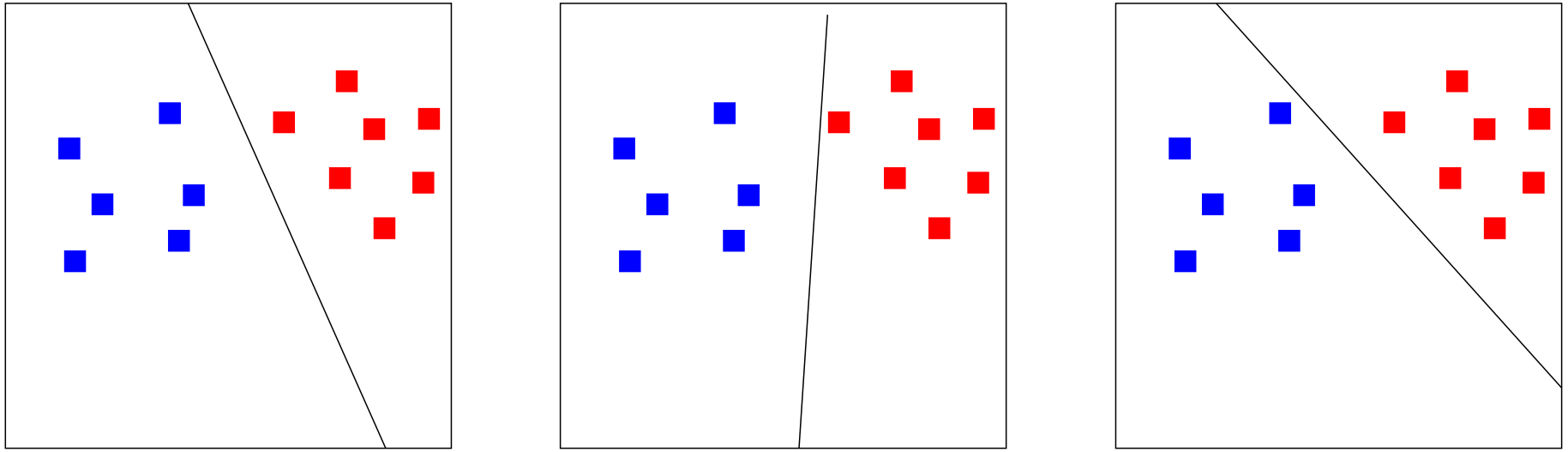
SVM

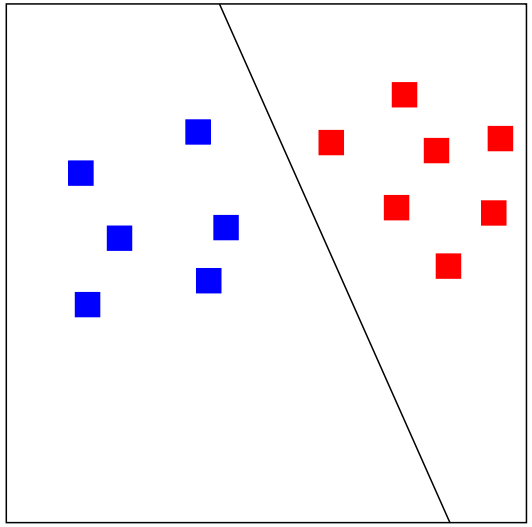
Séparation(s) linéaire(s)

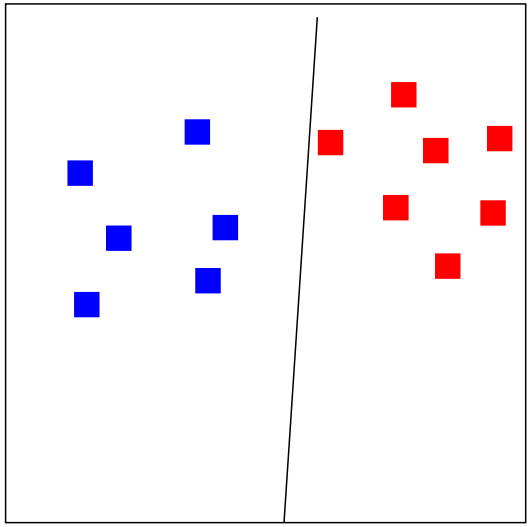
Plusieurs (une infinité) de séparations linéaires possibles …

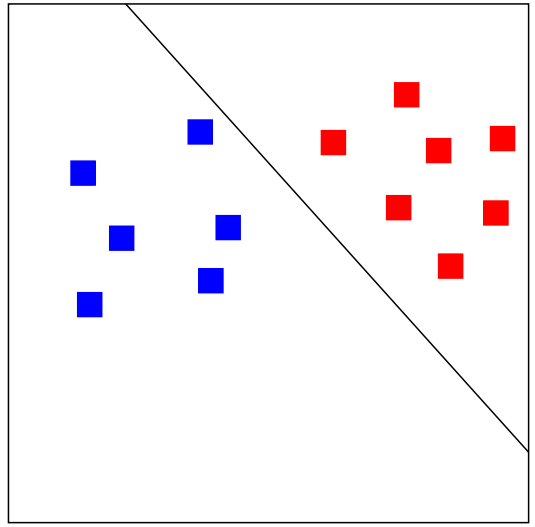


Séparation(s) linéaire(s)

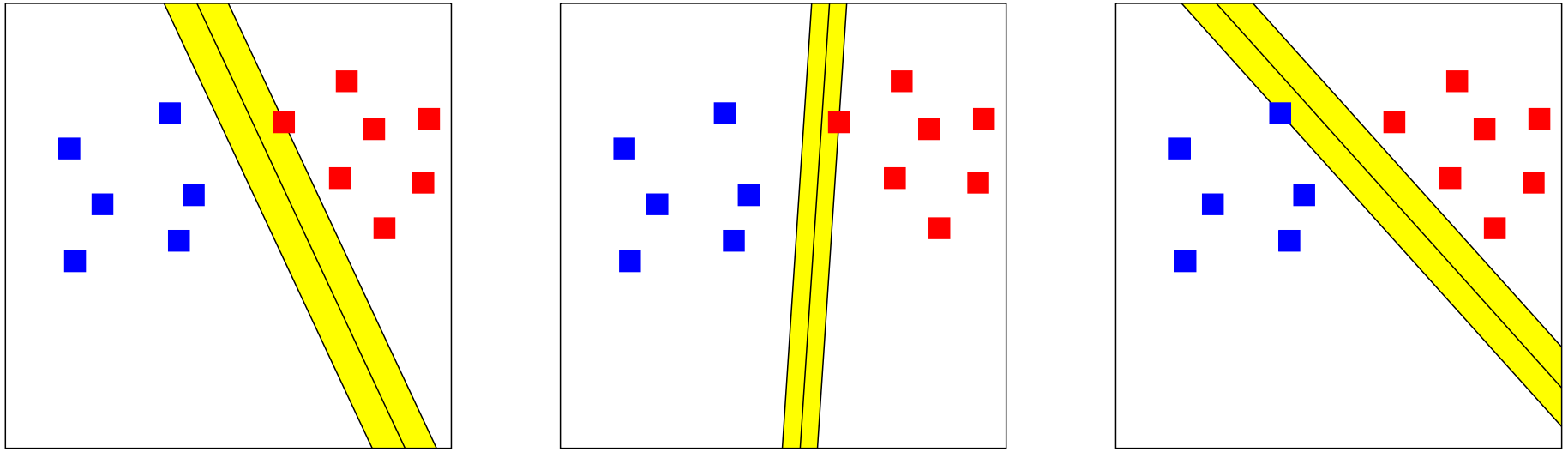
Y en a-t-il une meilleure ?





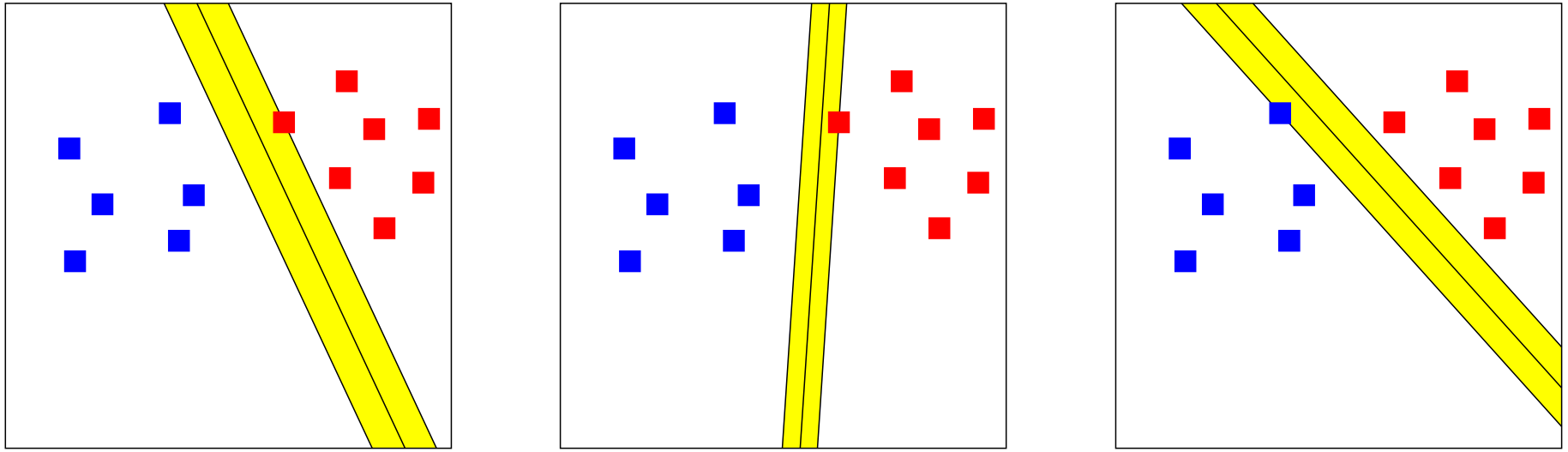


Notion de marge…



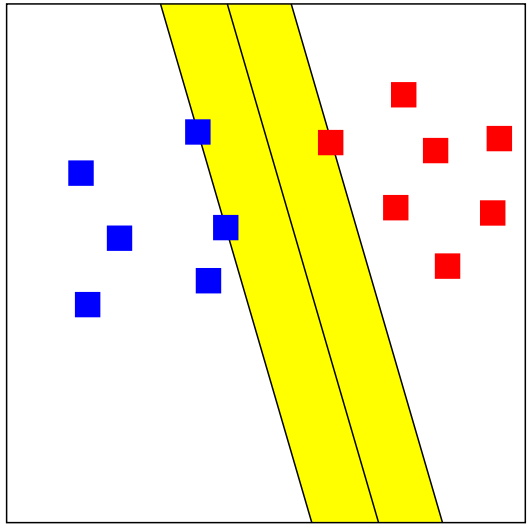
Notion de marge…

Y en a-t-il une meilleure ?



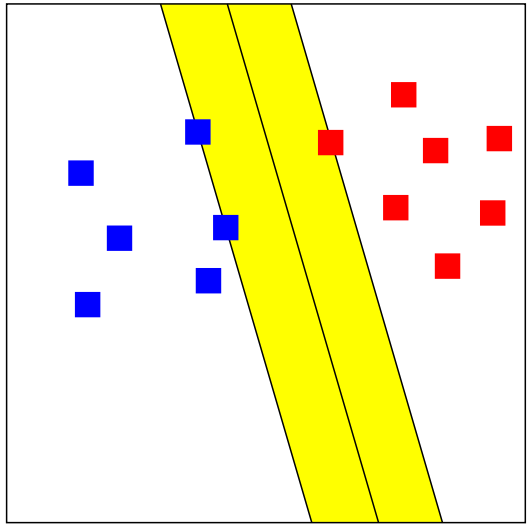
Notion de marge…

Y en a-t-il une meilleure ?



Notion de marge…

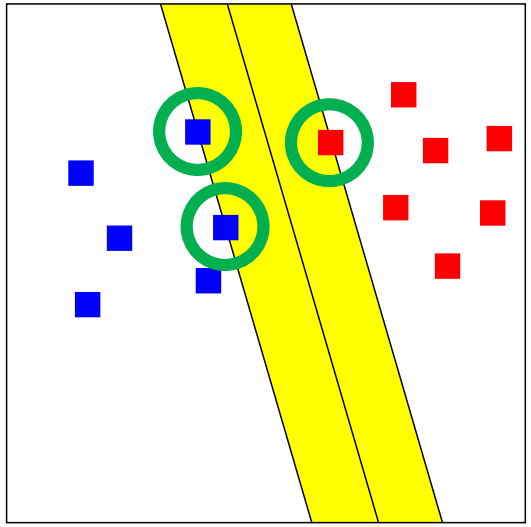
Nouveau problème : trouver les W qui maximisent la marge !



Notion de marge…

Nouveau problème : trouver les W qui maximisent la marge !

C.à.D. : Trouver les vecteurs supports :



Programmation quadratique à la rescousse !

Minimiser :

$$\frac{1}{2}\alpha^T \begin{bmatrix} y_1 y_1 x_1^T x_1 & \cdots & y_1 y_N x_1^T x_N \\ \vdots & \ddots & \vdots \\ y_N y_1 x_N^T x_1 & \cdots & y_N y_N x_N^T x_N \end{bmatrix} \alpha + [-1 \quad \dots \quad -1]\alpha$$

Sous contraintes :

$$y^T \alpha = 0$$

Avec :

$$\alpha \geq 0$$

Programmation quadratique à la rescousse !

Une fois α obtenu, on peut retrouver W :

$$W = \sum_{n=1}^{N} \alpha_n y_n x_n$$

Programmation quadratique à la rescousse !

Une fois α obtenu, on peut retrouver W :

$$W = \sum_{n=1}^{N} \alpha_n y_n x_n$$

Attention, il nous manque w_0 !

Programmation quadratique à la rescousse !

Pour trouver w_0 :

1 – Choisir un x_n tq $\alpha_n > 0$ c.à.d un Vecteur Support !

2 – Sachant que $y_n(w^T x_n + b) = 1$

3 – Cela nous donne :

$$b = \frac{1}{y_n} - \sum_i w_i x_{n_i}$$

Machine à noyaux

Retour sur les SVMs

Si nos exemples sont de grande dimension,

$$\begin{bmatrix} y_1 y_1 x_1^T x_1 & \cdots & y_1 y_N x_1^T x_N \\ \vdots & \ddots & \vdots \\ y_N y_1 x_N^T x_1 & \cdots & y_N y_N x_N^T x_N \end{bmatrix}$$

Sera difficile à calculer !

Retour sur les SVMs

Projection des entrées dans un autre espace (le retour) :

$$\begin{bmatrix} y_1 y_1 z_1^T z_1 & \cdots & y_1 y_N z_1^T z_N \\ \vdots & \ddots & \vdots \\ y_N y_1 z_N^T z_1 & \cdots & y_N y_N z_N^T z_N \end{bmatrix}$$

Si l'espace est de dimension supérieure à l'espace de départ, cela devrait être encore pire !

Retour su

Projection des

Si l’espace est art, cela devrait être en

Retour sur les SVMs

Projection des entrées dans un autre espace (le retour) :

$$\begin{bmatrix} y_1 y_1 z_1^T z_1 & \cdots & y_1 y_N z_1^T z_N \\ \vdots & \ddots & \vdots \\ y_N y_1 z_N^T z_1 & \cdots & y_N y_N z_N^T z_N \end{bmatrix}$$

Si l'espace est de dimension supérieure à l'espace de départ, cela devrait être encore pire !

Cela dépend du type de transformation !

Retour sur les SVMs

Cela dépend du type de transformation !

Nous n'avons besoin que de l'existence de la possibilité d'effectuer produit scalaire dans le nouvel espace !

$$\begin{bmatrix} y_1 y_1 K(x_1, x_1) & \cdots & y_1 y_N K(x_1, x_N) \\ \vdots & \ddots & \vdots \\ y_N y_1 K(x_N, x_1) & \cdots & y_N y_N K(x_N, x_N) \end{bmatrix}$$

Différents noyaux :

Noyau Polynomial de degré Q:

$$K(x_n, x_m) = (1 + {x_n}^T x_m)^Q$$

Noyau à Base Radiale :

$$K(x_n, x_m) = e^{-{x_n}^2} e^{-{x_m}^2} e^{2x_n x_m}$$

...vaux :

Equivalent à une projection dans un espace de dimension infinie !

...egré Q:

$$K(x_n, x_m) = (1 + x_n{}^T x_m$$

Sans augmentation du nombre de paramètres !

Noyau à Base Radiale :

$$K(x_n, x_m) = e^{-x_n{}^2} e^{-x_m{}^2} e^{2x_n x_m}$$

Conclusion

Recommandation et pièges

Généraliser

≠

Minimiser l'erreur sur les exemples

Recommandation et pièges

Quel modèle choisir ?

Privilégier les modèles simples avant tout !

L’explication (le modèle), la plus simple est la plus probable !

Quels autres modèles ?

Tous les modèles que nous avons vus sont utiles et continuent d'être améliorés !

Quels autres modèles ?

Le monde de l'apprentissage artificiel est très vaste …

Ouverture

Apprentissage par renforcement

Très utile pour le Jeu (vidéo) !

- Peut être utilisé avec les modèles neuronaux que nous avons vu !
- Idéal dans les cas ou les exemples étiquetés sont rares et arrivent au cours du temps.
- Q-Learning : indépendant du modèle

Metaheuristiques

Nous ne faisons qu'estimer des paramètres !

On peut utiliser un algorithme génétique pour trouver les poids d'un réseau de neurones à la place de la descente de gradient !

etc.

DeepLearning

La nouvelle classe de techniques ‘à la mode’, notamment les Convolutional Neural Networks,

Intrinsèquement liée au ‘Big Data’

Etat des lieux :

http://slideshot.epfl.ch/play/khnnunGF0elc

Prochaines étapes de votre parcours :

Recommandations quasi-exhaustives pour le ML :

http://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf

« Best MOOC Ever ! » :

http://work.caltech.edu/telecourse.html