Support Vector Machines

Machine Learning

Prof. Neylson Crepalde

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

A abordagem conhecida como *support vector machines* foi desenvolvida na comunidade da ciência da computação nos anos 1990's e tem crescido muito em popularidade.

Em um espaço *p-dimensional,* um **hiperplano** é um subespaço *flat* que não necessariamente precisa passar pela origem de dimensões p - 1. Por exemplo, num espaço de duas dimensões, o hiperplano será um subespaço *flat* de uma dimensão, isto é, uma linha. Num espaço de 3 dimensões, o hiperplano será um subespaço *flat* de duas dimensões, ou seja, um plano.

Num espaço de p > 3 dimensões fica difícil visualizarmos o hiperplano mas a noção de um subespaço de dimensões p - 1 ainda se aplica.

A definição matemática do hiperplano é simples. Em duas dimensões, o hiperplano é definido pela equação

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

Quando dizemos que essa equação "define" o hiperplano, queremos dizer que qualquer $X = (X_1, X_2)^T$ para a qual a equação acima seja verdadeira é um ponto do hiperplano.

A equação do hiperplano de 1 dimensão é simplesmente a equação de uma linha. Desse modo, essa equação pode ser facilmente estendida para um contexto *p-dimensional*.

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

Da mesma maneira, dizemos que se um ponto X = $(X_1, X_2, \ldots, X_p)^T$ em um espaço *p-dimensional* (um vetor de tamanho *p*) e satisfaz a equação acima, então o ponto X está no hiperplano.

Se X não satisfaz a equação acima de modo que

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0$$

isso nos mostra que X está de um dos lados do hiperplano. Por outro lado se

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0,$$

então o ponto X está do outro lado do hiperplano. Podemos então pensar no hiperplano como uma divisão do espaço *p-dimensional* em duas metades.

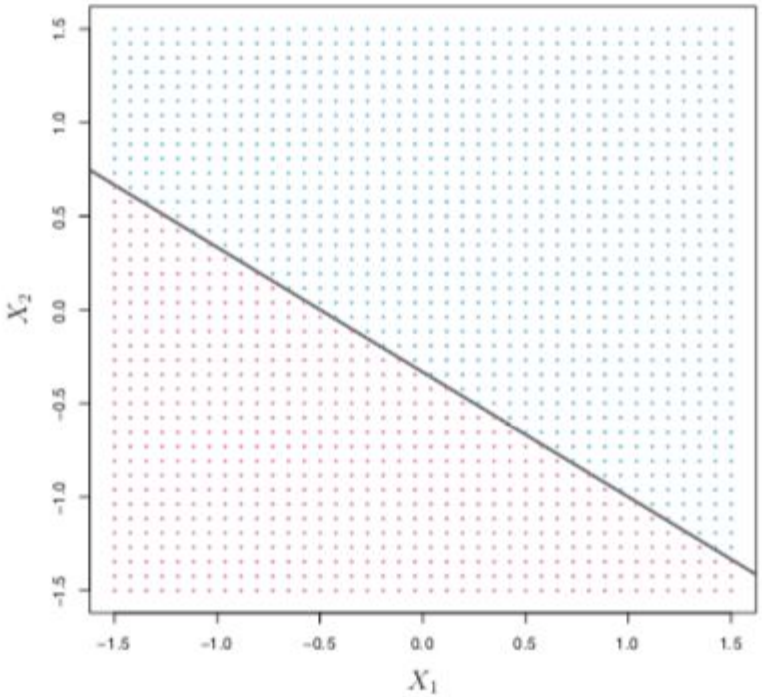


FIGURE 9.1. *The hyperplane* $1 + 2X_1 + 3X_2 = 0$ *is shown. The blue region is the set of points for which* $1 + 2X_1 + 3X_2 > 0$, *and the purple region is the set of points for which* $1 + 2X_1 + 3X_2 < 0$.

Imagine agora que temos uma matrix *n* X *p* que consiste de n observações e p preditores ou colunas.

$$x_1 = \begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix}, \ldots, x_n = \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix}$$

e que essas observações pertencem a duas classes {-1, 1}. Temos também uma observação de teste, um vetor tamanho *p* de preditores observados

$$x^* = \begin{pmatrix} x_1^* & \ldots & x_p^* \end{pmatrix}^T$$

Nosso objetivo é desenvolver um classificador baseado nos dados de treino que vai classificar corretamente a observação de teste dados os *p* preditores. Existem várias abordagens para essa tarefa (Regressão logística, LDA, *decision trees, Random Forests,* etc.). Vamos ver como executar essa tarefa com o hiperplano.

Suponha que possamos construir um hiperplano que separe perfeitamente as observações de treino perfeitamente de acordo com suas classes. Aqui estão exemplos de 3 hiperplanos que fazem esse serviço:

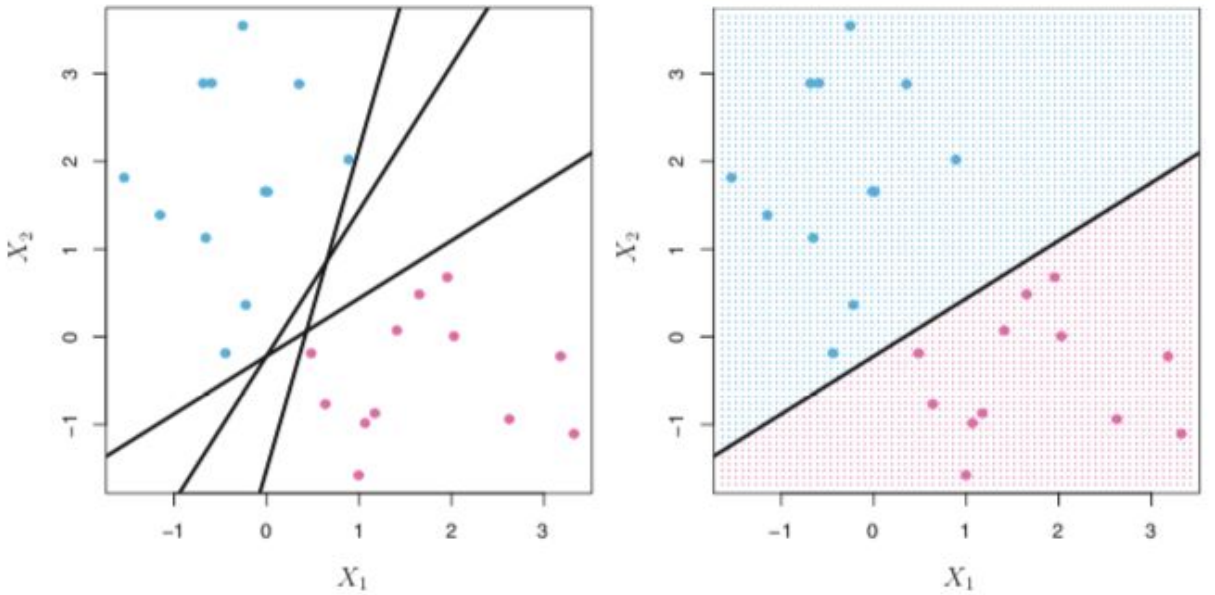


FIGURE 9.2. Left: *There are two classes of observations, shown in blue and in purple, each of which has measurements on two variables. Three separating hyperplanes, out of many possible, are shown in black.* Right: *A separating hyperplane is shown in black. The blue and purple grid indicates the decision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

Se o hiperplano separador existe, podemos usá-lo para construir um classificador de maneira bem simples: a observação de teste é classificada de acordo com a classe do lado que ela ficou posicionada em relação ao hiperplano.

Na verdade, se as classes podem ser perfeitamente separadas por um hiperplano, teremos um número infinito de possíveis hiperplanos. Precisamos, portanto, de um método razoável para decidir qual hiperplano usar. O ***Maximal Margin Classifier*** (também conhecido como *optimal separating hyperplane*) é uma escolha natural. Ele escolhe o hiperplano que está posicionado o mais longe possível das observações de treino. Isto é, podemos calcular uma distância perpendicular de cada observação para um dado hiperplano; a menor distância entre as observações e o hiperplano é chamada de *margin*. O *maximal margin hyperplane* é o hiperplano separador para o qual a *margin* é a maior possível, isto é, que tem a maior distância mínima de todas as observações de treino.

Embora o *maximal margin classifier* tem normalmente bons resultados, ele pode gerar *overfitting* se *p* for muito grande.

Podemos observar na figura ao lado 3 observações de treino que são equidistantes do hiperplano de máxima margem. Essas 3 observações são conhecidas como ***support vectors*** visto que elas são vetores no espaço *p-dimensional* e dão "suporte, apoio" ao *maximal margin hyperplane* no sentido de que se esses pontos se mexessem um pouco, o hiperplano se mexeria também.

É interessante notar que, na prática, o *maximal margin hyperplane* precisa um subset curto de observações.

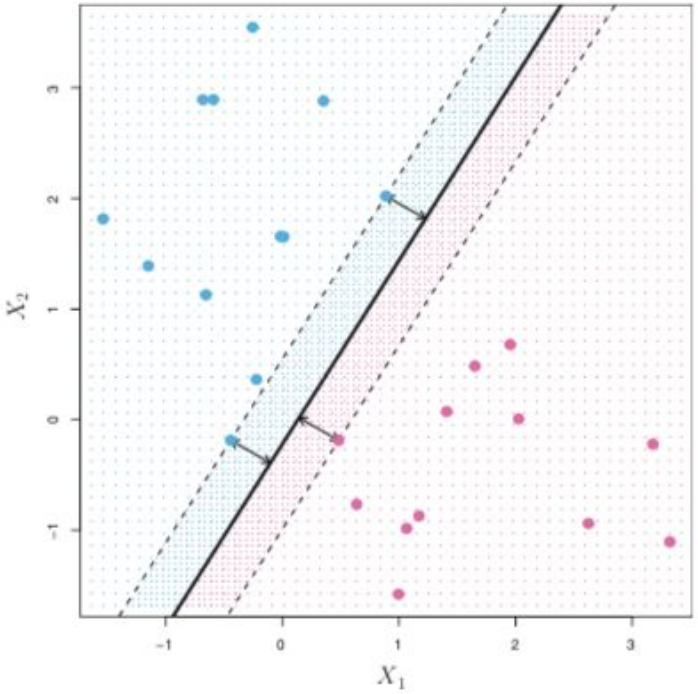
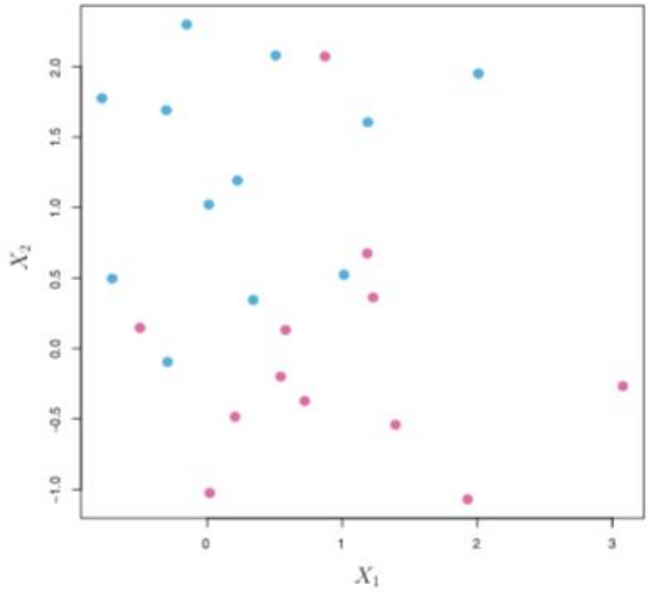


FIGURE 9.3. *There are two classes of observations, shown in blue and in purple. The maximal margin hyperplane is shown as a solid line. The margin is the distance from the solid line to either of the dashed lines. The two blue points and the purple point that lie on the dashed lines are the support 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.*

Ciências de Dados

O *maximal margin classifier* é uma forma muito natural de separar classes **se um hiperplano separador existe de fato**. Na grande maioria dos casos, as classes não são perfeitamente separáveis e, portanto, não há *maximal margin classifier*. Nós podemos estender o conceito de hiperplano separador para um hiperplano que **quase** separa as classes conhecido como ***soft margin*.** A generalização dos *maximal margin classifier* para o caso não complementamente separável é chamado ***support vector classifier*.**

O *support vector classifier* visa ser mais robusto aos casos individuais e ter uma melhor classificação *na maioria* das observações de treino. Esse método permite que algumas observações estejam do lado errado na margem ou às vezes do lado errado do hiperplano.



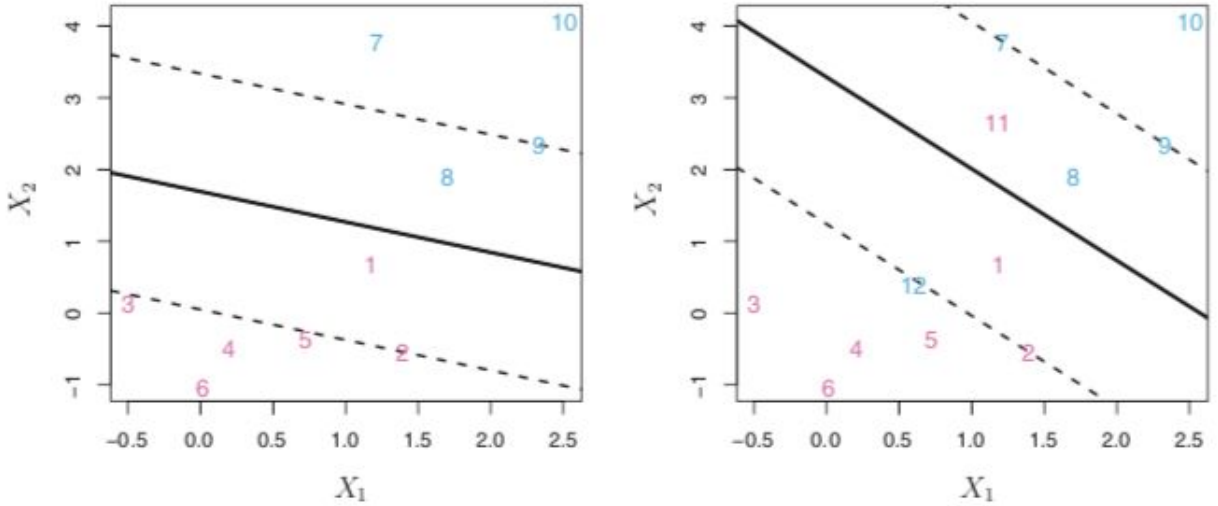


FIGURE 9.6. Left: *A support vector classifier was fit to a small data set. The hyperplane is shown as a solid line and the margins are shown as dashed lines.* Purple observations: *Observations* 3, 4, 5, *and* 6 *are on the correct side of the margin, observation* 2 *is on the margin, and observation 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.*

O *support vector classifier* classifica um uma observação de teste dependendo de qual lado do hiperplano ela está. O hiperplano é escolhido de modo que ele separe corretamente *a maioria* das observações de treino. Ele é a solução para o problema de otimização:

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} \; M$$

$$\text{subject to} \; \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i),$$

$$\epsilon_i \geq 0, \; \sum_{i=1}^{n} \epsilon_i \leq C,$$

onde *C* é um *tunning parameter* não negativo. M é a largura da margem. Queremos manter esse número tão grande quanto possível. $\epsilon_1, \ldots, \epsilon_n$ são *slack variables* que permitem as observações estarem do lado errado da margem ou do hiperplano.

O parâmetro *C* determina o número e a severidade das violações à margem e ao hiperplano que serão toleradas. Se $C = 0$, nenhuma violação é permitida e o classificador cai no problema do *maximal margin classifier*. Para $C > 0$, não mais do que *C* observações podem estar do lado errado do hiperplano. À medida que *C* aumenta, ficamos mais tolerantes a erros. *C* é normalmente escolhido via cross-validation.

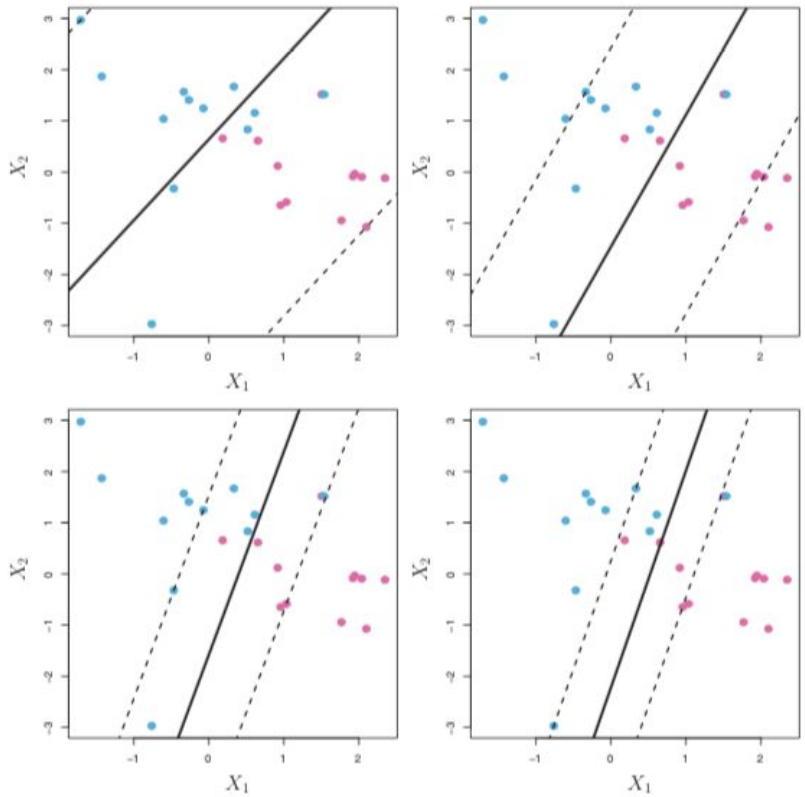


FIGURE 9.7. *A support vector classifier was fit using four different values of the tuning parameter C in (9.12)–(9.15). The largest value of C was used in the top left panel, and smaller values were used in the top right, bottom left, and bottom right panels. When C is large, then there is a high tolerance for observations being on the wrong side of the margin, and so the margin will be large. As C decreases, the tolerance for observations being on the wrong side of the margin decreases, and the margin narrows.*

O fato de que a regra de decisão do *support vector classifier* é baseada apenas em um potencial subset de observações (o *support vector*) significa que o método é bastante robusto quanto ao comportamento de observações que estão longe do hiperplano. Essa propriedade é bem diferente de outros métodos de classificação.

O *support vector classifier* é uma escolha natural para classificação binária e se os limites entre as classes é linear. Entretanto, na prática, é comum nos deparamos com divisões de classes num plano que não são lineares. Nesse caso, *support vector classifier* ou qualquer outro classificador linear terá uma má performance.

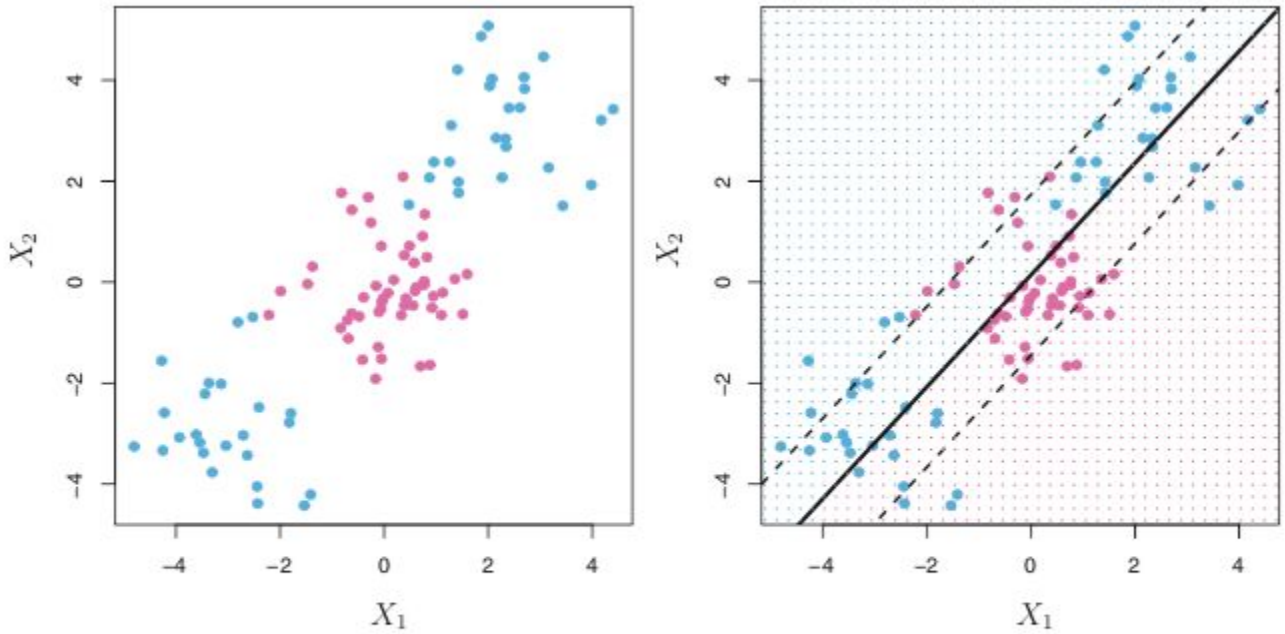


FIGURE 9.8. Left: *The observations fall into two classes, with a non-linear boundary between them.* Right: *The support vector classifier seeks a linear boundary, and consequently performs very poorly.*

Precisamos de um mecanismo que trace limites não lineares entre classes. As ***Support Vector Machines*** fazem isso de uma maneira automatizada. Para isso, *support vector machines* aumentam o espaço dimensional utilizando funções quadráticas, cúbicas ou de grau maior nos preditores. Por exemplo, ao invés de estimar um *support vector classifier* usando *p* preditores:

$$X_1, X_2, \ldots, X_p,$$

podemos estimar um *support vector classifier* usando *2p* preditores:

$$X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2.$$

A função de otimização tornaria-se, portanto,

$$\underset{\beta_0,\beta_{11},\beta_{12},\ldots,\beta_{p1},\beta_{p2},\epsilon_1,\ldots,\epsilon_n,M}{\text{maximize}} M$$

$$\text{subject to } y_i \left(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} x_{ij}^2 \right) \geq M(1 - \epsilon_i)$$

$$\sum_{i=1}^{n} \epsilon_i \leq C, \quad \epsilon_i \geq 0, \quad \sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1.$$

Obviamente, poderíamos aumentar indefinidamente o espaço de features acrescentando polinomiais o que causaria um custo computacional muito alto. ***Support vector machines*** faz isso de uma maneira computacionalmente eficiente usando ***kernels***. Em linhas muito gerais, um *kernel* é uma função que quantifica a similaridade de duas observações. Os *kernels* podem ser lineares ou polinomiais de grau *d*.

Quando o *support vector classifier* é combinado com com um *kernel* não linear, o classificador resultante é conhecido como ***support vector machine*.**

A função de classificação pode tomar a seguinte forma:

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i)$$

onde *K* é um *kernel* polinomial

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

Outro *kernel* bastante usado é o *radial kernel*:

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

onde *gama* é uma constante positiva.

A utilização dos *kernels* apena computa o *kernels* para todos os pares distintos de observações sem ter que de fato aument o espaço de preditores.

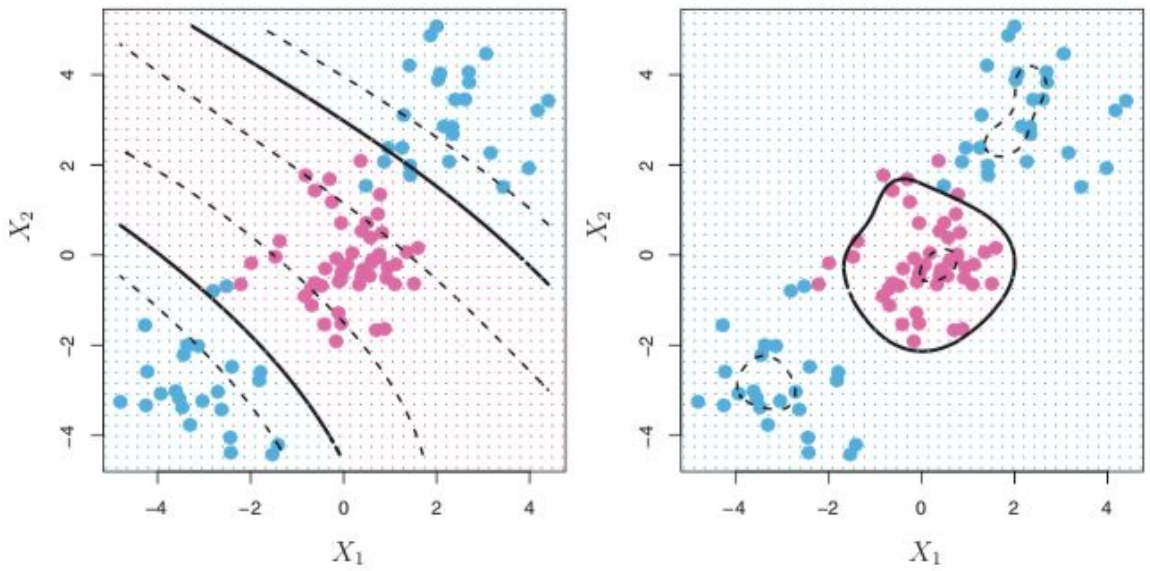


FIGURE 9.9. Left: *An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule.* Right: *An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.*

Ciências de Dados

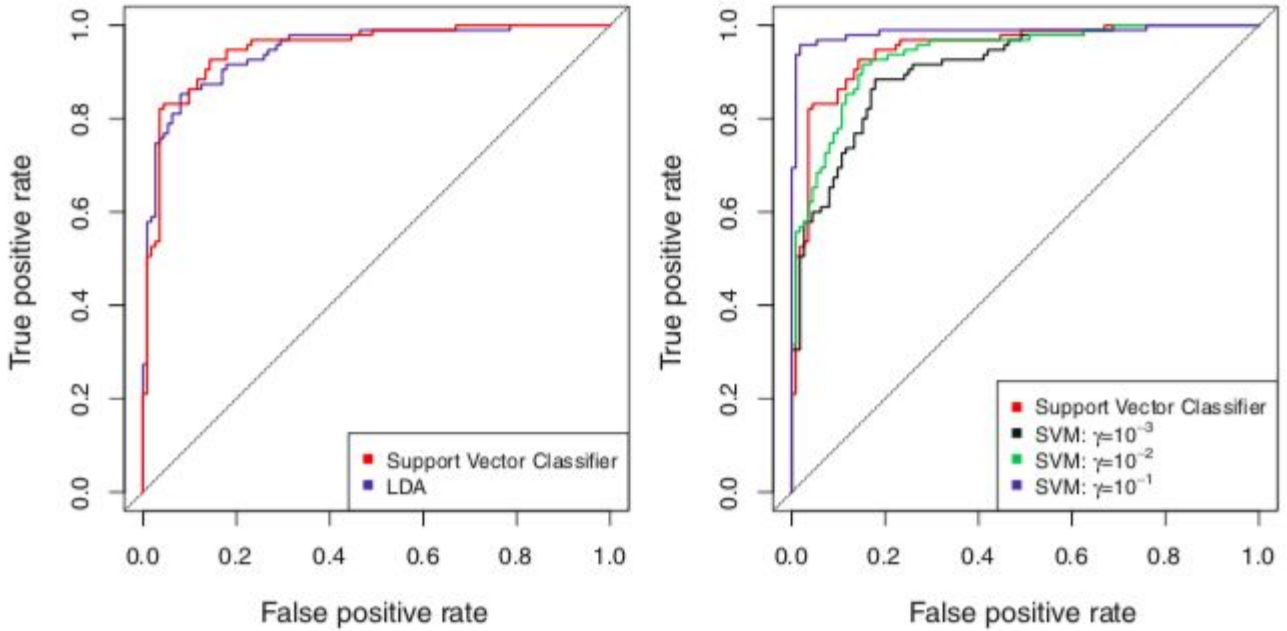


FIGURE 9.10. *ROC curves for the* `Heart` *data training set.* Left: *The support vector classifier and LDA are compared.* Right: *The support vector classifier is compared to an SVM using a radial basis kernel with* $\gamma = 10^{-3}$, 10^{-2}, *and* 10^{-1}.

Support vector machines para mais de 2 classes

Estender a lógica dos *support vector machines* para problemas de várias classes não é algo trivial. Duas abordagens são comuns, o caso *one-versus-one* e o caso *one-versus-all*.

One-versus-one constroi uma classificação binária para cada par de classes na variável resposta.

One-versus-all constroi uma classificação binária para uma classe e todas as demais aglutinadas em uma única classe geral.