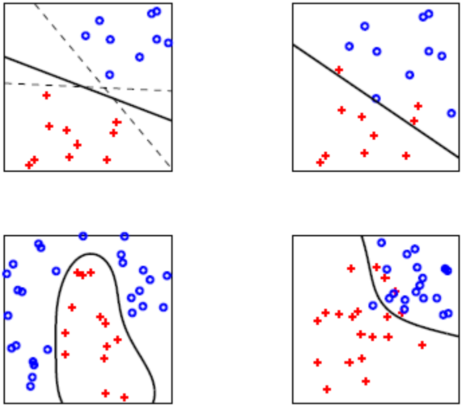
Support Vector Machines

Marine Demangeot

8 février 2017

Support Vector Machines

- Supervised learning algorithm
- Original SVM algorithm invented by Vladimir Vapnik (1990’s)



Sommaire

Model

- **input** : $x_1, \ldots, x_n \in \mathbb{R}^2$ $\quad x_i = \left(x_{i,(1)}, x_{i,(2)}\right)$
- **output** : $y_1, \ldots, y_n \in \{-1; 1\}$: two-class classification problem

$\hookrightarrow$ *training data set*

The training data set is linearly separable in the (two-dimensional) data space

$\iff \quad \exists (w, b) \in \mathbb{R}^2 \times \mathbb{R}$ s.t. $\forall i \in [\![1; n]\!]$:

$$y_i = g\left[w^t x_i + b\right]$$

with $g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0 \end{cases}$

Separating hyperplan

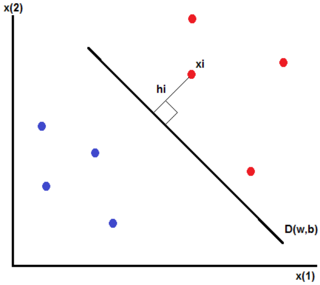


FIGURE – Binary discrimination

- $D(w,b) = \{x \in \mathbb{R}^2 : w^t x + b = 0\}$ decision boundary
- $h_i = \frac{|w^t x_i + b|}{\|w\|_2}$

Multiple separating hyperplanes

- $\forall k > 0,\ \ g\left[w^t x_i + b\right] = g\left[k \cdot w^t x_i + k \cdot b\right]$

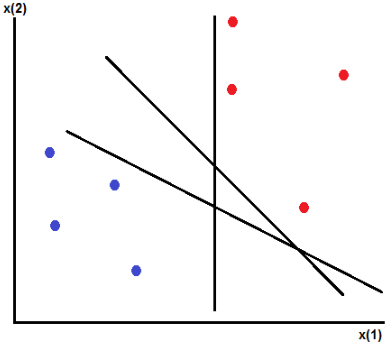


FIGURE – Binary discrimination : multiple solutions

Optimal separating hyperplan

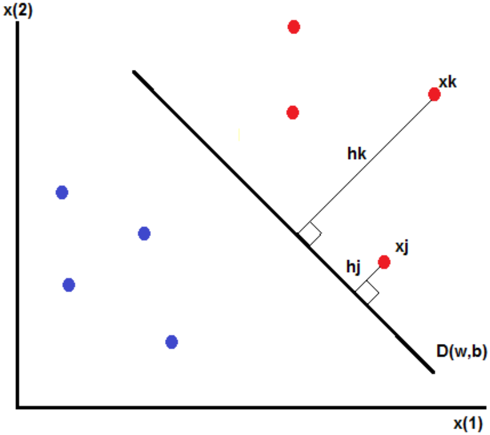
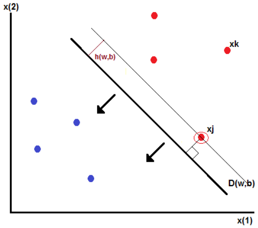
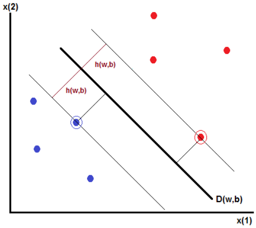


FIGURE – Confidence about discrimination

Optimal separating hyperplan



FIGURE



FIGURE

1. margin : $h(w,b) = \min_{i=1\ldots,n} \; h_i$
2. $\arg\max_{w,b} \; \{h(w,b)\}$ **maximum margin solution**

Optimization problem

$$h_i = \frac{|w^t x_i + b|}{\|w\|_2} = \frac{y_i(w^t x_i + b)}{\|w\|_2}$$

$$\arg\max_{w,b} \left\{ \frac{1}{\|w\|_2} \underbrace{\min_{i=1\dots,n} \left(y_i \left[w^t x_i + b \right] \right)}_{\widehat{h}} \right\} \tag{1}$$

$$s.t. \qquad y_i \left[w^t x_i + b \right] \geq \widehat{h}, \qquad i = 1, \dots, n$$

Primal optimization problem

Scaling contraint : $\widehat{h} = 1$
i.e. $y_i \left(w^t x_i + b \right) = 1$ for the point i that is closest to $D(w, b)$.

Primal optimization problem :

$$\begin{aligned} & \underset{w,b}{\arg\min} \quad \frac{1}{2} \|w\|_2^2 \\ & s.t. \qquad y_i \left[w^t x_i + b \right] \geq 1, \qquad i = 1, \ldots, n \end{aligned} \tag{2}$$

→ minimizing a convex quadratic function subject to a set of linear inequality constraints

Lagrangian function

We want to minimize :

$$\mathscr{L}(w,b,\alpha) = \frac{1}{2}\|w\|_2^2 - \sum_{i=1}^{n} \alpha_i \left\{ y_i \left[w^t x_i + b \right] - 1 \right\}$$

- $\alpha = (\alpha_1, \ldots, \alpha_n)^t$

Thus,

$$\begin{cases} \nabla_w \mathscr{L}(w,b,\alpha) = 0 & \Rightarrow \quad w = \sum_{i=1}^{n} \alpha_i y_i x_i \\ \frac{\delta}{\delta b}\mathscr{L}(w,b,\alpha) = 0 & \Rightarrow \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases}$$

Lagrange duality

Dual optimization problem :

$$\begin{array}{lll} \underset{\alpha}{\arg\max} & \frac{1}{2}\sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \; x_i^t x_j & \\ s.t. & \alpha_i \geq 0, & i = 1, \ldots, n \\ & \sum_{i=1}^{n} \alpha_i y_i = 0 & \end{array} \quad (3)$$

→ minimizing a convex quadratic function subject to a set of linear inequality constraints

Finding α^*, we have :

$$\begin{cases} w^* = \sum_{i=1}^{n} \alpha_i^* y_i x_i \\ b^* = -\dfrac{\max_{i:y_i=-1} {w^*}^t x_i + \min_{i:y_i=1} {w^*}^t x_i}{2} \end{cases}$$

Prediction

new point input : x_{n+1}

$$
\begin{aligned}
y_{n+1} &= g\left[{w^*}^t x_{n+1} + b^*\right] \\
&= g\left[\left(\sum_{i=1}^n \alpha_i^* y_i x_i\right)^t x_{n+1} + b^*\right] \\
&= g\left[\sum_{i=1}^n \alpha_i^* y_i x_i^t x_{n+1} + b^*\right] \\
&\quad \textit{where } \alpha_i = 0 \textit{ if } y_i\left(w^t x_i + b\right) > 1 \\
&= g\left[\sum_{i \in \mathscr{S}}^n \alpha_i^* y_i \left[x_i^t x_{n+1}\right] + b^*\right]
\end{aligned}
$$

with $\mathscr{S} = \{i : y_i\left(w^t x_i + b\right) = 1\}$ where x_i is called a **support vector**

→ **memory efficient** : once the model is training, only a subset of the training data, the support vectors, i.e. the points lying on the optimal margins, are used to calculate the output for a new point input.

Support vectors

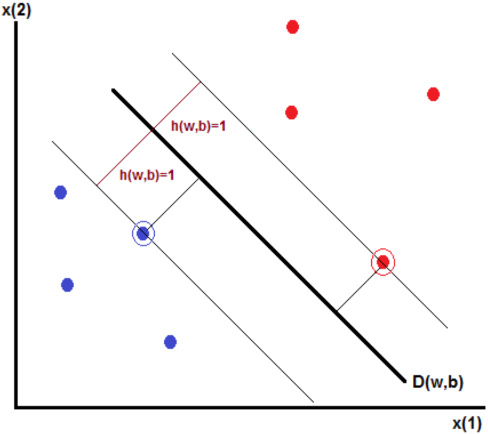
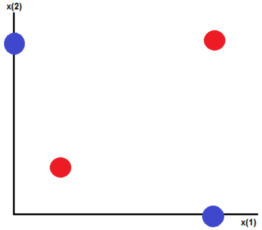


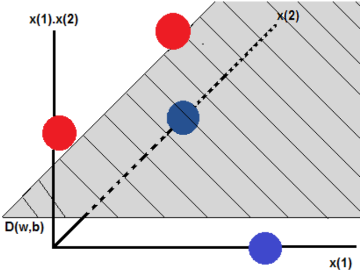
FIGURE – Support vectors

Sommaire

An example of non linear separability



FIGURE



FIGURE

$$\Phi(x) = (x_{(1)}, x_{(2)}, x_{(1)}x_{(2)})$$: feature mapping

Model

- **input** : $x_1, \dots, x_n \in \mathbb{R}^d$
- **output** : $y_1, \dots, y_n \in \{-1; 1\}$: two-class classification problem

$\hookrightarrow$ *training data set*

The training data set is linearly separable in the featuring space

$$\iff \quad \exists (w, b) \in \mathbb{R}^d \times \mathbb{R} \text{ s.t. } \forall i \in [\![1; n]\!] :$$

$$y_i = g\left[w^t \Phi(x_i) + b\right]$$

$\rightarrow$ replace x_i by $\Phi(x_i)$ in the primal or dual optimization problem.

$\Phi(x_i)$ may be very expensive to calculate.

Kernels

Let Φ a feature mapping.
The corresponding Kernel is :

$$K(x;z) = \Phi(x)^t \Phi(z), \qquad \forall x, z \in \mathbb{R}^d$$

K may be very inexpensive to calculate.

ex : $K(x,z) = (x^t z)^2 = \sum_{l,m=1}^{d} (x_{(l)} x_{(m)})(z_{(l)} z_{(m)}) = \Phi(x)^t \Phi(z)$

where $\Phi(x)^t = (x_{(l)} x_{(m)})_{1 \leq l,m \leq d}$

Calculating time $\begin{cases} K(x,z) & : \quad O(d) \text{ time} \\ \Phi(x) & : \quad O(d^2) \text{ time} \end{cases}$

We can replace, in the dual optimization problem, $x_i^t x_j$ by $K(x_i, x_j)$

Kernels

So we can get SVMs to learn in the high dimensional feature space but without ever having to explicitly find or represent vectors $\Phi(x)$, just specifying K. But how to know if the chosen function K is a valid kernel for your optimization problem ?

Let $x_1, \dots, x_n \in \mathbb{R}^d$
Kernel matrix : $K = (K_{i,j})_{1 \le i,j \le n}$ where $K_{i,j} = K(x_i, x_j)$

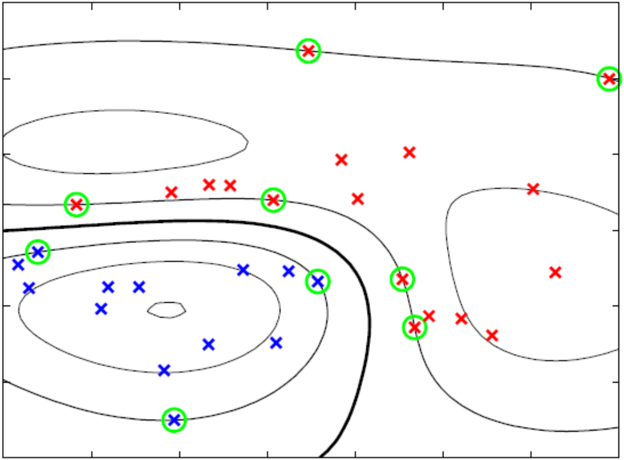
Theorem (Mercer)

Let $K : \mathbb{R}^n \times \mathbb{R}^n$ be given.
K is a valid (Mercer) kernel if and only if, for any $x_1, \dots, x_n \in \mathbb{R}^d$, $n < \infty$, the corresponding kernel matrix is symmetric positive semi-definite.

- Gaussian kernel : $K(x,z) = \left(-\frac{\|x-z\|^2}{2\sigma^2}\right)$
- Polynomial kernel : $K(x,z) = (x^t z)^p$

σ and p must be chosen

Example of discrimination with gaussian kernel



FIGURE

Sommaire

Outliers

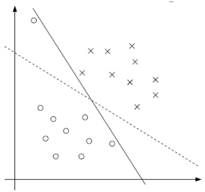


FIGURE – Outliers

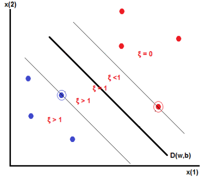


FIGURE – Slack variables

→ We want to allow some of the training points to be misclassified

slack variables :

$$\xi_i = \begin{cases} 0 & \text{if } y_i(w^t\Phi(x_i)+b) \geq 1 \\ 1 - y_i(w^t\Phi(x_i)+b) > 0 & \text{if } y_i(w^t\Phi(x_i)+b) < 1 \end{cases}$$

When x_i is on the *wrong side* of the margin, the penalty ξ_i increases with the distance from that boundary

Optimization problem with regularization

Primal optimization problem :

$$\begin{aligned} \arg\min_{w,b} \quad & \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n}\xi_i \\ s.t. \quad & y_i\left[w^t\Phi(x_i)+b\right] \geq 1-\xi_i, \quad i=1,\ldots,n \\ & \xi_i \geq 0, \quad i=1,\ldots,n \end{aligned} \tag{4}$$

$C > 0$ controls the **trade-off** between **minimizing training errors** (i.e. ensuring that most slack variables are null) and **controlling the model complexity** (i.e. making the margin large). Increasing C gives more importance to the minimizing training errors goal.

C must be chosen

Dual optimization problem :

$$
\begin{aligned}
\arg\max_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y_i y_j \alpha_i \alpha_j \, K(x_i, x_j) \\
s.t. \quad & 0 \le \alpha_i \le C, \qquad i = 1, \ldots, n \\
& \sum_{i=1}^{n} \alpha_i y_i = 0
\end{aligned}
\tag{5}
$$

Finding α^*, we can calculate w^* and b^*.

- x_i is a **support vector** $\iff y_i(w^t x_i + b) = 1 - \xi_i$
- only support vectors contribute to the predictive model

$\alpha_i < C \Rightarrow \xi_i = 0$: x_i lie on the margin
$\alpha_i = C \Rightarrow \xi_i > 0$: x_i lie inside the margins and can be correctly classified ($\xi_i \le 1$) or misclassified ($\xi_i > 1$)

Model selection

To select the model : compute, with the validation data set, the confusion matrix when the following parameters are varying :

- kernel type : gaussian, polynomial...
- parameters of the kernel : p, σ
- C

Conclusion

☺

- Convex optimization : the solution is the global minimum not a local minimum
- Effective in high dimensional spaces (but curse of dimensionality problem remains)
- Use a subset of training points in the decision function (memory efficient)
- Different kernel functions can be specified for the decision function

☹

- Do not directly provide probability estimates (these are calculated using an expensive five-fold cross-validation)
- It doesn't perform well, when we have large data set because the required training time is higher

Conclusion

Other possibilities :

- Multiclass SVMs (when y has more than two labels)
- SVMs for regression (when y is a continuous variable)

Bibliography

- Andrew Ng's lecture notes
- Bishop, C. (2007). Pattern Recognition and Machine Learning (Information Science and Statistics), 1st edn. 2006. corr. 2nd printing edn. Springer, New York.
- Scikit learn - Support Vector Machines