Lecture Outline

Reading: Chapter 7 Sparse Kernel Machines

This lecture covers

- Introduction
- Problem formulation for a maximum margin classifier
- Alternate problem formulation
- Dual Formulation

Students should study/review Appendix E on convex optimization using Lagrange multipliers.

Introduction

One of the significant limitations of the kernel method introduced in the previous chapter is that in order to compute the model parameters,

$$\mathbf{a} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{t} \tag{1}$$

where $K_{nm} = k(\mathbf{x}_n, \mathbf{x}_m)$, the kernel $k(\mathbf{x}_n, \mathbf{x}_m)$ must be evaluated for all possible pairs $\mathbf{x}_n$ and $\mathbf{x}_m$ of training points. In addition, the prediction

$$\begin{aligned} y(\mathbf{x}') &= \sum_{n=1}^{N} a_n k(\mathbf{x}_n, \mathbf{x}') \\ &= \mathbf{k}(\mathbf{x}')^T \mathbf{a} \end{aligned} \tag{2}$$

where $k_n(\mathbf{x}') = k(\mathbf{x}_n, \mathbf{x}')$ requires evaluation of the test point with each training point. These evaluations can be computational infeasible during training and can lead to excessive computation times during testing.

In this chapter we shall look at kernel-based algorithms that have *sparse* solutions, so that predictions for new inputs depend only on the kernel function evaluated at a subset $\mathcal{S}$ of the training data points i.e.

$$y(\mathbf{x}') = \sum_{m \in \mathcal{S}} a_m k(\mathbf{x}_m, \mathbf{x}'). \tag{3}$$

The training data points in $\mathcal{S}$ are known as *support vectors*.

An important property of the *support vector machine* (SVM) is that the determination of the model parameters corresponds to a convex optimization problem (solved with Lagrange multipliers), and so any local solution is also a global optimum. This differs from other methods' training which can be iterative, i.e. expectation maximization for GMM parameters or gradient descent for neural networks.

7.1 Maximum Margin Classifier

Consider the two-class (binary) linear classifier

$$y(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + b \tag{4}$$

where $\mathbf{w}$ is the $M \times 1$ vector of model parameters, $\boldsymbol{\phi}(\mathbf{x})$ denotes a fixed feature space transformation on the input $\mathbf{x}$ and b is the bias parameter. We assume training data $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and associated targets

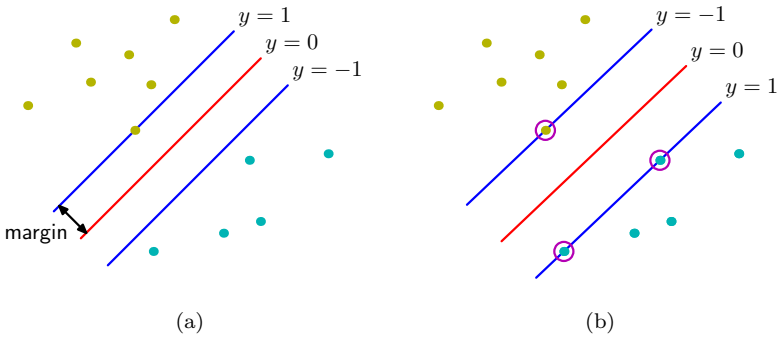
$\{t_1, \ldots, t_N\}$ where $t_n \in \{-1, 1\}$, i.e. if $\mathbf{x}_n \in \mathcal{C}_1$, $t_n = 1$ otherwise $t_n = -1$. Test data $\mathbf{x}'$ are classified according to the sign of $y(\mathbf{x}')$.

We assume that the training data set is linearly separable in *feature space*, so that there exists at least one choice of $\{\mathbf{w}, b\}$ such $y(\mathbf{x}_n) > 0$ for points having $t_n = +1$ and $y(\mathbf{x}_n) < 0$ for points having $t_n = -1$, so that $t_n y(\mathbf{x}_n) > 0$ for all training data points. In the next section, we shall relax this assumption.

As developed in Section 4.1, the perpendicular distance of a point $\mathbf{x}$ from a hyperplane defined by $y(\mathbf{x}) = 0$ is given by $|y(\mathbf{x})|/\|\mathbf{w}\|$. Thus the distance of a point $\mathbf{x}_n$ to the decision surface is

$$\frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|} = \frac{t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b)}{\|\mathbf{w}\|}. \tag{5}$$

The SVM determines the optimum choice of $\{\mathbf{w}, b\}$ by maximizing the *margin*, defined as the perpendicular distance between the decision boundary and the closest point $\mathbf{x}_n$ as illustrated in Fig. 1.



(a) (b)

Figure 1:

Thus the *maximum margin solution* is found by solving

$$\arg\max_{\mathbf{w},b} \left\{ \frac{1}{\|\mathbf{w}\|} \min_n \left[t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b) \right] \right\} \tag{6}$$

where the factor $1/\|\mathbf{w}\|$ is outside the optimization over n because $\mathbf{w}$ does not depend on n. Notice that we simultaneously find the $\mathbf{x}_n$ closest to the decision boundary and the $\{\mathbf{w}, b\}$ (which define the decision boundary) that maximizes the margin. We note also that

$$\arg\max_{\mathbf{w},b} \frac{1}{\|\mathbf{w}\|} \equiv \arg\min_{\mathbf{w},b} \|\mathbf{w}\| \tag{7}$$

which will be used later to reformulate the optimization function.

Direct solution of this optimization problem is very complex, so we convert it to an equivalent problem that is much easier to solve. This equivalent problems is a quadratic programming problem where we minimize a quadratic function subject to a set of linear constraints.

(Optional) Alternate Problem Formulation

We begin by noting that rescaling $\mathbf{w}, b$ does not change the perpendicular distance between the decision boundary and a point $\mathbf{x}$

$$\frac{t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + b)}{\|\mathbf{w}\|} = \frac{\kappa t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + b)}{\kappa\|\mathbf{w}\|} = \frac{t_n(\kappa\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + \kappa b)}{\|\kappa\mathbf{w}\|}. \tag{8}$$

We can use this freedom to rescale to conveniently set

$$t_n \left(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b\right) = t_n y(\mathbf{x}_n) = 1 \tag{9}$$

for the point closest to the decision boundary. Since all other training data points will be > 1, we have for all training data points

$$t_n \left(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b\right) = t_n y(\mathbf{x}_n) \geq 1. \tag{10}$$

The data points where equality holds are said to be *active* while the other data points are said to be *inactive*.

By definition, there will always be at least one active constraint because there will always be a closest point [inside term in (6)]. Once the margin has been maximized there will be at least two active constraints [$1/\|\mathbf{w}\|$ and the inside term in (6)]. Of course maximizing $1/\|\mathbf{w}\|$ is equivalent to minimizing $\|\mathbf{w}\|$ or $\|\mathbf{w}\|^2$. Thus we have to solve the optimization problem

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \tag{11}$$

subject to the constraint (10). Note that the optimization problem involves M variables, $\mathbf{w}$ (b is computed once $\mathbf{w}$ is known.

Dual Formulation

Using (7) with (6) gives a classic *quadratic programming* problem where we are trying to *minimize* a quadratic function subject to a set of linear inequality constraints

$$\arg\min_{\mathbf{w},b} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 \ \min_n \left[t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b) \right] \right\}. \tag{12}$$

This problem is solved using a Lagrangian function

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \left\{ t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b) - 1 \right\} \tag{13}$$

where $\mathbf{a} = (a_1, \ldots, a_N)^T$ are the *Lagrange multipliers*. Here we minimize with respect to $\mathbf{w}, b$ and maximize (because of the $-$ sign) with respect to $\mathbf{a}$. Methods for actually solving the quadratic programming problem will be discussed shortly.

We can eliminate $\mathbf{w}$ and b from $L(\mathbf{w}, b, \mathbf{a})$ and obtain the *dual formulation* to (13) (see text for derivation)

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \tag{14}$$

subject to the constraints

$$a_n \geq 0, \quad n = 1, \ldots, N \tag{15}$$

$$\sum_{n=1}^{N} a_n t_n = 0. \tag{16}$$

The dual formulation incorporates the kernel function, defined by $k(\mathbf{x}_n, \mathbf{x}_m) = \boldsymbol{\phi}(\mathbf{x}_n)^T \boldsymbol{\phi}(\mathbf{x}_m)$. In order to solve for $\mathbf{a}$, we maximize (14) subject to the constraints.

Although the dual formulation involves minimizing over N variables instead of M variables ($N \gg M$), it is expressed in terms of kernels and so the maximum margin classifier can be applied efficiently to feature spaces whose dimensionality exceeds the number of data points including infinite feature spaces.

We will discuss the actual solution on p. 335 where we use the Sequential Minimization Optimization (SMO) algorithm.

Support Vectors

Once the system has been trained, i.e. we have solved for $\mathbf{a}$ in the dual formulation, we classify new data $\mathbf{x}'$ according to

$$y(\mathbf{x}') = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}_n, \mathbf{x}') + b \tag{17}$$

where we decide $\mathcal{C}_1$ if $y(\mathbf{x}') \geq 0$ and $\mathcal{C}_2$ otherwise.

In order to use (17) all that remains is to solve for the bias b. It can be shown that for every training data point either $a_n = 0$ or $t_n y(\mathbf{x}_n) = 1$. Thus any data points for which $a_n = 0$ does not contribute to (17) can be discarded. Those data points for which $t_n y(\mathbf{x}_n) = 1$ correspond to points that lie on the maximum margin hyperplanes in feature space and are called *support vectors* and are stored in memory. This property is central to the applicability of SVMs. Hence we can solve for the bias via

$$t_n \left(\sum_{m \in \mathcal{S}} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m) + b \right) = 1 \tag{18}$$

where $\mathcal{S}$ denotes the set of indices of the support vectors.

Figure 2 shows an example of the classification resulting from training a SVM using a Gaussian kernel. Although the data set is not linearly separable in the 2-D space $\mathbf{x}$, it is linearly separable in the nonlinear feature space defined implicitly by the nonlinear kernel. In addition, the maximum margin hyperplane is defined by the location of the support vectors.

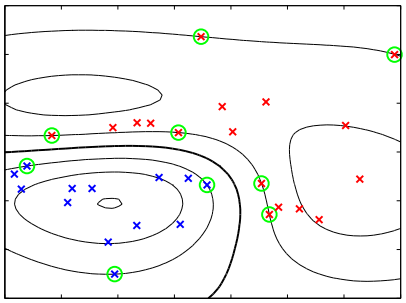


Figure 2: