x) = 1$ iff $\hat{\eta}_n(x) \geq 0.5$.

Is such a method consistent? It depends:

- on the complexity of the model;
- on the correctness of the model, i.e. whether $p(1|x) \in \mathcal{P}$.

Those two things are related: the bigger (complex) the model, the bigger the chance that it is correct. If the model is correct, then Bayes classifier belongs to the set

$$\mathcal{G} := \{g(x) = I_{\{\eta(x) \geq 0.5\}} : \eta \in \mathcal{P}\}$$

and the approximation error is zero. If the correct model in not too complex, it is plausible to hope that the estimation error converges to zero as well.

Correct model. Suppose the model is correct, i.e. $p(1|x) \in \mathcal{P}$. The consistency of the maximum likelihood method depends now on the complexity of $\mathcal{P}$. Note that we cannot any more measure the complexity of the set of corresponding classifiers $\mathcal{G}$, because we choose amongst the functions in $\mathcal{P}$. A very complex model can give a relatively simple $\mathcal{G}$. Theorem 15.2 in [1] gives sufficient conditions in terms of *metric entropy of $\mathcal{P}$* that guarantee the consistency of maximum likelihood classification (note: the consistency of maximum likelihood estimator is something else). It can be shown ([1], 15.3), that

$$\mathcal{P} := \left\{ \frac{\exp[w'x + w_o]}{1 + \exp[w'x + w_o]} : w \in \mathbb{R}^d, w_o \in \mathbb{R} \right\}$$

satisfies the assumptions of Theorem 15.2. Thus if (3.6.15) holds, then the classifier obtain by logistic regression is consistent, i.e. $R(g_n) \to R^*$ a.s.

Exercise: Prove that (3.6.15) holds, when class-conditional densities are

$$f_i(x) = c_i u(x) \exp[-\frac{1}{2}(x - m_i)'\Sigma(x - m_i)].$$

Hence, when class-conditional distributions are multivariate normal with the same covariance, then logistic regression gives consistent estimators.

Incorrect model. If the model is incorrect, then typically approximation error is bigger than zero, hence consistency is not possible. In this case we ask: does the convergence $R(g_n) \to R$, a.s. hold, where $R = \inf_{\mathcal{G}} R(g)$? It turns out that answer might be no even when the class $\mathcal{P}$ is finite. The reason is the following. Let, for any $\eta \in \mathcal{P}$, $l_n(\eta)$ be the conditional log-likelihood of random sample $(X_1, Y_1), \ldots, (X_n, Y_n)$:

$$l_n(\eta) = \sum_{i=1}^{n} \Big(\ln \eta(X_i) I_{\{1\}}(Y_i) + \ln(1 - \eta(X_i)) I_{\{0\}}(Y_i) \Big).$$

From SLLN, it follows that for every η

$$\frac{1}{n} l_n(\eta) \to E\big[p(1|X) \ln \eta(X) + \big(1 - p(1|X)\big) \ln(1 - \eta(X))\big] =: l(\eta) \quad \text{a.s.}.$$

If $\mathcal{P}$ is finite then from the convergence above, it follows that a.s.

$$\hat{\eta}_n := \arg\max_{\eta \in \mathcal{P}} l_n(\eta) = \arg\max_{\eta \in \mathcal{P}} l(\eta) =: \eta' \quad \text{eventually.} \tag{3.6.20}$$

If the model is correct, i.e. $p(1|x) \in \mathcal{P}$, then $\eta' = p(1|x)$ and that is the consistency of maximum-likelihood estimator. Generally, η' is the best *in log-likelihood sense* from the class $\mathcal{P}$. Unfortunately, the corresponding classifier $g'(x) = I_{\{\eta'(x) \geq 0.5\}}$ is not always the best classifier *in risk (error-probability) sense* from $\mathcal{G}$. The following simple exercise gives an easy counterexample.

Exercise: Let $d = 1$ and

$$\mathbf{P}(X = 0, Y = 0) = \mathbf{P}(X = 1, Y = 0) = \frac{2}{9}, \ \mathbf{P}(X = 0, Y = 1) = \frac{1}{9}, \ \mathbf{P}(X = 1, Y = 1) = \frac{4}{9}.$$

- Find R^* and g^*.
- Suppose $\mathcal{P} = \{\eta_1, \eta_2\}$, where $\eta_1(x) \equiv 0.45$, $\eta_2(x) \equiv 0.95$. Find the corresponding classifiers g_1, g_2, $R(g_1)$, $R(g_2)$ and R.
- Find $p(1|x)$, $l(\eta)$ and η'.
- Find

$$\arg\min_{i=1,2} E\big(Y - \eta_i(X)\big)^2.$$

Inconsistency of logistic regression: a counterexample. The counterexample of linear regression also shows the inconsistency of logistic regression. Let us recall the example: $d = 1$,

$$p(1|-m) = p(1|1) = p(0|m) = p(0|-1) = 1$$

and

$$\mathbf{P}(X = -m) = \epsilon, \quad \mathbf{P}(X = -1) = \frac{1}{2} - \epsilon, \quad \mathbf{P}(X = m) = \epsilon, \quad \mathbf{P}(X = 1) = \frac{1}{2} - \epsilon,$$

where $\epsilon < \frac{1}{4}$. The best linear – one point – classifier is any of them:

$$g_t(x) = 1 \quad \Leftrightarrow \quad x \geq t,$$

where $t \in (-1, 1]$. Thus $R = 2\epsilon$. Consider the set

$$\mathcal{P} = \Big\{ \frac{\exp[wx + a]}{1 + \exp[wx + a]} : w, a \in \mathbb{R} \Big\}.$$

Hence,every η is in the form

$$\eta(x) = \frac{\exp[wx + a]}{1 + \exp[wx + a]}.$$

The conditional log-likelihood is, thus,

$$\begin{aligned} l(\eta) = l(w, a) &= \ln \eta(-m)\epsilon + \ln(1 - \eta(-1))(0.5 - \epsilon) + \ln \eta(1)(0.5 - \epsilon) + \ln(1 - \eta(m))\epsilon \\ &= \big(- wm + a - \ln(1 + \exp[-wm + a])\big)\epsilon \\ &- \ln(1 + \exp[-w + a])(0.5 - \epsilon) \\ &+ \big(w + a - \ln(1 + \exp[w + a])\big)(0.5 - \epsilon) \\ &- \ln(1 + \exp[wm + a])\epsilon \\ \frac{\partial l}{\partial a} &= \big(1 - \frac{1}{1 + \exp[wm - a]}\big)\epsilon - \frac{1}{1 + \exp[w - a]}(0.5 - \epsilon) \\ &+ \big(1 - \frac{\exp[w + a]}{1 + \exp[w + a]}\big)(0.5 - \epsilon) - \frac{\exp[wm + a]}{1 + \exp[wm + a]}\epsilon. \end{aligned}$$

Then

$$\begin{aligned} \frac{\partial l}{\partial a}|_{a=0} &= \big(1 - \frac{1}{1 + \exp[wm]}\big)\epsilon - \big(\frac{1}{1 + \exp[w]}\big)(0.5 - \epsilon) \\ &+ \big(1 - \frac{\exp[w]}{1 + \exp[w]}\big)(0.5 - \epsilon) - \frac{\exp[wm]}{1 + \exp[wm]}\epsilon = 0. \end{aligned}$$

Hence, the optimal $a = 0$ and we have to minimize the fuction

$$\begin{aligned} l(w) :=: l(w, 0) &= \big(- wm - \ln(1 + \exp[-wm])\big)\epsilon - \ln(1 + \exp[-w])(0.5 - \epsilon) \\ &+ \big(w - \ln(1 + \exp[w])\big)(0.5 - \epsilon) - \ln(1 + \exp[wm])\epsilon. \end{aligned}$$

The equation $l'(w) = 0$ is

$$\epsilon\Big(- m + \frac{m \exp[-wm]}{1 + \exp[-mw]} - \frac{m \exp[mw]}{1 + \exp[mw]}\Big) + (0.5 - \epsilon)\Big(\frac{\exp[-w]}{1 + \exp[-w]} + 1 - \frac{\exp[w]}{1 + \exp[w]}\Big) =$$
$$\epsilon\Big(- m + \frac{m}{1 + \exp[mw]} - \frac{m \exp[mw]}{1 + \exp[mw]}\Big) + (0.5 - \epsilon)\Big(1 + \frac{1 - \exp[w]}{1 + \exp[w]}\Big)$$
$$\epsilon\Big(- \frac{2m \exp[wm]}{1 + \exp[mw]}\Big) + (0.5 - \epsilon)\Big(\frac{2}{1 + \exp[w]}\Big) = 0,$$

so that $l'(w) = 0$ is equivalent to

$$(0.5 - \epsilon) = \frac{\epsilon m \exp[mw]}{1 + \exp[mw]}(1 + \exp[w]) \tag{3.6.21}$$

When m is big enough, then the solution of (3.6.21) is strictly negative. For example, if $m = 1000$ and $\epsilon = 0.1$, then the solution of (3.6.21), say w^*, is $w^* \approx -0.0037$. It is important that w^* is strictly negative. Now

$$\eta'(x) = \frac{\exp[w^* x]}{1 + \exp[w^* x]}$$

and the corresponding rule is $g'(x) = 1$ iff $x < 0$ and the corresponding risk is $R' = 1 - 2\epsilon$.

References: About logistic discrimination read also [8] (sec 4.4), [9] (sec 10.7, 10.8), [7] (sec 4.4.).

3.6.3 (Fisher) linear discriminant analysis (LDA)

Recall that for any $w \in \mathbb{R}^d$, the class-conditional means and variances of $w'X$ are

$$E[w'X|Y=i] = w'm_i, \quad \mathrm{Var}[(w'X)|Y=i] = w'\Sigma_i w.$$

Also recall that

$$\mathrm{Var}(w'X) = E(\mathrm{Var}[w'X|Y]) + \mathrm{Var}(E[w'X|Y]) = w'\Sigma_W w + w'\Sigma_B w.$$

LDA aims to solve the following maximization problem:

$$\max_{w:\|w\|=1} \frac{w'\Sigma_B w}{w'\Sigma_W w} \tag{3.6.22}$$

based on data. With the solution w (and some constant w_0), a linear classifier is then constructed. We know that (3.6.22) is equivalent to

$$\max_{w:\|w\|=1} J'(w), \quad J'(w) = \frac{(w'(m_1 - m_0))^2}{w'\Sigma_W w}$$

and the solution is

$$w \propto \Sigma_W^{-1}(m_1 - m_0). \tag{3.6.23}$$

Since we are interested in the hyperplane spanned by w, we can take the solution as

$$w = \Sigma_W^{-1}(m_1 - m_0). \tag{3.6.24}$$

Theoretical justification:

1. The solution w is such that projecting the feature vector onto the sub-space spanned by w, the class-conditional means $w'm_i$ were possibly far from each other and the weighted sum of conditional variances $w'\Sigma_1 w\pi_1 + w'\Sigma_0 w\pi_0 = w'\Sigma_W w$ were relatively small.

2. In a special case, when $\Sigma_1 = \Sigma_0$, maximizing J' is equivalent to maximizing

$$J(w) = \frac{w'(m_1 - m_0)}{(w'\Sigma_0 w)^{\frac{1}{2}} + (w'\Sigma_1 w)^{\frac{1}{2}}}.$$

   According to Lemma 3.3.1, this equals to minimizing upper bound to R.

3. If class-conditional distributions are normal and $\Sigma_1 = \Sigma_0$, then our solution (3.6.24) is the one that defines the Bayes classifier.

Data-based estimates. Recall the sample-based estimates:

$$\hat{\Sigma}_X := \frac{1}{n}\sum_{j=1}^{n}(x_j-\hat{m})(x_j-\hat{m})', \quad \hat{m} := \frac{1}{n}\sum_{j=1}^{n} x_j,$$
$$\hat{\Sigma}_i := \frac{1}{n_i}\sum_{j:y_j=i}(x_j-\hat{m}_i)(x_j-\hat{m}_i)', \quad \hat{\pi}_i := \frac{n_i}{n}, \quad \hat{m}_i = \frac{1}{n_i}\sum_{j:y_j=i} x_j,$$
$$S_i := \sum_{j:y_j=i}(x_j-\hat{m}_i)(x_j-\hat{m}_i)', \quad S_W := \sum_{i=0}^{k-1} S_i, \quad S_B := \sum_{i=0}^{k-1} n_i(\hat{m}_i-m)(\hat{m}_i-m)'.$$

Replacing the unknown Σ_W and (m_1-m_0) by the corresponding estimates, we get the empirical version of the solution (3.6.24):

$$\hat{w} = \hat{\Sigma}_W^{-1}(\hat{m}_1-\hat{m}_0) = nS_W^{-1}(\hat{m}_1-\hat{m}_0). \tag{3.6.25}$$

The vector $\hat{w}$ maximizes

$$\hat{J}(w) := \frac{w'S_Bw}{w'S_Ww}$$

and since (recall (1.2.4))

$$S_B = \frac{n_1n_0}{n}(\hat{m}_1-\hat{m}_0)(\hat{m}_1-\hat{m}_0)',$$

we have that maximizing $\hat{J}(w)$ equals to maximizing:

$$\frac{w'(\hat{m}_1-\hat{m}_0)(\hat{m}_1-\hat{m}_0)'w}{w'S_Ww} = \frac{w'(\hat{m}_1-\hat{m}_0)(\hat{m}_1-\hat{m}_0)'w}{w'(S_0+S_1)w} = \frac{\big(w'(m_1-m_0)\big)^2}{s_0^2+s_1^2}, \tag{3.6.26}$$

where

$$s_i^2 := w'S_iw = \sum_{j:y_j=i}(w'x_j-w'\hat{m}_i)^2, \quad i=0,1.$$

Hence, for any $w\in\mathbb{R}^d$ such that $\|w\|=1$, we consider one-dimensional sample

$$(w'x_1,y_1),\dots,(w'x_n,y_n).$$

LDA looks for w such that the objects from different classes were well separated: most of the elements with label 0 in one side and the elements with label 1 in other side. If such a w exists, then classifying one-dimensional sample is usually easy. The separation is measured:

1. via the difference of means $|w'\hat{m}_0 - w'\hat{m}_1|$ (has to be large)
2. via the sum of scatters $s_1^2+s_0^2$ (has to be small).

These two considerations give (3.6.26) (this is the empirical version of theoretical justification argument 1).

Typically, LDA does not specify how to classify the one dimensional data

$$(\hat{w}'x_1, y_1), \ldots, (\hat{w}'x_n, y_n) \tag{3.6.27}$$

after the original sample has been projected. If one-dimensional classification is linear (one point), then the overall classification is linear as well. Recall, that when class-conditional distributions of X are multivariate normal with equal covariance matrix Σ, then the Bayes rule (recall (3.6.3)) is: $g^*(x) = 1$ iff

$$|\underline{w}'x - \underline{w}'m_1| < |\underline{w}'x - \underline{w}'m_0| - 2\ln\frac{\pi_1}{1-\pi_1},$$

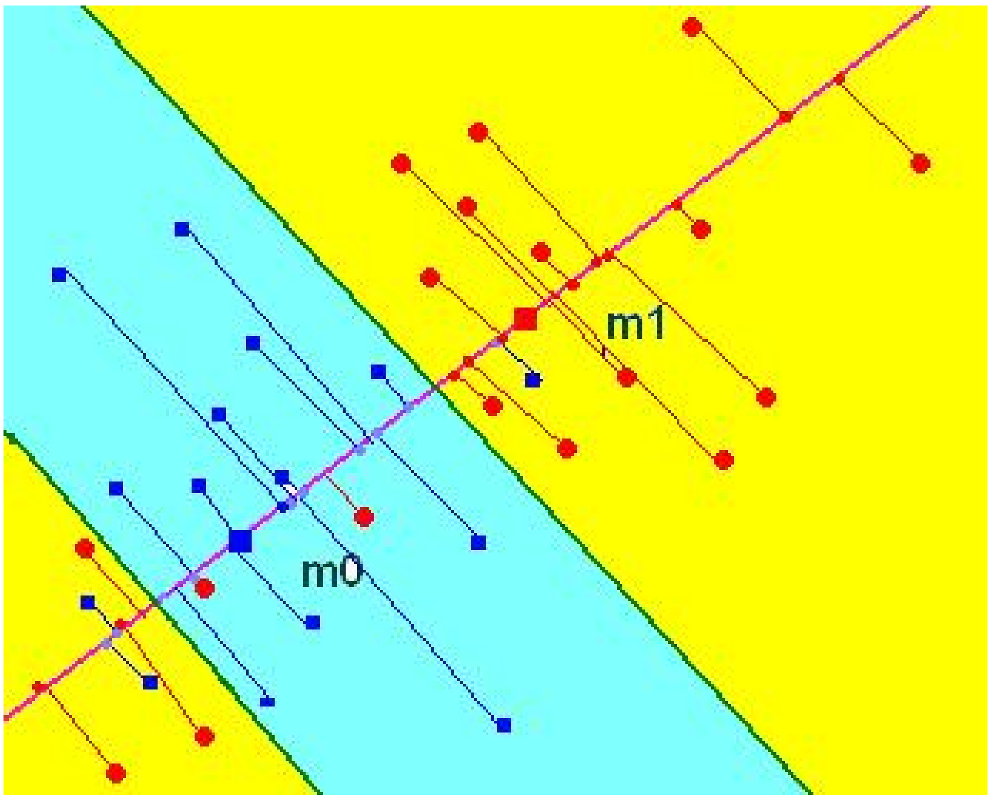
where $\underline{w} = \Sigma^{-1}(m_1 - m_0)$, i.e. $\underline{w}$ is the same as (3.6.24) (since in this particular case $\Sigma_W = \Sigma$). If it is reasonable to believe that this is the case, then the final rule could be as follows: $g_n(x) = 1$ iff

$$|\hat{w}'x - \hat{w}'\hat{m}_1| < |\hat{w}'x - \hat{w}'\hat{m}_0| - 2\ln\frac{\hat{\pi}_1}{\hat{\pi}_0}, \tag{3.6.28}$$

Note also that $\hat{w}$ is proportional to the solution of linear regression. Hence, when classifying linearly using LDA, the theoretical properties of the classifier g_n might be very similar to that of obtained by linear regression. Thus, without additional knowledge about the distribution $F(x, y)$, a linear classifier based on LDA can behave very badly for arbitrary big n.

In general, finding one-dimensional sample (3.6.27) can be considered as an intermediate step in classification for reducing the dimensionality. After it has been done, some other methods that could perhaps not used in d-dimensional space, can then applied. The authors of [7] suggest to use ERM-principle for classifying (3.6.27). But no matter how sophisticated classifier is used for (3.6.27), the final classifier in R^d is still quite limited (see picture below).

References: About LDA (especially for many-classes) read [7, 8] (sec 4.3 in both), [6] (sec 4.11), [9] (sec 6.6.).


m1
m0

Chapter 4

Support vector machines

4.1 Preliminaries: Lagrange's multipliers

Saddle-point. Let $L(x, \alpha)$ be an arbitrary function, and let

$$\tilde{f}(x) := \max_{\alpha} L(x, \alpha), \quad \theta(\alpha) := \min_{x} L(x, \alpha).$$

Exercise:

1. Prove that

$$\min_x \max_\alpha L(x, \alpha) = \min_x \tilde{f}(x) \geq \max_\alpha \theta(\alpha) = \max_\alpha \min_x L(x, \alpha).$$

2. Let

$$\tilde{f}(x^*) = \min_x \tilde{f}(x), \quad \theta(\alpha^*) = \max_\alpha \theta(\alpha). \tag{4.1.1}$$

   Prove that $\tilde{f}(x^*) = \theta(\alpha^*)$ – this is called **strong duality** – implies that (x^*, α^*) is a **saddle point**:

$$L(x, \alpha^*) \geq L(x^*, \alpha^*) \geq L(x^*, \alpha), \quad \forall x, \alpha$$

   and

$$\tilde{f}(x^*) = L(x^*, \alpha^*) = \theta(\alpha^*). \tag{4.1.2}$$

3. Prove that if (x^*, α^*) is a saddle point, then (4.1.2) and (4.1.1) hold.

Primal and dual problem. Consider the **(primal) problem**:

$$\begin{aligned} &\min\nolimits_{x \in \mathbb{R}^d} f(x) \\ \text{subject to} \quad &g(x) \leq 0, \end{aligned} \tag{4.1.3}$$

where

$$f : \mathbb{R}^d \to \mathbb{R}, \quad g : \mathbb{R}^d \to \mathbb{R}^m.$$

Here $g(x) \leq 0$ means $g_i(x) \leq 0$ for all $i = 1, \ldots, m$.

Lagrangian

$$L(x, \alpha) = f(x) + \alpha' g(x) = f(x) + \sum_{i=1}^{m} \alpha_i g_i(x),$$

where $\alpha' = (\alpha_1, \ldots \alpha_m) \in [0, \infty)^m$. It is easy to see (check!) that the problem (4.1.3) is equivalent to the following problem

$$\min_{x \in \mathbb{R}^d} \max_{\alpha \geq 0} L(x, \alpha) = \min_{x \in \mathbb{R}^d} \tilde{f}(x), \tag{4.1.4}$$

where

$$\tilde{f}(x) := \max_{\alpha \geq 0} L(x, \alpha).$$

The equivalence means: if x^* is the solution of (4.1.4), it is also the solution of the primal problem and $f(x^*) = \tilde{f}(x^*)$.

Change the order of maximization and minimization to get the **dual problem**:

$$\max_{\alpha \geq 0} \min_{x \in \mathbb{R}^d} L(x, \alpha) = \max_{\alpha \geq 0} \theta(\alpha), \tag{4.1.5}$$

where

$$\theta(\alpha) := \min_{x \in \mathbb{R}^d} L(x, \alpha).$$

Let x^* and α^* be the solutions of primal and dual problems, respectively. Thus

$$\tilde{f}(x^*) \geq \theta(\alpha^*).$$

If the inequality above is equality (strong duality), then (x^*, α^*) is a saddle point and (4.1.2) holds:

$$\tilde{f}(x^*) = f(x^*) + \alpha^{*\prime} g(x^*) = L(x^*, \alpha^*) = \theta(\alpha^*) = \min_{x} L(x, \alpha^*). \tag{4.1.6}$$

From the last equality we see that then $x^* = \arg\min_x L(x, \alpha^*)$.

Therefore, under strong duality, the solution of primal problem x^* can be found via the dual problem as follows:

- For every $\alpha \geq 0$ find x_α such that $L(x_\alpha, \alpha) = \min_x L(x, \alpha) = \theta(\alpha)$.
- Solve dual problem.
- With the solution α^* of dual problem, find corresponding x_{α^*}. This is the solution of primal problem.

Solving dual problem is usually easier as the original one, because dual problem is always concave, even if f and g are neither concave nor convex.

Recall that any solution x^* of primal problem is a minimizer of $\tilde{f}$ and $f(x^*) = \tilde{f}(x^*)$. From (4.1.6), it follows that under strong duality

$$f(x^*) = \tilde{f}(x^*) = f(x^*) + \alpha^{*\prime} g(x^*) = L(x^*, \alpha^*)$$

so that $\alpha^{*\prime} g(x^*) = 0$. Since $g(x^*) \leq 0$ and $\alpha^* \geq 0$, then $\alpha^{*\prime} g(x^*) = 0$ iff

$$\alpha_i^* g_i(x^*) = 0 \quad i = 1, \ldots, m.$$

Hence when strong duality holds, i.e. $\tilde{f}(x^*) = \theta(\alpha^*)$ then the pair of solutions (α^*, x^*) satisfies so-called **KKT (Karush-Kuhn-Tucker)** (optimality) conditions:

$$\begin{aligned} \alpha_i^* &\geq 0, \quad i = 1, \ldots, m \\ g_i(x^*) &\leq 0, \quad i = 1, \ldots, m \\ \alpha_i^* g_i(x^*) &= 0, \quad i = 1, \ldots, m \end{aligned}$$

Equality constraints. Some of the constraints could be equalities, hence the primal problem

$$\begin{aligned} &\min_{x \in \mathbb{R}^d} f(x) \\ \text{subject to } &g(x) \leq 0, \\ &e(x) = 0 \end{aligned} \tag{4.1.7}$$

where

$$f : \mathbb{R}^d \to \mathbb{R}, \quad g : \mathbb{R}^d \to \mathbb{R}^m, \quad e : \mathbb{R}^d \to \mathbb{R}^l$$

and $e(x) = 0$ means $e_i(x) = 0$ for every $i = 1, \ldots, l$.

Let us reformulate

$$\begin{aligned} &\min_{x \in \mathbb{R}^d} f(x) \\ \text{subject to } &g(x) \leq 0, \\ &e(x) \leq 0 \\ &-e(x) \leq 0. \end{aligned} \tag{4.1.8}$$

Corresponding Lagrangian:

$$L(x, \alpha, \beta^+, \beta^-) = f(x) + \alpha' g(x) - \beta_+' e(x) + \beta_-' e(x) = f(x) + \alpha' g(x) + (\beta_- - \beta_+)' e(x).$$

Define $\beta := \beta_- - \beta_+$ (can be negative), we get Lagrangian as

$$L(x, \alpha, \beta) = f(x) + \alpha' g(x) + \beta' e(x), \quad \text{where} \quad \alpha \in [0, \infty)^m, \quad \beta \in \mathbb{R}^l.$$

Dual problem:

$$\max_{\alpha\geq 0,\beta} \theta(\alpha,\beta), \tag{4.1.9}$$

where $\theta(\alpha,\beta) := \min_{x\in\mathbb{R}^d} L(x,\alpha,\beta)$. Note that in (4.1.9) the search space is *nonnegative* α's and *all* β's.

When strong duality holds? A sufficient condition is so called **Slater's condition**:

- functions f and g_i are convex and e_i affine;
- there exists x_o so that $g_i(x_o) < 0$ for every $i = 1,\ldots,m$ and $e_i(x_o) = 0$ for every $i = 1,\ldots,l$.

Quadratic programming:

$$\begin{aligned} &\min_x \frac{1}{2}x'Kx + c'x \\ \text{subject to}\quad & Ax + d \leq 0 \\ & Bx + e = 0 \end{aligned} \tag{4.1.10}$$

If K is positive semi-definite (symmetric) matrix, then this is a convex optimization problem. It can be shown that when the problem is feasible (i.e. the solution exists), then in this case the strong duality holds ([10], Thm 5.20). Hence, Lagrange method can be used to solve it.

Example. Consider the problem

$$\begin{aligned} &\min_x \frac{1}{2}x'Kx + c'x \\ \text{subject to}\quad & Ax + d \leq 0, \end{aligned} \tag{4.1.11}$$

where K is positive definite, so that K^{-1} exists. Lagrangian:

$$L(x,\alpha) = \frac{1}{2}x'Kx + c'x + \alpha'(Ax+d).$$

Now

$$\nabla_x L(x,\alpha) = 0 \quad\Leftrightarrow\quad Kx + A'\alpha + c = 0 \quad\Rightarrow\quad x_\alpha = -K^{-1}(A'\alpha + c).$$

Plugging x_α into Lagrangian, we obtain

$$\frac{1}{2}(A'\alpha+c)'K^{-1}KK^{-1}(A'\alpha+c) - c'K^{-1}(A'\alpha+c) + \alpha'\big(-AK^{-1}(A'\alpha+c)+d\big) =$$
$$\frac{1}{2}\alpha'AK^{-1}A'\alpha + \frac{1}{2}c'K^{-1}c + \alpha'AK^{-1}c - c'K^{-1}A'\alpha - c'K^{-1}c - \alpha'AK^{-1}A'\alpha - \alpha'AK^{-1}c + \alpha'd =$$
$$-\frac{1}{2}\alpha'AK^{-1}A'\alpha - \frac{1}{2}c'K^{-1}c - c'AK^{-1}A'\alpha + d'\alpha.$$

Hence the dual problem is

$$\max_{\alpha} -\frac{1}{2}\alpha' A K^{-1} A' \alpha + [d' - c' K^{-1} A']\alpha \tag{4.1.12}$$
$$\text{subject to} \quad \alpha \geq 0,$$

The dual problem can be solved by gradient method or by *matlab*. With α^* being the solution of dual problem, the solution of primal problem is

$$x^* = -K^{-1}(A'\alpha^* + c).$$

Exercise: Consider (4.1.10) where K is positive definite, so that K^{-1} exists. Show the the dual problem is

$$\max_{\alpha} -\frac{1}{2}(\alpha' A + \beta' B) K^{-1}(A'\alpha + B'\beta) + [d' - c'K^{-1}A']\alpha + [e' - c'K^{-1}B']\beta - \frac{1}{2}c'K^{-1}c \tag{4.1.13}$$
$$\text{subject to} \quad \beta, \quad \alpha \geq 0$$

and with (α^*, β^*) being the solutions of dual problem, the solution of primal is

$$x^* = -K^{-1}(A'\alpha^* + B'\beta^* + c).$$

4.2 Support vector classification

Assumption: In this chapter we consider the case $k = 2$ and classes are labeled as -1 and +1. The loss-function is symmetric. Every linear classifier is thus in the form

$$g(x) = \operatorname{sgn}(w'x + w_0), \quad \text{where } \operatorname{sgn}(x) = \begin{cases} 1, & \text{if } x \geq 0; \\ -1, & \text{if } x < 0. \end{cases}$$

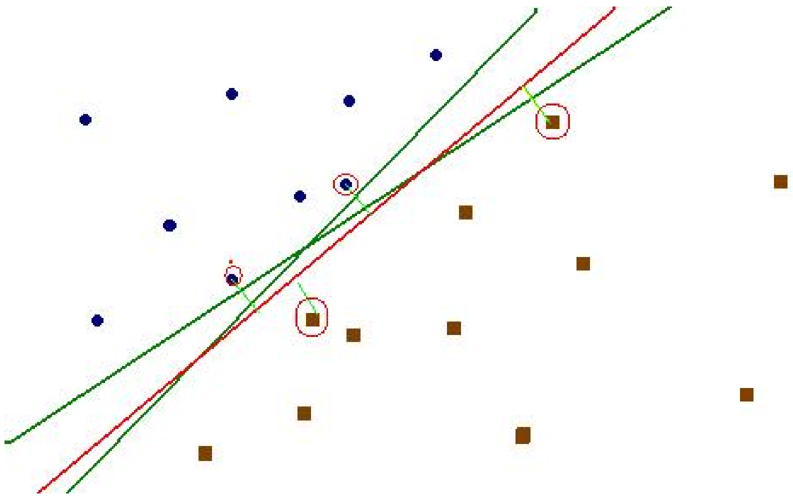
4.2.1 Linearly separable sample – hard margin

Suppose that the training set is *linearly separable*: there exists at least one hyperplane that classifies correctly (training error is zero). In other words, there exists a $w \in \mathbb{R}^d$ and w_o such that for every pair (x_i, y_i)

$$y_i(w'x_i + w_o) \geq 0, \quad i = 1, \ldots, n. \tag{4.2.1}$$

Every such hyperplane is called *separating hyperplane*. When $\|w\| = 1$ then the left side of (4.2.1) is the distance of x_i from the hyperplane. This distance is called **geometrical margin** of (x_i, y_i). Typically there are more than one separating hyperplanes and the idea of large margin methods is to choose the one that in some sense lies in the "middle" of different subsamples (a subsample is the elements of the sample with the same labels). In other words, amongst the separating hyperplanes choose the one with largest distance to the nearest point. This distance is called **geometrical margin of the sample**. Hence, if w, w_o correspond to a separating hyperplane and $\|w\| = 1$, the geometrical margin of sample is

$$\min_{i=1,\dots,n} y_i(w'x_i + w_o).$$



Hence, the separating hyperplane with maximal margin can be found by the following problem

$$\max_{\gamma,w,w_o} \gamma \tag{4.2.2}$$
$$\text{subject to } y_i(w'x_i + w_o) \geq \gamma, \quad i = 1,\dots,n$$
$$\|w\| = 1.$$

Here γ is margin. It is not hard to see that (4.2.2) is equivalent to the following problem:

$$\max_{w,w_o} \frac{1}{\|w\|} \tag{4.2.3}$$
$$\text{subject to } y_i(w'x_i + w_o) \geq 1, \quad i = 1,\dots,n.$$

Exercise: Let (w, w_o) be the solution of (4.2.2) and (w^*, w_o^*) be the solution of (4.2.3). Prove that they define the same hyperplane: $w'x + w_o = 0$ iff $w^{*\prime}x + w_o^* = 0$. Show that the margin of this hyperplane is $\|w^*\|^{-1}$

The problem (4.2.3) is equivalent to

$$\min_{w,w_o} \frac{1}{2}\|w\|^2 \qquad (4.2.4)$$
$$\text{subject to } y_i(w'x_i + w_o) \geq 1, \quad i = 1, \ldots, n.$$

Exercise: Show that the problem (4.2.4) is a quadrating programming problem just as (4.1.10), but the matrix K is positively semidefinite but not positively definite and K^{-1} is not unique.

However, (4.2.4) can be solved with Lagrangian method just like quadratic programming. Slater's condition holds (the functions are convex and the inequalities can be strict for some w and w_o) and Lagrangian is

$$L(w, w_o, \alpha) = \frac{\|w\|^2}{2} + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(w'x_i + w_o)\big).$$

Setting the partial derivatives to zero, we obtain

$$\frac{\partial}{\partial w_o} L(w, w_o, \alpha) = -\sum_{i=1}^{n} \alpha_i y_i = 0$$
$$\frac{\partial}{\partial w_j} L(w, w_o, \alpha) = w^j - \sum_{i=1}^{n} \alpha_i y_i x_i^j = 0, \quad j = 1, \ldots d.$$

Therefore $\sum_i \alpha_i y_i = 0$ and

$$w_\alpha^j = \sum_{i=1}^{n} \alpha_i y_i x_i^j, \quad i = 1, \ldots, d \quad \Leftrightarrow \quad w_\alpha = \sum_{i=1}^{n} \alpha_i y_i x_i \qquad (4.2.5)$$

Plugging these equalities into $L(w, w_o, \alpha)$, we get

$$\theta(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j}^{n} \alpha_i \alpha_j y_i y_j x_i' x_j \qquad (4.2.6)$$

so that dual problem is

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j}^{n} \alpha_i \alpha_j y_i y_j x_i' x_j \qquad (4.2.7)$$
$$\text{subject to } \alpha_i \geq 0, \quad \sum_{i=1}^{n} y_i \alpha_i = 0.$$

Let $\alpha^* = (\alpha_1^*, \ldots, \alpha_n^*)'$ be the solution of (4.2.7). From (4.2.5), we get the solution of primal problem (4.2.4)

$$w^* = w_{\alpha^*} = \sum_{i=1}^{n} \alpha_i^* y_i x_i. \qquad (4.2.8)$$

KKT:

$$\begin{aligned}\alpha_i^* &\geq 0, \quad && i=1,\dots,n\\ y_i(w^{*\prime}x_i+w_0^*) &\geq 1, \quad && i=1,\dots,n\\ \alpha_i^*\big(1-y_i(w^{*\prime}x_i+w_0^*)\big) &= 0, \quad && i=1,\dots,n\end{aligned}$$

Note: $\alpha_i^* > 0$ implies that $y_i(w^{*\prime}x_i + w_0^*) = 1$: the distance of x_i from the hyperplane is minimal (equals to margin). The sample points x_i having the corresponding α_i^* not zero ar called **support vectors** . The support vectors are closest to the separating hyperplane and they are most informative: the solution w^* depends on support vectors, only . Indeed,

$$w^* = \sum_{i\in\mathrm{SV}} \alpha_i^* y_i x_i, \tag{4.2.9}$$

where SV is the set of indexes of support vectors. Eliminating all other points from the sample apart the support vectors, we would have the same solution. Unfortunately we do not know them in advance.

To find the constant w_0^* take any support vector x_i. Thus $y_i(w^{*\prime}x_i + w_0^*) = 1$ or

$$w_0^* = y_i - w^{*\prime}x_i.$$

With obtained w^* and w_o^*, the classification rule is

$$g(x) = \mathrm{sgn}\Big(w^{*\prime}x + w_o^*\Big) = \mathrm{sgn}\Big(\sum_{i\in\mathrm{sv}} y_i\alpha_i^* x_i' x + w_o^*\Big). \tag{4.2.10}$$

By strong duality $\theta(\alpha^*) = f(x^*)$. Thus

$$\sum_i \alpha_i^* - \frac{1}{2}\sum_{i,j}^n \alpha_i^*\alpha_j^* y_i y_j x_i' x_j = \frac{1}{2}\|w^*\|^2 = \frac{1}{2}\sum_{i,j}^n \alpha_i^*\alpha_j^* y_i y_j x_i' x_j.$$

Therefore

$$\sum_i \alpha_i^* = \sum_{i,j}^n \alpha_i^*\alpha_j^* y_i y_j x_i' x_j = \|w^*\|^2$$

and the margin of the optimal hyperplane can be found via α^* as follows:

$$\gamma = \frac{1}{\|w^*\|} = \Big(\sum_{i\in\mathrm{sv}} \alpha_i^*\Big)^{-\frac{1}{2}}. \tag{4.2.11}$$

The smaller are α_i^*'s, the bigger margin.

The following exercise shows that the support vectors are not always uniquely defined. The support vectors that appear in all possible expansions ar called *essential support* *vectors.*

Exercise: Let $d = 2$,

$$x_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, x_2 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}, x_3 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, x_4 = \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

$y_1 = -1$, $y_2 = y_3 = y_4 = 1$.
This is a linearly separable sample with margin $\frac{1}{2}$.
Show that the matrix $(x_i' x_j)_{ij}$ is

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{pmatrix}$$

and

$$\sum_i \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j x_i' x_j = \sum_i \alpha_i - \frac{1}{2}(\alpha_1^2 + \alpha_2^2 + \alpha_4^2) + \alpha_2 \alpha_4 = \theta(\alpha).$$

Let $\alpha^* = (\alpha_i^*)$ be the solution of dual (4.2.7) and let w^* be the corresponding vector. Control whether α^* satisfies the conditions $\sum_i \alpha_i^* = 4$ so that $\|w^*\| = 2$, and $\theta(\alpha^*) = 2$. Show that the following vectors all are the solutions of (4.2.7):

$$(2, 0, 2, 0), \quad (2, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}), \quad (2, 1, 0, 1).$$

The following exercise gives a geometric interpretation to support vectors

Exercise. Let $x_1, \ldots, x_n$ be linearly separable; let K_+ and K_- be the convex hulls of subsamples:

$$K_+ := \{ \sum_{i:y_i=+1} c_i x_i \ : \ c_i \geq 0, \sum_i c_i = 1\}, \quad K_- := \{ \sum_{i:y_i=-1} c_i x_i \ : \ c_i \geq 0, \sum_i c_i = 1\}.$$

Show that optimal hyperplane is orthogonal to the shortest line joining K_+ and K_-. To show that consider the following optimization problem:

$$\min_{c_1, \ldots, c_n} \| \sum_{i:y_i=+1} c_i x_i - \sum_{i:y_i=+1} c_i x_i \| \tag{4.2.12}$$

$$\sum_{i:y_i=+1} c_i = 1, \quad \sum_{i:y_i=-1} c_i = 1, \quad c_i \geq 0.$$

Let c_i^* be the solution of (4.2.12),

$$v^* := c^+ - c^-, \quad c^+ := \sum_{i:y_i=+1} c_i^* x_i, \quad c^+ := \sum_{i:y_i=+1} c_i^* x_i.$$

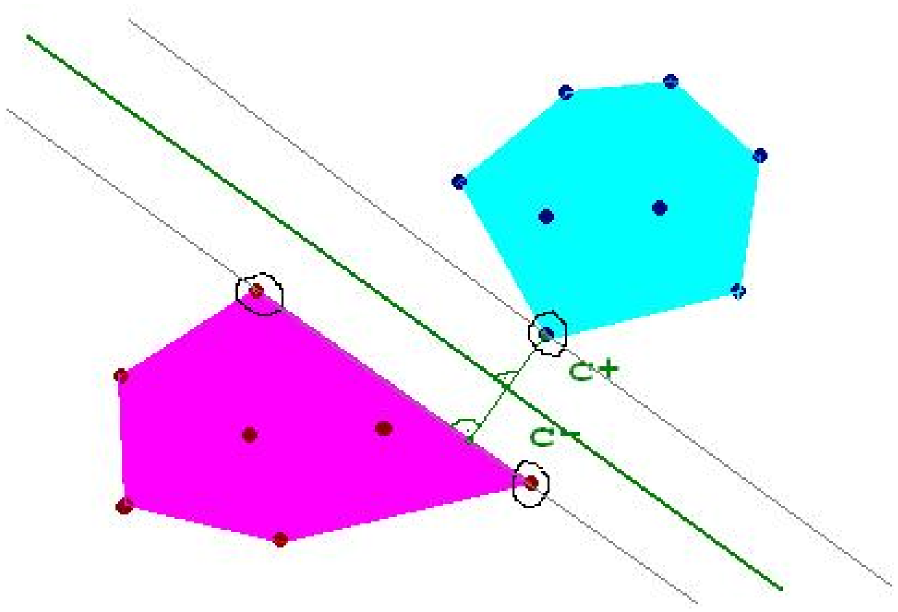
Prove that the solutions of (4.2.12) are

$$c_i^* = \frac{2\alpha_i^*}{\sum_i \alpha_i^*},$$

where α_i^* are the solution of dual problem (4.2.7). Conclude that

$$v^* = \frac{2w^*}{\|w^*\|^2},$$

where $w^* = \sum_i \alpha_i^* y_i x_i$. Hence v^* and w^* span the same one-dimensional hyperplane and w^* is proportional to the smallest interval joining K_+ and K_-. Show that margin γ satisfies the equality $2\gamma = \|v^*\|$. Show that w_o is such that optimal hyperplane crosses v^* at the middle. (Hint: show that $w^{*'}(c^+ + c^-) + 2w_o = 0$).



4.2.2 Linearly not separable sample – soft margin

If the sample is not linearly separable, then the problem (4.2.2) is not feasible. In this case concept of (geometrical) margin is relaxed as follows. Consider a hyperplane

$$H = \{x : w'x + w_o = 0\},$$

where $\|w\| = 1$ and let $\gamma > 0$ be a non-negative constant still called as margin. Then for any (x_i, y_i) the **margin error** is

$$\big(\gamma - y_i(w'x_i + w_o)\big)_+ \quad \text{where} \quad (x)_+ = \begin{cases} x, & \text{if } x > 0; \\ 0, & \text{if } x \leq 0. \end{cases}$$

Hence the margin error of (x_i, y_i) is zero, if the distance of x_i from H is at least γ and H classifies (x_i, y_i) correctly. Otherwise the margin error is positive even if (x_i, y_i) is correctly classified. If (x_i, y_i) is incorrectly classified, then the margin error is larger than γ. When the sample is not linearly separable, there is no hyperplane (no pair w, w_o) and no marginal γ so that all margin errors are zero. In the presence of margin errors, γ is not obviously any more the geometrical margin of the training sample, therefore it is called a **soft margin.**

If the sample is linearly not separable, then there are many alternative ways to find the best classifier. Typically the alternatives are in form

$$\min_{w:\|w\|=1, w_o, \gamma\geq 0} \Big[h(\gamma) + D\sum_{i=1}^{n} u\big((\gamma - y_i(w'x_i + w_o))_+\big)\Big], \tag{4.2.13}$$

where h is decreasing as γ grows and u is non-decreasing. Hence the aim is simultaneously maximize the margin γ and minimize the margin errors. The constant $D > 0$ is the regularization constant that regulates the tradeoff between two problems: increasing D increases the penalty of margin errors so that the solution of (4.2.13) has smaller margin.

1-norm soft margin. The most natural choice for u and h are the identity functions (in particular $h(\gamma) = -\gamma$) resulting the following optimization problem

$$\min_{w:\|w\|=1, w_o, \gamma\geq 0} \Big(-\gamma + D\sum_{i=1}^{n} \big(\gamma - y_i(w'x_i + w_o)\big)_+\Big). \tag{4.2.14}$$

Note that it is meaningful to take $D \geq \frac{1}{n}$, since with $D < \frac{1}{n}$ the optimal $\gamma^* = \infty$. On the other hand, if the sample is non-separable, there exists a $D_o \leq 1$ so that $\gamma^* = 0$, if $D > D_o$. In section 4.2.3, we show that (4.2.14) is in a sense equivalent to the following problem

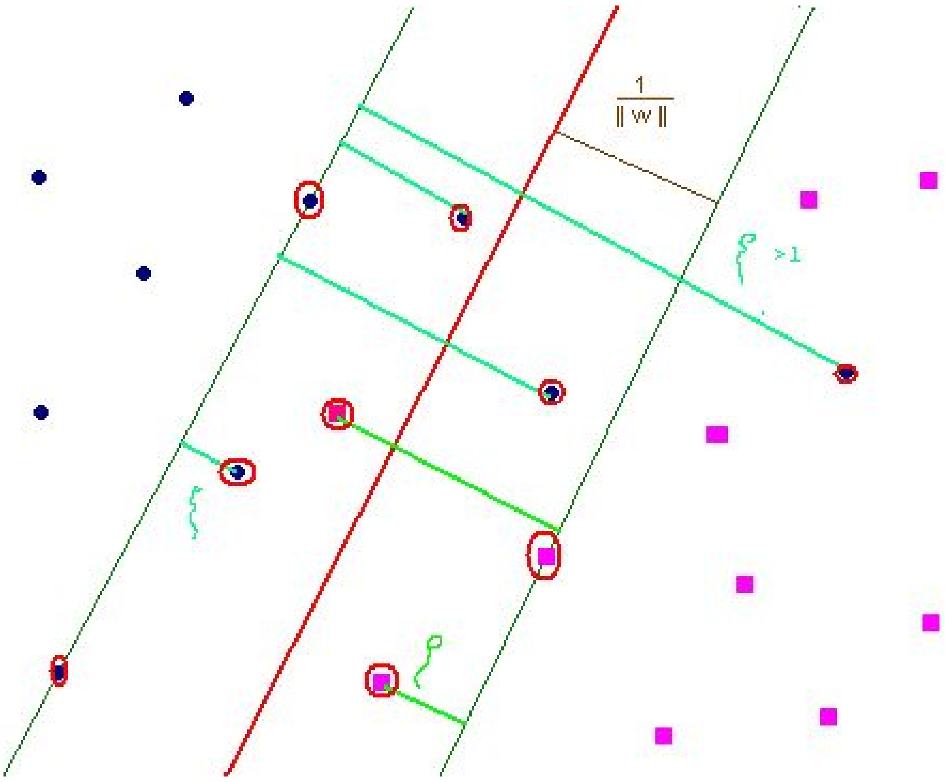
$$\min_{w:\|w\|=1, w_o, \gamma\geq 0} \Big(\frac{1}{2}\frac{1}{\gamma^2} + \frac{C}{\gamma}\sum_{i=1}^{n} \big(\gamma - y_i(w'x_i + w_o)\big)_+\Big). \tag{4.2.15}$$

The equivalence means that for every C, there exists a constant D (that depends on data) so that the solutions of (4.2.14) define the same hyperplane as the solutions of (4.2.15). Also for every D in a certain range, there exists a constant C (depending on the data) so that the solutions of (4.2.15) defines the same hyperplane as the ones of (4.2.14). Since (4.2.15) is more common in the literature, we shall mostly consider this problem. Note that (4.2.15) minimizes the sum of relative margin errors. Just like in the hard margin case, relaxing the condition $\|w\| = 1$ allows to replace γ by $\frac{1}{\|w\|}$ so that the problem (4.2.15) is equivalent to the following problem called as **1-norm soft margin SVM problem:**

$$\min_{w,w_o} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \big(1 - y_i(w'x_i + w_o)\big)_+. \tag{4.2.16}$$

To solve (4.2.16), the so called **slack variables** ξ_i $i = 1, \dots, n$ are introduced. With these variables the problem (4.2.16) is

$$\min_{w,w_o,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \tag{4.2.17}$$
$$\text{subject to } y_i(w'x_i + w_o) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \dots, n.$$



Let us solve (4.2.16). Lagrangian:

$$\begin{aligned} L(w, w_o, \xi, \alpha, \gamma) &= \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\gamma_i\xi_i + \sum_{i=1}^{n}\alpha_i\big(1 - \xi_i - y_i(w'x_i + w_o)\big) \\ &= \frac{1}{2}\|w\|^2 + \sum_{i=1}^{n}\xi_i(C - \alpha_i - \gamma_i) + \sum_{i=1}^{n}\alpha_i\big(1 - y_i(w'x_i + w_o)\big). \end{aligned}$$

Setting the derivatives with respect to w w_o and ξ equal to zero, we obtain

$$\begin{aligned} w_\alpha =& \sum_{i=1}^n \alpha_i y_i x_i \\ 0 =& \sum_{i=1}^n \alpha_i y_i \\ C =& \alpha_i + \gamma_i. \end{aligned}$$

Plugging w_α into $L(w, w_o, \xi, \alpha, \gamma)$ and taking account that $\sum_{i=1}^n \xi_i(C - \alpha_i - \gamma_i) = 0$, we obtain the dual problem

$$\theta(\alpha, \gamma) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}^n \alpha_i \alpha_j y_i y_j x_i' x_j.$$

Note that the obtained dual problem is the same we had for hard margin case (4.2.6), but the constraints are different. Indeed, the dual problem now is

$$\max_{\alpha,\gamma} \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}^n \alpha_i \alpha_j y_i y_j x_i' x_j, \tag{4.2.18}$$

$$\text{subject to } \alpha_i \geq 0, \quad \gamma_i \geq 0, \quad \sum_{i=1}^n \alpha_i y_i = 0, \quad \alpha_i + \gamma_i = C, \quad i = 1, \dots, n.$$

Since γ is not in the objective, the problem above is equivalent to

$$\max_{\alpha} \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}^n \alpha_i \alpha_j y_i y_j x_i' x_j, \tag{4.2.19}$$

$$\text{subject to } \sum_{i=1}^n \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C.$$

To summarize: the dual of the 1-norm soft margin problem differs from the dual of the hard-margin problem by the additional constraint $\alpha_i \leq C$ $\forall i$. In literature, that additional constraint is known as **box constraint**.

Let α^* and γ^* be the solutions of (4.2.19). Then the optimal vector is (as in the hard margin case (recall (4.2.8)):

$$w^* = \sum_{i=1}^n \alpha_i^* y_i x_i.$$

KKT:

$$\gamma^* + \alpha^* = C$$

$$\gamma_i^* \xi_i^* = 0, \quad \forall i \tag{4.2.20}$$

$$\alpha_i^*(1 - \xi_i^* - (w^{*\prime} x_i + w_o^*) y_i) = 0, \quad \forall i. \tag{4.2.21}$$

Support vectors are those x_i for which $\alpha_i^* > 0$. Thus, (4.2.21) implies that for every support vector the following holds:

$$1 - \xi_i^* - (w^{*\prime}x_i + w_o^*)y_i = 0 \quad \Leftrightarrow \quad (w^{*\prime}x_i + w_o^*)y_i \leq 1.$$

Hence, if x_i is a support vector, then (x_i, y_i) has a margin error: either misclassification, i.e. $(w^{*\prime}x_i + w_o^*)y_i < 0$ or the distance of x_i from the optimal hyperplane is smaller than marginal, i.e. $0 \leq (w^{*\prime}x_i + w_o^*)y_i \leq 1$.
Since $\alpha_i^* + \gamma_i^* = C$, from (4.2.20) it follows that $0 < \alpha_i^* < C$ implies $\xi_i^* = 0$ and (4.2.21) in turn implies $(w^{*\prime}x_i + w_o^*)y_i = 1$.
If $(w^{*\prime}x_i + w_o^*)y_i < 1$, then $\xi_i^* > 0$, so that $\gamma_i^* = 0$ and $\alpha_i^* = C$.
If $\alpha_i^* = 0$, then $\gamma_i^* = C$ so that by (4.2.21) $\xi_i^* = 0$ and $(w^{*\prime}x_i + w_o^*)y_i \geq 1$.

To summarize:

$$\begin{aligned}(w^{*\prime}x_i + w_o^*)y_i > 1 \Rightarrow \alpha_i^* = 0 &\Rightarrow (w^{*\prime}x_i + w_o^*)y_i \geq 1\\ 0 < \alpha_i^* < C &\Rightarrow (w^{*\prime}x_i + w_o^*)y_i = 1\\ (w^{*\prime}x_i + w_o^*)y_i < 1 \Rightarrow \alpha_i^* = C &\Rightarrow (w^{*\prime}x_i + w_o^*)y_i \leq 1.\end{aligned}$$

Sometimes the support vectors for which $0 < \alpha_i^* < C$ are called **in-bound** support vectors. They are classified correctly and their distance from the hyperplane equals to margin. The support vectors, for which $\alpha_i^* = C$ are called **bound** -support vectors. All vectors with margin error are bound support vectors.

The constant w_o^*. The constant could be determined via in-bound support vectors, since for them the following equality has to hold: $(w^{*\prime}x_i + w_o^*)y_i = 1$. With optimal w^* and w_o^* the classifications rule is, as previously, (recall (4.2.2)):

$$g(x) = \operatorname{sgn}\Big(w^{*\prime}x + w_o^*\Big) = \operatorname{sgn}\Big(\sum_{i\in \mathrm{sv}} y_i\alpha_i^* x_i' x + w_o^*\Big).$$

Remark: There might exist no in-bound support vectors. S

Exercise: Let $d = 1$, and $x_1 = y_1 = -1$, $x_2 = y_2 = 1$. Solve dual problem (4.2.19) and show the for some C, it holds: $\alpha_1 = \alpha_2 = C$.

2-norm soft margin SVM problem. Recall that 1-norm soft margin problem was obtained from the general problem (4.2.13) by taking $h(\gamma) = -\gamma$ and u as identity function. Another popular choice for u is quadratic function yielding the following problem

$$\min_{w:\|w\|=1, w_o, \gamma\geq 0} \Big(-\gamma + D\sum_{i=1}^n \big(\gamma - y_i(w'x_i + w_o)\big)_+^2 \Big). \tag{4.2.22}$$

Again, just as in 1-norm case, it can be shown that (in a sense) the problem is equivalent to the following problem

$$\min_{w:\|w\|=1, w_o, \gamma \geq 0} \Big(\frac{1}{2}\frac{1}{\gamma^2} + \frac{C}{2\gamma^2} \sum_{i=1}^{n} \big(\gamma - y_i(w'x_i + w_o)\big)_+^2 \Big).$$

Maximizing $\frac{1}{\|w\|}$ instead γ gives an equivalent form

$$\min_{w, w_o} \Big(\frac{1}{2}\|w\|^2 + \frac{C}{2} \sum_{i=1}^{n} \big(1 - y_i(w'x_i + w_o)\big)_+^2 \Big). \tag{4.2.23}$$

With slack variables (4.2.23) is

$$\begin{aligned} &\min_{w, w_o, \xi} \frac{1}{2}\|w\|^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 \\ &\text{subject to } y_i(w'x_i + w_o) \geq 1 - \xi_i \quad \xi_i \geq 0, \quad i = 1, \ldots, n. \end{aligned}$$

Note that we can remove the constraint $\xi_i \geq 0$, hence the problem (4.2.23) is

$$\begin{aligned} &\min_{w, \xi} \frac{1}{2}\|w\|^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 \\ &\text{subject to } y_i(w'x_i + w_o) \geq 1 - \xi_i \quad i = 1, \ldots, n. \end{aligned}$$

Lagrangian:

$$\begin{aligned} L(w, w_o, \xi, \alpha) &= \frac{1}{2}\|w\|^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 + \sum_{i=1}^{n} \alpha_i \big(1 - \xi_i - y_i(w'x_i + w_o)\big) \\ &= \frac{1}{2}\|w\|^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 - \sum_{i=1}^{n} \xi_i \alpha_i + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(w'x_i + w_o)\big) \\ &= \frac{1}{2}\|w\|^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 - \sum_{i=1}^{n} \xi_i \alpha_i + \sum_{i=1}^{n} \alpha_i \big(1 - y_i w'x_i\big) - w_o \sum_{i=1}^{n} \alpha_i y_i. \end{aligned}$$

Setting the derivatives with respect to w w_o and ξ to zero, we obtain

$$\begin{aligned} w_\alpha &= \sum_{i=1}^{n} \alpha_i y_i x_i \\ 0 &= \sum_{i=1}^{n} \alpha_i y_i \\ 0 &= C\xi_i - \alpha_i \quad \Rightarrow \quad \xi_i = \frac{\alpha_i}{C}, \quad i = 1, \ldots, n. \end{aligned}$$

Plugging the optimal w_α and ξ_i into $L(w, w_o, \xi, \alpha)$ we obtain the dual

$$\begin{aligned}\theta(\alpha) &= \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}^n \alpha_i\alpha_j y_i y_j x_i' x_j + \frac{1}{2C}\sum_{i=1}^n \alpha_i^2 - \frac{1}{C}\sum_{i=1}^n \alpha_i^2 \\ &= \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}^n \alpha_i\alpha_j y_i y_j x_i' x_j - \frac{1}{2C}\sum_{i=1}^n \alpha_i^2 \\ &= \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}^n \alpha_i\alpha_j y_i y_j \big(x_i' x_j + \frac{1}{C}\delta_{ij}\big),\end{aligned}$$

where δ_{ij} is Kronecker's delta. The dual problem is

$$\max \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j}^n \alpha_i\alpha_j y_i y_j \big(x_i' x_j + \frac{1}{C}\delta_{ij}\big) \tag{4.2.24}$$

$$\text{subject to } \alpha_i \geq 0, \quad \sum_{i=1}^n y_i\alpha_i = 0. \tag{4.2.25}$$

The obtained dual problem is almost the same as the dual problem in hard-margin case (4.2.7) except that the constants $\frac{1}{C}$ are added to the main diagonal of the matrix $(x_i' x_j)$.

KKT: $\alpha_i^* \geq 0$, $\xi_i^* = \frac{\alpha_i^*}{C}$ and

$$\sum_{i=1}^n \alpha_i^*\big(1 - \xi_i^* - y_i(w^{*\prime} x_i + w_o^*)\big) = 0.$$

Hence $\alpha_i^* > 0$ implies that $\xi_i^* > 0$ and

$$y_i(w^{*\prime} x_i + w_o^*) = 1 - \frac{\alpha_i^*}{C}.$$

Hence all support vectors (those x_i for which $\alpha_i > 0$) correspond to margin errors and the equality above can be used to determine the constant w_o^*.

Exercise. Prove that

$$\|w^*\|^2 = \sum_i \alpha_i^* - \frac{1}{C}\sum_i (\alpha_i^*)^2$$

so that the margin can be found as follows

$$\gamma = \Big(\sum_i \alpha_i^* - \frac{1}{C}\sum_i (\alpha_i^*)^2\Big)^{-\frac{1}{2}}.$$

Exercise. Generalize the problems (4.2.16) and (4.2.23) by defining a separate penalty constant C_i for every i.

Exercise. Find the dual problem of the primal problem

$$\min_{w,w_o,\xi} \frac{1}{2}\|w\|^2 + C_1\sum_{i=1}^n \xi_i + \frac{C_2}{2}\sum_{i=1}^n \xi_i^2$$
$$\text{subject to } y_i(w'x_i + w_o) \geq 1 - \xi_i \quad \xi_i \geq 0, \quad i = 1, \ldots, n.$$

4.2.3 Equivalent problems

Here we show the equivalence of (4.2.16) and (4.2.14). The similar relation between 2-norm problems (4.2.22) and (4.2.2) can be obtained similarly.

Recall the solutions of (4.2.16):

$$w^* = \sum_i \alpha_i^* y_i x_i, \quad w_o^* = -\frac{1}{2}(w^{*\prime}x_i + w^{*\prime}x_j), \quad \text{where } 0 < \alpha_i^*, \alpha_j^* < C \quad \text{and } y_i = 1, y_j = -1.$$

Here α^* is the solution of dual (4.2.19):

$$\max_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i' x_j,$$
$$\text{subject to } \sum_{i=1}^n \alpha_i y_i = 0$$
$$\alpha_i \in [0, C], \quad \forall i.$$

We shall show that to every C correspond constants D and $c > 0$ so that cw^* and cw_o^* are the solutions of (4.2.14):

$$\min_{u:\|u\|=1,b,\gamma\geq 0}\Big(-\gamma + D\sum_{i=1}^n \big(\gamma - y_i(u'x_i + b)\big)_+\Big).$$

Let $A = \sum_{i=1}^n \alpha_i^*$ and define $D := \frac{C}{A}$. With slack variables (4.2.14) is

$$\min_{\alpha,b,\gamma,\xi} \quad -\gamma + D\sum_{i=1}^n \xi_i$$
$$\text{subject to } y_i(u'x_i + b) \geq \gamma - \xi_i \quad \xi_i \geq 0, \quad i = 1, \ldots, n, \quad \|u\| = 1.$$

Lagrangian:

$$L(w, w_o, \gamma, \xi, \rho, \lambda) := -\gamma + D\sum_i \xi_i - \sum_i \beta_i\big(y_i(w'x_i - w_o) - \gamma + \xi_i\big) - \sum_i \rho_i\xi_i + \lambda\big(\|w\|^2 - 1\big),$$

where $\rho_i \geq 0, \beta_i \geq 0$. The dual problem is (see [11], ch 7.2)

$$\max_{\beta} \quad 1 - \frac{1}{2}\sum_{i,j}^{n} \beta_i \beta_j y_i y_j x_i' x_j, \tag{4.2.26}$$

$$\text{subject to} \sum_{i=1}^{n} \beta_i y_i = 0$$

$$\beta_i \in [0, D], \quad \forall i$$

$$\sum_{i=1}^{n} \beta_i = 1.$$

With the solutions β_i^* of the dual problem, the solution of the primal problem is

$$u^* = \frac{1}{2\lambda}\sum_i \beta_i^* y_i x_i, \quad \text{where } \lambda = \frac{1}{2}\sqrt{\sum_{i,j}^{n} \beta_i^* \beta_j^* y_i y_j x_i' x_j}$$

$$b^* = -\frac{1}{2}\big(u^{*\prime} x_j + u^{*\prime} x_i\big), \quad \text{where } 0 < \beta_i^*, \beta_j^* < D \quad \text{and } y_i = 1, y_j = -1.$$

Let us now show that the dual problems (4.2.19) and (4.2.26) are related as follows: $\frac{\alpha_i^*}{A}$ are the solutions of (4.2.26), i.e. $\frac{\alpha_i^*}{A} = \beta_i^*$. Clearly

$$\sum_i \frac{\alpha_i^*}{A} = 1, \quad \frac{\alpha_i^*}{A} \in [0, \frac{C}{A}]$$

so that the constraints are satisfied.
If $\frac{\alpha_i^*}{A}$ were not the solutions of (4.2.26), then for some $\beta_i \in [0, D]$ such that $\sum_i \beta_i = 1$, the following equality would hold:

$$-\frac{1}{2}\sum_{i,j}^{n} \beta_i \beta_j y_i y_j x_i' x_j > -\frac{1}{2A^2}\sum_{i,j}^{n} \alpha_i^* \alpha_j^* y_i y_j x_i' x_j.$$

Defining $\alpha_i := A\beta_i$, we would obtain that $\alpha_i \in [0, C]$ and

$$-\frac{1}{2}\sum_{i,j}^{n} \alpha_i \alpha_j y_i y_j x_i' x_j > -\frac{1}{2}\sum_{i,j}^{n} \alpha_i^* \alpha_j^* y_i y_j x_i' x_j,$$

so we would have a contradiction. Thus $\frac{\alpha_i^*}{A} = \beta_i^*$ so that

$$\frac{w^*}{A} = \sum_i \beta_i^* y_i x_i = 2\lambda u^*.$$

When $0 < \alpha_i^*, \alpha_j^* < C$ and $y_i = 1, y_j = -1$, then the corresponding $0 < \beta_i^*, \beta_j^* < D$. Hence

$$w_o^* = -\frac{1}{2}(w^{*\prime} x_i + w^{*\prime} x_j) = -2\lambda A \frac{1}{2}(u^{*\prime} x_i + u^{*\prime} x_j) = 2A\lambda b^*.$$

Thus
$$w^* = 2A\lambda u^*, \quad w_o^* = 2A\lambda b^*.$$
Since $\|u^*\| = 1$, we have that $\|w^*\| = 2A\lambda$. Recall that $\|w^*\|^{-1} = \gamma^*$, where γ^* is the optimal margin, so that $u^* = \gamma^* w^*$ and $b^* = \gamma^* w_o$.

The one-to-one correspondence. To show that to a D corresponds a C such that the solutions of (4.2.14) were proportional to the solutions (4.2.16), we could use the same argument, if there exists a C so that $D = \frac{C}{A}$ (recall that A depends on C). Let us calculate A. Since
$$\|w^*\|^2 = \sum_{i,j}^{n} \alpha_i^* \alpha_j^* y_i y_j x_i' x_j,$$
from the equality of dual and primal solution, we have
$$\theta(\alpha^*) = A - \frac{1}{2}\|w^*\|^2 = \frac{1}{2}\|w^*\|^2 + C\sum_i \xi_i^* \quad \Rightarrow \quad A = \|w^*\|^2 + C\sum_i \xi_i^*.$$
Hence
$$\frac{C}{A} = \frac{C}{\|w^*\|^2 + C\sum_i \xi_i^*}$$
and it is easy to verify that is non-decreasing and continuous in C. Hence there exists a limit $\bar{D} = \lim_{C\to\infty} \frac{C}{A}$ so that for $D < \bar{D}$, there exists a corresponding C meaning that in certain range of D the equivalence of two problems holds.

The constants C and D. Since $u^* = \gamma^* w^*$, $b^* = \gamma^* w_o$, it holds $(w^{*\prime} x_i + w_o^*) y_i = 1$ iff $(u^{*\prime} x_i + b^*) y_i = \gamma^*$. Since $\frac{\alpha_i^*}{A} = \beta_i^*$, from the properties of α_i^*, we obtain
$$\begin{aligned}(u^{*\prime} x_i + b^*) y_i > \gamma^* \Rightarrow \beta_i^* = 0 \Rightarrow& (w^{*\prime} x_i + b^*) y_i \geq \gamma^* \\ 0 < \beta_i^* < D \Rightarrow& (u^{*\prime} x_i + b^*) y_i = \gamma^* \\ (u^{*\prime} x_i + b^*) y_i < \gamma^* \Rightarrow \beta_i^* = D \Rightarrow& (u^{*\prime} x_i + b^*) y_i \leq \gamma^*.\end{aligned}$$
But, in addition $\sum_i \beta_i^* = 1$. This allows us to obtain a interpretation to the constat D. Indeed, since $\beta_i^* \leq D$, clearly it must hold $D \geq \frac{1}{n}$, otherwise the constraint $\sum_i \beta_i^* = 1$ cannot be satisfied. Often it is suggested to use $D = \frac{1}{\nu n}$, where $\nu \in (0, 1]$. In this case the parameter ν controls the proportion of bound support vectors: for bound support vectors $\beta_i^* = \frac{1}{\nu n}$ and $\sum_i \beta_i^* = 1$, it follows that there cannot be more than νn bound support vectors. Hence ν controls the number of margin errors. Similarly, it follows that there must be at least νn support vectors, because at least νn coefficients β_i^* must be positive. Hence, choosing D properly helps us to control the proportion of margin errors and support vectors.
The constant C does not have such a nice interpretation: although $C = DA$, we do not know the value of A in advance. In practice, however, choosing C is an important issue and typically several cross-validation methods are used.

4.3 Some risk bounds for large margin classifiers

Functional margin. Recall the linear discriminant:

$$g(x) = \mathrm{sgn}\big(f(x)\big) = \begin{cases} 1, & \text{if } f(x) \geq 0; \\ 0, & \text{if } f(x) < 0. \end{cases} \tag{4.3.1}$$

where $f(x) = w'x + w_0$ is a linear function. If $\|w\| = 1$ and g classifies x_i correctly, then $y_i f(x_i)$ (that is always positive) is geometric margin of (x_i, y_i). Let us generalize the concept of geometrical margin as follows.

Let $f : \mathbb{R}^d \to \mathbb{R}$ be a function and define a classifier g like (4.3.1). Hence g classifies (x_i, y_i) correctly if $y_i f(x_i) > 0$ and incorrectly if $y_i f(x_i) < 0$. The number $y_i f(x_i)$ is called **functional margin** of (x_i, y_i).

The risk and empirical risk of g as in (4.3.1) are

$$R(g) = \mathbf{P}(Y \neq \mathrm{sgn}(f(X)) \leq EI_{\{f(X)Y \leq 0\}}, \quad R_n(g) = \frac{1}{n}\sum_{i=1}^n I_{\{\mathrm{sgn} f(x_i) \neq y_i\}} \leq \frac{1}{n}\sum_{i=1}^n I_{\{y_i f(x_i) \leq 0\}}.$$

The quantity $A_n(f)$. Let, for every f

$$A_n(f) = \frac{1}{n}\sum_{i=1}^n \phi(-f(x_i)y_i), \quad \phi(z) = \begin{cases} 1, & \text{if } z \geq 0; \\ 1 + \frac{z}{\gamma}, & \text{if } -\gamma \leq z \leq 0; \\ 0, & \text{if } z < -\gamma \end{cases}$$

The quantity $A_n(f)$ can be estimated above as follows:

- From $\phi(z) \leq I_{(-\gamma,\infty)}(z)$, we obtain that

$$A_n(f) \leq \frac{1}{n}\sum_{i=1}^n I_{\{-f(x_i)y_i > -\gamma\}} = \frac{1}{n}\sum_{i=1}^n I_{\{f(x_i)y_i < \gamma\}} =: R_n^\gamma(f).$$

  The number R_n^γ is the proportion of margin errors. Recall that a pair (x_i, y_i) counts as a margin error even if $\mathrm{sgn} f(x_i) = y_i$ (i.e. $f(x_i)y_i > 0$) but the "confidence of classification" $f(x_i)y_i$ is below γ.

- On the other hand,

$$\phi(z) \leq (1 + \frac{z}{\gamma})_+, \quad \text{where} \quad (x)_+ = \begin{cases} x, & \text{if } x > 0; \\ 0, & \text{if } x \leq 0. \end{cases}$$

  Hence,

$$A_n(f) \leq \frac{1}{n}\sum_{i=1}^n (1 - \frac{y_i f(x_i)}{\gamma})_+ = \frac{1}{n\gamma}\sum_{i=1}^n (\gamma - y_i f(x_i))_+ = \frac{1}{n\gamma}\sum_{i=1}^n \xi_i,$$

  where

$$\xi_i := (\gamma - y_i f(x_i))_+ \geq 0.$$

  The number ξ_i, as we already know, measures the size of margin error.

Margin bounds for linear functions. Recall (4.2.9):

$$w^* = \sum_{i \in \mathrm{SV}} \alpha_i^* y_i x_i.$$

Let us consider the set of linear functions

$$\mathcal{F}_1 := \big\{ f(x) = w^{*\prime} x : \quad \|w\| \leq 1 \big\}.$$

Hence, the linear function $x \mapsto w^{*\prime} x$ with $\|w^*\| \leq 1$ belongs to $\mathcal{F}_1$.

Let $\gamma > 0$ be fixed and let $f_n \in \mathcal{F}_1$ be chosen using the data $(X_1, Y_1), \ldots, (X_n, Y_n)$ (when dealing with risk bounds, the sample is considered as random). Let $g_n = \mathrm{sgn} f_n$ be the corresponding classifier. Then the following risk bound holds ([11], p. 103): with probability at least $1 - \delta$

$$R(g_n) \leq A_n(f_n) + \frac{4}{n\gamma} \sqrt{\sum_{i=1}^n X_i' X_i} + 3 \sqrt{\frac{\ln \frac{2}{\delta}}{2n}}. \tag{4.3.2}$$

Since $A_n(f) \leq R_n^\gamma(g)$, (4.3.2) yields (see also [3] Cor 4.3): with probability $1 - \delta$,

$$R(g_n) \leq R_n^\gamma(g_n) + \frac{4}{n\gamma} \sqrt{\sum_{i=1}^n X_i' X_i} + 3 \sqrt{\frac{\ln \frac{2}{\delta}}{2n}}. \tag{4.3.3}$$

This bound justifies maximizing margin and minimizing the number of margin errors, simultaneously.

The estimate

$$A_n(f) \leq \frac{1}{n\gamma} \sum_{i=1}^n \xi_i$$

together with (4.3.2) yields another bound suitable for estimating the risk of 1-norm soft margin classifiers ([11], Thm 4.17): with probability at least $1 - \delta$,

$$R(g_n) \leq \frac{1}{n\gamma} \sum_{i=1}^n \xi_i + \frac{4}{n\gamma} \sqrt{\sum_{i=1}^n X_i' X_i} + 3 \sqrt{\frac{\ln \frac{2}{\delta}}{2n}}. \tag{4.3.4}$$

This bound justifies to maximize margin and minimize the sum of margin errors just like in 1-norm soft margin problems.

Using ϕ^2 instead of ϕ, be similar argument as above, on can obtain a bound suitable for estimating the risk of 2-norm soft margin classifiers ([11], (7.13)): with probability at least $1 - \delta$,

$$R(g_n) \leq \frac{1}{n\gamma^2} \sum_{i=1}^n \xi_i^2 + \frac{8}{n\gamma} \sqrt{\sum_{i=1}^n X_i' X_i} + 3 \sqrt{\frac{\ln \frac{2}{\delta}}{2n}}. \tag{4.3.5}$$

This bound justifies to maximize margin and minimize the sum of margin errors just like in 2-norm soft margin problems.

Remarks:

1 Note that obtained bounds hold for linear discriminants without constants, because the set $\mathcal{F}_1$ consists of linear (not affine) functions, only. That is no essential restriction, since the constants can be captured into feature vector by increasing the dimension by one and new feature s have dimension $d+1$ instead of d. But then the condition $\|w\| \leq 1$ should be replaced by $\|w\|^2 + w_o^2 \leq 1$ and $\sqrt{\sum_{i=1}^n X_i' X_i}$ should be replaced by $\sqrt{\sum_{i=1}^n X_i' X_i + 1}$. However, formally the risk bounds above cannot be used for exact risk bounds of support vector classifiers, since they contain w_o.

2 All the bounds above are valid for fixed γ, only. Hence they are not uniform bounds and, therefore, they do not apply when the (optimal) margin has been chosen using the data as it is done in support vector classification. Uniform bound would have an additional term. However, although recognized (see Remark 7.17) these two deficiencies (no constants and uniformity) are disregarded in [11] and the bounds (4.3.4) and (4.3.5) are stated for as valid bonds 1-norm and 2-norm support vector classifiers, respectively.

3 Another class of large margin bounds are based on the generalization of VC-dimension for taking account the margin – so-called *fat-shattering dimension* (the number of maximal set of points that can be shattered with given margin). If all the points are in a ball of radius $r < \infty$, then the fat-shattering dimension can be smaller than usual VC-dimension. Hence these bounds are useful under additional assumption that the feature vector has bounded support, i.e. for a $r < \infty$, $\mathbf{P}(\|X\| \leq r) = 1$. These bounds can be found, for example, in [5], Ch 10, [10], Ch. 4.

4.4 (Non-linear) support vector machines

So far, we have considered linear discriminants. The idea of many non-linear classification methods is to transform the data non-linearly into a space, where linear discriminants can be applied. Applying this idea for the support vector classification described in the last section gives us so-called **support vector machine (SVM) classifiers**. Thus, let

$$\Phi : \mathbb{R}^d \to \mathcal{H}$$

be a mapping from the feature space into another space $\mathcal{H}$. In space $\mathcal{H}$, the transformed sample is

$$(\Phi(x_1), y_1), \ldots, (\Phi(x_n), y_n)$$

and now, in principle, every linear discriminant can be applied. The resulting classifier in original feature space $\mathbb{R}^d$ is then typically non-linear.

The idea of projecting the data into a space that is more convenient for linear analysis is very old (recall LDA), but unlike classical multivariate analysis, in SVM classification the space $\mathcal{H}$ is not a low-dimensional subspace, but typically very high, possible infinite-dimensional. At the first sight, such an approach might seem useless, since working in a high dimensional space is typically much more difficult as working in the original space. It turns out that for a large class of transformations Φ, the high dimensionality of $\mathcal{H}$ does not cause additional computational difficulties.

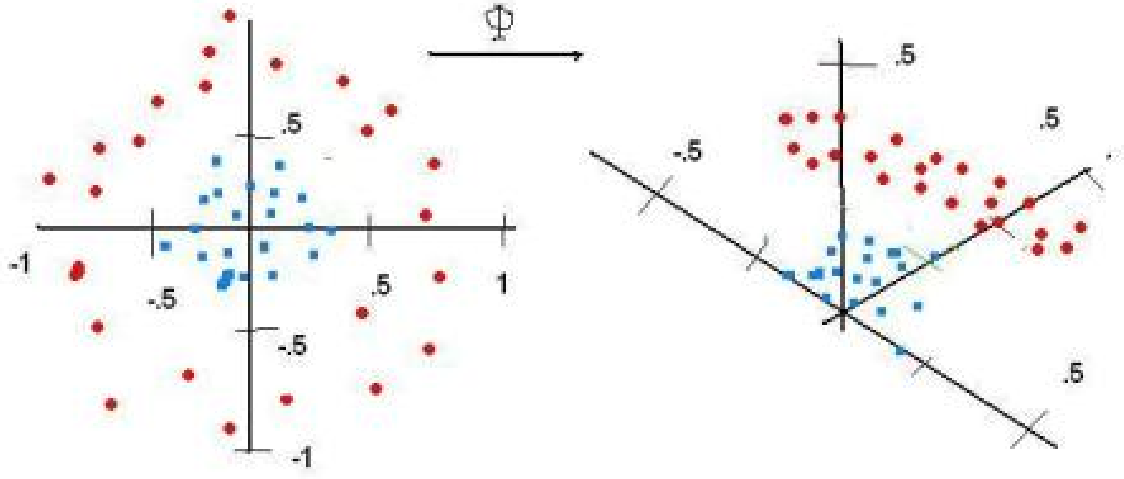
Example. Let

$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3, \quad \Phi(x) = \Phi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) = (z_1, z_2, z_3).$$

Every hyperspace $H = \{z : w'z + w_0 = 0\}$ in $\mathbb{R}^3$ is a second order curve

$$\{(x^1, x^2) : w^1x_1^2 + w^2\sqrt{2}x_1x_2 + w^3x_2^2 + w_0 = 0\}$$

in $\mathbb{R}^2$ so that the corresponding classifier in is non-linear. The figure illustrates the case when in the original space $\mathbb{R}^2$ the data are not linearly separable, but after transformation into $\mathbb{R}^3$, they are linearly separable.



In SVM, the space $\mathcal{H}$ is a Hilbert space (complete inner product (dot product) space). Every hyperplane in $\mathcal{H}$ is then

$$\{z \in \mathcal{H} : \langle w, z\rangle + w_0 = 0\}, \qquad \text{where } w \in \mathcal{H}, \quad w_0 \in \mathbb{R}^1.$$

Applying a linear discriminant to transformed data $\Phi(x_1), \ldots, \Phi(x_n)$, we obtain a non-linear classifier in $\mathbb{R}^d$:

$$g(x) = \operatorname{sgn}(\langle w, \Phi(x)\rangle + w_0). \tag{4.4.1}$$

1 and 2-norm soft margin SVM. Applying the support vector classifiers from the last section, we get the following optimization problems:

$$\min_{w\in\mathcal{H},w_o\in\mathbb{R}} \frac{1}{2}\langle w,w\rangle + C\sum_{i=1}^{n}\big(1-y_i(\langle w,\Phi(x_i)\rangle+w_o)\big)_+$$

with the corresponding dual

$$\max_{\alpha\in\mathbb{R}^n}\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}^{n}\alpha_i\alpha_j y_i y_j\langle\Phi(x_i),\Phi(x_j)\rangle$$

$$\text{subject to } C\geq\alpha_i\geq 0,\quad \sum_{i=1}^{n}y_i\alpha_i=0$$

(1-norm soft margin), or

$$\min_{w\in\mathcal{H},w_o\in\mathbb{R}} \frac{1}{2}\langle w,w\rangle + \frac{C}{2}\sum_{i=1}^{n}\big(1-y_i(\langle w,\Phi(x_i)\rangle+w_o)\big)_+^2$$

with the corresponding dual

$$\max_{\alpha\in\mathbb{R}^n}\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i,j}^{n}\alpha_i\alpha_j y_i y_j\big(\langle\Phi(x_i),\Phi(x_j)\rangle+\frac{1}{C}\delta_{ij}\big)$$

$$\text{subject to } \alpha_i\geq 0,\quad \sum_{i=1}^{n}y_i\alpha_i=0$$

(2-norm soft margin).

Using the solutions α^* of dual problems, the optimal solution for both problems is

$$w^*=\sum_{i=1}^{n}\alpha_i^* y_i\Phi(x_i).$$

For 1-norm soft margin, the constant w_0^* will be determined from the equality

$$y_i(\langle w^*,\Phi(x_i)\rangle+w_0^*)=y_i\big(\sum_{j=1}^{n}\alpha_j^* y_j\langle\Phi(x_j),\Phi(x_i)\rangle+w_o^*\big)=1,$$

where x_i is a in bound support vector, i.e. $0<\alpha_i^*<C$.
For 2-norm soft margin problem, the constant w_o^* will be determined from the equation

$$1-\frac{\alpha_i^*}{C}=y_i(\langle w^*,\Phi(x_i)\rangle+w_0^*)=y_i\big(\sum_{j=1}^{n}\alpha_j^* y_j\langle\Phi(x_j),\Phi(x_i)\rangle+w_o^*\big),$$

where x_i is a support vector, i.e. $\alpha_i^* > 0$.

The classifier (4.4.1) is thus

$$g(x) = \operatorname{sgn}\big(\langle w^*, \Phi(x)\rangle + w_o^*\big) = \operatorname{sgn}\big(\sum_{i=1}^{n} \alpha_i^* y_i \langle \Phi(x_i), \Phi(x)\rangle + w_o^*\big). \tag{4.4.2}$$

Remark. If $\mathcal{H}$ is a sequence space or $\mathbb{R}^m$, then it can incorporate also the constant. Hence the discriminative hyperplane in $\mathcal{H}$ can be defined without constant and the classifier (4.4.1) is then $g(x) = \operatorname{sgn}\big(\langle w^*, \Phi(x)\rangle\big)$. For example, if Φ is as in the example above, i.e. $\Phi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$, then

$$\langle w, \Phi(x)\rangle + w_o = \langle u, \Phi^*(x)\rangle,$$

where

$$\Phi^*(x^1, x^2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2, 1), \quad u' = (w', w_o) \in \mathbb{R}^4.$$

4.5 Kernel

4.5.1 Kernel trick

Note: the optimal classifier (4.4.2) as well as the coefficients α_i^* depend on Φ via the product $\langle \Phi(x), \Phi(y)\rangle$ only. Hence in order to find and use (4.4.2) it suffices to know the function:

$$K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}, \quad K(x, y) := \langle \Phi(x), \Phi(y)\rangle. \tag{4.5.1}$$

The function $K(x, y)$ is known as **kernel** and typically $K(x, y)$ can be calculated directly in space $\mathbb{R}^d$ without applying Φ. This means that all computations for applying SVM cold be done using d-dimensional feature vectors but not very high dimensional elements in $\mathcal{H}$. This so-called **kernel trick** makes SVM's possible.

Example. In the previous example,

$$K(x, y) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)\begin{pmatrix} y_1^2 \\ \sqrt{2}y_1y_2 \\ y_2^2 \end{pmatrix} = (x_1y_1 + x_2y_2)^2 = (x'y)^2.$$

Hence, finding $K(x, y)$ is relatively easy and the could be done without applying Φ. Note that different mappings Φ can define the same kernel. For example, the same kernel $K(x, y) = (x'y)^2$ is defined by mapping

$$\Phi' : \mathbb{R}^2 \to \mathbb{R}^4, \quad \Phi'(x_1, x_2) = (x_1^2, x_1x_2, x_1x_2, x_2^2).$$

With kernel K, the classifier (4.4.1) is

$$g(x) = \operatorname{sign}\big(\sum_{i \in SV} \alpha_i^* y_i K(x_i, x) + w_o^*\big) \tag{4.5.2}$$

where α_i^* are the solutions of dual problem. With kernels, the 1-norm soft margin dual problem (4.2.7) is as follows

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j).$$

Now it is clear, why in SVM-classification the main object of interest is not the mapping Φ, rather than the kernel K. Different kernels define different classifiers, hence choosing the kernel means choosing the class of classifiers $\mathcal{G}$. After choosing the kernel – the class $\mathcal{G}$ – SVM picks a classifier $g_n \in \mathcal{G}$ that in a sense is the best.

Risk bounds. The risk bound from Section 4.3 hold also for non-linear SVM's. The space $\mathcal{F}_1$ is now

$$\mathcal{F}_1 = \Big\{ f(x) = \langle w, \Phi(x) \rangle : \quad w \in \mathcal{H}, \quad \|w\| \leq 1 \Big\}$$

and for every classifier

$$g_n(x) = \operatorname{sgn}\big(\langle w, \Phi(x) \rangle\big) \quad \text{such that } \langle \Phi(x), \Phi(x) \rangle = \|w\|^2 \leq 1$$

and *fixed* γ the bound (4.3.4) is: with probability $1 - \delta$

$$R(g_n) \leq \frac{1}{n\gamma} \sum_{i=1}^n \xi_i + \frac{4}{n\gamma} \sqrt{\sum_{i=1}^n K(X_i, X_i)} + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2n}}.$$

The bound (4.3.5) is: with probability $1 - \delta$

$$R(g_n) \leq \frac{1}{n\gamma^2} \sum_{i=1}^n \xi_i^2 + \frac{8}{n\gamma} \sqrt{\sum_{i=1}^n K(X_i, X_i)} + 3\sqrt{\frac{\ln \frac{2}{\delta}}{2n}}.$$

4.5.2 How to recognize a kernel?

Replacing Φ by the kernel seems like a reasonable ida, but how do we know that a function $K(x, y)$ is a kernel? Formally kernel is defined via Φ, but would it be possible to decide whether a function $K(x, y)$ is a kernel without knowing or Φ. Clearly a kernel is symmetric function, i.e. $K(x, y) = K(y, x)$. Also it clear that for any $x_1, \ldots, x_n$ in $\mathbb{R}^d$, the **Gram matrix**:

$$\big(K(x_i, x_j)\big)_{i,j}$$

must be positively semi-definite.

Exercise: Prove that $\big(K(x_i, x_j)\big)_{i,j}$ is positively semi-definite.

Definition 4.5.1 *Let $\mathcal{X}$ be arbitrary. The function*

$$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

is positively (semi)-definite, if it is symmetric and for every finite set $\{x_1, \dots, x_n\} \subset \mathcal{X}$ the corresponding Gram matrix is positively (semi)-definite.

Thus, when K is a kernel, i.e. $\exists\ \Phi : \mathcal{X} \to \mathcal{H}$ such that $K(x,y) = \langle \Phi(x), \Phi(y) \rangle$, then it is positively semi-definite. The following theorem states that positive semi-definiteness is also sufficient for being a kernel.

Theorem 4.5.2 (Moore-Aronszajn, 1950) *Let $\mathcal{X}$ be arbitrary not empty set. A function*

$$K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$$

is a kernel if and only is it is positively semi-definite.

Tho only if part is obvious.

The idea of proof. At fist show the existence of a inner product space $\mathcal{F}$ and mapping

$$\Phi : \mathcal{X} \to \mathcal{F} \text{ such that } \langle \Phi(x), \Phi(y) \rangle = K(x,y).$$

Then the space is completed. Then there exist a *Hilbert space* $\mathcal{H}$ so that

$$\Phi : \mathcal{X} \to \mathcal{H} \text{ and } \langle \Phi(x), \Phi(y) \rangle = K(x,y).$$

The construction. Let K be a positive semi-definite function. Define the mapping

$$\Phi : \mathcal{X} \to \mathbb{R}^{\mathcal{X}}, \quad \Phi(x) = K(x, \cdot). \tag{4.5.3}$$

Let us define a vector space $\mathcal{F}$ as follows

$$\mathcal{F} = \text{span}\{K(x,\cdot) : x \in \mathcal{X}\} = \Big\{\sum_{i=1}^{m} \alpha_i K(x_i, \cdot) : x_1, \dots, x_m, m \in \mathbb{N}, \alpha_i \in \mathbb{R}\Big\}. \tag{4.5.4}$$

Now we define a inner product in $\mathcal{F}$ as follows. Let

$$f = \sum_{i=1}^{m} \alpha_i K(x_i, \cdot), \quad g = \sum_{i=1}^{n} \beta_i K(y_i, \cdot)$$

and

$$\langle f, g \rangle = \Big\langle \sum_{i=1}^{m} \alpha_i K(x_i, \cdot), \sum_{i=1}^{n} \beta_i K(y_i, \cdot) \Big\rangle := \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha_i \beta_j K(x_i, y_j). \tag{4.5.5}$$

To see that $\langle \cdot, \cdot \rangle$ is a inner product, one has to make sure that it is properly defined and satisfies all the axioms of a inner product. The correctness of the definition means that if the representation of f or g is not unique, then $\langle f,g\rangle$ is always the same. To see that note

$$\langle f,g\rangle = \sum_{i=1}^{m}\sum_{j=1}^{n}\alpha_i\beta_j K(x_i,y_j) = \sum_{i=1}^{m}\alpha_i g(x_i) = \sum_{j=1}^{n}\beta_j f(y_j). \tag{4.5.6}$$

Hence, if there exists m' and α'_i such that $f=\sum_{i=1}^{m'}\alpha'_i K(x'_i,\cdot)$, then the $\sum_{j=1}^{n}\beta_j f(y_j)$ will remain unchanged. Hence $\langle f,g\rangle$ is correctly defined.
The axioms of a inner product are:

1. $\langle f,f\rangle \geq 0$
2. $\langle f,g\rangle = \langle g,f\rangle$
3. $\langle f+h,g\rangle = \langle f,g\rangle + \langle h,g\rangle$
4. $\langle \lambda f,g\rangle = \lambda\langle f,g\rangle$
5. $\langle f,f\rangle = 0 \Rightarrow f=0.$

Exercise: Prove the axioms $1-4$.

The axioms $1-4$ state that $\langle f,g\rangle$ is a semi inner product. Then Cauchy-Schwartz inequality holds:

$$\langle f,g\rangle^2 \leq \langle f,f\rangle\langle g,g\rangle. \tag{4.5.7}$$

To prove the last axiom, note that for every f and x

$$\langle f,K(x,\cdot)\rangle = \langle\sum_{i=1}^{m}\alpha_i K(x_i,\cdot),K(x,\cdot)\rangle = \sum_{i=1}^{m}\alpha_i K(x_i,x) = f(x). \tag{4.5.8}$$

Thus, if $\langle f,f\rangle = 0$, then for every $x\in\mathcal{X}$,

$$f^2(x) = |\langle K(x,\cdot),f\rangle|^2 \leq \langle K(x,\cdot),K(x,\cdot)\rangle\langle f,f\rangle = K(x,x)\langle f,f\rangle = 0,$$

so that semi inner product is an inner product.

Hence the first part of the proof is completed: we have defined an inner product space $\mathcal{F}$ and a mapping $\Phi:\mathcal{X}\to\mathcal{F}$. An inner product space is a normed vector space where norm is $\|f\|=\sqrt{\langle f,f\rangle}$ and Cauchy-Schwartz inequality is

$$|\langle f,g\rangle| \leq \|f\|\|g\|.$$

The space $\mathcal{F}$ can be completed so that all properties hold. Thus there exists a Hilbert space $\mathcal{H}$ of functions $f:\mathcal{X}\to\mathbb{R}$ so that $K(x,\cdot)\in\mathcal{H}$ for every x and, most importantly, the property (4.5.8) holds: for every $f\in\mathcal{H}$

$$\langle K(x,\cdot),f\rangle = f(x).$$

The obtained space $\mathcal{H}$ is called **reproducing kernel Hilbert space (RKHS)**.

Definition 4.5.3 *A Hilbert space $\mathcal{H}$ of functions on a set $\mathcal{X}$ is called a reproducing kernel Hilbert space, if there exists a* **reproducing kernel** $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *such that the following conditions are satisfied:*

i) *for every $x \in \mathcal{X}$, $K(x, \cdot) \in \mathcal{H}$;*

ii) *the reproducing property (4.5.8) holds.*

The following exercise shows that the reproducing kernel in the sense of above-stated definition is the kernel in our sense, i.e. it is positively semi-definite.

Exercise: Let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric function so that for every x, it holds that $K(x, \cdot) \in \mathcal{F}$, where $\mathcal{F}$ is an inner product space of functions. Prove that if (4.5.8) holds, then K is positively semi-definite (hence symmetric).

The following exercise shows that if a Hilbert space of functions $\mathcal{H}$ admits a reproducing kernel, then it is unique.

Exercise: Let K and K' be two reproducing kernels of a Hilbert space of functions. Show that $K = K'$, i.e. for every $x, y \in \mathcal{X}$, $K(x, y) = K'(x, y)$.

The next exercise gives a nice characterization of a Hilbert space of functions to be RKHS.

Exercise: Using Riesz Representation theorem prove that a Hilbert space $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ is RKHS iff the functional $f \mapsto f(x)$ is continuous.

A bounded linear functional is continuous. Hence there is another commonly used definition of RKHS: a Hilbert space of functions $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ is RKHS, if, for every $x \in \mathcal{X}$ the pointwise evaluation $f \mapsto f(x)$ is bounded.

Exercise: Are the following Hilbert spaces RKHS: $L_2[0, 1]$? l_2? $\mathbb{R}^d$? If yes, find the reproducing kernel.

To summarize: for any kernel K (positive semi-definite function), there exists a Hilbert space of functions $\mathcal{H}$ so that K is reproducing kernel of $\mathcal{H}$ (and $\mathcal{H}$ is therefore RKHS). Then, for any $x \in \mathcal{X}$, $K(x, \cdot) \in \mathcal{H}$ and $K(x, y) = \langle K(x, \cdot), K(y, \cdot) \rangle$. Hence, we can (and we shall do so) always take $\Phi : \mathcal{X} \to \mathcal{H}$, $\Phi(x) = K(x, \cdot)$. Then any element $w \in \mathcal{H}$ is a function and the reproducing property (4.5.8) is

$$\langle \Phi(x), w \rangle = w(x). \tag{4.5.9}$$

In the next subsection we see that using RKHS is beneficial in many ways.

4.5.3 The Representer theorem

Recall SVM classification: an element $w \in \mathcal{H}$ was searched to minimize the objective

$$\min_{w \in \mathcal{H}, w_o \in \mathbb{R}} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\big(1 - y_i(\langle w, \Phi(x_i)\rangle + w_o)\big)_+^p, \tag{4.5.10}$$

where $p = 1$ or $p = 2$. And we saw that the optimal w^* was always in the form

$$w^* = \sum_{i=1}^{n}(\alpha_i^* y_i)\Phi(x_i).$$

In the following we shall see that the property that the solution of an optimization problem is a linear combination of the (support) vectors holds for a rather large class of problems.

At first note that solving (4.5.10) can be done in two parts: fix a constant w_o and then minimize over all vectors $w \in \mathcal{H}$; finally minimize over w_o. Hence, let us fix w_o and consider the problem

$$\min_{w} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\big(1 - y_i(\langle w, \Phi(x_i)\rangle + w_o)\big)_+^p, \tag{4.5.11}$$

Let now $\mathcal{H}$ be RKHS. Hence w is a function and the reproducing property (4.5.9) holds: $\langle w, \Phi(x_i)\rangle = w(x_i)$. Thus

$$\big(1 - y_i(\langle w, \Phi(x_i)\rangle + w_o)\big)_+^p = \big(1 - y_i(w(x_i) + w_o)\big)_+^p$$

so that (4.5.11) is as follows

$$\min_{w \in \mathcal{H}} \frac{1}{2}\|w\|^2 + L(w(x_1), \ldots, w(x_n)), \tag{4.5.12}$$

where $L : \mathbb{R}^n \to \mathbb{R}$.

Theorem 4.5.4 (Representer Theorem) *Let $\mathcal{X}$ be a set, let $\mathcal{H}$ be a RKHS and let K be the corresponding kernel. For every function $L : \mathbb{R}^n \to \mathbb{R}$ and strictly monotonic increasing function $\Omega : \mathbb{R} \to \mathbb{R}$ every minimizer of the problem*

$$\min_{w \in \mathcal{H}} \Omega(\|w\|^2) + L(w(x_1), \ldots, w(x_n)), \tag{4.5.13}$$

admits a reperesentation of the form

$$w = \sum_{i=1}^{n} \alpha_i K(x_i, \cdot) = \sum_{i=1}^{n} \alpha_i \Phi(x_i), \tag{4.5.14}$$

where $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$.

Thus, for every fixed w_o, the solution of (4.5.11) is in the form (4.5.14). Hence the overall minimizer w^* (corresponding to the best w_o has the similar representation.

Proof. Define

$$\mathcal{H}_{\|} = \mathrm{span}\big\{K(x_i, \cdot), i = 1, \ldots, n\big\}.$$

Hence $\mathcal{H}_{\|}$ is a subspace of functions $\sum_{i=1}^n \alpha_i K(x_i, \cdot)$. Every $w \in \mathcal{H}$ has an unique decomposition

$$w = w_{\|} + w_{\perp},$$

where $w_{\|} \in \mathcal{H}_{\|}$ and $w_{\perp}$ belongs to the orthogonal complement of $\mathcal{H}_{\|}$. Since, for every j, $K(x_j, \cdot) \in \mathcal{H}_{\|}$, it holds that for every j

$$w_{\perp}(x_j) = \langle w_{\perp}, K(x_j, \cdot)\rangle = 0,$$

implying that

$$w(x_j) = w_{\|}(x_j) + w_{\perp}(x_j) = w_{\|}(x_j)\Big(= \sum_{i=1}^n \alpha_i K(x_j, x_i)\Big).$$

Thus, for every w,

$$L(w(x_1), \ldots, w(x_n)) = L(w_{\|}(x_1), \ldots, w_{\|}(x_n))$$

implying that minimizing $L(w(x_1), \ldots, w(x_n))$ over $\mathcal{H}_{\|}$ is the same as minimizing it over $\mathcal{H}$. Since Ω is increasing, it holds that

$$\Omega(\|w\|^2) = \Omega(\|w_{\|}\|^2 + \|w_{\perp}\|^2) \geq \Omega(\|w_{\|}\|^2).$$

Therefore, minimizing $\Omega(\|w\|^2) + L(w(x_1), \ldots, w(x_n))$ over $\mathcal{H}$ is equivalent to minimizing it over $\mathcal{H}_{\|}$. ■

Hence, if $\mathcal{H}$ is RKHS, then the solution of (4.5.13) can be searched from the set

$$\mathrm{span}\Big\{\Phi(x_i), \quad i = 1, \ldots, n\Big\} = \Big\{\sum_{i=1}^n c_i \Phi(x_i), \quad c_i \in \mathbb{R}\Big\},$$

and the optimization problem is n-dimensional even if the space $\mathcal{H}$ is infinite-dimensional.

Example. Let us consider 1-norm soft margin problem (4.5.10) with $p = 1$:

$$\min_{w \in \mathcal{H}, w_o \in \mathbb{R}} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^n \big(1 - y_i(\langle w, \Phi(x_i)\rangle + w_o)\big)_+. \tag{4.5.15}$$

Due to Representer theorem, we search the solution in the form $w = \sum_{i=1}^n c_i \Phi(x_i)$. Hence

$$\|w\|^2 = \langle \sum_{i=1}^n c_i \Phi(x_i), \sum_{i=1}^n c_i \Phi(x_i)\rangle = \sum_{i,j=1}^n c_i c_j \langle \sum_{i=1}^n \Phi(x_i), \Phi(x_i)\rangle = \sum_{i,j=1}^n c_i c_j K(x_i, x_j) = c'Kc,$$

where K is Gram matrix and $c = (c_1, \dots, c_n)'$ is the vector of unknown coefficients. Since for every $i = 1, \dots, n$

$$\langle w, \Phi(x_i)\rangle = \langle \sum_{j=1}^{n} c_j \Phi(x_j), \Phi(x_i)\rangle = \sum_{j=1}^{n} c_j K(x_j, x_i) = \sum_{j=1}^{n} K(x_i, x_j) c_j,$$

the optimization problem is now

$$\min_{c \in \mathbb{R}^n, w_o \in \mathbb{R}} \frac{1}{2} c'Kc + C \sum_{i=1}^{n} \big(1 - y_i \sum_{j=1}^{n} K(x_i, x_j) c_j + w_o)\big)_+$$

and with help of slack variables, the problem is

$$\min_{c \in \mathbb{R}^n, w_o \in \mathbb{R}} \frac{1}{2} c'Kc + C \sum_{i=1}^{n} \xi_i$$

$$\text{subject to } y_i \big(\sum_{j=1}^{n} K(x_i, x_j) c_j + w_o\big) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \dots, n.$$

Lagrangian:

$$\begin{aligned} L(c, \xi, \alpha, \gamma) &= \frac{1}{2} c'Kc + C \sum_{i=1}^{n} \xi_i + \sum_i \alpha_i \Big(1 - \xi_i - y_i \big(\sum_{j=1}^{n} K(x_i, x_j) c_j + w_o\big)\Big) - \sum_i \gamma_i \xi_i. \\ &= \frac{1}{2} c'Kc - u'Kc + \sum_{i=1} \xi_i (C - \alpha_i - \gamma_i) + \sum_i \alpha_i, \end{aligned}$$

where $u = (u_1, \dots, u_n)$, $u_i = y_i \alpha_i$.
The gradient

$$\nabla_c L(c, \xi, \alpha, \gamma) = Kc - Ku, \quad \Rightarrow \quad \nabla_c L(c, \xi, \alpha, \gamma) = 0 \quad \Leftrightarrow \quad K(c - u) = 0.$$

Hence (recall K is symmetric) $c'Kc = c'Ku = u'Kc = u'Ku$. Setting the derivatives with respect to ξ and w_o equal to zero as well, we get the usual conditions

$$\sum_i y_i \alpha_i = 0, \quad \alpha_i + \gamma_i = C, \quad i = 1, \dots, n.$$

Hence

$$\theta(\alpha, \gamma) = \frac{1}{2} u'Ku - u'Kc + \sum_i \alpha_i.$$

With $u'K = c'K$, we get the familiar dual problem:

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} u'Ku = \max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$\text{subject to } \sum_i y_i \alpha_i = 0, \quad C \geq \alpha_i \geq 0, \quad i = 1, \dots, n.$$

With α^* being the solution of dual problem, we get that the solution of primal problem, let it be c^*, must satisfy the equality $Kc^* = Ku^*$, where $u^* = (y_1\alpha_1^*, \ldots, y_n\alpha_n^*)$. If K^{-1} exists, then it follows immediately that $c^* = u^*$ and the solution of (4.5.15) is again

$$w^* = \sum_i y_i\alpha_i^*\Phi(x_i). \tag{4.5.16}$$

In general, c^* should be such that $K(c^* - u^*) = 0$. It means that the sum $\sum_i(c_i^* - u^*)\Phi(x_i)$ must satisfy the conditions

$$\sum_i K(x_i, x_j)(c_i^* - u_i^*) = \langle \sum_i (c_i^* - u_i^*)\Phi(x_i), \Phi(x_j)\rangle = 0, \quad j = 1, \ldots, n.$$

Hence the vector $\sum_i(c_i^* - u^*)\Phi(x_i)$ must be orthogonal to every element in the space $\text{span}\{\Phi(x_j) : \ j = 1, \ldots, n\}$. Since $\sum_i(c_i^* - u^*)\Phi(x_i)$ belongs to the same space, it means that

$$\sum_i (c_i^* - u^*)\Phi(x_i) = 0, \quad \Rightarrow \quad w^* = \sum_i c_i^*\Phi(x_i) = \sum_i y_i\alpha_i^*\Phi(x_i).$$

Hence the solution of (4.5.15) is again (4.5.16) even if $c^* \neq u^*$.

References: For several generalization of the representer theorem, see [12], Ch 4.

4.5.4 The properties of kernels

Let $\mathcal{X}$ is an arbitrary set. Recall that every kernel is a positively semidefinite mapping on $\mathcal{X} \times \mathcal{X}$. The following lemma shows that the set of kernels is closed under many algebraic operations. These result help to construct kernels.

Lemma 4.5.1 *Let K_1 and K_2 be kernels. Then the following functions are also kernels:*

1. $K(x, y) = K_1(x, y) + K_2(x, y)$;
2. $K(x, y) = aK_1(x, y)$, $a \in \mathbb{R}^+$;
3. $K(x, y) = g(x)g(y)$, $g : \mathcal{X} \to \mathbb{R}$;
4. $K(x, y) = x'Ay$, *where A is a positively semi-definite matrix and* $\mathcal{X} = \mathbb{R}^d$;
5. $K(x, y) = K_1(x, y)K_2(x, y)$;
6. $K(x, y) = p(K_1(x, y))$, *where p is a polynomial with positive coefficients;*
7. $K(x, y) = K_1(f(x), f(y)), \quad f : \mathcal{X} \to \mathcal{X}$
8. $K(x, y) = \exp[K_1(x, y)]$
9. $K(x, y) = \exp[-\frac{\|x-y\|^2}{2\sigma^2}]$, $x, y \in \mathbb{R}^d$.

Proof. Exercise: Prove the properties 1, 2, 3, 4.

5) Let $(V_1, \ldots, V_n)$ and $(W_1, \ldots, W_n)$ be two independent multivariate normal random vectors with means zero, and respective covariance matrices K_1 and K_2. Then the matrix K, whose elements are the products of the corresponding elements of K_1 and K_2 is is the covariance matrix of random vector $(V_1W_1, \ldots, V_mW_m)$ so that it is positive semi-definite.

Exercise Prove the properties 6) and 7).

8) Use Taylor expansion: for every x, y

$$\exp[K_1(x,y)] = \sum_{i=0}^{\infty} \frac{1}{i!} K_1(x,y)^i = \lim_n p_n(K_1(x,y)),$$

where $p_n(x) = \sum_{i=0}^n \frac{1}{i!}x^i$. From property 6, we know that $K_n(\cdot,\cdot) := p_n(K_1(\cdot,\cdot))$ is a kernel. Hence, there exists kernels K_n so that for all x, y, $\exp[K_1(x,y)] = \lim_n K_n(x,y)$. Now it suffices to notice that a pointwise limit of kernels is a kernel

9) Since $\|x-y\|^2 = \|x\|^2 + \|y\|^2 - 2\langle x, y\rangle$, we have

$$\exp[-\frac{\|x-y\|^2}{2\sigma^2}] = \exp[-\frac{\|x\|^2}{2\sigma^2}] \exp[-\frac{\|y\|^2}{2\sigma^2}] \exp[\frac{\langle x,y\rangle}{\sigma^2}] = K_1(x,y)K_2(x,y),$$

where

$$K_1(x,y) = \exp[-\frac{\|x\|^2}{2\sigma^2}] \exp[-\frac{\|y\|^2}{2\sigma^2}], \quad K_2(x,y) = \exp[\frac{\langle x,y\rangle}{\sigma^2}].$$

From the property 3) it follows that K_1 is a kernel. From the property 8) it follows that K_2 is a kernel. From 5) it follows that K is a kernel. ■

Bochner's theorem. The kernel

$$K(x,y) = \exp[-\frac{\|x-y\|^2}{2\sigma^2}] \tag{4.5.17}$$

is in form $f(x-y)$, where $f : \mathbb{R}^d \to \mathbb{R}$. Is it possible to characterize functions f so that $K(x,y) := f(x-y)$ is a kernel? Those functions are called it positive semi-definite functions.

Theorem 4.5.5 (Bochner) *A continuous function $f : \mathbb{R}^d \to \mathbb{R}$ is positve semi-definite if and only if if there there exists a finite measure μ on $\mathcal{B}(\mathbb{R}^d)$ so that*

$$f(x) = \int \exp[-ix'z]\mu(dz).$$

Note: when $f(0) = 1$, then μ is a probability measure and f is its characteristic function. Now 9) of Lemma 4.5.1 immediately follows, since $f(x) = \exp[-\frac{\|x\|^2}{2\sigma^2}]$ is the characteristic function of multivariate normal distribution $N(0, \sigma^2 I_d)$.

4.5.5 Examples of kernels

Polynomial kernel. Let $\mathcal{X} = \mathbb{R}^d$, $R > 0$ and define **polynomial kernel**

$$K(x, y) := \big(x'y + R\big)^p = \sum_{s=0}^{p} \binom{p}{s} R^{p-s} (x'y)^s = \sum_{s=0}^{p} \binom{p}{s} R^{p-s} (x'y)^s \tag{4.5.18}$$

From Lemma 4.5.1 we know that for any polynomial p with positive coefficients, the mapping $(x, y) \mapsto p(x'y)$ is a kernel. Hence (4.5.18) is a kernel as well, because the coefficients are non-negative: $\binom{p}{s} R^{p-s} \geq 0$.
With polynomial kernel, a classifier (4.5.2) is

$$\text{sign}\big(\sum_{i \in SV} \alpha_i^* y_i K(x_i, x) + w_o^*\big) = \text{sign}\big(\sum_{i \in SV} \alpha_i^* y_i (x_i' x + R)^p + w_o^*\big),$$

so that the classification is based on the p-order surface

$$\{x : \sum_{i \in SV} \alpha_i^* y_i (x_i' x + R)^p = -w_o^*\}$$

The meaning of R becomes apparent from (4.5.18):

$$K(x, y) = \sum_{s=0}^{p} a_s (x'y)^s, \quad a_s := \binom{p}{s} R^{p-s}.$$

Hence R controls the relative weightings of the different degree monomials $(x'y)^s$: the bigger R, the smaller is the relative weight of higher order monomials. If $R \approx 0$, then only $(x'y)^p$ – the monomial with maximum power – counts.

Gaussian kernel. Let $\mathcal{X} = \mathbb{R}^d$. **Gaussian kernel** or *RBF kernel* is (4.5.17), i.e.

$$K(x, y) = \exp[-\frac{\|x - y\|^2}{2\sigma^2}].$$

Classifier (4.5.2) is

$$\text{sign}\big(\sum_{i \in SV} \alpha_i^* y_i \exp[-\frac{\|x_i - x\|^2}{2\sigma^2}] + w_o^*\big),$$

so that the classification is based on the surface

$$\Big\{x : \sum_{i \in SV} \alpha_i^* y_i \exp[-\frac{\|x_i - x\|^2}{2\sigma^2}] = -w_o^*\Big\}.$$

Exercise: Prove that any Φ corresponding to Gaussian kernel satisfies $\Phi : \mathcal{X} \mapsto S_{\mathcal{H}}$, where $S_{\mathcal{H}}$ is unit sphere on $\mathcal{H}$.

Gaussian kernel has the following important property: for any finite subset $\{x_1, \ldots, x_n\} \subset \mathbb{R}^d$, the corresponding Gram matrix K has full rank ([12], Thm 2.18). From this follows that for any sample where $x_i \neq x_j$ if $i \neq j$, and for any Φ associated to Gaussian kernel, the sample in the Hilbert space

$$(\Phi(x_1), y_1), \ldots, (\Phi(x_n), y_n) \tag{4.5.19}$$

is linearly separable. Let us show this. Since Gram matrix has full rank, its columns form a base in $\mathbb{R}^n$. Hence, to every vector of labels $(y_1, \ldots, y_n)$ there exist constants $w^1, \ldots w^n$ so that

$$\sum_{j=1}^n w^j K(x_i, x_j) = \sum_{j=1}^n w^j \langle \Phi(x_i), \Phi(x_j) \rangle = \langle \sum_{j=1}^n w^j \Phi(x_j), \Phi(x_i) \rangle = y_i.$$

Since $f := \sum_{j=1}^n w^j \Phi(x_i) \in \mathcal{H}$, we obtain that the hyperplane $H := \{g \in \mathcal{H} : \langle g, f \rangle = 0\}$ separates (4.5.19). In other words it means that Gaussian kernel induces so large class of (non-linear) classifiers on $\mathbb{R}^d$ that every sample (without repetitions) can be correctly classified. This means that the VC-dimension of the corresponding set of classifiers is infinite. Let us remark that VC-dimension of the set of classifiers induced by polynomial kernel is finite.
Hence for Gaussian kernel theoretically hard-margin SVM classification can be used. But this typically means overfitting. To avoid overfitting, the box constraints or 2-norm soft margin SVM-classification is used.
The parameter σ controls the flexibility of the kernel in a similar way to the degree p in the polynomial kernel – small values of σ correspond to large values of p so that given some additional (say box) constraint, the class of classifiers is more complex for smaller σ. The Gram matrix has full rank for any σ, but given some additional constraints that prevent overfitting, the parameter σ affects the performance of resulting classifier.

Polynomial kernel and Gaussian kernel are probably most commonly used kernels in case $\mathcal{X} = \mathbb{R}^d$. Let us note that there are many other kernels similar to polynomial kernels (e.g *all-subsets kernels* or *ANOVA kernels*), see [11], Ch 9.

Kernels for texts

The advantage of kernel methods is that the set $\mathcal{X}$ can be arbitrary so that the methods apply for very large class of problems. As an example, we briefly examine some commonly used kernels for classifications of documents (texts). Document classification is often refereed to as **categorization**.

The kernels are derived from the representation of text. A commonly used representation is so-called **vector space model** or **bag of words approach**. Let $W = \{w_1, \ldots, w_m\}$ be a finite *dictionary*. A dictionary can be given, in practice it typically consists of all words from a set of documents typically referred to as *corpus.* Let $T \in W^*$ be a text (document) and define

$$\Phi(T) = \big(\Phi_{w_1}(T), \dots, \Phi_{w_m}(T)\big),$$

where $\Phi_w(T)$ is the frequency of the word w in the text T. Thus every text is represented as a very long vector, where most of the entries are zero. In other words, a text is considered as a bag of words, where a bag is a set in which repeated elements are allowed (even simpler approach would be to ignore the repetitions). Kernel is then

$$K(T_1, T_2) = \langle \Phi(T_1), \Phi(T_2) \rangle_{l_2} = \sum_w \Phi_w(T_1)\Phi_w(T_2) = \Phi'(T_1)\Phi(T_2).$$

Although the vectors are very long having often more entries than the sample size, it turns out that since the representation is extremely sparse (most of the entries are zero) the computation of inner product $K(T_1, T_2)$ can be implemented in a time proportional to the sum of the lengths of two documents

$$O(|T_1| + |T_2|).$$

The process allowing to compute the inner product with such a low cost is known in computer science as *tokenisation.*

Weighting. In order to incorporate sematic into a kernel, the first step is to apply different weights the every word. For instance to assign zero-weight to non-informative words like "and", "of", "the" and so on. These words are often referred to as *stop words.* With $\mu(w)$ being the weight of the word w, we get

$$\Phi(T) = \big(\Phi_{w_1}(T)\mu(w_1), \dots, \Phi_{w_s}(T)\mu(w_s)\big),$$

so that the kernel is

$$K(T_1, T_2) = \sum_w \mu^2(w)\Phi_w(T_1)\Phi_w(T_2) = \Phi'(T_1)R'R\Phi(T_1),$$

where R is a diagonal matrix with entries $\mu(w)$ on the diagonal. In other words, the vector Φ is replaced weighted vector $R\Phi$.
A possibility for $\mu(w)$ is the so called *inverse document frequency* of the word w. Suppose there are n documents in the corpus and let $df(w)$ be the number of documents containing the word w. The weight of inverse document frequency is then

$$\mu(w) := \ln\left(\frac{n}{df(w)}\right).$$

Hence, if the word w is in every text, then $\mu(w) = 0$. The use of the logarithmic function ensures that none of the weights can become too large relative to the weight of a word that occurs in roughly halve of the documents. Note that with inverse document frequency weights, the evaluation of kernel involves both frequencies $\Phi_w(T)$ as well as inverse document frequencies. It is therefore often referred to as the *tf-idf* representation.

Proximity matrix. The *tf-idf* representation implements a downweighting of irrelevant terms as well as highlighting potentially informative ones, but it is not capable of recognizing semantically related words like synonyms. To incorporate the semantic similarity, the *proximity matrix* $P = \big(P(w,w')\big)_{w,w'\in W}$ is introduced. In the proximity matrix, $P(w,w) = 1$, whereas the off-diagonal entry $P(w,w') > 0$, if the words w and w' are semantically related. The linear transformation P is applied to weighted vector $R\Phi(T)$ so that the new vector is $\Phi^*(T) = PR\Phi(T)$. The vector $\Phi^*(T)$ is less sparse then the weighted vector $R\Phi(T)$. With $Q = (RP)'PR$, the kernel is now

$$K(T_1,T_2) = \sum_w \Phi_w(T_1)\Phi^*_w(T_2) = \Phi'(T_1)R'P'P\ R\Phi(T_2) = \Phi'(T_1)Q\Phi(T_2).$$

References: About the properties of kernels, polynomial, Gaussian and related kernels, see [11], Ch 9, [12], Ch 13 and 2. About kernels for text, see [11], Ch 10. About kernels for other types of data (strings, sets, generative models), see [11], Ch 11, Ch 9, Ch 12.

4.6 Regression with kernels

Most of linear methods of multivariate analysis have their kernel counterparts. For example, there exists kernel principal component analysis (Kernel PCA, see [11], Ch 6 or [12], Ch 14) and kernel version of Fisher discriminant analysis (Kernel FDA, see [12], Ch 15; [11] Chs 5, 6; [13]). In the following we briefly introduce the kernel regression methods.

4.6.1 Ridge and lasso regression

In regression, the data are $(x_1,y_1),\dots,(x_n,y_n)$, where $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$. We know that in *ordinary least squares* the parameters $\hat{w} \in \mathbb{R}^d$ and $\hat{a} \in \mathbb{R}$ are the solution of the following problem:

$$\min_{w,a} \sum_{i=1}^n \big(y_i - (w'x_i + a)\big)^2. \tag{4.6.1}$$

We also know that (recall (3.6.8) and replace expectations with sample average)

$$\hat{w} = \hat{\Sigma}^{-1}\big(\frac{1}{n}\sum_{i=1}^n x_i y_i - \bar{y}\bar{x}\big), \quad \hat{a} = \bar{y} - \hat{w}'\bar{x}.$$

With

$$Z := \begin{pmatrix} x_1^1 - \bar{x}^1 & x_1^2 - \bar{x}^2 & \cdots & x_1^d - \bar{x}^d \\ x_2^1 - \bar{x}^1 & x_1^2 - \bar{x}^2 & \cdots & x_1^d - \bar{x}^d \\ \cdots & \cdots & \cdots & \cdots \\ x_n^1 - \bar{x}^1 & x_n^2 - \bar{x}^2 & \cdots & x_n^d - \bar{x}^d \end{pmatrix},$$

we have

$$\frac{1}{n}\sum_{i=1}^n x_i y_i - \bar{y}\bar{x} = \frac{1}{n}Z'(y - \bar{y}), \quad \hat{\Sigma} = \frac{1}{n}Z'Z,$$

so that

$$\hat{w} = (Z'Z)^{-1}Z'y^o, \quad \text{where} \quad y^o = y - \bar{y}.$$

Ridge regression. When $\hat{\Sigma}$ is not invertible, several alternatives are used. A popular method is **ridge regression**, where instead (4.6.1) the problem is minimized:

$$\min_{w,a} \sum_{i=1}^{n} \big(y_i - (w'x_i + a)\big)^2 + \lambda \|w\|^2, \tag{4.6.2}$$

where $\lambda > 0$. Hence ridge regression prefers the smaller coefficients. Alternatively, the ridge regression problem can be defined as follows

$$\min_{w,a} \sum_{i=1}^{n} \big(y_i - (w'x_i + a)\big)^2 \tag{4.6.3}$$
$$\text{subject to } \|w\| \leq B.$$

Exercise: Show that to every λ, there corresponds a $B < \infty$ (depending on the data) so that the solution of (4.6.3) is the same as the solution of (4.6.2). (Hint: take $B = \|\hat{w}\|$)

The solutions of (4.6.2) are easy to find:

$$\hat{w} = \big(Z'Z + \lambda I_d\big)^{-1} Z'y^o, \quad \hat{a} = \bar{y} - \hat{w}'\bar{x}. \tag{4.6.4}$$

The matrix $Z'Z + \lambda I_d$ is invertible even if $Z'Z$ is not.

Exercise: Prove (4.6.4). Hint: use *centering* and proceed as follows.

1. Replace x_i by $z_i = x_i - \bar{x}$ and show that the solutions $\hat{v}$ and $\hat{b}$ of

$$\min_{v,b} \sum_{i=1}^{n} \Big(y_i - \big(v'z_i + b\big)\Big)^2 + \lambda \|v\|^2. \tag{4.6.5}$$

   are related to the solutions of (4.6.2) as follows: $\hat{w} = \hat{v}$ and $\hat{a} = \hat{b} - \bar{x}'\hat{v}$.

2. To find $\hat{b}$, replace y by $y^o = y - \bar{y}$ and reformulate (4.6.5) as follows

$$\min_{v,b} \sum_{i=1}^{n} \Big(y_i^o - \big(v'z_i + (b - \bar{y})\big)\Big)^2 + \lambda \|v\|^2. \tag{4.6.6}$$

   Then show that $\hat{b} = \bar{y}$.

3. To find $\hat{v}$, it suffices now to consider the problem

$$\min_{v} \sum_{i=1}^{n} \big(y_i^o - v'z_i\big)^2 + \lambda \|v\|^2 = \min_{v \in \mathbb{R}^d} \langle y^o - Zv, y^o - Zv \rangle + \lambda v'v \tag{4.6.7}$$

   and show that its solution is

$$\hat{v} = \big(Z'Z + \lambda I_d\big)^{-1} Z'y^o.$$

Theoretical reasonings for ridge regression

1. Typically d is big and the regression tries to select the features that are important in regression. In ridge regression this goal is tried to achieve via penalizing the sum of squares of coefficients. When some features are not related to the response variable, then corresponding coefficients tend to be small.

2. Let $\hat{y}(x_o) = \hat{w}'x_o + \hat{a}$ be the prognose at x_o, and let $f(x_o) = E[Y|X = x_o]$. It is easy to see that the following decomposition of mean square error holds

$$E\big[\big(Y - \hat{y}(x_o)\big)^2|X = x_o\big] =$$
$$E\big[(Y - f(x_o))^2|X = x_o\big] + \big(f(x_o) - E\hat{y}(x_o)\big)^2 + E\big(\hat{y}(x_o) - E\hat{y}(x_o)\big)^2.$$

   It can be shown that increasing λ decreases the variance $E\big(\hat{f}(x_o) - E\hat{y}(x_o)\big)^2$ and increases the bias. Hence λ controls the so-called *bias-variance tradeoff*.

3. Matrix $(Z'Z + \lambda I)$ is always invertible, even if $Z'Z$ is not.

Lasso regression. In **lasso regression** (Tibshirani, 1996), the problem is

$$\min_{w,a} \sum_{i=1}^{n} \big(y_i - (w'x_i + a)\big)^2 + \lambda \sum_i |w^i|. \tag{4.6.8}$$

As in case of ridge regression, for some B (depending on λ and the data) the problem (4.6.8) can be written

$$\min_{w,a} \sum_{i=1}^{n} \big(y_i - (w'x_i + a)\big)^2 \tag{4.6.9}$$
$$\text{subject to} \|w\|_1 \leq B.$$

Hence the difference between lasso and ridge regression is in the penalty term – in ridge regression, the size of $\hat{w}$ is measured in 2-norm, whereas in lasso regression the size of $\hat{w}$ is measured in 1-norm. It turns out that this difference is essential – due to the non-smoothness of l_1-norm, the solution of (4.6.8) is typically much sparser (having more zero's) than the one of ridge regression. Therefore, lasso regression in often preferred over ridge regression, since the the number of selected features are smaller. On the other hand, because of the non-smoothness of l_1-norm, the solution of (4.6.8) is not known analytically and has to find iteratively. In [14], so called LARS-software is introduced.

References: About ridge and lasso regression read [7].

4.6.2 Kernel ridge regression

Because of the kernel trick, the ridge regression is suitable for kernel methods. To start with, let us rewrite the obtained solutions in terms of kernels. At first note: when the feature vectors x_i are centered (as the columns of Z-matrix) and Φ is identity, then Gram matrix K^o (the o in notation reflects the centering) is $K^o = ZZ'$ and the scatter matrix is $S = Z'Z$. Since K^o is $n \times n$ matrix and S is $d \times d$ matrix then in classical multivariate analysis, where d is typically much smaller than n, the main object is the scatter or covariance matrix. In kernel methods, however, d is typically very large, perhaps infinity, so that the Gram matrix becomes the main object of analysis. Note that when $d > n$, the S is not invertible, whilst K^o can be.

Let us re-examine the solutions of ridge regression (4.6.4). We know that

$$\hat{w} = \big(Z'Z + \lambda I_d\big)^{-1} Z'y^o.$$

Exercise: Prove that $\hat{w} = Z'\alpha^*$, where

$$\alpha^* = \lambda^{-1}(y^o - Z\hat{v}) = (K^o + \lambda I_n)^{-1} y^o.$$

Recall that $\hat{w}$ is also the solution of centered problem (4.6.5). Because of the Representer theorem, we know that the solution $\hat{w}$ can be written in the form

$$\hat{w} = \sum_{i=1} \alpha_i^* z_i = Z'\alpha^*, \quad \alpha^* \in \mathbb{R}^n$$

so that the exercise above is not surprising. We can thus search the solution w in the form $w = Z'\alpha$ so that $w'w = \alpha' K^o \alpha$ and the problem (4.6.7) is

$$\min_\alpha \langle y^o - K^o\alpha, y^o - K^o\alpha \rangle + \lambda \alpha' K^o \alpha. \tag{4.6.10}$$

Setting the gradient with respect to α equal to zero, one gets

$$K^o\big((K^o + \lambda I_n)\alpha - y^o\big) = 0.$$

If K^o is not invertible, then there might be more than one α satisfying the equality above. However, for any such α,

$$Z'\big((K^o+\lambda I_n)\alpha - y^o\big) = 0 \quad \Rightarrow \quad Z'(K^o+\lambda I_n)\alpha = Z'ZZ'\alpha + \lambda Z'\alpha = (S+\lambda I_d)Z'\alpha = Z'y^o.$$

Since $(S + \lambda I_d)$ is invertible, we get that all such α define unique $Z'\alpha$ that is equal to $Z'\alpha^*$, where

$$\alpha^* = (K^o + \lambda I_n)^{-1} y^o.$$

As mentioned, when $d > n$, the matrix S is not invertible, the matrix K^o might be. Suppose for a moment that this is the case, and $\lambda = 0$. Then the problem is OLS-regression. Then $\alpha^* = (K^o)^{-1} y^o$ and applying $\hat{w}$ to the set of feature vectors, we get

$$\hat{y}' = \hat{w}'Z' = \alpha^{*\prime} ZZ' = (y^o)'(K^o)^{-1}ZZ' = y^{o\prime}$$

so that the regression hyperplane passes all data point – clear overfitting. Hence, doing regression in the high-dimensional spaces some regularization is inevitable and ridge regression is one easy applicable option.

Ridge regression with Lagrange method. For big n, finding $(K^o+\lambda I_n)^{-1}$ might be difficult. Let us restate the problem as the problem of constrained optimization. Solving the dual problem with some iterative methods can be easier than directly calculating the inverse matrix. With the help of slack variables, the problem (4.6.10) is

$$\begin{gathered}\min_{a,\xi,\alpha}\ \lambda\alpha' K^o\alpha+\|\xi\|^2\\ \text{subject to } K^o\alpha=y^o-\xi.\end{gathered}\tag{4.6.11}$$

Lagrangian:

$$L(\alpha,\xi,\gamma)=\lambda\alpha' K^o\alpha+\|\xi\|^2+\gamma'(y^o-K^o\alpha-\xi).$$

The derivatives with respect to the primal variables

$$\begin{aligned}\nabla_\alpha L(\alpha,\xi,\gamma)&=2\lambda K^o\alpha-\gamma' K^o=0\quad\Rightarrow\quad K^o\alpha=\frac{1}{2\lambda}K^o\gamma.\\ \nabla_\xi L(\alpha,\xi,\gamma)&=2\xi-\gamma=0\quad\Rightarrow\quad \xi=\frac{1}{2}\gamma.\end{aligned}$$

Hence

$$\alpha' K^o\alpha=\frac{1}{2\lambda}\alpha' K^o\gamma=\frac{1}{2\lambda}\gamma' K^o\alpha=\frac{1}{4\lambda^2}\gamma' K^o\gamma,\quad \|\xi\|^2=\frac{1}{4}\gamma'\gamma$$

and plugging these formulas into Lagrangian, we obtain

$$\theta(\gamma)=\gamma' y^o-\frac{1}{4\lambda}\gamma K^o\gamma-\frac{1}{4}\gamma'\gamma=\gamma' y^o-\frac{1}{4\lambda}\gamma'\big(K^o+\lambda I_n\big)\gamma$$

so that the dual problem is

$$\max_\gamma \gamma' y^o-\frac{1}{4\lambda}\gamma'\big(K^o+\lambda I_n\big)\gamma.\tag{4.6.12}$$

If K^o is invertible, then (4.6.11) has unique solution $\alpha^*=\frac{1}{2\lambda}\gamma^*$, where γ^* is the solution of dual. Otherwise there might be many α^* satisfying equality

$$K^o\alpha^*=\frac{1}{2\lambda}K^o\gamma^*,$$

but the corresponding weight vector $\hat{w}=Z'\alpha^*$ is for all of them equal to

$$\hat{w}=\frac{1}{2\lambda}Z'\gamma^*.$$

KKT:

$$y_i^o-\sum_j K_{ij}^o\alpha_j^*=\xi_i^*=\frac{\gamma_i^*}{2}.$$

Hence,

$$\gamma_i^*=0\quad\Leftrightarrow\quad y_i^o=\sum_j K_{ij}^o\alpha_j^*.\tag{4.6.13}$$

Mapping Φ. Let $\Phi : \mathcal{X} \to \mathcal{H}$. The transformed sample in Hilbert space $\mathcal{H}$ is

$$(\Phi(x_1), y_1), \dots, (\Phi(x_n), y_n).$$

Kernel ridge regression problem is

$$\min_{w \in \mathcal{H}, a \in \mathbb{R}} \sum_i^n \big(y_i - (\langle w, \Phi(x_i)\rangle + a)\big)^2 + \lambda \|w\|^2. \tag{4.6.14}$$

Centering in space $\mathcal{H}$. Just like in $\mathbb{R}^d$, we can use *centering*. But now the centering has to be carried out in $\mathcal{H}$. Hence, let Φ_S be the average of $\Phi(x_1), \dots, \Phi(x_n)$, i.e.

$$\Phi_S = \frac{1}{n} \sum_{i=1}^n \Phi(x_i).$$

Let the centered mapping be

$$\Phi^o : \mathcal{X} \to \mathcal{H}, \quad \Phi^o(x) = \Phi(x) - \Phi_S, \quad i = 1, \dots, n$$

and let K^o be the centered Gram matrix, i.e. K^o is $n \times n$ matrix with the entries

$$\begin{aligned} K^o_{ij} :=: K^o(x_i, x_j) &= \langle \Phi^o(x_i), \Phi^o(x_j)\rangle = \langle \Phi(x_i), \Phi(x_j)\rangle - \langle \Phi(x_i), \Phi_S\rangle - \langle \Phi_S, \Phi(x_j)\rangle + \langle \Phi_S, \Phi_S\rangle \\ &= K(x_i, x_j) - \frac{1}{n}\sum_{k=1}^n K(x_i, x_k) - \frac{1}{n}\sum_{k=1}^n K(x_k, x_j) + \frac{1}{n^2}\sum_{i,j} K(x_i, x_j). \end{aligned}$$

Thus, from the original Gram matrix K, the matrix K^o can be obtained by simple transformation

$$K^o = K - \frac{1}{n} K 1 1' - \frac{1}{n} 1 1' K + \frac{1}{n^2} 1' K 1, \tag{4.6.15}$$

where 1 is vector consisting of ones.

After centering the transformed features and response vector, we get (as previously) that the solutions of (4.6.14) are $\hat{a}$ and $\hat{w}$, where

$$\hat{a} = \bar{y} - \langle \hat{v}, \Phi_S\rangle, \quad \hat{w} = \hat{v}$$

and $\hat{v}$ is the solution of the following problem

$$\min_{v \in \mathcal{H}} \sum_i^n \big(y^o_i - \langle v, \Phi^o(x_i)\rangle\big)^2 + \lambda \|v\|^2.$$

Without loss of generality, we can take $\mathcal{H}$ as RKHS, so that Representer theorem applies and

$$\hat{v} = \sum_i \alpha^*_i \Phi^o(x_i),$$

where α^* is the solution of the problem (4.6.10):

$$\min_{\alpha \in \mathbb{R}^n} \langle y^o - K^o\alpha, y^o - K^o\alpha\rangle + \lambda\alpha' K^o\alpha.$$

We already know that a solution is

$$\alpha^* = (K^o + \lambda I_n)^{-1} y^o$$

and all others (if they exists) define the same $\hat{v}$. We also know that (4.6.10) can be reformulated as a constrained optimization problem (4.6.11) and all solutions (if not unique) of it are equivalent (i.e. giving the same w) to $\alpha^* = \frac{1}{2\lambda}\gamma^*$, where γ^* is the solution of (4.6.12). With α^*, the optimal constant is

$$\hat{a} = \bar{y} - \langle \hat{v}, \Phi_S\rangle = \bar{y} - \langle \sum_j \alpha_j^* \Phi^o(x_j), \Phi_S\rangle = \bar{y} - \langle \sum_j \alpha_j^* \Phi(x_j), \Phi_S\rangle - (\sum_j \alpha_j^*)\langle \Phi_S, \Phi_S\rangle$$
$$= \bar{y} - \frac{1}{n} 1' K\alpha^* - (1'\alpha^*)\langle \Phi_S, \Phi_S\rangle = \bar{y} - \frac{1}{n} 1' K\alpha^* - \frac{(1'\alpha^*)}{n^2} 1' K 1.$$

For any $x \in \mathcal{X}$, the prognose is

$$\hat{y} := \langle w^*, \Phi(x)\rangle + \hat{a} = \sum_i \alpha_i^* K(x, x_i) + \hat{a}.$$

Support vectors. Recall the equality (4.6.13): if $\alpha_i^* = 0$, then

$$y_i^o = \sum_j K_{ij}^o \alpha_j^* = \langle w^*, \Phi^o(x_i)\rangle,$$

i.e. the coefficient α_i^* can be zero zero only if the pair $(\Phi^o(x_i), y_i^o)$ lies on the regression hyperplane. When $\lambda > 0$, this is rather untypical, hence most of the coefficients α_i^* are not zero, i.e. in terms of support vectors, most of x_i are support vectors.

4.6.3 ϵ-insensitive regression

We saw the the the solution of ridge regression $\hat{w} = \sum_i \alpha_i^* \Phi(x_i)$ is typically not sparse, i.e. mostly $\alpha_i^* \neq 0$. To obtain sparser solutions, several alternatives are used. One of them is ignoring the errors that are smaller than a certain threshold $\epsilon > 0$. For this reason the band around the true output is sometimes referred to as a ϵ-*tube*.

Formally, let $\epsilon > 0$ be fixed and for any pair $y, f(x) \in \mathbb{R}$ define **ϵ-insensitive p loss** as follows

$$|y - f(x)|_\epsilon^p, \quad \text{where} \quad |y - f(x)|_\epsilon := \max\{0, |y - f(x)| - \epsilon\}.$$

Hence

$$\big(|y - f(x)|_\epsilon\big)^p = 0 \quad \text{iff } |y - f(x)| \leq \epsilon.$$

Typically $p = 1$ or $p = 2$, and both cases are combined with ridge regression.

The case $p = 2$. When $p = 2$, we obtain a generalization of kernel ridge regression problem as follows

$$\min_{w\in\mathcal{H},a\in\mathbb{R}} \sum_{i=1}^{n} |y_i - (\langle w, \Phi(x_i)\rangle + a)|_\epsilon^2 + \lambda\|w\|^2. \tag{4.6.16}$$

Considering this as a constrained optimization problem and searching solution in form $\sum_i \alpha_i\Phi(x_i)$, we obtain the following primal problem (in matrix form)

$$\begin{aligned} \min\nolimits_{a,\xi,\alpha}\ & \lambda\alpha' K\alpha + \|\xi^+\|^2 + \|\xi^-\|^2 \\ \text{subject to } & K\alpha + a1 - y \leq \xi^- + \epsilon 1 \\ & y - K\alpha - a1 \leq \xi^+ + \epsilon 1. \end{aligned} \tag{4.6.17}$$

Since for $\epsilon > 0$, the ϵ-insensitive loss is not any more quadratic, we cannot benefit for centering and find the constant separately. Hence, in (4.6.17), the original Gram matrix K and vector y are used, the constant a will be later determined via KKT conditions.

The dual of that problem is (recall (4.6.12))

$$\begin{aligned} \max\nolimits_\gamma\ \gamma' y - \frac{1}{4\lambda}\gamma'(K + \lambda I_n)\gamma - \epsilon\|\gamma\|_1 \\ \text{subject to } 1'\gamma = 0. \end{aligned} \tag{4.6.18}$$

With γ^* being the solution of (4.6.18), the solutions of (4.6.17) and (4.6.16) are

$$\alpha^* = \frac{1}{2\lambda}\gamma^*, \quad \hat{w} = \sum_i \alpha_i^*\Phi(x_i).$$

From KKT conditions, it follows that when $\gamma_i^* \neq 0$, then

$$y_i - \sum_j K_{ij}\alpha_i^* - a^* = y_i - \langle w^*, \Phi(x_i)\rangle - a^* = \begin{cases} \frac{\gamma_i^*}{2} + \epsilon, & \text{when } \gamma_i^* > 0; \\ \frac{\gamma_i^*}{2} - \epsilon, & \text{when } \gamma_i^* < 0; \end{cases} \tag{4.6.19}$$

These relations can be used to found the optimal a^*. From (4.6.19), it also follows that

$$|y_i - (\sum_j K_{ij}\alpha_i^* + a^*)|_\epsilon = |y_i - (\langle w^*, \Phi(x_i)\rangle + a^*)|_\epsilon = \frac{|\gamma_i^*|}{2}.$$

Hence the support vectors (x_i such that $\alpha_i^* \neq 0$) are all features x_i such that $(y_i, \Phi(x_i))$ is located outside the ϵ-tube of regression hyperplane.

The case $p = 1$. Then the problem is

$$\min_{w \in \mathcal{H}, a \in \mathbb{R}} \sum_{i=1}^{n} |y_i - (\langle w, \Phi(x_i)\rangle + a)|_\epsilon + \lambda \|w\|^2. \tag{4.6.20}$$

Considering this as a constrained optimization problem and searching solution in form $\sum_i \alpha_i \Phi(x_i)$, we obtain the following primal problem

$$\begin{aligned} \min_{a,\xi,\alpha} \ & \lambda \alpha' K \alpha + \|\xi^+\|_1 + \|\xi^-\|_1 \\ \text{subject to } & K\alpha + a1 - y \le \xi^- + \epsilon 1 \\ & y - K\alpha - a1 \le \xi^+ + \epsilon 1 \\ & \xi^+ \ge 0, \quad \xi^- \ge 0. \end{aligned}$$

Again, the solution α^* satisfies

$$\alpha^* = \frac{\gamma^*}{2\lambda},$$

where γ^* is the solution of the dual problem

$$\begin{aligned} \max_\gamma \ & -\frac{1}{4\lambda}\gamma' K \gamma + \gamma' y - \epsilon \|\gamma\|_1 \\ \text{subject to } & \gamma' 1 = 0 \\ & |\gamma_i| \le 1, \quad \forall i. \end{aligned}$$

Note that we encounter the box constraints again: $|\alpha_i^*| \le \frac{1}{2\lambda}$, and as in the case of support vector classification, $|\alpha_i^*| = \frac{1}{2\lambda}$, when $(y_i, \Phi(x_i))$ is outside of the ϵ-tube.
The in-bound support vectors, again, are those sample elements, for which

$$|\alpha_i^*| \in \big(0, \frac{1}{2\lambda}\big).$$

From KKT conditions, it follows that for an in-bound support vector x_i, it holds

$$y_i - \sum_j K_{ij}\alpha_i^* - a^* = y_i - \langle w^*, \Phi(x_i)\rangle - a^* = \begin{cases} \epsilon, & \text{when } \gamma_i^* > 0; \\ -\epsilon, & \text{when } \gamma_i^* < 0. \end{cases} \tag{4.6.21}$$

These relations can be used to found the optimal a^*, and they show that every in-bound support vector x_i is such that $(y_i, \Phi(x_i))$ lays exactly on the boundary of ϵ-tube.

References: About regression with kernels, read [11], Ch 7.3; [12], Ch 9; [10], Ch 6.

Chapter 5

Boosting

Assumption: Two classes, labeled as $+1$ and -1.

5.1 Risk and ϕ-risk

Recall soft margin classification problem:

$$\min_{w\in\mathcal{H},w_o} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^n \Big(1 - y_i\big(w(x_i)+w_o\big)\Big)_+^p, \tag{5.1.1}$$

where $\mathcal{H}$ is a RKHS, p is 1 or 2. As in the case of regression, there exists a constant B, depending on the data so that the solution of (5.1.1) is the solution of the problem

$$\min_{w,w_o:\|w\|\le B} \frac{1}{n}\sum_{i=1}^n \Big(1 - y_i\big(w(x_i)+w_o\big)\Big)_+^p. \tag{5.1.2}$$

Indeed, let w^* and w_o^* be the solutions of (5.1.1) and take $B = \|w^*\|$. Let us show that the pair w^* and w_o^* is also the solution of (5.1.2). If not, there would be a pair w and w_o such that $\|w\| \le B$ and

$$\frac{1}{n}\sum_{i=1}^n \Big(1 - y_i\big(w(x_i)+w_o\big)\Big)_+^p < \frac{1}{n}\sum_{i=1}^n \Big(1 - y_i\big(w^*(x_i)+w_o^*\big)\Big)_+^p.$$

That would contradict the assumption that w^* and w_o^* are the solutions of (5.1.1).

Let for any $B > 0$

$$\mathcal{F}_B := \{f_{w,w_o} : \|w\| \le B\}, \quad \text{where} \quad f_{w,w_o} : \mathcal{X} \to \mathbb{R}, \quad f_{w,w_o}(x) = w(x) + w_o.$$

Hence the problem (5.1.2) is

$$\min_{f\in\mathcal{F}_B} \frac{1}{n}\sum_{i=1}^n \big(1 - y_i f(x_i)\big)_+^p = \min_{f\in\mathcal{F}_B} \frac{1}{n}\sum_{i=1}^n \phi\big(y_i f(x_i)\big), \tag{5.1.3}$$

where

$$\phi(t) = (1-t)_+^p.$$

Hence SVM-classification is one of the many methods that aims to maximize functional margins $y_i f(x_i)$ via minimizing the **empirical ϕ-risk**

$$\frac{1}{n}\sum_{i=1}^{n}\phi\big(y_i f(x_i)\big)$$

for a non-increasing function ϕ over a class of functions $\mathcal{F}$. If n is big enough, then the sample average is close to the expectation so that the empirical ϕ-risk is close to **ϕ-risk**:

$$R_\phi(f) := E\phi(Yf(X)) = \int \phi(yf(x))dF(x,y)$$

and, therefore, minimizing empirical ϕ-risk over $\mathcal{F}$ can be considered ERM (empirical risk minimization) method for minimizing unknown (because $F(x,y)$ is not known) ϕ risk R_ϕ.

Why ϕ risk? Recall that for any $f \in \mathcal{F}$, our primal interest is the classifier

$$g(x) = \mathrm{sgn} f(x)$$

and the objective is to select f so that the risk

$$R(g) = \mathbf{P}(Y \neq g(X)) = \mathbf{P}(Y \neq \mathrm{sgn}(f(X)) =: R(f)$$

would be as small of possible. Therefore, the natural question arises: how are the risks $R_\phi(f)$ and $R(f)$ connected? In particular, if R^*_ϕ is a minimum of ϕ-risk over *all functions* and $R_\phi(f)$ is close to R^*_ϕ, then does it imply that $R(f)$ is close to Bayes risk R^*?

The last question is easy to answer in a special case when ϕ satisfies the (rather natural) condition

$$\phi(t) \geq I_{\{t\leq 0\}}, \quad \forall t. \tag{5.1.4}$$

Then for every f it holds $R(f) \leq R_\phi(f)$. Hence, if (5.1.4) holds and $R_\phi(f)$ is close to zero, then also $R(f)$ is close to zero. But Bayes risk being close to zero is more an expectation as a rule, and general case needs further analysis.

Classification-calibrated ϕ. From now on, let us denote

$$\eta(x) := \mathbf{P}(Y = 1|X = x) = p(1|x).$$

Hence Bayesi classifier is

$$g^*(x) = \mathrm{sgn}(\eta(x) - 0.5).$$

For any f the ϕ-risk is, thus

$$R_\phi(f) = E\big(E[\phi(Yf(X))|X]\big) = E\big(\phi(f(X))\eta(X) + \phi(-f(X))(1-\eta(X))\big).$$

Just like in case of (classification risk) $R(g)$, in order to minimize ϕ-risk over *all possible functions*, we have to minimize *conditional ϕ-risk*

$$E\big[\phi(Yf(X))|X=x\big] = \eta(x)\phi(f(x)) + \phi(-f(x))(1-\eta(x))$$

over all possible values of $f(x)$, hence over $\mathbb{R}$. Since the minimization depends on x via $\eta(x)$, we can fix $\eta \in [0,1]$. The minimal conditional ϕ-risk given η is thus

$$H(\eta) := \inf_{r\in\mathbb{R}} \big(\eta\phi(r) + \phi(-r)(1-\eta)\big)$$

and

$$R^*_\phi := \inf_f R_\phi(f) = E\big(H(\eta(X))\big).$$

Define

$$r(\eta) := \arg \min_{r\in[-\infty,\infty]} \big(\eta\phi(r) + \phi(-r)(1-\eta)\big)$$

Hence the function

$$f^*(x) := r(\eta(x))$$

minimizes ϕ-risk over all functions. Hence $f^*(x)$ is the best function in the sense of ϕ-risk. But is it the best in the sense of classification? In other words, is $\text{sgn}f^*(x)$ the Bayes classifier? From the definition of $f^*(x)$, it follows that this is so, when the following condition is fulfilled

$$\text{sgn}(r(\eta)) = \text{sgn}(\eta - 0.5), \quad \forall \eta \neq 0.5. \tag{5.1.5}$$

The condition (5.1.5) presupposes the existence and uniqueness of r. Therefore, it is usually restated as follows

$$H^-(\eta) > H(\eta), \quad \forall \eta \neq 0.5, \tag{5.1.6}$$

where $H^-(\eta)$ is the minimum of $r \mapsto \eta\phi(r) + \phi(-r)(1-\eta)$ over these arguments that have different sign as $(2\eta - 1)$. Formally

$$H^-(\eta) := \inf_{r\in\mathbb{R}:r(2\eta-1)\leq 0} \big(\eta\phi(r) + \phi(-r)(1-\eta)\big).$$

Clearly $H^-(\eta) \geq H(\eta)$. Also note that the functions H and H^- depend on ϕ, only. This justifies the following definition

Definition 5.1.1 *If (5.1.6) holds, then the function ϕ is* **classification-calibrated**.

Note: if for any η, there exists (possibly not unique) $r(\eta)$, then (5.1.6) guarantees that any such $r(\eta)$ satisfies (5.1.5).

Exercise:

- Prove that $H^-(\frac{1}{2}) = H(\frac{1}{2})$.
- Prove that $H : [0,1] \to \mathbb{R}^+$ is concave and symmetric with respect to 0.5, i.e.
$$H(\eta) = H(1-\eta).$$
Hence $H(\eta) \leq H(0.5)$.
- Prove that $H^- : [0,1] \to \mathbb{R}^+$ is symmetric with respect to 0.5, i.e.
$$H^-(\eta) = H^-(1-\eta).$$
- Prove that H^- is concave in $[0, \frac{1}{2}]$ and in $[\frac{1}{2}, 1]$.
- Prove that if ϕ is convex, then $H(\frac{1}{2}) = \phi(0)$.

For every ϕ, thus $H(\eta) \leq H(0.5)$. It can be shown ([15], Lemma 4) that when ϕ is classification-calibrated, then the inequality is strict for $\eta \neq 0.5$, i.e. if ϕ is classification-calibrated, then
$$H(\eta) < H(0.5), \quad \forall \eta \neq 0.5. \tag{5.1.7}$$

Examples.

- Let $\phi(t) = (1-t)_+$. Then the function
$$\eta(1-r)_+ + (1-\eta)(1+r)_+$$
achieves the minimum at $+1$ or at -1. Hence
$$H(\eta) = 2\min(\eta, 1-\eta), \quad r(\eta) = \begin{cases} 1, & \text{if } \eta > 0.5; \\ -1, & \text{if } \eta < 0.5. \end{cases}$$
Show that $H^-(\eta) = 1$. Thus ϕ is classification calibrated.
- Exercise: Let $\phi(t) = (1-t)_+^2$. Prove that
$$H(\eta) = 4\eta(1-\eta), \quad r(\eta) = 2\eta - 1.$$
Is ϕ classification-calibrated?
- Exercise: Let $\phi(t) = \exp[-t]$. Prove that
$$r(\eta) = \frac{1}{2}\ln\left(\frac{\eta}{1-\eta}\right), \quad H(\eta) = 2\sqrt{\eta(1-\eta)}.$$
Is ϕ classification-calibrated?
- Exercise: Let $\phi(t) = I_{(-\infty,0]}$. Prove that
$$H(\eta) = \min(\eta, 1-\eta), \quad H^-(\eta) = \max(\eta, 1-\eta).$$
Is ϕ classification-calibrated?

Convex ϕ. The following statement gives an easy criterion to check whether ϕ is classification-calibrated for *convex* ϕ. For the proof, see [15], Lemma 5.

Proposition 5.1.1 *A convex ϕ is classification-calibrated if and only if $\phi'(0) < 0$.*

Corollary 5.1.1 *Let ϕ be convex and classification-calibrated. Then, for every $\eta \in [0,1]$,*

$$H^-(\eta) = \phi(0).$$

Proof. Since ϕ is convex, it follows $\eta\phi(r) + (1-\eta)\phi(r) \geq \phi(r(2\eta-1))$. Hence

$$\min_{r:r(2\eta-1)\leq 0} \big(\eta\phi(r) + \phi(-r)(1-\eta)\big) \geq \min_{r:r(2\eta-1)\leq 0} \phi\big(r(2\eta-1)\big) = \min_{u\leq 0} \phi(u) = \phi(0),$$

since $\phi'(0) < 0$ implies that ϕ is decreasing in $(-\infty, 0]$. It follows that $H^- = \phi(0)$. ■

The function ψ. Let $\psi : [0,1] \to \mathbb{R}^+$,

$$\psi(t) := H^-(\frac{1+t}{2}) - H(\frac{1+t}{2})$$

<u>Properties of ψ:</u>

- $\psi(t) \geq 0$;
- $\psi(0) = 0$;
- ψ is continuous, since H and H^- are continuous (concave and piecewise concave).

If ϕ is convex and classification-calibrated, then by Corollary 5.1.1, $H^-(\eta) = \phi(0)$, hence

$$\psi(t) = \phi(0) - H(\frac{1+t}{2})$$

Hence, if ϕ is convex and classification-calibrated, then (in addition to the previous properties):

- ψ is convex (since $H(\frac{1+t}{2})$ is concave);
- $\psi(t) > 0$ iff $t > 0$ (follows from (5.1.7)).

Examples.

- Let $\phi(t) = (1-t)_+$. Then $H(\eta) = 2\min\{\eta, 1-\eta\}$

$$\psi(t) = \phi(0) - H(\frac{1+t}{2}) = 1 - 2\min(\frac{1+t}{2}, 1 - \frac{1-t}{2}) = 1 - (1-t) = t,$$

- Let $\phi(t) = (1-t)_+^2$. Then $H(\eta) = 4\eta(1-\eta)$

$$\psi(t) = \phi(0) - H(\frac{1+t}{2}) = 1 - 4(\frac{1+t}{2})(\frac{1-t}{2}) = 1 - (1-t^2) = t^2.$$

- Let $\phi(t) = \exp[-t]$. Then $H(\eta) = 2\sqrt{\eta(1-\eta)}$,

$$\psi(t) = 1 - 2\sqrt{(\frac{1+t}{2})(\frac{1-t}{2})} = 1 - \sqrt{1-t^2}.$$

- Let $\phi(t) = I_{(-\infty,0]}$. Then $H(\eta) = \min(\eta, 1-\eta)$, $H^-(\eta) = \max(\eta, 1-\eta)$ and

$$\psi(t) = \frac{1+t}{2} - \frac{1-t}{2} = t.$$

The importance of ψ will be apparent from the following theorem (for the proof, see [15], Thm 3) that relates the ϕ-risk to the classification risk.

Theorem 5.1.2 (Bartlett, Jordan, McAuliffe, 2004) *Let ϕ be convex and classification-calibrated. Then for every f*

$$\psi\big(R(f) - R^*\big) \leq R_\phi(f) - R^*_\phi \tag{5.1.8}$$

The theorem is the missing link between ϕ-risk and classification risk; it follows that for any sequence f_n to prove $R(f_n) \to R^*$ (consistency), it suffices to prove the convergence of ϕ-risks: $R_\phi(f_n) \to R^*_\phi$, given ϕ is convex and classification-calibrated. Indeed, ψ is continuous and if ϕ is convex and classification-calibrated, then $\psi(t) > 0$ iff $t > 0$. Thus $\psi(t_n) \to 0$ implies that $t_n \to 0$.

References: About the ϕ-risk theory, read the papers of P. Bartlett *et al.*, for instance [15, 16].

5.2 AdaBoost

5.2.1 Boosting and AdaBoost: the principles

Boosting is based on observation that finding many rough rules of thumb can be a lot easier that finding a single, highly accurate prediction rule. The **boosting** method starts with a simple method or rough classifier of thumb called **weak** or **base learner**. Each time t it is called, the weak learner generates a *base classifier* $h_t : \mathbb{R}^d \to \{-1, 1\}$ and after T rounds, the boosting algorithm must combine these base classifiers $h_1, \ldots, h_T$ into a single prediction rule:

$$g(x) = \text{sgn}\big(\sum_{t=1}^{T} \alpha_t h_t(x)\big), \tag{5.2.1}$$

where $\alpha_t \in \mathbb{R}$ are the weights. Finding base classifiers h_t is iterative: with the help of h_t, the new sample weights $D_{t+1}(i), i = 1, \ldots, n$ is designed. This new sample is used to train h_{t+1} and so on. Although every base classifer can be hardly better than random guess, their combination (5.2.1) might perform remarkably well.

First boosting type of algorithms were introduced in the end of 1980.-s (Kearns, Valiant, Schapire); in 1995, Y. Freud and R. Schapire introduced the most popular and well-known boosting algorithm called AdaBoost (adaptive boosting).

AdaBoost: Choose a weak learner (the set of base classifiers $\mathcal{H}$).

1. **Input:** Sample $S = (x_1, y_1), \dots, (x_n, y_n)$; the number of iterations T.
2. **Initialize:** $D_1(i) = \frac{1}{n}, \quad i = 1, \dots, n$;
3. **Do for** $t = 1, \dots, T$**:**

   a Train a base classifier
   $$h_t : \mathcal{X} \to \{-1, 1\}$$
   with respect to the weighted sample (S, D_t).

   b Calculate the *weighted training error*:
   $$\epsilon_t := \sum_{i=1}^{n} D_t(i) I_{\{y_i \neq h_t(x_i)\}}.$$

   c Set
   $$\alpha_t := \frac{1}{2} \ln \frac{1 - \epsilon_t}{\epsilon_t}.$$

   d Update the weights
   $$D_{t+1}(i) := \frac{D_t(i) \exp[-\alpha_t y_i h_t(x_i)]}{Z_t},$$
   where Z_t is a normalization constant so that $\sum_i D_{t+1}(i) = 1$.
4. **Break if** $\epsilon_t = 0$ or $\epsilon_t \geq \frac{1}{2}$; in this case set $T = t - 1$.
5. **Output:**
   $$g_T(x) := \operatorname{sgn}\big(f_T(x)\big), \quad \text{where } f_T = \sum_{t=1}^{T} \alpha_t h_t.$$

Thus, h_1 is trained based on the original sample, because $D_1(i) = \frac{1}{n}$. The weight α_t of the base classifier h_t is found based on the training error ϵ_t. If $\epsilon_t \leq 0.5$, then $\alpha_t \geq 0$. The smaller error ϵ_t, the bigger weight α_t. After training h_t and finding α_t, new weights $D_{t+1}(i)$ are calculated: for every i, the old weight $D_t(i)$ is multiplied by

$$\exp[-\alpha_t y_i h_t(x_i)] = \begin{cases} e^{\alpha_t} > 1, & \text{if } h_t \text{ misclassifies } x_i \text{ ;} \\ e^{-\alpha_t} < 1, & \text{if } h_t \text{ classifies } x_i \text{ correctly.} \end{cases}$$

Hence, the misclassified vectors will have larger weight and by training h_{t+1}, these vectors count more. New weights also depend on α_t: the bigger α_t i.e. the more correct is h_t, the bigger is the relative weight of misclassified vectors.

Note that

$$-y_i h_t(x_i) = 2I_{\{y_i \neq h_t(x_i)\}} - 1,$$

hence

$$\exp[-\alpha_t y_i h_t(x_i)] = \exp[2\alpha_t I_{\{y_i \neq h_t(x_i)\}}] \exp[-\alpha_t]$$

and the new weights can be defined as (e.g. [7])

$$D_{t+1}(i) := \frac{D_t(i) \exp[2\alpha_t I_{\{y_i \neq h_t(x_i)\}}]}{Z_t}. \tag{5.2.2}$$

5.2.2 AdaBoost and exponential loss

Let us show that AdaBoost minimizes empirical ϕ-risk over a class $\mathcal{F}$, where

$$\phi(t) = \exp[-t] \text{ and } \mathcal{F} := \Big\{ \sum_{t=1}^{T} \alpha_t h_t : \alpha_t \in \mathbb{R},\, h_t \in \mathcal{H} \Big\}.$$

Here $\mathcal{H}$ stands for the set of all base-classifiers. The empirical ϕ-risk is, thus,

$$\frac{1}{n}\sum_{i=1}^{n} \exp[-y_i f(x_i)] = \frac{1}{n}\sum_{i=1}^{n} \exp[-y_i(\alpha_1 h_1(x_i) + \cdots + \alpha_T h_t(x_i))], \tag{5.2.3}$$

and minimization is over all $\alpha_1, \ldots, \alpha_T$ and $h_1, \ldots, h_T$.

Minimizing (5.2.3) over all functions in form $\alpha_1 h_1 + \cdots + \alpha_T h_T$ is complicated and AdaBoost uses so-called *forward stagewise modeling:* at first the function

$$\sum_{i=1}^{n} \exp[-y_i \alpha h(x_i)]$$

is minimized over $h_1 \in \mathcal{H}$ and $\alpha_1 \in \mathbb{R}$.
On round t, AdaBoost seeks for α_t and $h_t \in \mathcal{H}$ such that

$$(h_t, \alpha_t) = \arg\min_{\alpha,h} \sum_{i=1}^{n} \exp[-y_i(f_{t-1}(x_i) + \alpha h(x_i))], \tag{5.2.4}$$

where

$$f_{t-1} = \sum_{s=1}^{t-1} \alpha_s h_s, \quad f_0 = 0.$$

Hence AdaBoost minimizes (5.2.3) in a greedy manner: at first $\alpha_1 h_1$ is found, then, given $\alpha_1 h_1$, the best $\alpha_2 h_2$ is found; then, given $\alpha_1 h_1 + \alpha_2 h_2$, the best $\alpha_3 h_3$ is found and so on.

Proof. Let us show this. At fist recall

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \exp[-\alpha_t y_i h_t(x_i)], \quad Z_t = \sum_i D_t(i) \exp[-\alpha_t y_i h_t(x_i)].$$

Thus, for every i,

$$D_1(i) = \frac{1}{n},\ D_2(i) = \frac{\exp[-\alpha_1 y_i h_1(x_i)]}{nZ_1},\ D_3(i) = \frac{\exp[-y_i(\alpha_1 h_1(x_i) + \alpha_2 h_2(x_i))]}{nZ_1 Z_2},$$
$$D_t(i) = \frac{\exp[-y_i(\alpha_1 h_1(x_i) + \alpha_2 h_2(x_i) + \cdots + \alpha_{t-1} h_{t-1}(x_i))]}{Z_0 Z_1 \cdots Z_{t-1}} = \frac{\exp[-y_i f_{t-1}(x_i)]}{Z_0 Z_1 Z_2 \cdots Z_{t-1}},$$

where $Z_0 := n$. Hence

$$\exp[-y_i f_{t-1}(x_i)] = \prod_{s=0}^{t-1} Z_s D_t(i) = n \prod_{s=1}^{t-1} Z_s D_t(i). \tag{5.2.5}$$

From (5.2.5), it follows

$$\begin{aligned}\sum_{i=1}^n \exp[-y_i(f_{t-1}(x_i) + \alpha h(x_i))] &= \sum_{i=1}^n \exp[-y_i(f_{t-1}(x_i))] \exp[-y_i \alpha h(x_i)] \\ &= \prod_{s=0}^{t-1} Z_s \Big(\sum_{i=1}^n D_t(i) \exp[-y_i \alpha h(x_i)] \Big).\end{aligned}$$

Consider the problem

$$\min_{\alpha,h} \sum_{i=1}^n D_t(i) \exp[-y_i \alpha h(x_i)]. \tag{5.2.6}$$

Since

$$\begin{aligned}\sum_{i=1}^n D_t(i) \exp[-y_i \alpha h(x_i)] &= e^{-\alpha} \sum_{i:h(x_i)=y_i} D_t(i) + e^{\alpha} \sum_{i:h(x_i)\neq y_i} D_t(i) \\ &= (e^{\alpha} - e^{-\alpha}) \sum_{i=1}^n D_t(i) I_{\{y_i \neq h(x_i)\}} + e^{-\alpha} \sum_{i=1}^n D_t(i),\end{aligned}$$

it follows that given α, minimizing (5.2.6) over h is equivalent to minimizing weighted empirical risk:

$$\min_{h\in\mathcal{H}} \sum_{i=1}^n D_t(i) I_{\{h(x_i)\neq y_i\}}. \tag{5.2.7}$$

Hence the solution does not depend on α, so that if (α_t, h_t) is the solution of (5.2.6), then

$$h_t = \arg\min_{h\in\mathcal{H}} \sum_{i=1}^n D_t(i) I_{\{h(x_i)\neq y_i\}}. \tag{5.2.8}$$

Plugging this h_t into (5.2.6) and minimizing over α, one gets easily that

$$\alpha_t := \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}. \tag{5.2.9}$$

Exercise: Prove (5.2.9).

Remark. Hence AdaBoost minimizes empirical ϕ-risk if the base classifier is obtained by minimizing (re-weighted) empirical risk, i.e. h_t is as in (5.2.8). In its general form, AdaBoost does not require that, but any other method can be considered as a substitute of ERM-principle. Often AdaBoost is defined so that h_t is obtained via ERM-principle, i.e. h_t is obtained as in (5.2.8).

5.2.3 Empirical risk

Upper bound to empirical risk. Let f_T be the outcome of AdaBoost, i.e.

$$f_T = \alpha_1 h_1 + \cdots + \alpha_T h_T.$$

Recall once again

$$Z_t = \sum_i D_t(i) \exp[-\alpha_t y_i h_t(x_i)]$$

and recall (5.2.5): for every $t \geq 1$,

$$\exp[-y_i f_t(x_i)] = \prod_{s=0}^{t} Z_s D_t(i) = n \prod_{s=1}^{t} Z_s D_{t+1}(i).$$

Hence,

$$(n \prod_{t=1}^{T} Z_t) = (n \prod_{t=1}^{T} Z_t) \sum_i D_{T+1}(i) = \sum_i \exp[-y_i f_T(x_i)],$$

so that we obtain a nice estimate to empirical risk (training error)

$$R_n(f_T) = \frac{1}{n} |\{i : \operatorname{sgn}(f_T(x_i)) \neq y_i\}| \leq \frac{1}{n} \sum_i \exp[-y_i f_T(x_i)] = \prod_{t=1}^{T} Z_t.$$

The obtained inequality suggests that empirical risk can be reduced most rapidly (in a greedy way) by choosing α_t and h_t on each round to minimize

$$Z_t = \sum_i D_t(i) \exp[-\alpha_t y_i h_t(x_i)],$$

and this is exactly what AdaBoost does.

Exercise: Prove

$$Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}. \tag{5.2.10}$$

From (5.2.10), it follows

$$R_n(f_T) \leq \prod_{t=1}^{T} Z_t = \prod_{t=1}^{T} 2\sqrt{\epsilon_t(1-\epsilon_t)} = \prod_{t=1}^{T} \sqrt{1-4u_t^2} \leq \exp[-2\sum_{t=1}^{T} u_t^2], \tag{5.2.11}$$

where

$$u_t := \frac{1}{2} - \epsilon_t.$$

The number u_t measures how much h_t is better than random guess or majority label rule. Thus, when $\sum_{t=1}^{\infty} u_t^2 = \infty$, the training error approach to zero meaning that it is zero eventually. If there exists $u_o > 0$ so that $u_t \geq u_o$ for every t, then

$$R_n(f_T) \leq e^{-(2u_o^2)T}$$

so that empirical risk drops down exponentially fast. Whether such a u_o exists, depends on sample and the base learner (set $\mathcal{H}$). For example, if $\mathcal{H}$ is the set of linear classifiers, then for any distribution over a training set of distinct points $u_o > \frac{c}{n}$, where $c > 0$ is an universal constant [17, 18].

Upper bound to the proportion of margin errors. Note that AdaBoost algorithm might not terminate, when empirical risk (training error) is zero. Indeed, $R_n(f_t) = 0$ does not imply that ϵ_t is zero, since ϵ_t depends on the base classifier h_t, only. Hence AdaBoost can go on after the training sample is separated. Does it mean overfitting? It turns out that AdaBoost continues to increase functional margin and, as it will be (partially) explained in the next section, it does not necessarily mean overfitting.

In AdaBoost, all weights are non-negative, i.e. $\alpha_t \geq 0$. Hence, the classifier $g_T = \text{sgn} f_T$ will remain unchanged, if the weights are rescaled so that they sum up to one. Hence, we can divide f_T by the sum $\sum_{t=1}^{T} \alpha_t$, and so we define the *(functional) margin* of f_T at pair (x_i, y_i), as

$$\rho_i(f_T) := \frac{y_i f_T(x_i)}{\sum_{t=1}^{T} \alpha_t} = y_i\big(\sum_{t=1}^{T} \beta_t f_t(x_i)\big), \quad \beta_t := \frac{\alpha_t}{\sum_{t=1}^{T} \alpha_t}. \tag{5.2.12}$$

The next theorem shows how the number of iterations affects the upper bound to the proportion of margin errors. In the next section, we see how the proportion of margin errors, in turn, affects the risk bound.

Theorem 5.2.1 *Let f_T be the output of AdaBoost. Then, for every $\gamma \geq 0$,*

$$\frac{1}{n}|\{i : \rho_i(f_T) \leq \gamma\}| \leq \prod_{t=1}^{T} 2\sqrt{\epsilon_t^{1-\gamma}(1-\epsilon_t)^{1+\gamma}} = \prod_{t=1}^{T} (1-2u_t)^{\frac{1-\gamma}{2}}(1+2u_t)^{\frac{1+\gamma}{2}}, \tag{5.2.13}$$

where $\rho_i(f_T)$ is defined as in (5.2.12).

Proof. Note

$$\rho_i(f_T) \leq \gamma \quad \Leftrightarrow \quad y_i f_T(x_i) - \gamma \sum_i \alpha_i \leq 0 \quad \Leftrightarrow \quad \exp[\gamma \sum_i \alpha_i] \exp[-y_i f_T(x_i)] \geq 1.$$

Hence

$$\exp[\gamma \sum_i \alpha_i] \exp[-y_i f_T(x_i)] \geq I_{(-\infty,\gamma]}(\rho_i(f_T)) \tag{5.2.14}$$

Sum both sides of (7.3.24) over i to obtain

$$\exp[\gamma \sum_i \alpha_i] \frac{1}{n} \sum_i \exp[-y_i f_T(x_i)] \geq \frac{1}{n} |\{i : \rho_i(f_T) \leq \gamma\}|.$$

Using $(n \prod_{t=1}^T Z_t) = \sum_i \exp[-y_i f_T(x_i)]$, one gets

$$\frac{1}{n} |\{i : \rho_i(f_T) \leq \gamma\}| \leq \prod_{t=1}^T Z_t \exp[\gamma \sum_i \alpha_i].$$

From the definition of α_t:

$$\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t},$$

it follows

$$\exp[\gamma \sum_i \alpha_i] = \prod_{t=1}^T (\frac{1-\epsilon_t}{\epsilon_t})^{\frac{\gamma}{2}}.$$

We know that $Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$, so that

$$\prod_{t=1}^T Z_t \exp[\gamma \sum_i \alpha_i] = \prod_{t=1}^T 2 (\frac{1-\epsilon_t}{\epsilon_t})^{\frac{\gamma}{2}} \big(\epsilon_t(1-\epsilon_t)\big)^{\frac{1}{2}} = \prod_{t=1}^T 2\sqrt{\epsilon_t^{1-\gamma}(1-\epsilon_t)^{1+\gamma}}.$$

The last inequality in (5.2.13) follows from the definiton of u_t: $u_t = \frac{1}{2} - \epsilon_t$. ■

Remark. With $\gamma = 0$, we now get the first inequality of (5.2.11):

$$R_n(f_T) = \frac{1}{n} |\{i : \operatorname{sgn}(f_T(x_i)) \neq y_i\}| \leq \frac{1}{n} |\{i : \rho_i(f_T) \leq 0\}| \leq \prod_{t=1}^T 2\sqrt{\epsilon_t(1-\epsilon_t)}.$$

5.2.4 Risk bounds for AdaBoost

Let $f : \mathbb{R}^d \to \mathbb{R}$ and recall the quantity $A_n(f)$ from Subsection 4.3:

$$A_n(f) = \frac{1}{n} \sum_{i=1}^n \phi'(-f(x_i)y_i), \quad \phi'(t) = \begin{cases} 1, & \text{if } t \geq 0; \\ 1 + \frac{t}{\gamma}, & \text{if } -\gamma \leq t \leq 0; \\ 0, & \text{if } t < -\gamma \end{cases}$$

Here $\gamma > 0$ is fixed. Clearly $A_n(f)$ is nothing but empirical ϕ-risk, where $\phi(t) = \phi'(-t)$. Also recall that $A_n(f)$ is upper bounded by the proportion of margin errors $R_n^\gamma(f)$:

$$A_n(f) \leq \frac{1}{n}\sum_{i=1}^{n} I_{\{f(x_i)y_i \leq \gamma\}} = R_n^\gamma(f).$$

Our object of interest is the (classification) risk

$$R(f_T) = \mathbf{P}\big(Y \neq \mathrm{sgn}(f_T)\big), \quad \text{where } f_T = \sum_{t=1}^{T} \alpha_i h_t \text{ is the outcome of AdaBoost.}$$

As explained in the previous section, the classifier $\mathrm{sgn}(f_T)$ and hence the risk $R(f_T)$ remains same if the weights α_t are normalized so that they sum up to one. Thus, we are interested in uniform risk bounds to risk $R(f)$ over the class

$$\mathrm{co}_T(\mathcal{H}) := \Big\{ \sum_{t=1}^{T} \beta_t h_t : \beta_t \geq 0, \quad \sum_{t=1}^{T} \beta_t = 1, \quad h_t \in \mathcal{H} \Big\}$$

and $\mathcal{H}$ is the set of $\{1,-1\}$-valued base-classifiers. Note that due to the rescaling, all functions in $\mathcal{F}$ are $[-1,1]$-valued.

Perhaps the easiest way to obtain the risk bounds, would be using the VC-dimension of class of corresponding classifiers

$$\mathcal{G}_T := \{\mathrm{sgn}(f) : f \in \mathrm{co}_T(\mathcal{H})\}.$$

Unfortunately, the VC dimension of $\mathcal{G}_T$ increases with T. It turns out that using some more refined methods, the risk bounds that are *independent of the number of iterations* T can be found. An example of such a bound is as follows (see [18], Cor 1). The bound depends on VC-dimension of $\mathcal{H}$ and since the set of base-classifiers are simple, the VC-dimension of $\mathcal{H}$ is typically small.

Theorem 5.2.2 *Let $\gamma > 0$ be fixed. Then, with probability at least $1-\delta$ (over samples of length n) every $f \in \mathrm{co}_T(\mathcal{H})$ satisfies*

$$R(f) \leq A_n(f) + \frac{8}{\gamma}\sqrt{\frac{2V_{\mathcal{H}}\ln(n+1)}{n}} + \sqrt{\frac{\ln\frac{2}{\delta}}{2n}}, \tag{5.2.15}$$

where $V_{\mathcal{H}}$ is the VC-dimension of $\mathcal{H}$.

Note that for any $f \in \mathrm{co}_T(\mathcal{H})$, the functional margin $y_i f(x_i)$ is $\rho_i(f)$ so that

$$A_n(f) \leq R_n^\gamma(f) = \frac{1}{n}|\{i : \rho_i(f) \leq \gamma\}| \leq \prod_{t=1}^{T} 2\sqrt{\epsilon_t^{1-\gamma}(1-\epsilon_t)^{1+\gamma}} = \prod_{t=1}^{T}(1-2u_t)^{\frac{1-\gamma}{2}}(1+2u_t)^{\frac{1+\gamma}{2}},$$

where the last inequality follows from Theorem 5.2.1. Hence, when $u_t \geq u_o > 0$ and γ is small enough, then quite contra-intuitively, we get a risk bound that *decreases in* T. This implies that the risk bound can also decrease after the training error has become zero. This (partially) explains, why AdaBoost does not encounter overfitting after many iterations.

5.2.5 Some insights to (5.2.15)

Rademacher complexity. The central concept for proving bounds like (5.2.15) is a complexity measure called **Rademacher complexity (Rademacher average)**. For a class of functions $\mathcal{F}$ from $\mathbb{R}^d \to \mathbb{R}$, the Rademacher complexity $\mathcal{R}_n(\mathcal{F})$ is defined as follows

$$\mathcal{R}_n(\mathcal{F}) := E \sup_{\mathcal{F}} \Big|\frac{1}{n}\sum_{i=1}^n \sigma_i f(X_i)\Big|,$$

where $X_1, \ldots, X_n$ are iid random feature vectors and σ_i, $i = 1, \ldots, n$ are iid random variables with law $\mathbf{P}(\sigma_i = 1) = \mathbf{P}(\sigma_i = -1) = 0.5$. The random variables σ_i and random vectors X_i, $i = 1, \ldots, n$ are independent and the expectation is taken over X_i's as well as over σ_i's. Some authors e.g. the ones of [11] multiply everything by 2 in the definition of Rademacher complexity.

If $\mathcal{G}$ is class of classifiers $\mathbb{R}^d \to \{-1, 1\}$, then using Rademacher complexity $\mathcal{R}_n(\mathcal{G})$, one gets more refined generalization (risk) bounds as with VC-dimension, namely the following theorem holds (see. e.g. [4], Thm 5; see also [11], Thm 4.9 or [3], Thm 3.2).

Theorem 5.2.3 *Let $\mathcal{G}$ be a class of classifiers. For all $\delta > 0$, with probability at least $1 - \delta$*

$$R(g) \le R_n(g) + \mathcal{R}_n(\mathcal{G}) + \sqrt{\frac{\ln \frac{1}{\delta}}{2n}}. \tag{5.2.16}$$

Unfortunately, Rademacher complexity is hard to estimate, hence obtained bound is often not very practical. Rademacher complexity can be upper bounded by VC-dimension of $\mathcal{G}$ as follows ([3], eq (6))

$$\mathcal{R}_n(\mathcal{G}) \le 2\sqrt{\frac{2V \ln(n+1)}{n}}. \tag{5.2.17}$$

Margin bound. The key inequality for large margin bounds is the following bound ([18], Thm 3; [19]; see also [3], Thm 4.1).

Theorem 5.2.4 *Let $\mathcal{F}$ be the class of functions from $\mathbb{R}^d \to [-1, 1]$ and let $\gamma \in (0, 1)$. Let $\delta > 0$. Then with probability $1 - \delta$, every $f \in \mathcal{F}$ satisfies*

$$R(f) \le A_n(f) + \frac{4\mathcal{R}_n(\mathcal{F})}{\gamma} + \sqrt{\frac{\ln \frac{2}{\delta}}{2n}}. \tag{5.2.18}$$

To apply (5.2.18) for the output of AdaBoost, we take $\mathcal{F} = \mathrm{co}_T(\mathcal{H})$ (note that every f in $\mathrm{co}_T(\mathcal{H})$ takes values in $[-1, 1]$). The usefulness of Rademacher complexity bound comes from the fact that unlike VC dimension, the Rademacher complexity of $\mathrm{co}_T(\mathcal{H})$ does not depend on T, moreover, it holds ([18, 20])

$$\mathcal{R}_n\big(\mathrm{co}_T(\mathcal{H})\big) = \mathcal{R}_n(\mathcal{H}).$$

Hence, $\mathcal{R}_n(\mathcal{F})$ in (5.2.18) can be replaced by $\mathcal{R}_n(\mathcal{H})$, which, in turn, can now estimate above by VC dimension of $\mathcal{H}$ using (5.2.17). Hence, we get that

$$\mathcal{R}_n(\mathcal{F}) = \mathcal{R}_n(\mathcal{H}) \leq 2\sqrt{\frac{2V_{\mathcal{H}} \ln(n+1)}{n}}.$$

Plugging the obtained bound into (5.2.18), we obtain (5.2.15).

5.2.6 Consistency of AdaBoost

Using their theory of ϕ-risk, P. Bartlett and M. Traskin proved that when base learner is suitable and the number of iterations goes to zero with suitable speed, then AdaBoost is strongly consistent [21]. An important condition for consistency is that h_t is chosen by using ERM-principle, i.e.

$$h_t = \arg\min_{h \in \mathcal{H}} \sum_{i=1}^{n} D_t(i) I_{\{h(x_i) \neq y_i\}}.$$

We already know that in this case h_t and α_t minimize

$$\sum_{i=1}^{n} \exp[-y_i(f_{t-1}(x_i) + \alpha h(x_i))]$$

over $\alpha \in \mathbb{R}$ and $h \in \mathcal{H}$, so that for every t, the function $f_t = f_{t-1} + \alpha_t h_t$ is

$$\sum_{i=1}^{n} \exp[-y_i f_t(x_i)] = \inf_{\alpha,h} \sum_{i=1}^{n} \exp[-y_i(f_{t-1}(x_i) + \alpha h(x_i))].$$

Hence, with ERM-principle, AdaBoost can be defined as follows.

AdaBoost: Choose a weak learner (the class of base classifiers $\mathcal{H}$).

1. **Input:** Sample $S = (x_1, y_1), \dots, (x_n, y_n)$; the number of iterations T.

2. **Initialize:** $f_1 \equiv 0$;

3. **Do for** $t = 1, \dots, T$ define

$$f_t = f_{t-1} + \alpha_t h_t,$$

   where

$$\sum_{i=1}^{n} \exp[-y_i f_t(x_i)] = \inf_{\alpha \in \mathbb{R}, h \in \mathcal{H}} \sum_{i=1}^{n} \exp[-y_i(f_{t-1}(x_i) + \alpha h(x_i))].$$

4. **Output:**

$$g_T = \text{sgn} f_T, \quad f_T = \sum_{t=1}^{T} \alpha_t h_t.$$

Let, for a $\lambda > 0$,

$$\mathcal{F}_\lambda := \Big\{\sum_{i=1}^{m} \lambda_i h_i, \quad m = 0,1,2,\ldots, \quad \sum_{i=1}^{m} \lambda_i = \lambda, \quad h_i \in \mathcal{H}\Big\}, \quad \mathcal{G}_\lambda := \{\mathrm{sgn} f : f \in \mathcal{F}_\lambda\}.$$

Clearly any output $f_T \in \mathcal{G}_\lambda$ for some λ. The consistency is proven via the inequality (5.1.8)

$$\psi\big(R(f_{T_n}) - R^*\big) \leq R_\phi(f_{T_n}) - R^*_\phi,$$

where, $\phi(t) = \exp[-t]$, $\psi(t) = 1 - \sqrt{1-t^2}$ and T_n is suitably chosen. Hence the goal is to prove the consistency of ϕ-risks:

$$R_\phi(f_{T_n}) \to R^*_\phi.$$

For that, the approximation error has to tend to zero, so the following condition is necessary:

$$\lim_{\lambda\to\infty} \inf_{g\in\mathcal{G}_\lambda} R_\phi(f) = R^*_\phi. \tag{5.2.19}$$

The property (5.2.19) depends on the unknown distribution $F(x,y)$ and on $\mathcal{H}$. For many classes $\mathcal{H}$, (5.2.19) is satisfied for all possible distributions.

Theorem 5.2.5 (Bartlett, Trashkin, 2007). *Assume that the VC dimension of $\mathcal{H}$ is finite, $R^* > 0$ and (5.2.19) holds. Let T_n be such that $T_n \to \infty$ and $T_n = O(n^\nu)$, where $\nu < 1$. Then*

$$R(f_{T_n}) \to R^*, \quad \text{a.s.},$$

i.e, the rule $\{sgn(f_{T_n})\}$ is strongly consistent.

5.3 GradientBoost

AdaBoost uses forward stagewise modeling to minimize empirical ϕ-risk

$$R^n_\phi(f) = \sum_i \phi(y_i f(x_i))$$

over a set of functions. Here, $\phi(t) = \exp[-t]$. We shall show that another way to look at AdaBoost is a kind of gradient descent method that at every iteration t looks for $h_t \in \mathcal{H}$ so that the decrease of the function

$$\alpha \mapsto R^n_\phi\Big(\sum_{s=1}^{t-1} \alpha_s h_s + \alpha h_t\Big)$$

at 0 were maximal. This can be considered as a *coordinate descent method*, where the elements of $\mathcal{H}$ are considered as coordinates.

GradientBoost algorithm. Let us find a h_t that for general ϕ ensures the deepest gradient descent at 0:

$$\begin{aligned} h_t &= \arg\min_{h\in\mathcal{H}} \frac{\partial R^n_\phi\big(f_{t-1}+\alpha h\big)}{\partial\alpha}\Big|_{\alpha=0} \\ &= \arg\min_{h\in\mathcal{H}} \sum_i \frac{\partial\phi\Big(y_i\big(f_{t-1}(x_i)+\alpha h(x_i)\big)\Big)}{\partial\alpha}\Big|_{\alpha=0} \\ &= \arg\min_{h\in\mathcal{H}} \sum_i \phi'\big(y_i f_{t-1}(x_i)\big) y_i h(x_i) \\ &= \arg\min_{h\in\mathcal{H}} -\sum_i \big(2I_{\{y_i\neq h(x_i)\}}-1\big)\phi'(y_i f_{t-1}(x_i)). \end{aligned}$$

In case ϕ is decreasing, then $-\phi'\big(y_i f_{t-1}(x_i)\big)\geq 0$ for every i and in this case h_t minimizes weighted empirical risk

$$h_t = \arg\min_{h\in\mathcal{H}} \sum_{i=1}^n D^t_\phi(i) I_{\{y_i\neq h(x_i)\}},$$

where

$$D^t_\phi(i) \propto -\phi'\big(y_i f_{t-1}(x_i)\big). \tag{5.3.1}$$

After finding h_t, let us choose the weight α_t to minimize

$$\alpha \mapsto R^n_\phi\big(f_{t-1}+\alpha h_t\big).$$

For convex ϕ, α_t is the solution of the equation

$$\frac{\partial R^n_\phi\big(f_{t-1}+\alpha h_t\big)}{\partial\alpha}\Big|_{\alpha_t} = \sum_i \phi'\Big(y_i\big(f_{t-1}(x_i)+\alpha_t h_t(x_i)\big)\Big) y_i h_t(x_i) = 0.$$

So we get a general boosting algorithm as follows.

GradientBoost: Choose the weak learner (the class of base classifiers $\mathcal{H}$).

1. **Input:** Sample $S=(x_1,y_1),\dots,(x_n,y_n)$; the number of iterations T.
2. **Initialize:** $D_1(i)=\frac{1}{n}, \quad i=1,\dots,n;$
3. **Do for** $t=1,\dots,T$**:**

   a Train a base classifier

   $$h_t:\mathcal{X}\to\{-1,1\}.$$

   with respect to the weighted sample (S,D_t)

   b Define

   $$\alpha_t := \arg\min_{\alpha\in\mathbb{R}} R^n_\phi\big(f_{t-1}+\alpha h_t\big)$$

c Update the weights

$$D_\phi^{t+1}(i) \propto -\phi'\big(y_i(f_{t-1}(x_i) + \alpha_t h_t(x_i))\big) = -\phi'\big(y_i(f_t(x_i))\big).$$

4. **Output:**

$$g(x) := \operatorname{sgn}\big(\sum_{t=1}^T \alpha_t h_t(x)\big).$$

AdaBoost is GradientBoost. Indeed, if $\phi(t) = e^{-t}$, then (5.3.1):

$$D_\phi^t(i) \propto -\phi'\big(y_i f_{t-1}(x_i)\big) = \exp[-\big(y_i f_{t-1}(x_i)\big)]$$

so that the sample weights are the same as in AdaBoost. The coefficient α_t is the solution of the equation

$$\begin{aligned}\sum_i \phi'\Big(y_i\big(f_{t-1}(x_i) + \alpha_t h_t(x_i)\big)\Big) y_i h_t(x_i) &= -\sum_i \exp[-y_i\big(f_{t-1}(x_i) + \alpha_t h_t(x_i)\big)] y_i h_t(x_i) \\ &= -\sum_i \exp[-y_i f_{t-1}(x_i)] \exp[-y_i \alpha_t h_t(x_i)] y_i h_t(x_i) = 0\end{aligned}$$

Thus the equation for α_t is equivalent to the following equation

$$\begin{aligned}\sum_i D_t(i) \exp[-y_i \alpha_t h_t(x_i)] y_i h_t(x_i) &= \sum_{i: h_t(x_i) \neq y_i} D_t(i) e^{\alpha_t} + \sum_{i: h_t(x_i) = y_i} D_t(i) e^{-\alpha_t} \\ &= \epsilon_t e^{\alpha_t} + (1-\epsilon_t) e^{-\alpha_t} = 0,\end{aligned}$$

and the solution of it is

$$\alpha_t = \frac{1}{2} \ln \frac{1-\epsilon_t}{\epsilon_t}.$$

5.3.1 LogitBoost

If $\phi(t) = \ln(1 + e^{-t})$, then empirical ϕ-risk is

$$R_\phi^n(f) = \sum_i \ln\big(1 + \exp[-y_i f(x_i)]\big) \tag{5.3.2}$$

and it penalizes less large negative margin. In this case

$$\phi'(t) = \frac{-e^{-t}}{1+e^{-t}} = \frac{-1}{1+e^t},$$

so that (5.3.1) is

$$D_\phi^t(i) \propto \frac{1}{1 + e^{y_i f_{t-1}(x_i)}}.$$

A boosting algorithm with these weights is known as **GradientBoost**. Unfortunately, the weights α_t are not analytically known. In the literature the constant weights $\alpha_t \equiv \frac{1}{2}$ or the ones of AdaBoost are used.

LogitBoost and logistic regression. Recall logistic regression: given a class $\mathcal{F}$, the best fit to the ratio

$$\ln \frac{\eta(x)}{1-\eta(x)}$$

from that class was searched. In section 3.6.2, $\mathcal{F}$ was the class of linear functions. If $f \in \mathcal{F}$, then the estimate of unknown η is

$$\eta_f(x) = \frac{e^{f(x)}}{1+e^{f(x)}} = \frac{1}{1+e^{-f(x)}}, \quad 1-\eta_f(x) = \frac{1}{1+e^{f(x)}}. \tag{5.3.3}$$

In logistic regression, the following conditional likelihood was maximized:

$$\begin{aligned}\sum_{i:y_i=1} \ln \eta_f(x_i) + \sum_{i:y_i=-1} \ln(1-\eta_f(x_i)) &= -\sum_{i:y_i=1} \ln(1+e^{-f(x_i)}) - \sum_{i:y_i=-1} \ln(1+e^{f(x_i)}) \\ &= -\sum_i \ln\bigl(1+\exp[-y_i f(x_i)]\bigr).\end{aligned}$$

Hence the logistic regression and LogitBoost minimize the same function (5.3.2) or, equivalently, maximize the same conditional likelihood. The difference between two methods lies in the class $\mathcal{F}$ and the methods of minimization. The Newton-Rapshon method tries to minimize (5.3.2) directly and hence the class $\mathcal{F}$ has to be simple. GradientBoost aims to minimize the objective in a steepest gradient descent way over the class $\mathcal{F} = \text{span}\mathcal{H}$.

Exercise: Let $\phi(t) = \ln(1+\exp[-t])$ and $\phi(t) = \ln(1+\exp[-2t])$. Show that

$$r(\eta) = \ln \frac{\eta}{1-\eta}, \quad r(\eta) = \frac{1}{2} \ln \frac{\eta}{1-\eta}.$$

Sometimes, in LogitBoost, the function $\phi(t) = \ln(1+\exp[-2t])$ is used. Hence, the objective is to minimize

$$\sum_i \ln\bigl(1+\exp[-2y_i f(x_i)]\bigr) \tag{5.3.4}$$

and, according to the Exercise above, it means estimating the following function via (conditional) maximum likelihood from $\mathcal{F}$:

$$\frac{1}{2} \ln \frac{\eta(x)}{1-\eta(x)}.$$

Estimating η. The goal of boosting in classification is the classifier $\text{sgn}(f_T)$. The goal of logistic regression is more general – estimating η. With f_T being the output of LogitBoost (5.3.2), i.e. with $\phi(t) = \ln(1+\exp[-t])$, the estimate of $\eta(x)$ is as in (5.3.3):

$$\hat{\eta}(x) = \frac{e^{f_T(x)}}{1+e^{f_T(x)}} = \frac{1}{1+e^{-f_T(x)}}, \quad 1-\hat{\eta} = \frac{1}{1+e^{f_T(x)}}.$$

With f_T being the output of LogitBoost (5.3.4), i.e. with $\phi(t) = \ln(1+\exp[-2t])$, the estimate of η is

$$\hat{\eta}(x) = \frac{1}{1+e^{-2f_T(x)}}. \tag{5.3.5}$$

5.4 Regression

The boosting type of algorithms can also be used in regression, where $y_i \in \mathbb{R}$ and the objective is to minimize the loss function

$$\sum_{i=1}^{n} L(y_i, f(x_i)) \tag{5.4.1}$$

over a class $\mathcal{F}$. Typically $L(y,x) = (y-x)^2$, and by boosting algorithms

$$\mathcal{F} = \Big\{ \sum_{t=1}^{T} \alpha_t h_t : h_t \in \mathcal{H} \Big\},$$

where $\mathcal{H}$ is a class of simple regression functions. Popular choices are trees (MART – *multiple additive regression trees*, Friedman, 2001), splines (*componentwise smoothing splines*, Bühlmann, Yu, 2003) or just single coordinates x^j (*componentwise linear least squares*). When for every $\alpha \in \mathbb{R}$ and $h \in \mathcal{H}$ also $\alpha h \in \mathcal{H}$, then every $f \in \mathcal{F}$ is just the sum $h_1 + \cdots + h_T$. In boosting, minimizing (5.4.1) goes in a greedy manner – in t-th iteration, one looks for the optimal function h_t and corresponding coefficient α_t to add to the current expansion $f_{t-1} := \alpha_1 h_1 + \cdots + \alpha_{t-1} h_{t-1}$ without adjusting previously obtained f_t.

Forward stagewise additive modelling

1. **Input:** Sample $S = (x_1, y_1), \ldots, (x_n, y_n)$; the number of iterations T.
2. **Initialization:** $f_0 \equiv 0$;
3. **Do for** $t = 1, \ldots, T$**:**
   - $$(\alpha_t, h_t) := \arg \min_{\alpha_t \in \mathbb{R}, h_t \in \mathcal{H}} \sum_{i=1}^{n} L\big(y_i, f_{t-1}(x_i) + \alpha_t h_t(x_i)\big). \tag{5.4.2}$$
   - $f_t = f_{t-1} + \alpha_t h_t$.
4. **Output:** $\hat{f} = f_T$

In L_2**boosting** $L(y,x) = (y-x)^2$. Then

$$\sum_i L(y_i, f_{t-1}(x_i) + \alpha_t h_t) = \sum_i \big(y_i - f_{t-1}(x_i) - \alpha_t h_t(x_i)\big)^2,$$

so that at every step of the iteration, the function $\alpha_t h_t$ is the best fit to the residuals of f_{t-1}: $r_{t-1}(x_i) = y_i - f_{t-1}(x_i)$. It is easy to see that

$$\arg\min_\alpha \sum_i \big(r_{t-1}(x_i) - \alpha h(x_i)\big)^2 = \frac{\sum_{i=1}^{n} r_{t-1}(x_i) h(x_i)}{\sum_i h^2(x_i)}$$

so that minimizing (5.4.2) is basically over $\mathcal{H}$, only.

Boosting and and lasso. Let $\mathcal{H}$ be finite, $\mathcal{H} = \{h_1, \dots, h_K\}$ so that every $f \in \mathcal{F}$ is the sum

$$f = \sum_{k=1}^{K} \alpha_k h_k.$$

As special case $K = d + 1$ and $h_k(x) = x^k$, $k = 1, \dots, d$ and $h_K \equiv 1$ corresponds to linear regression.

We consider the regression problem

$$\min_{\alpha} \Big[\sum_{i=1}^{n} \big(y_i - \sum_{k=1}^{K} \alpha_k h_k(x_i)\big)^2 + \lambda J(\alpha) \Big],$$

where $\alpha := (\alpha_1, \dots, \alpha_K)$ and $J(\alpha)$ is a penalty like $J(\alpha) = \sum_k \alpha_k^2$ for ridge regression and $J(\alpha) = \sum_k |\alpha_k|$ for lasso. Often K is very big and so regularization is necessary.

We know already that for lasso, finding the optimal vector α is impossible. The following algorithm has shown good performances in approximating the effect of the lasso.

Forward stagewise linear regression

```
1. Input: Sample $S = (x_1, y_1), \dots, (x_n, y_n)$; the number of iterations $T$; a small constant $\epsilon > 0$.
2. Initialize: $\alpha_k^0 = 0$, $k = 1, \dots, K$,   $f_0 \equiv 0$;
3. Do for $t = 1, \dots, T$:
    • $r_{t-1}(x_i) = y_i - f_{t-1}(x_i)$;
    • $(\beta, l) = \arg\min_{\beta,l} \sum_{i=1}^{n} \big(r_{t-1}(x_i) - \beta h_l(x_i)\big)^2$;
    • $\alpha_l^t = \alpha_l^{t-1} + \epsilon \cdot \mathrm{sgn}(\beta)$,   $\alpha_k^t = \alpha_k^{t-1}$, if $t \neq l$;
    • $f_t = \sum_{k=1}^{K} \alpha_k^t h_k$
4. Output: $\hat{f} = f_T$.
```

Hence T replaces the regularization parameter λ – the more iterations, the bigger $\sum_k |\alpha_k|$.

References: About boosting read [7], Ch 10, or the overview papers [18, 23, 22, 24].

Chapter 6

Overview of some other methods

Assumption: class-labels are 0 and 1. We consider two classes, but all of the methods naturally apply for more than two classes.

6.1 Plug-in rules

Recall $\eta(x) = \mathbf{P}(Y = 1|X = x)$ and Bayes rule

$$g^*(x) = \begin{cases} 0, & \text{if } \eta(x) \le 0.5; \\ 1, & \text{if } \eta(x) > 0.5. \end{cases}$$

Let η_n be an estimate of η. **Plug-in classifier** is defined as follows

$$g_n(x) = \begin{cases} 0, & \text{if } \eta_n(x) \le 0.5; \\ 1, & \text{if } \eta_n(x) > 0.5. \end{cases} \tag{6.1.1}$$

For analyzing the performance of plug-in classifiers, the following inequality is useful. For the proof, see [1], Thm 2.2. Let g be any classifier and $F(x)$ the distribution (function) of feature vector X. Then

$$R(g) - R^* = 2\int_{\{g(x)\neq g^*(x)\}} |\eta(x) - \frac{1}{2}|F(dx), \tag{6.1.2}$$

If g_n is plug-in classifier, then

$$g_n(x) \neq g^*(x) \Rightarrow |\eta(x) - \frac{1}{2}| \le |\eta(x) - \eta_n(x)|. \tag{6.1.3}$$

Plugging (7.2.6) into (6.1.2), we get

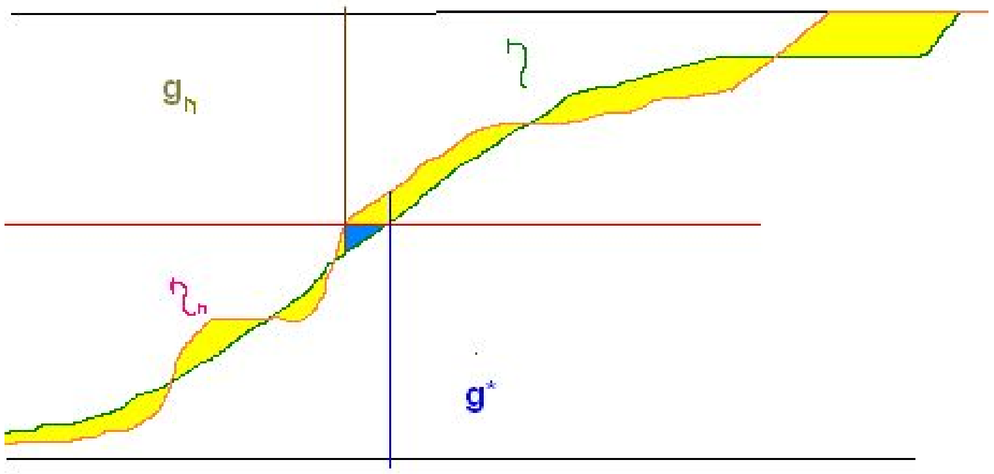
Corollary 6.1.1 *Let g_n be a plug-in classifier based on the estimate η_n. Then*

$$R(g_n) - R^* = 2\int_{\{g_n(x)\neq g^*(x)\}} |\eta(x) - \frac{1}{2}|F(dx) \le 2\int |\eta(x) - \eta_n(x)|F(dx). \tag{6.1.4}$$

Using Ljapunov (or Cauchy-Schwartz) inequality, we get

$$R(g_n) - R^* \leq 2 \int |\eta(x) - \eta_n(x)| F(dx) \leq 2 \sqrt{\int \big(\eta(x) - \eta_n(x)\big)^2 F(dx)}. \tag{6.1.5}$$

Hence, if η_n is consistent in L_2 or in L_1 sense, i.e. the integrals above converge to zero, then the plug-in classifier is consistent. The following picture explains that the above-mentioned convergences are indeed only sufficient, since for consistency of g_n, actually much less is needed.



Sometimes the probabilities $\eta(x)$ and $1 - \eta(x)$ are estimated separately by $\hat{\eta}_1$ and $\hat{\eta}_0$, so that x they need not definitely sum up to one. The plug-in classifier in this case is

$$g_n(x) = \begin{cases} 1, & \text{if } \hat{\eta}_1(x) > \hat{\eta}_0(x); \\ 0, & \text{if } \hat{\eta}_1(x) \leq \hat{\eta}_0(x). \end{cases}$$

Exercise: Let $\hat{\eta}_i(x) \in [0,1]$, $i = 0, 1$. Prove

$$R(g_n) - R^* \leq \int |\eta(x) - \hat{\eta}_1(x)| F(dx) + \int |(1 - \eta(x)) - \hat{\eta}_0(x)| F(dx).$$

Exercise: Let f_1 and f_0 the class-conditional densities, $\pi = \mathbf{P}(Y = 1)$. Suppose $\hat{p}_1$ and $\hat{p}_0$ are the estimates of π and $1 - \pi$, respectively; let $\hat{f}_i$ be the estimate of f_i. Define a plug-in classifier

$$g_n(x) = \begin{cases} 1, & \text{if } \hat{p}_1 \hat{f}_1(x) > \hat{p}_0 \hat{f}_0(x); \\ 0, & \text{if } \hat{p}_1 \hat{f}_1(x) \leq \hat{p}_0 \hat{f}_0(x). \end{cases}$$

Prove that

$$R(g_n) - R^* \leq \int |\pi f_1(x) - \hat{p}_1 \hat{f}_1(x)| dx + \int |(1 - \pi) f_0(x) - \hat{p}_0 \hat{f}_0(x)| dx.$$

6.1.1 Standard plug-in: parametrization

Suppose class-conditional densities f_1 and f_0 are known in form: $f_1(\theta_1, x)$ and $f_0(\theta_0, x)$. The parameters θ_1 and θ_0 as well as the probability $\pi = \mathbf{P}(Y = 1)$ are estimated from data, with corresponding estimates $\hat{\theta}_1$, $\hat{\theta}_0$ and $\hat{\pi}$, the so-called **standard plug-in** is as follows

$$g_n(x) = \begin{cases} 1, & \text{if } \hat{\pi} f_1(\hat{\theta}_1, x) > (1 - \hat{\pi}) f_0(\hat{\theta}_0, x); \\ 0, & \text{if } \hat{\pi} f_1(\hat{\theta}_1, x) \leq (1 - \hat{\pi}) f_0(\hat{\theta}_0, x). \end{cases}$$

If the model is correct, then there exist the true parameters θ_i^* so that $f_i(\theta_i^*, x)$, $i = 0, 1$ are the true class-conditional densities. The estimates $\hat{\theta}_i$ are *(strongly) consistent* if $\hat{\theta}_i \to \theta_i^*$ for every $i = 0, 1$ in probability (a.s.). If $\hat{\pi}$ is the proportion of ones in the sample, then by SLLN $\hat{\pi} \to \pi$, a.s. Is standard plug-in consistent rule, given the parameter estimates are consistent? General answer is No.

Example. Let $f(\theta, x)$ be a uniform on $[-\theta, 0]$, if $\theta \neq 1$ and uniform over $[0, 1]$, if $\theta = 1$, $\pi = 0.5$. Suppose the true parameters are $\theta_0^* = 2$ and $\theta_1^* = 1$. A reasonable estimate would be

$$\hat{\theta}_i := \max_{j: Y_j = i} |X_i|.$$

Clearly $\hat{\theta}_i \to \theta_i^*$ a.s.. On the other hand, $\hat{\theta}_i \neq 1$ a.s.. Therefore if $x > 0$, we have $f(\hat{\theta}_1, x) = f(\hat{\theta}_2, x) = 0$ and according to the standard plug-in rule, $g_n(x) = 0$. Hence $R(g_n) \geq \mathbf{P}(Y = 1) = 0.5$, but $R^* = 0$.

The counter-example above is possible because of the lack of continuity. We shall now introduce some necessary continuity assumptions. Let

$$\eta_\theta(x) := \frac{\pi f_1(\theta_1, x)}{\pi f_1(\theta_1, x) + (1 - \pi) f_0(\theta_0, x)}.$$

The function η_θ is continuous in space $L_1(\mathbb{R}^d, F)$, if $\theta_n \to \theta$ implies

$$\int |\eta_{\theta_n}(x) - \eta_\theta(x)| F(dx) \to 0.$$

Theorem 6.1.1 *Let $\theta \mapsto \eta_\theta$ be continuous in $L_1(\mathbb{R}^d, F)$. Assume that the estimates $\hat{\theta}_i$ are (strongly) consistent. Then standard plug-in rule $\{g_n\}$ is (strongly) consistent.*

Exercise: Prove the theorem.

Exercise: Show that in the counter-example above, the function η_θ is not continuous in $L_1(\mathbb{R}^d, F)$.

A necessary condition for continuity of η_θ in $L_1(\mathbb{R}^d, F)$ is the following: for any x, and $i = 0, 1$, the mapping $\theta_i \mapsto f_i(\theta_i, x)$ is continuous. Indeed, then, for any x also $\theta \mapsto \eta_\theta(x)$ is continuous and dominated convergence theorem implies the continuity in $L_1(\mathbb{R}^d, F)$.

6.2 Partitioning rules

Let $\mathcal{S} := \{S_1, S_2, \ldots\}$ be a **partition** of $\mathbb{R}^d$ (disjoint parts that cover all space). For every $x \in \mathbb{R}^d$, let $S(x)$ be the part, also called as a *cell* containing vector x. Let g_n be a classifier that on each cell classifies according to the majority vote amongst the labels from the same cell. Formally

$$g_n(x) = \begin{cases} 0, & \text{if } \sum_{i=1}^n I_{\{y_i=1\}} I_{\{x_i \in S(x)\}} \leq \sum_{i=1}^n I_{\{y_i=0\}} I_{\{x_i \in S(x)\}}; \\ 1, & \text{else.} \end{cases} \tag{6.2.1}$$

In other words, g_n minimizes empirical risk over all possible classifiers that are constants on the cells of the partition $\{S_1, S_2, \ldots\}$.

Definition 6.2.1 **Partitioning rule** *consists of classifiers $\{g_n\}$, where for every n the classifier g_n is defined as in (6.2.1) and the corresponding partition $\mathcal{S}_n$ can depend on n and on the sample $x_1, \ldots, x_n$.*

In principle, any classifiers (finite-valued function) is defined via a partition, but by partitioning rules:

1. the partition is independent of the labels $y_1, \ldots, y_n$
2. the value of g_n in each cell is defined via majority votes.

Despite to these restrictions, the class of partitioning rules is large containing histograms, nearest neighbor rules, several trees etc.

The consistency of partitioning rules. We shall present a general theorem that gives sufficient conditions for consistency of partitioning rule. Let $\{g_n\}$ be a partitioning rule as defined above. In order g_n approximate Bayes classifier g^* more and more precisely as n grows, it is obvious that the cells S_k^n have to decrease as n grows. On the other hand, for every cell the class label is determined via majority vote. Since the preciseness of majority vote rule has to increase, it follows that the number of training pairs (x_i, y_i) in each cell has to increase. The following theorem states that these two conditions are sufficient for consistency.
For any set $S \subset \mathbb{R}^d$, let $\text{diam}S$ be its diameter: $\text{diam}S = \sup_{a,b \in S} \|a - b\|$.
Given a partition and a random sample $X_1, \ldots, X_n$, let $N(x)$ be the number of X_i's in $S(x)$, i.e.

$$N(x) = \sum_{i=1}^n I_{\{X_i \in S(x)\}}.$$

Theorem 6.2.2 *Let $\{g_n\}$ be a partitioning rule as defined above. Let X be a random vector with distribution $F(x)$ and independent of the sample. Then $ER(g_n) \to R^*$, if the following two conditions simultaneously hold:*

1. $\text{diam}S(X) \to 0$ *in probability*
2. $N(X) \to \infty$ *in probability.*

For the proof, see [1], Thm 6.1.

General risks and regression with partitioning rule. The partitioning rules are not limited with two classes or symmetric loss. Indeed, the majority vote rule in a cell is nothing but the best constant in the ERM-sense in that cell. Hence, for any loss function and any number of classes, we can define the partitioning rule by taking $g_n(x)$ as the constant that minimizes the empirical risk over the cell $S(x)$. Formally (recall $\mathcal{Y}$ is the set of classes), thus

$$g_n(x) = \arg\min_{j \in \mathcal{Y}} \sum_{i=1}^{n} L(y_i, j) I_{\{x_i \in S(x)\}}.$$

Now it is clear that the partitioning rules can be used by regression: the obtained regression function has a constant value on the cells, the value is obtained by ERM-principle over the cell. Formally thus

$$g_n(x) = \arg\min_{r \in \mathbb{R}} \sum_{i=1}^{n} L(y_i, r) I_{\{x_i \in S(x)\}}.$$

If $L(y,r) = (r-y)^2$, then $g_n(x)$ is the sample average over the cell:

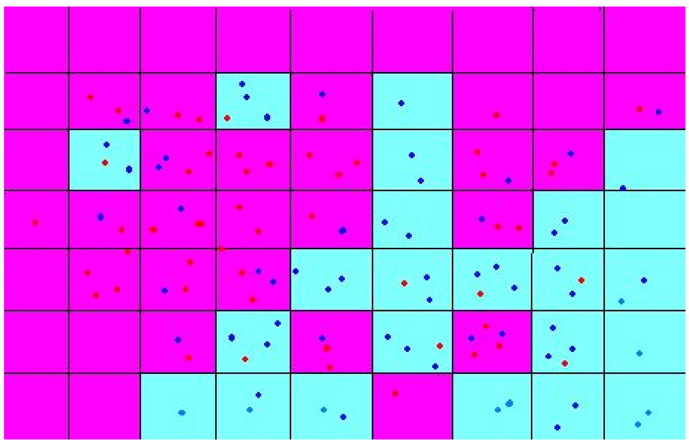
$$g_n(x) = \frac{1}{n(x)} \sum_{i=1}^{n} y_i I_{\{x_i \in S(x)\}}, \quad \text{where} \quad n(x) := \sum_i I_{\{x_i \in S(x)\}}.$$

6.2.1 Histogram rule

Histogram rules are partition rules where the partition is obtained by partitioning consists of cubes of equal size with edge length h_n. Hence a cell of the partition is

$$\prod_{i=1}^{d} [k_i h_n, k_{i+1} h_n),$$

where $k_i \in \mathbb{Z}$ and the partition is independent of the training data.



Consistency. A cube with edge length h_n has diameter $\sqrt{d}h_n^d$. Hence, the first assumption of Theorem 6.2.2 holds if and only if $h_n \to 0$. In order the second assumption to hold, the convergence $h_n \to 0$ cannot be too fast. It turns out that it suffices if h_n converges to infinity so slow that $nh_n^d \to \infty$. Then both assumptions of Theorem 6.2.2 are fulfilled and the following theorem holds. For proof, see [1], Thm 6.2.

Theorem 6.2.3 *Let $h_n \to 0$ and $nh_n^d \to \infty$. Then histogram rule is universally consistent.*

It turns out that the assumptions of the previous theorem are sufficient also for strong consistency.

Theorem 6.2.4 *Let $h_n \to 0$ and $nh_n^d \to \infty$. Then for every distribution $F(x,y)$ and $\epsilon > 0$ there exists n_0 such that the risk of histogram rule $R(g_n)$ satisfies the inequality*

$$\mathbf{P}\big(R(g_n) - R^* > \epsilon\big) \leq 2e^{\frac{-n\epsilon^2}{32}}.$$

Hence histogram rule is universally strongly consistent.

6.2.2 Nearest neighbor rules

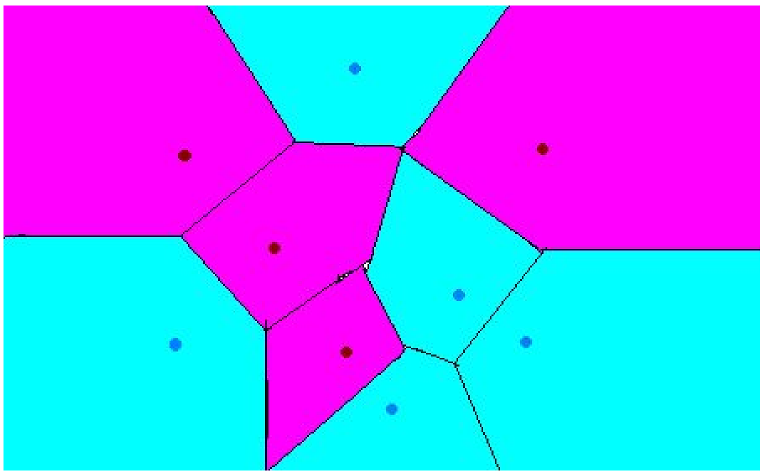
Let $x \in \mathbb{R}^d$ be a feature vector to classify, and let

$$x_{(1)}, \dots, x_{(k)} \tag{6.2.2}$$

be the k nearest elements from the training sample to the point x (in Euclidian distance). In case of a distance tie, the candidate with the smaller index is said to be closer to x. The **k-nearest neighbor rule** for classifying x uses the majority votes amongst the labels of (6.2.2). Formally, the rule is

$$g_n(x) = \begin{cases} 0, & \text{if } \sum_{i=1}^k I_{\{y_{(i)}=1\}} \leq \sum_{i=1}^k I_{\{y_{(i)}=0\}}; \\ 1, & \text{else.} \end{cases},$$

where $y_{(i)}$ is the label of $x_{(i)}$. It is convenient to let k to be odd to avoid ties. Sometimes the 1-nearest neighbor rule will be referred to as the nearest neighbor rule. Clearly nearest neighbor rules are partitioning rules – a cell consists of these x that share the same set of k nearest neighbors. For 1-nearest neighbor method, the partition consists of n cells $\{S_1, \dots, S_n\}$, the cell S_j consists of the feature vectors x having x_j as the closest sample point. Such a partition is often called as **Voronoi partition (tesselation)**. Note that the empirical risk (training error) of 1-nearest neighbor rule is always zero. In general, training error tends to increase as k grows.



Limit behavior with fixed k

Let us briefly state some results about the behavior of the k-nearest neighbor rule, when k is fixed, but $n \to \infty$. At first, we see that for *any* distribution $F(x)$ of X, the k-th nearest neighbor converges to x, a.s. Recall that a **support** of a distribution of F is the smallest closed set with measure 1. It means that if x belongs to the support of the distribution of X, then for any ball $B(x, r)$ with radius $r > 0$, it holds $\mathbf{P}(X \in B(x, r)) > 0$.

Proposition 6.2.1 *Let $X_1, \ldots, X_n$ be iid random vectors with distribution F. Let x belong to the support of F and let $X_{(k)}(x)$ be the k-th neighbor of x. Then, as n grows*

$$\|X_{(k)}(x) - x\| \to 0 \quad a.s. \tag{6.2.3}$$

Exercise: Prove the proposition by proceeding as follows. Let B be a ball with center x and radius $\delta > 0$. Since x belongs to support of x, it holds that $\mathbf{P}(X \in B) =: \mu > 0$. Show that

$$\{\|X_{(k)} - x\| > \delta\} = \Big\{\frac{1}{n}\sum_{i=1}^{n} I_B(X_i) < \frac{k}{n}\Big\}.$$

Now, using the fact that for $0 < \epsilon < \mu$, there exists n_o so that for any $n > n_o$

$$\Big\{\frac{1}{n}\sum_{i=1}^{n} I_B(X_i) - \mu < \frac{k}{n} - \mu\Big\} \subset \Big\{\frac{1}{n}\sum_{i=1}^{n} I_B(X_i) - \mu < -\epsilon\Big\}$$

prove the convergence in probability

$$\mathbf{P}(\|X_{(k)}(x) - x\| > \epsilon) \to 0.$$

Deduce (6.2.3) using the criterion

$$X_n \to X \quad \text{a.s.} \quad \Leftrightarrow \quad \lim_n \mathbf{P}(\sup_{m \geq n} |X_m - X| > \epsilon) = 0, \quad \forall \epsilon > 0. \tag{6.2.4}$$

The following exercise generalizes Proposition 6.2.1 by letting k depend on n.

Exercise: Let x belong to the support of F and let $\frac{k(n)}{n} \to 0$. Prove that as n grows

$$\mathbf{P}(\sup_{m \geq n} \|X_{(k(m))}(x) - x\| > \epsilon) \to 0.$$

Deduce that

$$X_{(k(n))}(x) \to x \quad \text{a.s.} \tag{6.2.5}$$

Hint: Show

$$\big\{ \sup_{m \geq n} \|X_{(k(m))}(x) - x\| > \epsilon \big\} = \bigcup_{m \geq n} \Big\{ \frac{1}{m} \sum_{i=1}^{n} I_B(X_i) - \mu < \frac{k(m)}{m} - \mu \Big\}.$$

Use SLLN and(6.2.4).

Let X be a random vector with distribution F and independent of the sample. From (6.2.5), it follows that

$$X_{(k(n))}(X) \to X \quad \text{a.s.}$$

Odd k. From (6.2.3) it follows that when $x \mapsto \eta(x)$ is continuous, then for n big enough $\eta(x_{(k)}) \approx \eta(x)$ so that $Y_{(k)}$ (the class of k-th neighbor) has approximatively the Bernoulli distribution with parameter $\eta(x)$. Therefore, the sum of the labels of k nearest neighbors, $\sum_{l=1}^{k} Y_{(l)}$, has approximatively $B(\eta(x), k)$ distribution. Hence, the probability that the majority vote equals to 0, is for odd k approximatively $\mathbf{P}(B < \frac{k}{2})$, when $B \sim B(\eta(x), k)$. Thus, for odd k and big n, the conditional risk $R(g(x)|x)$ of k-nearest neighbor classifier is approximatively

$$\mathbf{P}\Big(B > \frac{k}{2}, Y = 0\Big) + \mathbf{P}\Big(B < \frac{k}{2}, Y = 1\Big), \tag{6.2.6}$$

where Y is the label of x, hence Bernoulli random variable with parameter $\eta(x)$ and independent of B. Since

$$\mathbf{P}\Big(B > \frac{k}{2}\Big) = \sum_{j=\frac{k}{2}+1}^{k} \binom{k}{j} \eta(x)^j (1-\eta(x))^{k-j}, \quad \mathbf{P}\Big(B < \frac{k}{2}\Big) = \sum_{j=0}^{\frac{k}{2}-1} \binom{k}{j} \eta(x)^j (1-\eta(x))^{k-j},$$

the probability (6.2.6) is

$$\sum_{j=0}^{k} \binom{k}{j} \eta(x)^j (1-\eta(x))^{k-j} \big(\eta(x) I_{\{j<\frac{k}{2}\}} + (1-\eta(x)) I_{\{j>\frac{k}{2}\}}\big)$$

Averaging over the features, we obtain that when η is continuous, then for odd k and big n, the conditional risk $R(g)$ of k-nearest neighbor classifier is approximatively

$$\sum_{j=0}^{k} \binom{k}{j} E\Big[\eta(X)^j (1-\eta(X))^{k-j} \big(\eta(X) I_{\{j<\frac{k}{2}\}} + (1-\eta(X)) I_{\{j>\frac{k}{2}\}}\big)\Big] =: R_k$$

It turns out that the intuition above is correct even when the function η is not continuous. Namely, the following theorem ([1], Thm 5.2) holds.

Theorem 6.2.5 *Let k be odd and fixed. Then, for the k-nearest neighbor rule $\{g_n\}$,*

$$ER(g_n) \to R_k. \tag{6.2.7}$$

It can be shown

$$R^* \leq \cdots \leq R_{2k+1} \leq R_{2k-1} \leq \cdots \leq R_5 \leq R_3 \leq R_1$$

and the strict inequalities hold whenever

$$\mathbf{P}(\eta(X) \in \{0, 1, 0.5\}) > 0. \tag{6.2.8}$$

Hence, asymptotically the k-nearest neighbor rule works best for bigger k. However, as the following example shows, it can be so that the nearest neighbor rule works better than k-nearest neighbor rule for any n.

Example. Let S_0 and S_1 be two spheres of radius 1 centered at a and b with $\|a-b\| > 4$. Let $\mathbf{P}(Y = 1) = \mathbf{P}(Y = 0) = \frac{1}{2}$ and given $Y = i$ the feature X is uniform over S_i. Let g_n be 1-nearest neighbor rule. Then

$$ER(g_n) = \mathbf{P}(Y = 0, Y_1 = \cdots = Y_n = 1) + \mathbf{P}(Y = 1, Y_1 = \cdots = Y_n = 0) = 2^{-n}.$$

When g_n is k nearest neighbor rule for $k \geq 3$ (odd), then

$$ER(g_n) = 2^{-n} \sum_{j=1}^{\lfloor \frac{k}{2} \rfloor} \binom{n}{j} > 2^{-n}.$$

Exercise: Prove the equality. Does (6.2.8) hold?

Cover-Hart inequalities. For $k = 1$, the limit risk R_1 is as follows

$$R_1 = 2E\big[\eta(X)(1 - \eta(X))\big].$$

Denote $A(X) := \eta(X) \wedge (1 - \eta(X))$ and recall that $R^* = EA(X)$. Jensen's inequality shows

$$E[\eta(X)(1-\eta(X))] = E[A(X)(1-A(X))] \leq EA(X) - (EA(X))^2 = R^*(1-R^*) \leq R^*.$$

The inequalities

$$R_1 \leq 2R^*(1-R^*) \leq 2R^*$$

are known as **Cover-Hart inequalities**. Hence, small Bayes risk R^* implies small R_1. In particular, when $R^* = 0$, then $R_1 = R^* = 0$ and from Theorem 6.2.5, it follows then that nearest neighbor rule (as well as k-nearest neighbor rule) is consistent. On the other hand, when $\eta(x) \neq \frac{1}{2}$, then $2\eta(x)(1-\eta(x)) > \eta(x) \wedge (1-\eta(x))$. Since $0 < R^* < \frac{1}{2}$ implies that (6.2.8) holds, it follows that

$$\mathbf{P}\big(2\eta(X)(1-\eta(X)) > \eta(X) \wedge (1-\eta(X))\big) > 0$$

and $R_1 > R^*$. Then the nearest neighbor rule is not consistent.

Weighted nearest neighbor rule. In k-nearest neighbor rule, each of the nearest neighbors plays an equally important role. However, just like in the kernel methods (recall Gaussian kernel) intuition tells that nearest neighbors should provide more information. This idea brings to **weighted nearest neighbor rule**, where for every n, the i-th nearest neighbor receives weight w_{ni}, typically decreasing in i. The rule is

$$g_n(x) = \begin{cases} 0, & \text{if } \sum_{i=1}^n w_{ni} I_{\{y_{(i)}=1\}} \leq \sum_{i=1}^n w_{ni} I_{\{y_{(i)}=0\}}; \\ 1, & \text{else.} \end{cases}$$

The k-nearest neighbor rule corresponds to the case when $w_{ni} = \frac{1}{k}$ for $i = 1, \ldots, k$ and $w_{in} = 0$, elsewhere. Suppose now that the weights $w_i = w_{ni}$ are independent of n and $w_i = 0$ if $i > k$. Then there exists a constant $R(w_1, \ldots, w_k)$ so that

$$ER(g_n) \to R(w_1, \ldots, w_k).$$

Moreover, when k is odd then $R(w_1, \ldots, w_k) \geq R_k$ and in the most cases of interest the inequality is strict. Hence, the standard k-neighbor rules are to be preferred in an asymptotic sense.

Even k. Odd k avoids the voting ties. For even k (in the case of ties), a common tie breaking rule is to use the label of the nearest neighbor. Formally the rule is

$$g_n(x) = \begin{cases} 1 & \text{if } \sum_{i=1}^k y_{(i)} > \frac{k}{2}, \\ 0 & \text{if } \sum_{i=1}^k y_{(i)} < \frac{k}{2}, \\ y_{(1)} & \text{if } \sum_{i=1}^k y_{(i)} = \frac{k}{2}. \end{cases}$$

This is equivalent to a weighted k-nearest neighbor rule with weight vector $(3, 2, \ldots, 2)$. Hence, also for even k, the limit risk R_k exists. However, the limit risk for even k is not better than the one of R_{k-1}, since it can be shown that for any distribution and even k, the following equality holds: $R_k = R_{k-1}$ ([1] Thm. 5.5). This equality justifies the practice of using odd k.

Consistency of nearest neighbor rules

Although k-nearest neighbor rule works asymptotically better for big k, it is not consistent even for a very big but fixed k. It turns out that for consistency the number of nearest neighbors k has to grow with n. But, as usually, the growth cannot be too fast.

Theorem 6.2.6 (Stone, 1977) *If $k \to \infty$ and $\frac{k}{n} \to 0$, then the nearest neighbor rule $\{g_n\}$ is universally consistent, i.e. for every distribution $F(x, y)$, $ER(g_n) \to R^*$.*

For proof, see [1], thm 6.4. Theorem 6.2.6 was the first universal consistency result.

It turns out that when the problem of distance ties are suitably taken care of, then the assumptions of Theorem 6.2.6 are sufficient for strong consistency of nearest neighbor rule. The distance ties do not occur (with probability one), if the distribution of X is absolutely continuous. Then the following large deviation inequality holds.

Theorem 6.2.7 *Assume that the distribution of X is absolutely continuous. If $k \to \infty$ and $\frac{k}{n} \to 0$, then there exists n_o so that for every $n > n_o$ the nearest neighbor rule $\{g_n\}$ satisfies*

$$\mathbf{P}(R(g_n) - R^* > \epsilon) \leq 2e^{\frac{-n\epsilon^2}{c}}, \tag{6.2.9}$$

where c is a constant depending on d. Thus, the nearest neighbor rule is strongly consistent, i.e. $R(g_n) \to R^*$ a.s..

In the case of distance ties, the strong consistency can be achieved by suitable tie-breaking method. It can be shown that the tie-breaking by indices (the sample element with smaller index is declared to be closer) might destroy the inequality (6.2.9) so that the universal consistency is very unlikely. A way to circumvent the aforementioned difficulty is artificial increase the dimension of feature vector by one. The new feature vectors are $(x_1, u_1), \ldots, (x_n, u_n)$, where $u_1, \ldots, u_n$ are the realizations if i.i.d. random variables with absolutely continuous distribution independent of everything. Since the additional component is independent of the pair (X, Y), the Bayes risk remains unchanged, and since now ties do not occur any more (a.s.), it can be shown that Theorem 6.2.7 holds and the rule is universally consistent.

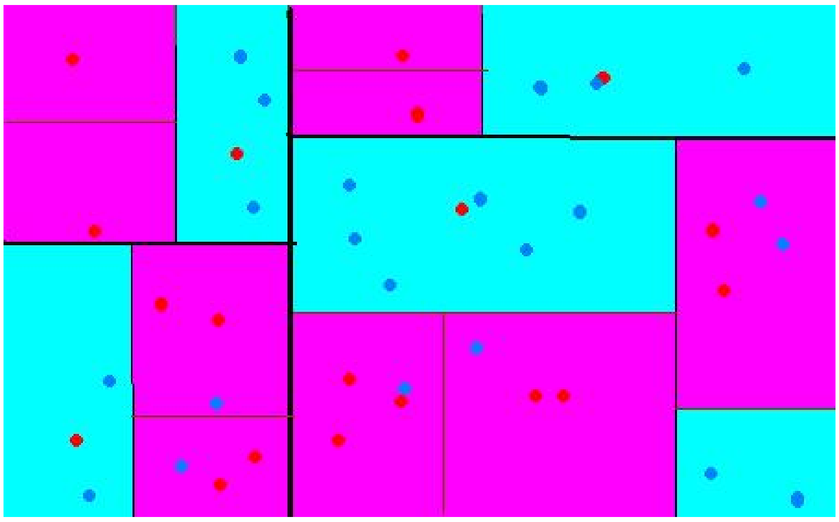
References: About nearest neighbors, read [1], Ch 5,6,11; [7], Ch 3, 13.

6.3 CART-trees

6.3.1 Trees

The tree structure is very popular decision procedure. To every terminal node (leave) of a decision tree corresponds a cell and a label. The cells form a partition of $\mathbb{R}^d$. Typically the label of a cell is determined via the training sample by ERM-principle like majority vote amongst the elements in the cell. If the tree is constructed using the features $x_1, \ldots, x_n$ but not the labels, then such a tree is a partitioning rule.

We shall consider only *binary trees*, where each node has exactly two or zero children. Hence, every node represents a region S and in the case the node is not the terminal one, the children of the node represent regions S' and S'' so that $S = S' \cup S''$ and $S' \cap S'' = \emptyset$. The regions corresponding to the terminal nodes form the partition. By trees, it is important that the *splitting rule* for deciding whether a feature vector from S belongs to S' or S'' were possibly simple, i.e. moving along a tree would involve relatively "easy" questions. Most common splitting rule uses single coordinates of the feature vector $x = (x^1, \ldots, x^d)$ and moving along the decision tree is based on the questions (conditions) like: Is $x^i \leq t$? A decision tree based on such splitting rules are called **ordinary binary (classification) tree**. The partition corresponding to such tree consists of cells having the boundaries parallel to coordinate axes.



6.3.2 CART-trees

Perhaps the most popular class of decision trees are so-called CART (classification and regression trees). They can be used by regression as well as classification, to capture the idea, it is better to start with regression. CART trees are typically ordinary binary trees, so that the splitting rule is based on the coordinates.

Regression. As usually by regression, we consider the quadratic loss. The construction of the tree is based on greedy algorithm: at first step the splitting variable j and splitting point t is chosen so that the after the splitting the decrease of sum of squares were maximal. Formally, let for every $j = 1, \ldots, d$ and $t \in \mathbb{R}$ the sets

$$S_1(j,t) := \{x \in \mathbb{R}^d : x^j \leq t\}, \quad S_2(j,t) := \{x \in \mathbb{R}^d : x^j > t\}$$

be the candidates to first split. We look for j an t that are the solutions of the following problem:

$$\min_{t,j} \Big[\min_{c_1} \sum_{x_i \in S_1(t,j)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in S_2(t,j)} (y_i - c_2)^2 \Big]. \tag{6.3.1}$$

Clearly (6.3.1) is equivalent to

$$\min_{t,j} \Big[n_1 \cdot \frac{1}{n_1} \sum_{x_i \in S_1(t,j)} (y_i - \hat{c}_1)^2 + n_2 \cdot \frac{1}{n_2} \sum_{x_i \in S_2(t,j)} (y_i - \hat{c}_2)^2 \Big], \tag{6.3.2}$$

where n_l, $l = 1,2$ is the number of sample elements in S_l and $\hat{c}_l$ is the conditional mean over S_l, i.e.

$$\hat{c}_l := \frac{1}{n_l} \sum_{x_i \in S_l} y_i.$$

"For each splitting variable, the determination of the split point can be done very quickly ..." ([7], p. 307.)

Having found the best split, we partition the data into the two resulting regions and repeat the splitting process on each of the two regions. Then this process is repeated on all of the resulting regions. How large should the tree grow? Clearly a very large tree might overfit the data, while a small tree might not capture the important structure. One approach would be to split tree nodes only if the decrease in sum-of-squares due to the split exceeds some threshold. This strategy is too short-sighted, however, since a seemingly worthless split might lead to a very good split below it. The preferred strategy is to grow a large tree T_0, stopping the splitting process only when some minimum node size (say 5) is reached. Then this large tree is pruned using **cost-complexity pruning**, which we now describe.

We define a subtree $T \subseteq T_0$ to be any tree that can be obtained by "pruning" T_0, that is, collapsing any number of its internal (non-terminal) nodes. Let, for any subtree, $|T|$ be the number of its leaves. Given a regularization constant λ, for any subtree T, we define the (the cost-complexity) function

$$R_\lambda(T) = \sum_{i=1}^{|T|} \sum_{x_j \in S_i} (y_j - \hat{c}_i)^2 + \lambda |T|, \tag{6.3.3}$$

where $S_1, \dots, S_{|T|}$ is the partition corresponding to the subtree, and $\hat{c}_i$ is the conditional average on S_i. The idea is to find, for each λ, the subtree T_λ that minimizes $R_\lambda(T)$. Large values of λ result in smaller trees. With $\lambda = 0$ the solution is the full tree T_0.

Weakest link pruning. To find T_λ, the so-called **weakest link pruning** is used. In this method, the internal nodes that produce the smallest per-node increase in

$$\sum_{i=1}^{|T|} \sum_{x_j \in S_i} (y_j - \hat{c}_i)^2$$

are successively collapsed. The procedure is continued until the single root-tree is left.

Thus, at the first step the subtree T is searched so that the following ratio were minimal

$$\frac{\sum_{i=1}^{|T|} \sum_{x_j \in S_i} (y_j - \hat{c}_i)^2 - \sum_{i=1}^{|T_0|} \sum_{x_j \in S_i^0} (y_j - \hat{c}_i)^2}{|T_0| - |T|}, \tag{6.3.4}$$

where $S_1, \dots, S_{|T|}$ and $S_1^0, \dots, S_{|T_0|}^0$ are the partitions of trees T and T_0, respectively. Recall that T is obtained from T_0 by collapsing the tree starting from an internal node, say t. Let S^t be the union of all cells of T_0 corresponding to the offsprings of t, i.e.

$$S^t = \cup_{k=1}^{d_t} S_{i_k}^0,$$

where d_t be the number of offsprings of the node t. The partitions $S_1, \ldots, S_{|T|}$ and $S_1^0, \ldots, S_{|T_0|}^0$ coincide, except the cells $S_{i_k}^0$ are replaced by their union S^t. Let $\hat{c}_t$ be the conditional mean over S_t. Hence (6.3.4) is actually a function of node:

$$g_1(t) := \frac{\sum_{x_j \in S^t} (y_j - \hat{c}_t)^2 - \sum_{k=1}^{d_t} \sum_{x_j \in S_{i_k}^0} (y_j - \hat{c}_{i_k})^2}{d_t - 1}$$

Therefore, in the first step of the weakest link pruning, the terminal node t_1 with smallest $g_1(t)$ is searched. Then T_0 is pruned so that t_1 is a leave and S^{t_1} is the corresponding cell. Let T^1 be the pruned tree. Now, for T^1, the criterion (6.3.4) is minimized. This means (re)calculating the coefficients $g_2(t)$ for each terminal node of T^1, and finding the minimizer t_2. Proceeding that, we end up with a nested sequence of subtrees

$$\{\text{root}\} = T^m \subset T^{m-1} \subset \cdots T^1 \subset T_0.$$

The following theorem by L. Breiman shows that the sequence above contains T_λ for any λ. Let $\lambda_i = g_i(t_i)$, where $i = 1, \ldots, m$, $\lambda_0 := 0$ and $\lambda_{m+1} := \infty$.

Theorem 6.3.1 *It holds $\lambda_0 < \lambda_1 < \lambda_2 < \cdots < \lambda_m$. When $\lambda \in [\lambda_k, \lambda_{k+1})$, then $T_\lambda = T_{\lambda_k} = T^k$, $\forall k = 0, 1, \ldots, m$.*

Often the training sample is divided into two parts: a test set and validation set. The test set is used building T_0 and validation set is used to prune.

Classification. Classification is based on the same principle as regression: a big tree T_0 is build and then pruned. Of course, by classification, instead of quadratic loss, another function is used in (6.3.1). A natural choice were empirical risk, but in CART-methodology also some alternatives are used.
Given a tree T and corresponding partition $S_1, \ldots, S_{|T|}$, let

$$\hat{p}(S_i) =: \hat{p}_i := \frac{1}{n_i} \sum_{x_j \in S_i} y_j.$$

Building and pruning the tree is based on impurity measure $\phi(\hat{p}_i)$, where $\phi : [0,1] \to \mathbb{R}^+$ is so-called **impurity function** that is symmetric around 0.5, has its maximum at 0.5 and equals to zero at 0 and 1. The most commonly used impurity functions are

- $\phi(p) = \min(p, 1-p)$. Then $\phi(\hat{p}_i)$ is empirical risk on S_i;
- $\phi(p) = 2p(1-p)$: *Gini index*;
- $\phi(p) = -p \log p - (1-p)\log(1-p)$: *binary entropy function.*

With an impurity function (6.3.2) is

$$\min_{t,j} \Big[n_1 \phi\big(\hat{p}(S_1(t,j))\big) + n_2 \phi\big(\hat{p}(S_2(t,j))\big) \Big]. \tag{6.3.5}$$

Example: Assume that $n = 800$ and we have 400 observations in both class (denote this by $(400, 400)$). Consider two splits.

- The first split creates nodes $(300, 100)$ and $(100, 300)$. Then $\hat{p}_1 = 0.75, n_1 = 400$ and $\hat{p}_2 = 0.25, n_2 = 400$. Let us calculate
$$N_1\phi\big(\hat{p}(S_1)\big) + N_2\phi\big(\hat{p}(S_2)\big) \tag{6.3.6}$$
for ϕ being empirical risk and Gini index. For empirical risk (6.3.6) equals to 200, for Gini index, it equals to
$$n_1 2\frac{1}{4}\frac{3}{4} + n_2 2\frac{1}{4}\frac{3}{4} = 400\frac{6}{8}.$$

- The second split creates nodes $(200, 400)$ and $(200, 0)$. Then $\hat{p}_1 = \frac{2}{6}, n_1 = 600$ and $\hat{p}_2 = 1, n_2 = 200$. For empirical risk (6.3.6) equals to 200, again, but for Gini index, it is smaller:
$$n_1\frac{1}{3}\frac{2}{3} = 600\frac{2}{9} = \frac{400}{3}.$$

Hence, using Gini index, the second split is preferred, since it produces a pure node. Based on such arguments, the authors of [7] suggest to use Gini index or entropy function as the impurity function.

Also the pruning is similar to regression. The function (6.3.3) is now
$$R_\lambda(T) = \sum_{i=1}^{|T|} n_i\phi\big(\hat{p}(S_i)\big) + \lambda|T|.$$

The ratio (6.3.4) is
$$\frac{\sum_{i=1}^{|T|} n_i\phi(\hat{p}(S_i)) - \sum_{i=1}^{|T_0|} n_i^0\phi(\hat{p}_i(S_i^0))}{|T_0| - |T|}$$

Finally, the function $g_1(t)$ is
$$g_1(t) := \frac{n^t\phi(\hat{p}(S^t)) - \sum_{k=1}^{d_t} n_{i_k}^0\phi(\hat{p}(S_{i_k}^0))}{d_t - 1}, \tag{6.3.7}$$
where n^t and $n_{i_k}^0$ are the number of elements in S^t and $S_{i_k}^0$, so that $n^t = \sum_k n_{i_k}^0$.

Exercise: Let ϕ be one of the three impurity functions above; let g_1 be defined as in (6.3.7). Prove that for every terminal node $g_1(t) \geq 0$.

After the node t_1 minimizing $g_1(t)$ is found, the subtree started from t_1 is pruned and the process is repeated with pruned tree T^1 instead of the T_0 just like by regression.

The authors of [7] suggest to use empirical risk by pruning. Hence, they suggest to use different impurity functions by growing the tree and pruning.

The obtained rule is not a partitioning rule. Also, it is possible to show that when the splitting rule is based on single coordinates, then the CART-trees are not consistent. For a counterexample, see ([1], 20.8).

More than two classes. The CART-methodology naturally generalizes for more than two classes. Indeed, for each cell and class $m = 0, \dots, k-1$ define the proportion of m-labels in cell i:

$$\hat{p}_{i,m} := \frac{1}{n_i} \sum_{x_j \in S_i} I_{\{y_j = m\}}.$$

The impurity functions generalize for k classes as follows.

- empirical risk: $\phi(p_0, \dots, p_{k-1}) = 1 - \max_{i=0,\dots,k-1} p_i$;
- Gini index: $\phi(p_0, \dots, p_{k-1}) = \sum_{i=0}^{k-1} p_i(1-p_i)$;
- entropy: $\phi(p_0, \dots, p_{k-1}) = -\sum_{i=0}^{k-1} p_i \ln p_i$.

References: About CART-trees, read [7], Ch 9; [8], Ch 7. About other tree-based methods, read [1], Ch 20.

Chapter 7

Unsupervised learning: Hidden Markov models

So far, we have been considering the supervised learning – also called *learning with teacher* – where the training data are given with the labels, i.e. to every feature vector x_i, the corresponding label (or output on the regression) y_i is known. Now, we consider the case of **unsupervised learning**, where the training sample $x_1, \ldots, x_n$ consists of feature vectors without the labels. To do some meaningful predictions without any further information about the data is difficult, although several general methods like cluster analysis exists. Often the lack of labels is compensated by assuming or knowing the probabilistic (generic) model of the data. Often the model is known up to the parametrization, and the parameters are estimated from the data. Sometimes a part (typically a small fraction) of the training data is known with labels and the rest of the data consist of feature vectors, solely. This case is referred to as **semi-supervised learning**. In semi-supervised learning, typically, the supervised part of the data (with labels) is used for fitting the model, the rest of the data (without labels) are then analyzed with using the fitted model.

7.1 Definition of Hidden Markov model

We shall consider a simple yet general model for classification in the case of unsupervised learning. Let us start with the basic definitions.

Underlying Markov chain – the regime. As previously, let $\mathcal{Y} = \{0, \ldots, k-1\}$ be the set of classes; in this chapter the classes will be referred to as the **states**, and the set $\mathcal{Y}$ will be now called as the **state space.** Let now $Y = \{Y_t\}_{t \in \mathbb{N}}$ be a **homogeneous Markov chain** that takes values in $\mathcal{Y}$. As usually, we shall denote by $\pi_j = P(Y_1 = j)$, $j \in \mathcal{Y}$ the probability that the initial state is j; the vector π is thus the distribution of Y_1. We shall denote via

$$\mathbb{P} = \begin{pmatrix} p_{00} & p_{01} & \cdots & p_{0(k-1)} \\ p_{10} & p_{11} & \cdots & p_{1(k-1)} \\ \cdots & \cdots & \cdots & \cdots \\ p_{(k-1)0} & p_{(k-1)1} & \cdots & p_{(k-1)(k-1)} \end{pmatrix}$$

the transition matrix of Y, hence, for any $i_1, \ldots, i_{t-2} \in \mathcal{Y}$

$$p_{ij} = \mathbf{P}(Y_t = j | Y_{t-1} = i) = \mathbf{P}(Y_t = j | Y_{t-1} = i, Y_{t-2} = i_{t-2}, \ldots, Y_1 = i_1),$$

where the last equality follows from the Markov property. The chain Y is often called as **regime** and it is uniquely determined by the parameters π and $\mathbb{P}$. In practice, the regime Y is often defined to be as simple as possible, hence also stationary, irreducible and aperiodic, but we do not need any of these assumptions in this chapter.

Hidden Markov process – the observations. The second component of our model is the random (feature) process $X = \{X_t\}_{t \in \mathbb{N}}$ with X_t taking values in $\mathcal{X} \subseteq \mathbb{R}^d$. We assume that the pair of random processes (Y, X) is such that:

1) given $\{Y_t\}$, the random variables $\{X_t\}$ are conditionally independent,

2) the distribution of X_t depends on $\{Y_t\}$ only through Y_t.

When **1)** and **2)** hold, then the process X is called a **hidden Markov process** and the pair (Y, X) is called a **hidden Markov model (HMM).** The hidden Markov process $X = \{X_1, X_2, \ldots\}$ models the observations so that the random features $X_1, \ldots, X_n$ are the first n vectors of it. The corresponding outcomes of $Y_1, \ldots, Y_n$ are not observed, the Markov chain is *hidden.*

Note that, in general, $X_1, \ldots, X_n$ are neither independent nor identically distributed any more, hence our model allows to drop both assumptions made so far.

Emission distributions. The assumption **2)** states that for any t, the distribution of X_t depends on the regime only through the value of Y_t. Hence the distribution of X_t is independent of t and of the outcomes Y_s, where $s \neq t$. Therefore, to every state $j \in \mathcal{Y}$ corresponds a probability measure P_j such that for every Borel set A and for every $t \geq 1$, it holds

$$P_j(A) = \mathbf{P}(X_t \in A | Y_t = j).$$

The probability measures P_j, $j \in \mathcal{Y}$ are called **emission distributions.** Without loss of generality, we will assume that the emission distributions P_j all have densities f_j with respect to a common reference measure dx. As usually, the Lebesgue'i measure and counting measure (both denoted by dx) are of particular interest.

Finite alphabet HMP. When the emission distributions P_j all have finite (countable) support, then X is referred to as **finite (countable)-alphabet HMP**. In this case, the set $\mathcal{X}$ can be taken as the union of all supports, hence finite (countable). It will be referred to as the *alphabet*. Then, for every state, the density $f_j(x)$ is just the probability to emit the observation or letter x from the state j.

Mixture model: a special case. In a special case, when the columns of the transition matrix $\mathbb{P}$ are all equal to π, i.e. $p_{ij} = \pi$, for all $i, j \in \mathcal{Y}$, then $Y_1, Y_2, \ldots$ are i.i.d. random variables with distribution π. Then also $X_1, X_2, \ldots$ are i.i.d. random variables with density function

$$f(x) = \sum_{j=0}^{k-1} \pi_j f_i(x).$$

This model is sometimes called **mixture model.** Hence HMM also covers the case studied in Chapter 1.

7.2 Forward-backward recursions

7.2.1 Some notations and preliminaries.

Given a set $\mathcal{A}$, integers m and n, $m < n$, and a sequence $a_1, a_2, \ldots \in \mathcal{A}^\infty$, we write a_m^n for the subsequence $(a_m, \ldots, a_n)$. When $m = 1$, it will be often suppressed.

Thus, $x^n := (x_1, \ldots, x_n)$ and $y^n := (y_1, \ldots, y_n)$ stand for the fixed observed and unobserved *realizations* of $X^n = (X_1, \ldots, X_n)$ and $Y^n = (Y_1, \ldots, Y_n)$, respectively. Any sequence $y^n \in \mathcal{Y}^n$ is called a **path**.

Let $1 \leq r \leq m \leq n$. From the definition of HMM, it follows that the conditional density of observing x_r^m given the realization y^n is the product of emission densities. Hence, we define

$$p(x_r^m | y^n) := \prod_{t=r}^{m} f_{y_t}(x_t).$$

In the following, let for any y_u^v, where $1 \leq u \leq v$

$$p(y_u^v) := \mathbf{P}(Y_u = y_u, Y_{u+1} = y_{u+1}, \ldots, Y_v = y_v).$$

Hence, we shall denote the joint probability density of (x_r^m, y^n) by

$$p(x_r^m, y^n) := p(x_r^m | y^n) p(y^n) = \prod_{t=r}^{m} f_{y_t}(x_t) \mathbf{P}(Y^n = y^n) = \prod_{t=r}^{m} f_{y_t}(x_t) \pi_{y_1} \prod_{t=2}^{n} p_{y_{t-1}, y_t}.$$

The unconditional density of a piece of observations x_u^v is now

$$p(x_r^m) := \sum_{y^n \in \mathcal{Y}^n} p(x_r^m, y^n).$$

The joint probability density of (x_r^m, y_u^v), where $1 \leq u \leq v \leq n$ is thus

$$p(x_r^m, y_u^v) := \sum_{s^n \in \mathcal{Y}^n : s_u^v = y_u^v} p(x_r^m, s^n).$$

With joint density $p(x_r^m, y_u^v)$ and unconditional $p(y_u^v)$ and $p(x_r^m)$ we can define the conditional probabilities

$$\mathbf{P}(Y_u = y_u, \ldots, Y_v = y_v | X_r = x_r, \ldots, X_m = x_n) =: p(y_u^v | x_r^m) := \frac{p(x_r^m, y_u^v)}{p(x_r^m)}$$

and conditional densities

$$p(x_r^m | Y_u = y_u, \ldots, Y_v = y_v) =: p(x_r^m | y_u^v) := \frac{p(x_r^m, y_u^v)}{p(y_u^v)}.$$

We shall more closely consider the conditional probabilities $p(y_t | x^n)$. Therefore, they will have a special notation: for any $t \geq 1$ and $j \in \mathcal{Y}$, let

$$p_t(j | x^n) := \mathbf{P}(Y_t = j | X^n = x^n).$$

The probabilities $p_t(\cdot | x^n)$ are called:

- **smoothing** (posterior) probabilities, when $t < n$;
- **filtering** (posterior) probabilities, when $t = n$;
- **prediction** (posterior) probabilities, when $t > n$.

We shall also define

$$p_t(j) := \mathbf{P}(Y_t = j).$$

For stationary chain, clearly $p_t(j) = \pi_j$ for any t, but in general not.

In a special case of finite-alphabet HMP, the joint and conditional densities are just

$$p(x_r^m, y_u^v) = \mathbf{P}(X_r = x_r, \ldots, X_m = x_n; Y_u = y_u, \ldots, Y_v = y_v) = \mathbf{P}(X_r^m = x_r^m; Y_u^v = y_u^v);$$
$$p(x_r^m | y_u^v) = \mathbf{P}(X_r = x_r, \ldots, X_m = x_n | Y_u = y_u, \ldots, Y_v = y_v) = \mathbf{P}(X_r^m = x_r^m | Y_u^v = y_u^v).$$

7.2.2 Forward and backward recursions

Let us for every $j \in \mathcal{Y}$ and $x^n \in \mathcal{X}^n$ define **forward** and **backward** variables

$$\alpha(j,x^t) := p_t(j|x^t)p(x^t), \quad \beta(x^n_{t+1}|j) := \begin{cases} 1, & \text{if } t=n \\ p(x^n_{t+1}|Y_t=j), & \text{if } t<n \end{cases}$$

Clearly with $y_t = j$

$$\alpha(j,x^t) = p(x^t,y_t) = \sum_{y^t: y_t=j} p(x^t,y^t), \tag{7.2.1}$$

$$\beta(x^n_{t+1}|j) = \sum_{y^n_{t+1}} p(x^n_{t+1},y^n_{t+1}|Y_t=j) = \sum_{y^n_{t+1}} p(x^n_{t+1},y^n_{t+1}|y_t). \tag{7.2.2}$$

For any $y^n \in \mathcal{Y}^n$ and $x^n \in \mathcal{X}^n$ the following **factorization** holds

$$\begin{aligned} p(x^n,y^n) =& p(x^n|y^n)p(y^n) = p(x^t|y^n)p(x^n_{t+1}|y^n)p(y^n) = p(x^t|y^t)p(x^n_{t+1}|y^n_t)p(y^n) \\ =& p(x^t|y^t)p(x^n_{t+1}|y^n_t)p(y^t)p(y^n_{t+1}|y_t) = p(x^t,y^t)p(x^n_{t+1}|y^n_t)p(y^n_{t+1}|y_t) \\ =& p(x^t,y^t)p(x^n_{t+1},y^n_{t+1}|y_t). \end{aligned}$$

The second equality follows from the assumption 1) of HMM (conditional independence of x^n), the third inequlity follows from the assumption 2) and the fourth equality follows from the Markov property. The last equality follows from

$$p(x^n_{t+1}|y^n_t)p(y^n_{t+1}|y_t) = \frac{p(x^n_{t+1},y^n_t)}{p(y^n_t)}\frac{p(y^n_t)}{p(y_t)} = \frac{p(x^n_{t+1},y^n_t)}{p(y_t)} = p(x^n_{t+1},y^n_{t+1}|y_t).$$

Summing over all paths y^n passing the state j at t, using (7.2.1) and (7.2.2), from the factorization $p(x^n,y^n) = p(x^t,y^t)p(x^n_{t+1},y^n_{t+1}|y_t)$ we obtain with $y_t = j$

$$p(x^n,y_t) = \sum_{y^n: y_t=j} p(x^n,y^n) = \Big(\sum_{y^t: y_t=j} p(x^t,y^t)\Big)\Big(\sum_{y^n_{t+1}} p(x^n_{t+1},y^n_{t+1}|y_t)\Big) = \alpha(j,x^t)\beta(x^n_{t+1}|j). \tag{7.2.3}$$

From (7.2.3), it follows that α and β-variables can be used for finding $p_t(j|x^n)$:

$$p(x^n) = \sum_{y_t \in \mathcal{Y}} p(x^n,y_t) = \sum_{j\in\mathcal{Y}} \alpha(j,x^t)\beta(x^n_{t+1}|j), \tag{7.2.4}$$

$$p_t(j|x^n) = \frac{\alpha(j,x^t)\beta(x^n_{t+1}|j)}{\sum_{j\in\mathcal{Y}}\alpha(j,x^t)\beta(x^n_{t+1}|j)}, \tag{7.2.5}$$

In a special case of finite-alphabet HMP the variables are

$$\alpha(j,x^t) = \mathbf{P}(X^t = x^t; Y_t = j), \quad \beta(x^n_{t+1}|j) = \mathbf{P}(X^n_{t+1} = x^n_{t+1}|Y_t = j).$$

and the equation (7.2.3) is, thus,

$$\mathbf{P}(X^n = x^n; Y_t = j) = \mathbf{P}(X^t = x^t; Y_t = j)\mathbf{P}(X^n_{t+1} = x^n_{t+1}|Y_t = j).$$

Standard recursions. From the factorization

$$p(x^n, y^n) = p(x^t, y^t)p(x^n_{t+1}, y^n_{t+1}|y_t) \tag{7.2.6}$$

with $n = t+1$, we get

$$p(x^{t+1}, y^{t+1}) = p(x^t, y^t)p(x_{t+1}, y_{t+1}|y_t). \tag{7.2.7}$$

Summing over y^{t-1}, we get

$$p(x^{t+1}, y^{t+1}_t) = p(x^t, y_t)p(x_{t+1}, y_{t+1}|y_t)$$

and summing over y_t, we obtain

$$p(x^{t+1}, y_{t+1}) = \sum_{y_t} p(x^t, y_t)p(x_{t+1}, y_{t+1}|y_t).$$

Finally, since

$$p(x_{t+1}, y_{t+1}|y_t) = p(y_{t+1}|y_t)f_{y_{t+1}}(x_{t+1}) = p_{y_t y_{t+1}} f_{y_{t+1}}(x_{t+1})$$

we have obtained a **forward recursion** for calculation α-variables:

$$\alpha(j, x^{t+1}) = \sum_{i=0}^{k-1} \alpha(i, x^t)p_{ij}f_j(x_{t+1}).$$

Similarly, one can show the **backward recursion** for calculation β-variables (see [27]):

$$\beta(x^n_t|j) = \sum_{i=0}^{k-1} \beta(x^n_{t+1}|i)p_{ji}f_i(x_t).$$

In a special case of finite-alphabet HMP the forward and backward recursions are

$$\mathbf{P}(Y_{t+1} = j; X^{t+1} = x^{t+1}) = \sum_i \mathbf{P}(Y_t = i; X^t = x^t)\mathbf{P}(Y_{t+1} = j|Y_t = i)\mathbf{P}(X_{t+1} = x_{t+1}|Y_{t+1} = j)$$

$$\mathbf{P}(X^t_t = x^n_t|Y_{t-1} = j) = \sum_i \mathbf{P}(X^n_{t+1} = x^n_{t+1}|Y_t = i)\mathbf{P}(Y_t = i|Y_{t-1} = j)\mathbf{P}(X_t = x_t|Y_t = i).$$

Derin's recursion. The above-described (standard) backward and forward recursions are not numerically stable, hence they are not very practical. Therefore, their normalized versions as well as several alternative recursions for calculating $p_t(j|x^n)$ in simular fashion are used. For example, the so-called **Derin's recursion** for finding $p_t(j|x^n)$ backward is as follows:

$$p_t(j|x^n) = p_t(j|x^t) \sum_{i=0}^{k-1} \frac{p_{ji}p_{t+1}(i|x^n)}{p_{t+1}(i|x^t)}.$$

For proof, see [27]. Here the smoothing probabilities $p_t(\cdot|x^t)$ and prediction probabilities $p_{t+1}(\cdot|x^t)$ can be found by forward recursion as follows:

$$p_1(j|x_1) = \frac{\pi_j f_j(x_1)}{\sum_i \pi_i f_i(x_1)},\ p_{t+1}(j|x^t) = \sum_i p_{ij} p_t(i|x^t),\ p_{t+1}(j|x^{t+1}) = \frac{p_{t+1}(j|x^t) f_j(x_{t+1})}{\sum_i p_{t+1}(i|x^t) f_i(x_{t+1})} \tag{7.2.8}$$

Exercise: Prove (7.2.8).

7.2.3 Conditional chain

The following proposition shows an important property: Y is an conditioonally inhomogeneous Markov chain given X.

Proposition 7.2.1 *Given the observations x^n, the regime has Markov property:*

$$\mathbf{P}(Y_{t+1} = j|Y_t = i, Y^{t-1} = y^{t-1}; X^n = x^n) = \mathbf{P}(Y_{t+1} = j|Y_t = i; X^n = x^n). \tag{7.2.9}$$

Moreover, the conditional transition probablities

$$\mathbf{P}(Y_{t+1} = j|Y_t = i; X^n = x^n)$$

depend on the observations x^n_{t+1}, only.

Proof. It suffices to show that for any $y^{t+1} \in \mathcal{Y}^{t+1}$, the following equality holds

$$p(y_{t+1}|y^t, x^n) = p(y_{t+1}|y_t, x^n_{t+1}). \tag{7.2.10}$$

Indeed, (7.2.10) shows that the conditional transition probability $p(y_{t+1}|y^t, x^n)$ does not depend on y^{t-1}, hence Markov property (7.2.9), holds; the equality (7.2.10) also shows that given y_t the conditional distribution of y^n_{t+1} is independent of x^t. To show (7.2.10), recall the factorization (7.2.6):

$$p(x^n, y^n) = p(x^t, y^t) p(x^n_{t+1}, y^n_{t+1}|y_t).$$

Summing over y^n_{t+2} and y^n_{t+1}, we obtain

$$p(x^n, y^{t+1}) = p(x^t, y^t) p(x^n_{t+1}, y_{t+1}|y_t), \quad p(x^n, y^t) = p(x^t, y^t) p(x^n_{t+1}|y_t).$$

Thus,

$$p(y_{t+1}|y^t, x^n) = \frac{p(x^n, y^{t+1})}{p(x^n, y^t)} = \frac{p(x^t, y^t) p(x^n_{t+1}, y_{t+1}|y_t)}{p(x^t, y^t) p(x^n_{t+1}|y_t)} = \frac{p(x^n_{t+1}, y_{t+1}|y_t)}{p(x^n_{t+1}|y_t)} = p(y_{t+1}|y_t, x^n_{t+1}).$$

■

7.3 Segmentation

Let $x^n = x_1, \ldots, x_n$ be the given observations. We do not know the corresponding regime (the Markov chain is hidden), instead we assume that our model (HMM) is exactly known. The **problem of segmentation (problem of decoding)** is to estimate or prognose the hidden state sequence $y_1, \ldots, y_n$. This can be regarded as a classification problem, where input $x^n \in \mathcal{X}^n$, the set of outputs (classes) is $\mathcal{Y}^n$ and the classifier is a function

$$g = (g_1, \ldots, g_n) : \mathcal{X}^n \to \mathcal{Y}^n. \tag{7.3.1}$$

However, since the input and the output (the set of classes) both depend on n, the segmentation is the problem in its own rights.

7.3.1 Decision theory for HMM's

What is the best classifier (7.3.1)? To answer that, let us apply some ideas of Bayesian decision theory also for HMM's case. Recall the Bayesian decision theory – the Bayes classifier g^* is the one that for every feature vector x minimizes the conditional risk at x, i.e.

$$g^*(x) = \arg\min_{j \in \mathcal{Y}} R(j|x).$$

Also recall that the conditional risk was obtained via loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$, where $L(y, j)$ is the loss of misclassifying the true class y as j. With loss function, for any $j \in \mathcal{Y}$,

$$R(j|x) = \sum_{y \in \mathcal{Y}} L(y, j) p(y|x).$$

Suppose now that we have defined the conditional risk function also for the segmentation problem. Thus, for every x^n, we have the conditional risk

$$R(\cdot|x^n) : \quad \mathcal{Y}^n \to [0, \infty]$$

so that $R(s^n|x^n)$ measures the goodness of path $s^n \in \mathcal{Y}^n$ given x^n. Just like in Bayesian decision theory, the risk of classifier $g = (g_1, \ldots, g_n)$ is then the expectation of conditional risk

$$R(g) := ER(g(X^n)|X^n)$$

and the classifier with minimal risk (over all possible classifiers) is called **Bayes classifier** that can be obtained by minimizing the conditional risk:

$$g^*(x^n) := \arg\min_{s^n \in \mathcal{Y}^n} R(s^n|x^n).$$

Loss function. The conditional risk $R(\cdot|x^n)$ depends on the task and (just like in Bayesian desicion theory) it is often meaningful to define it via **loss function**

$$L : \mathcal{Y}^n \times \mathcal{Y}^n \to [0, \infty],$$

where $L(y^n, s^n)$ is loss, when the actual state sequence is y^n and the prognose is s^n. The conditional expectation

$$R(s^n|x^n) := E[L(Y^n, s^n)|X^n = x^n]$$

is the conditional risk of s^n. Then, for any classifier g, the risk $R(g)$ is just the expected loss

$$R(g) = EL(Y^n, g(X^n))$$

and the Bayes classifier is

$$g^*(x^n) = \arg\min_{s^n \in S^n} \sum_{y^n \in S^n} L(y^n, s^n)\mathbf{P}(Y^n = y^n|X^n = x^n).$$

7.3.2 Viterbi aligment

Let us start with symmetric loss-function:

$$L(y^n, s^n) = \begin{cases} 1, & \text{when } y^n \neq s^n; \\ 0, & \text{when } y^n = s^n. \end{cases} \tag{7.3.2}$$

Then the conditional risk (in this case denoted by R_∞) is

$$R_\infty(s^n|x^n) = 1 - \mathbf{P}(Y^n = s^n|X^n = x^n)$$

and the Bayes classifier – for symmetric loss denoted by v – maps every sequence of observations into sequence s^n with maximum likelihood.

$$v(x^n) := \arg\max_{s^n \in S^n} \mathbf{P}(Y^n = s^n|X^n = x^n).$$

The maximum-likelihood state sequence $v(x^n)$ is known as **Viterbi alignment**. It inherits its name from the **Viterbi algorithm** – a dynamic programming algorithm for finding $v(x^n)$. Partially due to the simplicity of Viterbi algorithm, Viterbi alignment is by far most popular classifier in segmentation.

Viterbi algorithm. Recall the problem: given observations x^n, we are trying to find (all) path(s) $v^n \in \mathcal{Y}^n$ that maximizes the joint density $p(x^n, y^n)$ over all $y^n \in \mathcal{Y}^n$. Since the size of the search space is k^n, the disrect optimization is imposible even for moderate n. The Viterbi algorithm allows to solve the problem with complexity $O(n)$.

Let us introduce some notation. Since x^n is fixed, we skip it from the notation. Let, for every $t = 1, \ldots, n$ and state $j \in \mathcal{Y}$, the **(Viterbi) scores** be

$$\delta_t(j) = \max_{y^t, y_t = j} p(x^t, y^t).$$

Hence $\delta_t(j)$ is the maximum joint likelihood over all paths ending at the state j. The scores can be found recursively (in t). Indeed, let y^{t+1} be a path that ends with j, i.e $y_{t+1} = j$. Then, using the factorization again,

$$p(x^{t+1}, y^{t+1}) = p(x^t, y^t)p(x_{t+1}, y_{t+1}|y_t) = p(x^t, y^t)p_{y_t j} f_j(x_{t+1}).$$

Let $s^t \in \mathcal{Y}^t$ be the path that maximizes $p(x^t, y^t)p_{y_t j}$ over all paths. If it ends at the state i, i.e. $s_t = i$, then

$$p(x^t, s^t)p_{s_t j} = p(x^t, s^t)p_{ij}$$

so that s^t has to be the path that maximizes $p(x^t, y^t)$ over all paths y^t ending with i. This is **Bellman's optimality principle**. In other words, if $s_t = i$, then $p(x^t, s^t) = \delta_t(i)$. That holds for every state i, thus

$$\delta_{t+1}(j) = \max_i \big(\max_{y^{t+1}: y_{t+1} = i} p(x^t, y^t)p_{ij} \big) f_j(x^{t+1}) = \max_i \big(\delta_t(i)p_{ij} \big) f_j(x_{t+1}).$$

Hence, we have the following (Viterbi) recursion for finding $\delta_t(i)$ for every i and t:

$$\delta_1(j) = \pi_j f_j(x_1), \quad \delta_{t+1}(j) = \max_i \big(\delta_t(i)p_{ij} \big) f_j(x_{t+1}) \tag{7.3.3}$$

Viterbi algorithm is a standard dynaming programming algorithm: at each $t = 1, \ldots, n$ the scores $\delta_t(j)$ are calculated using recursion (7.3.3). By that, the algorithm stores

$$i_t(j) := \arg\max_i \delta_t(i)p_{ij}, \quad t = 1, \ldots, n-1.$$

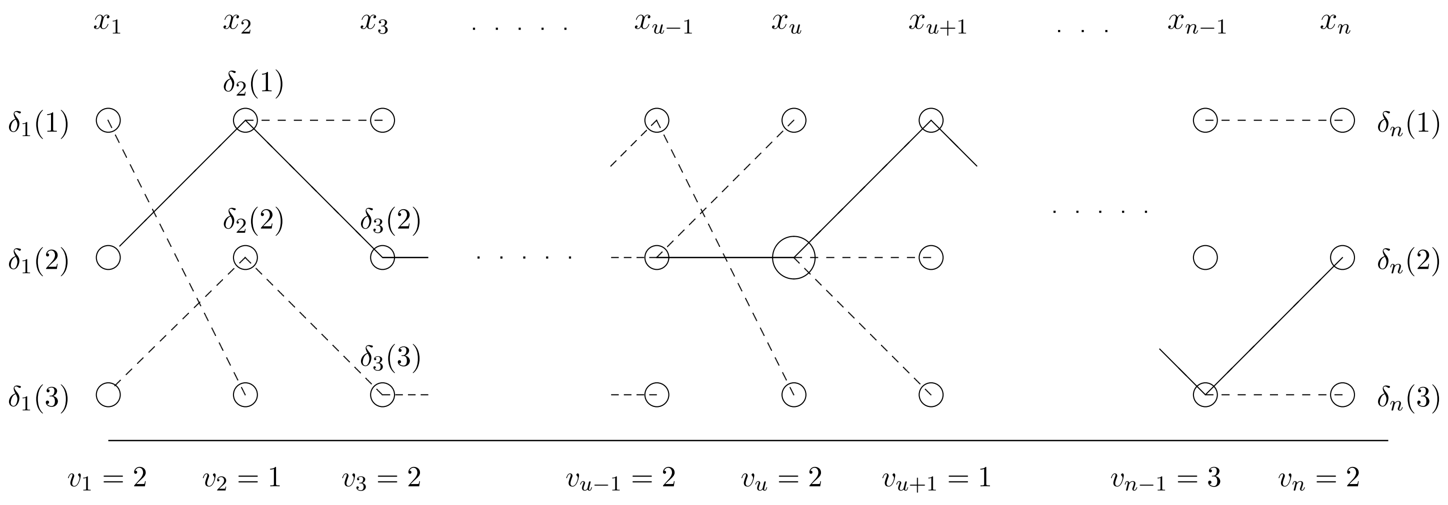
In case of ties, any choise will do. The solution can now found by *backtracking* as follows

$$v_n = \arg\max_j \delta_n(j), \quad v_t = i_t(v_{t+1}), \quad t = n-1, \ldots, 1.$$

Viterbi algorithm

1. **Initialize:** For every $j \in \mathcal{Y}$, define $\delta_1(j) := \pi_j f_j(x_1)$;
2. **Do for** $t = 1, \ldots, n-1$**:**
   - Update
   $$\delta_{t+1}(j) = \max_i \big(\delta_t(i) p_{ij}\big) f_j(x_{t+1}); \tag{7.3.4}$$
   - Record
   $$i_t(j) := \arg\max_i \delta_t(i) p_{ij}$$
3. **Output:** Find Viterbi alignment v^n by backtracking:
$$v_n := \arg\max_j \delta_n(j), \quad v_t = i_t(v_{t+1}), \quad t = n-1, \ldots, 1.$$

The following picture illustrates Viterbi algorithm in action. The solid lines indicates the output, the dashed lines indicate $i_t(j)$.



7.3.3 PMAP alignment

The symmetric loss (7.3.2) penalizes all differences alike: no matter whether y^n and s^n differ from one entry or all entries, the penalty is one. Often it is natural to penalize every different entry. This can be done by **pointwise loss-function**

$$l : \mathcal{Y} \times \mathcal{Y} \to [0, \infty) \quad \text{where } l(s,s) = 0, \quad \forall s \in \mathcal{Y}. \tag{7.3.5}$$

Using l, we can define loss-function L as follows

$$L(y^n, s^n) := \sum_{t=1}^{n} l(y_t, s_t). \tag{7.3.6}$$

With (7.3.6), the conditional risk is

$$R(s^n|x^n) = E[L(Y^n, s^n)|X^n = x^n] = \sum_{t=1}^{n} E[l(Y_t, s_t)|X^n = x^n] \tag{7.3.7}$$

and minimizing $R(s^n|x^n)$ over s^n equals to minimizing $E[l(Y_t, s_t)|X^n = x^n]$ over s_t at every t. Hence, the Bayes classifier is obtained *pointwise*: $g^* = (g_1^*, \ldots, g_n^*)$, where

$$g_t^*(x^n) = \arg\min_{s \in \mathcal{Y}} E[l(Y_t, s)|X^n = x^n].$$

Counting errors. The most popular choice for l is, again, symmetric (pointwise) loss:

$$l(y,s) = \begin{cases} 0, & \text{if } y = s; \\ 1, & \text{if } y \neq s. \end{cases} .$$

Then the loss-function L counts the differences between y^n and s^n when compared pairwise: $L(y^n, s^n)$ is the number of pairwise errors or the difference in **Hamming distance**. Therefore, the conditional risk (in this case denoted by R_1) $R_1(s^n|x^n)$ measures expected number of misclassification errors of s^n given the observations are x^n. From (7.3.7), it follows that

$$R_1(s^n|x^n) := n - \sum_{t=1}^{n} \mathbf{P}(Y_t = s_t|X^n = x^n),$$

hence the Bayes classifier in this case – let us denote it by u – is the one that at each time t chooses the state with conditional probability:

$$u_t(x^n) = \arg\max_j \mathbf{P}(Y_t = j|X^n = x^n) = \arg\max_j p_t(j|x^n), \quad t = 1, \ldots, n.$$

We shall call u as **PMAP (pointwise maximum aposteriori) alignment**. Other names encountered: *marginal posterior mode*, *maximum posterior marginals*, *optimal symbol-by-symbol detection*, *symbol-by-symbol MAP estimation*, *MAP-state estimation*.

We know that the conditional probabilities $p_t(j|x^n)$ can be found with complexity $O(n)$ by several forward-backward recursions. This makes PMAP-classifier implementable and that is why PMAP alignment is the second most popular choice in practice.

Remark: Note that for mixture model these two alignments – Viterbi and PMAP alignment – coincide.

Logarithmic risks. Let us denote the logarithmic counterparts of R_∞ and R_1 risks:

$$\bar{R}_\infty(s^n|x^n) := -\ln \mathbf{P}(Y^n = s^n|x^n)$$

$$\bar{R}_1(s^n|x^n) := -\sum_{t=1}^{n} \ln \mathbf{P}(Y_t = s_t|x^n) = -\sum_{t=1}^{n} \ln p_t(s_t|x^n).$$

Clearly Viterbi alignment $v(x^n)$ minimizes $\bar{R}_\infty(\cdot|x^n)$ and PMAP alignment $u(x^n)$ minimizes $\bar{R}_1(\cdot|x^n)$.

7.3.4 Between PMAP and Viterbi

If the aim is to minimize the number of errors, then one should use PMAP-alignment. Unfortunately, it can be with very low or zero conditional likelihood, i.e. it might be that $p(u^n|x^n) = 0$. We call paths that have zero conditional likelihood **inadmissible**. This drawback is probably the main reason, why Viterbi alignment, although it might make in average more errors is preferred over PMAP alignment. In the following, we consider some options how to adjust PMAP alignment so that it still results an output with possible small number of expected misclassification errors, but at the same time remains admissible.

Restricted R_1-risk. The simplest solution is **restricted R_1-risk**:

$$\min_{y^n : p(y^n|x^n) > 0} R_1(y^n|x^n) \quad \Leftrightarrow \quad \max_{y^n : p(y^n|x^n) > 0} \sum_{t=1}^{n} p_t(y_t|x^n). \tag{7.3.8}$$

This problem can be solved by dynamic programming algorithm similar to the one of Viterbi algorithm.

Algorithm for restricted optimization:

1. **Initialize:**
   - Using forward-backward recursions, compute $p_t(j|x^n)$ for every $1 \leq t \leq n$ and $j \in \mathcal{Y}$;
   - For every $j \in \mathcal{Y}$, set
$$\delta_1(j) := p_1(j|x^n);$$

2. **Do for $t = 1, \ldots, n-1$:**

   - For every $j \in \mathcal{Y}$, update
   $$\delta_{t+1}(j) = \big(\max_i \delta_t(i) r_{ij} + p_{t+1}(j|x^n)\big) r_j^{t+1}, \tag{7.3.9}$$
   where
   $$r_{ij} = \mathbb{I}_{\{p_{ij}>0\}}, \quad r_j^t = \mathbb{I}_{\{p_t(j|x^n)>0\}}.$$
   - Record
   $$i_t(j) := \arg\max_i \delta_t(i) r_{ij}, \quad t = 1, \ldots, n-1.$$

3. **Output:** Find the optimal alignment s^n by backtracking:
$$s_n := \arg\max_j \delta_n(j), \quad s_t = i_t(s_{t+1}), \quad t = n-1, \ldots, 1.$$

Note that without the multipliers r_{ij} and r_j^t, the algorithm would indeed result PMAP-alignment, because, for every t and j, $\delta_{t+1}(j) = \max_i \delta_t(i) + p_{t+1}(j|x^n)$ so that
$$\max_j \delta_{t+1}(j) = \max_i \delta_t(i) + \max_j p_{t+1}(j|x^n).$$
If $p_{t+1}(j|x^n) > 0$, then $\arg\max_i \delta_t(i) r_{ij} > 0$, since otherwise there would no admissible path passing j at $t+1$ with positive likelihood. This contradicts $p_{t+1}(j|x^n) > 0$.

Finally note that the recursion (7.3.9) is equivalent to
$$\begin{aligned} \delta_1(j) &:= p_1(j|x^n), \\ \delta_{t+1}(j) &:= \max_i (\delta_t(i) + \log r_{ij}) + p_{t+1}(j|x^n) + \log r_j^{t+1}. \end{aligned} \tag{7.3.10}$$

Restricted $\bar{R}_1$-risk. In the presence of restrictions, (7.3.8) is not necessarily the solution of the following problem:
$$\min_{s^n: p(s^n|x^n)>0} \bar{R}_1(s^n|x^n) \quad \Leftrightarrow \quad \max_{s^n: p(s^n|x^n)>0} \sum_{t=1}^n \ln p_t(s_t|x^n). \tag{7.3.11}$$
The solution of (7.3.11) is sometimes called as the **posterior Viterbi decoding (PVD)** and it can be found the similar algorithm, where the recursion (7.3.9) is replaced by the following recursion
$$\begin{aligned} \delta_1(j) &:= p_1(j|x^n), \\ \delta_{t+1}(j) &:= \max_i \delta_t(i) r_{ij} \times p_{t+1}(j|x^n). \end{aligned} \tag{7.3.12}$$
Recursion (7.3.12) is clearly equivalent to
$$\begin{aligned} \delta_1(j) &:= \log p_1(j|x^n), \\ \delta_{t+1}(j) &:= \max_i \big(\delta_t(i) + \log r_{ij}\big) + \log p_{t+1}(j|x^n). \end{aligned} \tag{7.3.13}$$

Towards increasing the probability: $\bar{R}_1$-risk. Although admissible minimizers of R_1 and $\bar{R}_1$ risk are by definition of positive probability, this probability might still be very small. Indeed, in the above recursions, the weight r_{ij} is 1 even when p_{ij} is very small. Hence, in recursion (7.3.13) , we replace r_{ij} by the true transition (initial) probability p_{ij} and add $\log \pi_j$ to $\log p_1(j|x^n)$. Thus the modified recursion (7.3.13) is now

$$\begin{aligned} \delta_1(j) &:= \log p_1(j|x^n) + \log \pi_j \\ \delta_{t+1}(j) &:= \max_i \big(\delta_t(i) + \log p_{ij}\big) + \log p_{t+1}(j|x^n). \end{aligned} \tag{7.3.14}$$

Obviously, similar changes might be done in recursion (7.3.12). It is not difficult to see that with the recursion (7.3.14) the algorithm for restricted optimization solves the following problem

$$\max_{s^n} \Big[\sum_{t=1}^{n} \log p_t(s_t|x^n) + \log p(s^n)\Big] \quad \Leftrightarrow \quad \min_{s^n} \Big[\bar{R}_1(s^n|x^n) + h(s^n)\Big], \tag{7.3.15}$$

where the penalty term

$$h(s^n) \ = -\log p(s^n) \ =: \bar{R}_\infty(s^n) \tag{7.3.16}$$

is the prior log-likelihood risk which does not depend on the data.

<u>More general problem.</u> The recursion (7.3.14) immediately generalize as follows:

$$\begin{aligned} \delta_1(j) &:= \log p_1(j|x^n) + C \log \pi_j, \\ \delta_{t+1}(j) &:= \max_i \big(\delta_t(i) + C \log p_{ij}\big) + \log p_{t+1}(j|x^n), \end{aligned} \tag{7.3.17}$$

solving the following optimization problem

$$\min_{s^n} \Big[\bar{R}_1(s^n|x^n) + Ch(s^n)\Big], \tag{7.3.18}$$

where $C > 0$ is a regularization or trade-off constant and $h(s^n) = \bar{R}_\infty(s^n)$. Then, PVD, i.e. the problem solved by the original recursions (7.3.12) and (7.3.13), can be recovered by taking C sufficiently small. Alternatively, the PVD problem can also be formally written in the form (7.3.18) with $C = \infty$ and $h(s^n)$ given, for example, by $\mathbb{I}_{\{p(s^n)=0\}}$.

Towards increasing the probability: R_1-risk. What if the actual probabilities p_{ij} (π_j) were also used in the optimal accuracy/PMAP decoding, i.e. optimization (7.3.9)-(7.3.10)? It appears more sensible to replace the indicators r_{ij} with p_{ij} (and adding π_j) in (7.3.10). The new recursion is now

$$\begin{aligned} \delta_1(j) &:= p_1(j|x^n) + \log \pi_j, \\ \delta_{t+1}(j) &:= \max_i (\delta_t(i) + \log p_{ij}) + p_{t+1}(j|x^n) + \log r_j^{t+1}. \end{aligned} \tag{7.3.19}$$

This solves the following problem:

$$\max_{s^n : p(s^n|x^n)>0} \Big[\sum p_t(s_t|x^n) + \log p(s^n)\Big] \quad \Leftrightarrow \quad \min_{s^n : p(s^n|x^n)>0} \Big[R_1(s^n|x^n) + \bar{R}_\infty(s^n)\Big]. \tag{7.3.20}$$

7.3.5 Combined risks

Motivated by the previous section, we consider the following general problem

$$\min_{s^n}\Big[C_1\bar{R}_1(s^n|x^n)+C_2\bar{R}_\infty(s^n|x^n)+C_3\bar{R}_1(s^n)+C_4\bar{R}_\infty(s^n)\Big], \tag{7.3.21}$$

where $C_i\geq 0$, $i=1,2,3,4$, $\sum_{i=1}^4 C_i>0$ and (recall)

$$\bar{R}_1(s^n|x^n)=-\sum_{t=1}^n\log p_t(s_t|x^n),\quad \bar{R}_\infty(s^n|x^n)=-\log p(s^n|x^n),$$
$$\bar{R}_1(s^n):=-\sum_{t=1}^n\log p_t(s_t),\quad \bar{R}_\infty(s^n)=-\log p(s^n)=-[\log\pi_{s_1}+\sum_{t=1}^{n-1}\log p_{s_ts_{t+1}}].$$

Some important special cases:

- the combination $C_1>0, C_2=C_3=C_4=0$ yields the PMAP case;
- the combination $C_2>0, C_1=C_3=C_4=0$ corresponds to the Viterbi decoding;
- the combination $C_3>0, C_1=C_2=C_4=0$ maximizes
$$-\bar{R}_1(s^n)=\sum_{t=1}^n\log p_t(s_t)=\sum_{t=1}^n\log\mathbf{P}(Y_t=s_t),$$
sometimes called *marginal prior mode* decoding;
- the combination $C_4>0, C_1=C_2=C_3=0$ maximizes $p(s^n)=\mathbf{P}(Y^n=s^n)$, sometimes called *maximum a priori* decoding;
- the combination case $C_1>0, C_4>0, C_2=C_3=0$ subsumes (7.3.18):
$$\min_{s^n}\Big[C_1\bar{R}_1(s^n|x^n)-C_4\log p(s^n)\Big];$$
- the combination $C_1=C_3=0$ is the problem
$$\min_{s^n}\Big[\bar{R}_\infty(s^n|x^n)+C\bar{R}_\infty(s^n)\Big]$$
and its solution is a generalization of the Viterbi decoding that allows one to suppress $(C>0)$ contribution of the data;
- the combination $C_1>0, C_2>0, C_3=C_4=0$ is the problem
$$\min_{s^n}\Big[C_1\bar{R}_1(s^n|x^n)+C_2\bar{R}_\infty(s^n|x^n)\Big],$$
and when $0<C_2\ll C_1$, it equals to minimizing $\bar{R}_1(s^n|x^n)$ under the condition
$$\bar{R}_\infty(s^n|x^n)<\infty\quad\Leftrightarrow\quad p(s^n|x^n)>0.$$
Thus the problem is the same as (7.3.11) and the solution of this problem is PVD-alignment.

Remark. It is important to note that with $C_2 > 0$ every solution of (7.3.21) is admissible. No less important, and perhaps a bit less obvious, is that $C_1, C_4 > 0$ also guarantees admissibility of the solutions.

Dynamic programming algorithm for solving (7.3.21) With suitable recursion, the algorithm for restricted optimization can be used to solve 7.3.21). To state the recursion, let

$$g_t(j) := C_1 \log p_t(j|x^n) + C_2 \log f_j(x_t) + C_3 \log p_t(j).$$

Note that the function g_t depends on the entire data x^n and they involve $p_t(j|x^n)$ as well as $p_t(j)$. The general recursion is the following

$$\begin{aligned} \delta_1(j) &:= C_1 \log p_1(j|x^n) + (C_2 + C_3 + C_4) \log \pi_j + C_2 \log f_j(x_1), \\ \delta_{t+1}(j) &:= \max_i \big(\delta_t(i) + (C_2 + C_4) \log p_{ij}\big) + g_{t+1}(j). \end{aligned} \tag{7.3.22}$$

Some important special cases (check):

- the combination $C_1 > 0, C_2 = C_3 = C_4 = 0$ yields the PMAP alignment ;
- the combination $C_2 > 0, C_1 = C_3 = C_4 = 0$ gives

  $$\begin{aligned} \delta_1(j) &:= C_2 \log \big(\pi_j f_j(x_1)\big), \\ \delta_{t+1}(j) &:= \max_i \big(\delta_t(i) + C_2 \log p_{ij}\big) + C_2 \log f_j(x_{t+1}) = \max_i \Big(\delta_t(i) + C_2 \log \big(p_{ij} f_j(x_{t+1})\big)\Big) \end{aligned}$$

  and that equals to Viterbi recursion (7.3.4);
- the combination $C_3 > 0, C_1 = C_2 = C_4 = 0$ gives the recursion

  $$\begin{aligned} \delta_1(j) &:= C_3 \log \pi_j, \\ \delta_{t+1}(j) &:= \max_i \delta_t(i) + C_3 \log p_{t+1}(j) \end{aligned}$$

  that, indeed, maximizes $\sum_{t=1}^n \log p_t(s_t)$;
- the combination $C_4 > 0, C_1 = C_2 = C_3 = 0$ gives the recursion

  $$\begin{aligned} \delta_1(j) &:= C_4 \log \pi_j, \\ \delta_{t+1}(j) &:= \max_i \big(\delta_t(i) + C_4 \log p_{ij}\big) \end{aligned}$$

  that, indeed, maximizes $\log p(s^n)$;
- the combination case $C_1 > 0, C_4 > 0, C_2 = C_3 = 0$ gives the recursion

  $$\begin{aligned} \delta_1(j) &:= C_1 \log p_1(j|x^n) + C_4 \log \pi_j, \\ \delta_{t+1}(j) &:= \max_i \big(\delta_t(i) + C_4 \log p_{ij}\big) + C_1 \log p_{t+1}(j|x^n) \end{aligned} \tag{7.3.23}$$

that for $C_1 = 1$ and $C_4 = C$ is exactly the same as (7.3.17). Clearly (7.3.23) solves the problem

$$\max_{s^n} \Big[C_1 \sum_t \log p_t(s_t|x^n) + C_4 \log p(s^n) \Big] = \min_{s^n} \Big[C_1 \bar{R}_1(s^n|x^n) + C_4 \bar{R}_\infty(s^n) \Big];$$

- the combination $C_2 > 0, C_4 > 0, C_1 = C_3 = 0$ gives the recursion

$$\begin{aligned} \delta_1(j) &:= C_2 \log \big(\pi_j f_j(x_1) \big) + C_4 \log \pi_j, \\ \delta_{t+1}(j) &:= \max_i \Big(\delta_t(i) + C_2 \big(\log p_{ij} \log f_j(x_{t+1}) \big) + C_4 \log p_{ij} \Big), \end{aligned}$$

  that solves

$$\max_{s^n} \Big[C_2 \log p(s^n|x^n) + C_4 \log p(s^n) \Big] = \min_{s^n} \Big[C_2 \bar{R}_\infty(s^n|x^n) + C_4 \bar{R}_\infty(s^n) \Big], ;$$

- the combination $C_1 > 0, C_2 > 0, C_3 = C_4 = 0$ gives the recursion

$$\begin{aligned} \delta_1(j) &:= C_1 \log p_1(j|x^n) + C_2 \log \big(\pi_j f_j(x_1) \big), \\ \delta_{t+1}(j) &:= \max_i \Big(\delta_t(i) + C_2 \log \big(p_{ij} f_j(x_{t+1}) \big) \Big) + C_1 \log p_{t+1}(j|x^n) \quad (7.3.24) \end{aligned}$$

  that solves

$$\max_{s^n} \Big[C_1 \sum_t \log p_t(s_t|x^n) + C_2 \log p(s^n|x^n) \Big] = \min_{s^n} \Big[C_1 \bar{R}_1(s^n|x^n) + C_2 \bar{R}_\infty(s^n|x^n) \Big].$$

7.3.6 k-block alignments

Rabiner's k blocks. In his celebrated tutorial [25], L. Rabiner proposes instead of maximize the expected number of correctly decoded pairs or triples of (adjacent) states. Hence, with k being the length of the overlapping block ($k = 2, 3, \ldots$), he proposed to maximize the sum

$$p(s^k|x^n) + p(s_2^{k+1}|x^n) + p(s_2^{k+2}|x^n) + \cdots + p(s_{n-k+1}^n|x^n). \qquad (7.3.25)$$

With $k = 1$, (7.3.25) is the sum $\sum_t p_t(s_t|x^n)$, hence in case $k = 1$, the maximizer of (7.3.25) is PMAP-alignment.

Formally, maximizing (7.3.25) equals to minimizing the conditional risk

$$R_k(s^n|x^n) := E\big[L_k(Y^n, s^n)|X^n = x^n\big], \qquad (7.3.26)$$

where L_k is the following loss function:

$$L_k(y^n, s^n) := \mathbb{I}_{\{s^k \neq y^k\}} + \mathbb{I}_{\{s_2^{k+1} \neq y_2^{k+1}\}} + \mathbb{I}_{\{s_3^{k+2} \neq y_3^{k+1}\}} + \cdots + \mathbb{I}_{\{s_{n-k+1}^n \neq y_{n-k+1}^n\}}.$$

It is natural to think that minimizers of R_k-risk – **Rabiner's k-block alignment** – "move" towards Viterbi paths "monotonically" as k increases to n. Indeed, when $k = n$, then (7.3.25) is $p(s^n|x^n)$, hence the maximizer of it is Viterbi alignment. However, as the Example below shows, minimizers of $R_k(s^n|x^n)$ are not guaranteed to be admissible for $k > 1$.

Admissible k-blocks. The above-mentioned drawback (possible non-admissibility) is easily overcome when the sum in (7.3.25) is replaced by the product. Or, equivalently, the probabilities $p(s_t^{t+k-1}|x^n)$ are replaced by $\log p(s_t^{t+k-1}|x^n)$ in the sum (7.3.25). Certainly, except the case $k=1$, these problems are not equivalent, but with the product in place of the sum the k-block idea works well. Namely, the longer the block, the larger the resulting path probability and, more importantly, the solution is clearly guaranteed to be positive already for $k=2$. Indeed, if $p(s^n)=0$, then there must exist a pair (transition) s_t^{t+1} such that $p(s_t^{t+1})=0$. Then $p(s_t^{t+1}|x^n)=0$ and if one multiplier equals to zero, so does the whole product. If $p(s^n)>0$, but $p(s^n|x^n)=0$, then there must exists at least one x_t so that $f_{s_t}(x_t)=0$. That, in turn implies $p(s_t|x^n)=0$ and, therefore $p(s_t^{t+1}|x^n)=0$ for *any pair* (s_t,s_{t+1}) that begins with s_t.

By replacing the probabilities $p(s_t^{t+k-1}|x^n)$ by $\log p(s_t^{t+k-1}|x^n)$, we would get a natural candidate for logarithmic version of R_k-risk as follows

$$\log p(s^k|x^n)+\log p(s_2^{k+1}|x^n)+\log p(s_2^{k+2}|x^n)+\cdots+\log p(s_{n-k+1}^n|x^n).$$

However, to get a nice connection with above defined general family of alignments, we define the logarithmic counterpart of R_k slightly different. Namely, let

$$\begin{aligned}U_1^k &:= p(s_1)\cdots p(s_1^{k-2})p(s_1^{k-1})\\ U_2^k &:= p(s_1^k)p(s_2^{k+1})\cdots p(s_{n-k}^{n-1})p(s_{n-k+1}^n)\\ U_3^k &:= p(s_{n-k+2}^n)p(s_{n-k+3}^n)\cdots p(s_n).\end{aligned}$$

and let

$$U_k(s^n):=U_1^k(s^n)\cdot U_2^k(s^n)\cdot U_3^k(s^n).$$

The logarithmic version of R_k-risk will be defined as

$$\bar R_k(s^n):=-\log U_k(s^n)=-\log U_1^k(s^n)-\log U_2^k(s^n)-\log U_3^k(s^n).$$

Hence, with $k=3$,

$$\begin{aligned}\bar R_3(s^n)=-\big(&\log p(s_1|x^n)+\log p(s_1^2|x^n)+\\ &\log p(s_1^3|x^n)+\log p(s_2^4|x^n)+\cdots+\log p(s_{n-3}^{n-1}|x^n)+\log p(s_{n-2}^n|x^n)\\ &+\log p(s_{n-1}^n|x^n)+\log p(s_n|x^n)\big).\end{aligned}$$

Clearly when k is small in comparison with n, the modification is minor. Also note that for $k=1$, the newly introduced risk equals to $\bar R_1$, hence the notation is correct. But the next theorem shows that being so defined, the $\bar R_k$-risk has a nice interpretation. Let

$$v(k)=\arg\min_{s^n} R_k(s^n|x^n).$$

Theorem 7.3.1 *For every $x^n \in \mathcal{X}^n$, for every $s^n \in \mathcal{Y}^n$ and for every $k = 2, \ldots, n$, it holds*

$$\bar{R}_k(s^n|x^n) = (k-1)\bar{R}_\infty(s^n|x^n) + \bar{R}_1(s^n|x^n).$$

Moreover, $v(k)$ is admissible and

$$\bar{R}_\infty(v(k)|x^n) \leq \bar{R}_\infty(v(k-1)|x^n), \quad \bar{R}_1(v(k)|x^n) \geq \bar{R}_1(v(k-1)|x^n).$$

For proof, see [30]. The main statement of the above-stated theorem is the first one saying that our k-block alignment belongs to our general family of alignments (7.3.21) with the constants $C_1 = 1$ and $C_2 = k-1$. Hence, for any k, the optimal path $v(k)$ can be found via recursion (7.3.24).
We already know that $C_2 > 0$ guarantees the admissibility of the solution, so the second statement of the theorem, is an immediate consequence of the first one.
The last two statements guarantee that if k increases, then the likelihood of $v(k)$ increases and

$$\sum_t \log p_t(v_t(k)|x^n)$$

decreases. This is, indeed, what one expects from a k-block alignment.

Example. Consider the following four-state MC transition matrix

$$\frac{1}{8}\begin{pmatrix} 0 & 4 & 2 & 2 \\ 4 & 1 & 1 & 2 \\ 2 & 1 & 1 & 4 \\ 2 & 2 & 4 & 0 \end{pmatrix}$$

Suppose observations x_1, x_2, x_3, x_4 and the emission densities f_s $s = 1,2,3,4$ are such that

$$f_s(x_1) = f_s(x_4) = \begin{cases} 1, & \text{if } s = 2; \\ 0, & \text{if } s \neq 2. \end{cases}, \quad f_s(x_3) = f_s(x_2) = \begin{cases} A > 1, & \text{if } s = 1; \\ 1, & \text{if } s \neq 1. \end{cases}$$

Hence every admissible path begins and ends with 2.

Exercise: Show that

- Viterbi alignments are $(2,1,2,2), (2,2,1,2), (2,1,4,2), (2,4,1,2)$;
- PMAP-alignment is $(2,1,1,2)$ – inadmissible;
- Rabiner's 2-block alignment is $(2,1,1,2)$ – inadmissible;
- $v(2)$ alignments are $(2,1,4,2)$ and $(2,4,1,2)$, both admissible.

References: About HMM's in general, read [26, 27, 25, 28], about theory of segmentation read [29, 30]

Bibliography

[1] A probabilistic theory of pattern recognition
L. Devroye, L. Györfi, G. Lugosi.
Springer, 1996.

[2] Pattern classification and learning theory.
G. Lugosi
In: Principles of Nonparametric Learning Springer, Wien, New York, pp. 1–56, Springer 2002.
http://www.econ.upf.edu/ lugosi/surveys.html.

[3] Theory of classification: a survey of some recent advances
S. Boucheron, O. Bousquet, G. Lugosi
ESAIM: Probability and Statistics, 9:323–375, 2005.
http://www.econ.upf.edu/ lugosi/surveys.html.

[4] Introduction to statistical learning theory
S. Boucheron, O. Bousquet, G. Lugosi
http://www.econ.upf.edu/ lugosi/surveys.html.

[5] Statistical learning theory
V. Vapnik
Wiley, 1998

[6] Pattern classification (2nd edition)
R. Duda, P. Hart, D. Stork
Wiley, 2000.

[7] The elements of statistical learning
T. Hastie, R. Tibshirani, T. Friedman.
Springer, 2001.

[8] Statistical pattern recognition
A. Webb
Wiley, 2002

[9] Introduction to machine learning
E. Alpaydin
MIT, 2004.

[10] An introduction to support vector machines and other kernel-based learning methods
N. Cristianini, J. Shawe-Taylor.
Cambridge University Press, 2003.

[11] Kernel methods for pattern analysis
J. Shawe-Taylor, N. Cristianini.
Cambridge University Press, 2004.

[12] Learning with kernels: support vector machines, regularization, optimization, and beyond
B. Schölkopf ; A. J. Smola.
MIT Press, 2002.

[13] Kernel Fisher discriminants
S. Mika
PhD 2002
edocs.tu-berlin.de/diss/2002/mika-sebastian.pdf

[14] LARS software for R and Splus
B.Efron ja T.Hastie
www-stat.stanford.edu/∼ hastie/Papers/LARS.

[15] Large margin classifiers: convex loss, low noise and convergence rates
P. Bartlett, M. Jordan, J. McAuliffe
In: Advances of Neural Information Processing Systems, 16, 2004

[16] Convexity, Classification, and Risk bounds
P. Bartlett, M. Jordan, J. McAuliffe
Journal of the American Statistical Association, 101(473):138-156, 2006
http://www.stat.berkeley.edu/ bartlett/papers/bjm-ccrb-05.pdf

[17] On the existence of weak lerners and applications to boosting.
S. Mannor, R. Meir
Machine Learning, 48(1-2):219-251, 2002.

[18] An Introduction to Boosting and Leveraging
R. Meir, G. Rätsch
Lecture Notes In Artificial Intelligence, 2003.
http://www.face-rec.org/algorithms/Boosting-Ensemble/8574x0tm63nvjbem.pdf

[19] Empirical margin distributions and bounding the generalization error of combined classifiers.
V. Kolchinskii, D. Panchenko
Ann, Statis., 30(1), 2002

[20] Rademacher and Gaussian Complexities: Risk Bounds and Structural Results
P. Bartlett, S. Mendelson
Journal of Machine Learning Research, 3:463-487, 2002
http://www.ai.mit.edu/projects/jmlr/papers/volume3/bartlett02a/bartlett02a.pdf

[21] AdaBoost is Consistent
P. Bartlett and M. Traskin
Journal of Machine Learning Research, 8:2347-2368, 2007
http://jmlr.csail.mit.edu/papers/volume8/bartlett07b/bartlett07b.pdf

[22] The Boosting Approach to Machine Learning: An Overview.
Robert E. Schapire
2002.
http://tjure.sfs.uni-tuebingen.de/files/Kursmaterialien/Kuebler/ML-ss05/schapire.pdf

[23] A Short Introduction to Boosting.
Y. Freund and R. E. Schapire
In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 1401–1406, 1999.
http://www.yorku.ca/gisweb/eats4400/boost.pdf

[24] Boosting Algorithms: Regularization, Prediction and Model Fitting.
P. Bühlmann, and T. Hothorn
Statistical Science 22, no. 4: 477-505, 2007.
http://projecteuclid.org/euclid.ss/1207580163.

[25] A tutorial on Hidden Markov Models and selected applications in speech recognition
L. Rabiner
Proceedings of the IEEE 77 (2): 257Ũ286, 1989
http://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/tutorial

[26] Inference in hidden Markov models
O. Cappe, E. Moulines, T. Ryden
Springer, 2005

[27] Hidden Markov models for bioinformatics
T. koski
Computational Biology Series, 2.
Kluwer Academic, 2001

[28] Y. Ephraim, N. Merhav
Hidden Markov processes
Special issue on Shannon theory: perspective, trends, and applications.
IEEE Trans. Inform. Theory 48(6), 2002

[29] J. Lember, K. Kuljus, A. Koloydenko
Theory of segmentation
Hidden Markov models: theory and applications
InTech, 2011
http://www.intechopen.com/books/hidden-markov-models-theory-and-applications

[30] J. Lember, A. Koloydenko
A generalized risk-based approach to segmentation based on hidden Markov models
arXiv:1007.3622v1, 2010