Universidade Federal do Paraná

Laboratório de Estatística e Geoinformação - LEG

Support Vector Machines

Prof.: Eduardo Vargas Ferreira

Definição

- **Support Vector Machines** são baseados no conceito de planos de decisão (que definem limites de decisão);
- Tentamos encontrar o plano que separa as classes no espaço de características;
- Em geral, a equação para o hiperplano tem a forma

$$f(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

- Um hiperplano de dimensão p é um subespaço de dimensão $(p-1)$.
- Por exemplo, quando $p = 2$ o hiperplano é uma linha;
- Se $\beta_0 = 0$ dizemos que o hiperplano passa na origem, caso contrário não.

Definição

- **Support Vector Machines** são baseados no conceito de planos de decisão (que definem limites de decisão);
- Tentamos encontrar o plano que separa as classes no espaço de características;
- Em geral, a equação para o hiperplano tem a forma

$$f(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

- Um hiperplano de dimensão p é um subespaço de dimensão $(p-1)$.
- Por exemplo, quando $p = 2$ o hiperplano é uma linha;
- Se $\beta_0 = 0$ dizemos que o hiperplano passa na origem, caso contrário não.

Definição

- **Support Vector Machines** são baseados no conceito de planos de decisão (que definem limites de decisão);
- Tentamos encontrar o plano que separa as classes no espaço de características;
- Em geral, a equação para o hiperplano tem a forma

$$f(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

- Um hiperplano de dimensão p é um subespaço de dimensão $(p-1)$.
- Por exemplo, quando $p = 2$ o hiperplano é uma linha;
- Se $\beta_0 = 0$ dizemos que o hiperplano passa na origem, caso contrário não.

Definição

- **Support Vector Machines** são baseados no conceito de planos de decisão (que definem limites de decisão);
- Tentamos encontrar o plano que separa as classes no espaço de características;
- Em geral, a equação para o hiperplano tem a forma

$$f(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

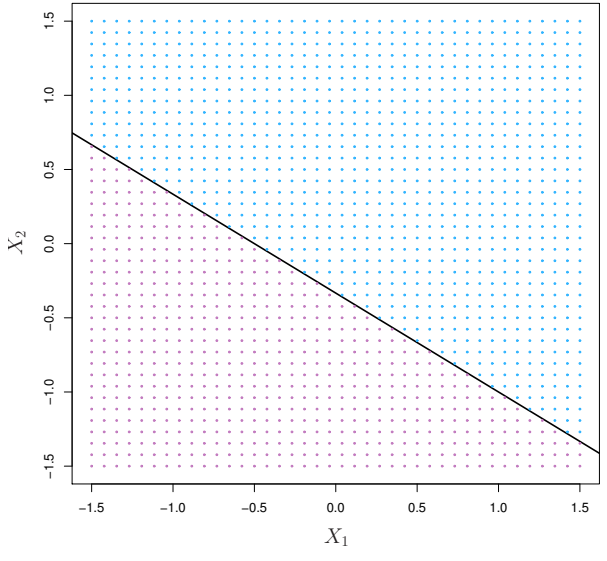
- Um hiperplano de dimensão p é um subespaço de dimensão $(p-1)$.
- Por exemplo, quando $p = 2$ o hiperplano é uma linha;
- Se $\beta_0 = 0$ dizemos que o hiperplano passa na origem, caso contrário não.

Definição

- **Support Vector Machines** são baseados no conceito de planos de decisão (que definem limites de decisão);
- Tentamos encontrar o plano que separa as classes no espaço de características;
- Em geral, a equação para o hiperplano tem a forma

$$f(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

- Um hiperplano de dimensão p é um subespaço de dimensão $(p-1)$.
- Por exemplo, quando $p = 2$ o hiperplano é uma linha;
- Se $\beta_0 = 0$ dizemos que o hiperplano passa na origem, caso contrário não.

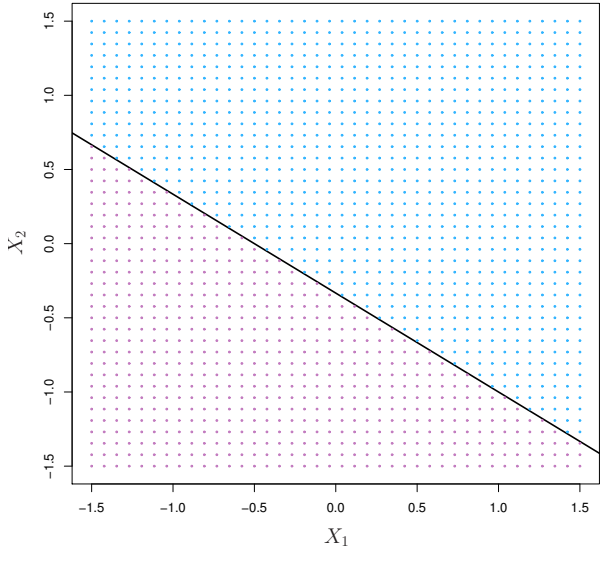


Definição

- **Support Vector Machines** são baseados no conceito de planos de decisão (que definem limites de decisão);
- Tentamos encontrar o plano que separa as classes no espaço de características;
- Em geral, a equação para o hiperplano tem a forma

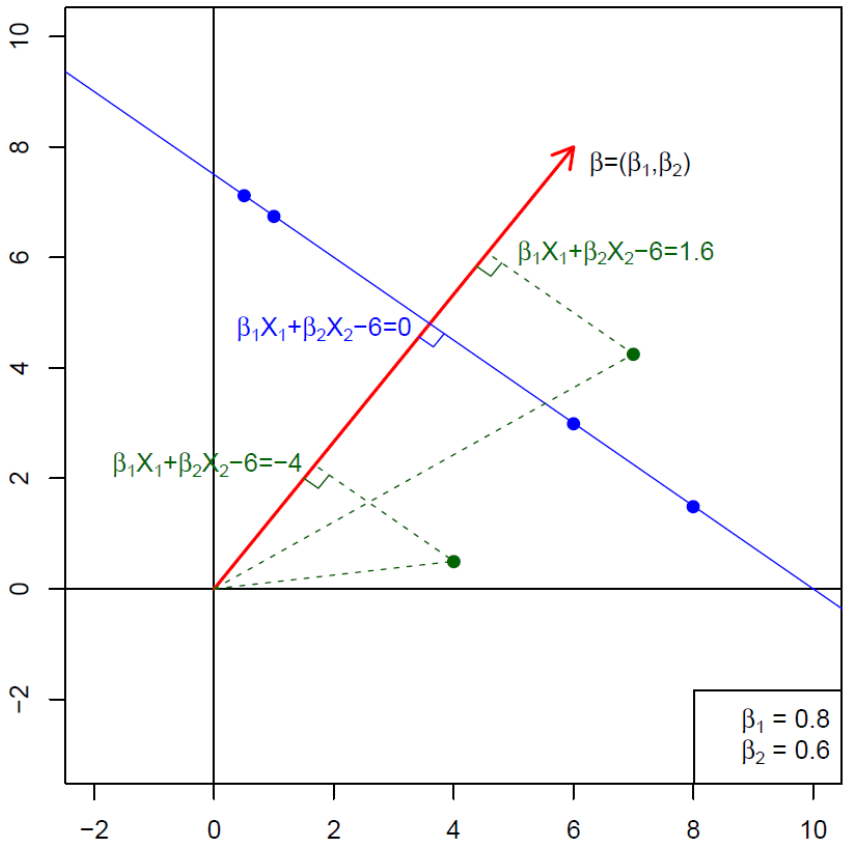
$$f(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

- Um hiperplano de dimensão p é um subespaço de dimensão $(p-1)$.
- Por exemplo, quando $p = 2$ o hiperplano é uma linha;
- Se $\beta_0 = 0$ dizemos que o hiperplano passa na origem, caso contrário não.



O que é um hiperplano?

- O vetor $\beta = (\beta_0, \beta_1, \ldots, \beta_p)$ é chamado vetor normal. Ele aponta a direção ortogonal à superfície do hiperplano;



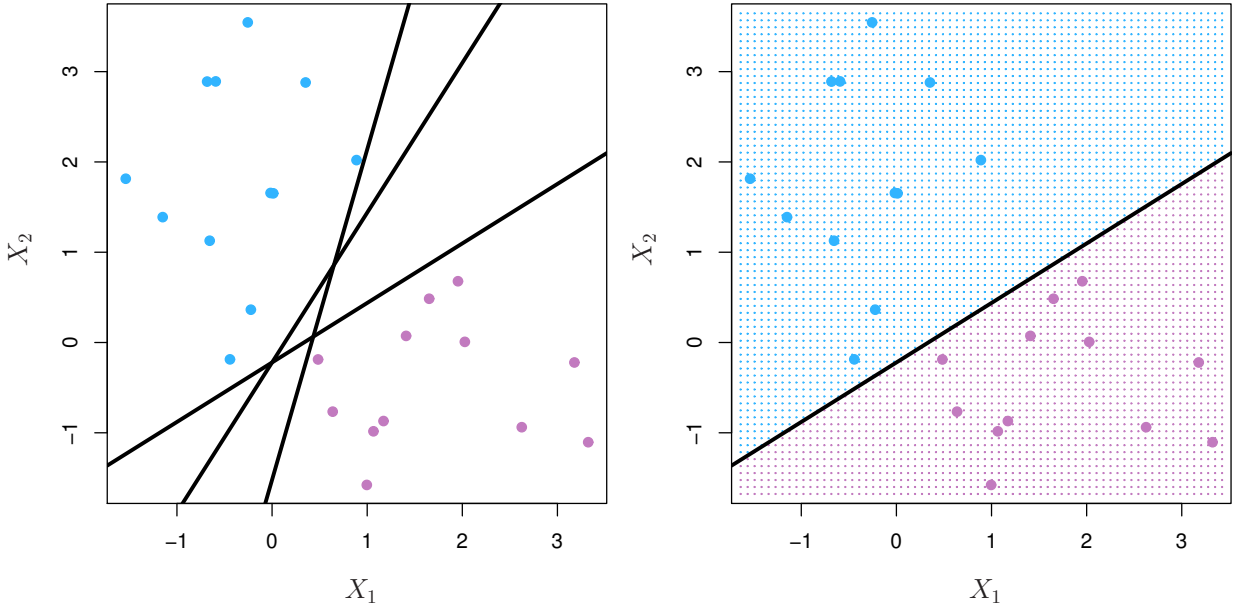
- Para cada ponto em verde, podemos definir hiperplanos ortogonais ao vetor $\boldsymbol{\beta}$;

Separando as classes

- Como vimos anteriormente, de um lado do hiperplano a função assume valores positivos e do outro negativo;
- Assim, se $f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$, então $f(X) > 0$ para ponto em um lado do hiperplano, e $f(X) < 0$ para pontos do outro lado;
- Note que se codificarmos como $Y_i = +1$ os pontos em azul, e $Y_i = -1$ os pontos rosa, então $Y_i \cdot f(X_i) > 0$ para todo i.
- Mas, em meio a tantos hiperplanos possíveis, qual escolher?

Separando as classes

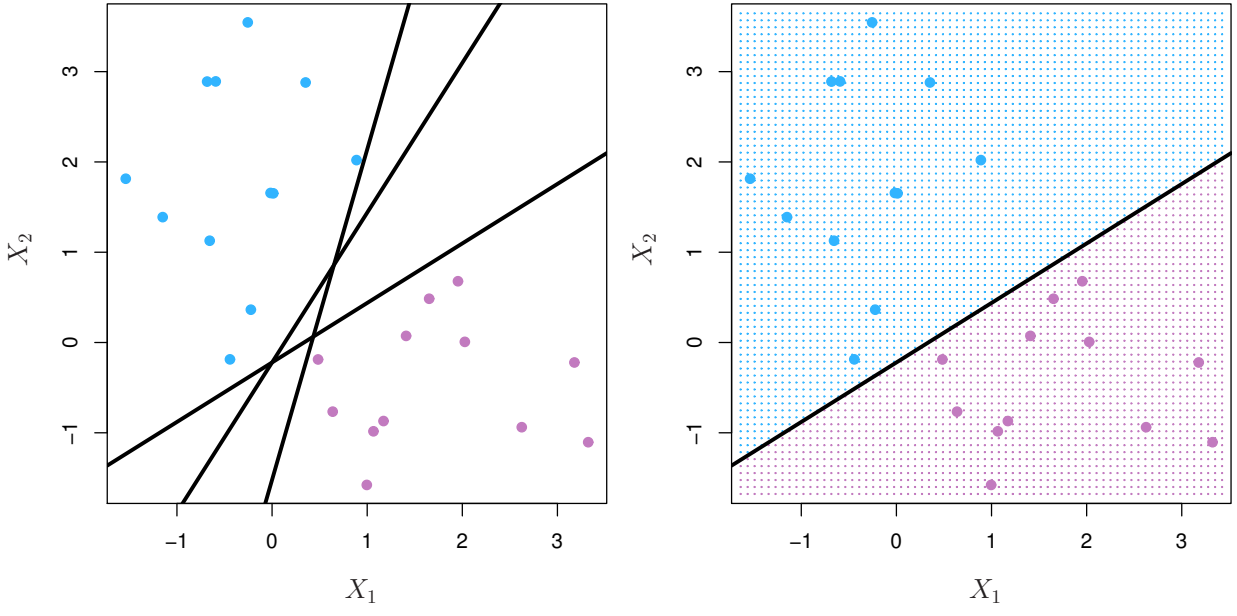
- Como vimos anteriormente, de um lado do hiperplano a função assume valores positivos e do outro negativo;
- Assim, se $f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$, então $f(X) > 0$ para ponto em um lado do hiperplano, e $f(X) < 0$ para pontos do outro lado;



- Note que se codificarmos como $Y_i = +1$ os pontos em azul, e $Y_i = -1$ os pontos rosa, então $Y_i \cdot f(X_i) > 0$ para todo i.
- Mas, em meio a tantos hiperplanos possíveis, qual escolher?

Separando as classes

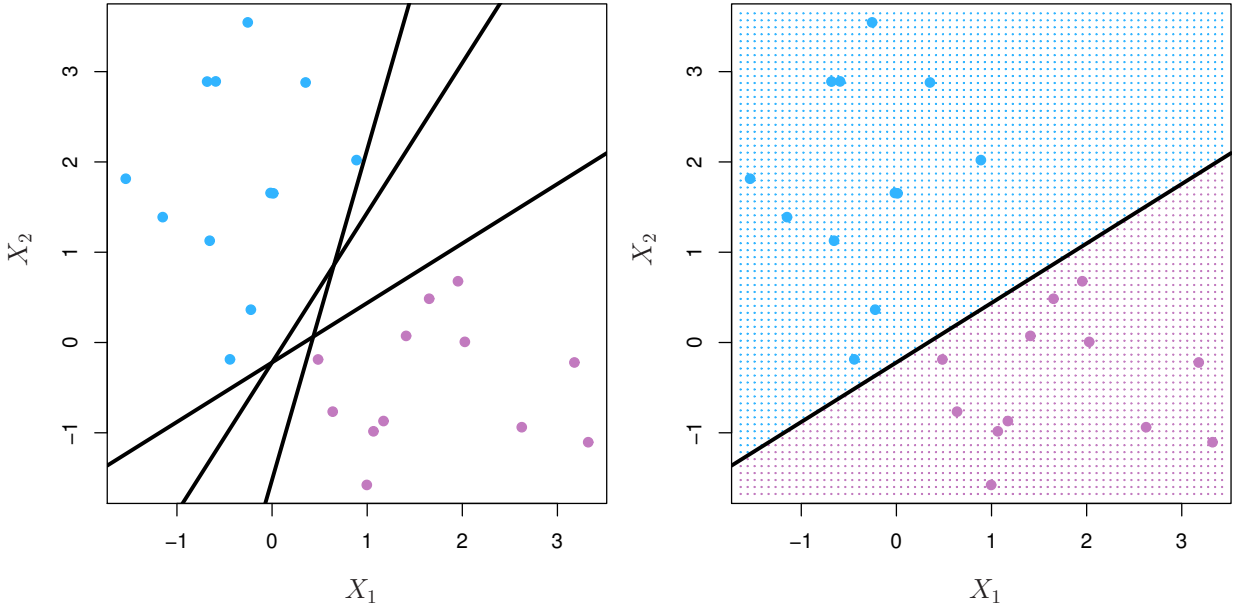
- Como vimos anteriormente, de um lado do hiperplano a função assume valores positivos e do outro negativo;
- Assim, se $f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$, então $f(X) > 0$ para ponto em um lado do hiperplano, e $f(X) < 0$ para pontos do outro lado;



- Note que se codificarmos como $Y_i = +1$ os pontos em azul, e $Y_i = -1$ os pontos rosa, então $Y_i \cdot f(X_i) > 0$ para todo i.
- Mas, em meio a tantos hiperplanos possíveis, qual escolher?

Separando as classes

- Como vimos anteriormente, de um lado do hiperplano a função assume valores positivos e do outro negativo;
- Assim, se $f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$, então $f(X) > 0$ para ponto em um lado do hiperplano, e $f(X) < 0$ para pontos do outro lado;



- Note que se codificarmos como $Y_i = +1$ os pontos em azul, e $Y_i = -1$ os pontos rosa, então $Y_i \cdot f(X_i) > 0$ para todo i.
- Mas, em meio a tantos hiperplanos possíveis, qual escolher?

Maximal Margin Classifier

- Dentre todos os hiperplanos, buscamos aquele que apresenta maior distância entre as margens das duas classes (seria a largura do corredor);

- Veja que não nos interessa a distância de todos os elementos x_i. Basta dos que estão próximos ao hiperplano (estes serão os vetores de suporte).

Maximal Margin Classifier

- Dentre todos os hiperplanos, buscamos aquele que apresenta maior distância entre as margens das duas classes (seria a largura do corredor);

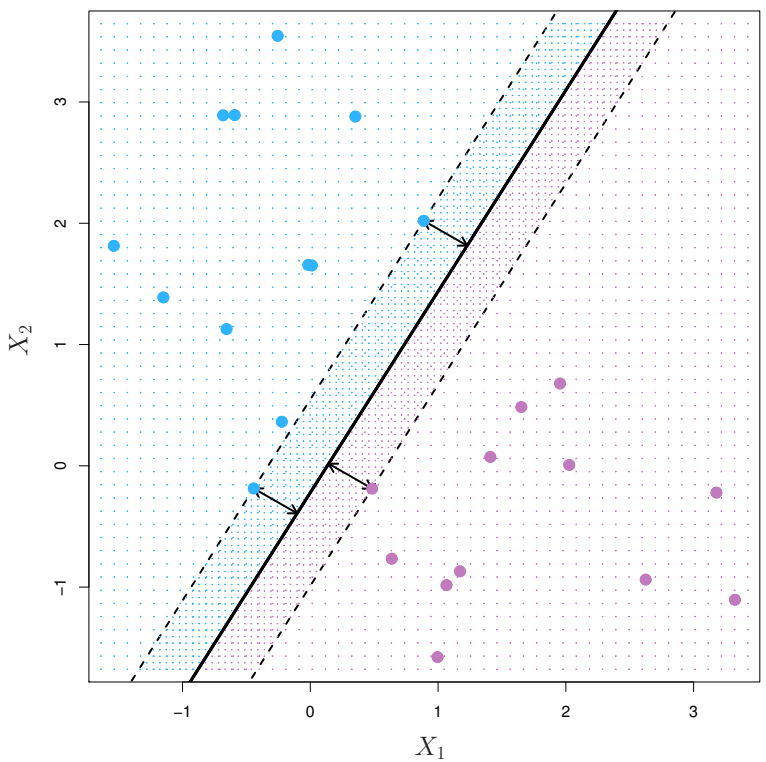
- Veja que não nos interessa a distância de todos os elementos x_i. Basta dos que estão próximos ao hiperplano (estes serão os vetores de suporte).

Maximal Margin Classifier

- Dentre todos os hiperplanos, buscamos aquele que apresenta maior distância entre as margens das duas classes (seria a largura do corredor);

- Veja que não nos interessa a distância de todos os elementos x_i. Basta dos que estão próximos ao hiperplano (estes serão os vetores de suporte).

Problema de otimização com restrição



$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{argmax} \; M, \quad \text{sujeito a} \sum_{j=1}^{p} \beta_j^2 = 1, \quad \text{e}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq M, \forall i = 1 : N$$

Maximal Margin Classifier

- Note que a partir de

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ M, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N.$$

- Podemos reescrever a restrição na forma

$$\frac{1}{\|\boldsymbol{\beta}\|} y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N \quad \text{(que apenas redefine } \beta_0\text{)}.$$

- Ou equivalentemente,

$$y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M \|\boldsymbol{\beta}\|, \forall i = 1 : N.$$

- Tomando, arbitrariamente, $\|\boldsymbol{\beta}\| = \frac{1}{M}$, chega-se em

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ \frac{1}{\|\boldsymbol{\beta}\|}, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq 1, \forall i = 1 : N.$$

- **Voltamos aos Multiplicadores de Lagrange!!**

Maximal Margin Classifier

- Note que a partir de

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ M, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N.$$

- Podemos reescrever a restrição na forma

$$\frac{1}{\|\boldsymbol{\beta}\|} y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N \quad \text{(que apenas redefine } \beta_0\text{)}.$$

- Ou equivalentemente,

$$y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M \|\boldsymbol{\beta}\|, \forall i = 1 : N.$$

- Tomando, arbitrariamente, $\|\boldsymbol{\beta}\| = \frac{1}{M}$, chega-se em

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ \frac{1}{\|\boldsymbol{\beta}\|}, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq 1, \forall i = 1 : N.$$

- **Voltamos aos Multiplicadores de Lagrange!!**

Maximal Margin Classifier

- Note que a partir de

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ M, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N.$$

- Podemos reescrever a restrição na forma

$$\frac{1}{\|\boldsymbol{\beta}\|} y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N \quad \text{(que apenas redefine } \beta_0\text{)}.$$

- Ou equivalentemente,

$$y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M \|\boldsymbol{\beta}\|, \forall i = 1 : N.$$

- Tomando, arbitrariamente, $\|\boldsymbol{\beta}\| = \frac{1}{M}$, chega-se em

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ \frac{1}{\|\boldsymbol{\beta}\|}, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq 1, \forall i = 1 : N.$$

- **Voltamos aos Multiplicadores de Lagrange!!**

Maximal Margin Classifier

- Note que a partir de

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ M, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N.$$

- Podemos reescrever a restrição na forma

$$\frac{1}{\|\boldsymbol{\beta}\|} y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N \quad \text{(que apenas redefine } \beta_0\text{)}.$$

- Ou equivalentemente,

$$y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M \|\boldsymbol{\beta}\|, \forall i = 1 : N.$$

- Tomando, arbitrariamente, $\|\boldsymbol{\beta}\| = \frac{1}{M}$, chega-se em

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ \frac{1}{\|\boldsymbol{\beta}\|}, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq 1, \forall i = 1 : N.$$

- Voltamos aos Multiplicadores de Lagrange!!

Maximal Margin Classifier

- Note que a partir de

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ M, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N.$$

- Podemos reescrever a restrição na forma

$$\frac{1}{\|\boldsymbol{\beta}\|} y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M, \forall i = 1 : N \quad \text{(que apenas redefine } \beta_0\text{)}.$$

- Ou equivalentemente,

$$y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq M \|\boldsymbol{\beta}\|, \forall i = 1 : N.$$

- Tomando, arbitrariamente, $\|\boldsymbol{\beta}\| = \frac{1}{M}$, chega-se em

$$\underset{\beta_0,\boldsymbol{\beta}}{\textit{argmax}}\ \frac{1}{\|\boldsymbol{\beta}\|}, \quad \textbf{sujeito a} \quad y_i(\langle x_i, \boldsymbol{\beta}\rangle + \beta_0) \geq 1, \forall i = 1 : N.$$

- **Voltamos aos Multiplicadores de Lagrange!!**

Voltamos ao problema de otimização restrita

- Utilizando a técnica dos Multiplicadores de Lagrange chega-se em

$$J(\boldsymbol{\beta}, \beta_0, \boldsymbol{\alpha}) = \frac{1}{2} \|\boldsymbol{\beta}\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\langle x_i, \boldsymbol{\beta} \rangle + \beta_0) - 1]. \quad (1)$$

- A resolução das equações $\frac{\partial J}{\partial \boldsymbol{\beta}} = 0$ e $\frac{\partial J}{\partial \beta_0} = 0$ leva ao seguinte problema (mais detalhes no material complementar)

$$\underset{\boldsymbol{\alpha}}{\text{argmax}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle \mathbf{x_i}, \mathbf{x_k} \rangle, \quad \text{com} \begin{cases} \alpha_i \geq 0, \forall i = 1, \ldots, N \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases}$$

- Note que a maximização depende apenas de produtos internos;

- Além disso, $\alpha_i \neq 0$ somente para os vetores nas margens.

Voltamos ao problema de otimização restrita

- Utilizando a técnica dos Multiplicadores de Lagrange chega-se em

$$J(\boldsymbol{\beta}, \beta_0, \boldsymbol{\alpha}) = \frac{1}{2} \|\boldsymbol{\beta}\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\langle x_i, \boldsymbol{\beta} \rangle + \beta_0) - 1]. \quad (1)$$

- A resolução das equações $\frac{\partial J}{\partial \boldsymbol{\beta}} = 0$ e $\frac{\partial J}{\partial \beta_0} = 0$ leva ao seguinte problema (mais detalhes no ▸ material complementar)

$$\underset{\boldsymbol{\alpha}}{\mathit{argmax}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle \mathbf{x_i}, \mathbf{x_k} \rangle, \quad \text{com} \begin{cases} \alpha_i \geq 0, \forall i = 1, \ldots, N \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases}$$

- Note que a maximização depende apenas de produtos internos;

- Além disso, $\alpha_i \neq 0$ somente para os vetores nas margens.

Voltamos ao problema de otimização restrita

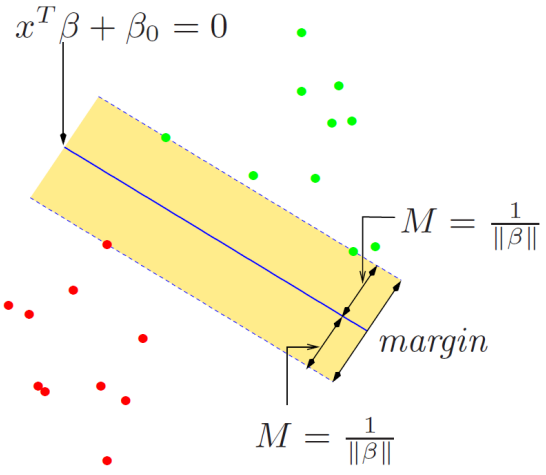
- Utilizando a técnica dos Multiplicadores de Lagrange chega-se em

$$J(\boldsymbol{\beta}, \beta_0, \boldsymbol{\alpha}) = \frac{1}{2} \|\boldsymbol{\beta}\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\langle x_i, \boldsymbol{\beta} \rangle + \beta_0) - 1]. \quad (1)$$

- A resolução das equações $\frac{\partial J}{\partial \boldsymbol{\beta}} = 0$ e $\frac{\partial J}{\partial \beta_0} = 0$ leva ao seguinte problema (mais detalhes no ▸ material complementar)

$$\underset{\boldsymbol{\alpha}}{\text{argmax}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle \mathbf{x_i}, \mathbf{x_k} \rangle, \quad \text{com} \begin{cases} \alpha_i \geq 0, \forall i = 1, \ldots, N \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases}$$

- Note que a maximização depende apenas de produtos internos;
- Além disso, $\alpha_i \neq 0$ somente para os vetores nas margens.



Voltamos ao problema de otimização restrita

- Utilizando a técnica dos Multiplicadores de Lagrange chega-se em

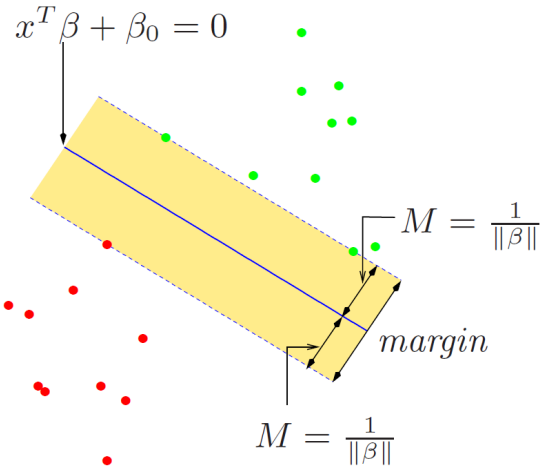
$$J(\boldsymbol{\beta}, \beta_0, \boldsymbol{\alpha}) = \frac{1}{2} \|\boldsymbol{\beta}\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\langle x_i, \boldsymbol{\beta} \rangle + \beta_0) - 1]. \quad (1)$$

- A resolução das equações $\frac{\partial J}{\partial \boldsymbol{\beta}} = 0$ e $\frac{\partial J}{\partial \beta_0} = 0$ leva ao seguinte problema (mais detalhes no ▸ material complementar)

$$\underset{\boldsymbol{\alpha}}{\text{argmax}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle \mathbf{x_i}, \mathbf{x_k} \rangle, \quad \text{com} \begin{cases} \alpha_i \geq 0, \forall i = 1, \ldots, N \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases}$$

- Note que a maximização depende apenas de produtos internos;

- Além disso, $\alpha_i \neq 0$ somente para os vetores nas margens.

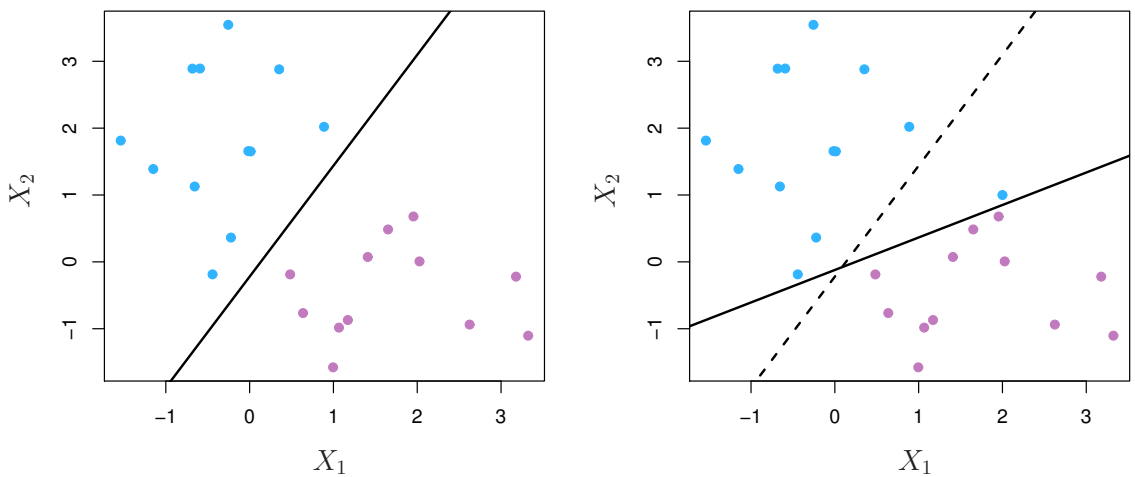


Perturbação nos dados

- A busca por um classificador que separa perfeitamente **todas** as observações dos dados de treinamento torna-o sensível a outliers;
- Mais ainda, o fato de uma observação gerar uma dramática mudança no hiperplano sugere a possível ocorrência de overfit;
- Note que adicionado uma observação em azul acarreta em uma mudança no hiperplano;
- Além disso, em situações reais, é difícil encontrar aplicações cujos dados sejam linearmente separáveis (o **hiperplano não existe**);

Perturbação nos dados

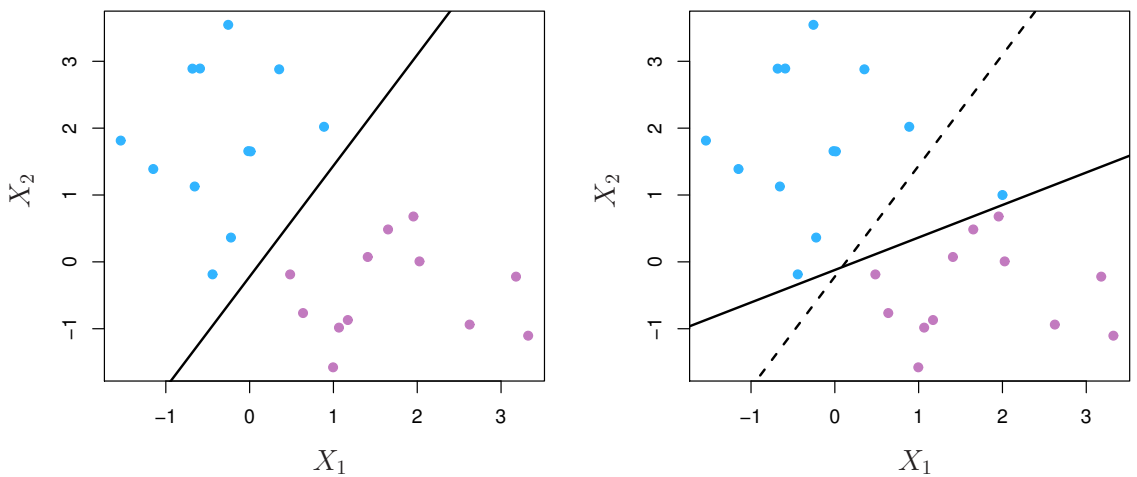
- A busca por um classificador que separa perfeitamente **todas** as observações dos dados de treinamento torna-o sensível a outliers;
- Mais ainda, o fato de uma observação gerar uma dramática mudança no hiperplano sugere a possível ocorrência de overfit;



- Note que adicionado uma observação em azul acarreta em uma mudança no hiperplano;
- Além disso, em situações reais, é difícil encontrar aplicações cujos dados sejam linearmente separáveis (o **hiperplano não existe**);

Perturbação nos dados

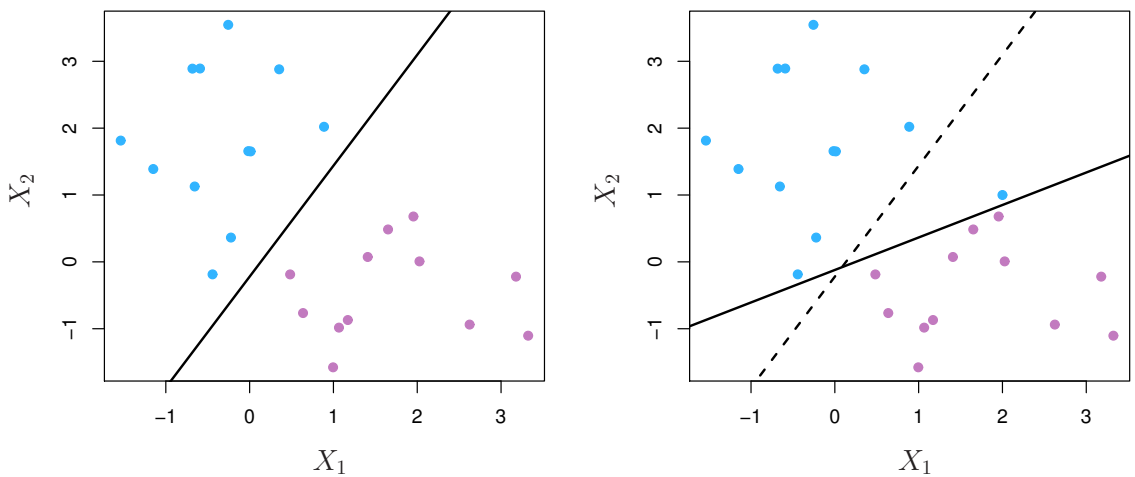
- A busca por um classificador que separa perfeitamente **todas** as observações dos dados de treinamento torna-o sensível a outliers;
- Mais ainda, o fato de uma observação gerar uma dramática mudança no hiperplano sugere a possível ocorrência de overfit;



- Note que adicionado uma observação em azul acarreta em uma mudança no hiperplano;
- Além disso, em situações reais, é difícil encontrar aplicações cujos dados sejam linearmente separáveis (o **hiperplano não existe**);

Perturbação nos dados

- A busca por um classificador que separa perfeitamente **todas** as observações dos dados de treinamento torna-o sensível a outliers;
- Mais ainda, o fato de uma observação gerar uma dramática mudança no hiperplano sugere a possível ocorrência de overfit;



- Note que adicionado uma observação em azul acarreta em uma mudança no hiperplano;
- Além disso, em situações reais, é difícil encontrar aplicações cujos dados sejam linearmente separáveis (o **hiperplano não existe**);

Support Vector Classifier

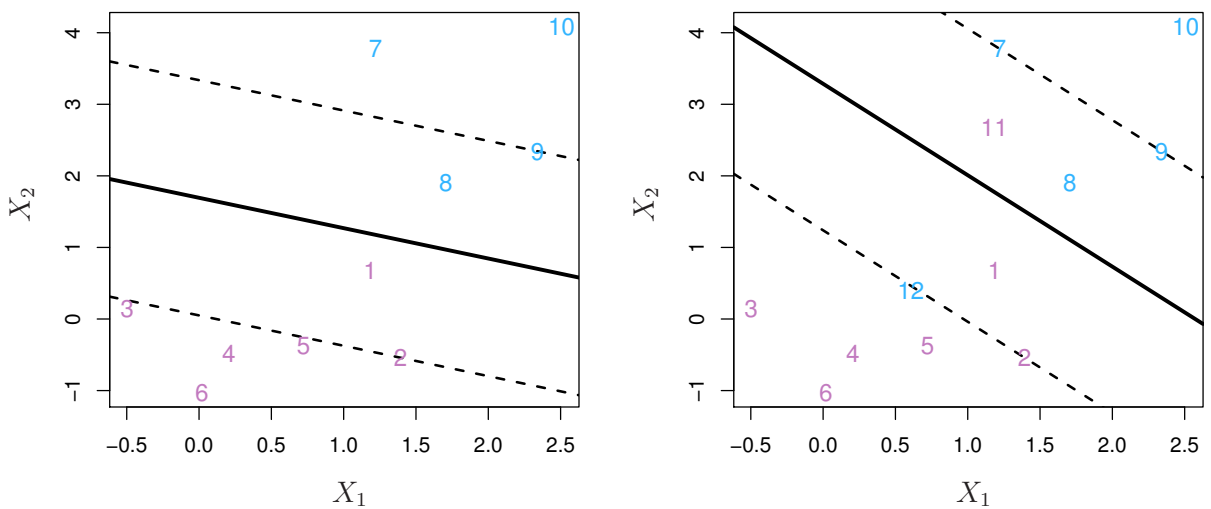
- Diante destes problemas surgiu a ideia de se considerar um classificador que **não separa as classes perfeitamente**, tal que:
  - ⋆ Seja mais robusto a observações individuais;
  - ⋆ Classifique a maior parte dos dados de treinamento.
- O **Support Vector Classifier** (ou **Soft Margin Classifier**) faz isso;
- P. ex. abaixo, os dados 1, 8 e 11 estão do "lado errado" da margem (representada pela linha tracejada).

Support Vector Classifier

- Diante destes problemas surgiu a ideia de se considerar um classificador que **não separa as classes perfeitamente**, tal que:
  - ⋆ Seja mais robusto a observações individuais;
  - ⋆ Classifique a maior parte dos dados de treinamento.
- O **Support Vector Classifier** (ou **Soft Margin Classifier**) faz isso;
- P. ex. abaixo, os dados 1, 8 e 11 estão do "lado errado" da margem (representada pela linha tracejada).

Support Vector Classifier

- Diante destes problemas surgiu a ideia de se considerar um classificador que **não separa as classes perfeitamente**, tal que:
  - ⋆ Seja mais robusto a observações individuais;
  - ⋆ Classifique a maior parte dos dados de treinamento.
- O **Support Vector Classifier** (ou **Soft Margin Classifier**) faz isso;
- P. ex. abaixo, os dados 1, 8 e 11 estão do "lado errado" da margem (representada pela linha tracejada).



Support Vector Classifier

- O problema de otimização fica então

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{argmax}} \quad M, \quad \text{sujeito a} \quad \sum_{j=1}^{p} \beta_j^2 = 1.$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq \underbrace{M(1-\epsilon_i)}_{\text{violação da margem}}$$

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

- Novamente, queremos que a margem M seja a maior possível;
- C é o **tuning parameter** (decide o quanto aceitamos errar);
- E ϵ_i são as **variáveis de folga**:
  - ⋆ Se $\epsilon_i = 0$, então a i-ésima obs. está no lado correto da margem;
  - ⋆ Se $0 < \epsilon_i \leq 1$, então a i-ésima obs. está no lado errado da margem;
  - ⋆ Se $\epsilon_i > 1$, então a i-ésima obs. está no lado errado do hiperplano.

Support Vector Classifier

- O problema de otimização fica então

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{argmax}} \quad M, \quad \text{sujeito a} \quad \sum_{j=1}^{p} \beta_j^2 = 1.$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq \underbrace{M(1-\epsilon_i)}_{\text{violação da margem}}$$

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

- Novamente, queremos que a margem M seja a maior possível;
- C é o **tuning parameter** (decide o quanto aceitamos errar);
- E ϵ_i são as **variáveis de folga**:
  - ⋆ Se $\epsilon_i = 0$, então a i-ésima obs. está no lado correto da margem;
  - ⋆ Se $0 < \epsilon_i \leq 1$, então a i-ésima obs. está no lado errado da margem;
  - ⋆ Se $\epsilon_i > 1$, então a i-ésima obs. está no lado errado do hiperplano.

Support Vector Classifier

- O problema de otimização fica então

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{argmax}} \quad M, \quad \text{sujeito a} \quad \sum_{j=1}^{p} \beta_j^2 = 1.$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq \underbrace{M(1-\epsilon_i)}_{\text{violação da margem}}$$

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

- Novamente, queremos que a margem M seja a maior possível;
- C é o **tuning parameter** (decide o quanto aceitamos errar);
- E ϵ_i são as **variáveis de folga**:
  - ⋆ Se $\epsilon_i = 0$, então a i-ésima obs. está no lado correto da margem;
  - ⋆ Se $0 < \epsilon_i \leq 1$, então a i-ésima obs. está no lado errado da margem;
  - ⋆ Se $\epsilon_i > 1$, então a i-ésima obs. está no lado errado do hiperplano.

Support Vector Classifier

- O problema de otimização fica então

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{argmax}} \quad M, \quad \text{sujeito a} \quad \sum_{j=1}^{p} \beta_j^2 = 1.$$

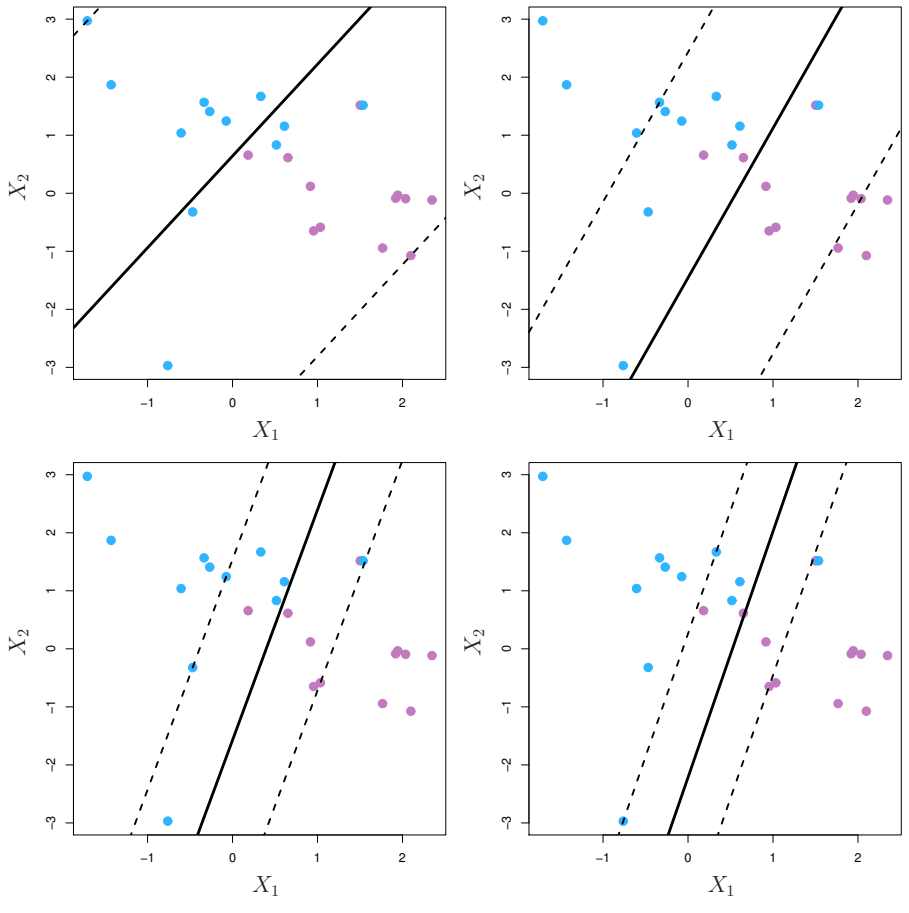
$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq \underbrace{M(1-\epsilon_i)}_{\text{violação da margem}}$$

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

- Novamente, queremos que a margem M seja a maior possível;
- C é o **tuning parameter** (decide o quanto aceitamos errar);
- E ϵ_i são as **variáveis de folga**:
  - ⋆ Se $\epsilon_i = 0$, então a i-ésima obs. está no lado correto da margem;
  - ⋆ Se $0 < \epsilon_i \leq 1$, então a i-ésima obs. está no lado errado da margem;
  - ⋆ Se $\epsilon_i > 1$, então a i-ésima obs. está no lado errado do hiperplano.

Variando o tuning parameter, C

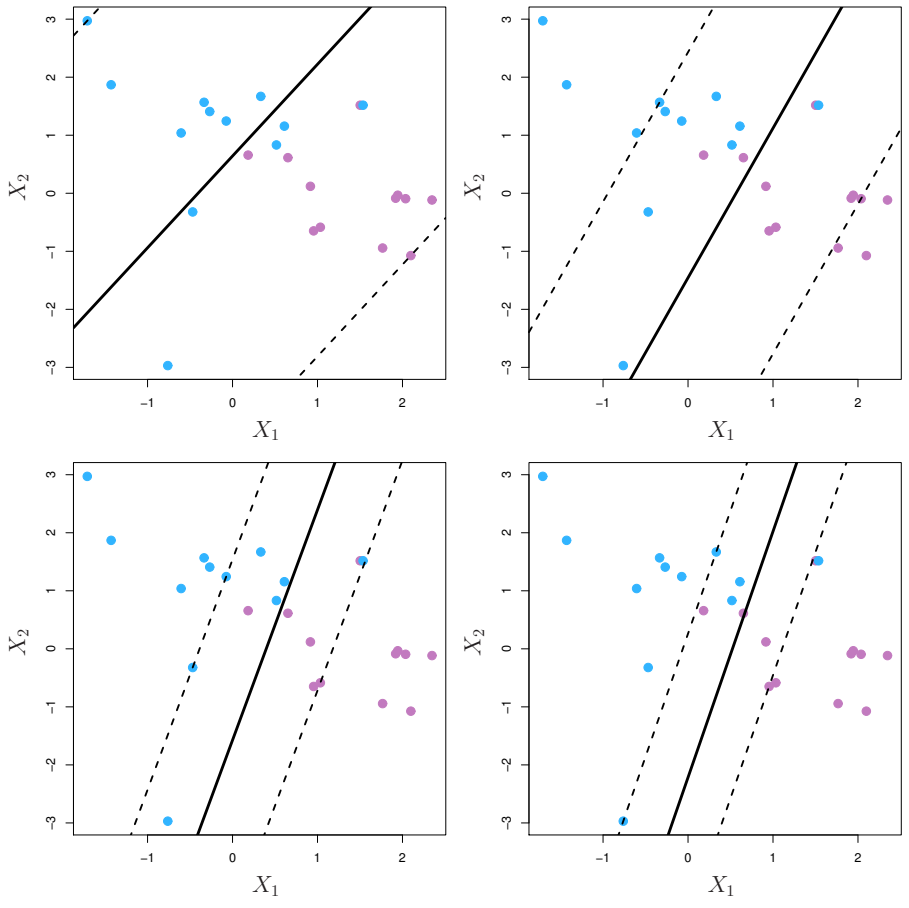
- Abaixo, *support vector classifier* ajustado considerando quatro diferentes valores de C.



- Quando C é grande, temos alta tolerância para as observações estarem no lado errado da margem (a margem será mais larga).

Variando o tuning parameter, C

- Abaixo, *support vector classifier* ajustado considerando quatro diferentes valores de C.



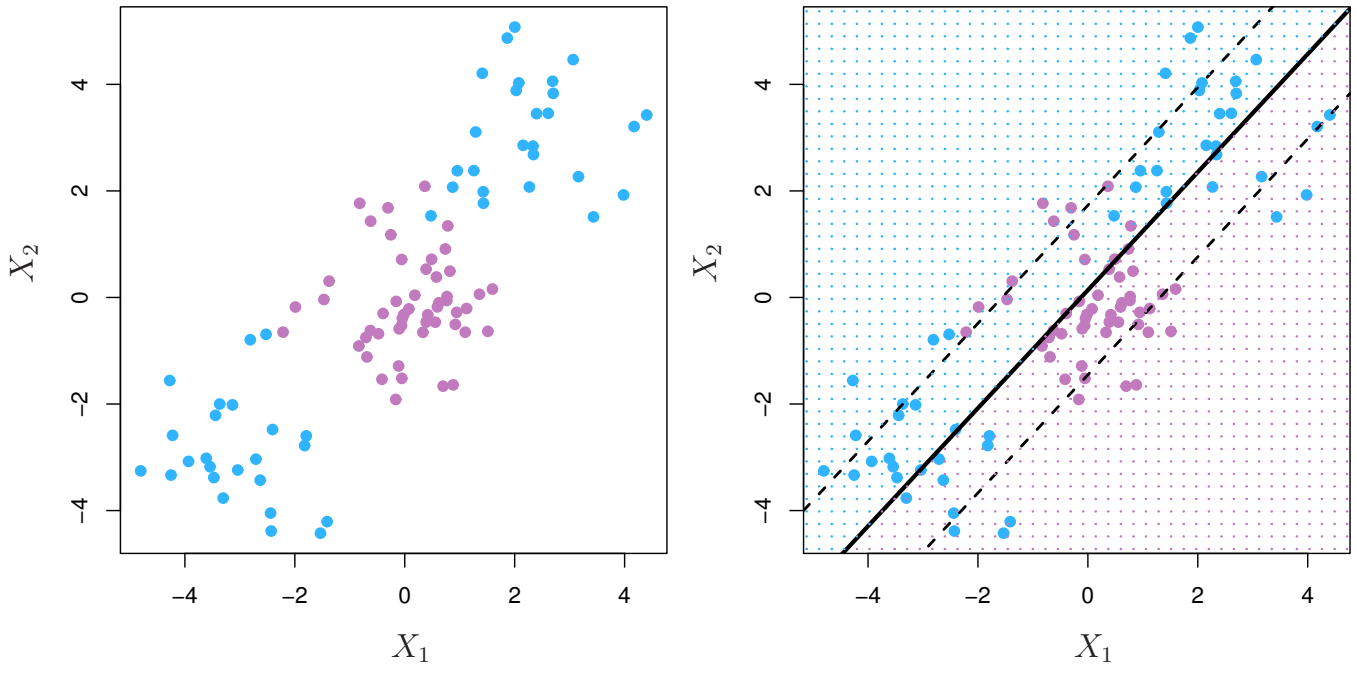
- Quando C é grande, temos alta tolerância para as observações estarem no lado errado da margem (a margem será mais larga).

Limite linear pode falhar

- O *Support Vector Classifier* é muito útil quando o limite entre as duas classes é linear;
- Entretanto, por vezes nos defrontamos com **limites de classes não lineares**.
- Neste caso, o desempenho do método não é satisfatório. Então, o que fazer?

Limite linear pode falhar

- O *Support Vector Classifier* é muito útil quando o limite entre as duas classes é linear;
- Entretanto, por vezes nos defrontamos com **limites de classes não lineares**.

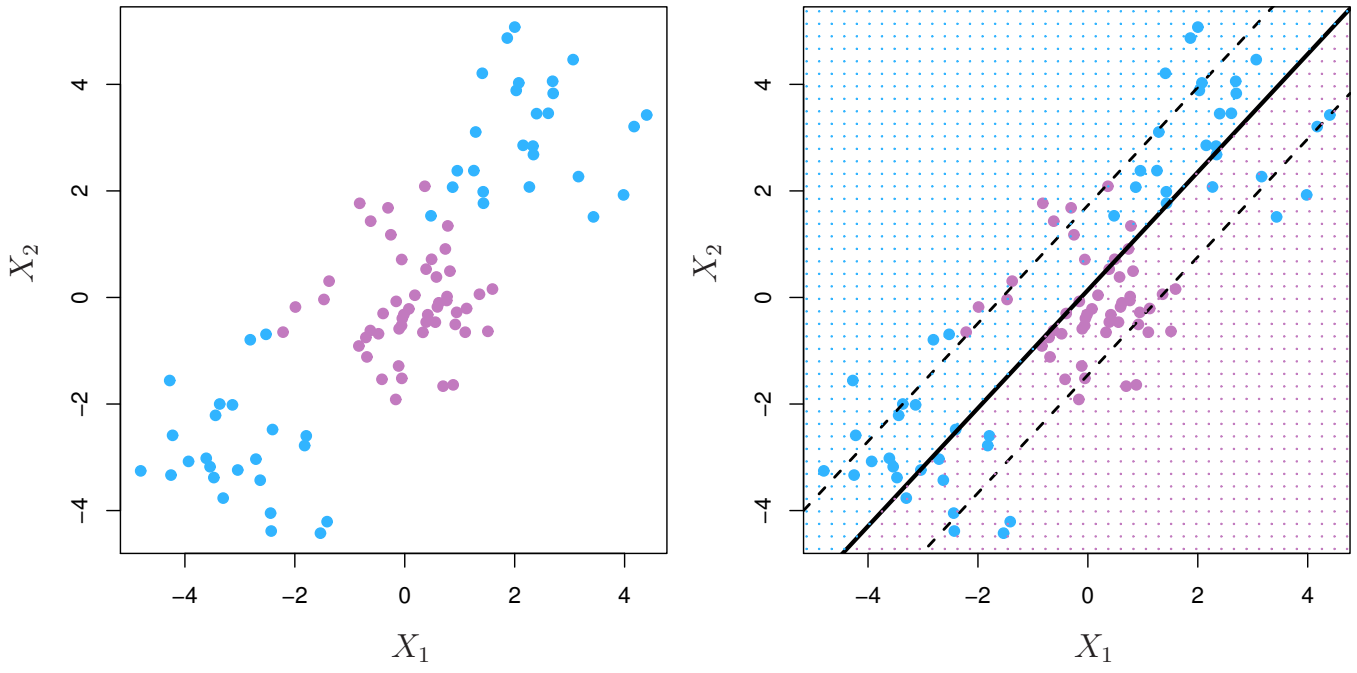


- Neste caso, o desempenho do método não é satisfatório. Então, o que fazer?

Limite linear pode falhar

- O *Support Vector Classifier* é muito útil quando o limite entre as duas classes é linear;

- Entretanto, por vezes nos defrontamos com **limites de classes não lineares**.

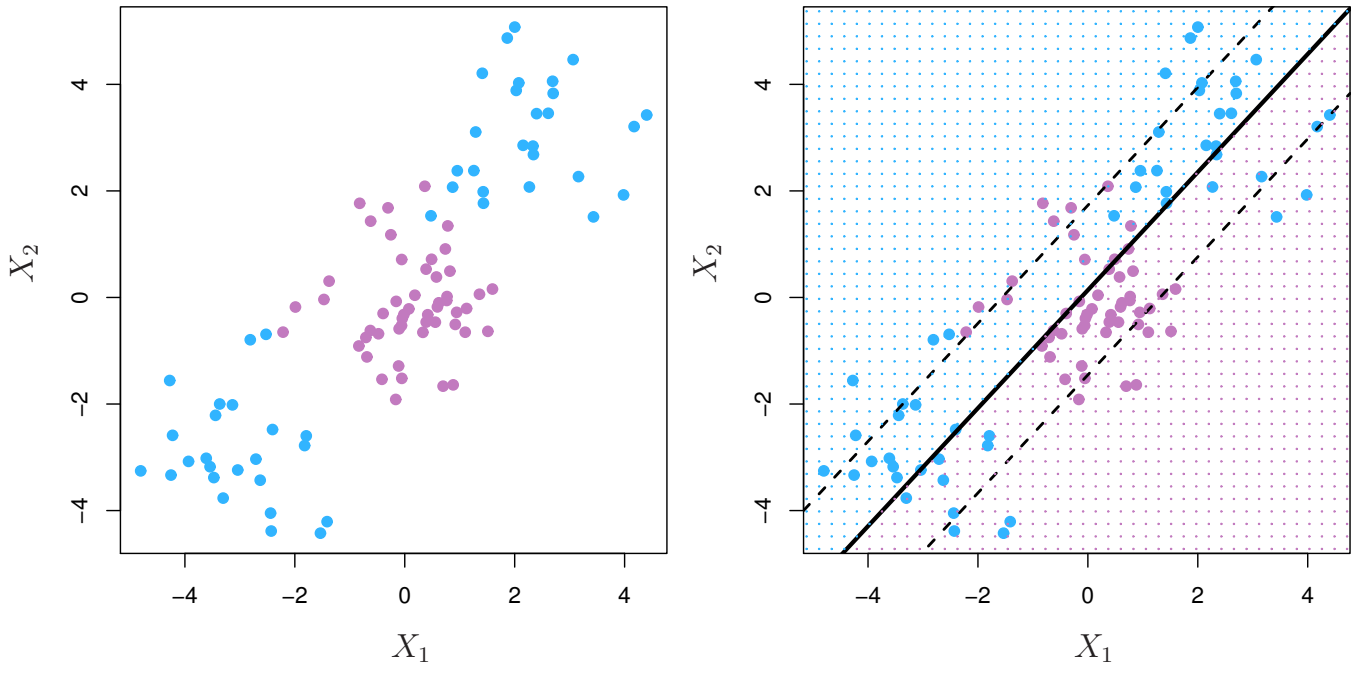


- Neste caso, o desempenho do método não é satisfatório. Então, o que fazer?

Limite linear pode falhar

- O *Support Vector Classifier* é muito útil quando o limite entre as duas classes é linear;

- Entretanto, por vezes nos defrontamos com **limites de classes não lineares**.



- Neste caso, o desempenho do método não é satisfatório. Então, o que fazer? **Truque do kernel e a expansão das características**

Truque do Kernel

- Voltando ao problema de otimização

$$\underset{\boldsymbol{\alpha}}{argmax} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle \mathbf{x_i}, \mathbf{x_k} \rangle.$$

- Note que a atualização pelos dados ocorre somente através de $\langle \mathbf{x_i}, \mathbf{x_k} \rangle$. A ideia então é expandir as características $\mathbf{x}$ através de funções Kernel

$$K(\mathbf{x_i}, \mathbf{x_k}) \stackrel{def}{=} \langle \mathbf{\Phi}(\mathbf{x_i}), \mathbf{\Phi}(\mathbf{x_k}) \rangle$$

- Por exemplo, considere o espaço de características com x_1 e x_2

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_k) &= (1 + \langle \mathbf{x}_i, \mathbf{x}_k \rangle)^2 \\ &= 1 + 2\mathbf{x}_{i1}\mathbf{x}_{k1} + 2\mathbf{x}_{i2}\mathbf{x}_{k2} + (\mathbf{x}_{i1}\mathbf{x}_{k1})^2 + (\mathbf{x}_{i2}\mathbf{x}_{k2})^2 + 2\mathbf{x}_{i1}\mathbf{x}_{k1}\mathbf{x}_{i2}\mathbf{x}_{k2} \end{aligned}$$

- Escolhendo

$$\Phi_1(x) = 1, \Phi_2(x) = \sqrt{2}x_1, \Phi_3(x) = \sqrt{2}x_2,$$
$$\Phi_4(x) = x_1^2, \Phi_5(x) = x_2^2 e \Phi_6(x) = \sqrt{2}x_1x_2$$

- Temos, $K(\mathbf{x}_i, \mathbf{x}_k) = \langle \mathbf{\Phi}(\mathbf{x_i}), \mathbf{\Phi}(\mathbf{x_k}) \rangle$.

Truque do Kernel

- Voltando ao problema de otimização

$$\underset{\boldsymbol{\alpha}}{argmax} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle \mathbf{x_i}, \mathbf{x_k} \rangle.$$

- Note que a atualização pelos dados ocorre somente através de $\langle \mathbf{x_i}, \mathbf{x_k} \rangle$. A ideia então é expandir as características $\mathbf{x}$ através de funções Kernel

$$K(\mathbf{x_i}, \mathbf{x_k}) \stackrel{def}{=} \langle \mathbf{\Phi}(\mathbf{x_i}), \mathbf{\Phi}(\mathbf{x_k}) \rangle$$

- Por exemplo, considere o espaço de características com x_1 e x_2

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_k) &= (1 + \langle \mathbf{x}_i, \mathbf{x}_k \rangle)^2 \\ &= 1 + 2\mathbf{x}_{i1}\mathbf{x}_{k1} + 2\mathbf{x}_{i2}\mathbf{x}_{k2} + (\mathbf{x}_{i1}\mathbf{x}_{k1})^2 + (\mathbf{x}_{i2}\mathbf{x}_{k2})^2 + 2\mathbf{x}_{i1}\mathbf{x}_{k1}\mathbf{x}_{i2}\mathbf{x}_{k2} \end{aligned}$$

- Escolhendo

$$\Phi_1(x) = 1, \Phi_2(x) = \sqrt{2}x_1, \Phi_3(x) = \sqrt{2}x_2,$$
$$\Phi_4(x) = x_1^2, \Phi_5(x) = x_2^2 e \Phi_6(x) = \sqrt{2}x_1x_2$$

- Temos, $K(\mathbf{x}_i, \mathbf{x}_k) = \langle \mathbf{\Phi}(\mathbf{x_i}), \mathbf{\Phi}(\mathbf{x_k}) \rangle$.

Truque do Kernel

- Voltando ao problema de otimização

$$\underset{\boldsymbol{\alpha}}{argmax} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle \mathbf{x_i}, \mathbf{x_k} \rangle.$$

- Note que a atualização pelos dados ocorre somente através de $\langle \mathbf{x_i}, \mathbf{x_k} \rangle$. A ideia então é expandir as características $\mathbf{x}$ através de funções Kernel

$$K(\mathbf{x_i}, \mathbf{x_k}) \stackrel{def}{=} \langle \mathbf{\Phi}(\mathbf{x_i}), \mathbf{\Phi}(\mathbf{x_k}) \rangle$$

- Por exemplo, considere o espaço de características com x_1 e x_2

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_k) &= (1 + \langle \mathbf{x}_i, \mathbf{x}_k \rangle)^2 \\ &= 1 + 2\mathbf{x}_{i1}\mathbf{x}_{k1} + 2\mathbf{x}_{i2}\mathbf{x}_{k2} + (\mathbf{x}_{i1}\mathbf{x}_{k1})^2 + (\mathbf{x}_{i2}\mathbf{x}_{k2})^2 + 2\mathbf{x}_{i1}\mathbf{x}_{k1}\mathbf{x}_{i2}\mathbf{x}_{k2} \end{aligned}$$

- Escolhendo

$$\Phi_1(x) = 1, \Phi_2(x) = \sqrt{2}x_1, \Phi_3(x) = \sqrt{2}x_2,$$
$$\Phi_4(x) = x_1^2, \Phi_5(x) = x_2^2 e \Phi_6(x) = \sqrt{2}x_1x_2$$

- Temos, $K(\mathbf{x}_i, \mathbf{x}_k) = \langle \mathbf{\Phi}(\mathbf{x_i}), \mathbf{\Phi}(\mathbf{x_k}) \rangle$.

Truque do Kernel

- Voltando ao problema de otimização

$$\underset{\boldsymbol{\alpha}}{argmax} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \langle \mathbf{x_i}, \mathbf{x_k} \rangle.$$

- Note que a atualização pelos dados ocorre somente através de $\langle \mathbf{x_i}, \mathbf{x_k} \rangle$. A ideia então é expandir as características $\mathbf{x}$ através de funções Kernel

$$K(\mathbf{x_i}, \mathbf{x_k}) \stackrel{def}{=} \langle \mathbf{\Phi}(\mathbf{x_i}), \mathbf{\Phi}(\mathbf{x_k}) \rangle$$

- Por exemplo, considere o espaço de características com x_1 e x_2

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_k) &= (1 + \langle \mathbf{x}_i, \mathbf{x}_k \rangle)^2 \\ &= 1 + 2\mathbf{x}_{i1}\mathbf{x}_{k1} + 2\mathbf{x}_{i2}\mathbf{x}_{k2} + (\mathbf{x}_{i1}\mathbf{x}_{k1})^2 + (\mathbf{x}_{i2}\mathbf{x}_{k2})^2 + 2\mathbf{x}_{i1}\mathbf{x}_{k1}\mathbf{x}_{i2}\mathbf{x}_{k2} \end{aligned}$$

- Escolhendo

$$\Phi_1(x) = 1, \Phi_2(x) = \sqrt{2}x_1, \Phi_3(x) = \sqrt{2}x_2,$$
$$\Phi_4(x) = x_1^2, \Phi_5(x) = x_2^2 e \Phi_6(x) = \sqrt{2}x_1x_2$$

- Temos, $K(\mathbf{x}_i, \mathbf{x}_k) = \langle \mathbf{\Phi}(\mathbf{x_i}), \mathbf{\Phi}(\mathbf{x_k}) \rangle$.

Truque do Kernel

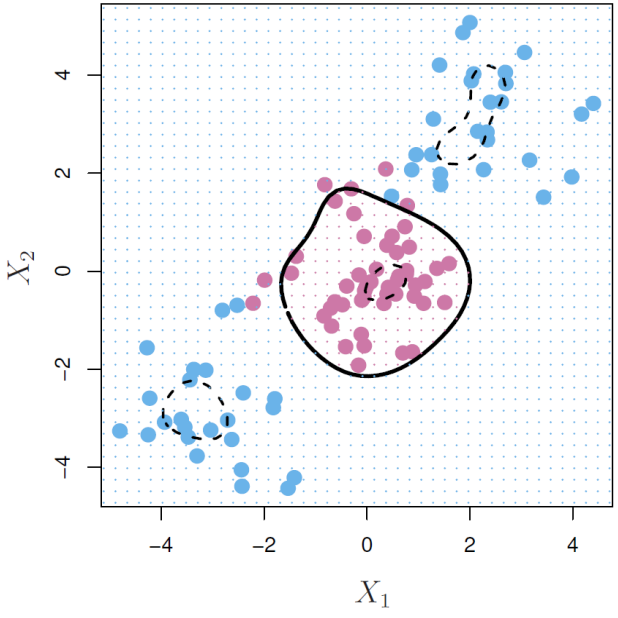
- Lembrando que $||x_i - x_k||^2 = \langle x_i, x_i \rangle + \langle x_k, x_k \rangle - 2\langle x_i, x_k \rangle$, abaixo alguns exemplos de Kernel

  - ⋆ **Kernel linear:** $K(x_i, x_k) = \langle x_i, x_k \rangle$;
  - ⋆ **Kernel gaussiano:** $K(x_i, x_k) = exp\left(-\gamma||x_i - x_k||^2\right)$;
  - ⋆ **Kernel exponencial:** $K(x_i, x_k) = exp\left(-\gamma||x_i - x_k||\right)$;
  - ⋆ **Kernel polinomial:** $K(x_i, x_k) = (p + \langle x_i, x_k \rangle)^q$;
  - ⋆ **Kernel híbrido:** $K(x_i, x_k) = (p + \langle x_i, x_k \rangle)^q \, exp\left(-\gamma||x_i - x_k||^2\right)$;
  - ⋆ **Kernel sigmoidal:** $K(x_i, x_k) = tanh\left(k\langle x_i, x_k \rangle - \delta\right)$;

- Os parâmetros que devem ser determinados pelo usuário.

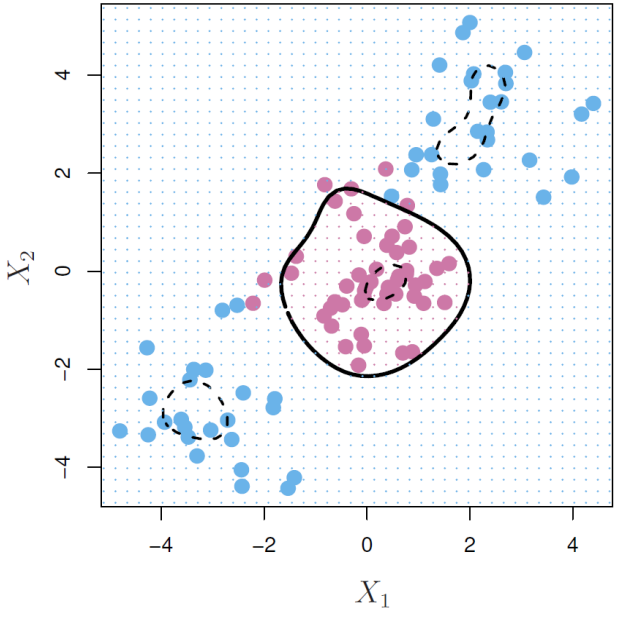
Exemplo de aplicação do Kernel gaussiano

- Seja $\mathbf{x}^* = (\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_p^*)^t$ a observação do teste;
- Se $\mathbf{x}^*$ for distante (distância euclidiana) de $\mathbf{x}_i$ (obs. de treinamento), então $\sum_{j=1}^{p} \left(x_j^* - x_{ij}\right)^2$ será grande;
- $K(x_i, x_k) = exp\left(-\gamma ||x_i - x_k||^2\right)$ será pequeno;
- E $\mathbf{x}_i$ não terá influência sobre $\mathbf{x}^*$;
- Isto significa que apenas observações nas proximidades do treinamento tem um efeito sobre o rótulo da classe (um comportamento local).



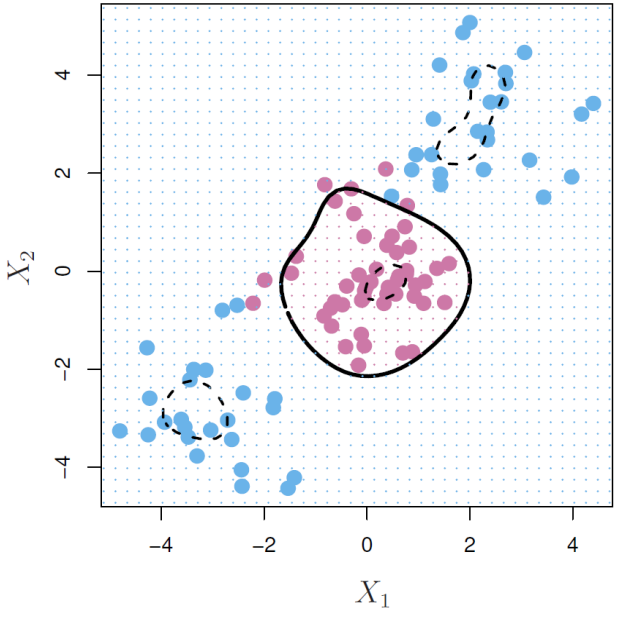
Exemplo de aplicação do Kernel gaussiano

- Seja $\mathbf{x}^* = (\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_p^*)^t$ a observação do teste;
- Se $\mathbf{x}^*$ for distante (distância euclidiana) de $\mathbf{x}_i$ (obs. de treinamento), então $\sum_{j=1}^{p} \left(x_j^* - x_{ij}\right)^2$ será grande;
- $K(x_i, x_k) = exp\left(-\gamma||x_i - x_k||^2\right)$ será pequeno;
- E $\mathbf{x}_i$ não terá influência sobre $\mathbf{x}^*$;
- Isto significa que apenas observações nas proximidades do treinamento tem um efeito sobre o rótulo da classe (um comportamento local).



Exemplo de aplicação do Kernel gaussiano

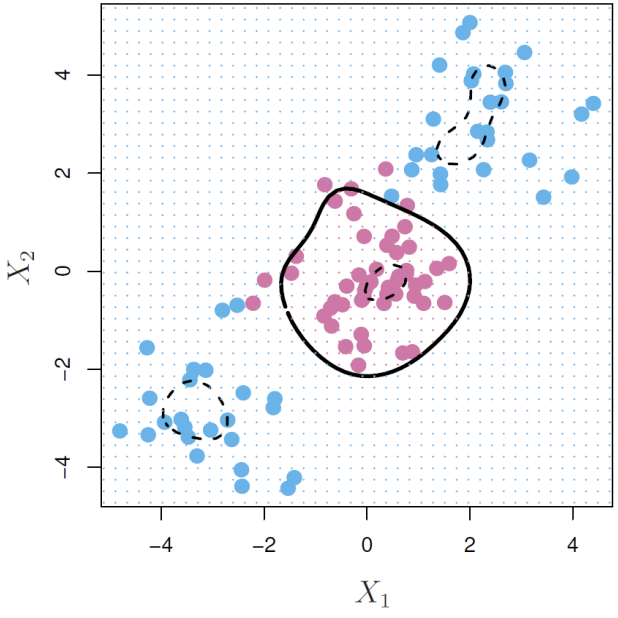
- Seja $\mathbf{x}^* = (\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_p^*)^t$ a observação do teste;
- Se $\mathbf{x}^*$ for distante (distância euclidiana) de $\mathbf{x}_i$ (obs. de treinamento), então $\sum_{j=1}^{p} \left(x_j^* - x_{ij}\right)^2$ será grande;
- $K(x_i, x_k) = exp\left(-\gamma||x_i - x_k||^2\right)$ será pequeno;
- E $\mathbf{x}_i$ não terá influência sobre $\mathbf{x}^*$;



- Isto significa que apenas observações nas proximidades do treinamento tem um efeito sobre o rótulo da classe (um comportamento local).

Exemplo de aplicação do Kernel gaussiano

- Seja $\mathbf{x}^* = (\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_p^*)^t$ a observação do teste;
- Se $\mathbf{x}^*$ for distante (distância euclidiana) de $\mathbf{x}_i$ (obs. de treinamento), então $\sum_{j=1}^{p} \left(x_j^* - x_{ij}\right)^2$ será grande;
- $K(x_i, x_k) = exp\left(-\gamma||x_i - x_k||^2\right)$ será pequeno;
- E $\mathbf{x}_i$ não terá influência sobre $\mathbf{x}^*$;



- Isto significa que apenas observações nas proximidades do treinamento tem um efeito sobre o rótulo da classe (um comportamento local).

Ideia da expansão das características

Exemplo 1

- Considere as classes na seguinte situação

- Não são linearmente separáveis. Mas incluindo a característica $x_2 = x_1^2$ (expandindo x_1) temos

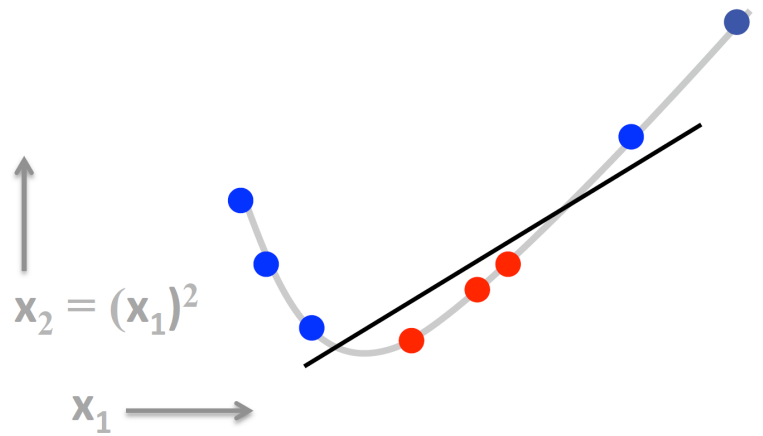
- São linearmente separáveis com a nova variável!

Ideia da expansão das características

Exemplo 1

- Considere as classes na seguinte situação

- Não são linearmente separáveis. Mas incluindo a característica $x_2 = x_1^2$ (expandindo x_1) temos



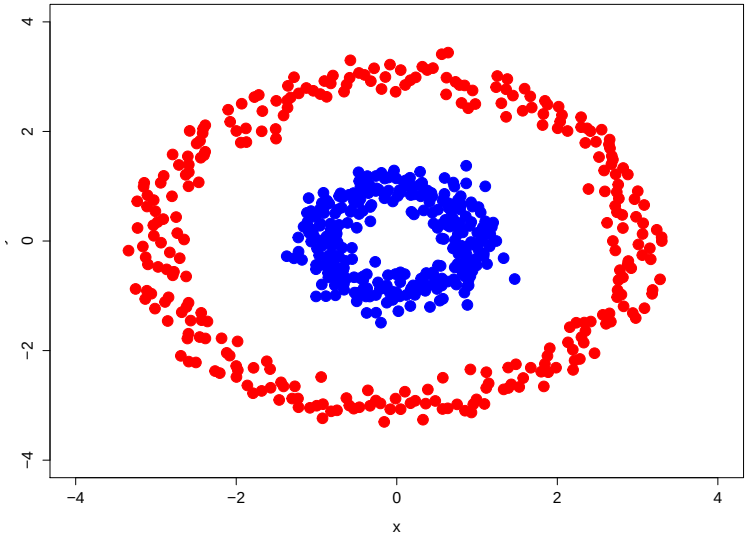
- São linearmente separáveis com a nova variável!

Ideia da expansão das características

Exemplo 2

- Temos um padrão linearmente não separável;

```
> dc<-read.table("double_circle.txt")

> plot(dc[,-3],col=c(rep('blue',300),
       rep('red',300)),xlim=c(-4,4),
 ylim=c(-4,4),pch=19,cex=2)
```



- Mas, expandindo o espaço torna-se simples a classificação;

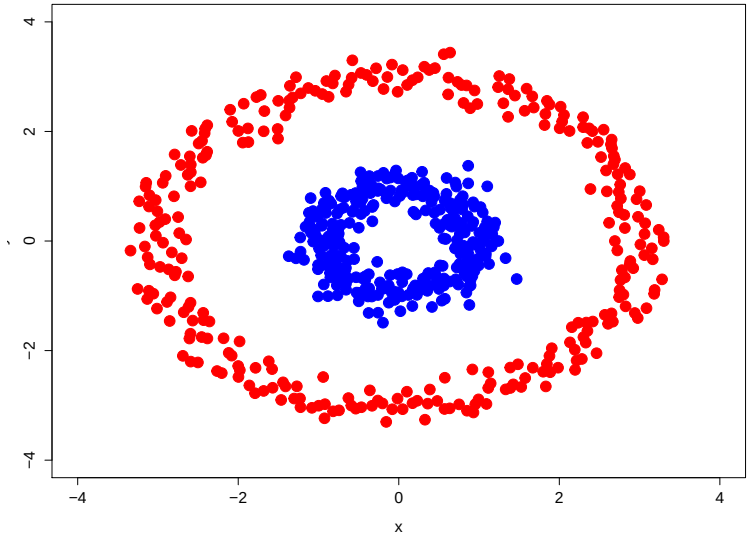
```
library(scatterplot3d)

> scatterplot3d(dc$x,dc$y,zdc,color=c(rep('blue',300),rep('red',300)),
                pch=19,cex.symbols=1.5)

> scatterplot3d(dc$x,dc$y,zdc,color=c(rep('blue',300),rep('red',300)),
                pch=19,cex.symbols=1.5,angle=0)
```

Ideia da expansão das características

Exemplo 2

- Temos um padrão linearmente não separável;

```
> dc<-read.table("double_circle.txt")

> plot(dc[,-3],col=c(rep('blue',300),
       rep('red',300)),xlim=c(-4,4),
 ylim=c(-4,4),pch=19,cex=2)
```

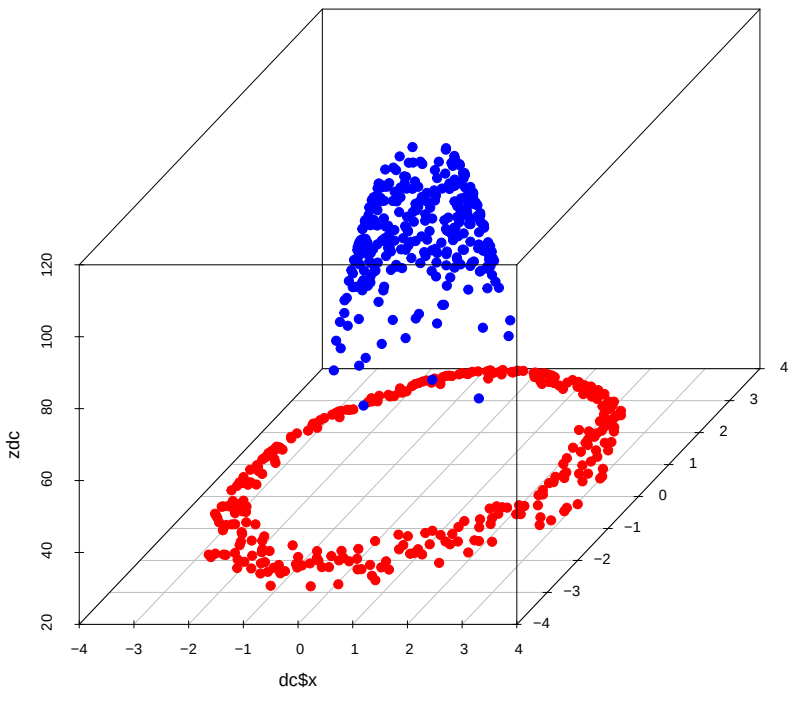


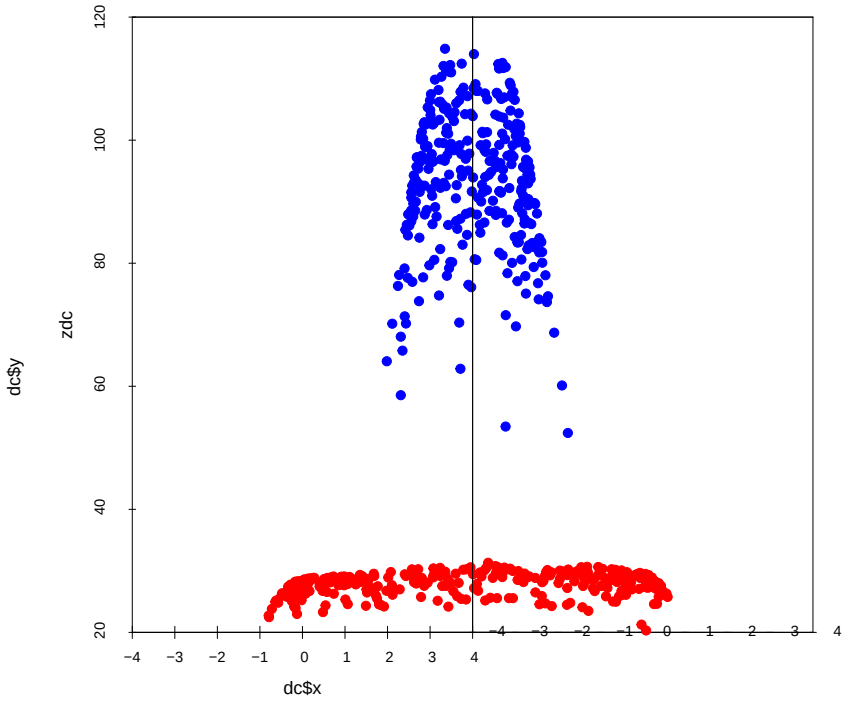
- Mas, expandindo o espaço torna-se simples a classificação;

```
library(scatterplot3d)

> scatterplot3d(dc$x,dc$y,zdc,color=c(rep('blue',300),rep('red',300)),
                pch=19,cex.symbols=1.5)

> scatterplot3d(dc$x,dc$y,zdc,color=c(rep('blue',300),rep('red',300)),
                pch=19,cex.symbols=1.5,angle=0)
```

Ideia da expansão das características

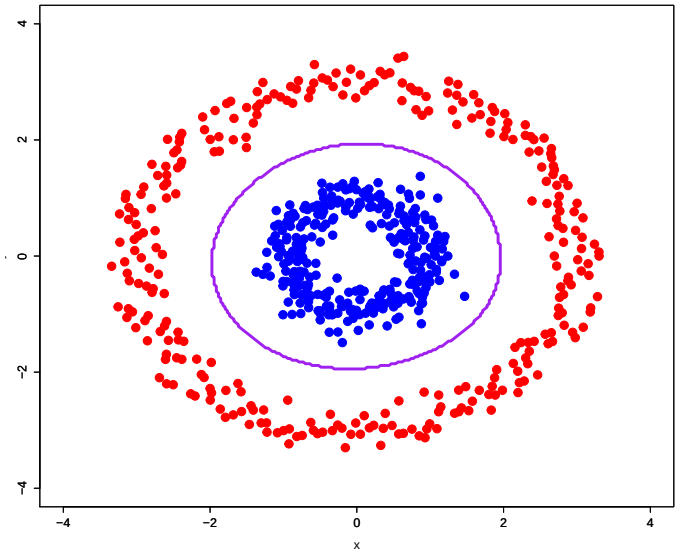
- Agora, podemos classificar a partir de um plano horizontal.





Ideia da expansão das características

```
> library(e1071)

> dc$label<-as.factor(dc$label)

> dc.svm<-svm(label~.,dc)

> px<-seq(-4,4,0.03); py<-seq(-4,4,0.03)

> pgrid<-expand.grid(px,py)

> names(pgrid)<-names(dc)[-3]
```




```
> plot(dc[,-3],col=c(rep('blue',300),rep('red',300)),xlim=c(-4,4),
      ylim=c(-4,4),pch=19,cex=2)

> par(new=T)

> contour(px,py,array(predict(dc.svm,newdata=pgrid),dim = c(length(px),
        length(py))),col='purple',lwd=5,levels=0.5,drawlabels=F,
        xlim=c(-4,4),ylim=c(-4,4))
```

Ideia da expansão das características

Exemplo 3

- Suponha que utilizemos $(X_1, X_1^2, X_2, X_2^2, X_1X_2)$, ao invés de (X_1, X_2);
- A fronteira de decisão ficará

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

- Que continua linear para essas novas variáveis;
- Mas, conduz à decisão não linear no espaço original.
- Se fosse um polinômio cubico sairíamos de 2 para 9 variáveis!

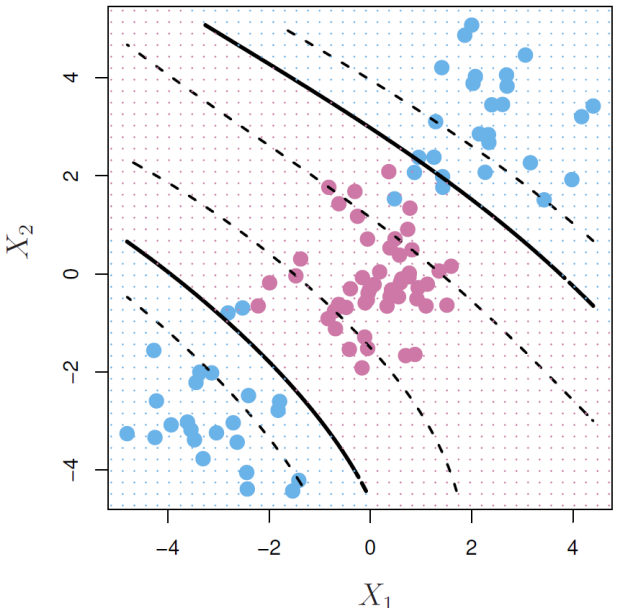
Ideia da expansão das características

Exemplo 3

- Suponha que utilizemos $(X_1, X_1^2, X_2, X_2^2, X_1X_2)$, ao invés de (X_1, X_2);
- A fronteira de decisão ficará

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

- Que continua linear para essas novas variáveis;
- Mas, conduz à decisão não linear no espaço original.
- Se fosse um polinômio cubico sairíamos de 2 para 9 variáveis!



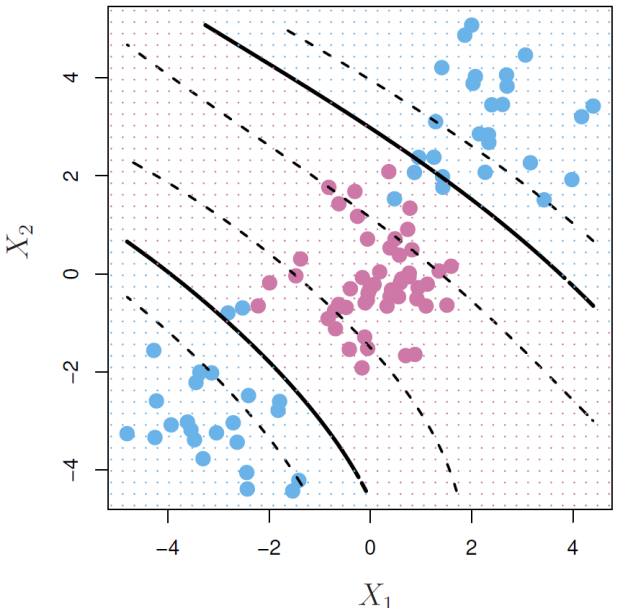
Ideia da expansão das características

Exemplo 3

- Suponha que utilizemos $(X_1, X_1^2, X_2, X_2^2, X_1X_2)$, ao invés de (X_1, X_2);
- A fronteira de decisão ficará

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

- Que continua linear para essas novas variáveis;
- Mas, conduz à decisão não linear no espaço original.
- Se fosse um polinômio cubico sairíamos de 2 para 9 variáveis!



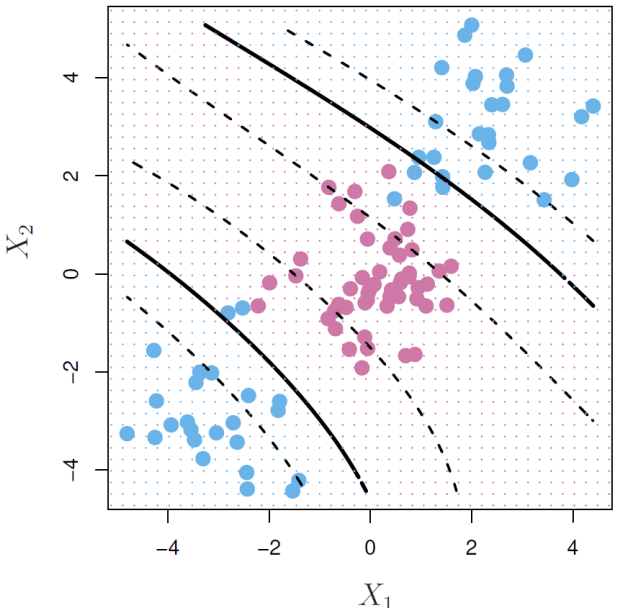
Ideia da expansão das características

Exemplo 3

- Suponha que utilizemos $(X_1, X_1^2, X_2, X_2^2, X_1X_2)$, ao invés de (X_1, X_2);
- A fronteira de decisão ficará

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

- Que continua linear para essas novas variáveis;
- Mas, conduz à decisão não linear no espaço original.
- Se fosse um polinômio cubico sairíamos de 2 para 9 variáveis!



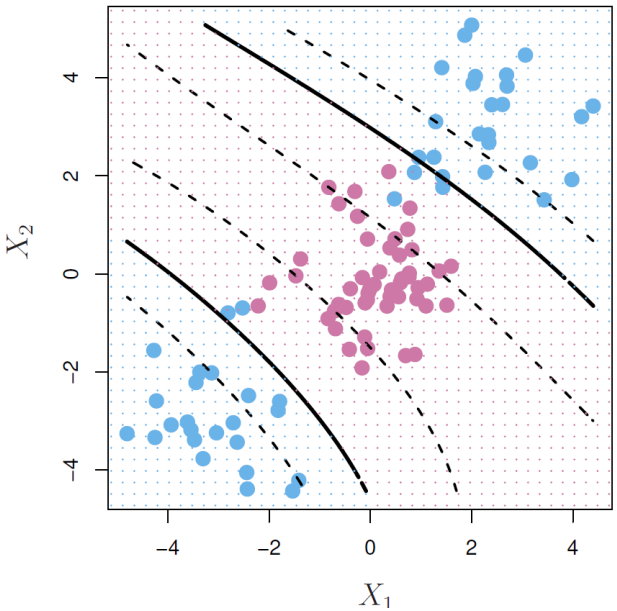
Ideia da expansão das características

Exemplo 3

- Suponha que utilizemos $(X_1, X_1^2, X_2, X_2^2, X_1X_2)$, ao invés de (X_1, X_2);
- A fronteira de decisão ficará

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

- Que continua linear para essas novas variáveis;
- Mas, conduz à decisão não linear no espaço original.
- Se fosse um polinômio cubico sairíamos de 2 para 9 variáveis!



Exemplo: `Heart` data set

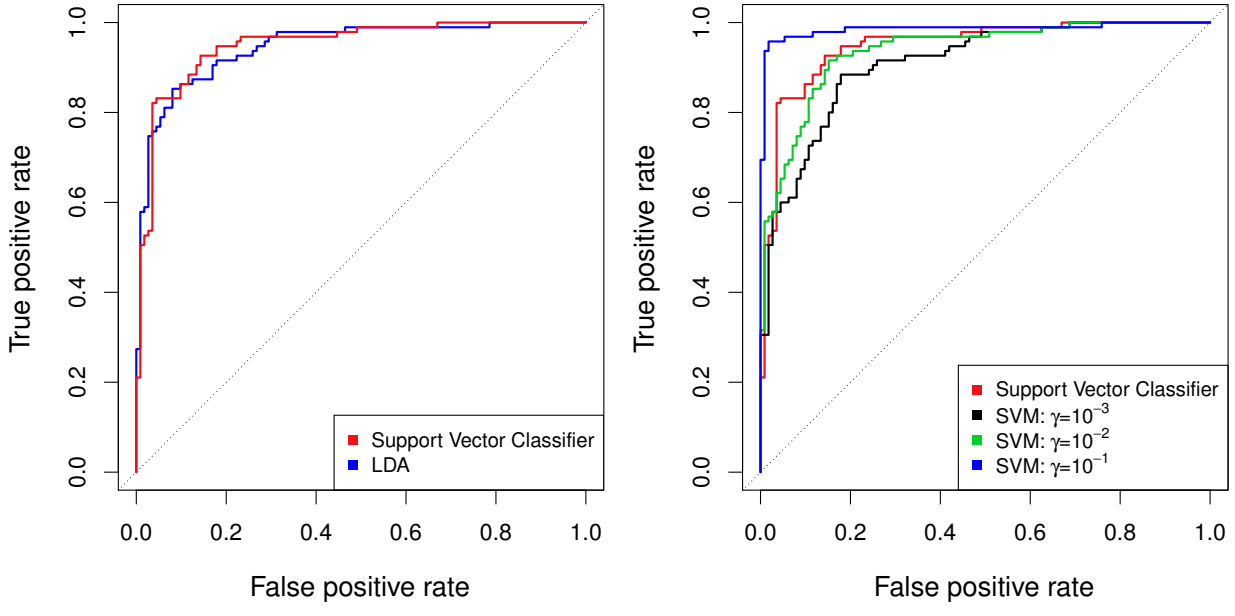
- O objetivo é utilizar 13 preditores (p. ex., `age`, `sex` e `chol`) a fim de prever se o indivíduo tem ou não doença cardíaca;
- Vamos comparar, através da curva ROC, o desempenho do Support Vector Classifier frente à Análise de Discriminante Linear (ADL);
- E Support Vector Classifier *vs* Support Vector Machine com kernel gaussiano (variando γ - utilizando os dados de treino);
- No gráfico da direita, note que não temos uma comparação muito justa, pois quanto maior γ mais complexo é o modelo (e melhor o ajuste);

Exemplo: `Heart` data set

- O objetivo é utilizar 13 preditores (p. ex., `age`, `sex` e `chol`) a fim de prever se o indivíduo tem ou não doença cardíaca;
- Vamos comparar, através da curva ROC, o desempenho do Support Vector Classifier frente à Análise de Discriminante Linear (ADL);
- E Support Vector Classifier *vs* Support Vector Machine com kernel gaussiano (variando γ - utilizando os dados de treino);
- No gráfico da direita, note que não temos uma comparação muito justa, pois quanto maior γ mais complexo é o modelo (e melhor o ajuste);

Exemplo: `Heart` data set

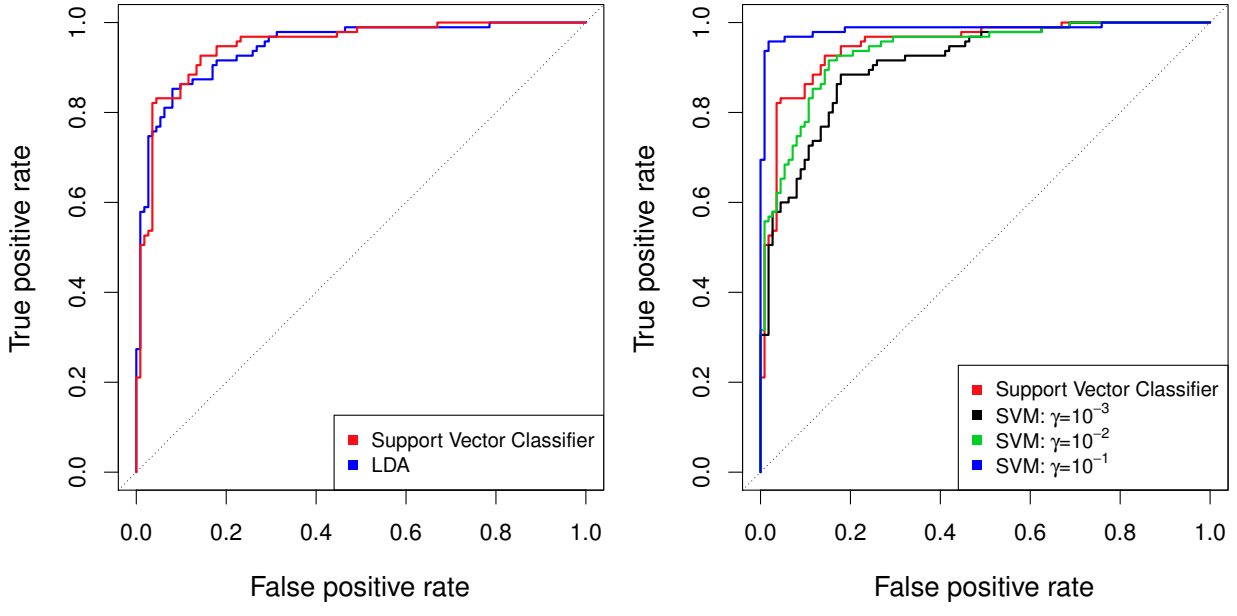
- O objetivo é utilizar 13 preditores (p. ex., `age`, `sex` e `chol`) a fim de prever se o indivíduo tem ou não doença cardíaca;
- Vamos comparar, através da curva ROC, o desempenho do Support Vector Classifier frente à Análise de Discriminante Linear (ADL);
- E Support Vector Classifier *vs* Support Vector Machine com kernel gaussiano (variando γ - utilizando os dados de treino);



- No gráfico da direita, note que não temos uma comparação muito justa, pois quanto maior γ mais complexo é o modelo (e melhor o ajuste);

Exemplo: `Heart` data set

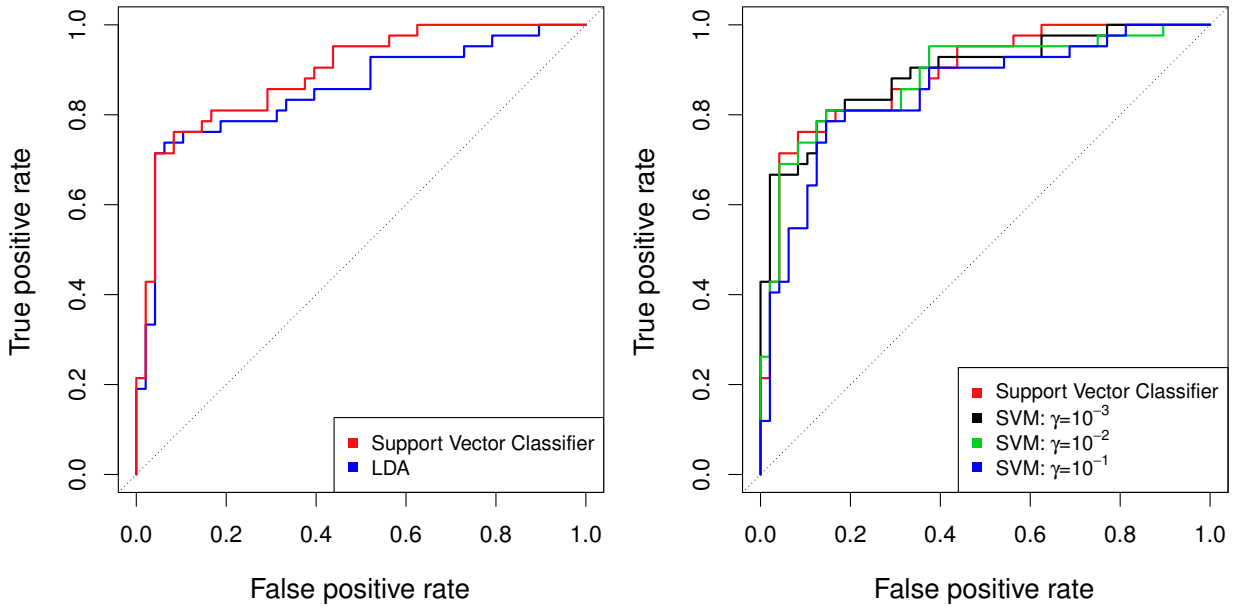
- O objetivo é utilizar 13 preditores (p. ex., `age`, `sex` e `chol`) a fim de prever se o indivíduo tem ou não doença cardíaca;
- Vamos comparar, através da curva ROC, o desempenho do Support Vector Classifier frente à Análise de Discriminante Linear (ADL);
- E Support Vector Classifier *vs* Support Vector Machine com kernel gaussiano (variando γ - utilizando os dados de treino);



- No gráfico da direita, note que não temos uma comparação muito justa, pois quanto maior γ mais complexo é o modelo (e melhor o ajuste);

Exemplo: `Heart` data set

- Agora, com os dados de teste, o comportamento da curva ROC é um pouco diferente;

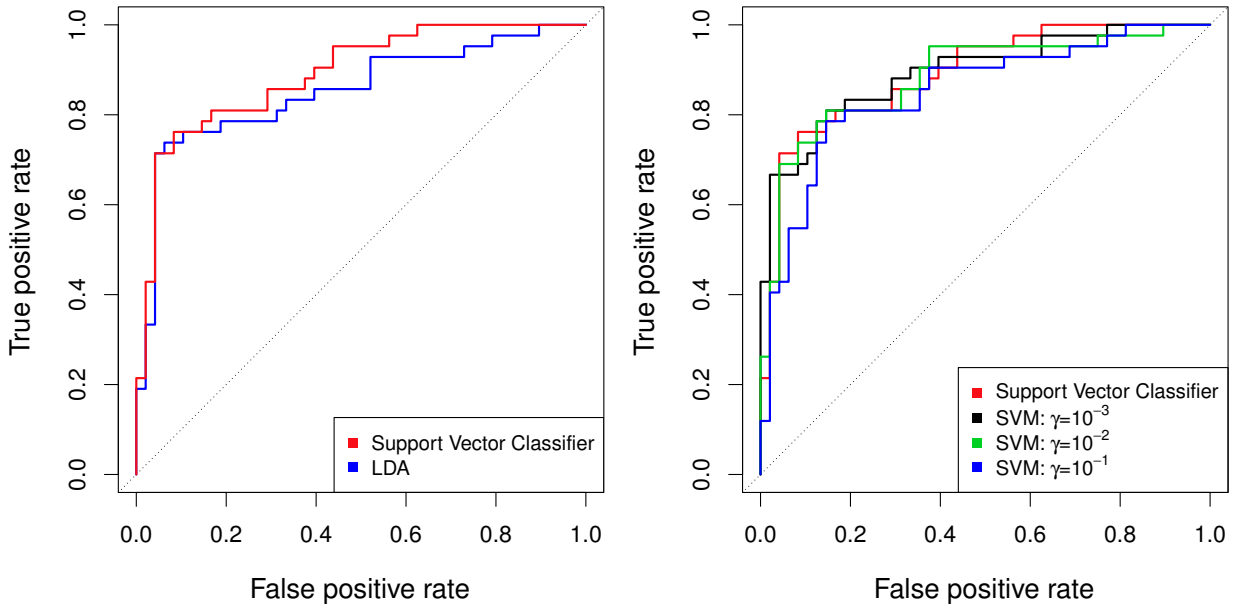


- Note que, embora nos dados de treino (gráfico anterior) LDA e SVC comportavam-se de modo semelhante, no teste SVC se mostrou superior;

- E SVM com $\gamma = 10^{-1}$ apresentou um pior desempenho;

Exemplo: `Heart` data set

- Agora, com os dados de teste, o comportamento da curva ROC é um pouco diferente;

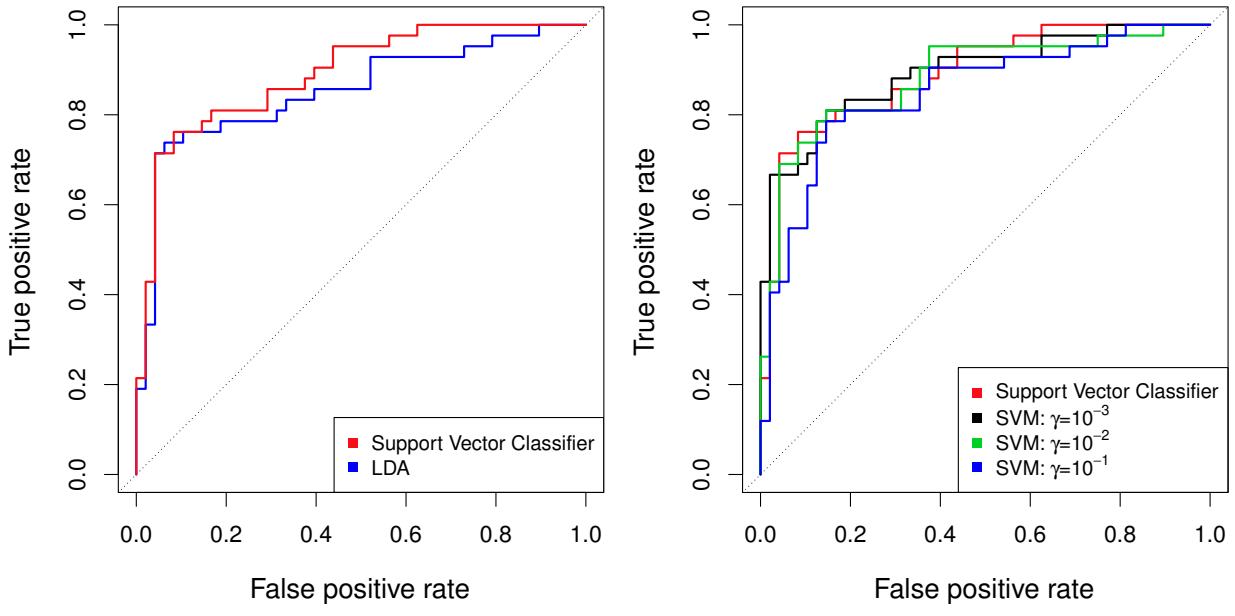


- Note que, embora nos dados de treino (gráfico anterior) LDA e SVC comportavam-se de modo semelhante, no teste SVC se mostrou superior;

- E SVM com $\gamma = 10^{-1}$ apresentou um pior desempenho;

Exemplo: `Heart` data set

- Agora, com os dados de teste, o comportamento da curva ROC é um pouco diferente;



- Note que, embora nos dados de treino (gráfico anterior) LDA e SVC comportavam-se de modo semelhante, no teste SVC se mostrou superior;

- E SVM com $\gamma = 10^{-1}$ apresentou um pior desempenho;

Se temos mais de duas classes?

- O SVM como definido funciona para $K = 2$ classes. O que fazemos então se temos $K > 2$ classes?

⋆ **One versus All (OVA):**

  - Ajusta-se K diferentes classificadores de duas classes, $\hat{f}_k(x)$, $k = 1, \ldots, K$;
  - Cada classe versus o restante;
  - Classifica x^* para classe na qual $\hat{f}_k(x^*)$ for maior.

⋆ **One versus One (OVO):**

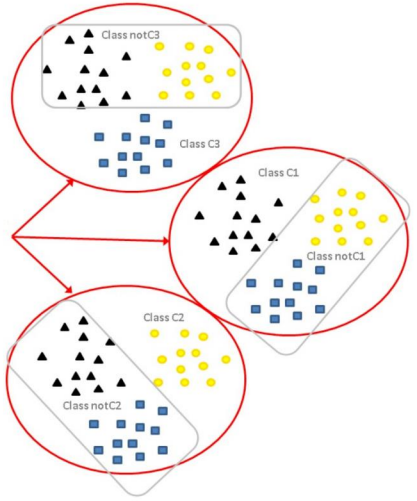
  - Ajusta-se todos os $\binom{K}{2}$ classificadores, $\hat{f}_{kl}(x)$;
  - Classifica x^* para a classe que vencer a maioria das competições duas a duas.

Se temos mais de duas classes?

- O SVM como definido funciona para $K = 2$ classes. O que fazemos então se temos $K > 2$ classes?

⋆ **One versus All (OVA):**

  - Ajusta-se K diferentes classificadores de duas classes, $\hat{f}_k(x)$, $k = 1, \ldots, K$;
  - Cada classe versus o restante;
  - Classifica x^* para classe na qual $\hat{f}_k(x^*)$ for maior.



⋆ **One versus One (OVO):**

  - Ajusta-se todos os $\binom{K}{2}$ classificadores, $\hat{f}_{kl}(x)$;
  - Classifica x^* para a classe que vencer a maioria das competições duas a duas.