

Lecture 09. Support Vector Machines (Chapter 9)

Ping Yu

HKU Business School
The University of Hong Kong

- Here we approach the two-class classification problem in a direct way:

We try and find a plane that separates the classes in feature space.

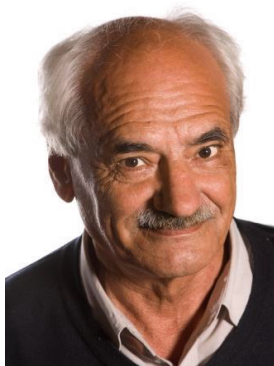
This results in the **maximal margin classifier**.

- If we cannot, we get creative in two ways:
 - 1 We soften what we mean by “separates”, which results in the **support vector classifier**.
 - 2 We enrich and enlarge the feature space (to accommodate non-linear class boundaries) so that separation is possible, which results in the **support vector machine (SVM)**.
- The SVM approach was developed in the 1990s, and is often considered one of the best "out of the box" classifiers (like BART).

History of SVM



Vladimir N. Vapnik (1936-, Meta)



Alexey Ya. Chervonenkis (1938-2014, ULondon)¹

¹He died of hypothermia in a park near Moscow.

Maximal Margin Classifier

(Section 9.1)

What Is a Hyperplane?

- A **hyperplane** in p dimensions is a flat affine subspace of dimension $p - 1$, e.g., a line in \mathbb{R}^2 , or a plane in \mathbb{R}^3 .
 - The word "affine" indicates that the subspace need not pass through the origin, i.e., there is an intercept.
- In general the equation for a hyperplane has the form

$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p =: \beta_0 + \beta^T X = 0,$$

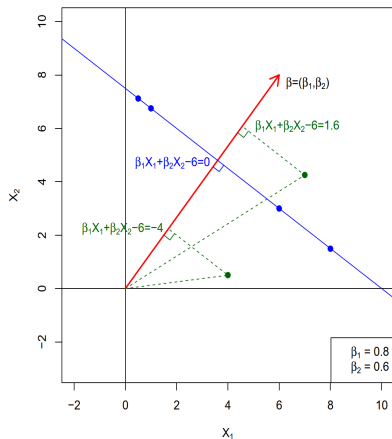
where $\beta = (\beta_1, \beta_2, \dots, \beta_p)^T$, and $X = (X_1, X_2, \dots, X_p)^T$.

- If $p = 2$, $X_2 = -\frac{\beta_0}{\beta_2} - \frac{\beta_1}{\beta_2} X_1$, which is a line going through the origin if $\beta_0 = 0$, and not through the origin otherwise.

- The vector β is called the **normal vector** – it points in a direction orthogonal to the surface of a hyperplane. [\[figure here for \$p = 2\$ \]](#)²
- The hyperplane divides the p -dimensional space into two halves, one with $\beta_0 + \beta^T X > 0$ and one with $\beta_0 + \beta^T X < 0$. [\[figure here\]](#)

^{2(**)} Why is $\beta = (\beta_1, \beta_2)^T$ orthogonal to the hyperplane in the figure? Take two points on the hyperplane, $(x_1^{(1)}, x_2^{(1)})^T$ and $(x_1^{(2)}, x_2^{(2)})^T$, whose difference, $\Delta x := (x_1^{(1)} - x_1^{(2)}, x_2^{(1)} - x_2^{(2)})^T$, is the direction of the hyperplane; since $\beta_1 x_1^{(i)} + \beta_2 x_2^{(i)} - 6 = 0$, $i = 1, 2$, taking difference of the two equations, we have $\beta_1(x_1^{(1)} - x_1^{(2)}) + \beta_2(x_1^{(1)} - x_1^{(2)}) = \beta^T \Delta x = 0$, i.e., β is orthogonal to the hyperplane.

Hyperplane in 2 Dimensions



- (**) How will you determine which half has $\beta_1 X_1 + \beta_2 X_2 - 6 > 0$ and which half has $\beta_1 X_1 + \beta_2 X_2 - 6 < 0$?

Classification Using a Separating Hyperplane

- If $f(X) = \beta_0 + \beta^T X$, then $f(X) > 0$ for points on one side of the hyperplane, and $f(X) < 0$ for points on the other.
- If we code the colored points as $y_i = +1$ for blue, say, and $y_i = -1$ for purple, and if they can be perfectly separated, then we can construct a **separating hyperplane** $f(X) = 0$ that has the property that

$$f(x_i) > 0 \text{ if } y_i = 1$$

and

$$f(x_i) < 0 \text{ if } y_i = -1.$$

[see [Figure 9.2](#)]

- In other words, a separating hyperplane has the property that

$$y_i \cdot f(x_i) > 0 \text{ for all } i.$$

- It turns out that $\frac{y_i}{\|\beta\|} f(x_i)$ is the distance of x_i to the separating hyperplane [see [the next² slide](#)], where $\|\beta\| = \sqrt{\sum_{j=1}^p \beta_j^2}$ is the Euclidean norm of β .
 - Note that β_0 is not included in β , i.e., only slopes of $f(X)$ appear.
 - If we normalize $\|\beta\| = 1$, then $y_i f(x_i)$ ($f(x_i)$) is the (signed) distance of x_i to the separating hyperplane.

Separating Hyperplane

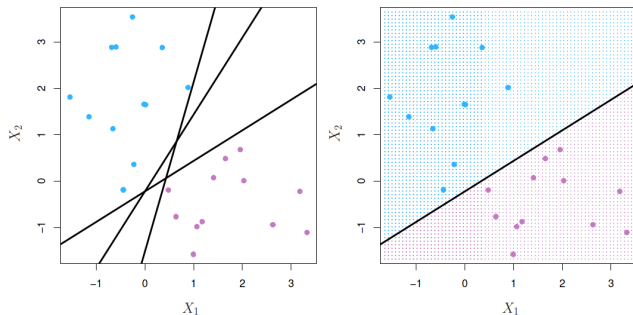


FIGURE 9.2. Left: There are two classes of observations, shown in blue and in purple, each of which has measurements on two variables. Three separating hyperplanes, out of many possible, are shown in black. Right: A separating hyperplane is shown in black. The blue and purple grid indicates the decision 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.

(**) Distance from a Point to a Hyperplane

- In the figure, for any point x_0 in the hyperplane L , $\beta^T x_0 = -\beta_0$.
- The signed distance of any point x to L is given by

$$\beta^{*T} (x - x_0) = \frac{1}{\|\beta\|} (\beta^T x - \beta_0^T x) = \frac{1}{\|\beta\|} (\beta^T x + \beta_0) = \frac{1}{\|f'(x)\|} f(x),$$

where $\beta^* = \beta / \|\beta\|$.

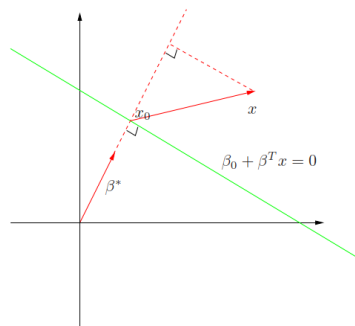


FIGURE The linear algebra of a hyperplane (affine set).

The Maximal Margin Classifier

- Among the infinite possible separating hyperplanes, find the one that makes the biggest gap or margin between the two classes.
- This result in the **maximal margin hyperplane** (aka the **optimal separating hyperplane**).
 - The maximal margin hyperplane represent the mid-line of the widest "slab" that we can insert between the two classes.³ [see [Figure 9.3](#)]
- The three observations that "support" the maximal margin hyperplane are called **support vectors**, i.e., these points were moved slightly then the maximal margin hyperplane would move as well.
 - The maximal margin hyperplane is determined **only** by the support vectors; of course, which points are the support vectors depends on all observations.
- The associated **maximal margin classifier** classifies the test observation x^* based on the sign of $f(x^*) = \beta_0 + \beta^T x^*$.

³The term "margin" refers to either the smallest distance from the observations to a separating hyperplane or the slab around a separating hyperplane.

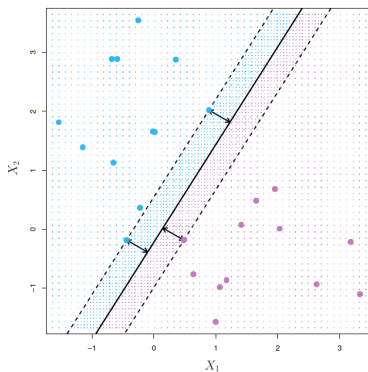


FIGURE 9.3. There are two classes of observations, shown in blue and in purple. The maximal margin hyperplane is shown as a solid line. The margin is the distance from the solid line to either of the dashed lines. The two blue points and the purple point that lie on the dashed lines are the support vectors,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

Construction of the Maximal Margin Classifier

- The maximal margin classifier solves a constrained optimization problem:

$$\begin{aligned} & \max_{\beta_0, \beta_1, \dots, \beta_p} M \\ & \text{subject to } \|\beta\| = 1, \\ & y_i (\beta_0 + \beta^T x_i) \geq M, \forall i = 1, \dots, n. \end{aligned}$$

- From our discussions above, the two constraints ensure that each observation is on the correct side of the hyperplane and at least a distance M from the hyperplane, i.e., M is the margin of the hyperplane.

The Non-separable Case

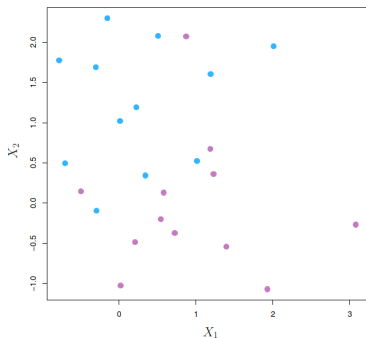


FIGURE 9.4. There are two classes of observations, shown in blue and in purple. In this case, the two classes are not separable by a hyperplane, and so the maximal margin classifier cannot be used.

- The optimization problem for the maximal margin classifier has no solution with $M > 0$.
 - This is often the case, unless $n \leq p + 1$.
- In the next section, we show how to **almost** separate the classes, using a so-called **soft margin**.

Support Vector Classifier

(Section 9.2)

Overview of the Support Vector Classifier

- Even if the data are separable, they are sometimes noisy. This can lead to a poor solution for the maximal-margin classifier – (i) a tiny margin, so less confident about whether an observation was correctly classified; (ii) sensitive to individual observations, so may have overfit the training data (especially when p is large). [see [Figure 9.5](#)]
- So we may prefer a classifying hyperplane that does **not** perfectly separate the two classes even if this could be done, in the interest of
 - Greater robustness to individual observations, and
 - Better classification of **most** of the training observations.
- Different from the maximal margin classifier where all observations are not only on the correct side of the hyperplane but also on the correct side of the margin, the **support vector classifier**, sometimes called a **soft margin classifier**, allows some observations to be on the incorrect side of the margin [see [Figure 9.6](#), left panel], or even the incorrect side of the hyperplane (which is inevitable when no separating hyperplane exists) [see [Figure 9.6](#), right panel].
 - The margin is **soft** because it can be violated by some of the training observations.

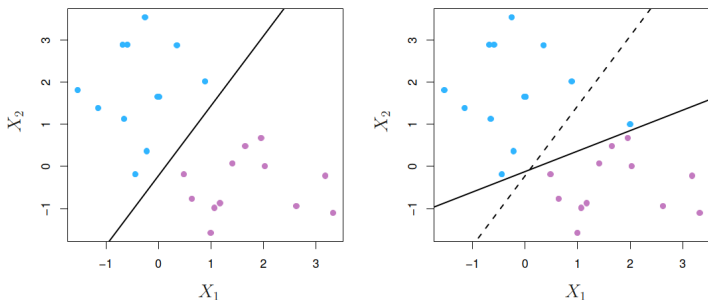


FIGURE 9.5. Left: Two classes of observations are shown in blue and in purple, along with the maximal margin hyperplane. Right: An additional blue observation has been added, leading to a dramatic shift in the maximal margin hyperplane shown as a solid line. The dashed line indicates the maximal margin hyperplane that was obtained in the absence of this additional point.

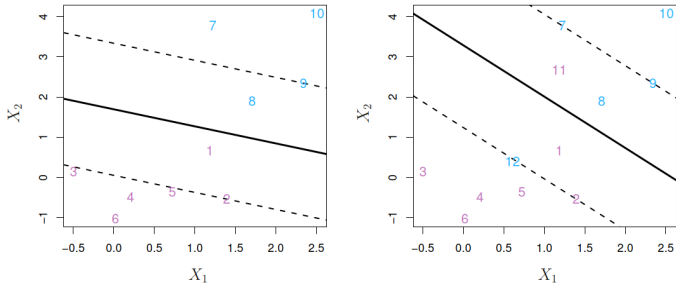


FIGURE 9.6. Left: A support vector classifier was fit to a small data set. The hyperplane is shown as a solid line and the margins are shown as dashed lines. Purple observations: Observations 3, 4, 5, and 6 are on the correct side of the margin, observation 2 is on the margin, and observation 1 is on the wrong side of the margin. Blue observations: Observations 7 and 10 are on the correct side of the margin, observation 9 is on the margin, and observation 8 is on the wrong side of the margin. No observations are on the wrong side of the hyperplane. Right: Same as left panel with two additional points, 11 and 12. These two observations are on the wrong side of the hyperplane and the wrong side of the margin.

Details of the Support Vector Classifier

- The constrained optimization problem for the support vector classifier is

$$\begin{aligned} & \max_{\beta_0, \beta_1, \dots, \beta_p, \varepsilon_1, \dots, \varepsilon_n} M \\ & \text{subject to } \|\beta\| = 1, \\ & y_i (\beta_0 + \beta^T x_i) \geq M(1 - \varepsilon_i), \\ & \varepsilon_i \geq 0, \sum_{i=1}^n \varepsilon_i \leq C. \end{aligned}$$

- M is the width of the margin, and $\varepsilon_1, \dots, \varepsilon_n$ are **slack variables**.
- $\varepsilon_i = 0$ (> 0 , > 1) implies the i th observation is on the correct side of the margin (wrong side of the margin, wrong side of the hyperplane). [\[figure here\]](#)⁴
- $C \geq 0$ is a tuning parameter: $C = 0$ implies the maximal margin hyperplane (suppose it exists), $C > 0$ is the total violations to the margin that we can tolerate, and it implies no more than C observations can be on the wrong side of the hyperplane. [\[see Figure 9.7 for the effect of decreasing \$C\$ on the margin\]](#)
 - C can be chosen by CV based on the misclassification error and controls the bias-variance trade-off [C small \Rightarrow low bias but high variance].

⁴ $M(1 - \varepsilon_i)$ rather than $M - \varepsilon_i$ is due to a technical reason.

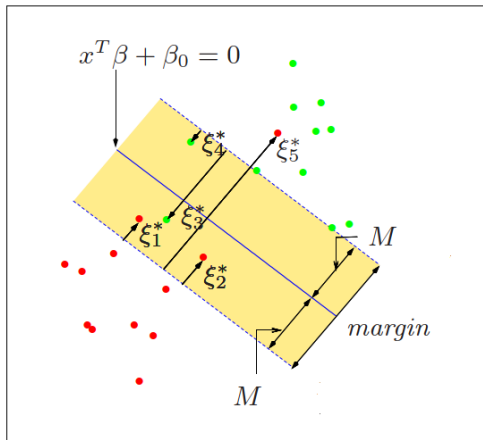


Figure: The points labeled ξ_i^* violate their margin by an amount $\xi_i^* = M\epsilon_i$; points on the correct side have $\xi_i^* = 0$. The margin is maximized subject to a total budget $\sum_{i=1}^n \epsilon_i \leq \text{constant}$. Hence $\sum_{i=1}^n \xi_i^*$ is the total distance of points violating their margin.

C as a Regularization Parameter

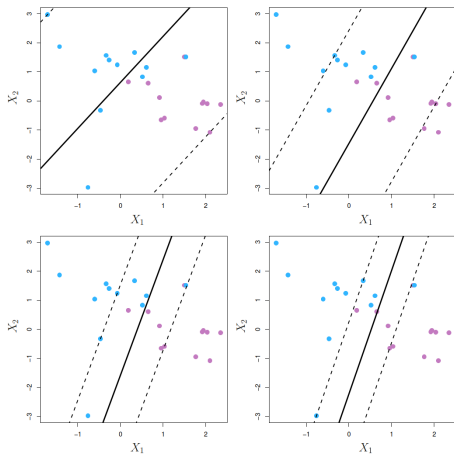


FIGURE 9.7. A support vector classifier was fit using four different values of the tuning parameter C

More Discussions on the Support Vector Classifier

- Similar to the maximal margin classifier, only the **support vectors** are relevant in determining the classifier, but here support vectors include observations that either lie on the margin or that violate the margin.
 - Note that support vectors need not stay between the two margin boundaries [$\epsilon_j > 2$].
- This implies that the support vector classifier is robust to the behaviour of observations that are far away from the hyperplane.
 - On the contrary, the classification rule in the LDA depends on all of the observations equally (through the mean of each class and the common within-class covariance matrix).
 - In this aspect, logistic regression is close to the support vector classifier; see Section 9.5 below.
- When C is large, the margin is wide, and there are many support vectors, so the classifier has low variance but high bias, and vice versa.
- Sometime a linear boundary simply won't work, no matter what value of C [see [Figure 9.8](#)]. The **support vector machine** in the next section will handle this issue.

Linear Boundary Can Fail

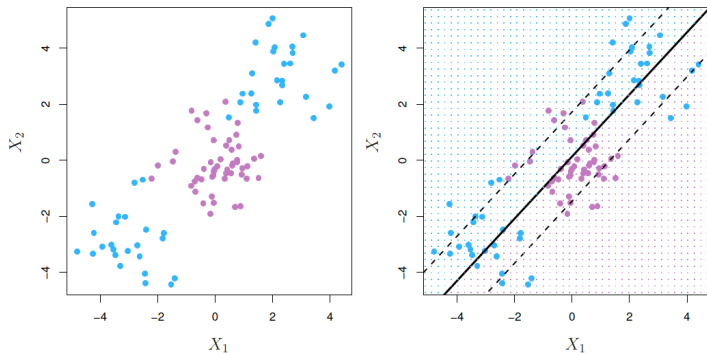


FIGURE 9.8. Left: The observations fall into two classes, with a non-linear boundary between them. Right: The support vector classifier seeks a linear boundary, and consequently performs very poorly.

Support Vector Machines

(Section 9.3)

Feature Expansion

- As in [Lecture 7](#), we can enlarge the space of features by including transformations, e.g., $X_1^2, X_1^3, X_1X_2, X_1X_2^2, \dots$. Hence go from a p -dimensional space to a $q > p$ dimensional space.
- Fit a support-vector classifier in the enlarged space.
- This results in non-linear decision boundaries in the original space.
- For example, suppose we use $(X_1, X_2, X_1^2, X_2^2, X_1X_2)$ instead of just (X_1, X_2) . Then the decision boundary would be of the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 = 0.$$

- This leads to nonlinear decision boundaries in the original space (quadratic conic sections [[figure here](#)]).
 - The support-vector classifier in the enlarged space solves the problem in the lower-dimensional space (5 vs. 2).
- There are many possible expansions and maybe end up with a huge q , which results in an unmanageable computational burden.
- The SVM will do this feature expansion in an automatic and computationally efficient way through the use of **kernels**.

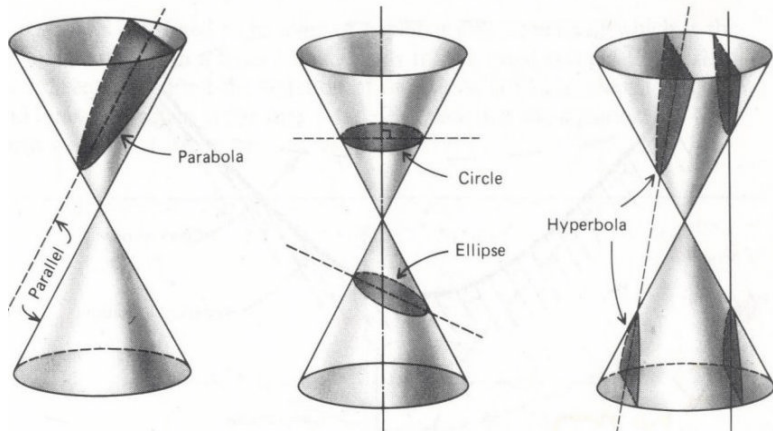


Figure: Quadratic Conic Sections

Inner Products

- We did not discuss how the support vector classifier is computed; actually this can be done by setting up the Lagrangian and solving the Kuhn-Tucker conditions.
- Anyway, it turns out that the solution to the support vector classifier problem involves only the **inner products** of the observations (as opposed to the observations themselves).
- We have met the Euclidean inner product:

$$\langle \mathbf{x}_i, \mathbf{x}_{i'} \rangle = \sum_{j=1}^p x_{ij} x_{i'j} =: \mathbf{x}_i^T \mathbf{x}_{i'}.$$

- The linear support vector classifier can be represented as

$$f(\mathbf{x}) = \beta_0 + \sum_{i=1}^n \alpha_i \langle \mathbf{x}, \mathbf{x}_i \rangle = \beta_0 + \left(\sum_{i=1}^n \alpha_i \mathbf{x}_i \right)^T \mathbf{x},$$

i.e., $\beta = \sum_{i=1}^n \alpha_i \mathbf{x}_i$.

Support Vectors

- To estimate the n parameters $\alpha_1, \dots, \alpha_n$, all we need are the C_2^n inner products $\langle \mathbf{x}_i, \mathbf{x}_{i'} \rangle$ between all pairs of training observations.
- It turns out that most of the $\hat{\alpha}_i$'s are zero:

$$\hat{f}(\mathbf{x}) = \hat{\beta}_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i \langle \mathbf{x}, \mathbf{x}_i \rangle,$$

where \mathcal{S} is the **support set** of indices i such that $\hat{\alpha}_i > 0$.

- Where** is y_i used? Actually, $\hat{\alpha}_i = \hat{a}_i y_i$ for some \hat{a}_i that is nonzero only for support vectors.
- How** about β_0 ? Any point on the margin can be used to solve for β_0 by $y_i f(\mathbf{x}_i) = 1$, and typically average all the solutions for numerical stability.
 - Why 1 here? This involves some transformations of the original problem. Specifically, $y_i (\beta_0 + \beta^T \mathbf{x}_i) \geq M(1 - \varepsilon_i)$ can be re-expressed as $y_i (b_0 + b^T \mathbf{x}_i) \geq 1 - \varepsilon_i$, where $b_0 = \beta_0 / M$, and $b = \beta / M$. Because $\|\beta\| = 1$, $\|b\| = 1 / M$. In other words, maximizing M is equivalent to minimizing $\|b\|$. Now, the margin point has $\varepsilon_i = 0$, so all points on the margin satisfy $y_i (b_0 + b^T \mathbf{x}_i) = 1$.

Kernels and SVM

- As we have seen, to get $\hat{f}(x)$, we need only compute some inner products.
- The Euclidean inner product is not the only choice; we can define the inner product through **kernels** which quantify the similarity of two observations [think about the Euclidean inner product].
 - From this perspective, the kernel should emphasize the features that are relevant for the classification when there are many features.
- Here, we provide two other popular nonlinear kernels besides the **linear** kernel $K(x_i, x_{j'}) = x_i^T x_{j'}$.
 - This kernel is a linear kernel because $\hat{f}(x)$ is linear in x ; it essentially quantifies the similarity of a pair of observations using Pearson (standard) correlation.
- When the support vector classifier is combined with a non-linear kernel, the resulting classifier is known as a **SVM**.

Polynomial Kernel

- **Polynomial Kernel** of degree d :

$$K(x_i, x_{i'}) = (1 + \langle x_i, x_{i'} \rangle)^d,$$

where d is a positive integer.

- It computes the inner-products needed for d dimensional polynomials – C_d^{p+d} (which is often written as $\binom{p+d}{d}$) basis functions!⁵ e.g., if $p = d = 2$, then $C_d^{p+d} = 6$, and

$$K(x, y) = 1 + 2x_1y_1 + 2x_2y_2 + x_1^2y_1^2 + x_2^2y_2^2 + 2x_1x_2y_1y_2 = \langle h(x), h(y) \rangle$$

with $h(x)^T = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)$.

- The solution has the form $\hat{f}(x) = \hat{\beta}_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i K(x, x_i)$ which is nonlinear in x if $d > 1$. [see [the Appendix](#)]

⁵Why? Choose d from $p + 1$ unordered with replacement.

Radial Kernel

- The **Radial (basis function) Kernel (RBF Kernel)** or **squared exponential kernel (SE kernel)** takes the form


$$K(x_i, x_{i'}) = \exp\left(-\gamma\|x_i - x_{i'}\|^2\right),$$

which is essentially a Gaussian kernel, where γ is a positive constant.

- The feature space is **implicit** and **infinite-dimensional** [Exercise], so the direct computation $\langle h(x_i), h(x_{i'}) \rangle$ is formidable if not impossible.
- Again, $\hat{f}(x) = \hat{\beta}_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i K(x, x_i)$ [see the Appendix], but $K(\cdot, \cdot)$ is the radial kernel now.
 - This classifier controls variance by squashing down most dimensions severely since most $\hat{\alpha}_i$'s are zero.

How does the Radial Kernel Work?

- Given a test observation x^* , if $\|x^* - x_j\|$ is large, then $K(x^*, x_j)$ will be tiny so that x_j will play virtually no role in $\hat{f}(x^*)$.⁶
- Recall that x^* is classified based on $\text{sign}(\hat{f}(x^*))$, so training observations far from x^* will play essentially no role in the predicted class label for x^* ; in other words, the radial kernel is "local", similar to the kernel function in local regression of [Lecture 7](#).
- The larger γ is, the more local K is, the more effective dimension the feature space has, and the more nonlinear $\hat{f}(x)$ is.

⁶Correlation and a decreasing function of Euclidean distance are two popular measures of similarity; see hierarchical clustering in [Lecture 3](#) for some closely related dissimilarity measures. 

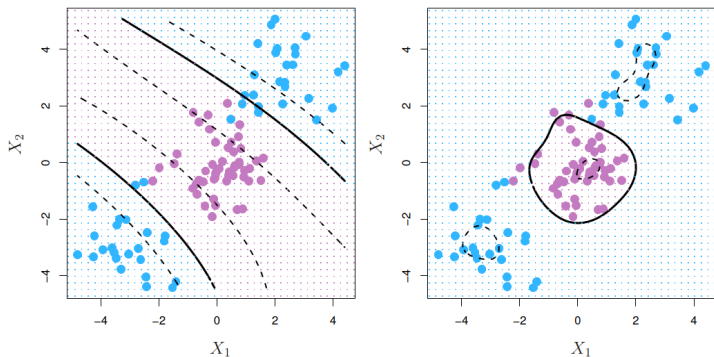


FIGURE 9.9. Left: An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule. Right: An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.

- The dashed lines are the margins, $f(x) = \pm 1$.
- Left Panel: $d = 3$, so $C_d^{p+d} = C_3^5 = 10$.⁷ Write out $f(x)$ explicitly after class.

⁷There are totally 78 cubic plane curves.

An Application to the Heart Disease Data

- $n = 303 - 6(\text{missing}) = 297$, $p = 13$, $n_{\text{train}} = 207$, and $n_{\text{test}} = 90$.
- In [Figure 9.10](#), recall that "false positive rate" is $\Pr(\text{type I error})$ and "true positive rate" is the power $= 1 - \Pr(\text{type II error})$ in testing H_0 : negative vs. H_1 : positive.
- The fp and tp rates are changing by changing the cutoff t in the decision

$$2 \cdot I(\hat{f}(X) \geq t) - 1.$$

- In the left panel, LDA and the support vector classifier perform similarly with the latter being slightly better.
- In the right panel, $\gamma = 10^{-1}$ gives an almost perfect ROC curve, but this may overfit. [see [Figure 9.11](#)]
- In the left panel of [Figure 9.11](#), the support vector classifier seems to have a small advantage over LDA (although statistically insignificant) on the test data; in the right panel, the SVM with $\gamma = 10^{-1}$ gives the worst result on the test data, and the other three classifiers perform similarly.

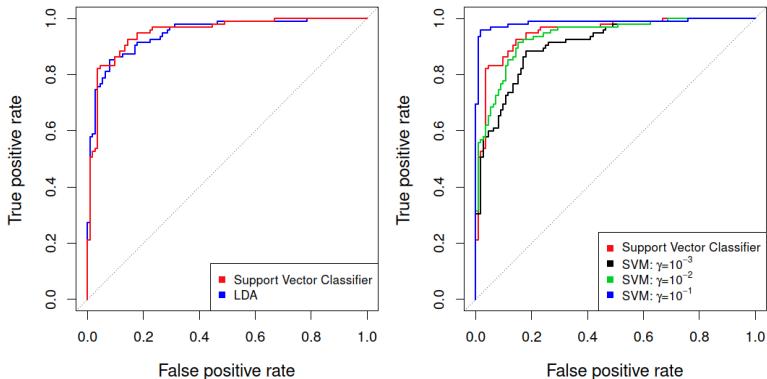


FIGURE 9.10. ROC curves for the **Heart** data training set. Left: The support vector classifier and LDA are compared. Right: The support vector classifier is compared to an SVM using a radial basis kernel with $\gamma = 10^{-3}$, 10^{-2} , and 10^{-1} .

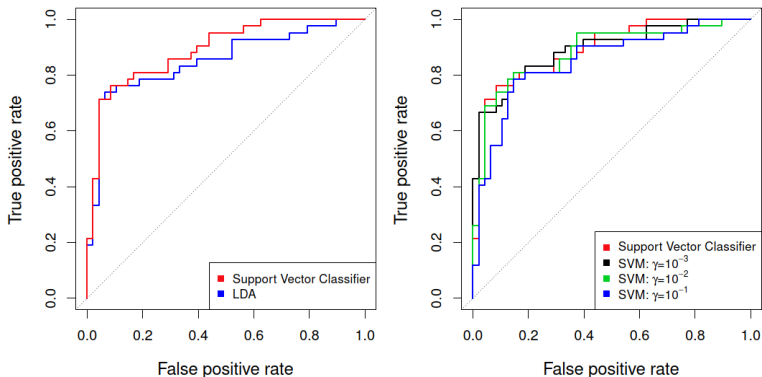


FIGURE 9.11. ROC curves for the test set of the Heart data. Left: The support vector classifier and LDA are compared. Right: The support vector classifier is compared to an SVM using a radial basis kernel with $\gamma = 10^{-3}$, 10^{-2} , and 10^{-1} .

SVMs with More than Two Classes

(Section 9.4)

SVM with More than Two Classes

- The SVM as defined works for $K = 2$ classes. What do we do if we have $K > 2$ classes?
- **OVO** (one-versus-one) or **all-pairs** approach: Fit all C_2^K pairwise classifiers $\hat{f}_{k\ell}(x)$. Classify x^* to the class that wins the most pairwise competitions.
- **OVA** (one-versus-all) approach: Fit K different 2-class SVM classifiers $\hat{f}_k(x)$, $k = 1, \dots, K$; each class versus the rest. Classify x^* to the class for which $\hat{f}_k(x^*)$ is largest, as this amounts to a high level of confidence that x^* belongs to the k th class rather than to any of the other classes.
- **Which** to choose? If K is not too large, use OVO.
- Anyway, some experiments show that OVA is as accurate as OVO, and has the merit of being conceptually simple and straightforward to implement.

(*) Relationship to Logistic Regression

(Section 9.5)

Relationship to Logistic Regression

- It turns out that one can rephrase the support vector classifier optimization when $f(X) = \beta_0 + \beta^T X$ as

$$\min_{\beta_0, \beta} \left\{ \sum_{i=1}^n \max[0, 1 - y_i f(x_i)] + \lambda \|\beta\|^2 \right\},$$

which takes the "Loss + Penalty" form, where $\lambda \geq 0$ is a tuning parameter which is **positively** related to the budget C above, and $\lambda \|\beta\|^2$ is a ridge penalty term.

- The loss function as a function of $yf(x)$, $L(y, f) = \max[0, 1 - yf] =: (1 - yf)_+$ is the positive part of $1 - yf$, known as the **hinge loss**, due to its shape.
 - Only support vectors (i.e., $yf \leq 1$) count! It is the flat part of the hinge function that gives rise to the sparsity of the SVM solution.
- In logistic regression with $y_i \in \{1, -1\}$ rather than $\{1, 0\}$, the binomial deviance (i.e., negative log-likelihood) loss is

$$\begin{aligned} L_B(y, f) &= -\{I(y = -1) \log \Pr(y = -1|x) + I(y = +1) \log \Pr(y = +1|x)\} \\ &= -\left\{ I(y = -1) \log \left(\frac{1}{1 + e^{f(x)}} \right) + I(y = +1) \log \left(\frac{e^{f(x)}}{1 + e^{f(x)}} \right) \right\} \\ &= \log \left(1 + e^{-yf(x)} \right). \text{ [see Figure 9.12]} \end{aligned}$$

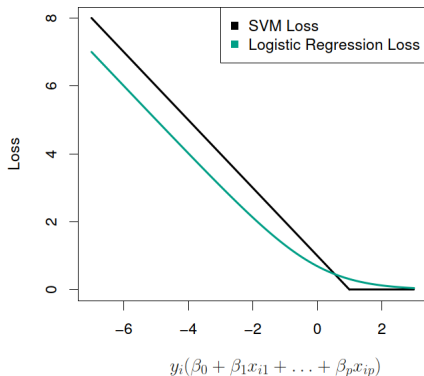
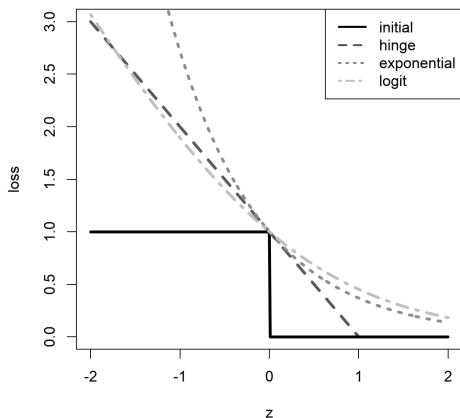


FIGURE 9.12. *The SVM and logistic regression loss functions are compared, as a function of $y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip})$. When $y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip})$ is greater than 1, then the SVM loss is zero, since this corresponds to an observation that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

Which to Use? SVM or Logistic Regression?

- When the classes are well separated, SVMs tend to behave better than logistic regression. So does LDA.
- When not, i.e., in more overlapping regimes, logistic regression (with ridge penalty) and SVM behave similarly although logistic regression is often preferred.
- If you wish to estimate probabilities, logistic regression is the choice.
 - (**) This is because the SVM does not have a probabilistic interpretation, i.e., it cannot be interpreted as a MLE or MAP.
- For nonlinear boundaries, kernel SVMs are popular. Can use kernels with logistic regression and LDA as well, but computations are more expensive (because none of the $\hat{\alpha}_j$'s is zero!).

Comparison with Other Popular Loss Functions



- Adaboost, SVM and logistic regression all provide **convex surrogates** for the misclassification error loss $I(z < 0)$ with $z = yf$.
 - This is just like using RR and lasso as convex surrogates of best subset selection to avoid computational difficulties.

Support Vector Regression

- When y is quantitative, suppose we consider the linear regression model, $E[y|x] =: f(x) = x^T \beta + \beta_0$.
- Support vector regression** machine estimates β by minimizing

$$H(\beta, \beta_0) = \sum_{i=1}^N V(y_i - f(x_i)) + \lambda \|\beta\|^2, \quad (1)$$

where $V(\cdot)$ is an " **ε -insensitive**" error measure:

$$V_{\varepsilon}(r) = (|r| - \varepsilon)_+ = \max\{0, |r| - \varepsilon\} = \begin{cases} 0, & \text{if } |r| < \varepsilon, \\ |r| - \varepsilon, & \text{otherwise,} \end{cases}$$

which ignores errors of size less than ε . [[figure here](#)]

- The ε -insensitive error function inherits some properties of the SVM classifier: the counterparts of the points well inside their class boundary are those with small residuals; these low error points do not incur loss.
- We can also extend $f(x)$ to nonlinear functions by kernels.
- Similar to SVM, $\hat{f}(x) = \hat{\beta}_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i K(x, x_i)$, where the support vectors are points for which the errors lie on or outside the ε tube of $\hat{f}(\cdot)$.

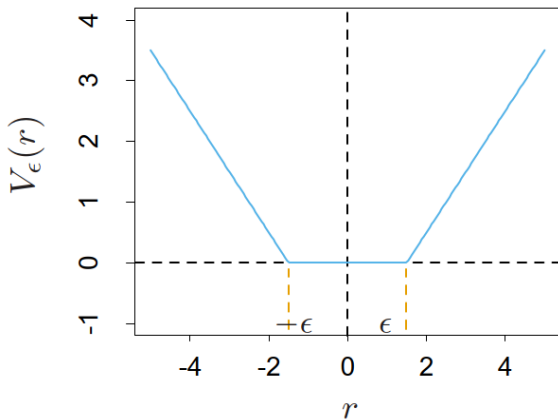


Figure: The ϵ -insensitive error function used by the support vector regression machine.

Lab: Support Vector Machines

(Section 9.6)

- Support Vector Classifier
- Support Vector Machine
- ROC Curves
- SVM with Multiple Classes

Appendix: Reproducing Kernel Hilbert Space

Kernels

- This appendix is based on Appendix E and Section 11.3 of Giraud (2021).

Definition

A function $K: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is said to be a **nonnegative definite kernel** if it is symmetric (i.e., $K(x, y) = K(y, x)$ for all $x, y \in \mathcal{X}$), and if for any $N \in \mathbb{N}$, $x_1, \dots, x_N \in \mathcal{X}$ and $a_1, \dots, a_N \in \mathbb{R}$ we have

$$\sum_{i,j=1}^N a_i a_j K(x_i, x_j) \geq 0.$$

- Some examples of nonnegative definite kernel in $\mathcal{X} = \mathbb{R}^d$:
 - linear kernel: $K(x, y) = \langle x, y \rangle$
 - Gaussian kernel: $K(x, y) = e^{-\|x-y\|^2/2\sigma^2}$
 - histogram kernel ($d = 1$ and $\mathcal{X} = [0, 1]$): $K(x, y) = \min(x, y)$
 - exponential kernel: $K(x, y) = e^{-\|x-y\|/\sigma}$

RKHS

- **Parzen's theorem:** Every nonnegative definite kernel K possesses a unique **reproducing kernel Hilbert space (RKHS)**, denoted \mathcal{H}_K , defined as follows:
 - \mathcal{H}_K is a Hilbert space with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_K}$.
 - $K(\cdot, x) \in \mathcal{H}_K$ for any $x \in \mathcal{X}$.
 - Reproducing property: $\langle K(\cdot, x), f(\cdot) \rangle_{\mathcal{H}_K} = f(x)$ for $f \in \mathcal{H}_K$ and $x \in \mathcal{X}$.
- It can be checked that the following space $(\mathcal{F}, \langle \cdot, \cdot \rangle_{\mathcal{F}})$ can serve as $(\mathcal{H}_K, \langle \cdot, \cdot \rangle_{\mathcal{H}_K})$:

$$F = \left\{ f: \mathcal{X} \rightarrow \mathbb{R} \mid f(x) = \sum_{i=1}^{\infty} a_i K(x, x_i), (x_i) \in \mathbb{R}^{\mathbb{N}}, \sum_{i,j \geq 1} a_i a_j K(x_i, x_j) < \infty \right\}$$

$$\langle f, g \rangle_{\mathcal{F}} = \sum_{i,j \geq 1} a_i b_j K(x_i, y_j) = \sum_{i=1}^{\infty} a_i g(x_i) = \sum_{i=1}^{\infty} b_j f(y_i)$$

for $f = \sum_{i=1}^{\infty} a_i K(\cdot, x_i)$ and $g = \sum_{i=1}^{\infty} b_j K(\cdot, y_j)$, where the last two equalities ensures that $\langle f, g \rangle_{\mathcal{F}}$ does not depend on the choice of the expansion of f and g .

- Because

$$|f(x) - f(x')| = \left| \langle f, K(\cdot, x) \rangle_{\mathcal{H}_K} - \langle f, K(\cdot, x') \rangle_{\mathcal{H}_K} \right| \leq \|f\|_{\mathcal{H}_K} \|K(\cdot, x) - K(\cdot, x')\|_{\mathcal{H}_K},$$

the smoothness of a function in the RKHS is driven by its norm.

More Discussions

- Let's reverse the discussion in the last slide, i.e., first define a functional on f , and then check how the kernel is deduced.
- For any Hilbert space \mathcal{H} of functions on a set \mathcal{X} , we may define for each $x \in \mathcal{X}$ the evaluation functional $f \mapsto f(x)$.
- If every such evaluation functional is bounded, then by the Riesz representation theorem, we can find for each x an $R_x \in \mathcal{H}$ so that $f(x) = \langle f(\cdot), R_x(\cdot) \rangle_{\mathcal{H}}$, i.e., \mathcal{H} is a RKHS.

• The kernel $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ can be defined as $K(x, y) = R_x(y)$.

• K is **symmetric**: $K(x, y) = R_x(y) = \langle R_x(\cdot), R_y(\cdot) \rangle = \langle R_y(\cdot), R_x(\cdot) \rangle = R_y(x) = K(y, x)$.

• K is **nonnegative definite**: $\sum_{i,j=1}^N a_i a_j K(x_i, x_j) = \langle \sum_{i=1}^N a_i R_{x_i}(\cdot), \sum_{i=1}^N a_i R_{x_i}(\cdot) \rangle = \left\| \sum_{i=1}^N a_i R_{x_i}(\cdot) \right\|^2 \geq 0$.

• K is **unique**: if K and K' are both kernels, then for each $x \in \mathcal{X}$,

$$\begin{aligned} \|K(\cdot, x) - K'(\cdot, x)\|^2 &= \langle K(\cdot, x) - K'(\cdot, x), K(\cdot, x) - K'(\cdot, x) \rangle \\ &= \langle K(\cdot, x) - K'(\cdot, x), K(\cdot, x) \rangle - \langle K(\cdot, x) - K'(\cdot, x), K'(\cdot, x) \rangle \\ &= (K(x, x) - K'(x, x)) - (K(x, x) - K'(x, x)) = 0, \end{aligned}$$

where the second to last equality uses the reproducing property.

SVM with Kernel Bases

- The objective function here is

$$\min_{f \in \mathcal{C}} \left\{ \sum_{i=1}^n [1 - y_i f(x_i)]_+ \right\},$$

where $\mathcal{C} = \{f \in \mathcal{F} \mid \|f\|_{\mathcal{F}} \leq R\}$ which neglects the constant β_0 for simplicity, \mathcal{F} is a RKHS, and $\|\cdot\|_{\mathcal{F}}$ is the associated norm.

- Since the smoothness of a function in an RKHS is driven by its norm, the ball \mathcal{C} corresponds to a set of smooth functions.

- The dual Lagrangian problem is

$$\min_{f \in \mathcal{F}} \left\{ \sum_{i=1}^n [1 - y_i f(x_i)]_+ + \lambda \|f\|_{\mathcal{F}}^2 \right\}. \quad (2)$$

- It turns out that the solution of (2) fulfills a representation formula:

$$\hat{f} = \sum_{j=1}^n \hat{\alpha}_j K(\cdot, x_j), \quad (3)$$

with

$$\hat{\alpha} = \arg \min_{\alpha \in \mathbb{R}^n} \left\{ \sum_{i=1}^n \left[1 - \sum_{j=1}^n \alpha_j y_i K(x_i, x_j) \right]_+ + \lambda \sum_{i,j=1}^n \alpha_i \alpha_j K(x_i, x_j) \right\}.$$

- This formula reduces the infinite-dimensional minimization problem to an n -dim convex minimization problem, but we must narrow the space of candidate functions f from the space of all functions on \mathcal{X} to the set \mathcal{C} in the RKHS \mathcal{F} .

Proof

Proof.

Let V be the linear space spanned by $\{K(\cdot, x_j)\}_{j=1}^n$. Decomposing $f = f_V + f_{V^\perp}$, where f_V is the orthogonal projection of f on V , we have by the reproducing property,

$$f(x_i) = \langle f, k(\cdot, x_i) \rangle = \langle f_V, k(\cdot, x_i) \rangle = f_V(x_i).$$

Combining this formula with the Pythagorean formula, we obtain

$$\sum_{i=1}^n [1 - y_i f(x_i)]_+ + \lambda \|f\|_{\mathcal{F}}^2 = \sum_{i=1}^n [1 - y_i f_V(x_i)]_+ + \lambda \|f_V\|_{\mathcal{F}}^2 + \lambda \|f_{V^\perp}\|_{\mathcal{F}}^2.$$

Since $\lambda > 0$, any minimizer \hat{f} must have $\hat{f}_{V^\perp} = 0$, so it is of the form

$$\hat{f} = \sum_{j=1}^n \hat{\alpha}_j K(\cdot, x_j).$$

Furthermore, the reproducing property ensures that $\langle k(\cdot, x_j), k(\cdot, x_i) \rangle = k(x_i, x_j)$, so

$$\|f_V\|_{\mathcal{F}}^2 = \sum_{i,j=1}^n \alpha_i \alpha_j K(x_i, x_j).$$



Further Simplification for the Hinge Loss

- The representation formula actually holds for any loss function; for the hinge loss, we have further simplification.
- It turns out that

$$\begin{aligned}\hat{\alpha}_i &= 0 \text{ if } y_i \hat{f}(x_i) > 1; \\ \hat{\alpha}_i &= \frac{y_i}{2\lambda} \text{ if } y_i \hat{f}(x_i) < 1; \\ 0 &\leq y_i \hat{\alpha}_i \leq \frac{1}{2\lambda} \text{ if } y_i \hat{f}(x_i) = 1;\end{aligned}$$

so only the support vectors count in determining \hat{f} .

- **Why?** Denote K for the symmetric matrix $[K(x_i, x_j)]_{i,j=1,\dots,n}$; then

$$\hat{\alpha} = \arg \min_{\alpha \in \mathbb{R}^n} \left\{ \sum_{i=1}^n [1 - y_i [K\alpha]_i]_+ + \lambda \alpha^T K \alpha \right\}.$$

This objective function is not smooth, so we introduce some slack variables $\xi_j = (1 - y_j [K\alpha]_j)_+$ and rewrite the minimization problem as

$$(\hat{\alpha}, \hat{\xi}) = \arg \min_{\alpha, \xi \in \mathbb{R}^n \text{ s.t. } \xi_j \geq 1 - y_j [K\alpha]_j, \xi_j \geq 0} \left\{ \sum_{i=1}^n \xi_i + \lambda \alpha^T K \alpha \right\},$$

which is now smooth and convex.

Continued

- The Kuhn-Tucker conditions for the Lagrangian dual problem

$$(\hat{\alpha}, \hat{\xi}) = \arg \min_{\alpha, \xi \in \mathbb{R}^n} \left\{ \sum_{i=1}^n \xi_i + \lambda \alpha^T K \alpha - \sum_{i=1}^n [\eta_i (\xi_i - 1 + y_i [K\alpha]_i) + \gamma_i \xi_i] \right\}$$

gives the formula for $i, j = 1, \dots, n$

$$\text{FOCs} \quad : \quad 2\lambda [K\hat{\alpha}]_j = \sum_{i=1}^n K_{ij} \eta_i y_i \text{ and } \eta_i + \gamma_i = 1,$$

$$\text{Slackness Conditions} \quad : \quad \min(\eta_i, \hat{\xi}_i - 1 + y_i [K\hat{\alpha}]_i) = 0 \text{ and } \min(\gamma_i, \hat{\xi}_i) = 0.$$

- The FOC is fulfilled with $\hat{\alpha}_i = \eta_i y_i / (2\lambda)$. Since $\hat{f}(x_i) = [K\hat{\alpha}]_i$, the first slackness condition enforces that $\hat{\alpha}_i = 0$ if $y_i \hat{f}(x_i) > 1$. The second slackness condition, together with the second FOC, enforces that $\hat{\alpha}_i = y_i / (2\lambda)$ if $\hat{\xi}_i > 0$ and $0 \leq y_i \hat{\alpha}_i \leq 1 / (2\lambda)$ otherwise.
- When $\hat{\xi}_i > 0$, we have $\hat{\alpha}_i$ and η_i nonzero, and therefore $y_i \hat{f}(x_i) = 1 - \hat{\xi}_i < 1$ according to the first slackness condition.

Geometrical Interpretation

- Denote $\phi : \mathcal{X} \rightarrow \mathcal{F}$ for the map $\phi(x) = K(\cdot, x)$, usually called the "feature map".
- By the reproducing property, we have

$$\hat{f}(x) = \langle \hat{f}, \phi(x) \rangle = \left\langle \sum_{j=1}^n \hat{\alpha}_j \phi(x_j), \phi(x) \right\rangle.$$

- A point $x \in \mathcal{X}$ is classified by $\text{sign}(\hat{f}(x))$, therefore, the function $\phi(x) \in \mathcal{F}$ is classified according to the linear classifier on \mathcal{F}

$$g \mapsto \text{sign}(\langle \hat{w}_\phi, g \rangle) \text{ where } \hat{w}_\phi = \sum_{j=1}^n \hat{\alpha}_j \phi(x_j).$$

- The separating frontier $\{x \in \mathcal{X} \mid \hat{f}(x) = 0\}$ is the reciprocal image by ϕ of the hyperplane $\{f \in \mathcal{F} \mid \langle \hat{w}_\phi, f \rangle = 0\}$ in \mathcal{F} , which is nonlinear in general. [\[figure here\]](#)
- For all the discussions above, we need only know the kernel K and need not identify the RKHS associated with K ; such a property is referred to as the "kernel trick".

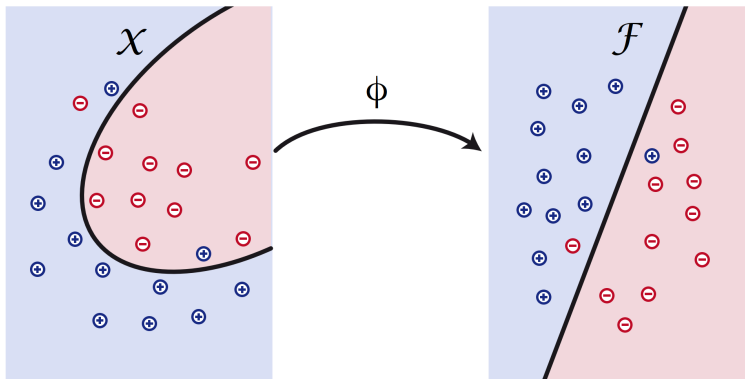


Figure: Classification with a nonlinear kernel: The linear classification in \mathcal{F} produces a non-linear classification in \mathcal{X} via the reciprocal image of ϕ

Smoothing Splines

- Recall that the objective function for smoothing splines is

$$\sum_{i=1}^n L(y_i, f(x_i)) + \lambda J(f) \quad (4)$$

where $L(y_i, f(x_i)) = (y_i - f(x_i))^2$, $J(f) = \int f''(t)^2 dt$, and WLOG, let $\mathcal{X} = [0, 1]$ so $x_i \in [0, 1]$.

- Our goal is to construct an RKHS whose norm corresponds to this J .
- Recall that Taylor's Theorem with integral remainder term states that if f' is absolutely continuous on $[0, 1]$ and $f'' \in L^2[0, 1]$, then

$$f(t) = f(0) + f'(0)t + \int_0^t (t-x)f''(x) dx. \quad (5)$$

- It will be helpful to rewrite the integral as $\int_0^1 (t-x)_+ f''(x) dx$.
- If we define W_2^0 to be the functions under the hypotheses of (5) with $f(0) = f'(0) = 0$, then for $f \in W_2^0$, we have

$$f(t) = \int_0^1 (t-x)_+ f''(x) dx, \quad (6)$$

where the subscript 2 in W_2^0 represents the 2nd derivative, and the superscript 0 means all derivatives less than 2 are 0.

W_2^0 and \mathcal{H}_0 as RKHS

- With the inner product

$$\langle f, g \rangle = \int_0^1 f''(x) g''(x) dx,$$

and the kernel

$$K^1(s, t) = \int_0^1 (s-x)_+ (t-x)_+ dx,$$

the space W_2^0 forms an RKHS.

- To see that K^1 is a kernel, first notice that $K_t^{1''}(s) = (t-s)_+$ from (6).
- Then

$$\langle f, K^1(\cdot, t) \rangle = \int_0^1 f''(x) (t-x)_+ dx = f(t).$$

- Now define the nullspace of the penalty functional:

$$\mathcal{H}_0 = \text{span}(\{1, x\}) =: \text{span}(\{\phi_1(x), \phi_2(x)\}) =: \text{span}(\phi(x)).$$

- The kernel for \mathcal{H}_0 is

$$K^0(s, t) = 1 + st,$$

which is the Euclidean inner product of $(1, s)$ and $(1, t)$.

Solution for a General Loss

- It is not hard to see that the space W_2 of functions under the hypotheses of (5) can be written as a direct sum

$$W_2 = \mathcal{H}_0 \oplus W_2^0$$

with kernel $K = K^0 + K^1$.

- Also, $J(f)$ corresponds to the squared norm of the projection Pf of f onto W_2^0 , so for $f \in W_2$ (4) becomes

$$\sum_{i=1}^n L(y_i, f(x_i)) + \lambda J(Pf). \quad (7)$$

- Following the proof of (3), it is easy to prove that the solution to (7) is the natural generalization of (3):

$$f_\lambda(x) = \sum_{i=1}^n c_i K^1(x, x_i) + (d_1 + d_2 x) =: K^1(x, \mathbf{X})^T \mathbf{c} + \phi(x)^T \mathbf{d}. \quad (8)$$

- In other words, f_λ consists of an unpenalized component from \mathcal{H}_0 as well as a linear combination of the projections onto W_2^0 of the representers of evaluation at the n input data points.

Solution for the Squared-Error Loss

- First,

$$K^1(s, t) = \int_0^1 (s-x)_+ (t-x)_+ dx = \begin{cases} \frac{1}{2}ts^2 - \frac{1}{6}s^3, & \text{if } s \leq t, \\ \frac{1}{2}st^2 - \frac{1}{6}t^3, & \text{if } t \leq s, \end{cases}$$

which is a cubic in s on $[0, t]$ and linear on $[t, 1]$ with continuous first and second derivatives.

- Then (8) differs from the natural cubic spline only in that the latter is required be linear on the interval $[0, x_1]$, which would be shown below.
- Define the $n \times 2$ matrix $\mathbf{T} = (\phi_j(x_i))$ and the $n \times n$ matrix $\mathbf{S} = (K^1(x_i, x_j))$.
- Then in matrix notation, (4), with (8) plugged in, becomes

$$\|\mathbf{y} - (\mathbf{S}\mathbf{c} + \mathbf{T}\mathbf{d})\|^2 + \lambda \mathbf{c}^T \mathbf{S} \mathbf{c}. \quad (9)$$

- Taking derivatives to minimize (9), we find that

$$\mathbf{c} = \mathbf{M}^{-1} \left(\mathbf{I} - \mathbf{T} (\mathbf{T}^T \mathbf{M}^{-1} \mathbf{T})^{-1} \mathbf{T}^T \mathbf{M}^{-1} \right) \mathbf{y},$$

where $\mathbf{M} = \mathbf{S} + \lambda \mathbf{I}$. Therefore, $\mathbf{T}^T \mathbf{c} = \mathbf{0}$, from which we obtain $\sum_{i=1}^n c_i = 0$ and $\sum_{i=1}^n c_i x_i = 0$.

Continued

- Therefore, for $t \in [0, x_1]$, we have

$$\begin{aligned} Pf_\lambda(t) &= \sum_{i=1}^n c_i K^1(t, x_i) = \sum_{i=1}^n c_i \int_0^1 (t-x)_+ (x_i-x)_+ dx \\ &= \sum_{i=1}^n c_i \int_0^t (t-x)(x_i-x) dx = \int_0^t (t-x) \sum_{i=1}^n c_i (x_i-x) dx = 0, \end{aligned}$$

where the first equality in the second line is because for $x \leq t$, $x_i \geq x$.

- As a result, only the $d_1 + d_2x$ term appears in $Pf_\lambda(t)$ for $t \in [0, x_1]$.
- We can re-organize the terms of f_λ in (8) as

$$f_\lambda(x) = \sum_{j=1}^n N_j(x) \theta_j,$$

where the $N_j(x)$ are the basis functions of natural cubic splines.

Continued

- Then the objective function reduces to

$$\text{PRSS}(\theta) = (\mathbf{y} - \mathbf{N}\theta)^T (\mathbf{y} - \mathbf{N}\theta) + \lambda \theta^T \Omega_n \theta,$$

where $\mathbf{N} = (N_j(x_i))$ is full rank, and $\Omega_n = \left(\int N_i''(t) N_j''(t) dt \right)$ whose first two rows and columns are zeros so is not full rank.

- The solution is easily seen to be

$$\hat{\theta} = \left(\mathbf{N}^T \mathbf{N} + \lambda \Omega_n \right)^{-1} \mathbf{N}^T \mathbf{y},$$

a generalized RR.

- The fitted smoothing spline is given by

$$\hat{f}(x) = \sum_{j=1}^n N_j(x) \hat{\theta}_j.$$

- The vector of n fitted values can be written as

$$\hat{\mathbf{g}}_\lambda = \mathbf{N} \left(\mathbf{N}^T \mathbf{N} + \lambda \Omega_n \right)^{-1} \mathbf{N}^T \mathbf{y} = \left(\mathbf{I} + \lambda \left(\mathbf{N}^T \right)^{-1} \Omega_n \mathbf{N}^{-1} \right)^{-1} \mathbf{y} := \mathbf{S}_\lambda \mathbf{y}.$$

Smoothing Spline as a Local Smoother

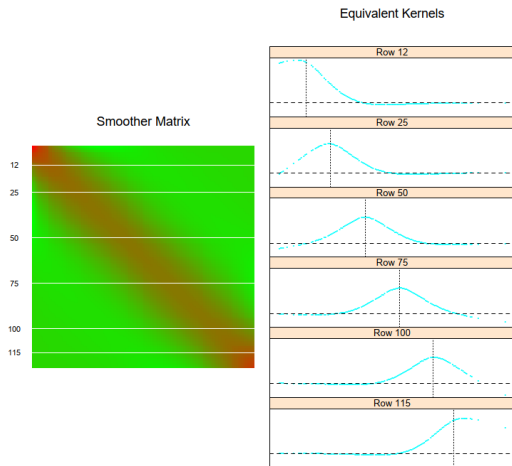


FIGURE 5.8. The smoother matrix for a smoothing spline is nearly banded, indicating an equivalent kernel with local support. The left panel represents the elements of S as an image. The right panel shows the equivalent kernel or weighting function in detail for the indicated rows.