MATHEMATICS FOR ML

12. CLASSIFICATION WITH SUPPORT VECTOR MACHINES

Introduction

- In Chapter 9, we considered a prediction problem with continuous-valued outputs.
- In many situations, we want our machine learning algorithm to predict one of a number of (discrete) outcomes.
  - An email client sorts mail: personal or junk.
  - A telescope identifies: galaxy, star, or planet.
- In this chapter, we consider predictors （预测器） that output binary values, i.e., there are only two possible outcomes. This machine learning task is called binary classification （二分类）.

Introduction

- For binary classification, the set of possible values that the label/output can attain is binary, and for this chapter we denote them by $\{+1, -1\}$. In other words, we consider predictors of the form

$$f : \mathbb{R}^D \to \{+1, -1\}.$$

  - Each example (示例, data point) $\boldsymbol{x}_n$ is represented as a feature vector of D real numbers;
  - The labels are often referred to as the positive and negative classes（正类和负类）, respectively.
  - Be careful not to infer intuitive attributes of positiveness of the +1 class. For example, in a cancer detection task, a patient with cancer is often labeled +1.
  - In principle, any two distinct values can be used, e.g., {True, False}, {0,1} or {red, blue}.

Introduction

- As in regression, we have a supervised learning task, where we have a set of examples $\boldsymbol{x}_n \in \mathbb{R}^D$ along with their corresponding (binary) labels $y_n \in \{+1, -1\}$.
- Given a training data set consisting of example-label pairs $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$, we would like to estimate parameters of the model that will give the smallest classification error.
- We present an approach known as the support vector machine （SVM, 支持向量机）, which solves the binary classification task.
- Similar to Chapter 9, we consider a linear model, and hide away the nonlinearity in a transformation ϕ of the examples.

Introduction

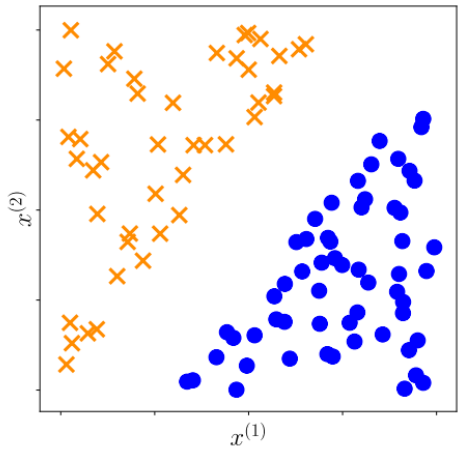
- The SVM provides state-of-the-art results in many applications, with sound theoretical guarantees.
- There are two main reasons why we chose to illustrate binary classification using SVMs.
  - The SVM allows for a geometric way (inner product, projection, Ch. 3) to think about supervised machine learning;
  - The optimization problem for SVM does not admit an analytic solution and we need to sort to optimization tools.

Introduction

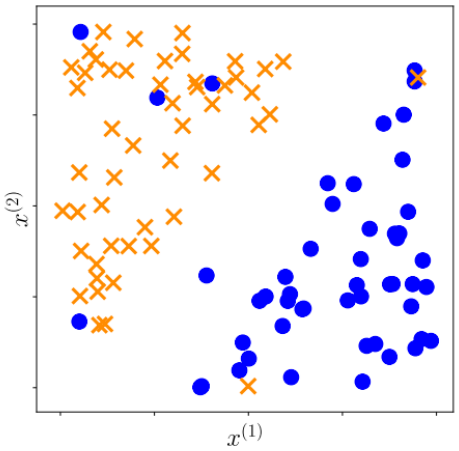
- The SVM view of machine learning is subtly different from the maximum likelihood view of Chapter 9.
  - The maximum likelihood view proposes a model based on a probabilistic view of the data distribution, from which an optimization problem is derived.
  - In contrast, the SVM view starts by designing a particular function that is to be optimized during training, based on geometric intuitions (PCA?).
  - Designing a particular loss function; Empirical risk minimization.

Introduction

- The binary classification data can be separated by a hyperplane (Fig. 12.1);
- Find a linear separator (hyperplane) of the two classes;



(a) Linearly separable data, with a large margin



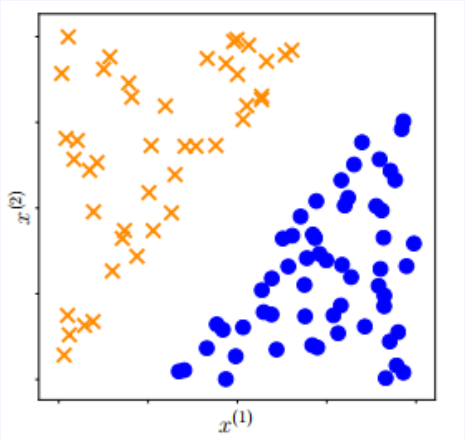
(b) Non-linearly separable data

Introduction

- Introduce the idea of the margin and then extend linear separators to allow for examples to fall on the “wrong” side, incurring a classification error.
- Present two equivalent ways of formalizing the SVM:
  - The geometric view (Section 12.2.4);
  - The loss function view (Section 12.2.5).
- Derive the dual version of the SVM using Lagrange multipliers. The dual SVM allows us to observe a third way of formalizing the SVM: in terms of the convex hulls of the examples of each class (Section 12.3.2).
- We conclude by briefly describing kernels and how to numerically solve the nonlinear kernel-SVM optimization problem.

12.1 Separating Hyperplanes

- The main idea behind many classification algorithms is to represent data in $\mathbb{R}^D$ and then partition this space, ideally in a way that examples with the same label (and no other examples) are in the same partition.
- In the case of binary classification, the space would be divided into two parts corresponding to the positive and negative classes, respectively.
- We consider a particularly convenient partition, which is to (linearly) split the space into two halves using a hyperplane.



12.1 Separating Hyperplanes

- Let example $\boldsymbol{x} \in \mathbb{R}^D$ be an element of the data space. Consider a function

$$f : \mathbb{R}^D \to \mathbb{R}$$
$$\boldsymbol{x} \mapsto \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b,$$

  parametrized by $\boldsymbol{w} \in \mathbb{R}^D$ and $b \in \mathbb{R}$.
- Define the hyperplane that separates the two classes in our binary classification problem as

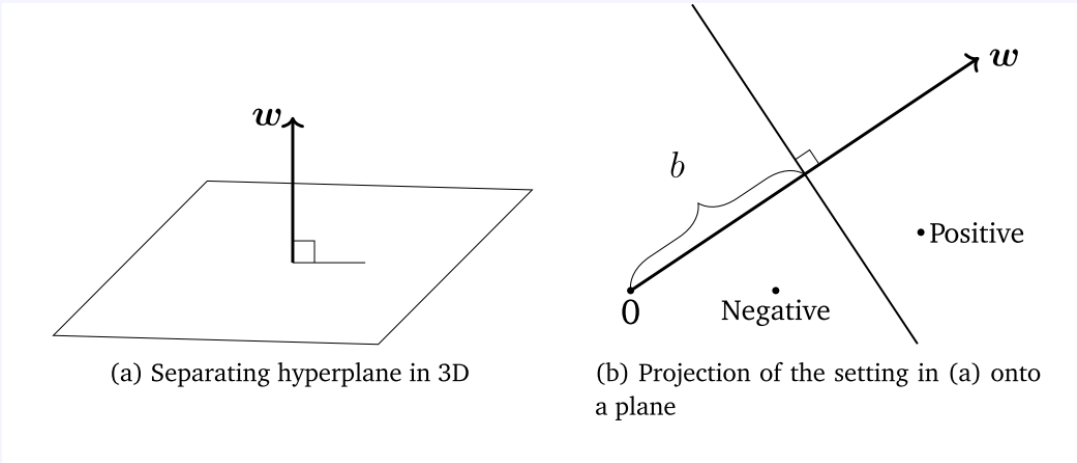
$$H = \left\{ \boldsymbol{x} \in \mathbb{R}^D : f(\boldsymbol{x}) = 0 \right\}.$$

- $\boldsymbol{w}$ is a normal vector to the hyperplane H: $\forall \boldsymbol{x}', \boldsymbol{x}'' \in H$,

$$0 = f\left(\boldsymbol{x}'\right) - f\left(\boldsymbol{x}''\right) = \left\langle \boldsymbol{w}, \boldsymbol{x}' \right\rangle + b - \left(\left\langle \boldsymbol{w}, \boldsymbol{x}'' \right\rangle + b \right) = \left\langle \boldsymbol{w}, \boldsymbol{x}' - \boldsymbol{x}'' \right\rangle.$$

12.1 Separating Hyperplanes

- An illustration of the hyperplane is shown in Figure 12.2, where the vector $\boldsymbol{w}$ is a vector normal to the hyperplane and b the intercept.



(a) Separating hyperplane in 3D

(b) Projection of the setting in (a) onto a plane

Figure. 12.2.

- The direction of the hyperplane H is also defined. ⇒The positive and negative side（正侧和负侧）.
- To classify a test example $\boldsymbol{x}_{\text{test}}$, we calculate the value of the function $f\left(\boldsymbol{x}_{\text{test}}\right)$ and classify the example as $+1$ if $f\left(\boldsymbol{x}_{\text{test}}\right) \geqslant 0$ and -1 otherwise.

12.1 Separating Hyperplanes

- When training the classifier（分类器）, we want to ensure that
  - The examples with positive labels are on the positive side of the hyperplane, i.e.,

$$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \geqslant 0 \quad \text{when} \quad y_n = +1 \tag{12.5}$$

  - The examples with negative labels are on the negative side, i.e.,

$$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b < 0 \quad \text{when} \quad y_n = -1. \tag{12.6}$$

- These two conditions are often presented in a (equivalent) single equation

$$y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) \geqslant 0. \tag{12.7}$$

12.2 Primal Support Vector Machine

- Based on the concept of distances from points to a hyperplane, we now are in a position to discuss the support vector machine.
- For a dataset $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$ that is linearly separable, we have infinitely many candidate hyperplanes, and therefore classifiers, that solve our classification problem without any (training) errors.

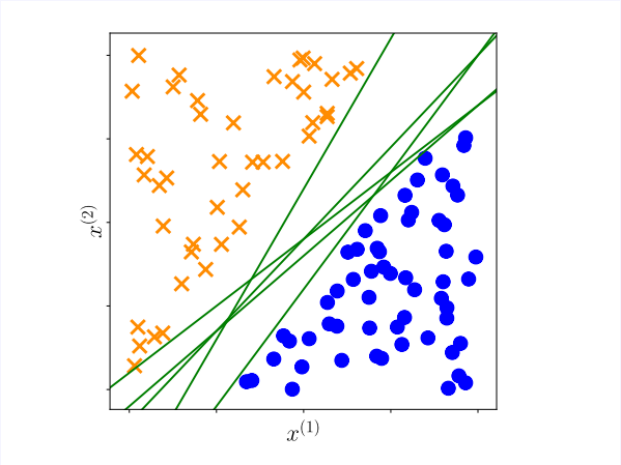
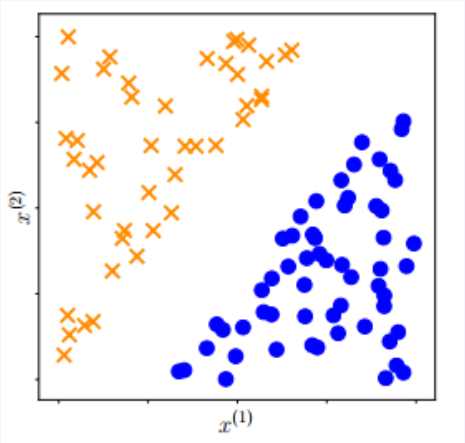


Figure. 12.3.

12.2 Primal Support Vector Machine

- To find a unique (the best) solution, one idea is to choose the separating hyperplane that maximizes the margin （最大化间隔） between the positive and negative examples. In other words, we want the positive and negative examples to be separated by a large margin.



- In the following, we compute the distance between an example and a hyperplane to derive the margin. Recall that the closest point on the hyperplane to a given point (example $\boldsymbol{x}_n$) is obtained by the orthogonal projection (Section 3.8).

12.2 Primal Support Vector Machine

12.2.1 Concept of the Margin

- The concept of the margin is intuitively simple: It is the distance of the separating hyperplane to the closest examples in the dataset, assuming that the dataset is linearly separable.
- However, when trying to formalize this distance, there is a technical wrinkle that may be confusing. The technical wrinkle is that we need to define a scale at which to measure the distance.
- A potential scale is to consider the scale of the data, i.e., the raw values of $\boldsymbol{x}_n$. There are problems with this, as we could change the units of measurement of $\boldsymbol{x}_n$ and change the values in $\boldsymbol{x}_n$, and, hence, change the distance to the hyperplane. As we will see shortly, we define the scale based on the equation of the hyperplane H itself.

12.2 Primal Support Vector Machine

- Consider a hyperplane $\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$, and an example $\boldsymbol{x}_a$ as illustrated in Figure 12.4.

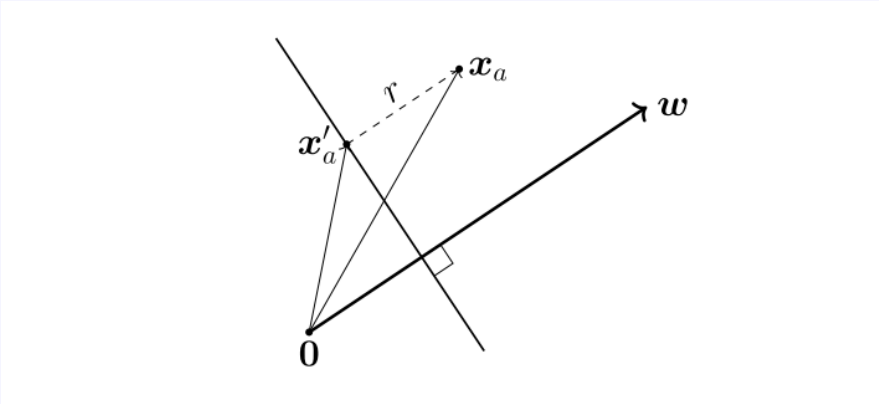


Figure. 12.4

- Without loss of generality, we can consider the example $\boldsymbol{x}_a$ to be on the positive side of the hyperplane, i.e., $\langle \boldsymbol{w}, \boldsymbol{x_a} \rangle + b > 0$.
- We would like to compute the distance $r > 0$ of x_a from the hyperplane. We do so by considering the orthogonal projection (Section 3.8) of x_a onto the hyperplane, which we denote by $\boldsymbol{x}'_a$.

12.2 Primal Support Vector Machine

- For convenience, we choose to use a vector of unit length, $\|\boldsymbol{w}\| = 1$.
- Let r be the distance of $\boldsymbol{x}_a$ from the hyperplane. Then

$$\boldsymbol{x}_a = \boldsymbol{x}'_a + r\boldsymbol{w}, \tag{12.8}$$

and

$$\langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b = \langle \boldsymbol{w}, \boldsymbol{x}'_a \rangle + b + r = r \text{（距离！）}.$$

- If we choose $\boldsymbol{x}_a$ to be the point closest to the hyperplane, this distance r is the margin.
- We would like the positive examples to be further than r from the hyperplane, and the negative examples to be further than distance r (in the negative direction) from the hyperplane.

$$y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) \geqslant r. \tag{12.9}$$

12.2 Primal Support Vector Machine

- Collecting the three requirements into a single constrained optimization problem, we obtain

$$\begin{aligned} &\max_{\boldsymbol{w},b,r} \underbrace{r}_{\text{margin}} \\ &\text{subject to } \underbrace{y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) \geqslant r}_{\text{data fiting}}, \quad \underbrace{\|\boldsymbol{w}\| = 1}_{\text{normalization}}, \quad r > 0, \end{aligned} \tag{12.10}$$

  which says that we want to maximize the margin r while ensuring that the data lies on the correct side of the hyperplane.

12.2 Primal Support Vector Machine

- **Remark** The concept of the margin turns out to be highly pervasive in machine learning. It was used by Vladimir Vapnik and Alexey Chervonenkis to show that when the margin is large, the ”complexity” of the function class is low, and hence learning is possible (Vapnik, 2000). It turns out that the concept is useful for various different approaches for theoretically analyzing generalization error (Steinwart and Christmann, 2008; Shalev-Shwartz and Ben-David, 2014).

12.2 Primal Support Vector Machine

12.2.2 Traditional Derivation of the Margin

- In the previous section, we derived (12.10) by making the observation that we are only interested in the direction of w and not its length, leading to the assumption that $\|\boldsymbol{w}\| = 1$.
- In this section, we derive the margin maximization problem by making a different assumption. Instead of choosing that the parameter vector is normalized, we choose a scale for the data. We choose this scale such that the value of the predictor $\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$ is 1 at the closest example.
- Let us also denote the example in the dataset that is closest to the hyperplane by x_a.

12.2 Primal Support Vector Machine

- $\boldsymbol{x}_a$ lies on the margin H_+, i.e., $\langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b = 1$.

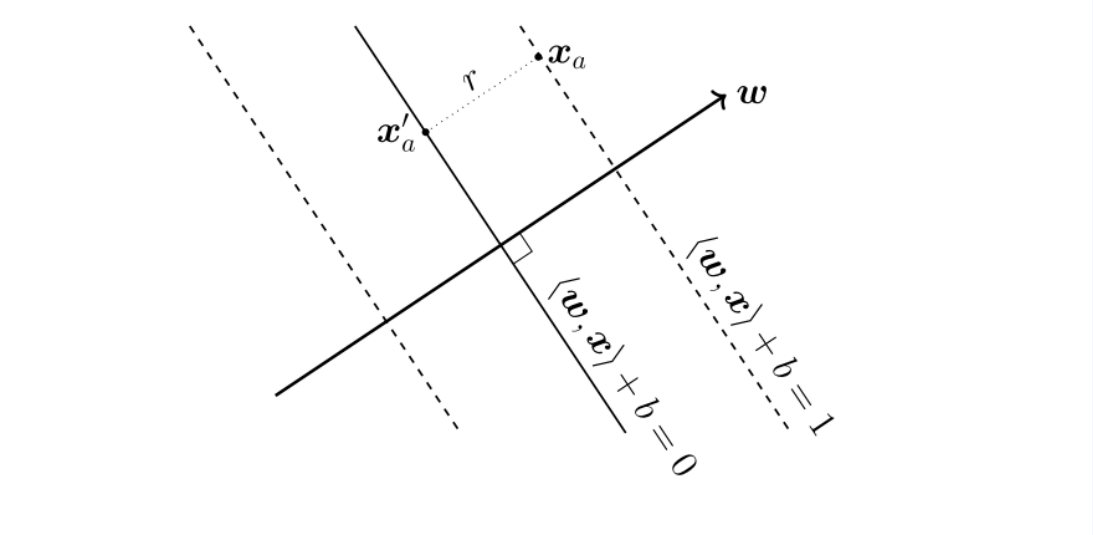


Figure. 12.5: Derivation of the margin: $r = \frac{1}{\|\boldsymbol{w}\|}$.

- Let x'_a be the orthogonal projection of x_a onto the hyperplane H, then

$$\boldsymbol{x}_a = \boldsymbol{x}'_a + r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}, \quad \langle \boldsymbol{w}, \boldsymbol{x}'_a \rangle + b = 0.$$

12.2 Primal Support Vector Machine

- We get

$$0 = \left\langle \boldsymbol{w}, \boldsymbol{x}_a - r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|} \right\rangle + b = \langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b - r\frac{\langle \boldsymbol{w}, \boldsymbol{w} \rangle}{\|\boldsymbol{w}\|} = 1 - r\|w\|,$$

and hence

$$r = \frac{1}{\|\boldsymbol{w}\|}.$$

12.2 Primal Support Vector Machine

- Similar to the argument to obtain (12.9), we want the positive and negative examples to be at least 1 away from the hyperplane, which yields the condition

$$y_n \left(\langle \boldsymbol{w}, x_n \rangle + b \right) \geqslant 1$$

- Combining the margin maximization with the fact that examples need to be on the correct side of the hyperplane (based on their labels) gives us

$$\max_{\boldsymbol{w}, b} \frac{1}{\|\boldsymbol{w}\|} \tag{12.16}$$

$$\text{subject to } y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) \geqslant 1 \quad \text{for all } n = 1, \ldots, N \tag{12.17}$$

12.2 Primal Support Vector Machine

- Instead of maximizing the reciprocal of the norm as in (12.16), we often minimize the squared norm. We also often include a constant $\frac{1}{2}$ that does not affect the optimal $\boldsymbol{w}, b$ but yields a tidier form when we compute the gradient. Then, our objective becomes

$$\begin{aligned} \min_{\boldsymbol{w},b} \quad & \frac{1}{2}\|w\|^2 \\ \text{subject to} \quad & y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1, \ n = 1, \ldots, N \end{aligned} \tag{12.18}$$

- (12.18) is known as the hard margin SVM. The reason for the expression "hard" is because the formulation does not allow for any violations of the margin condition. We will see in Section 12.2.4 that this "hard" condition can be relaxed to accommodate violations if the data is not linearly separable.

12.2 Primal Support Vector Machine

12.2.3 Why We Can Set the Margin to 1

- In Section 12.2.1, we argued that we would like to maximize some value r, which represents the distance of the closest example to the hyperplane.
- In Section 12.2.2, we scaled the data such that the closest example is of distance 1 to the hyperplane.
- In this section, we relate the two derivations, and show that they are equivalent.

12.2 Primal Support Vector Machine

Theorem 12.1. Maximizing the margin r, where we consider normalized weights as in (12.10),

$$\begin{aligned} \max_{\boldsymbol{w},b,r} \quad & r \\ \text{subject to} \quad & y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) \geqslant r, \|\boldsymbol{w}\| = 1, r > 0, \end{aligned} \tag{12.20}$$

is equivalent to scaling the data, such that the margin is unity:

$$\begin{aligned} \min_{\boldsymbol{w},b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{subject to} \quad & y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) \geqslant 1. \end{aligned} \tag{12.21}$$

12.2 Primal Support Vector Machine

Proof. Consider (12.20). Since the square is a strictly monotonic transformation for non-negative arguments, the maximum stays the same if we consider r^2 in the objective. Since $\|\boldsymbol{w}\| = 1$ we can reparametrize the equation with a new weight vector $\boldsymbol{w}'$ that is not normalized by explicitly using $\frac{\boldsymbol{w}'}{\|\boldsymbol{w}'\|}$. We obtain

$$\begin{aligned} &\max_{\boldsymbol{w}', b, r} && r^2 \\ &\text{subject to} && y_n \left(\left\langle \frac{\boldsymbol{w}'}{\|\boldsymbol{w}'\|}, \boldsymbol{x}_n \right\rangle + b \right) \geqslant r, \quad r > 0 \end{aligned} \tag{12.22}$$

Equation (12.22) explicitly states that the distance r is positive. Therefore, we can divide the first constraint by r, which yields

12.2 Primal Support Vector Machine

$$\begin{aligned} \max_{\boldsymbol{w}',b,r} \quad & r^2 \\ \text{subject to} \quad & y_n \left(\left\langle \underbrace{\frac{\boldsymbol{w}'}{\|\boldsymbol{w}'\| r}}_{\boldsymbol{w}''}, \boldsymbol{x}_n \right\rangle + \underbrace{\frac{b}{r}}_{b''} \right) \geqslant 1, \quad r > 0 \end{aligned} \tag{12.23}$$

renaming the parameters to $\boldsymbol{w}''$ and b''. Since $\boldsymbol{w}'' = \frac{\boldsymbol{w}'}{\|\boldsymbol{w}'\| r}$, rearranging for r gives

$$\|\boldsymbol{w}''\| = \left\| \frac{\boldsymbol{w}'}{\|\boldsymbol{w}'\| r} \right\| = \frac{1}{r} \cdot \left\| \frac{\boldsymbol{w}'}{\|\boldsymbol{w}'\|} \right\| = \frac{1}{r}$$

12.2 Primal Support Vector Machine

By substituting this result into (12.23), we obtain

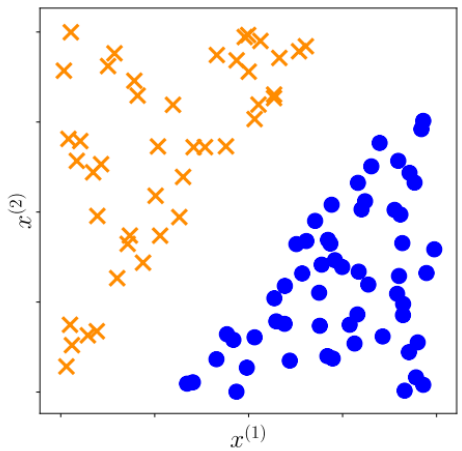
$$\begin{aligned} \max_{\boldsymbol{w}'', b''} \quad & \frac{1}{\|\boldsymbol{w}''\|^2} \\ \text{subject to} \quad & y_n \left(\langle \boldsymbol{w}'', \boldsymbol{x}_n \rangle + b'' \right) \geqslant 1 \end{aligned}$$

The final step is to observe that maximizing $\frac{1}{\|\boldsymbol{w}''\|^2}$ yields the same solution as minimizing $\frac{1}{2} \|\boldsymbol{w}''\|^2$, which concludes the proof of Theorem 12.1.

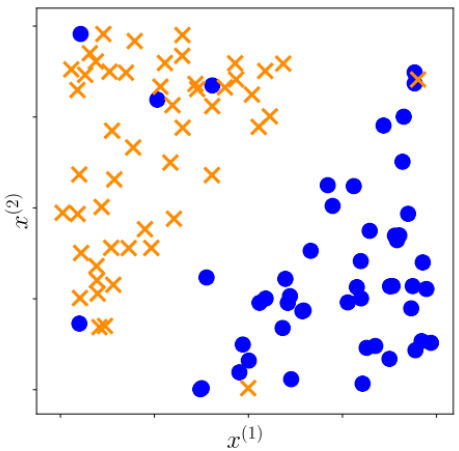
12.2 Primal Support Vector Machine

12.2.4 Soft Margin SVM: Geometric View

- In the case where data is not linearly separable, we may wish to allow some examples to fall within the margin region, or even to be on the wrong side of the hyperplane as illustrated in Figure 12.6.



(a) Linearly separable data, with a large margin



(b) Non-linearly separable data

- The model that allows for some classification errors is called the soft margin SVM.

12.2 Primal Support Vector Machine

The key geometric idea is to introduce a slack variable ξ_n corresponding to each example-label pair $(\boldsymbol{x}_n, y_n)$ that allows a particular example to be within the margin or even on the wrong side of the hyperplane (refer to Figure 12.7).

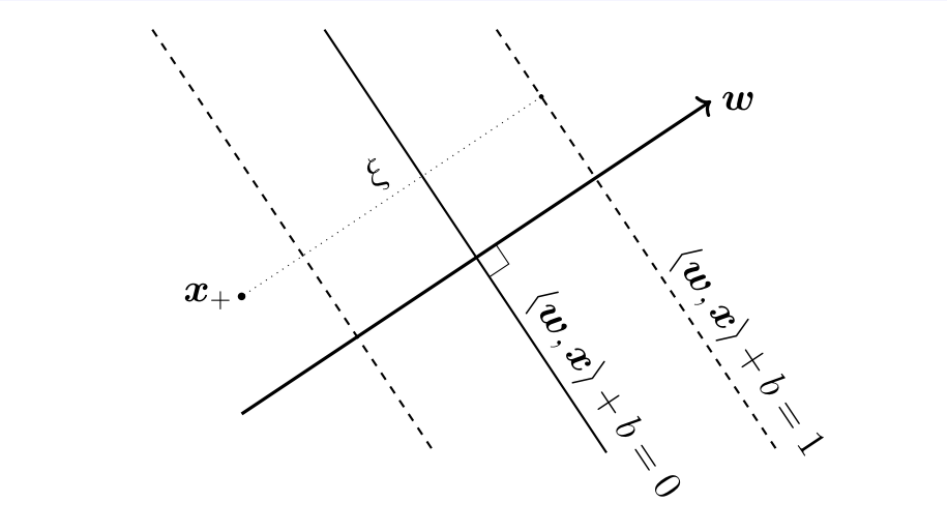


Figure. 12.7: Soft margin SVM allows examples to be within the margin or on the wrong side of the hyperplane.

12.2 Primal Support Vector Machine

- We subtract the value of ξ_n from the margin, constraining ξ_n to be non-negative. To encourage correct classification of the samples, we add ξ_n to the objective

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n \tag{12.26a}$$

$$\text{subject to} \quad y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) \geqslant 1 - \xi_n \tag{12.26b}$$

$$\xi_n \geqslant 0 \tag{12.26c}$$

  for $n = 1, \ldots, N$.
- (12.16) is called the soft margin SVM.
- The parameter $C > 0$ trades off the size of the margin and the total amount of slack that we have. This parameter is called the regularization parameter. The margin term $\|\boldsymbol{w}\|^2$ is called the regularizer (b is not regularized).

12.2 Primal Support Vector Machine

12.2.5 Soft Margin SVM: Loss Function View

- Let us consider a different approach for deriving the SVM, following the principle of empirical risk minimization (Section 8.2). For the SVM, we choose hyperplanes as the hypothesis class, that is

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b.$$

- We will see that the margin corresponds to the regularization term.
- What is the loss function?
- For binary classification problems, the output of the predictor is one of two labels $\{+1, -1\}$. We define the loss to be zero if they match, and one if they do not match. This is denoted by $\mathbf{1}\left(f\left(\boldsymbol{x}_n\right) \neq y_n\right)$ and is called the zero-one loss.

12.2 Primal Support Vector Machine

- The zero-one loss results in a combinatorial optimization problem, which is in general more challenging to solve.
- What is the loss function corresponding to the SVM?
- Use the hinge loss: $\ell(t) = \max\{0, 1 - t\}$ where $t = yf(\boldsymbol{x}) = y(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)$.
  - If $\boldsymbol{x}$ is on the correct side of the hyperplane, and further than distance 1 ($t \geqslant 1$), the hinge loss is zero.
  - If $\boldsymbol{x}$ is on the correct side but too close to the hyperplane $(0 < t < 1)$, the hinge loss returns a positive value.
  - When the example is on the wrong side of the hyperplane $(t < 0)$, the hinge loss returns an even larger value, which increases linearly.
- In other words, we pay a penalty once we are closer than the margin to the hyperplane, even if the prediction is correct, and the penalty increases linearly.

12.2 Primal Support Vector Machine

- An alternative way to express the hinge loss is by considering it as two linear pieces

$$\ell(t) = \begin{cases} 0 & \text{if} \quad t \geqslant 1 \\ 1 - t & \text{if} \quad t < 1 \end{cases} \tag{12.29}$$

as illustrated in Figure 12.8.

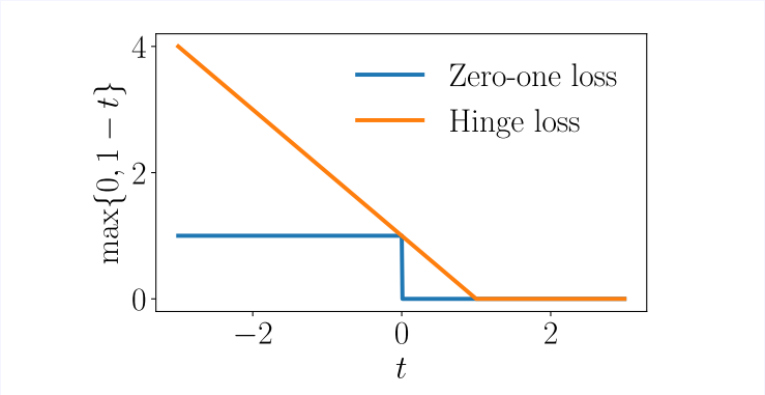


Figure. 12.8: The hinge loss is a convex upper bound of zero-one loss.

12.2 Primal Support Vector Machine

- The loss corresponding to the hard margin SVM is defined as

$$\ell(t) = \begin{cases} 0 & \text{if} \quad t \geqslant 1 \\ \infty & \text{if} \quad t < 1 \end{cases} \tag{12.30}$$

  This loss can be interpreted as never allowing any examples inside the margin.

- For a given training set $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$, we seek to minimize the total loss, while regularizing the objective with ℓ_2-regularization (see Section 8.2.3).

- Using the hinge loss (12.28) gives us the unconstrained optimization problem

$$\min_{\boldsymbol{w},b} \underbrace{\frac{1}{2}\|\boldsymbol{w}\|^2}_{\text{regularizer}} + \underbrace{C\sum_{n=1}^{N} \max\{0, 1 - y_n(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b)\}}_{\text{error term}}. \tag{12.31}$$

12.2 Primal Support Vector Machine

- **Proposition.**

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N} \max\left\{0, 1 - y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right)\right\}. \tag{12.31}$$

is equivalent to

$$\begin{aligned} \min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n \\ \text{subject to} \quad & y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) \geqslant 1 - \xi_n \\ & \xi_n \geqslant 0 \end{aligned} \tag{12.26}$$

12.3 Dual Support Vector Machine

- The description of the SVM in the previous sections, in terms of the variables $\boldsymbol{w} \in \mathbb{R}^D$ and b, is known as the primal SVM. The number of parameters (the dimension of $\boldsymbol{w}$) of the optimization problem grows linearly with the number of features.
- In the following, we consider an equivalent optimization problem (the so-called dual view), which is independent of the number of features.
- Instead, the number of parameters increases with the number of examples in the training set.
  - This is useful for problems where we have more features than the number of examples in the training dataset.
  - It easily allows kernels to be applied, as we shall see at the end of this chapter.

12.3 Dual Support Vector Machine

12.3.1 Convex Duality via Lagrange Multipliers

- The primal SVM

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n \tag{12.26a}$$

$$\text{subject to} \quad y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) \geqslant 1 - \xi_n \tag{12.26b}$$

$$\xi_n \geqslant 0 \tag{12.26c}$$

- Lagrangian

$$\begin{aligned}\mathfrak{L}(\boldsymbol{w}, b, \xi, \alpha, \gamma) =& \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n \\ &- \sum_{n=1}^{N}\alpha_n\left(y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) - 1 + \xi_n\right) - \sum_{n=1}^{N}\gamma_n\xi_n.\end{aligned} \tag{12.34}$$

- We call the variables w, b, and ξ the primal variables.
- $\alpha_n \geqslant 0$ and $\gamma_n \geqslant 0$ are the Lagrange multipliers, or dual variables.

12.3 Dual Support Vector Machine

- By differentiating the Lagrangian (12.34) with respect to the three primal variables $\boldsymbol{w}, b$, and ξ respectively, we obtain

$$\frac{\partial \mathfrak{L}}{\partial \boldsymbol{w}} = \boldsymbol{w}^\top - \sum_{n=1}^{N} \alpha_n y_n \boldsymbol{x}_n^\top, \tag{12.35}$$

$$\frac{\partial \mathfrak{L}}{\partial b} = -\sum_{n=1}^{N} \alpha_n y_n, \tag{12.36}$$

$$\frac{\partial \mathfrak{L}}{\partial \xi_n} = C - \alpha_n - \gamma_n. \tag{12.37}$$

- By setting (12.35) to zero, we find (the representer theorem)

$$\boldsymbol{w} = \sum_{n=1}^{N} \alpha_n y_n \boldsymbol{x}_n, \quad \text{(affine combination, by (12.36)).} \tag{12.38}$$

- $\alpha_n = 0$, do not contribute to the solution $\boldsymbol{w}$ at all. The other examples, where $\alpha_n > 0$, are called support vectors.

12.3 Dual Support Vector Machine

By substituting the expression for $\boldsymbol{w}$ into the Lagrangian (12.34), we obtain the dual

$$\begin{aligned}\mathfrak{D}(\xi,\alpha,\gamma) =& \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle - \sum_{i=1}^{N} y_i \alpha_i \left\langle \sum_{j=1}^{N} y_j \alpha_j \boldsymbol{x}_j, \boldsymbol{x}_i \right\rangle \\ &+ C\sum_{i=1}^{N}\xi_i - b\sum_{i=1}^{N} y_i \alpha_i + \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\alpha_i \xi_i - \sum_{i=1}^{N}\gamma_i \xi_i.\end{aligned} \tag{12.39}$$

Note that there are no longer any terms involving the primal variable $\boldsymbol{w}$. By setting (12.36) to zero, we obtain $\sum_{n=1}^{N} y_n \alpha_n = 0$. Therefore, the term involving b also vanishes. Recall that inner products are symmetric and bilinear (see Section 3.2). Therefore, the first two terms in (12.39) are over the same objects.

12.3 Dual Support Vector Machine

These terms (colored blue) can be simplified, and we obtain the Lagrangian

$$\mathfrak{D}(\xi, \alpha, \gamma) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle + \sum_{i=1}^{N} \alpha_i + \sum_{i=1}^{N} (C - \alpha_i - \gamma_i)\, \xi_i. \quad (12.40)$$

The last term in this equation is a collection of all terms that contain slack variables ξ_i. By setting (12.37) to zero, we see that the last term in (12.40) is also zero. Furthermore, by using the same equation and recalling that the Lagrange multipliers γ_i are non-negative, we conclude that $\alpha_i \leqslant C$.

12.3 Dual Support Vector Machine

We now obtain the dual optimization problem of the SVM, which is expressed exclusively in terms of the Lagrange multipliers α_i. Recall from Lagrangian duality (Definition 7.1) that we maximize the dual problem. This is equivalent to minimizing the negative dual problem, such that we end up with the dual SVM

$$\begin{aligned} \min_{\alpha} \quad & \tfrac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle - \sum_{i=1}^{N} \alpha_i \\ \text{subject to} \quad & \sum_{i=1}^{N} y_i \alpha_i = 0 \\ & 0 \leqslant \alpha_i \leqslant C, \; i = 1, \ldots, N. \Rightarrow \text{“box constraints”} \end{aligned} \tag{12.41}$$

The equality constraint in (12.41) is obtained from setting (12.36) to zero. The inequality constraint $\alpha_i \geqslant 0$ is the condition imposed on Lagrange multipliers of inequality constraints (Section 7.2). The inequality constraint $\alpha_i \leqslant C$ is discussed in the previous paragraph.

12.3 Dual Support Vector Machine

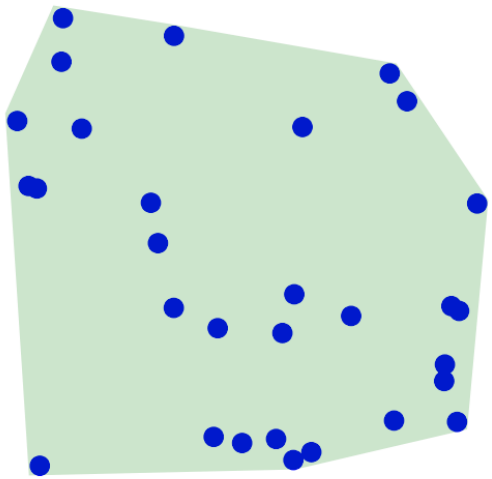
- Once we obtain the dual parameters $\boldsymbol{\alpha}$, we can recover the primal parameters $\boldsymbol{w}$ by using the representer theorem (12.38). Let us call the optimal primal parameter $\boldsymbol{w}^*$. However, there remains the question on how to obtain the parameter b^*. Consider an example $\boldsymbol{x}_n$ that lies exactly on the margin's boundary $(0 < \alpha_n < C)$, i.e., $y_n(\langle \boldsymbol{w}^*, \boldsymbol{x}_n \rangle + b) = 1$. Recall that y_n is either $+1$ or -1. Therefore, the only unknown b can be computed by

$$b^* = y_n - \langle \boldsymbol{w}^*, \boldsymbol{x}_n \rangle . \tag{12.42}$$

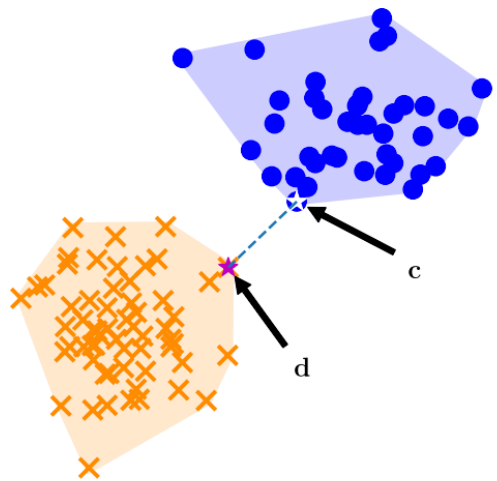
- **Remark.** In principle, there may be no examples that lie exactly on the margin (hard margin SVM?). In this case, we should compute $|y_n - \langle \boldsymbol{w}^*, \boldsymbol{x}_n \rangle|$ for all support vectors and take the median value of this absolute value difference to be the value of b^*. A derivation of this can be found in http://fouryears. eu/2012/06/07/the-svm-bias-term-conspiracy/.

12.3 Dual Support Vector Machine

Dual SVM: Convex Hull View Another approach to obtain the dual SVM is to consider an alternative geometric argument. Consider the set of examples $\boldsymbol{x}_n$ with the same label. We would like to build a convex set that contains all the examples such that it is the smallest possible set. This is called the convex hull and is illustrated in Figure 12.9.



(a) Convex hull.



(b) Convex hulls around positive (blue) and negative (red) examples. The distance between the two convex sets is the length of the difference vector $\boldsymbol{c} - \boldsymbol{d}$.

Figure. 12.9 Convex hulls.

Dual Support Vector Machine

Let us first build some intuition about a convex combination of points. Consider two points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ and corresponding non-negative weights $\alpha_1, \alpha_2 \geqslant 0$ such that $\alpha_1 + \alpha_2 = 1$. The equation $\alpha_1 \boldsymbol{x}_1 + \alpha_2 \boldsymbol{x}_2$ describes each point on a line between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. Consider what happens when we add a third point $\boldsymbol{x}_3$ along with a weight $\alpha_3 \geqslant 0$ such that $\sum_{n=1}^{3} \alpha_n = 1$. The convex combination of these three points $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$ spans a two-dimensional area.

Dual Support Vector Machine

The convex hull of this area is the triangle formed by the edges corresponding to each pair of of points. As we add more points, and the number of points becomes greater than the number of dimensions, some of the points will be inside the convex hull, as we can see in Figure 12.9(a).
In general, building a convex convex hull can be done by introducing non-negative weights $\alpha_n \geqslant 0$ corresponding to each example $\boldsymbol{x}_n$. Then the convex hull can be described as the set

$$\text{conv}(\boldsymbol{X}) = \left\{ \sum_{n=1}^{N} \alpha_n \boldsymbol{x}_n \right\} \quad \text{with} \quad \sum_{n=1}^{N} \alpha_n = 1 \quad \text{and} \quad \alpha_n \geqslant 0, \tag{12.43}$$

for all $n = 1, \ldots, N$.

Dual Support Vector Machine

If the two clouds of points corresponding to the positive and negative classes are separated, then the convex hulls do not overlap. Given the training data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)$, we form two convex hulls, corresponding to the positive and negative classes respectively. We pick a point $\boldsymbol{c}$, which is in the convex hull of the set of positive examples, and is closest to the negative class distribution. Similarly, we pick a point $\boldsymbol{d}$ in the convex hull of the set of negative examples and is closest to the positive class distribution; see Figure 12.9(b).

Dual Support Vector Machine

We define a difference vector between $\boldsymbol{d}$ and $\boldsymbol{c}$ as

$$\boldsymbol{w} := \boldsymbol{c} - \boldsymbol{d} \tag{12.44}$$

Picking the points c and d as in the preceding cases, and requiring them to be closest to each other is equivalent to minimizing the length/norm of $\boldsymbol{w}$, so that we end up with the corresponding optimization problem

$$\arg\min_{\boldsymbol{w}} \|\boldsymbol{w}\| = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{12.45}$$

Since c must be in the positive convex hull, it can be expressed as a convex combination of the positive examples, i.e., for non-negative coefficients α_n^+

$$\boldsymbol{c} = \sum_{n:y_n=+1} \alpha_n^+ \boldsymbol{x}_n \tag{12.46}$$

In (12.46), we use the notation $n : y_n = +1$ to indicate the set of indices n for which $y_n = +1$. Similarly, for the examples with negative labels, we obtain

$$\boldsymbol{d} = \sum_{n:y_n=-1} \alpha_n^- \boldsymbol{x}_n \tag{12.47}$$

Dual Support Vector Machine

By substituting (12.44), (12.46), and (12.47) into (12.45), we obtain the objective

$$\min_{\alpha} \frac{1}{2} \left\| \sum_{n:y_n=+1} \alpha_n^+ \boldsymbol{x}_n - \sum_{n:y_n=-1} \alpha_n^- \boldsymbol{x}_n \right\|^2 \tag{12.48}$$

Let $\boldsymbol{\alpha}$ be the set of all coefficients, i.e., the concatenation of $\boldsymbol{\alpha}^+$and $\boldsymbol{\alpha}^-$. Recall that we require that for each convex hull that their coefficients sum to one,

$$\sum_{n:y_n=+1} \alpha_n^+ = 1 \quad \text{and} \quad \sum_{n:y_n=-1} \alpha_n^- = 1 \tag{12.49}$$

This implies the constraint

$$\sum_{n=1}^{N} y_n \alpha_n = 0 \tag{12.50}$$

Dual Support Vector Machine

This result can be seen by multiplying out the individual classes

$$\sum_{n=1}^{N} y_n \alpha_n = \sum_{n:y_n=+1} (+1)\alpha_n^+ + \sum_{n:y_n=-1} (-1)\alpha_n^- \tag{12.51a}$$

$$= \sum_{n:y_n=+1} \alpha_n^+ - \sum_{n:y_n=-1} \alpha_n^- = 1 - 1 = 0. \tag{12.51b}$$

The objective function (12.48) and the constraint (12.50), along with the assumption that $\boldsymbol{\alpha} \geqslant \mathbf{0}$, give us a constrained (convex) optimization problem. This optimization problem can be shown to be the same as that of the dual hard margin SVM (Bennett and Bredensteiner, 2000a).

Dual Support Vector Machine

Remark

To obtain the soft margin dual, we consider the reduced hull. The reduced hull is similar to the convex hull but has an upper bound to the size of the coefficients α. The maximum possible value of the elements of $\boldsymbol{\alpha}$ restricts the size that the convex hull can take. In other words, the bound on $\boldsymbol{\alpha}$ shrinks the convex hull to a smaller volume (Bennett and Bredensteiner, 2000b).

12.4 Kernels

- The dual SVM

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle - \sum_{i=1}^{N} \alpha_i \\ \text{subject to} \quad & \sum_{i=1}^{N} y_i \alpha_i = 0 \\ & 0 \leqslant \alpha_i \leqslant C,\ i = 1, \ldots, N. \end{aligned} \tag{12.41}$$

- Notice that the inner product in the objective occurs only between examples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. There are no inner products between the examples and the parameters. Therefore, if we consider a set of features $\phi(\boldsymbol{x}_i)$ to represent $\boldsymbol{x}_i$, the only change in the dual SVM will be to replace the inner product.
- Predict:

$$f(\boldsymbol{x}) = \langle \boldsymbol{w}^*, \boldsymbol{x} \rangle + b^* = \sum_{n=1}^{N} \alpha_n y_n \langle \boldsymbol{x}_n, \boldsymbol{x} \rangle + b^*.$$

$$b^* = y_n - \langle \boldsymbol{w}^*, \boldsymbol{x}_n \rangle .$$

12.4 Kernels

- This modularity, where the choice of the classification method (the SVM) and the choice of the feature representation $\phi(\boldsymbol{x})$ can be considered separately, provides flexibility for us to explore the two problems independently.
- In this section, we discuss the representation $\phi(\boldsymbol{x})$ and briefly introduce the idea of kernels, but do not go into the technical details.
- Since $\phi(\boldsymbol{x})$ could be a non-linear function, we can use the SVM (which assumes a linear classifier) to construct classifiers that are nonlinear in the examples $\boldsymbol{x}_n$.
- This provides a second avenue, in addition to the soft margin, for users to deal with a dataset that is not linearly separable.
- It turns out that there are many algorithms and statistical methods that have this property that we observed in the dual SVM: the only inner products are those that occur between examples.

12.4 Kernels

- Instead of explicitly defining a non-linear feature map $\phi(\cdot)$ and computing the resulting inner product between examples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, we define a similarity function $k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$ between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$.
- For a certain class of similarity functions, called kernels, the similarity function implicitly defines a non-linear feature map $\phi(\cdot)$.
- Kernels are by definition functions $k : X \times X \to \mathbb{R}$ for which there exists a Hilbert space H and $\phi : X \to H$ a feature map such that

$$k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \left\langle \phi\left(\boldsymbol{x}_i\right), \phi(\boldsymbol{x}_j) \right\rangle_H. \tag{12.52}$$

12.4 Kernels

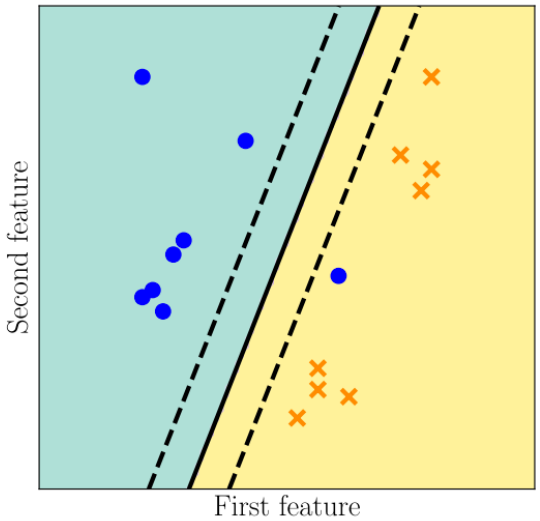
- There is a unique reproducing kernel Hilbert space associated with every kernel k (Aronszajn, 1950; Berlinet and Thomas-Agnan, 2004). In this unique association, $\phi(\boldsymbol{x}) = k(\cdot, \boldsymbol{x})$ is called the canonical feature map.
- The generalization from an inner product to a kernel function (12.52) is known as the kernel trick (Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004), as it hides away the explicit non-linear feature map.
- The matrix $\boldsymbol{K} \in \mathbb{R}^{N \times N}$, resulting from the inner products or the application of $k(\cdot, \cdot)$ to a dataset, is called the Gram matrix, and is often just referred to as the kernel matrix.
- Kernels must be symmetric and positive semidefinite functions so that every kernel matrix $\boldsymbol{K}$ is symmetric and positive semidefinite:

$$\forall \boldsymbol{z} \in \mathbb{R}^N : \boldsymbol{z}^\top \boldsymbol{K} \boldsymbol{z} \geqslant 0. \tag{12.53}$$

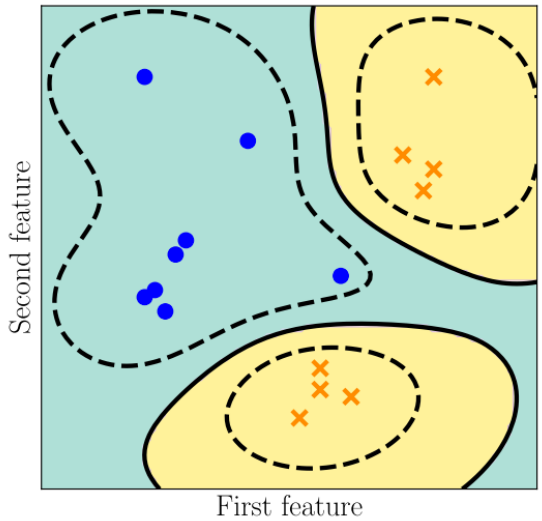
12.4 Kernels

- Some popular examples of kernels for multivariate real-valued data $\boldsymbol{x}_i \in \mathbb{R}^D$ are
  - the linear kernel;
  - the polynomial kernel;
  - the Gaussian radial basis function kernel;
  - the rational quadratic kernel (Schölkopf and Smola, 2002; Rasmussen and Williams, 2006).
- Figure 12.10 illustrates the effect of different kernels on separating hyperplanes on an example dataset.

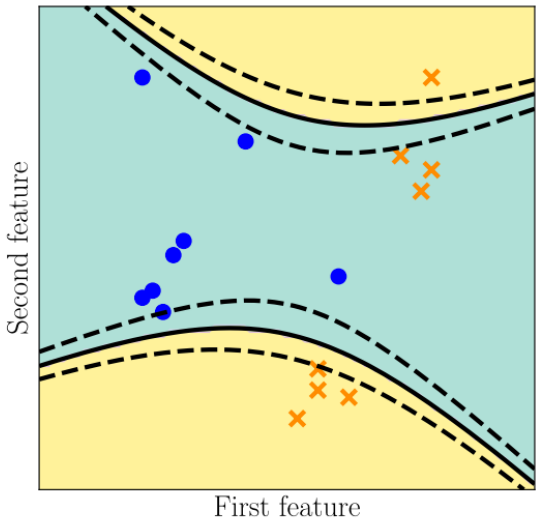
12.4 Kernels

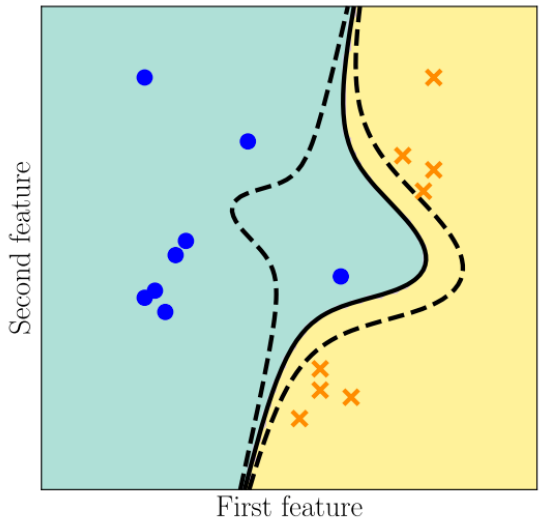


(a) SVM with linear kernel



(b) SVM with RBF kernel





12.4 Kernels

- Note that we are still solving for hyperplanes, that is, the hypothesis class of functions are still linear. The non-linear surfaces are due to the kernel function.
- **Remark.** There are multiple meanings of the word ”kernel.” In this chapter, the word ”kernel” comes from the idea of the reproducing kernel Hilbert space (RKHS) (Aronszajn, 1950; Saitoh, 1988). We have discussed the idea of the kernel in linear algebra (Section 2.7.3), where the kernel is another word for the null space. The third common use of the word ”kernel” in machine learning is the smoothing kernel in kernel density estimation (Section 11.5).

12.4 Kernels

- Since the explicit representation $\phi(\boldsymbol{x})$ is mathematically equivalent to the kernel representation $k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$, a practitioner will often design the kernel function such that it can be computed more efficiently than the inner product between explicit feature maps.
- For example, consider the polynomial kernel (Schölkopf and Smola, 2002), where the number of terms in the explicit expansion grows very quickly (even for polynomials of low degree) when the input dimension is large. The kernel function only requires one multiplication per input dimension, which can provide significant computational savings.
- Another example is the Gaussian radial basis function kernel (Schölkopf and Smola, 2002; Rasmussen and Williams, 2006), where the corresponding feature space is infinite dimensional.

12.4 Kernels

- In this case, we cannot explicitly represent the feature space but can still compute similarities between a pair of examples using the kernel.
- Another useful aspect of the kernel trick is that there is no need for the original data to be already represented as multivariate real-valued data.
- Note that the inner product is defined on the output of the function $\phi(\cdot)$, but does not restrict the input to real numbers. Hence, the function $\phi(\cdot)$ and the kernel function $k(\cdot,\cdot)$ can be defined on any object, e.g., sets, sequences, strings, graphs, and distributions (Ben-Hur et al., 2008; Gärtner, 2008; Shi et al., 2009; Sriperumbudur et al., 2010; Vishwanathan et al., 2010).
- Kernel Linear Regression: $\theta = \left(\Phi^\top \Phi + \lambda I\right)^{-1} \Phi^\top \boldsymbol{y}$, $y = \theta^\top \phi(\boldsymbol{x})$.
- Kernel PCA!

12.5 Numerical Solution

We conclude our discussion of SVMs by looking at how to express the problems derived in this chapter in terms of the concepts presented in Chapter 7. We consider two different approaches for finding the optimal solution for the SVM. First we consider the loss view of SVM 8.2.2 and express this as an unconstrained optimization problem. Then we express the constrained versions of the primal and dual SVMs as quadratic programs in standard form 7.3.2.

Consider the loss function view of the SVM (12.31). This is a convex unconstrained optimization problem, but the hinge loss (12.28) is not differentiable. Therefore, we apply a subgradient approach for solving it. However, the hinge loss is differentiable almost everywhere, except for one single point at the hinge $t = 1$. At this point, the gradient is a set of possible values that lie between 0 and -1.

12.5 Numerical Solution

Therefore, the subgradient g of the hinge loss is given by

$$g(t) = \begin{cases} -1 & t < 1 \\ [-1, 0] & t = 1 \\ 0 & t > 1 \end{cases} \tag{12.54}$$

Using this subgradient, we can apply the optimization methods presented in Section 7.1.

Both the primal and the dual SVM result in a convex quadratic programming problem (constrained optimization). Note that the primal SVM in (12.26a) has optimization variables that have the size of the dimension D of the input examples. The dual SVM in (12.41) has optimization variables that have the size of the number N of examples.

12.5 Numerical Solution

To express the primal SVM in the standard form (7.45) for quadratic programming, let us assume that we use the dot product (3.5) as the inner product. We rearrange the equation for the primal SVM (12.26a), such that the optimization variables are all on the right and the inequality of the constraint matches the standard form. This yields the optimization

$$\begin{aligned} \min_{\boldsymbol{w}, b, \boldsymbol{\xi}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N} \xi_n \\ \text{subject to} \quad & -y_n \boldsymbol{x}_n^\top \boldsymbol{w} - y_n b - \xi_n \leqslant -1 \\ & -\xi_n \leqslant 0 \end{aligned} \tag{12.55}$$

$n = 1, \ldots, N$.

12.5 Numerical Solution

By concatenating the variables $\boldsymbol{w}, b, \boldsymbol{x}_n$ into a single vector, and carefully collecting the terms, we obtain the following matrix form of the soft margin SVM:

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\begin{bmatrix} \boldsymbol{w} \\ b \\ \boldsymbol{\xi} \end{bmatrix}^{\top} \begin{bmatrix} \boldsymbol{I}_D & \boldsymbol{0}_{D,N+1} \\ \boldsymbol{0}_{N+1,D} & \boldsymbol{0}_{N+1,N+1} \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ b \\ \boldsymbol{\xi} \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{D+1,1} & C\boldsymbol{1}_{N,1} \end{bmatrix}^{\top} \begin{bmatrix} \boldsymbol{w} \\ b \\ \boldsymbol{\xi} \end{bmatrix}$$

$$\text{subject to} \quad \begin{bmatrix} -\boldsymbol{YX} & -\boldsymbol{y} & -\boldsymbol{I}_N \\ \boldsymbol{0}_{N,D+1} & & -\boldsymbol{I}_N \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ b \\ \boldsymbol{\xi} \end{bmatrix} \leqslant \begin{bmatrix} -\boldsymbol{1}_{N,1} \\ \boldsymbol{0}_{N,1} \end{bmatrix}.$$

12.5 Numerical Solution

In the preceding optimization problem, the minimization is over the parameters $\left[\boldsymbol{w}^\top, b, \boldsymbol{\xi}^\top\right]^\top \in \mathbb{R}^{D+1+N}$, and we use the notation: $\boldsymbol{I}_m$ to represent the identity matrix of size $m \times m$, $\mathbf{0}_{m,n}$ to represent the matrix of zeros of size $m \times n$, and $\mathbf{1}_{m,n}$ to represent the matrix of ones of size $m \times n$. In addition, $\boldsymbol{y}$ is the vector of labels $\left[y_1, \cdots, y_N\right]^\top$, $\boldsymbol{Y} = \operatorname{diag}(\boldsymbol{y})$ is an N by N matrix where the elements of the diagonal are from $\boldsymbol{y}$, and $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ is the matrix obtained by concatenating all the examples.

12.5 Numerical Solution

We can similarly perform a collection of terms for the dual version of the SVM (12.41). To express the dual SVM in standard form, we first have to express the kernel matrix $\boldsymbol{K}$ such that each entry is $K_{ij} = k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$. If we have an explicit feature representation $\boldsymbol{x}_i$ then we define $K_{ij} = \left\langle \boldsymbol{x}_i, \boldsymbol{x}_j \right\rangle$. For convenience of notation we introduce a matrix with zeros everywhere except on the diagonal, where we store the labels, that is, $\boldsymbol{Y} = \operatorname{diag}(\boldsymbol{y})$. The dual SVM can be written as

$$\min_{\boldsymbol{\alpha}} \frac{1}{2} \boldsymbol{\alpha}^\top \boldsymbol{Y} \boldsymbol{K} \boldsymbol{Y} \boldsymbol{\alpha} - \mathbf{1}_{N,1}^\top \boldsymbol{\alpha}$$

$$\text{subject to} \begin{bmatrix} \boldsymbol{y}^\top \\ -\boldsymbol{y}^\top \\ -\boldsymbol{I}_N \\ \boldsymbol{I}_N \end{bmatrix} \boldsymbol{\alpha} \leqslant \begin{bmatrix} \mathbf{0}_{N+2,1} \\ C\mathbf{1}_{N,1} \end{bmatrix}.$$

12.5 Numerical Solution

Remark. In Sections 7.3.1 and 7.3.2, we introduced the standard forms of the constraints to be inequality constraints. We will express the dual SVM's equality constraint as two inequality constraints, i.e.,

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \text{ is replaced by } \boldsymbol{A}\boldsymbol{x} \leqslant \boldsymbol{b} \text{ and } \boldsymbol{A}\boldsymbol{x} \geqslant \boldsymbol{b}.$$

Particular software implementations of convex optimization methods may provide the ability to express equality constraints.

12.5 Numerical Solution

Since there are many different possible views of the SVM, there are many approaches for solving the resulting optimization problem. The approach presented here, expressing the SVM problem in standard convex optimization form, is not often used in practice. The two main implementations of SVM solvers are Chang and Lin (2011) (which is open source) and Joachims (1999). Since SVMs have a clear and well-defined optimization problem, many approaches based on numerical optimization techniques (Nocedal and Wright, 2006) can be applied (Shawe-Taylor and Sun, 2011).