Mineração de Dados

Aula 10: Truque do Kernel e SVM

Rafael Izbicki

Truque do Kernel

O truque: Usamos transformações das variáveis cujos respectivos produtos internos podem ser calculados eficientemente.

No KNN (K vizinhos mais próximos), tudo o que precisamos calcular é $d^2(\mathbf{x}_i, \mathbf{x}_j)$.

Álgebra Linear:

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||^2 = \langle \mathbf{x}_i, \mathbf{x}_i \rangle + \langle \mathbf{x}_j, \mathbf{x}_j \rangle - 2\langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$\langle \mathbf{x}_i, \mathbf{x}_l \rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k} \mathbf{x}_{l,k}$$

Assim, tudo o que precisamos é do produto interno entre todos os pares de observações.

Truque do Kernel

O truque: Usamos transformações das variáveis cujos respectivos produtos internos podem ser calculados eficientemente.

No KNN (K vizinhos mais próximos), tudo o que precisamos calcular é $d^2(\mathbf{x}_i, \mathbf{x}_j)$.

Álgebra Linear:

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||^2 = \langle \mathbf{x}_i, \mathbf{x}_i \rangle + \langle \mathbf{x}_j, \mathbf{x}_j \rangle - 2\langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$\langle \mathbf{x}_i, \mathbf{x}_l \rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k} \mathbf{x}_{l,k}$$

Assim, tudo o que precisamos é do produto interno entre todos os pares de observações.

Truque do Kernel

O truque: Usamos transformações das variáveis cujos respectivos produtos internos podem ser calculados eficientemente.

No KNN (K vizinhos mais próximos), tudo o que precisamos calcular é $d^2(\mathbf{x}_i, \mathbf{x}_j)$.

Álgebra Linear:

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||^2 = \langle \mathbf{x}_i, \mathbf{x}_i \rangle + \langle \mathbf{x}_j, \mathbf{x}_j \rangle - 2\langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$\langle \mathbf{x}_i, \mathbf{x}_l \rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k} \mathbf{x}_{l,k}$$

Assim, tudo o que precisamos é do produto interno entre todos os pares de observações.

Truque do Kernel

O truque: Usamos transformações das variáveis cujos respectivos produtos internos podem ser calculados eficientemente.

No KNN (K vizinhos mais próximos), tudo o que precisamos calcular é $d^2(\mathbf{x}_i, \mathbf{x}_j)$.

Álgebra Linear:

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||^2 = \langle \mathbf{x}_i, \mathbf{x}_i \rangle + \langle \mathbf{x}_j, \mathbf{x}_j \rangle - 2\langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$\langle \mathbf{x}_i, \mathbf{x}_l \rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k} \mathbf{x}_{l,k}$$

Assim, tudo o que precisamos é do produto interno entre todos os pares de observações.

Truque do Kernel

O truque: Usamos transformações das variáveis cujos respectivos produtos internos podem ser calculados eficientemente.

No KNN (K vizinhos mais próximos), tudo o que precisamos calcular é $d^2(\mathbf{x}_i, \mathbf{x}_j)$.

Álgebra Linear:

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = ||\mathbf{x}_i - \mathbf{x}_j||^2 = \langle \mathbf{x}_i, \mathbf{x}_i \rangle + \langle \mathbf{x}_j, \mathbf{x}_j \rangle - 2\langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$\langle \mathbf{x}_i, \mathbf{x}_l \rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k} \mathbf{x}_{l,k}$$

Assim, tudo o que precisamos é do produto interno entre todos os pares de observações.

Considere a transformação $\mathbf{w} = (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2, \sqrt{2}x_1x_2)$. Temos

$$\langle \mathbf{w}_i, \mathbf{w}_l \rangle = 1+2x_{i,1}\times x_{l,1}+x_{i,1}^2\times x_{l,1}^2+2x_{i,2}\times x_{l,2}+x_{i,2}^2\times x_{l,2}^2+2x_{i,1}x_{i,2}\times x_{l,1}x_{l,2}$$

$$= (1 + x_{i,1} \times x_{l,1} + x_{i,2} \times x_{l,2})^2 = (1 + \langle \mathbf{x}_i, \mathbf{x}_l \rangle)^2 := K(\mathbf{x}_i, \mathbf{x}_l)$$

Portanto, não precisamos calcular a transformação $\mathbf{w} = (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2, \sqrt{2}x_1x_2)$ explicitamente!

Isso economiza muito tempo, principalmente quanto temos muitas covariáveis. $K(\mathbf{x}_i, \mathbf{x}_l)$ é o produto interno das novas covariáveis, mas pode ser calculado diretamente sem passar pelo cálculo de w.

$$K(\mathbf{x}_i, \mathbf{x}_l) = (1 + \langle \mathbf{x}_i, \mathbf{x}_l \rangle)^d \ : \ \text{kernel polinomial}$$

$$K(\mathbf{x}_i, \mathbf{x}_l) = e^{-\frac{d^2(\mathbf{x}_i, \mathbf{x}_l)}{2h^2}} \ : \ \text{kernel Gaussiano}$$

Considere a transformação $\mathbf{w} = (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2, \sqrt{2}x_1x_2)$.
Temos

$$\langle \mathbf{w}_i, \mathbf{w}_l \rangle = 1+2x_{i,1}\times x_{l,1}+x_{i,1}^2\times x_{l,1}^2+2x_{i,2}\times x_{l,2}+x_{i,2}^2\times x_{l,2}^2+2x_{i,1}x_{i,2}\times x_{l,1}x_{l,2}$$

$$= (1 + x_{i,1} \times x_{l,1} + x_{i,2} \times x_{l,2})^2 = (1 + \langle \mathbf{x}_i, \mathbf{x}_l \rangle)^2 := K(\mathbf{x}_i, \mathbf{x}_l)$$

Portanto, não precisamos calcular a transformação $\mathbf{w} = (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2, \sqrt{2}x_1x_2)$ explicitamente!

Isso economiza muito tempo, principalmente quanto temos muitas covariáveis. $K(\mathbf{x}_i, \mathbf{x}_l)$ é o produto interno das novas covariáveis, mas pode ser calculado diretamente sem passar pelo cálculo de w.

$$K(\mathbf{x}_i, \mathbf{x}_l) = (1 + \langle \mathbf{x}_i, \mathbf{x}_l \rangle)^d \ : \ \text{kernel polinomial}$$

$$K(\mathbf{x}_i, \mathbf{x}_l) = e^{-\frac{d^2(\mathbf{x}_i, \mathbf{x}_l)}{2h^2}} \ : \ \text{kernel Gaussiano}$$

Considere a transformação $\mathbf{w} = (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2, \sqrt{2}x_1x_2)$. Temos

$$\langle \mathbf{w}_i, \mathbf{w}_l \rangle = 1+2x_{i,1}\times x_{l,1}+x_{i,1}^2\times x_{l,1}^2+2x_{i,2}\times x_{l,2}+x_{i,2}^2\times x_{l,2}^2+2x_{i,1}x_{i,2}\times x_{l,1}x_{l,2}$$

$$= (1 + x_{i,1} \times x_{l,1} + x_{i,2} \times x_{l,2})^2 = (1 + \langle \mathbf{x}_i, \mathbf{x}_l \rangle)^2 := K(\mathbf{x}_i, \mathbf{x}_l)$$

Portanto, não precisamos calcular a transformação $\mathbf{w} = (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2, \sqrt{2}x_1x_2)$ explicitamente!

Isso economiza muito tempo, principalmente quanto temos muitas covariáveis. $K(\mathbf{x}_i, \mathbf{x}_l)$ é o produto interno das novas covariáveis, mas pode ser calculado diretamente sem passar pelo cálculo de w.

$$K(\mathbf{x}_i, \mathbf{x}_l) = (1 + \langle \mathbf{x}_i, \mathbf{x}_l \rangle)^d \ : \ \text{kernel polinomial}$$

$$K(\mathbf{x}_i, \mathbf{x}_l) = e^{-\frac{d^2(\mathbf{x}_i, \mathbf{x}_l)}{2h^2}} \ : \ \text{kernel Gaussiano}$$

Considere a transformação $\mathbf{w} = (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2, \sqrt{2}x_1x_2)$. Temos

$$\langle \mathbf{w}_i, \mathbf{w}_l \rangle = 1+2x_{i,1}\times x_{l,1}+x_{i,1}^2\times x_{l,1}^2+2x_{i,2}\times x_{l,2}+x_{i,2}^2\times x_{l,2}^2+2x_{i,1}x_{i,2}\times x_{l,1}x_{l,2}$$

$$= (1 + x_{i,1} \times x_{l,1} + x_{i,2} \times x_{l,2})^2 = (1 + \langle \mathbf{x}_i, \mathbf{x}_l \rangle)^2 := K(\mathbf{x}_i, \mathbf{x}_l)$$

Portanto, não precisamos calcular a transformação $\mathbf{w} = (1, \sqrt{2}x_1, x_1^2, \sqrt{2}x_2, x_2^2, \sqrt{2}x_1x_2)$ explicitamente!

Isso economiza muito tempo, principalmente quanto temos muitas covariáveis. $K(\mathbf{x}_i, \mathbf{x}_l)$ é o produto interno das novas covariáveis, mas pode ser calculado diretamente sem passar pelo cálculo de w.

$$K(\mathbf{x}_i, \mathbf{x}_l) = (1 + \langle \mathbf{x}_i, \mathbf{x}_l \rangle)^d : \text{ kernel polinomial}$$

$$K(\mathbf{x}_i, \mathbf{x}_l) = e^{-\frac{d^2(\mathbf{x}_i, \mathbf{x}_l)}{2h^2}} : \text{ kernel Gaussiano}$$

Outro Exemplo: Regressão Linear

Lembrando:

$$\widehat{r}(\mathbf{x}) = \widehat{\boldsymbol{\beta}}^t \mathbf{x} = \widehat{\beta}_0 + \sum_{i=1}^{d} \widehat{\beta}_i x_i,$$

onde

$$\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{i=1}^{d} \beta_i x_{k,i} \right)^2$$

Solução:

$$\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_0, \widehat{\beta}_1, \ldots, \widehat{\beta}_d) = (\mathbb{X}^t \mathbb{X})^{-1} \mathbb{X}^t Y,$$

Outro Exemplo: Regressão Linear

Lembrando:

$$\widehat{r}(\mathbf{x}) = \widehat{\boldsymbol{\beta}}^t \mathbf{x} = \widehat{\beta}_0 + \sum_{i=1}^{d} \widehat{\beta}_i x_i,$$

onde

$$\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{i=1}^{d} \beta_i x_{k,i} \right)^2$$

Solução:

$$\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_0, \widehat{\beta}_1, \ldots, \widehat{\beta}_d) = (\mathbb{X}^t \mathbb{X})^{-1} \mathbb{X}^t Y,$$

Um pouco de álgebra mostra que $\widehat{\beta} = \mathbb{X}^t(\mathbb{X}\mathbb{X}^t)^{-1}Y$.

Assim,

$$\widehat{r}(\mathbf{x}) = Y^t(\mathbb{X}\mathbb{X}^t)^{-1}\mathbb{X}\mathbf{x} = Y^t\mathbb{K}^{-1}\mathbf{k}$$

onde

$$\mathbf{k} = (\langle \mathbf{x}_1, \mathbf{x}\rangle, \ldots, \langle \mathbf{x}_n, \mathbf{x}\rangle)^t$$

$$\mathbb{K} = \begin{bmatrix} \langle \mathbf{x}_1, \mathbf{x}_1\rangle & \langle \mathbf{x}_1, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_1, \mathbf{x}_n\rangle \\ \langle \mathbf{x}_2, \mathbf{x}_1\rangle & \langle \mathbf{x}_2, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_2, \mathbf{x}_n\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathbf{x}_n, \mathbf{x}_1\rangle & \langle \mathbf{x}_n, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_n, \mathbf{x}_n\rangle \end{bmatrix}$$

$$\langle \mathbf{x}_i, \mathbf{x}_l\rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k}\mathbf{x}_{l,k}$$

Só depende de produtos internos entre as observações!

Um pouco de álgebra mostra que $\widehat{\beta} = \mathbb{X}^t(\mathbb{X}\mathbb{X}^t)^{-1}Y$.

Assim,

$$\widehat{r}(\mathbf{x}) = Y^t(\mathbb{X}\mathbb{X}^t)^{-1}\mathbb{X}\mathbf{x} = Y^t\mathbb{K}^{-1}\mathbf{k}$$

onde

$$\mathbf{k} = (\langle \mathbf{x}_1, \mathbf{x}\rangle, \ldots, \langle \mathbf{x}_n, \mathbf{x}\rangle)^t$$

$$\mathbb{K} = \begin{bmatrix} \langle \mathbf{x}_1, \mathbf{x}_1\rangle & \langle \mathbf{x}_1, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_1, \mathbf{x}_n\rangle \\ \langle \mathbf{x}_2, \mathbf{x}_1\rangle & \langle \mathbf{x}_2, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_2, \mathbf{x}_n\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathbf{x}_n, \mathbf{x}_1\rangle & \langle \mathbf{x}_n, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_n, \mathbf{x}_n\rangle \end{bmatrix}$$

$$\langle \mathbf{x}_i, \mathbf{x}_l\rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k}\mathbf{x}_{l,k}$$

Só depende de produtos internos entre as observações!

Um pouco de álgebra mostra que $\widehat{\beta} = \mathbb{X}^t(\mathbb{X}\mathbb{X}^t)^{-1}Y$.

Assim,

$$\widehat{r}(\mathbf{x}) = Y^t(\mathbb{X}\mathbb{X}^t)^{-1}\mathbb{X}\mathbf{x} = Y^t\mathbb{K}^{-1}\mathbf{k}$$

onde

$$\mathbf{k} = \left(\langle \mathbf{x}_1, \mathbf{x}\rangle, \ldots, \langle \mathbf{x}_n, \mathbf{x}\rangle\right)^t$$

$$\mathbb{K} = \begin{bmatrix} \langle \mathbf{x}_1, \mathbf{x}_1\rangle & \langle \mathbf{x}_1, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_1, \mathbf{x}_n\rangle \\ \langle \mathbf{x}_2, \mathbf{x}_1\rangle & \langle \mathbf{x}_2, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_2, \mathbf{x}_n\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathbf{x}_n, \mathbf{x}_1\rangle & \langle \mathbf{x}_n, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_n, \mathbf{x}_n\rangle \end{bmatrix}$$

$$\langle \mathbf{x}_i, \mathbf{x}_l\rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k}\mathbf{x}_{l,k}$$

Só depende de produtos internos entre as observações!

Um pouco de álgebra mostra que $\widehat{\beta} = \mathbb{X}^t(\mathbb{X}\mathbb{X}^t)^{-1}Y$.

Assim,

$$\widehat{r}(\mathbf{x}) = Y^t(\mathbb{X}\mathbb{X}^t)^{-1}\mathbb{X}\mathbf{x} = Y^t\mathbb{K}^{-1}\mathbf{k}$$

onde

$$\mathbf{k} = \left(\langle \mathbf{x}_1, \mathbf{x}\rangle, \ldots, \langle \mathbf{x}_n, \mathbf{x}\rangle\right)^t$$

$$\mathbb{K} = \begin{bmatrix} \langle \mathbf{x}_1, \mathbf{x}_1\rangle & \langle \mathbf{x}_1, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_1, \mathbf{x}_n\rangle \\ \langle \mathbf{x}_2, \mathbf{x}_1\rangle & \langle \mathbf{x}_2, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_2, \mathbf{x}_n\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathbf{x}_n, \mathbf{x}_1\rangle & \langle \mathbf{x}_n, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_n, \mathbf{x}_n\rangle \end{bmatrix}$$

$$\langle \mathbf{x}_i, \mathbf{x}_l\rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k}\mathbf{x}_{l,k}$$

Só depende de produtos internos entre as observações!

Um pouco de álgebra mostra que $\widehat{\beta} = \mathbb{X}^t(\mathbb{X}\mathbb{X}^t)^{-1}Y$.

Assim,

$$\widehat{r}(\mathbf{x}) = Y^t(\mathbb{X}\mathbb{X}^t)^{-1}\mathbb{X}\mathbf{x} = Y^t\mathbb{K}^{-1}\mathbf{k}$$

onde

$$\mathbf{k} = (\langle \mathbf{x}_1, \mathbf{x}\rangle, \ldots, \langle \mathbf{x}_n, \mathbf{x}\rangle)^t$$

$$\mathbb{K} = \begin{bmatrix} \langle \mathbf{x}_1, \mathbf{x}_1\rangle & \langle \mathbf{x}_1, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_1, \mathbf{x}_n\rangle \\ \langle \mathbf{x}_2, \mathbf{x}_1\rangle & \langle \mathbf{x}_2, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_2, \mathbf{x}_n\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathbf{x}_n, \mathbf{x}_1\rangle & \langle \mathbf{x}_n, \mathbf{x}_2\rangle & \cdots & \langle \mathbf{x}_n, \mathbf{x}_n\rangle \end{bmatrix}$$

$$\langle \mathbf{x}_i, \mathbf{x}_l\rangle = \sum_{k=1}^{d} \mathbf{x}_{i,k}\mathbf{x}_{l,k}$$

Só depende de produtos internos entre as observações!

Support Vector Machines

Random Forest (2001)

SVM - Vapnik (1995)

SVM é um classificador para quando há duas categorias. (Pode ser combinado no caso de várias categorias).

É conveniente usar respostas em $\{-1, 1\}$ ao invés de $\{0, 1\}$

Como em árvores, em SVM não estimamos probabilidades. O classificador tem a seguinte forma:

Se $f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p < 0$, predizemos que $Y = -1$ (i.e., $g(\mathbf{x}) = -1$)

Se $f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p > 0$, predizemos que $Y = 1$ (i.e., $g(\mathbf{x}) = 1$).

SVM é um classificador para quando há duas categorias. (Pode ser combinado no caso de várias categorias).

É conveniente usar respostas em $\{-1, 1\}$ ao invés de $\{0, 1\}$

Como em árvores, em SVM não estimamos probabilidades. O classificador tem a seguinte forma:

Se $f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p < 0$, predizemos que $Y = -1$ (i.e., $g(\mathbf{x}) = -1$)

Se $f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p > 0$, predizemos que $Y = 1$ (i.e., $g(\mathbf{x}) = 1$).

SVM é um classificador para quando há duas categorias. (Pode ser combinado no caso de várias categorias).

É conveniente usar respostas em $\{-1, 1\}$ ao invés de $\{0, 1\}$

Como em árvores, em SVM não estimamos probabilidades. O classificador tem a seguinte forma:

Se $f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p < 0$, predizemos que $Y = -1$ (i.e., $g(\mathbf{x}) = -1$)

Se $f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p > 0$, predizemos que $Y = 1$ (i.e., $g(\mathbf{x}) = 1$).

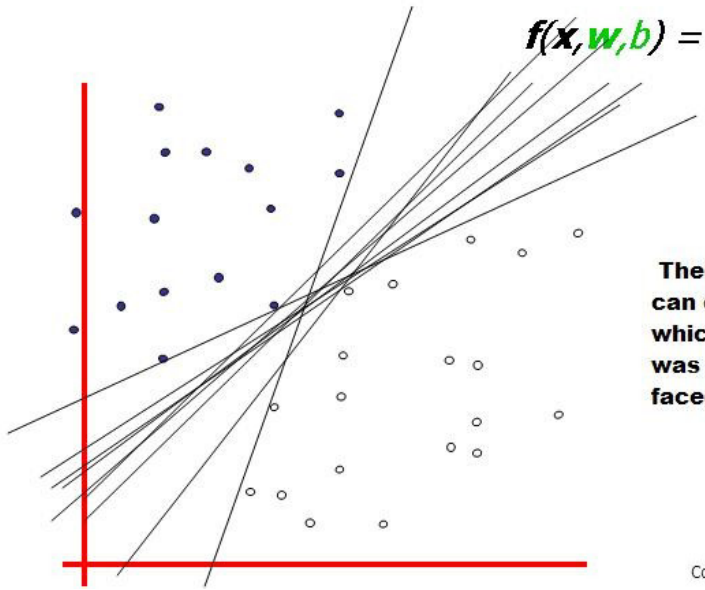
SVM é um classificador para quando há duas categorias. (Pode ser combinado no caso de várias categorias).

É conveniente usar respostas em $\{-1, 1\}$ ao invés de $\{0, 1\}$

Como em árvores, em SVM não estimamos probabilidades. O classificador tem a seguinte forma:

Se $f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p < 0$, predizemos que $Y = -1$ (i.e., $g(\mathbf{x}) = -1$)

Se $f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p > 0$, predizemos que $Y = 1$ (i.e., $g(\mathbf{x}) = 1$).



There are many fits which can classify this data but which one is the best? This was one of the challenges faced.

Se um hiperplano separa perfeitamente todas a observações do conjunto de treinamento, então:

$$y_i(\beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p}) = y_i f(\mathbf{x}_i) > 0$$

para todo $i = 1, \ldots, n$

Se $|f(\mathbf{x})|$ é muito alto, $\mathbf{x}$ está muito longe do plano separador.

Quando existem muitos hiperplanos que separam os dados, busca-se aqueles que tem maiores margens.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M \ \forall i = 1, \ldots, n$$

É fácil resolver esse problema (mas não entraremos em detalhes)

Se um hiperplano separa perfeitamente todas a observações do conjunto de treinamento, então:

$$y_i(\beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p}) = y_i f(\mathbf{x}_i) > 0$$

para todo $i = 1, \ldots, n$

Se $|f(\mathbf{x})|$ é muito alto, $\mathbf{x}$ está muito longe do plano separador.

Quando existem muitos hiperplanos que separam os dados, busca-se aqueles que tem maiores margens.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M \ \forall i = 1, \ldots, n$$

É fácil resolver esse problema (mas não entraremos em detalhes)

Se um hiperplano separa perfeitamente todas a observações do conjunto de treinamento, então:

$$y_i(\beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p}) = y_i f(\mathbf{x}_i) > 0$$

para todo $i = 1, \ldots, n$

Se $|f(\mathbf{x})|$ é muito alto, $\mathbf{x}$ está muito longe do plano separador.

Quando existem muitos hiperplanos que separam os dados, busca-se aqueles que tem maiores margens.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M \ \forall i = 1, \ldots, n$$

É fácil resolver esse problema (mas não entraremos em detalhes)

Se um hiperplano separa perfeitamente todas a observações do conjunto de treinamento, então:

$$y_i(\beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p}) = y_i f(\mathbf{x}_i) > 0$$

para todo $i = 1, \ldots, n$

Se $|f(\mathbf{x})|$ é muito alto, $\mathbf{x}$ está muito longe do plano separador.

Quando existem muitos hiperplanos que separam os dados, busca-se aqueles que tem maiores margens.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M \ \forall i = 1, \ldots, n$$

É fácil resolver esse problema (mas não entraremos em detalhes)

Se um hiperplano separa perfeitamente todas a observações do conjunto de treinamento, então:

$$y_i(\beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p}) = y_i f(\mathbf{x}_i) > 0$$

para todo $i = 1, \ldots, n$

Se $|f(\mathbf{x})|$ é muito alto, $\mathbf{x}$ está muito longe do plano separador.

Quando existem muitos hiperplanos que separam os dados, busca-se aqueles que tem maiores margens.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M \ \forall i = 1, \ldots, n$$

É fácil resolver esse problema (mas não entraremos em detalhes)

Se um hiperplano separa perfeitamente todas a observações do conjunto de treinamento, então:

$$y_i(\beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p}) = y_i f(\mathbf{x}_i) > 0$$

para todo $i = 1, \ldots, n$

Se $|f(\mathbf{x})|$ é muito alto, $\mathbf{x}$ está muito longe do plano separador.

Quando existem muitos hiperplanos que separam os dados, busca-se aqueles que tem maiores margens.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M \ \forall i = 1, \ldots, n$$

É fácil resolver esse problema (mas não entraremos em detalhes)

Se um hiperplano separa perfeitamente todas a observações do conjunto de treinamento, então:

$$y_i(\beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p}) = y_i f(\mathbf{x}_i) > 0$$

para todo $i = 1, \ldots, n$

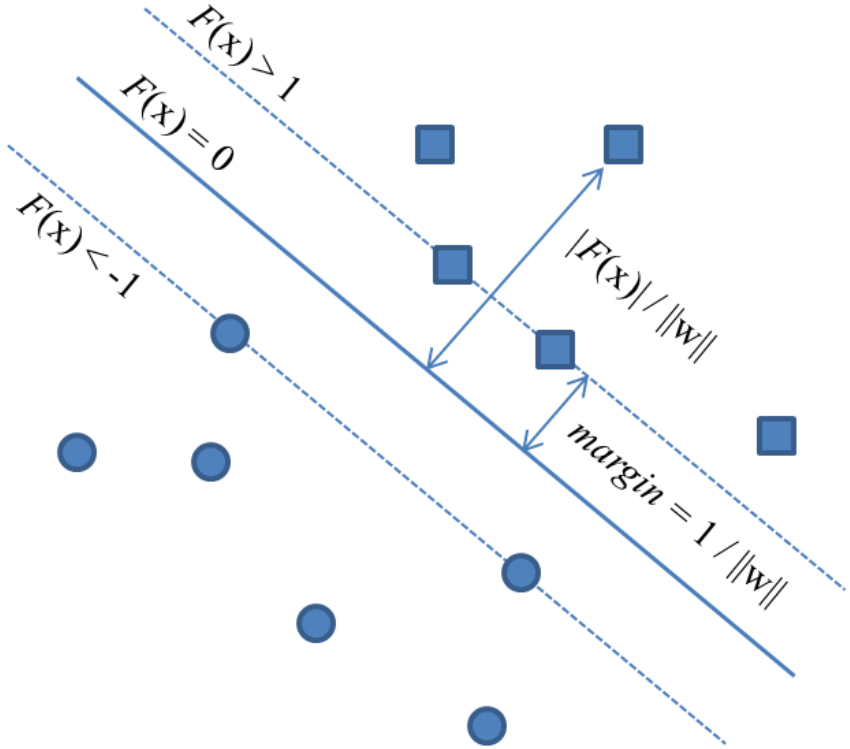
Se $|f(\mathbf{x})|$ é muito alto, $\mathbf{x}$ está muito longe do plano separador.

Quando existem muitos hiperplanos que separam os dados, busca-se aqueles que tem maiores margens.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M \ \forall i = 1, \ldots, n$$

É fácil resolver esse problema (mas não entraremos em detalhes)


F(x) > 1
F(x) = 0
F(x) < -1
|F(x)| / ||w||
margin = 1 / ||w||

Esta abordagem é muito sensível a pequenas mudanças nos dados.

Uma solução é permitir que alguns dos pontos estejam do lado "errado" das margens (e eventualmente do hiperplano). Ideia: classificar erroneamente algumas observações pode levar a melhor poder preditivo.

Em particular, isso permite que o classificador possa ser usado em situações em que não há um hiperplano que separa perfeitamente os dados.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M(1-\epsilon_i)$$

$$\text{com } \epsilon_i > 0 \text{ e } \sum_{i=1}^{n} \epsilon_i \leq C \quad \forall i = 1, \ldots, n$$

C é um tuning parameter, quanto maior, mais se permite que observações caiam do lado "errado" das margens.

Esta abordagem é muito sensível a pequenas mudanças nos dados.

Uma solução é permitir que alguns dos pontos estejam do lado "errado" das margens (e eventualmente do hiperplano). Ideia: classificar erroneamente algumas observações pode levar a melhor poder preditivo.

Em particular, isso permite que o classificador possa ser usado em situações em que não há um hiperplano que separa perfeitamente os dados.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M(1-\epsilon_i)$$

$$\text{com } \epsilon_i > 0 \text{ e } \sum_{i=1}^{n} \epsilon_i \leq C \quad \forall i = 1, \ldots, n$$

C é um tuning parameter, quanto maior, mais se permite que observações caiam do lado "errado" das margens.

Esta abordagem é muito sensível a pequenas mudanças nos dados.

Uma solução é permitir que alguns dos pontos estejam do lado "errado" das margens (e eventualmente do hiperplano). Ideia: classificar erroneamente algumas observações pode levar a melhor poder preditivo.

Em particular, isso permite que o classificador possa ser usado em situações em que não há um hiperplano que separa perfeitamente os dados.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M(1-\epsilon_i)$$

$$\text{com } \epsilon_i > 0 \text{ e } \sum_{i=1}^{n} \epsilon_i \leq C \quad \forall i = 1, \ldots, n$$

C é um tuning parameter, quanto maior, mais se permite que observações caiam do lado "errado" das margens.

Esta abordagem é muito sensível a pequenas mudanças nos dados.

Uma solução é permitir que alguns dos pontos estejam do lado "errado" das margens (e eventualmente do hiperplano). Ideia: classificar erroneamente algumas observações pode levar a melhor poder preditivo.

Em particular, isso permite que o classificador possa ser usado em situações em que não há um hiperplano que separa perfeitamente os dados.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M(1-\epsilon_i)$$

$$\text{com } \epsilon_i > 0 \text{ e } \sum_{i=1}^{n} \epsilon_i \leq C \quad \forall i = 1, \ldots, n$$

C é um tuning parameter, quanto maior, mais se permite que observações caiam do lado "errado" das margens.

Esta abordagem é muito sensível a pequenas mudanças nos dados.

Uma solução é permitir que alguns dos pontos estejam do lado "errado" das margens (e eventualmente do hiperplano). Ideia: classificar erroneamente algumas observações pode levar a melhor poder preditivo.

Em particular, isso permite que o classificador possa ser usado em situações em que não há um hiperplano que separa perfeitamente os dados.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M(1-\epsilon_i)$$

$$\text{com } \epsilon_i > 0 \text{ e } \sum_{i=1}^{n} \epsilon_i \leq C \quad \forall i = 1, \ldots, n$$

C é um tuning parameter, quanto maior, mais se permite que observações caiam do lado "errado" das margens.

Esta abordagem é muito sensível a pequenas mudanças nos dados.

Uma solução é permitir que alguns dos pontos estejam do lado "errado" das margens (e eventualmente do hiperplano). Ideia: classificar erroneamente algumas observações pode levar a melhor poder preditivo.

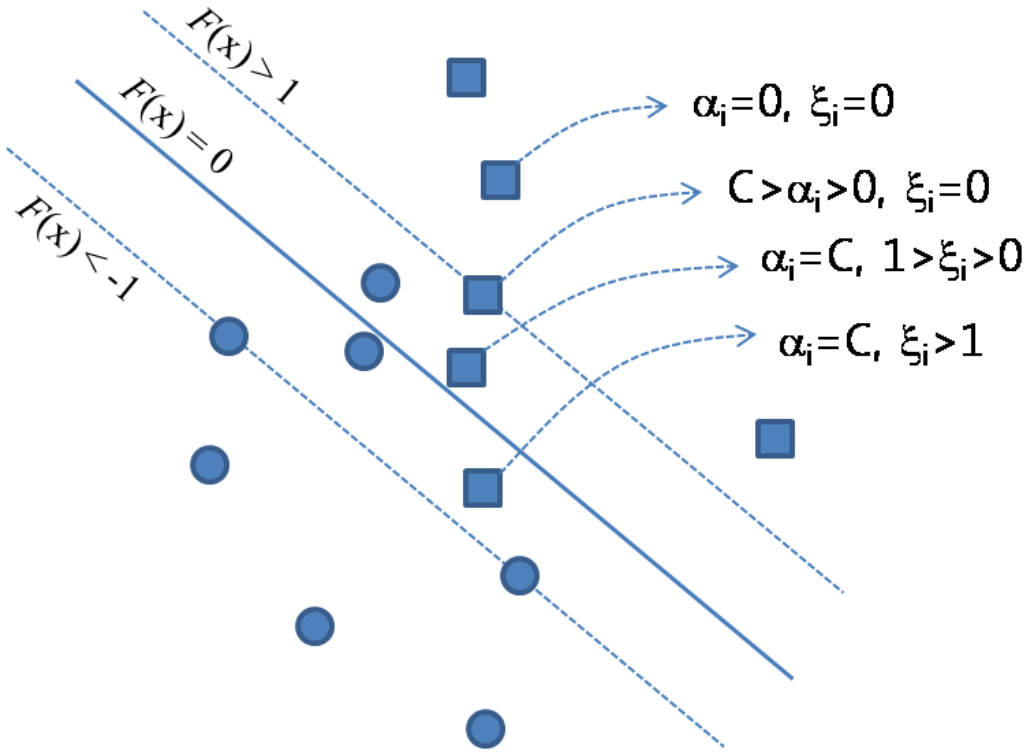
Em particular, isso permite que o classificador possa ser usado em situações em que não há um hiperplano que separa perfeitamente os dados.

Matematicamente:

$$\arg\max_{\beta} M \text{ sujeito a } \sum_{i=1}^{p} \beta_i^2 = 1 \text{ e } y_i f_\beta(\mathbf{x}_i) \geq M(1-\epsilon_i)$$

$$\text{com } \epsilon_i > 0 \text{ e } \sum_{i=1}^{n} \epsilon_i \leq C \quad \forall i = 1, \ldots, n$$

C é um tuning parameter, quanto maior, mais se permite que observações caiam do lado "errado" das margens.


F(x) > 1
F(x) = 0
F(x) < -1
αi=0, ξi=0
C>αi>0, ξi=0
αi=C, 1>ξi>0
αi=C, ξi>1

O que fazer se queremos divisões mais complexas que hiperplanos?

Truque do kernel!

Podemos considerar transformações das variáveis originais (ex: $\mathbf{w} = (x_1, x_1^2, x_2, x_2^2, x_1 x_2)$), e usar SVM nessas novas covariáveis $\mathbf{w}$

Ideia básica: a solução $f(\mathbf{x})$ pode ser reescrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle.$$

(não vamos mostrar isso)

O que fazer se queremos divisões mais complexas que hiperplanos?

Truque do kernel!

Podemos considerar transformações das variáveis originais (ex: $\mathbf{w} = (x_1, x_1^2, x_2, x_2^2, x_1 x_2)$), e usar SVM nessas novas covariáveis $\mathbf{w}$

Ideia básica: a solução $f(\mathbf{x})$ pode ser reescrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle.$$

(não vamos mostrar isso)

O que fazer se queremos divisões mais complexas que hiperplanos?

Truque do kernel!

Podemos considerar transformações das variáveis originais (ex: $\mathbf{w} = (x_1, x_1^2, x_2, x_2^2, x_1 x_2)$), e usar SVM nessas novas covariáveis $\mathbf{w}$

Ideia básica: a solução $f(\mathbf{x})$ pode ser reescrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle.$$

(não vamos mostrar isso)

O que fazer se queremos divisões mais complexas que hiperplanos?

Truque do kernel!

Podemos considerar transformações das variáveis originais (ex: $\mathbf{w} = (x_1, x_1^2, x_2, x_2^2, x_1 x_2)$), e usar SVM nessas novas covariáveis $\mathbf{w}$

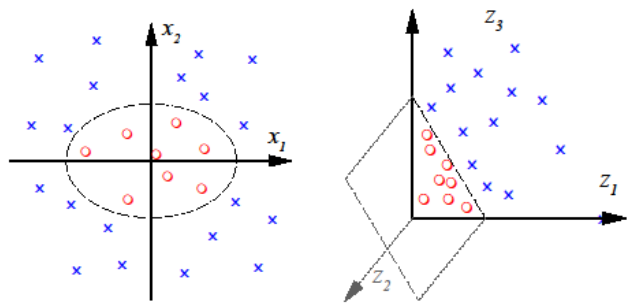
Ideia básica: a solução $f(\mathbf{x})$ pode ser reescrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle.$$

(não vamos mostrar isso)

21 SVMs : polynomial mapping

$$\Phi : R^2 \to R^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{(2)} x_1 x_2, x_2^2)$$



$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

onde $\langle \mathbf{x}, \mathbf{x}_k \rangle = \sum_{i=1}^{p} x_i x_{k,i}$

Assim, para calcular f (i.e., os coeficientes α_k), tudo o que precisamos é do produto interno entre todas as observações.

Para calcular α_k, também necessitamos apenas dos produtos internos entre as observações.

Podemos, portanto, trocar $\langle \mathbf{x}, \mathbf{x}_k \rangle$ por um kernel genérico $K(\mathbf{x}, \mathbf{x}_k)$

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

onde $\langle \mathbf{x}, \mathbf{x}_k \rangle = \sum_{i=1}^{p} x_i x_{k,i}$

Assim, para calcular f (i.e., os coeficientes α_k), tudo o que precisamos é do produto interno entre todas as observações.

Para calcular α_k, também necessitamos apenas dos produtos internos entre as observações.

Podemos, portanto, trocar $\langle \mathbf{x}, \mathbf{x}_k \rangle$ por um kernel genérico $K(\mathbf{x}, \mathbf{x}_k)$

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

onde $\langle \mathbf{x}, \mathbf{x}_k \rangle = \sum_{i=1}^{p} x_i x_{k,i}$

Assim, para calcular f (i.e., os coeficientes α_k), tudo o que precisamos é do produto interno entre todas as observações.

Para calcular α_k, também necessitamos apenas dos produtos internos entre as observações.

Podemos, portanto, trocar $\langle \mathbf{x}, \mathbf{x}_k \rangle$ por um kernel genérico $K(\mathbf{x}, \mathbf{x}_k)$

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

onde $\langle \mathbf{x}, \mathbf{x}_k \rangle = \sum_{i=1}^{p} x_i x_{k,i}$

Assim, para calcular f (i.e., os coeficientes α_k), tudo o que precisamos é do produto interno entre todas as observações.

Para calcular α_k, também necessitamos apenas dos produtos internos entre as observações.

Podemos, portanto, trocar $\langle \mathbf{x}, \mathbf{x}_k \rangle$ por um kernel genérico $K(\mathbf{x}, \mathbf{x}_k)$

Resumindo: SVM busca encontrar hiperplanos que separam bem os dados.

Sua solução pode ser escrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p} = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

Para classificar uma nova observação, basta ver se $f(\mathbf{x}) > 0$ ou não.

Para criar regiões de decisão que são não-lineares, podemos usar o truque do kernel:

$$f(\mathbf{x}) = \beta_0 + \sum_{k=1}^{n} \alpha_k K(\mathbf{x}, \mathbf{x}_k),$$

A ideia por trás disso é que estamos implicitamente fazendo uma transformação das variáveis originais. Contudo não é necessário calculá-la explicitamente, o que economiza muito tempo.

Dependendo da aplicação, um kernel pode ser melhor ou pior.

Resumindo: SVM busca encontrar hiperplanos que separam bem os dados.

Sua solução pode ser escrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p} = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

Para classificar uma nova observação, basta ver se $f(\mathbf{x}) > 0$ ou não.

Para criar regiões de decisão que são não-lineares, podemos usar o truque do kernel:

$$f(\mathbf{x}) = \beta_0 + \sum_{k=1}^{n} \alpha_k K(\mathbf{x}, \mathbf{x}_k),$$

A ideia por trás disso é que estamos implicitamente fazendo uma transformação das variáveis originais. Contudo não é necessário calculá-la explicitamente, o que economiza muito tempo.

Dependendo da aplicação, um kernel pode ser melhor ou pior.

Resumindo: SVM busca encontrar hiperplanos que separam bem os dados.

Sua solução pode ser escrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p} = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

Para classificar uma nova observação, basta ver se $f(\mathbf{x}) > 0$ ou não.

Para criar regiões de decisão que são não-lineares, podemos usar o truque do kernel:

$$f(\mathbf{x}) = \beta_0 + \sum_{k=1}^{n} \alpha_k K(\mathbf{x}, \mathbf{x}_k),$$

A ideia por trás disso é que estamos implicitamente fazendo uma transformação das variáveis originais. Contudo não é necessário calculá-la explicitamente, o que economiza muito tempo.

Dependendo da aplicação, um kernel pode ser melhor ou pior.

Resumindo: SVM busca encontrar hiperplanos que separam bem os dados.

Sua solução pode ser escrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p} = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

Para classificar uma nova observação, basta ver se $f(\mathbf{x}) > 0$ ou não.

Para criar regiões de decisão que são não-lineares, podemos usar o truque do kernel:

$$f(\mathbf{x}) = \beta_0 + \sum_{k=1}^{n} \alpha_k K(\mathbf{x}, \mathbf{x}_k),$$

A ideia por trás disso é que estamos implicitamente fazendo uma transformação das variáveis originais. Contudo não é necessário calculá-la explicitamente, o que economiza muito tempo.

Dependendo da aplicação, um kernel pode ser melhor ou pior.

Resumindo: SVM busca encontrar hiperplanos que separam bem os dados.

Sua solução pode ser escrita como

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_{i,1} + \ldots + \beta_p x_{i,p} = \beta_0 + \sum_{k=1}^{n} \alpha_k \langle \mathbf{x}, \mathbf{x}_k \rangle,$$

Para classificar uma nova observação, basta ver se $f(\mathbf{x}) > 0$ ou não.

Para criar regiões de decisão que são não-lineares, podemos usar o truque do kernel:

$$f(\mathbf{x}) = \beta_0 + \sum_{k=1}^{n} \alpha_k K(\mathbf{x}, \mathbf{x}_k),$$

A ideia por trás disso é que estamos implicitamente fazendo uma transformação das variáveis originais. Contudo não é necessário calculá-la explicitamente, o que economiza muito tempo.

Dependendo da aplicação, um kernel pode ser melhor ou pior.

Atenção: como no lasso, em geral é melhor padronizar as covariáveis antes

Código R.

Atenção: como no lasso, em geral é melhor padronizar as covariáveis antes

Código R.