Universidade Federal do Paraná

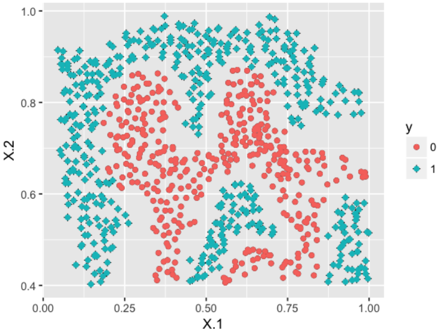
Laboratório de Estatística e Geoinformação - LEG

Support Vector Machines

Prof.: Eduardo Vargas Ferreira

Definição

- **Support Vector Machines** são baseados no conceito de planos de decisão (que definem limites de decisão);



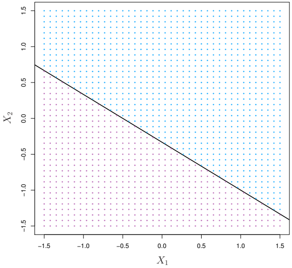
- Tentamos encontrar o plano que separa as classes no espaço de características, (X_1, X_2);

O que é um hiperplano?

- Em geral, a equação para o hiperplano tem a forma

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

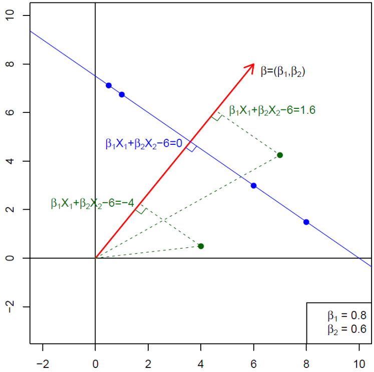
- Por exemplo, quando $p = 2$ o hiperplano é uma linha;



- Se $\beta_0 = 0$ dizemos que o hiperplano passa na origem, caso contrário não.

O que é um hiperplano?

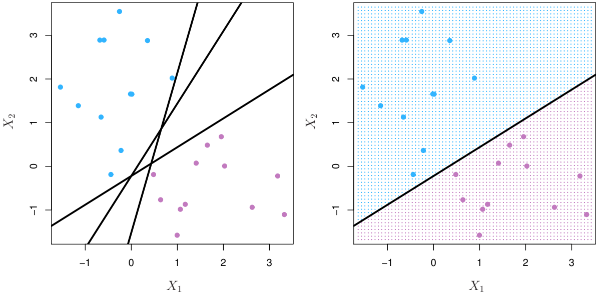
- O vetor $\beta = (\beta_0, \beta_1, \ldots, \beta_p)$ é chamado vetor normal. Ele aponta a direção ortogonal à superfície do hiperplano;



- Se $f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p > 0$, estamos em um lado do hiperplano, se $f(X) < 0$, estamos do outro lado.

Separando as classes

- Note que se codificarmos como $Y_i = +1$ os pontos em azul, e $Y_i = -1$ os pontos em rosa, então $Y_i \cdot f(X_i) > 0$ para todo i;



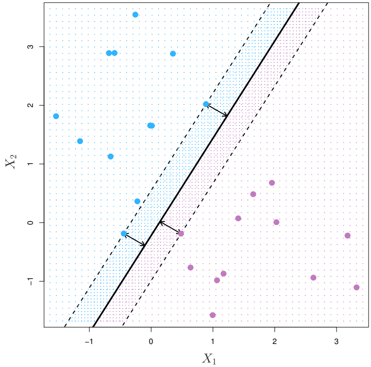
- Mas, em meio a tantos hiperplanos possíveis, qual escolher?

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).

Problema de otimização com restrição



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

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

Voltamos ao problema de otimização restrita

- Então, temos o seguinte problema:

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

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

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

- A resolução das equações $\frac{\partial J}{\partial \boldsymbol{\beta}} = 0$ e $\frac{\partial J}{\partial \beta_0} = 0$ leva ao seguinte problema

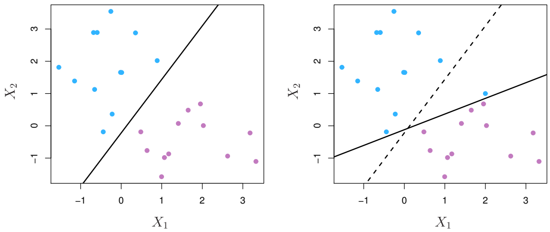
  ▸ maiores detalhes

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

Perturbação nos dados

- A busca por um classificador que separe perfeitamente **todas** as observações de treinamento torna-o sensível a outliers;

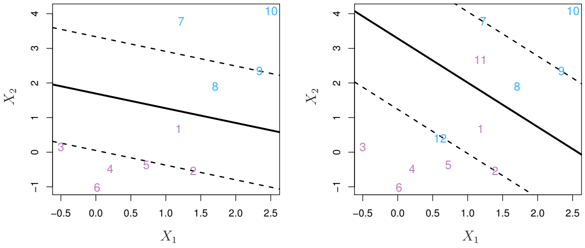
- Mais ainda, o fato de uma observação gerar uma dramática mudança no hiperplano sugere a possível ocorrência de overfit;



- Além disso, em situações reais, é difícil encontrar aplicações cujos dados sejam linearmente separáveis (o **hiperplano, geralmente, não existe**);

Support Vector Classifier

- Diante desses problemas, surgiu a ideia de se considerar um classificador que **não separe 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;



Support Vector Classifier

- O problema de otimização fica então

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

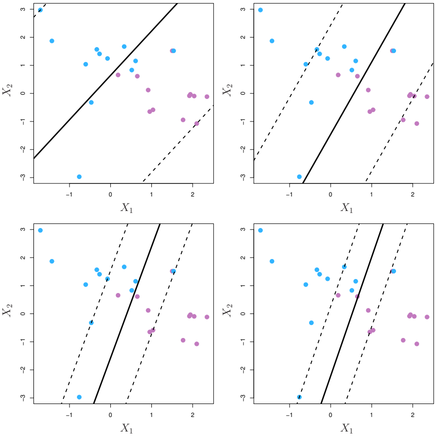
$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq \underbrace{M(1 - \epsilon_i)}_{\text{violação da margem}}$$

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

- 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* considerando diferentes valores de C.

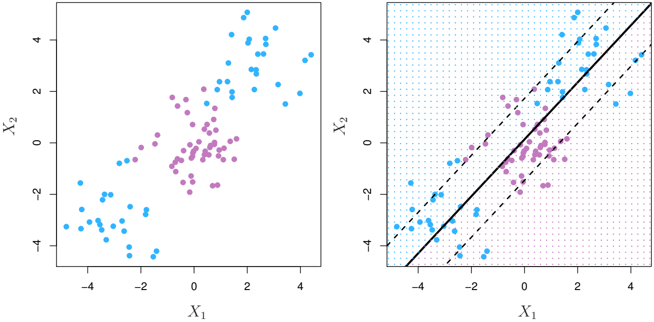


- Quando C é grande, temos alta tolerância para as observações estarem no lado errado da margem (a margem será mais larga).

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**.

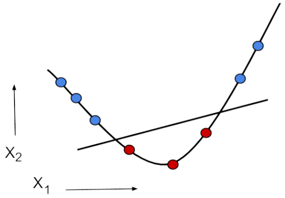


- Neste caso, o desempenho do método não é satisfatório. Então, o que fazer? **Expansão das características**

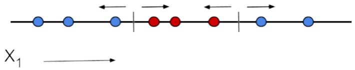
Ideia da expansão das características

- Considere o seguinte problema linearmente não separável

- Expandindo a característica de x_1, através de $x_2 = x_1^2$, conseguimos uma separação linear

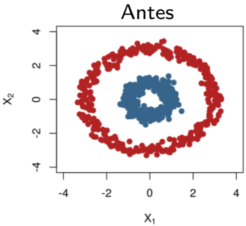


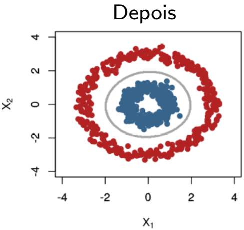
- Que projetada no espaço original, se transforma em



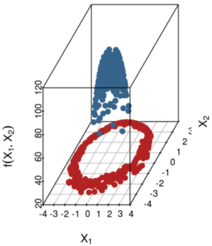
Ideia da expansão das características

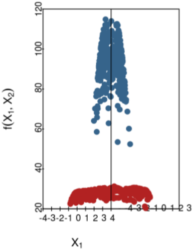
- Suponha outro problema de classificação:





- Expandindo o espaço, temos um hiperplano linearmente separável em $\mathbb{R}^3$.





Truque do Kernel

- Voltando ao problema de otimização

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

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

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

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

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

- Ao escolher $\Phi(\mathbf{x}_i) = \left(1, \sqrt{2}x_{i1}, \sqrt{2}x_{i2}, x_{i1}^2, x_{i2}^2, \sqrt{2}x_{i1}x_{i2}\right)$, chegamos em: $K(\mathbf{x}_i, \mathbf{x}_k) = \langle \Phi(\mathbf{x_i}), \Phi(\mathbf{x_k}) \rangle$.

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:** $K(x_i, x_k) = tanh\left(k\langle x_i, x_k \rangle - \delta\right)$.

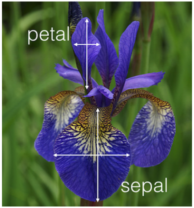
- Os parâmetros que devem ser determinados pelo usuário.

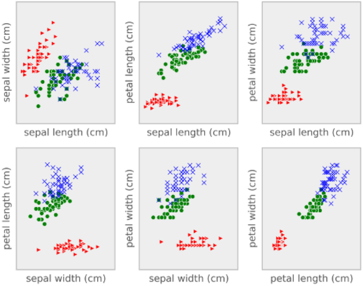
Exemplo 1: `Iris` dataset

- O objetivo deste estudo é classificar a flor em três categorias:

  - Versicolor;
  - Virginica;
  - Setosa.

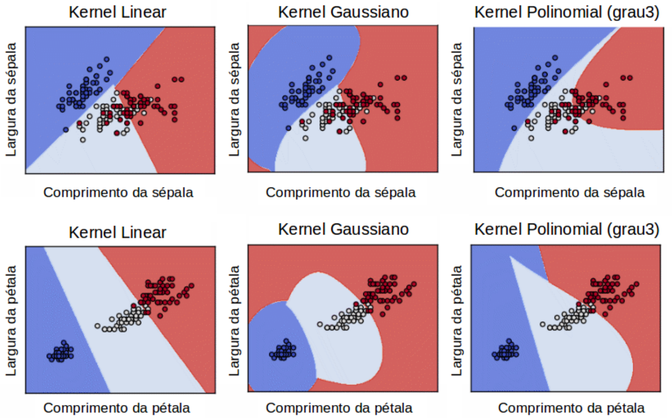
- Para tanto, utilizamos o comprimento e largura das pétalas e sépalas.





Exemplo 1: `Iris` dataset

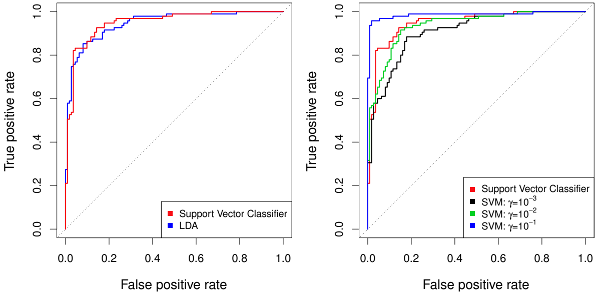
- Note como a mudança do kernel e das variáveis alteram a classificação.



Exemplo 2: Heart dataset

- O objetivo é utilizar 13 preditores (p. ex., age, sex e chol) a fim de prever se o indivíduo tem ou não doença cardíaca;

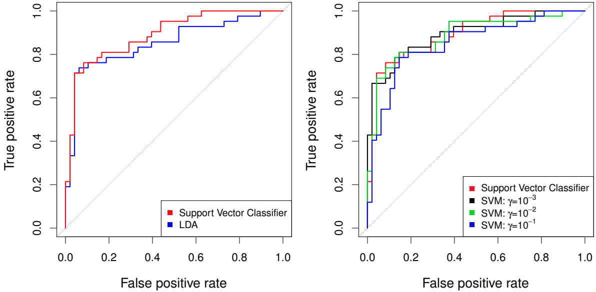
- Vamos comparar o desempenho dos métodos através da curva ROC, **utilizando os dados de treino**



- 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 2: 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 mostrou-se superior;

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

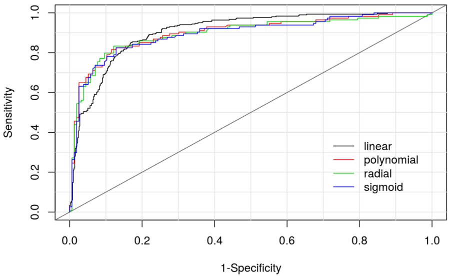
Exemplo 3: OJ (Oranje Juice) data set

- Os dados referem-se a 1070 observações sobre compra de sucos de laranja em um mercado;
- Os registros contemplam respostas sobre o suco comprado CC para *Citrus Hill* e MM para *Minute Maid* e demais 17 características:

```
## $ WeekofPurchase: num  237 239 245 227 228 230 232  ...
## $ StoreID       : num  1 1 1 1 7 7 7 7 7 7 ...
## $ PriceCH       : num  1.75 1.75 1.86 1.69 1.69  1.75 1.75 ...
## $ PriceMM       : num  1.99 1.99 2.09 1.69 1.69 1.99 1.99 ...
## $ DiscCH        : num  0 0 0.17 0 0 0 0 0 0 0 ...
## $ DiscMM        : num  0 0.3 0 0 0 0 0.4 0.4 0.4 0.4 ...
## $ SpecialCH     : num  0 0 0 0 0 0 1 1 0 0 ...
## $ SpecialMM     : num  0 1 0 0 0 1 1 0 0 0 ...
## $ LoyalCH       : num  0.5 0.6 0.68 0.4 0.957 ...
## $ SalePriceMM   : num  1.99 1.69 2.09 1.69 1.69 1.59 1.59 ...
## $ SalePriceCH   : num  1.75 1.75 1.69 1.69 1.69 1.75 1.75 ...
## $ PriceDiff     : num  0.24 -0.06 0.4 0 -0.16 -0.16 -0.16 ...
## $ Store7        : Factor w/ 2 levels "No","Yes": 1 1 1 2 2 2 ...
## $ PctDiscMM     : num  0 0.151 0 0 0 ...
## $ PctDiscCH     : num  0 0 0.0914 0 0 ...
## $ ListPriceDiff : num  0.24 0.24 0.23 0 0 0.3 0.3 0.24 0.24 0.24 ...
## $ STORE         : num  1 1 1 1 0 0 0 0 0 0 ...
```

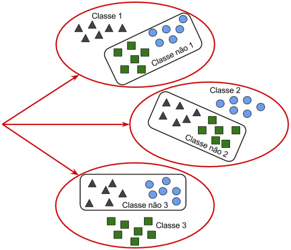
Exemplo 3: `OJ` (Oranje Juice) data set

	Linear	Polynomial	Radial	Sigmoid
Sensitivity	0.8515	0.8158	0.8333	0.8246
Specificity	0.831	0.8846	0.8782	0.8718
AUC	0.9064	0.9004	0.8954	0.8946



Se temos mais de duas classes?

- O que fazemos então se temos $K > 2$ classes?

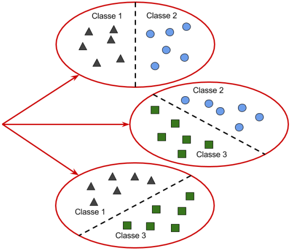


⋆ **One versus All (OVA):**

  – Compara-se cada classe *vs* as restantes;

  – Calcula-se $f_k(x^*), k = 1 : K$;

  – $x^* \in k \mid f_k(x^*) > f_{(-k)}(x^*)$.



⋆ **One versus One (OVO):**

  – Treina-se os $\binom{K}{2}$ classificadores;

  – Classifica x^* para a classe que vencer a maioria das competições.