BUSI97287: Advanced Machine Learning

Support Vector Machines and the Kernel Trick

Martin Haugh

Imperial College Business School

Email: `martin.b.haugh@gmail.com`

Some of the figures in this presentation are taken from "An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani.

Additional References: Bishop's PRML and HTF's TESL.

Required Reading: Chapter 9 of *ISLR* by James, Witten, Hastie and Tibshirani

Outline

Introduction to Support Vector Machines

Support vector machines are non-probabilistic binary linear classifiers.

The use of basis functions and the kernel trick mitigates the constraint of the SVM being a linear classifier

– in fact SVMs are particularly associated with the kernel trick.

Only a subset of data-points are required to define the SVM classifier

- these points are called support vectors.

SVMs are very popular classifiers and applications include

- text classification
- outlier detection
- face detection
- database marketing
- and many others.

SVMs can also be used for multi-class classification and regression.

Classification: the Separable Case

There are two classes which are assumed to be linearly separable

- A strong assumption – to be relaxed later.

Training data: $\mathbf{x}_1, \ldots, \mathbf{x}_n$ with corresponding targets $t_1, \ldots, t_n \in \{-1, 1\}$.

We consider a linear classification rule of the form of the form

$$\begin{aligned} h(\mathbf{x}) &= \text{sign}\left(\mathbf{w}^\top \mathbf{x} + b\right) \\ &= \text{sign}\left(y(\mathbf{x})\right) \end{aligned}$$

where $y(\mathbf{x}) := \mathbf{w}^\top \mathbf{x} + b$.

When data linearly separable there are infinitely many separating hyperplanes

- see Fig 9.2 from ISLR.

Question: Which hyperplane should we select?

Answer: The hyperplane with the maximum margin - see Fig 9.3 from ISLR.

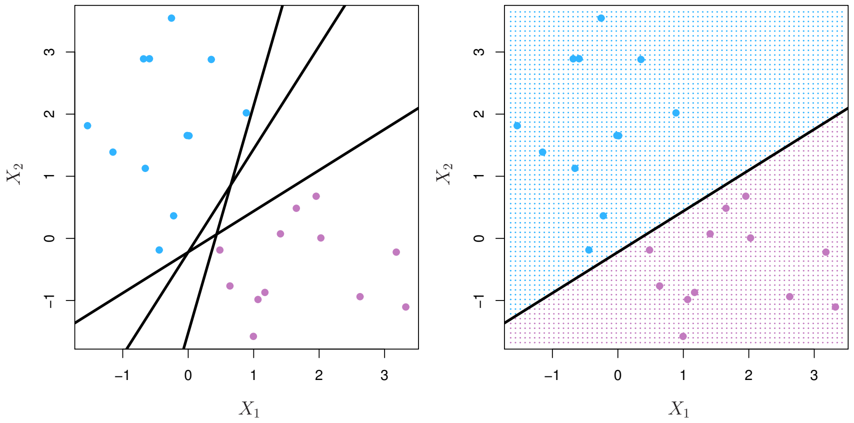


Figure 9.2 from ISLR: Left: There are two classes of observations, shown in blue and in purple, each of which has measurements on two variables. Three separating hyperplanes, out of many possible, are shown in black. Right: A separating hyperplane is shown in black. The blue and purple grid indicates the decision rule made by a classifier based on this separating hyperplane: a test observation that falls in the blue portion of the grid will be assigned to the blue class, and a test observation that falls into the purple portion of the grid will be assigned to the purple class.

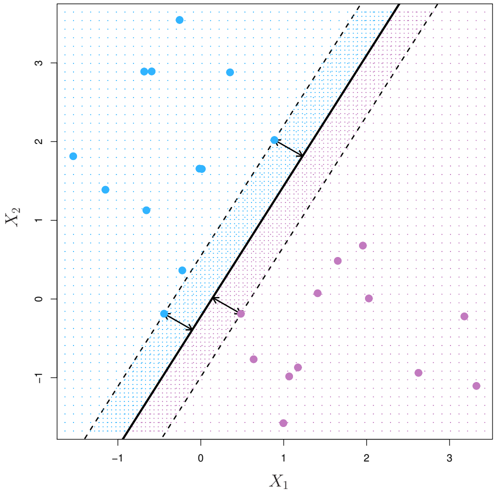


Figure 9.3 from ISLR: There are two classes of observations, shown in blue and in purple. The maximal margin hyperplane is shown as a solid line. The margin is the distance from the solid line to either of the dashed lines. The two blue points and the purple point that lie on the dashed lines are the support vectors, and the distance from those points to the margin is indicated by arrows. The purple and blue grid indicates the decision rule made by a classifier based on this separating hyperplane.

Classification: the Separable Case

Note that for a given hyperplane

$$y(\mathbf{x}) := \mathbf{w}^\top \mathbf{x} + b = 0$$

we can re-scale $(\mathbf{w}, b)$ without changing the decision boundary.

Can therefore choose $(\mathbf{w}, b)$ so that training points closest to boundary satisfy

$$y(\mathbf{x}) = \pm 1.$$

- Let $\mathbf{x}_1$ be a closest point from class with $t_1 = -1$. Then

$$\begin{aligned} y(\mathbf{x}_1) &= \mathbf{w}^\top \mathbf{x}_1 + b \\ &= -1. \end{aligned}$$

- And let $\mathbf{x}_2$ be a closest point from class with $t_2 = 1$. Then

$$\begin{aligned} y(\mathbf{x}_2) &= \mathbf{w}^\top \mathbf{x}_2 + b \\ &= 1. \end{aligned}$$

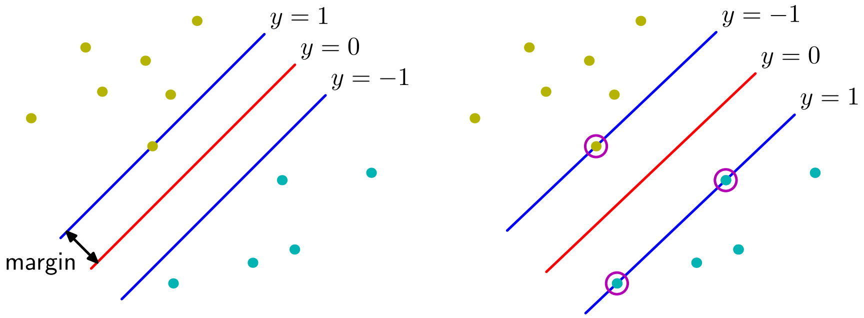


Figure 7.1 from Bishop: The margin is defined as the perpendicular distance between the decision boundary and the closest of the data points, as shown on the left figure. Maximizing the margin leads to a particular choice of decision boundary, as shown on the right. The location of this boundary is determined by a subset of the data points, known as support vectors, which are indicated by the circles.

Geometry of Maximizing the Margin

Perpendicular distance of a point $\mathbf{x}$ from hyperplane $\mathbf{w}^\top \mathbf{x} + b = 0$ given by

$$\frac{|\mathbf{w}^\top \mathbf{x} + b|}{||\mathbf{w}||}.$$

Therefore distance of closest points in each class to the hyperplane is $\frac{1}{||\mathbf{w}||}$.

An SVM seeks the maximum margin classifier that separates all the data

- seems like a good idea
- but can also be justified by statistical learning theory.

Geometry of Maximizing the Margin

Maximizing the margin $1/||\mathbf{w}||$ is equivalent to minimizing $f(\mathbf{w}) := \frac{1}{2}\mathbf{w}^\top\mathbf{w}$.

Therefore obtain the following primal problem for the separable case:

$$\min_{\mathbf{w},b} \quad f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} \tag{1}$$

$$\text{subject to} \quad t_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right) \geq 1, \quad i = 1, \ldots, n \tag{2}$$

Note that (2) ensures that all the training points are correctly classified.

The Primal Problem

The primal problem is a quadratic program with linear inequality constraints

- moreover it is convex and therefore has a unique minimum.

Clear from problem geometry that only points closest to the boundary are required to define the optimal hyperplane

- these points are called the support vectors – see Fig. 7.1 from Bishop.

Could stop here but will go to the corresponding dual problem to better understand SVMs and the kernel trick.

Will also see via the dual solution that the optimal hyperplane can be expressed using only the support vectors.

The Dual Problem in the Separable Case

We use a Lagrange multiplier $\alpha_i \geq 0$ to relax each constraint in (2).

Lagrangian then given by

$$L(\mathbf{w}, b; \alpha) \; = \; \frac{1}{2}\mathbf{w}^\top \mathbf{w} \; + \; \sum_{i=1}^{n} \alpha_i \left(1 - t_i \left(\mathbf{w}^\top \mathbf{x}_i + b\right)\right) \tag{3}$$

Now wish to solve for

$$g(\boldsymbol{\alpha}) := \min_{\mathbf{w}, b} \; L(\mathbf{w}, b; \boldsymbol{\alpha}).$$

Let $(\mathbf{w}^*, b^*)$ is the optimal solution to the primal problem.

Exercise: Argue that $g(\boldsymbol{\alpha}) \leq f(\mathbf{w}^*)$.

The Dual Problem in the Separable Case

Can therefore formulate the dual problem:

$$\max_{\boldsymbol{\alpha} \geq 0} \; g(\boldsymbol{\alpha}) \tag{4}$$

Since primal problem is convex it follows that minimum of primal equals maximum of dual problem.

That is:

$$g(\boldsymbol{\alpha}^*) = f(\mathbf{w}^*)$$

where $\boldsymbol{\alpha}^*$ is the optimal solution to the dual problem - this is strong duality.

The Dual Problem in the Separable Case

Can be shown the dual problem reduces to solving

$$\max_{\boldsymbol{\alpha} \geq 0} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j \mathbf{x}_i^\top \mathbf{x}_j \tag{5}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i t_i = 0 \tag{6}$$

- also a convex quadratic program, but now with a single linear constraint.

The Kernel Trick

The kernel-trick is a very commonly used technique in regression, classification, PCA, clustering etc.

It allows certain problems to be easily embedded in much higher dimensional spaces and often infinite dimensional spaces

- but without having to do an infinite amount of work.

Suppose instead of using $\mathbf{x} \in \mathbb{R}^m$ to describe the inputs we instead use a feature map, $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{R}^M$, often with $M >> m$.

Then if the data is linearly separable in $\mathbb{R}^M$ can solve the same dual problem as (5) and (6) except we replace $\mathbf{x}_i^\top \mathbf{x}_j$ with $\boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\phi}(\mathbf{x}_j)$:

$$\max_{\boldsymbol{\alpha} \geq 0} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j \, \boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\phi}(\mathbf{x}_j) \tag{7}$$

$$\text{subject to} \qquad \sum_{i=1}^{n} \alpha_i t_i = 0$$

A Detour on Kernels

Define the Gram matrix $\mathbf{K}$ to be the $n \times n$ matrix with

$$\begin{aligned} \mathbf{K}_{ij} &:= \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) \\ &=: k(\mathbf{x}_i, \mathbf{x}_j) \end{aligned} \tag{8}$$

For any set of points, $\mathbf{x}_1, \ldots, \mathbf{x}_n$, the kernel matrix $\mathbf{K}$ is positive semi-definite so that $\mathbf{z}^\top \mathbf{K} \mathbf{z} \geq 0$ for all $\mathbf{z} \in \mathbb{R}^n$.

Definition. We say a function, $k(\mathbf{x}, \mathbf{x}')$, is a kernel if it corresponds to a scalar product, $\phi(\mathbf{x})^\top \phi(\mathbf{x}')$ in some feature space, $\mathbb{R}^M$, possibly with $M = \infty$.

Mercer's Theorem. A necessary and sufficient condition for a function $k(\mathbf{x}, \mathbf{x}')$ to be a kernel is that the corresponding Gram matrix $\mathbf{K}$ be positive semi-definite for all possible choices of $\mathbf{x}_1, \ldots, \mathbf{x}_n$.

A Detour on Kernels

Key Implication of Mercer's Theorem:

Possible to define a feature map $\phi(\cdot)$ implicitly using kernel function $k(\cdot, \cdot)$.

Note that $\phi(\cdot)$ not explicitly required to state dual problem in (7) since

$$k(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\phi}(\mathbf{x}_j)$$

and so only kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ required.

- A big advantage since far less work may be required to compute $k(\cdot, \cdot)$.
- In fact if $\phi(\cdot)$ infinite-dimensional then could never write $\phi(\cdot)$ out explicitly!

A Detour on Kernels

e.g. Let $m = 2$ and define $k(\mathbf{x}, \mathbf{x}') := (\mathbf{x}^\top \mathbf{x}')^2$ for $\mathbf{x}$, $\mathbf{x}' \in \mathbb{R}^2$.

Easy to check that $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ where

$$\boldsymbol{\phi}(\mathbf{x}) := \begin{bmatrix} x_1^2 & \sqrt{2} x_1 x_2 & x_2^2 \end{bmatrix}^\top .$$

But:

- Calculating $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^\top \mathbf{x}')^2$ requires $O(m)$ work
- Whereas calculating $\boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ requires $O(M) = O(m^2)$ work.

More generally, could define a polynomial kernel of degree p:

$$k(\mathbf{x}, \mathbf{x}') := (\mathbf{x}^\top \mathbf{x}' + c)^p \tag{9}$$

- Then computing $k(\mathbf{x}, \mathbf{x}')$ will still require $O(m)$ work.
- But computing $\boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{x}')$ (for corresponding $\boldsymbol{\phi}(\cdot)$) requires $O(m^p)$ work.

Constructing New Kernels (Bishop)

There are many different kernel functions – and many ways to define new kernels from old kernels.

Suppose:

- $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$ are valid kernels.
- $c > 0$ is a constant.
- $f(\cdot)$ is any function.
- $q(\cdot)$ is a polynomial with nonnegative coefficients.
- $\boldsymbol{\phi}(\mathbf{x})$ is a function from $\mathbf{x}$ to $\mathbb{R}^M$.
- $k_3(\cdot, \cdot)$ is a valid kernel in $\mathbb{R}^M$.
- $\mathbf{A}$ is a symmetric positive semi-definite matrix.
- $\mathbf{x}_a$ and $\mathbf{x}_b$ are variables (not necessarily disjoint) with $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$.
- k_a and k_b are valid kernel functions over their respective spaces.

Constructing New Kernels (Bishop)

Then the following are all valid kernels:

$$\begin{aligned}
k(x,x') &= ck_1(\mathbf{x},\mathbf{x}') && (10)\\
k(x,x') &= f(\mathbf{x})k_1(\mathbf{x},\mathbf{x}')f(\mathbf{x}') && (11)\\
k(x,x') &= q(k_1(\mathbf{x},\mathbf{x}'))\\
k(x,x') &= \exp(k_1(\mathbf{x},\mathbf{x}')) && (12)\\
k(x,x') &= k_1(\mathbf{x},\mathbf{x}') + k_2(\mathbf{x},\mathbf{x}')\\
k(x,x') &= k_1(\mathbf{x},\mathbf{x}')k_2(\mathbf{x},\mathbf{x}')\\
k(x,x') &= k_3(\boldsymbol{\phi}(\mathbf{x}),\boldsymbol{\phi}(\mathbf{x}'))\\
k(x,x') &= \mathbf{x}^\top \mathbf{A}\mathbf{x}'\\
k(x,x') &= k_a(\mathbf{x}_a,\mathbf{x}'_a) + k_b(\mathbf{x}_b,\mathbf{x}'_b)\\
k(x,x') &= k_a(\mathbf{x}_a,\mathbf{x}'_a)k_b(\mathbf{x}_b,\mathbf{x}'_b)
\end{aligned}$$

Exercise: Use the kernel results above to show that (9) defines a kernel.

The Gaussian Kernel

Gaussian kernel given by:

$$k(\mathbf{x}, \mathbf{x}') \;:=\; \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right). \tag{13}$$

It's a valid kernel because

$$\begin{aligned}\exp\left(\frac{-||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right) &= \exp\left(\frac{-\mathbf{x}^\top\mathbf{x}}{2\sigma^2}\right)\exp\left(\frac{\mathbf{x}^\top\mathbf{x}'}{\sigma^2}\right)\exp\left(\frac{-\mathbf{x}'^\top\mathbf{x}'}{2\sigma^2}\right)\\ &= f(\mathbf{x})\exp\left(\frac{\mathbf{x}^\top\mathbf{x}'}{\sigma^2}\right)f(\mathbf{x}')\end{aligned}$$

and now can apply (in order) (10), (12) and (11).

Gaussian kernel sometimes called the (Gaussian) radial basis kernel.

Fact: Feature vector $\phi(\mathbf{x})$ corresponding to Gaussian kernel is ∞-dimensional!

Constructing Kernels for Other Objects

The kernel trick can be extended to inputs that are symbolic and not just vectors of real numbers.

Examples of such inputs are graphs, sets, strings, and text documents.

e.g. Consider a fixed set and define the space consisting of all possible subsets of this set. If A_1 and A_2 are two such subsets then let

$$k(A_1, A_2) := 2^{|A_1 \cap A_2|}$$

where $|A|$ denotes the number of elements in the set A.

$k(\cdot, \cdot)$ is a valid kernel because it can be shown to correspond to an inner product $\phi(\cdot)$ in a feature space

- so could easily use SVMs to classify these sets.

The Kernel-Separated Dual

Returning to SVMs, when the data is kernel-separated our dual problem becomes:

$$\begin{aligned} \max_{\alpha \geq \mathbf{0}} \quad & \textstyle\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} \quad & \textstyle\sum_{i=1}^{n} \alpha_i t_i = 0. \end{aligned}$$

Fact:

Given a solution $\boldsymbol{\alpha}^*$ to the dual, can obtain corresponding optimal b^* via

$$b^* = t_j - \sum_{i=1}^{n} \alpha_i^* t_i k(\mathbf{x}_i, \mathbf{x}_j) \quad \text{for any } \alpha_j^* > 0$$

and for a new data-point $\mathbf{x}$, the prediction

$$\text{sign}\left(\mathbf{w}^{*\top} \boldsymbol{\phi}(\mathbf{x}) + b^*\right) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i^* t_i k(\mathbf{x}_i, \mathbf{x}) + b^*\right).$$

So feature map $\boldsymbol{\phi}(\cdot)$ never required for prediction!

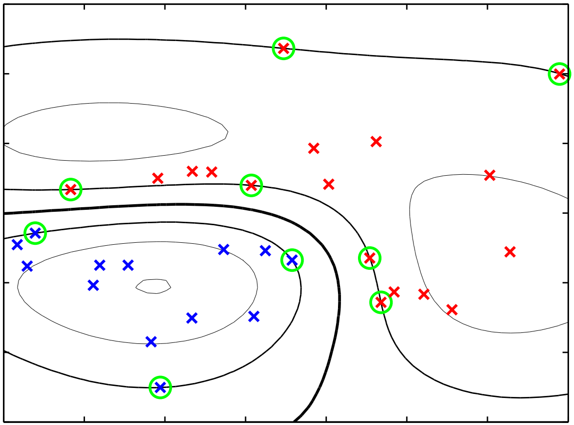
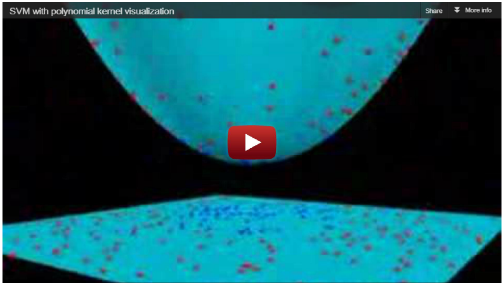


Figure 7.2 from Bishop: Example of synthetic data from two classes in two dimensions showing contours of constant $y(x)$ obtained from a support vector machine having a Gaussian kernel function. Also shown are the decision boundary, the margin boundaries, and the support vectors.

- Note that the data is linearly separable in the Gaussian-kernel space but not in the original space.



A Demo of SVM Classification with Polynomial Kernel by Udi Aharoni

Classification: the Non-Separable Case

In general data will be non-separable so primal problem of (1) and (2) infeasible.

Several ways to proceed: **e.g.** minimize $\#$ of misclassified points, but this is NP-hard.

Instead we allow points to violate margin constraints and penalize accordingly in objective function. This yields the more general non-separable primal problem:

$$\min_{\mathbf{w},\boldsymbol{\xi},b} \ \frac{1}{2}\mathbf{w}^\top\mathbf{w} \ + \ C\sum_{i=1}^{n}\xi_i \tag{14}$$

$$\begin{aligned} \text{subject to} \quad t_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right) &\geq 1-\xi_i, \quad i=1,\ldots,n \\ \xi_i &\geq 0, \quad i=1,\ldots,n \end{aligned} \tag{15}$$

- again a convex quadratic programming problem with linear constraints.

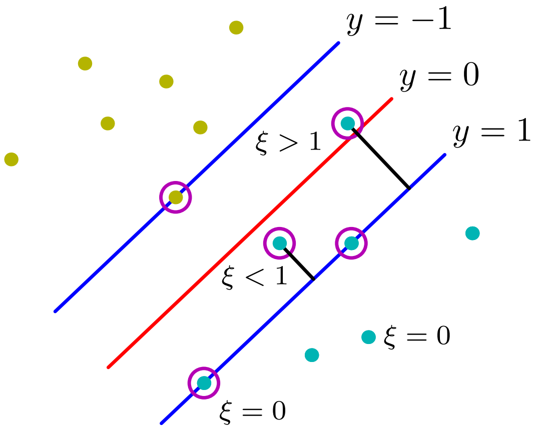


Figure 7.3 from Bishop: Illustration of the slack variables in $\xi_n \geq 0$. Data points with circles around them are support vectors.

- Note that the slack variables allow points to be misclassified.

Questions: What does C control? How might C be chosen?

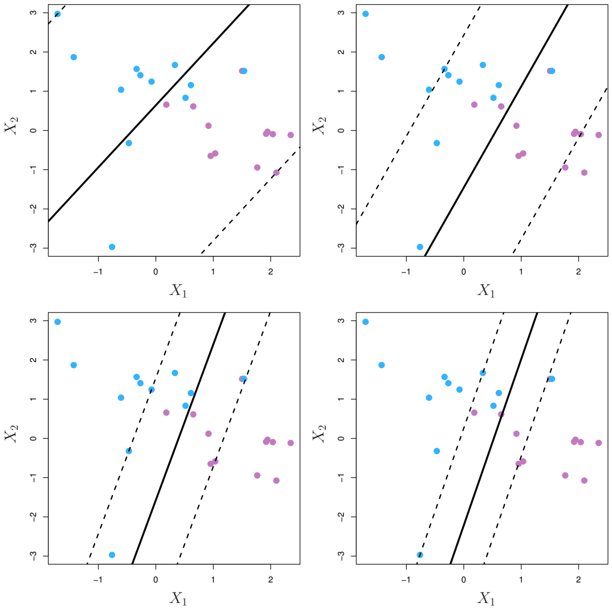


Figure 9.7 from ISLR: Different classifiers resulting from different values of C.

The Non-Separable Dual Problem

As with the separable case, it's more convenient to work with the dual.

Because the primal problem is convex the dual and primal have equal optimal objective functions.

The non-separable dual problem reduces to

$$\begin{array}{lll} \max\limits_{\boldsymbol{\alpha}\geq 0,\, \boldsymbol{\lambda}\geq 0} & \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i\alpha_j t_i t_j \mathbf{x}_i^\top \mathbf{x}_j & \\ \text{subject to} & \sum_{i=1}^{n} \alpha_i t_i = 0 & \\ & \alpha_i \leq C, & i = 1, \ldots, n \end{array}$$

- again a convex quadratic program with linear constraints
- same as earlier dual problem but now must also have $\alpha_i \leq C$.

But What If a Linear Classifier Clearly Inappropriate?

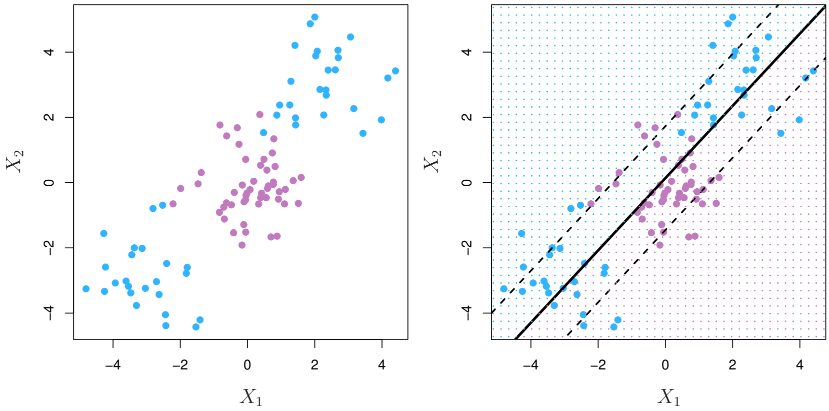


Figure 9.8 from ISLR: Left: The observations fall into two classes, with a non-linear boundary between them. Right: The support vector classifier seeks a linear boundary, and consequently performs very poorly.

Use a Kernel!

As with the separable case, can easily apply the kernel trick to obtain the following general non-separable dual problem:

$$\max_{\boldsymbol{\alpha} \geq 0} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{16}$$

subject to

$$\sum_{i=1}^{n} \alpha_i t_i = 0 \tag{17}$$

$$\alpha_i \leq C, \quad i = 1, \ldots, n \tag{18}$$

Given an optimal solution, $\boldsymbol{\alpha}^*$, can recover the SVM classifier as:

$$b^* = t_j - \sum_{i=1}^{n} \alpha_i^* t_i k(\mathbf{x}_i, \mathbf{x}_j) \quad \text{for any } C > \alpha_j^* > 0$$

and, for a new data-point $\mathbf{x}$, the prediction:

$$\text{sign}\left(\mathbf{w}^{*\top} \boldsymbol{\phi}(\mathbf{x}) + b^*\right) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i^* t_i \, k(\mathbf{x}_i, \mathbf{x}) + b^*\right). \tag{19}$$

But There's Rarely a Single "Right" Kernel ...

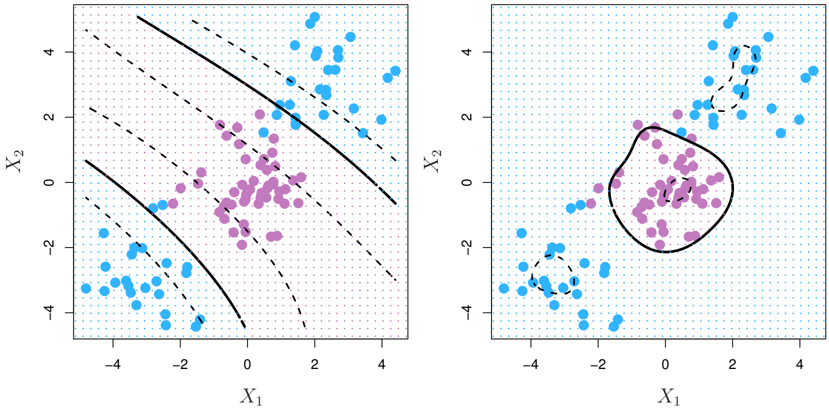


Figure 9.9 from ISLR: Left: An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule. Right: An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.

Appendix: Multiclass SVMs

SVMs can also be used for multi-class problem with K classes, $C_1, \ldots, C_k$.

There are two commonly used approaches:

1. One-Versus-the-Rest
2. One-Versus-One

Application of SVMs to multi-class problems is ad-hoc with many limitations

- SVMs were designed for binary classification!

Appendix: Multiclass SVMs

One-Versus-the-Rest:

Train K different SVM's: k^{th} SVM trained on "C_k" versus "not C_k".

- Generally not a good idea to use majority voting among the K classifiers.
- Instead final classification usually determined by taking

$$y(\mathbf{x}) = \max_k y_k(\mathbf{x}) \tag{20}$$

  where $y_k(\mathbf{x})$ is the linear classifier of k^{th} SVM.
- But there are scaling problems with using (20) and the balance of the training points in each of the K SVMs generally poor.

One-Versus-One:

Train all $\binom{K}{2}$ two-class SVMs and then use majority voting to obtain final classifier

- A lot of computational work required when K is large.

Appendix: Comparing the SVM and Logistic Loss Functions

Interesting to compare the error functions used by various classifiers.

Primal objective function of the SVM classifier may be written (why?) as

$$\begin{aligned} \text{Obj. Fun.} &= \frac{1}{2}\mathbf{w}^\top \mathbf{w} \; + \; C\sum_{i=1}^{n} \xi_i \\ &\equiv \frac{1}{2C}||\mathbf{w}||^2 + \sum_{i=1}^{n} E_{sv}(t_i y_i) \end{aligned} \tag{21}$$

where $y_i := y(\mathbf{x}_i)$ and $E_{sv}(\cdot)$ is the hinge error function:

$$E_{sv}(t_i y_i) \; := \; [1 - t_i y_i]^+.$$

Remark: Role of C clear from (21) – not so when SVMs originally introduced.

- E_{sv} responsible for inducing sparsity in optimal SVM dual solution.
- E_{sv} an approximation to $0 - 1$ loss function and is in fact similar to loss function used in logistic regression.

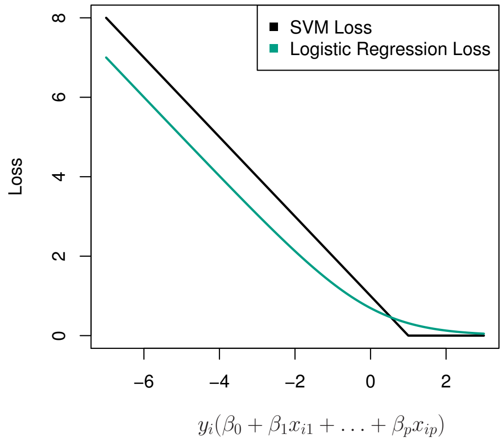


Figure 9.12 from ISLR: The SVM and logistic regression loss functions are compared, as a function of $y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip})$. When $y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip})$ is greater than 1, then the SVM loss is zero, since this corresponds to an observation that is on the correct side of the margin. Overall, the two loss functions have quite similar behavior.

- Similarity of SVM and logistic loss functions explains why they tend to have similar performance.
- But SVMs typically applied with a kernel and not in original space.

Appendix: Solving the Dual Problem in the Separable Case

To solve (4) first need to solve for $g(\boldsymbol{\alpha})$: the first order conditions (FOC) are

$$\begin{aligned}
\frac{\partial L}{\partial b} &= 0 \Rightarrow \sum_{i=1}^{n} \alpha_i t_i = 0 \\
\frac{\partial L}{\partial \mathbf{w}} &= 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{n} \alpha_i t_i \mathbf{x}_i
\end{aligned} \tag{22}$$

The FOC are necessary and sufficient for optimality. Can substitute them into (3) to obtain

$$\begin{aligned}
L(\mathbf{w}, b; \alpha) &= \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j \mathbf{x}_i^\top \mathbf{x}_j + \sum_{i=1}^{n} \alpha_i \left(1 - t_i \left(\sum_{j=1}^{n} \alpha_j t_j \mathbf{x}_j^\top \mathbf{x}_i + b \right) \right) \\
&= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j \mathbf{x}_i^\top \mathbf{x}_j
\end{aligned}$$

Appendix: Solving the Dual Problem in the Separable Case

Then dual problem in the separable case reduces to

$$\max_{\boldsymbol{\alpha} \geq 0} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j t_i t_j \mathbf{x}_i^\top \mathbf{x}_j \tag{23}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i t_i = 0 \tag{24}$$

- also a convex quadratic program, but now with a single linear constraint.

The complementary slackness conditions imply that only the support vectors will have non-zero α's in the optimum solution.

Let $\boldsymbol{\alpha}^*$ be the optimal solution to the dual problem. Then (22) yields

$$\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i^* t_i \mathbf{x}_i$$

and we obtain b^* by noting that $t_i \left(\mathbf{w}^{*\top} \mathbf{x}_i + b^* \right) = 1$ for any i with $\alpha_i^* > 0$

- so find such an i and then solve for b^*.

Appendix: Regression and SVMs

SVMs have also been proposed for regression – quite popular in practice.

We replace quadratic error function with an ϵ-insensitive error function, $E_\epsilon(\cdot)$:

$$E_\epsilon\left(y(\mathbf{x}) - t\right) \;:=\; \begin{cases} 0, & \text{if } |y(\mathbf{x}) - t| < \epsilon \\ |y(\mathbf{x}) - t| - \epsilon, & \text{otherwise} \end{cases} \tag{25}$$

where $y(\mathbf{x}) := \mathbf{w}^\top \mathbf{x}_i + b$ and t is the dependent variable.

$E_\epsilon(\cdot)$ only penalizes predictions that are more than ϵ away from the target

- results in sparse solutions where only a subset of the training points matter.

Regularized objective function of SVM regression then given by

$$C \sum_{i=1}^{N} E_\epsilon\left(\mathbf{w}^\top \mathbf{x}_i + b - t_i\right) \;+\; \frac{1}{2}||\mathbf{w}||^2$$

– C can be chosen via cross-validation.

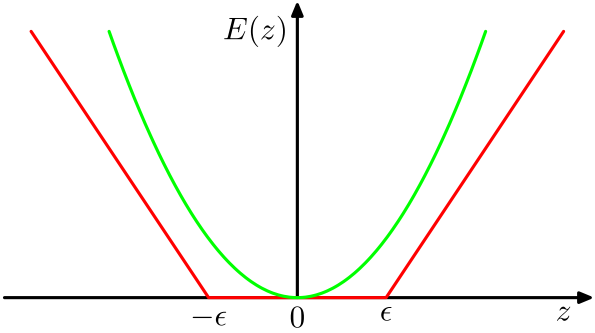


Figure 7.6 from Bishop: Plot of an ϵ-insensitive error function (in red) in which the error increases linearly with distance beyond the insensitive region. Also shown for comparison is the quadratic error function (in green).

Appendix: Regression and SVMs

Can reformulate primal regression SVM problem as

$$\min_{\mathbf{w}, b, \boldsymbol{\xi} \geq 0, \hat{\boldsymbol{\xi}} \geq 0} \quad C \sum_{i=1}^{n} \left(\xi_i + \hat{\xi}_i \right) \; + \; \frac{1}{2} ||\mathbf{w}||^2$$

$$\text{subject to} \quad t_i \leq y(\mathbf{x}_i) + \epsilon + \xi_i \tag{26}$$

$$t_i \geq y(\mathbf{x}_i) - \epsilon - \hat{\xi}_i \tag{27}$$

Slack variables $\boldsymbol{\xi}$ and $\hat{\boldsymbol{\xi}}$ non-zero only for predictions outside ϵ-insensitive tube.

Primal problem convex so can instead work with its dual problem

- obtained via the Lagrangian and relaxing (26), (27) and non-neg. constraints
- kernel trick can also be used with dual
- dual problem can be solved numerically to obtain a fitted function of the form

$$y(\mathbf{x}) = \sum_{i=1}^{n} (a_i - \hat{a}_i) k(\mathbf{x}, \mathbf{x}_i) + b$$

where a_i's and $\hat{a}_i$'s are Lagrange multipliers for kernel-consistent versions of (26) and (27).

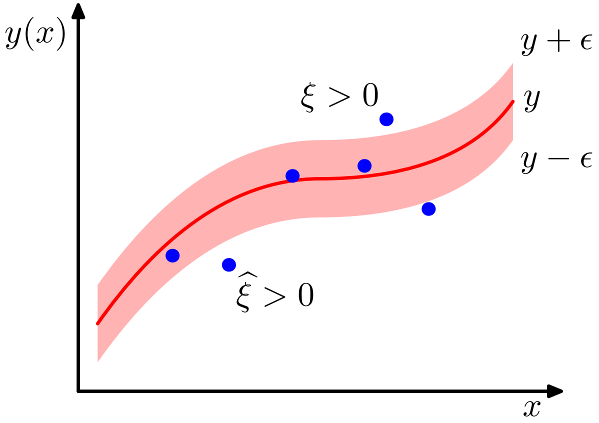


Figure 7.7 from Bishop: Illustration of SVM regression, showing the regression curve together with the ϵ-insensitive 'tube'. Also shown are examples of the slack variables ξ and $\hat{\xi}$. Points above the ϵ-tube have $\xi > 0$ and $\hat{\xi} = 0$, points below the ϵ-tube have $\xi = 0$ and $\hat{\xi} > 0$, and points inside the ϵ-tube have $\xi = \hat{\xi} = 0$.

Only points on the edge of the tube or outside the tube are support vectors.