

Machine/Statistical Learning  
Model free approaches  
SVM: Support Vector Machines

SICOM, M2 Sigma

2021-22

## Support Vector Machine (SVM)

Theory elaborated in the early 1990's (Vapnik *et al*) based on the idea of 'maximum margin'

- ▶ deterministic criterion learned on the training set ← supervised classification
- 👉 general, i.e. model free, linear classification rule
- 👉 classification rule is linear in a transformed space of higher (possible infinite) dimension than the original input feature/predictor space

### Supplementary materials

- 📺 Coursera online video with python notebook material (13mn)  
<https://www.coursera.org/lecture/data-analytics-accountancy-2/introduction-to-support-vector-machine-dDP0v>
- 🌐 Wikipedia page (quite complete and detailed)  
[https://en.wikipedia.org/wiki/Support\\_vector\\_machine](https://en.wikipedia.org/wiki/Support_vector_machine)
- 📄 Short and easy to understand Scikit-learn documentation (with examples)  
<https://scikit-learn.org/stable/modules/svm.html>

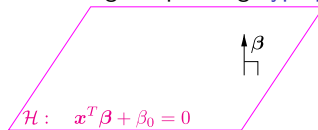
## Linear discrimination and Separating hyperplane

### Binary classification problem

- ▶  $X \in \mathbb{R}^p$
- ▶  $Y \in \{-1, 1\} \leftarrow 2 \text{ classes}$
- ▶ Training set  $(x_i, y_i)$ , for  $i = 1, \dots, n$

Defining a **linear** discriminant function  $h(x) \Leftrightarrow$  defining a separating **hyperplane**  $\mathcal{H}$  with equation

$$\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0,$$

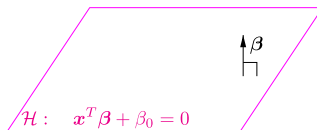


- ▶  $\boldsymbol{\beta} \in \mathbb{R}^p$  is the normal vector (vector normal to the hyperplane  $\mathcal{H}$ ),
- ▶  $\beta_0 \in \mathbb{R}$  is the intercept (regression interpretation) or offset (geometrical interpretation)
- 🔗  $\mathcal{H}$  is an *affine subspace* of dimension  $p - 1$
- 🔗  $h(x) \equiv \mathbf{x}^T \boldsymbol{\beta} + \beta_0$  is the associated (linear) discriminant function

## Separating hyperplane and prediction rule

For a given separating hyperplane  $\mathcal{H}$  with equation

$$\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0,$$



the **prediction rule** can be expressed as

- ▶  $\hat{y} = +1$ , if  $h(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\beta} + \beta_0 \geq 0$ , ( $\mathbf{x}$  is above  $\mathcal{H}$ )
- ▶  $\hat{y} = -1$ , otherwise, ( $\mathbf{x}$  is below  $\mathcal{H}$ )

or in an equivalent way:

$$\hat{y} \equiv G(\mathbf{x}) = \text{sign} \left[ \mathbf{x}^T \boldsymbol{\beta} + \beta_0 \right]$$

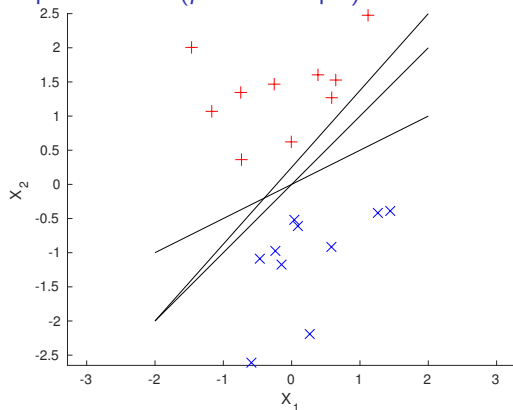
**Rk:**  $\mathbf{x}$  is in class  $y \in \{-1, 1\}$ : prediction  $G(\mathbf{x})$  is **correct** iff  $y (\mathbf{x}^T \boldsymbol{\beta} + \beta_0) \geq 0$

## Separating Hyperplane: separable case

**Linear separability assumption:**  $\exists \beta \in \mathbb{R}^p$  and  $\beta_0 \in \mathbb{R}$  s.t. the hyperplane  $\mathbf{x}^T \beta + \beta_0 = 0$  perfectly separates the two classes on the training set:

$$y_k \left( \mathbf{x}_k^T \beta + \beta_0 \right) \geq 0, \quad \text{for } k = 1, \dots, n,$$

Separable case ( $p = 2$  example)



**Pb:** infinitely **many** possible perfect  
**separating hyperplanes**  
 $\mathbf{x}^T \beta + \beta_0 = 0$

🔍 Find the 'optimal' separating  
hyperplane?

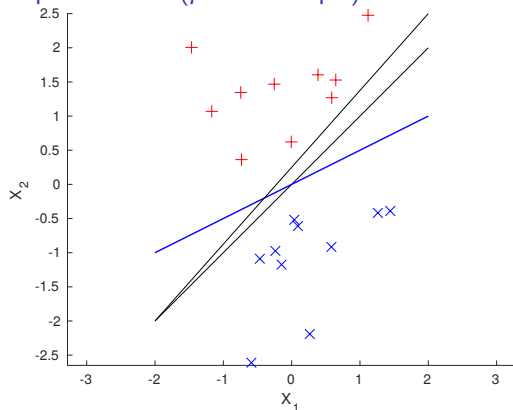
🔍 makes the 'biggest gap' from the  
samples

## Separating Hyperplane: separable case

**Linear separability assumption:**  $\exists \beta \in \mathbb{R}^p$  and  $\beta_0 \in \mathbb{R}$  s.t. the hyperplane  $\mathbf{x}^T \beta + \beta_0 = 0$  perfectly separates the two classes on the training set:

$$y_k (\mathbf{x}_k^T \beta + \beta_0) \geq 0, \quad \text{for } k = 1, \dots, n,$$

Separable case ( $p = 2$  example)



**Pb:** infinitely **many** possible perfect  
**separating hyperplanes**  
 $\mathbf{x}^T \beta + \beta_0 = 0$

- 🔍 Find the 'optimal' separating hyperplane?
- 🔍 makes the 'biggest gap' from the samples

## Maximum margin separating hyperplane (separable case)

Distance of a point  $\mathbf{x}_k$  to an hyperplane  $\mathcal{H}$  s.t.  $\mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0$ ,

$$d(x_k, \mathcal{H}) \equiv \min_{\mathbf{x}} \left\{ \|\mathbf{x} - \mathbf{x}_k\| : \mathbf{x}^T \boldsymbol{\beta} + \beta_0 = 0 \right\}$$

### Maximum margin principle

We are interested in the 'optimal' perfect separating hyperplane maximizing the distance  $M > 0$ , called the **margin**, between the samples of each class and the separating hyperplane

⇒ Find  $\boldsymbol{\beta} \in \mathbb{R}^p$  and  $\beta_0 \in \mathbb{R}$  s.t. the margin

$$M = \min_{1 \leq k \leq n} \{d(x_k, \mathcal{H})\}$$

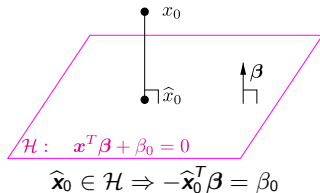
is **maximized**

## Signed distance

From the orthogonality principle,

$$d(x_0, \mathcal{H}) = \|x_0 - \hat{x}_0\|,$$

where  $\hat{x}_0$  is the orthogonal projection of  $x_0$  on  $\mathcal{H}$



$\Rightarrow x_0 - \hat{x}_0$  and  $\beta$  are collinear,

$\Rightarrow x_0 - \hat{x}_0 = \underbrace{\langle x_0 - \hat{x}_0, \beta^* \rangle}_{\text{unsigned distance}} \beta^*$ , where  $\beta^* = \frac{\beta}{\|\beta\|}$ ,

$\Rightarrow \text{signed distance} = (x_0 - \hat{x}_0)^T \frac{\beta}{\|\beta\|} = \frac{x_0^T \beta - \hat{x}_0^T \beta}{\|\beta\|} = \frac{x_0^T \beta + \beta_0}{\|\beta\|}$ ,

### Remarks

- ▶  $|\langle x_0 - \hat{x}_0, \beta^* \rangle| = \|x_0 - \hat{x}_0\| = d(x_0, \mathcal{H}) \leftarrow \text{"signed distance"}$
- ▶ for any perfect separating hyperplane  $y_k \langle x_k - \hat{x}_k, \beta^* \rangle = \frac{1}{\|\beta\|} y_k (x_k^T \beta + \beta_0) \geq 0$ , for  $k = 1, \dots, n$ ,



## Canonical separating hyperplane

For any perfect separating hyperplane, for  $k = 1, \dots, n$

$$y_k \langle \mathbf{x}_k - \widehat{\mathbf{x}}_k, \boldsymbol{\beta}^* \rangle = d(\mathbf{x}_k, \mathcal{H})$$

Hence, the margin reads

$$M \equiv \min_{1 \leq k \leq n} \{d(\mathbf{x}_k, \mathcal{H})\} = \frac{1}{\|\boldsymbol{\beta}\|} \min_{1 \leq k \leq n} \left\{ y_k (\mathbf{x}_k^T \boldsymbol{\beta} + \beta_0) \right\}$$

### Remarks

- The bound  $M$  is reached (min of a countable set),
- 👉 the samples at the margin are denoted as  $\mathbf{x}_{\text{margin}}$

### Canonical expression of the separating hyperplane

$\boldsymbol{\beta}$  and  $\beta_0$  are normalized s.t.

$$y_{\text{margin}} (\mathbf{x}_{\text{margin}}^T \boldsymbol{\beta} + \beta_0) = 1, \quad \text{thus } M = \frac{1}{\|\boldsymbol{\beta}\|}$$

## Primal problem (separable case)

Canonical hyperplane expression:

$$\begin{aligned} \text{maximizing the margin } M = \frac{1}{\|\beta\|} &\Leftrightarrow \text{minimizing } \|\beta\| \\ &\Leftrightarrow \text{minimizing } \frac{1}{2} \|\beta\|^2 \end{aligned}$$

### Primal optimization problem

$$\begin{cases} \min_{\beta, \beta_0} & \frac{1}{2} \|\beta\|^2, \\ \text{subject to} & y_k (\mathbf{x}_k^T \beta + \beta_0) \geq 1, \text{ for } 1 \leq k \leq n. \end{cases}$$

- quadratic criterion + linear inequality constraints
- 👉 convex optimization problem for which standard numerical procedures are available

## Reminder on constrained optimization

- ▶ Concept of feasible descent direction
- ▶ Primal problem (constrained form) / Dual problem (Lagrangian form)
- ▶ KKT necessary conditions

## Reminder on constrained optimization

Constrained problem: primal problem

$$\begin{cases} \min_{\mathbf{x}} & f(\mathbf{x}) \\ \text{s.t.} & g(\mathbf{x}) \leq 0 \end{cases}$$

Objective function  $f(\mathbf{x})$

To decrease the objective function  $f(\mathbf{x})$ , a descent direction  $\mathbf{d}$  must satisfy

$$f(\mathbf{x} + \epsilon \mathbf{d}) \approx J(\mathbf{x}) + \epsilon \nabla f(\mathbf{x})^T \mathbf{d} < f(\mathbf{x}),$$

hence  $\mathbf{d}$  is a **descent direction** iff  $\nabla f(\mathbf{x})^T \mathbf{d} < 0$

## Reminder on constrained optimization (Cont'd)

Objective  $f(\mathbf{x})$

descent direction:  $\nabla f(\mathbf{x})^T \mathbf{d} < 0$

Constraint  $g(\mathbf{x})$

To satisfy the constraint, a **feasible** descent direction  $\mathbf{d}$  must satisfy

$$g(\mathbf{x} + \epsilon \mathbf{d}) \approx g(\mathbf{x}) + \epsilon \nabla g(\mathbf{x})^T \mathbf{d} \leq 0,$$

hence

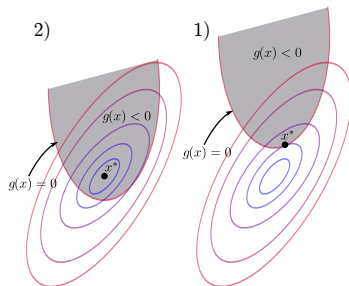
$$\text{feasible direction: } \begin{cases} g(\mathbf{x}) < 0 & \Rightarrow \text{no constraint on } \mathbf{d}, \\ g(\mathbf{x}) = 0 & \Rightarrow \nabla g(\mathbf{x})^T \mathbf{d} \leq 0 \end{cases}$$

## Reminder on constrained optimization (Cont'd)

Necessary conditions: two possibilities for optimality

There is no feasible descent direction in  $\mathbf{x}^*$  when either

1.  $\nabla f(\mathbf{x}^*) = -\alpha \nabla g(\mathbf{x}^*)$ , with  $\alpha > 0$ ,  $g(\mathbf{x}^*) = 0$
2.  $\nabla f(\mathbf{x}^*) = 0$ , i.e.  $\alpha = 0$ ,  $g(\mathbf{x}^*) < 0$



### Remarks

1.  $\mathbf{x}^*$  lies at the limit of the feasible domain (i.e.  $g(\mathbf{x}^*) = 0$ ) and the two gradients are **collinear** and in **opposite** direction
2.  $\mathbf{x}^*$  belongs to the interior of the feasible domain (i.e.  $g(\mathbf{x}^*) < 0$ ). Same 1st order necessary conditions as those obtained when there is no constraint

## Reminder on constrained optimization (Cont'd)

Constrained form: primal problem

$$\begin{cases} \min_{\mathbf{x}} & f(\mathbf{x}) \\ \text{s.t.} & g_j(\mathbf{x}) \leq 0, \text{ for all } j = 1, \dots, q \end{cases}$$

Lagrangian form: dual problem

Inequality convex constraints  $\Rightarrow$  introduction of the Lagrange multipliers  $\alpha_j$

$$\begin{cases} \min_{\mathbf{x}} & f(\mathbf{x}) + \sum_j \alpha_j g_j(\mathbf{x}^*) \\ \text{s.t.} & \alpha_j \geq 0, \text{ for all } j = 1, \dots, q \end{cases}$$

Karush–Kuhn–Tucker (KKT) conditions

For  $\mathbf{x}^*$  being a local min, it is necessary that

$$\begin{cases} \nabla f(\mathbf{x}^*) + \sum_{j=1}^q \alpha_j \nabla g_j(\mathbf{x}^*) = 0 & \leftarrow \text{first order conditions} \\ \text{s.t. } \alpha_j \geq 0 \text{ and } \alpha_j g_j(\mathbf{x}^*) = 0 & \leftarrow \text{complementary conditions} \end{cases}$$

End of reminder on constrained optimization

📌 Dual form for SVM optimization problem



## Lagrangian (separable case)

Linear constraints of positivity  $\Rightarrow$  introduction of the Lagrange multipliers

### Lagrangian

$$L(\boldsymbol{\beta}, \beta_0, \boldsymbol{\alpha}) = \frac{1}{2} \|\boldsymbol{\beta}\|^2 - \sum_{i=1}^n \alpha_i \underbrace{\left[ y_i (\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) - 1 \right]}_{\geq 0},$$

where  $\alpha_i$  are the Lagrange multipliers

### First order Karush–Kuhn–Tucker necessary conditions

Setting the partial derivatives w.r.t.  $\boldsymbol{\beta}$  and  $\beta_0$  to zero yields

$$\begin{cases} \hat{\boldsymbol{\beta}} &= \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i, \\ 0 &= \sum_{i=1}^n \alpha_i y_i, \end{cases}$$

► plugging these expression in the Lagrangian yields the dual expression

## Dual problem (separable case)

### Dual optimization problem

$$\begin{cases} \max_{\alpha} & \tilde{L}(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j, \\ \text{subject to} & \alpha_i \geq 0 \text{ and } \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

- 🔗 simple convex optimization problem for which standard numerical procedures are available
- 🔗 calculation of the optimum multipliers  $\hat{\alpha}_i$

## Support vectors and maximum margin hyperplane (separable case)

### Complementary slackness Karush–Kuhn–Tucker necessary conditions

$$\hat{\alpha}_i[y_i h(\mathbf{x}_i) - 1] = 0 \quad \Rightarrow \quad \hat{\alpha}_i = 0 \quad \text{as} \quad y_i h(\mathbf{x}_i) > 1$$

- ▶ since  $\hat{\beta} = \sum_{i=1}^n \hat{\alpha}_i y_i \mathbf{x}_i$ ,  $\hat{\beta}$  depends only on the points at the margin ← **support vectors**
- ▶  $\hat{\beta}_0$  can be derived from the complementary slackness expression for any of support vectors  $\mathbf{x}_{\text{margin}}$

$$\begin{aligned} y_{\text{margin}} h(\mathbf{x}_{\text{margin}}) - 1 &= 0 \quad \Rightarrow \quad \hat{\beta}^T \mathbf{x}_{\text{margin}} + \hat{\beta}_0 = y_{\text{margin}}, \\ &\Rightarrow \quad \hat{\beta}_0 = -\hat{\beta}^T \mathbf{x}_{\text{margin}} + y_{\text{margin}} \end{aligned}$$

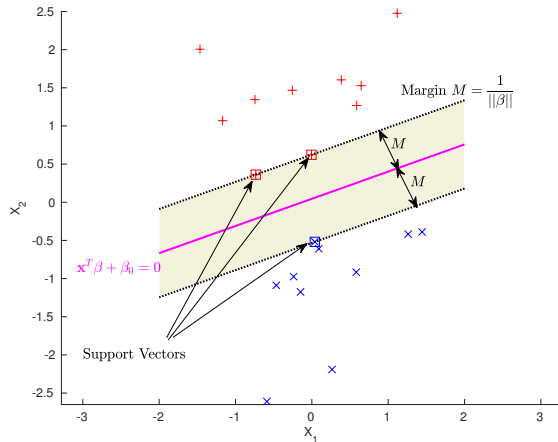
- 👉 the only **inputs used to construct the maximum margin hyperplane** are the **support vectors** and the discriminant function reads

$$h(\mathbf{x}) = \sum_{i=1}^n \hat{\alpha}_i y_i (\mathbf{x} - \mathbf{x}_{\text{margin}})^T \mathbf{x}_i + y_{\text{margin}}$$

## Maximum margin separating hyperplane (separable case)

### Separable case

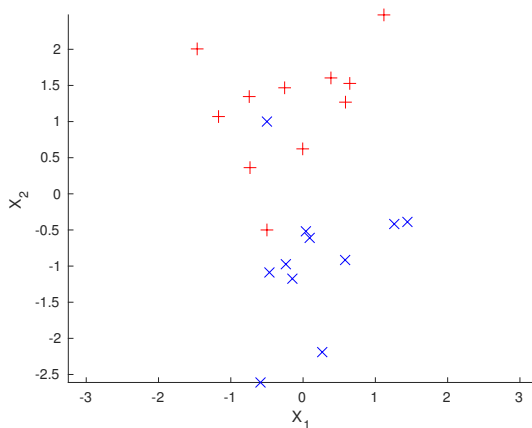
👉 Maximizing the *margin*  $M$  between the separating hyperplane and the training data:



The maximum margin hyperplane depends only on the points at the margin called the *support vectors*

## Nonseparable case

- ▶ in general, overlap of the 2 classes
- 🚫 No hyperplane that perfectly separates the training data



## Maximum margin separating hyperplane (nonseparable case)

### Soft-Margin solution for the nonseparable case

Considering a *soft-margin* that allows wrong classifications

- ▶ introduction of *slack variables*  $\xi_i \geq 0$  s.t.

$$y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq (1 - \xi_i)$$

Support vectors include now the wrong classified points, and the points inside the margins ( $\xi_i > 0$ )

- ▶ Primal problem: adding a penalty in the criterion

$$\begin{cases} \min_{\boldsymbol{\beta}, \beta_0, \xi} & \frac{1}{2} \|\boldsymbol{\beta}\|^2 + C \sum_{i=1}^n \xi_i, \\ \text{subject to} & y_i(\mathbf{x}_i^T \boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i, \end{cases}$$

where  $C > 0$  is the “cost” or “regularization” parameter

## Regularization parameter (nonseparable case)

$$\text{Criterion to be minimized: } \frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^n \xi_i,$$

### Influence of the regularization parameter $C > 0$

$C$  drives the margin size, thus the number of support vectors

- ▶  $C \gg 0$  : small margin, less support vectors ( $\sim$  overfitting)
- ▶  $C \rightarrow 0^+$  : large margin, more support vectors ( $\sim$  underfitting)
- ▶  $C \rightarrow +\infty$  : converges in the separable case to the *Hard-Margin* solution

**Rk:** strength of the regularization is inversely proportional to  $C$  (compared with the regularization parameter  $\lambda$  for ridge penalty,  $C \equiv \frac{1}{\lambda}$ )

### Choosing the regularization parameter $C > 0$

- ▶ the optimal  $C$  can be estimated by cross validation
- 👉 performance might not be very sensitive to choices of  $C$  (due to the rigidity of a linear boundary)
- 👉 usually  $C \approx 1$  yields a good trade-off

## Dual problem (nonseparable case)

Introducing the Lagrangian and substituting the first order KKT conditions w.r.t.  $\beta$ ,  $\beta_0$ ,  $\xi$  yields the dual expression

### Dual optimization problem

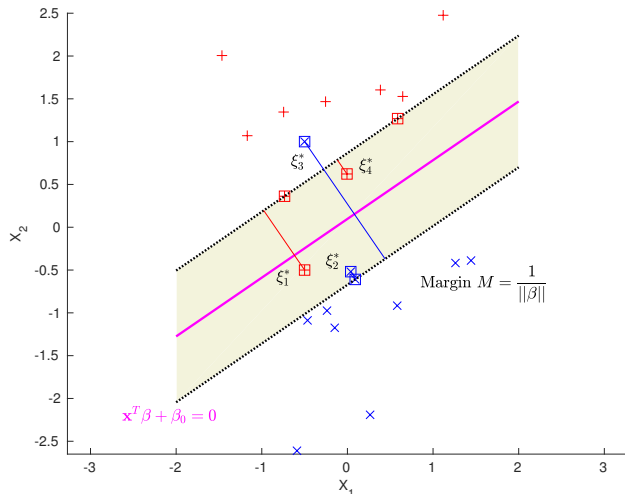
$$\begin{cases} \max_{\alpha} & \tilde{L}(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j, \\ \text{subject to} & 0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

- only difference w.r.t the separable case:  $\alpha_i \leq C$  constraint!
- simple convex optimization problem for which standard numerical procedure are available



## Optimal separating hyperplane

### Soft-Margin example (nonseparable case)



### Vector Supports

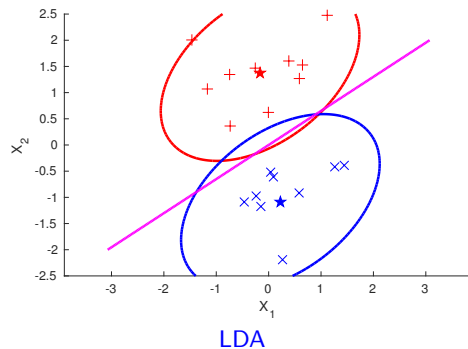
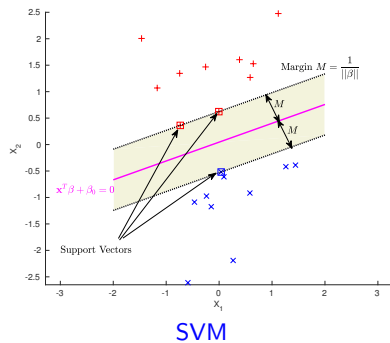
The support vectors are now the points at the margin, inside the margin, or wrongly classified.

$\xi_i^* \equiv M \xi_i \leftarrow$  distance between a support vector and the margin

## Linear discrimination: SVM vs LDA

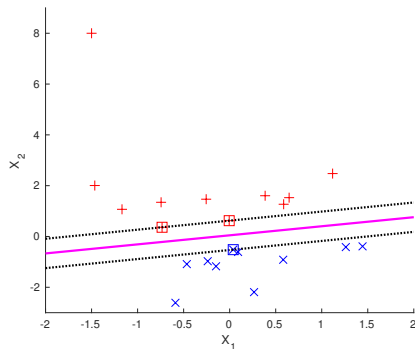
### Linear discrimination

- ▶ Linear Discriminant Analysis (LDA): Gaussian generative model
- ▶ SVM: criterion optimization (maximizing the margin)

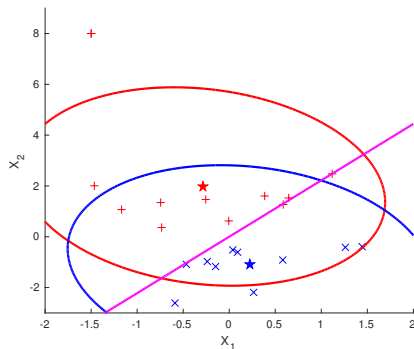


## Linear discrimination: SVM vs LDA (Cont'd)

### Adding one atypical data



SVM

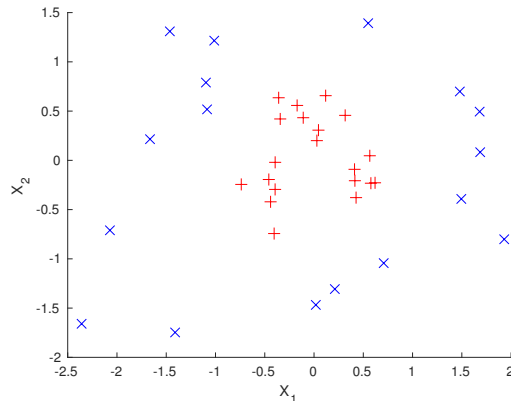


LDA

### SVM property

- Nonsensitive to atypical points (outliers) far from the margin
- 👉 sparse method (information  $\equiv$  support vectors)

## Nonlinear discrimination in the input space



Sometimes a linear separation won't work, whatever the slack variables...

Transformed space  $\mathcal{F}$

- ▶ Choice of a transformed space  $\mathcal{F}$  (expansion space) where the linear separation assumption is more relevant
- ▶ Nonlinear expansion map  $\phi : \mathbb{R}^p \rightarrow \mathcal{F}$ ,  $\mathbf{x} \mapsto \phi(\mathbf{x}) \leftarrow$  enlarged features

## Nonlinear discrimination in the input space

- Non-linear transformation: 'projection' in the space of monomials of order 2.

$$\phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

$$\mathbf{x} \mapsto \phi(\mathbf{x})$$

$$(\mathbf{x}_1, \mathbf{x}_2) \mapsto (\mathbf{x}_1^2, \mathbf{x}_2^2, \sqrt{2}\mathbf{x}_1\mathbf{x}_2)$$

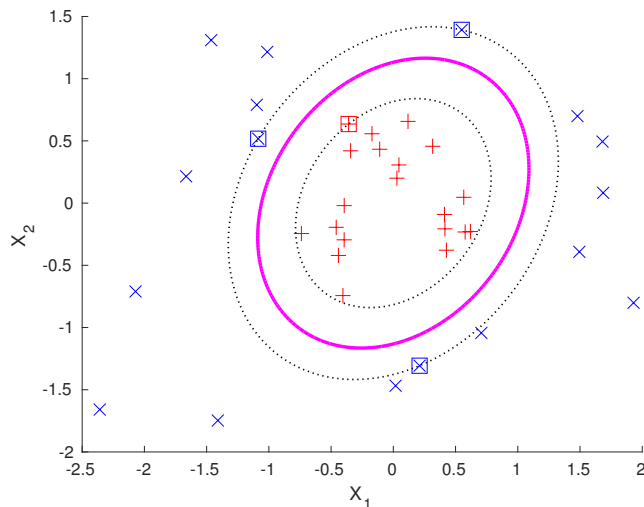
- In  $\mathbb{R}^3$ , the inner product can be expressed as

$$\begin{aligned}\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathbb{R}^3} &= \sum_{i=1}^3 \phi(\mathbf{x})_i \phi(\mathbf{x}')_i \\ &= \phi(\mathbf{x})_1 \phi(\mathbf{x}')_1 + \phi(\mathbf{x})_2 \phi(\mathbf{x}')_2 + \phi(\mathbf{x})_3 \phi(\mathbf{x}')_3 \\ &= \mathbf{x}_1^2 \mathbf{x}'_1{}^2 + \mathbf{x}_2^2 \mathbf{x}'_2{}^2 + 2\mathbf{x}_1 \mathbf{x}_2 \mathbf{x}'_1 \mathbf{x}'_2 \\ &= (\mathbf{x}_1 \mathbf{x}'_1 + \mathbf{x}_2 \mathbf{x}'_2)^2 \\ &= \langle \mathbf{x}, \mathbf{x}' \rangle_{\mathbb{R}^2}^2 \\ &= k(\mathbf{x}, \mathbf{x}').\end{aligned}$$

**Remark:**  $k(\mathbf{x}, \mathbf{x}')$  can be computed directly from the input data without computing  $\phi(\mathbf{x})$  (see later the 'Kernel Trick')!

## Nonlinear discrimination in the input space

►  $X \in \mathbb{R}^2$ ,  $\phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)^T$



Linear separation in the feature space  $\mathcal{F} \Rightarrow$  Nonlinear separation in the input space

## Kernel trick

The SVM solution depends only on the **inner product** between the input features  $\phi(\mathbf{x})$  and the support vectors  $\phi(\mathbf{x}_{\text{margin}})$

### Kernel trick

Use of a kernel function  $k$  associated with an expansion/feature map  $\phi$ :

$$\begin{aligned} k: \mathbb{R}^p \times \mathbb{R}^p &\rightarrow \mathbb{R} \\ (\mathbf{x}, \mathbf{x}') &\mapsto k(\mathbf{x}, \mathbf{x}') \equiv \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle \end{aligned}$$

and the separating hyperplane reads  $h(\mathbf{x}) = \sum_{i=1}^n \hat{\alpha}_i y_i k(\mathbf{x}_i, \mathbf{x}) + \hat{\beta}_0$

### Advantages

- ▶ computations are performed in the original input space: less expensive than in a high dimensional transformed space  $\mathcal{F}$
- ▶ explicit representations of the feature map  $\phi$  and enlarged feature space  $\mathcal{F}$  are not necessary, the only expression of  $k$  is required!
- 🔗 possibility of complex transformations in possible infinite space  $\mathcal{F}$
- 🔗 standard trick in machine learning not limited to SVM (kernel ridge regression, gaussian process, spectral clustering, kernel-PCA ...)

## Kernel function

### Definition (Positive semi-definite kernel)

$k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$  is positive semi-definite is

- ▶  $\forall (\mathbf{x}, \mathbf{x}') \in \mathbb{R}^d \times \mathbb{R}^d, k(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_j, \mathbf{x}_i).$
- ▶  $\forall n \in \mathbb{N}, \forall \xi_1 \dots \xi_n \in \mathbb{R}, \forall \mathbf{x}_1 \dots \mathbf{x}_n \in \mathbb{R}^d, \sum_{i,j}^n \xi_i \xi_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0.$

### Theorem (Mercer Theorem)

*To every positive semi-definite kernel  $k$ , there exists a Hilbert space  $\mathcal{F}$  and a feature map  $\phi : \mathbb{R}^d \rightarrow \mathcal{H}$  such that for all  $\mathbf{x}_i, \mathbf{x}_j$  we have  $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}}.$*



## Operations on kernels

Let  $k_1$  and  $k_2$  be positive semi-definite, and  $\lambda_{1,2} > 0$  then:

1.  $\lambda_1 k_1$ , (multiplication by a positive scalar)
2.  $\lambda_1 k_1 + \lambda_2 k_2$ , (sum of kernels),
3.  $k_1 k_2$ , (product of kernels),
4.  $\exp(k_1)$ , (exponential of kernel),
5.  $(\mathbf{x}_i, \mathbf{x}_j) \mapsto g(\mathbf{x}_i)g(\mathbf{x}_j)k_1(\mathbf{x}_i, \mathbf{x}_j)$ , with  $g : \mathbb{R}^d \rightarrow \mathbb{R}$ , (multiplication by a function)

are all positive semi-definite, hence **valid kernels**.

👉 These operations allow us to create more complicated kernels by combining simple ones.

## Choosing the Kernel function

### Usual kernel functions

- ▶ Linear kernel (  $\mathcal{F} \equiv \mathbb{R}^p$  ) :  $k(x, x') = x^T x'$
- ▶ Polynomial kernel (dimension of  $\mathcal{F}$  increases with the order  $d$ )

$$k(x, x') = (x^T x')^d \quad \text{or} \quad (x^T x' + 1)^d$$

- ▶ Gaussian radial function ( $\mathcal{F}$  with infinite dimension)

$$k(x, x') = \exp \left( -\gamma \|x - x'\|^2 \right)$$

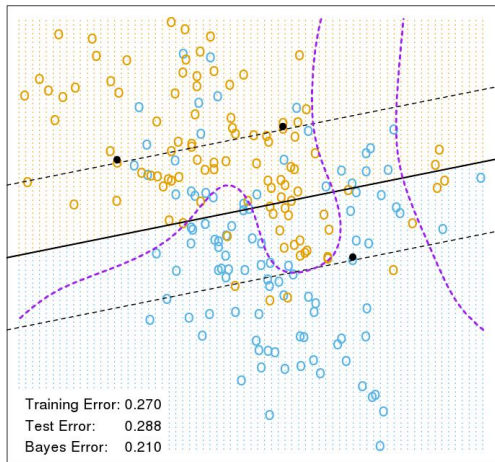
- ▶ Neural net kernel ( $\mathcal{F}$  with infinite dimension)

$$k(x, x') = \tanh \left( \kappa_1 x^T x' + \kappa_2 \right)$$

🔗 standard practice is to estimate the optimal kernel parameters by [cross-validation](#)

## Application: binary data (cf course 2 example)

### Linear kernel

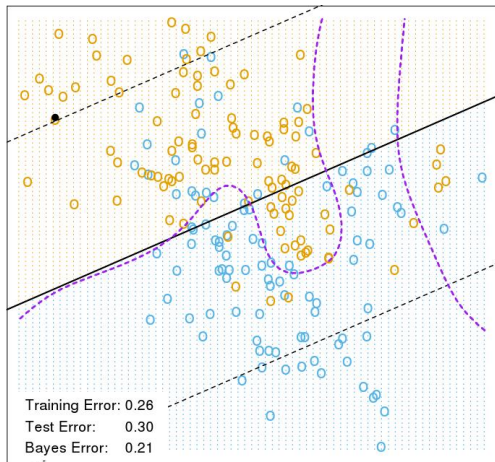


- SVM decision boundary
- - - SVM margin boundaries
- - - Bayes (optimal) decision boundary

$$C = 10000$$

## Application: binary data (cf course 2 example)

### Linear kernel

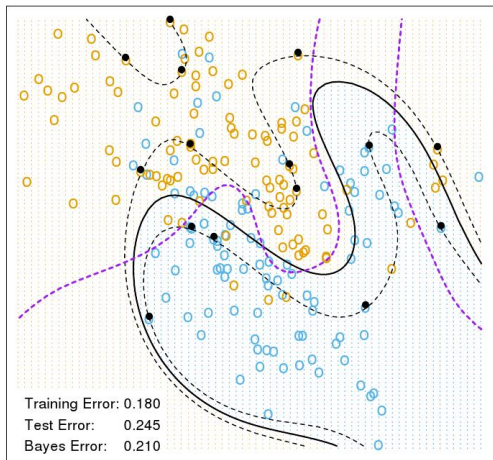


$$C = 0.01$$

- SVM decision boundary
- SVM margin boundaries
- - - Bayes (optimal) decision boundary

## Application: binary data (cf course 2 example)

Polynomial kernel ( $d = 4$ )

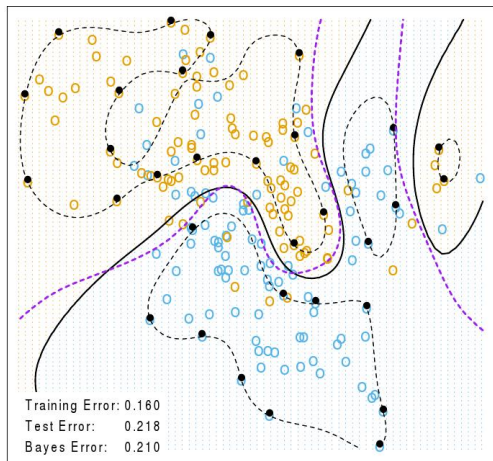


- SVM decision boundary
- - - SVM margin boundaries
- - - Bayes (optimal) decision boundary

$C \approx 1$

## Application: binary data (cf course 2 example)

Gaussian radial kernel ( $\gamma = 1$ )



$C \approx 1$

## Scale your data!

### Scaling of the variables matters!

For instance, with Gaussian kernel

$$\begin{aligned}k(x, x') &= \exp\left(-\gamma \|x - x'\|^2\right) \\ &= \exp\left(-\gamma \sum_{i=1}^p (x_i - x'_i)^2\right),\end{aligned}$$

the variables that have the greatest magnitudes are favored to compute distances or inner-products.

### Practical advices

- ▶ If the variables are in different units, scaling each is **strongly recommended**.
- ▶ If they are in the same units, you might or might not scale the variables (depend on your problem)

### Usual scaling methods

- ▶ **normalization** in  $[0, 1]$ :  $\tilde{x}_i = \frac{x_i - \min_i}{\max_i - \min_i}$
- ▶ **standardization** to get zero mean and unit variance:  $\tilde{x}_i = \frac{x_i - \mu_i}{\sigma_i}$

## Multiclass SVM

- ▶  $Y \in \{1, \dots, K\} \leftarrow K$  classes

Standard approach: direct generalization by using multiple binary SVMs

### OVA: one-versus-all strategy

- ▶  $K$  classifiers between one class (+1 label) versus all the other classes (−1 label)
- 👉 classifier with the highest confidence value (e.g. the maximum distance to the separator hyperplane) assigns the class

### OVO: one-versus-one strategy

- ▶  $\binom{K}{2} = K(K-1)/2$  classifiers between every pair of classes
- 👉 majority vote rule: the class with the most votes determines the instance classification

Which to choose? if  $K$  is not too large, choose OVO



## SVM vs Logistic regression (LR)

- ▶ When classes are nearly separable, SVM does better than LR. So does LDA.
- ▶ When not, LR (with ridge penalty) and SVM are very similar
- ▶ If one wants to estimate probabilities for each class, LR is the natural choice
- ▶ For non linear boundaries, kernel SVMs are popular. Can use kernels with LR and LDA as well, but computations are more expensive.

## Conclusions on Support Vector Machines

### SVM properties

- ▶ **model free approach** based on a maximum margin criterion: may be very efficient for real-world data (but do not directly provide probability estimates nor variable importance weights)
- ▶ **memory efficient** sparse solution characterized by the only support vectors
- ▶ **versatile** algorithm: different choices of kernels to make a **nonlinear classification** in the original input space by performing an implicit linear classification in a higher dimensional space
- ▶ Possible **extensions** to other tasks than classification like **regression** (*support vector regression*) or **anomaly detection** (*one-class SVM*)
- ▶ **effective in high dimensional** spaces even when  $p > n$ .
- ▶ **computationally expensive** to train for large  $n$  data sets: cost of the optimization procedure to solve the quadratic problem scales from  $O(pn^2)$  to  $O(pn^3)$  operations depending on the training set.
- ▶ **popular algorithm**, with a large literature

## Perspectives on 'Black Box' (model free) approaches

### Random Forests

- ▶ involve decision tree to split the prediction space in simple regions
- ▶ combine multiple decision trees to yield a single consensus prediction
- 👉 method able to scale efficiently to high dimensional data and large data sets

### Deep Neural Nets

- ▶ Neural Nets with multiple hidden layers between input and output ones
- ▶ many variants of deep architectures (Recurrent, Convolutional,...) used in specific domains (speech, vision, ...)
- ▶ very computationally expensive to train due to the high number of parameters
- ▶ supported by empirical evidence
- 👉 dramatic performance jump for some big data applications

## Outline for model free approaches

### Support Vector Machine (SVM)

- Separating Hyperplane

- Separable case

- Nonseparable case

- Linear discrimination: comparison of SVM vs LDA

- Transformed space and Kernel function

- Examples

- Multiclass SVM

- SVM vs Logistic regression (LR)

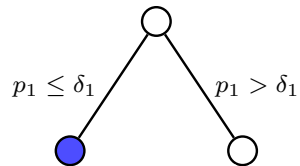
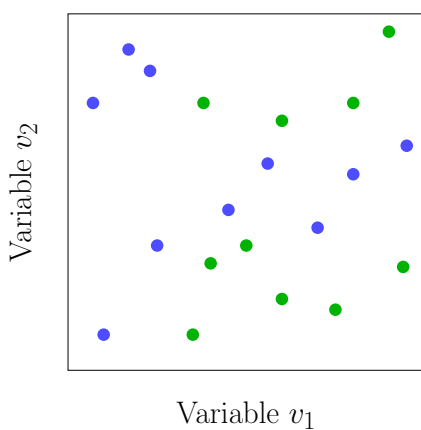
### Conclusions

### Appendix: Some words on Random Forests

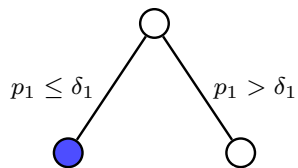
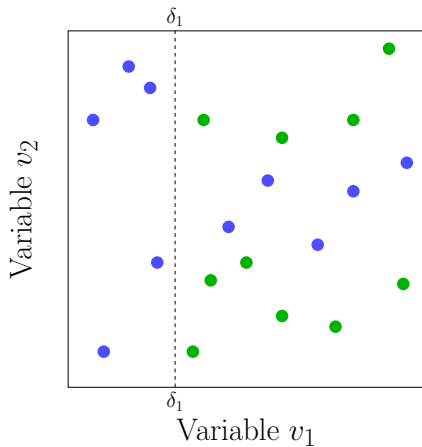
## Random Forests

- ▶ Introduced in 2001 (Breiman)
- ▶ Model free and non linear
- ▶ Build a large collection of de-correlated trees and average them
- ▶ Combination of weak learners

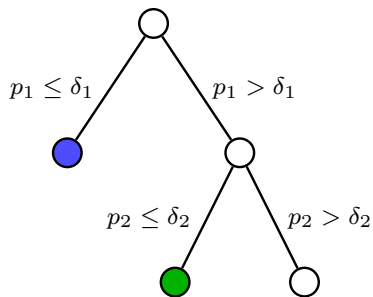
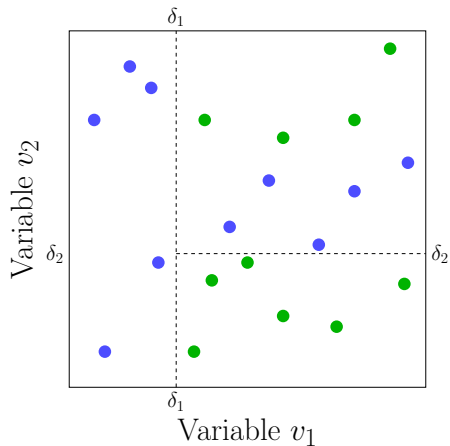
## Decision trees



## Decision trees

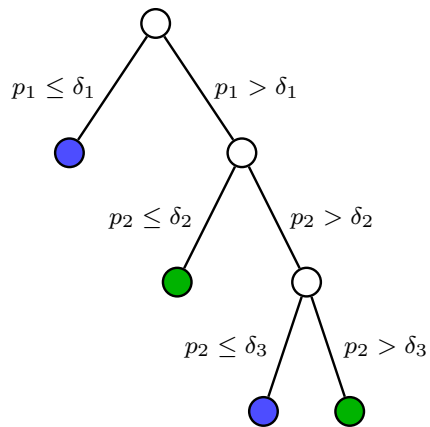
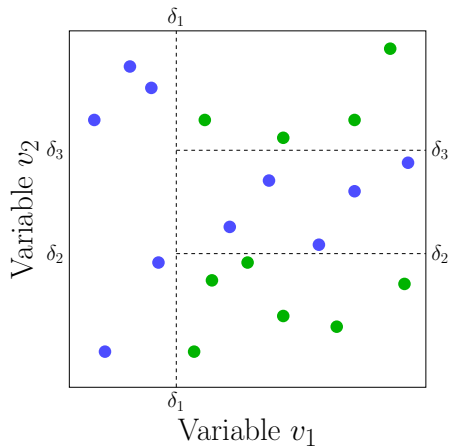


## Decision trees



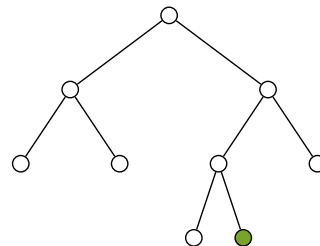
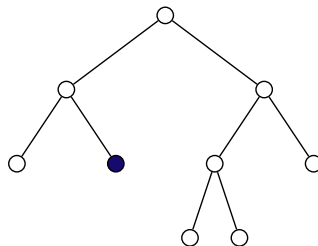
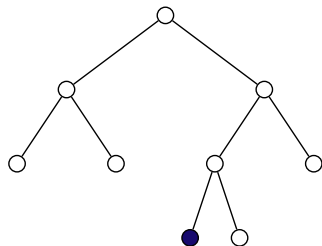


## Decision trees



## Random Forests

- ▶ For each tree:
  - ▶ Draw bootstrap sample  $X^b$  for training sample
  - ▶ Learn tree, for each node
    - ▶ select  $m$  features from the initial  $p$  features
    - ▶ Find the best split (e.g. Gini index, entropy ...)



## Application: binary data

