Option MAFQ

Apprentissage statistique

laurent.carraro@telecom-st-etienne.fr
www.telecom-st-etienne.fr/Carraro

Rappel introduction

- Vocabulaire
  - apprentissage :

    méthode permettant à un automate d'évoluer selon un processus dit d'apprentissage (à partir d'essais/exemples)
  - exemples :
    - **apprentissage du petit enfant**
    - reconnaissance de caractères
    - mise en évidence de facteurs de risques en santé
    - taxonomie
    - prévision de risques (assurances, finance, industrie...)

Types d'apprentissage

- Inputs et outputs :
  - chaque exemple est décrit par :
    - un input : conditions de l'essai
    - un output : résultat (pas toujours connu, cf. taxonomie)
- Apprentissage supervisé :
  - un expert peut classer les exemples

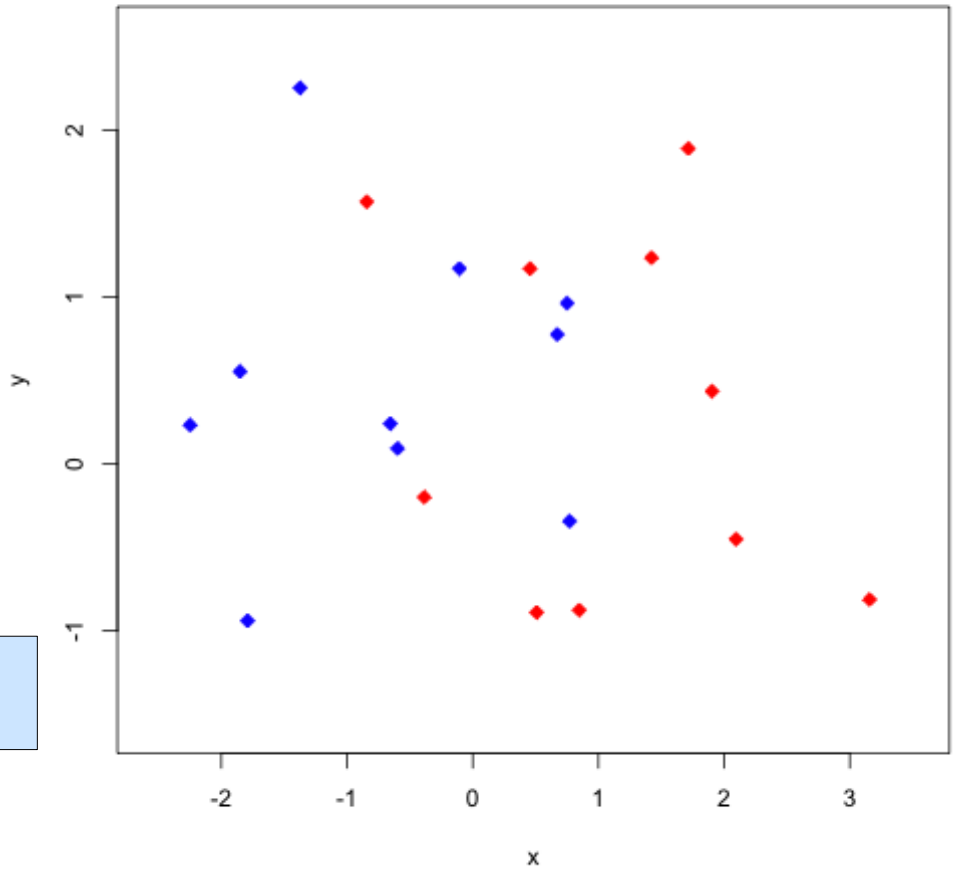
    pour chaque exemple sont disponibles input et output
- Apprentissage non supervisé :
  - aucun expert n'est disponible, même pas un proxy
- Classification et régression

De la régression linéaire aux k-NN

Un exemple simulé en 2D

- Double simulation :
  - simulation de 10 points selon la loi $N(m_1,\Sigma)$, puis la loi $N(m_2,\Sigma)$
    vecteurs M_1 et M_2 de taille 10, formés de points du plan.

M_1 en rouge, M_2 en bleu

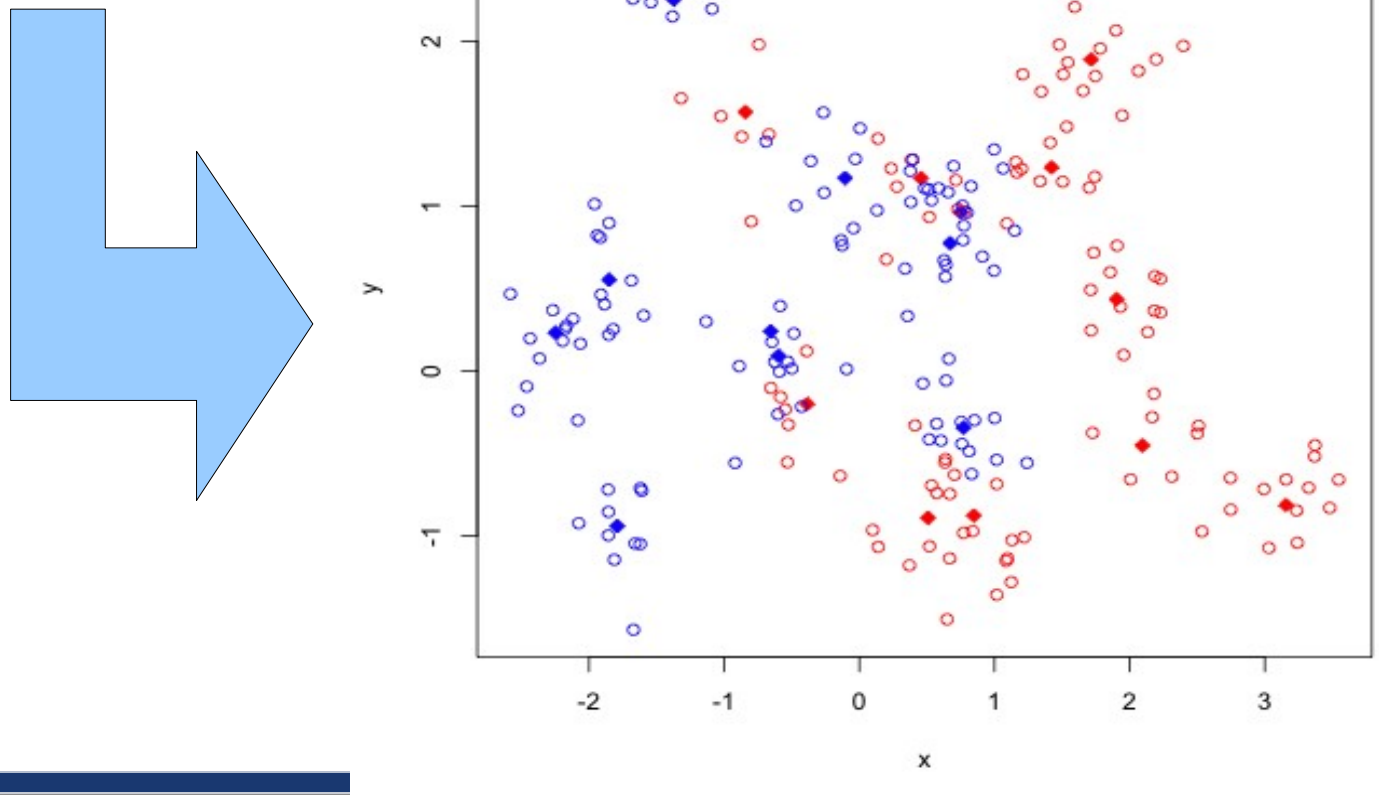


Exemple simulé

- deuxième simulation :

  on simule un échantillon de taille 100, mélange des lois normales $N(M_{1,i}, \Sigma')$ où i=1,...,10

  idem avec M_2



Comment séparer « au mieux » le nuage ?

- Illusoire de séparer parfaitement
- Formalisation :

Input : vecteur X de composantes X_j

Output : variable Y ou groupe G

Ensemble d'apprentissage (**X**, **G**)

Learner : variable $\hat{Y}(X)$ ou groupe $\hat{G}(X)$

à chaque tirage (X, G) est de loi $\frac{1}{2}\mu_1 \otimes \delta_1 + \frac{1}{2}\mu_2 \otimes \delta_2$.

Mesurer l'écart entre les v.a. G et $\hat{G}(X)$

Fonction de coût

Soit L une fonction de coût :

- $L : \{0,1\}^2 \quad [0,+\infty[$ NB : 1=rouge=population 1, 0=bleu=population 2

On suppose $L(1,1)=L(0,0)=0$ et $L(1,0)=L(0,1)=\alpha>0$

$$EPL(\hat{G}) = \mathbf{E}\left[L(G,\hat{G}(X))\right]$$ Expected Prediction Loss

Minimisation de EPL :

$$\hat{G}(X) = \begin{cases} 1 \text{ si } P(G=1/X=x) > P(G=0/X=x) \\ 0 \text{ si } P(G=1/X=x) < P(G=0/X=x) \\ ? \text{ si } P(G=1/X=x) = P(G=0/X=x) \end{cases}$$

$\hat{G} = \hat{G}_B$ est le classifieur de Bayes

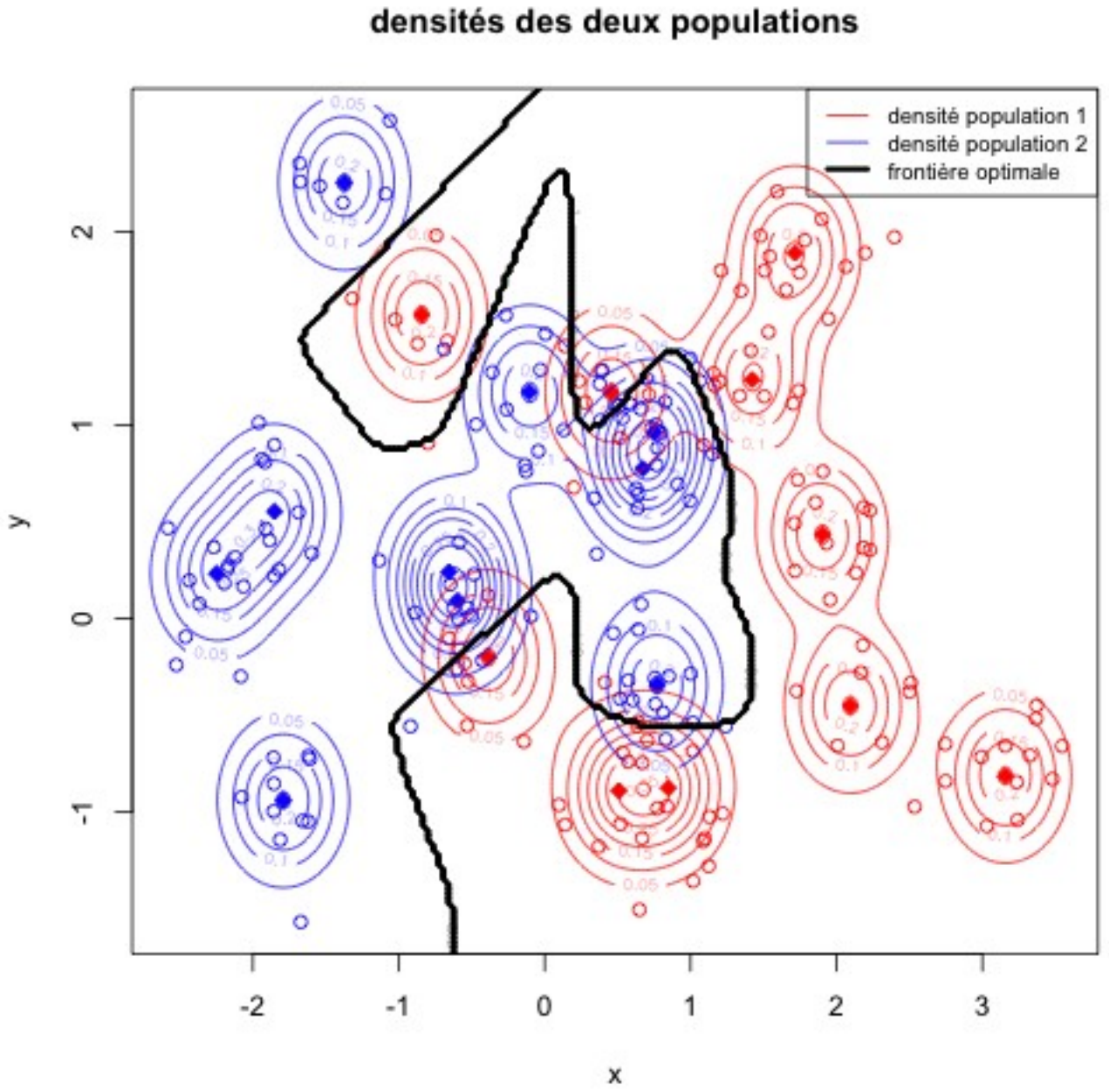
$EPL(\hat{G}_B)$ est le risque de Bayes ou risque optimal

Densités des points simulés

- Graphique RGL

- Contours et frontière optimal

86,5% de points bien classés



Séparation optimale à partir des échantillons ?

- Si les densités sont connues :

  estimation de densité ?

- Régression linéaire :

$$\hat{Y}(x) = \hat{\beta}_0 + \hat{\beta}_1 \, x_1 + \hat{\beta}_2 \, x_2$$

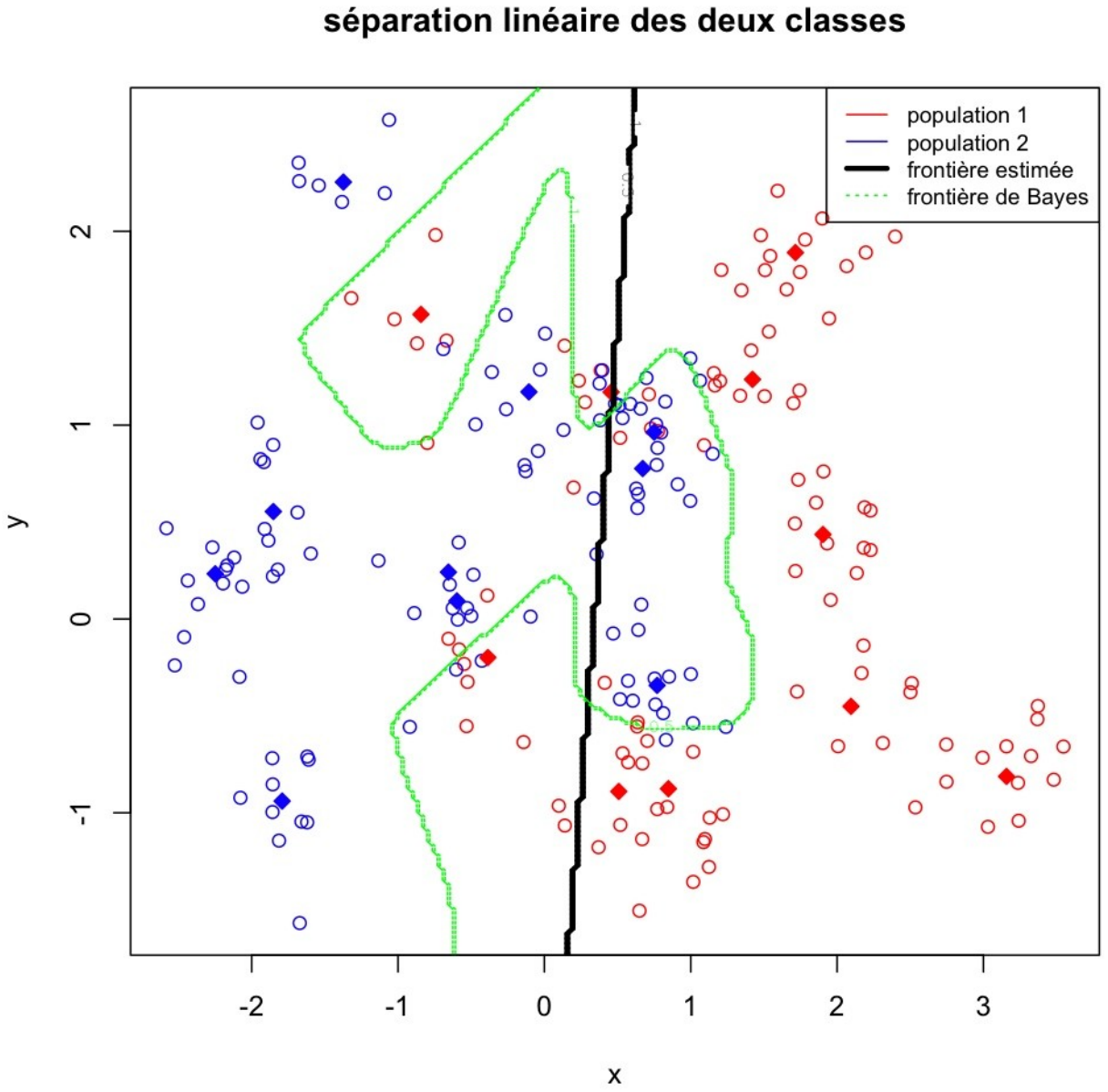
Coefficients estimés à partir de **X** et **G**

$$\hat{G}(X) = \begin{cases} 1 & \text{si } \hat{Y}(x) > ۰.۵ \\ 0 & \text{si } \hat{Y}(x) < ۰.۵ \\ ? & \text{si } \hat{Y}(x) = ۰.۵ \end{cases}$$

Modèle linéaire de degré 1

frontière linéaire

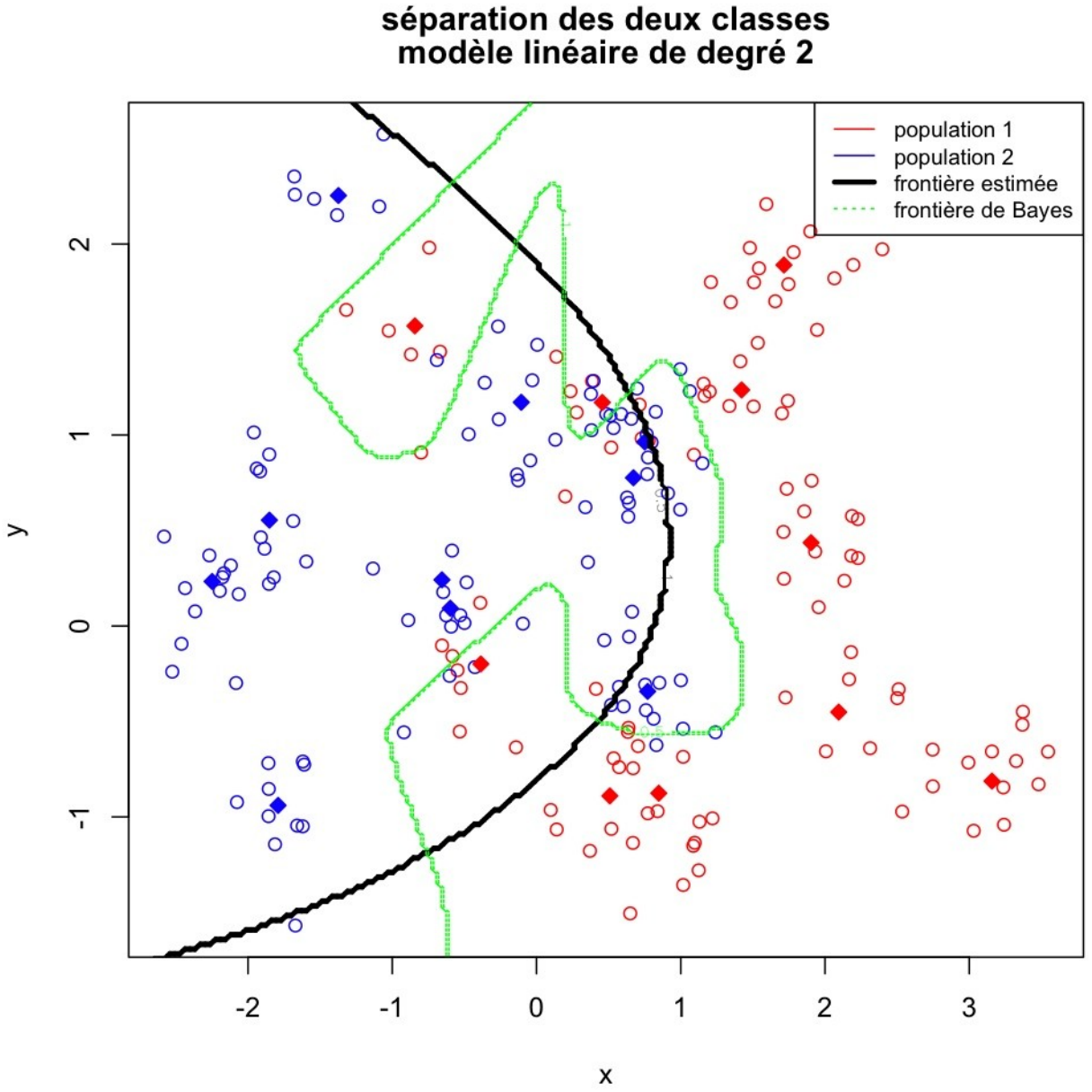
73,5% de points bien classés



Modèle linéaire de degré 2

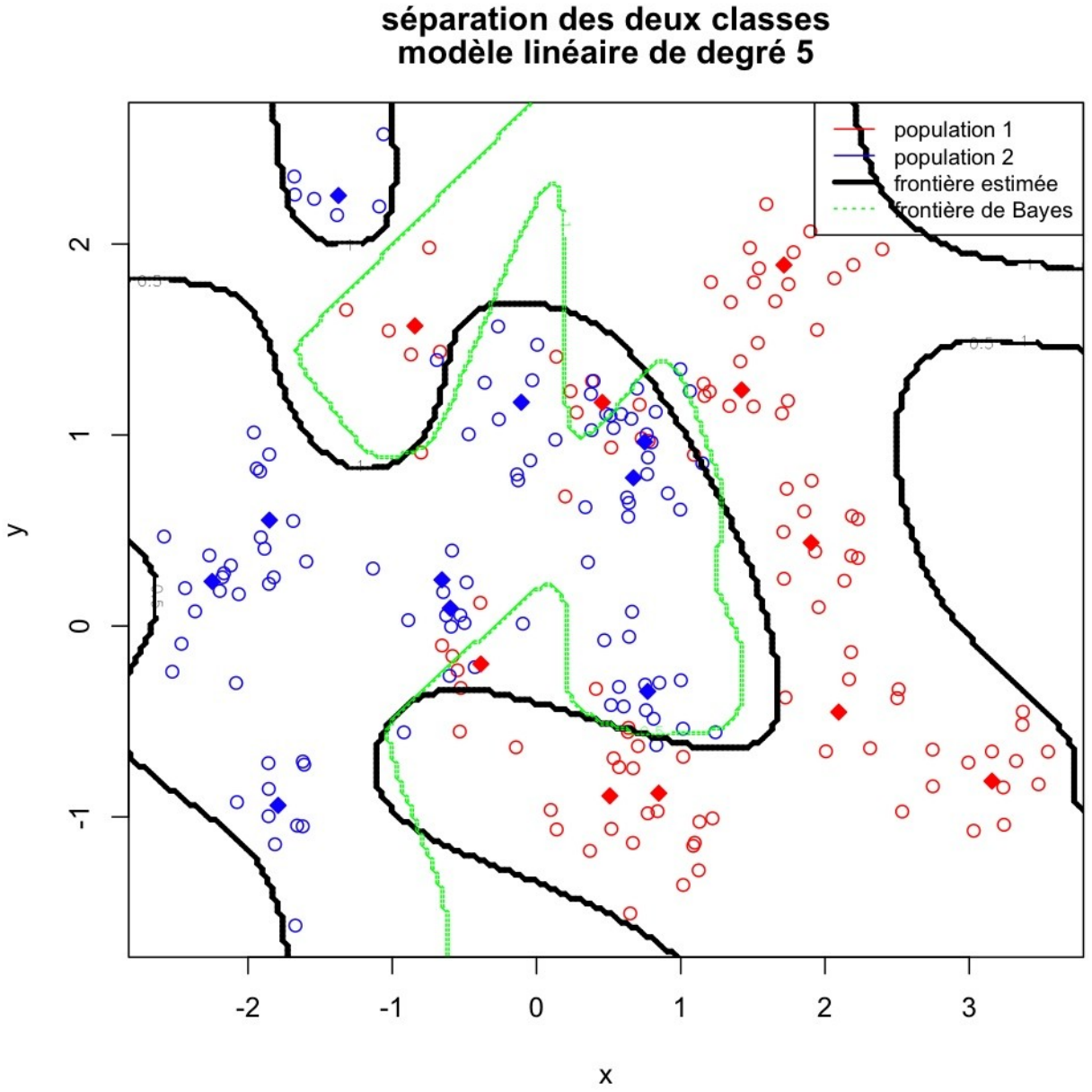
frontière = conique

79,5 % de bien classés



Modèle linéaire de degré 5

88% de bien classés



Méthodes kNN

kNN = k-th Nearest Neighbor

$$Soit\, x \in \mathbb{R}^2$$

$N_k(x)$ est l'ensemble des k x_i de **X** les plus proches de x

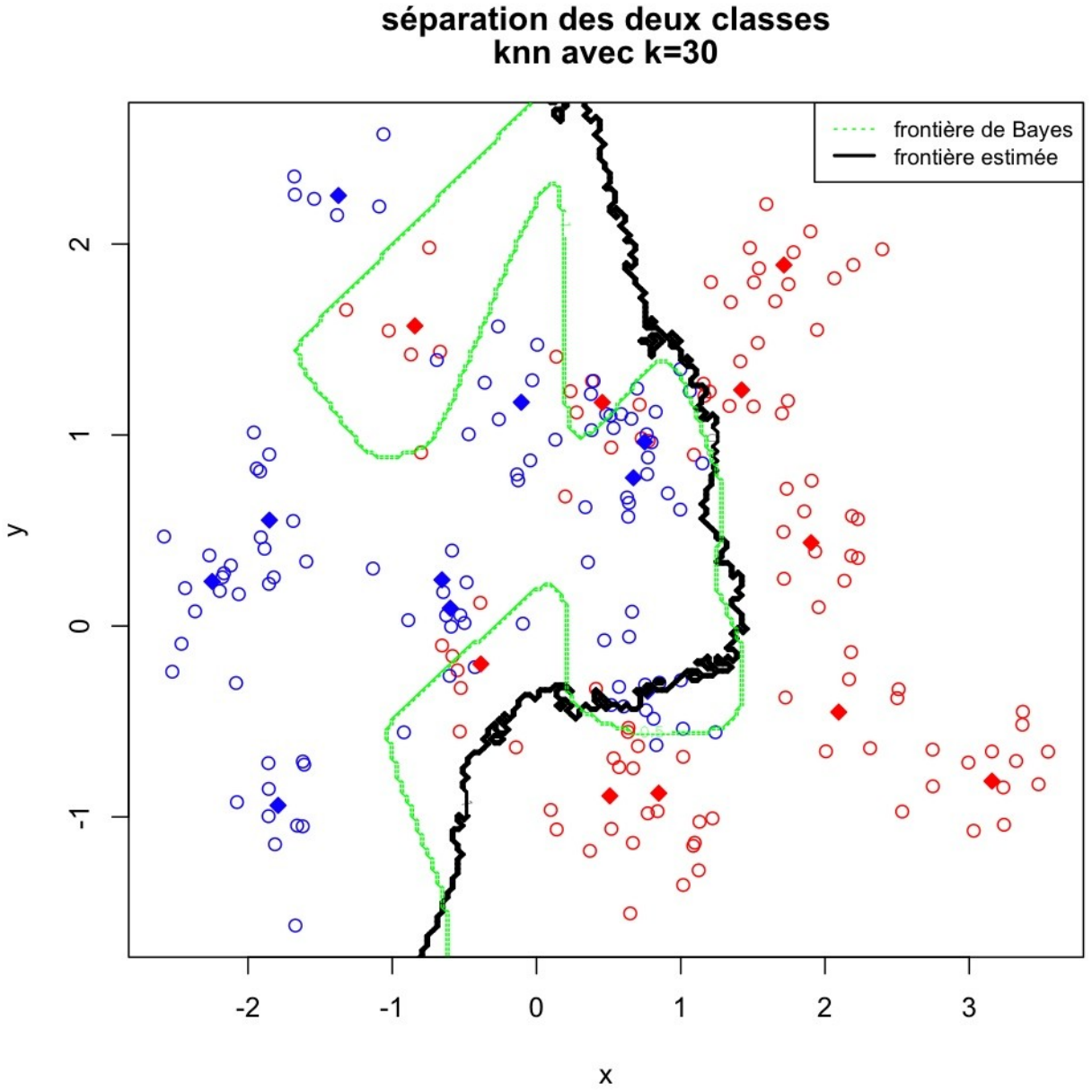
$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} g_i$$

$$\hat{G}(X) = \begin{cases} \backslash & si\ \hat{Y}(x) > 0.5 \\ \cdot & si\ \hat{Y}(x) < 0.5 \\ ? & si\ \hat{Y}(x) = 0.5 \end{cases}$$

Il s'agit ici d'une méthode de vote.

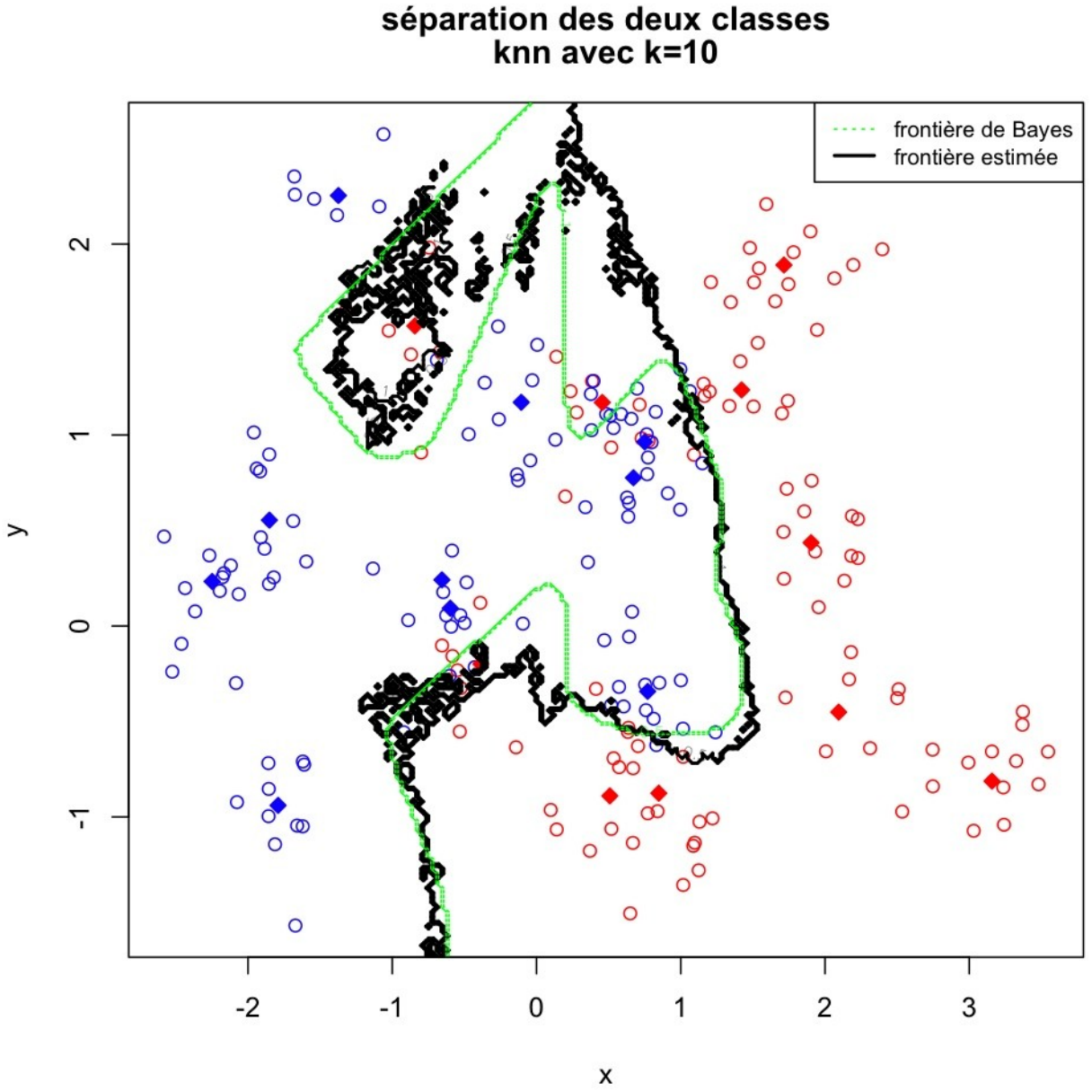
Modèle kNN pour k=30

84% de bien classés



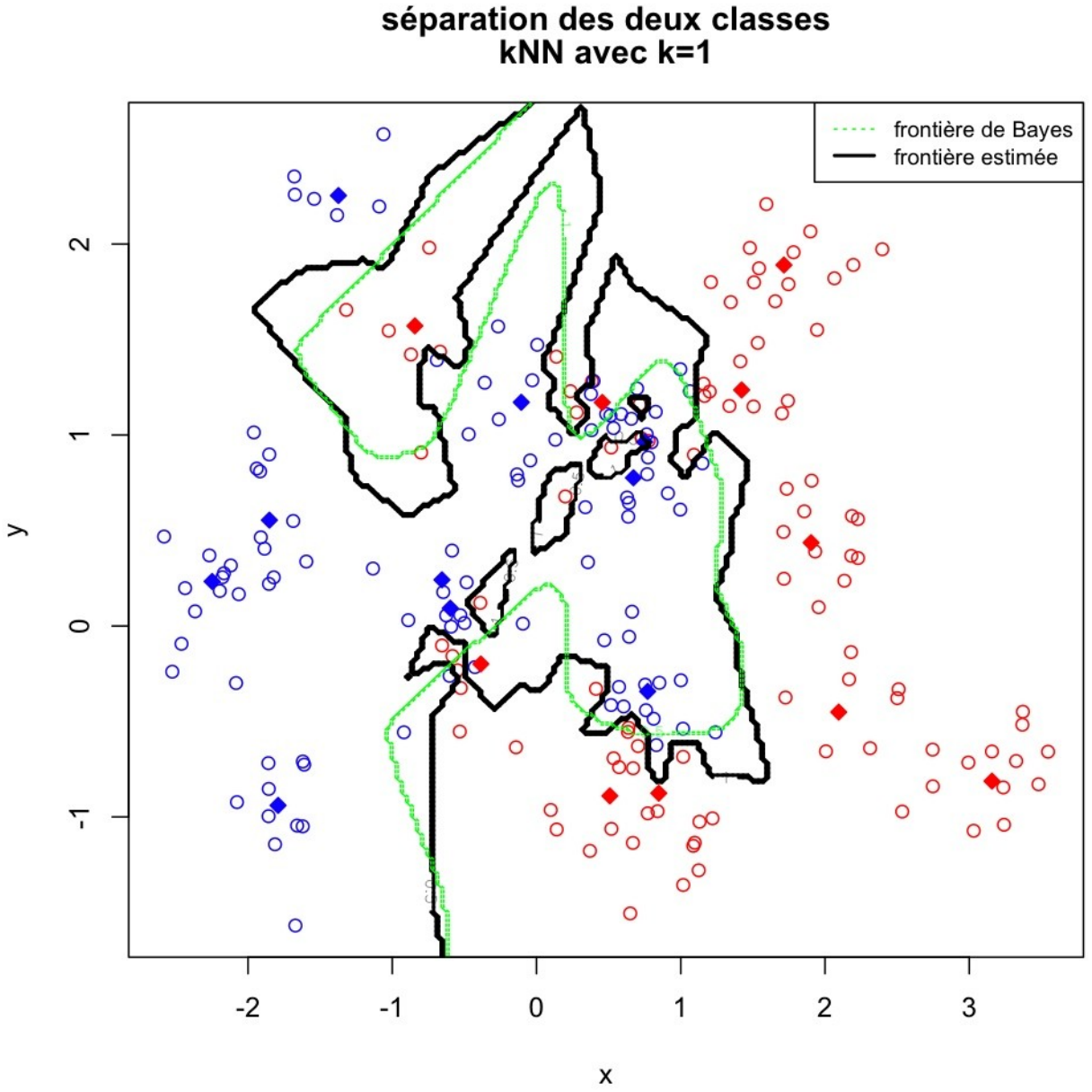
Modèle kNN pour k=10

88 % de bien classés



Modèle kNN pour k=1

100% de bien classés



Conclusion temporaire

- La méthode kNN s'approche davantage de la méthode optimale que la régression
- Paramètre à bien ajuster :
  - k pour kNN
  - degré du polynôme pour la régression
- 100% de bien classés n'est pas un bon objectif :

  voir validation

Ecart quadratique

Supposons Y(x) déterministe ; soit x un nouveau point :

$$EQ(x) = \mathbf{E}\left[\left(\hat{Y}(x) - Y(x)\right)^2\right]$$

$$EQ(x) = \left(\mathbf{E}(\hat{Y}(x)) - Y(x)\right)^2 + \mathbf{E}\left[\left(\hat{Y}(x) - \mathbf{E}(\hat{Y}(x))\right)^2\right]$$

Ecart = Biais^2 + Variance

Remarques :

- pour kNN, Biais ≈ 0
- pour un modèle linéaire, le biais est nul s'il n'y a pas d'erreur de modèle.

Malédiction de la dimension

- kNN semble plus performant car plus flexible
- Quand la dimension d du vecteur d augmente :

Exercice :

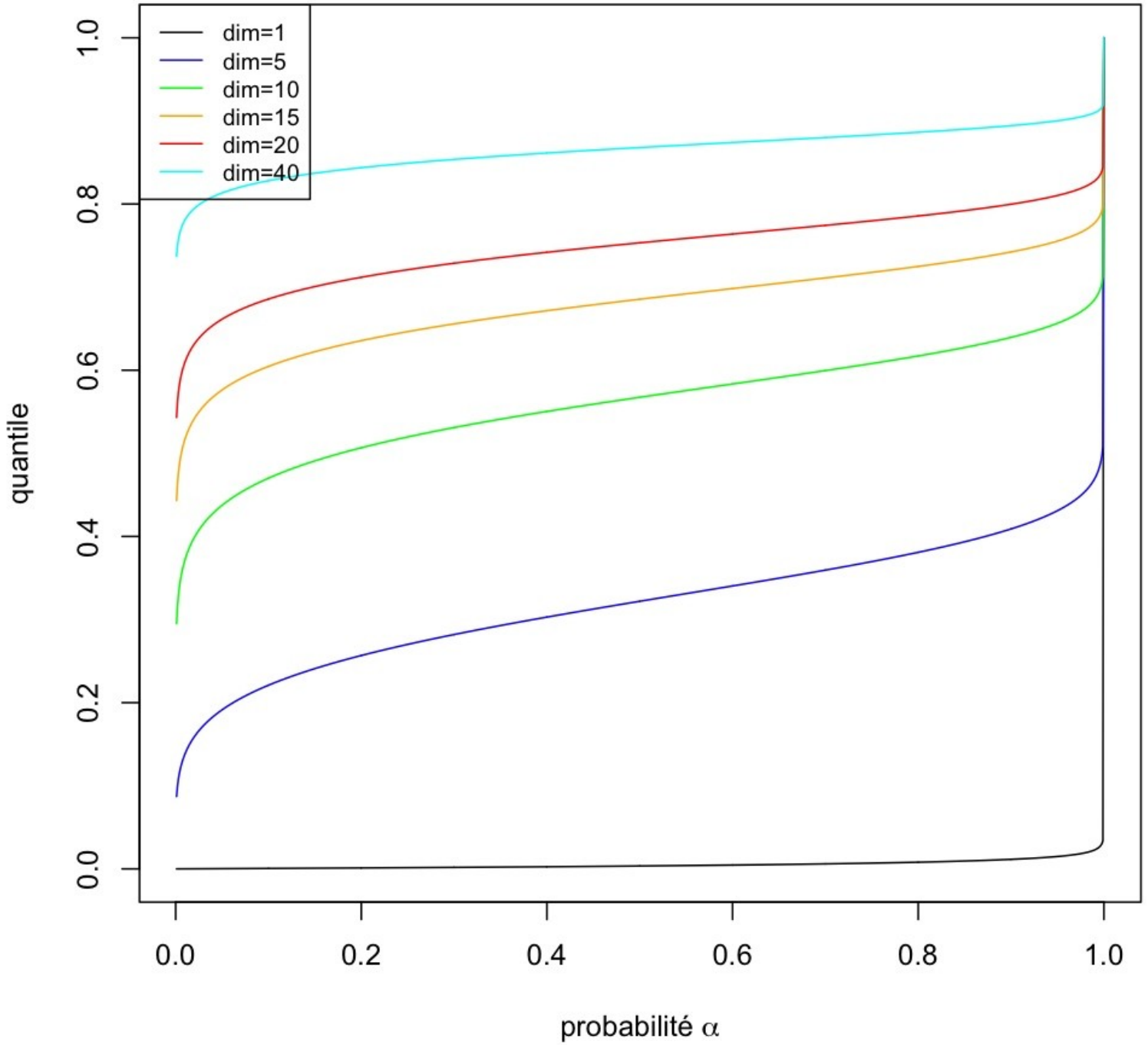
Soit $(X_i)_{1\leq i\leq n}$ des v.a.i.i.d. de loi uniforme sur $[-1,1]^d$

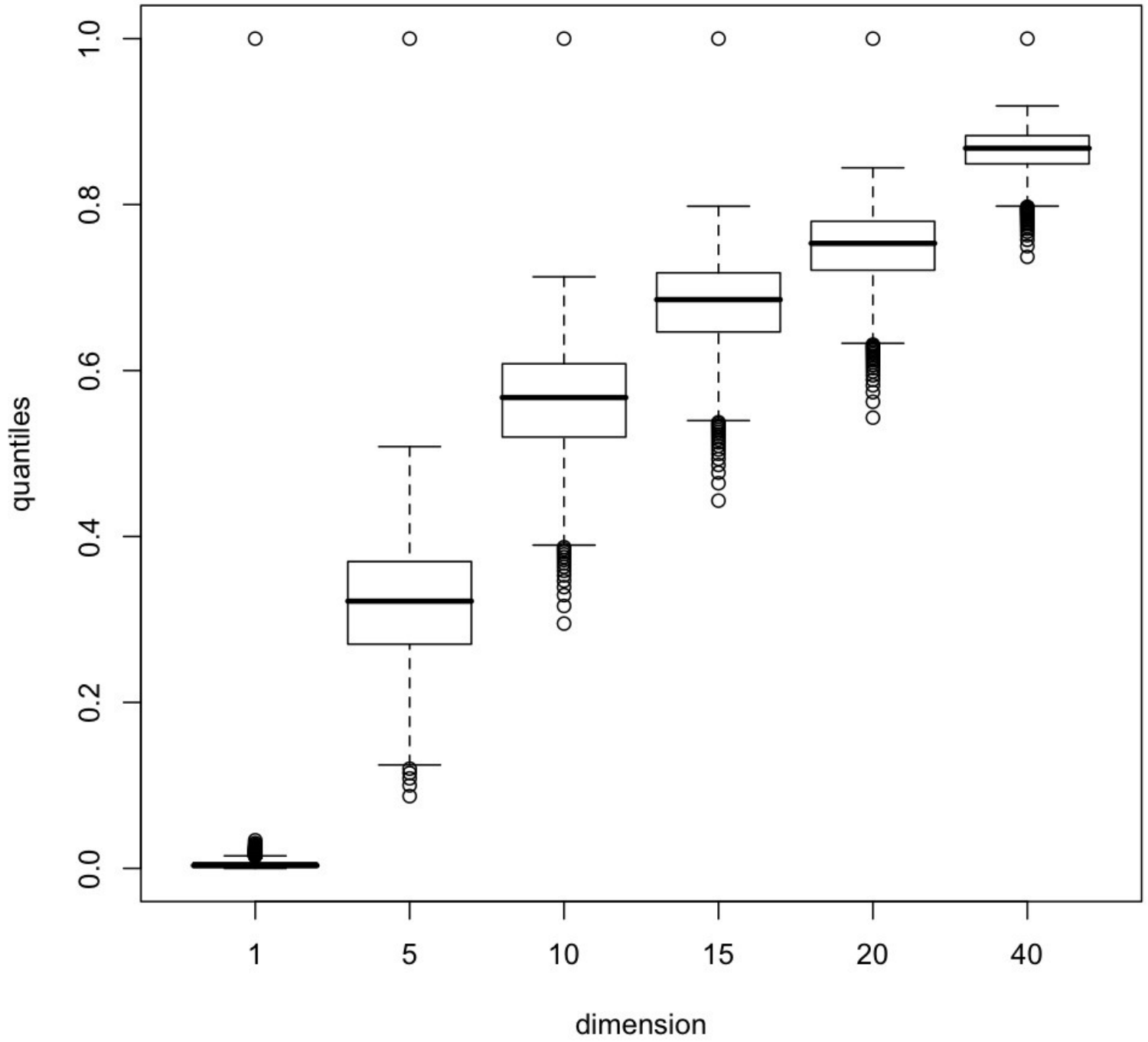
Soit $\| . \|$ la norme infinie, déterminer la loi de :

$$R = \text{Min} (\|X_i\|, 1\leq i\leq n)$$

Soit q_{10} le quantile à 10% de R, déterminer son comportement lorsque d augmente.

Conclusion ?

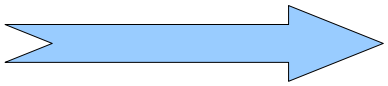

quantile du rayon du plus proche voisin en fonction de la dimension
échantillon de 200 points
dim=1
dim=5
dim=10
dim=15
dim=20
dim=40
quantile
0.0
0.2
0.4
0.6
0.8
1.0
probabilité α


boxplot de la loi du rayon du point
le plus proche de 0
quantiles
0.0
0.2
0.4
0.6
0.8
1.0
1
5
10
15
20
40
dimension

Observations et conclusion

- Les points se concentrent dans une couronne proche du bord du cube !
- Le point le plus proche est à distance supérieure à 0.6 en dimension 15.

  et il faut plusieurs points pour estimer correctement
- La moyenne locale opérée par kNN n'a plus aucun caractère local en grande dimension.



Méthode inapplicable en pratique

Validation

- Validation interne :
  - Déjà aperçue avec l'exemple
    - On ajuste un modèle à partir des données :

$$\hat{Y}(x) \text{ ou } \hat{G}(x)$$

    - On examine les écarts entre valeurs observées :

$$\text{écart entre } \hat{Y}(x_i) \text{ (ou } \hat{G}(x_i)) \text{ et } y_i \text{ (ou } g_i)$$

  - En classification
    - % de bien classés pour chaque classe
  - En régression :

$$\text{résidu} = y_i - \hat{Y}(x_i)$$

- Indicateur insuffisant

Validation externe

- Un modèle est fait pour prévoir
- Validation de l'apprentissage :
  - apprentissage sur des exemples (training set)
  - validation sur d'autres exemples (validation set)
- Plus généralement :
  - ensemble d'apprentissage : estimation des modèles considérés (ex : kNN pour k variant).
  - ensemble de validation : évaluation de la qualité de prévision des modèles ajustés pour choisir le « meilleur »
  - ensemble de test : pour valider le modèle choisi.

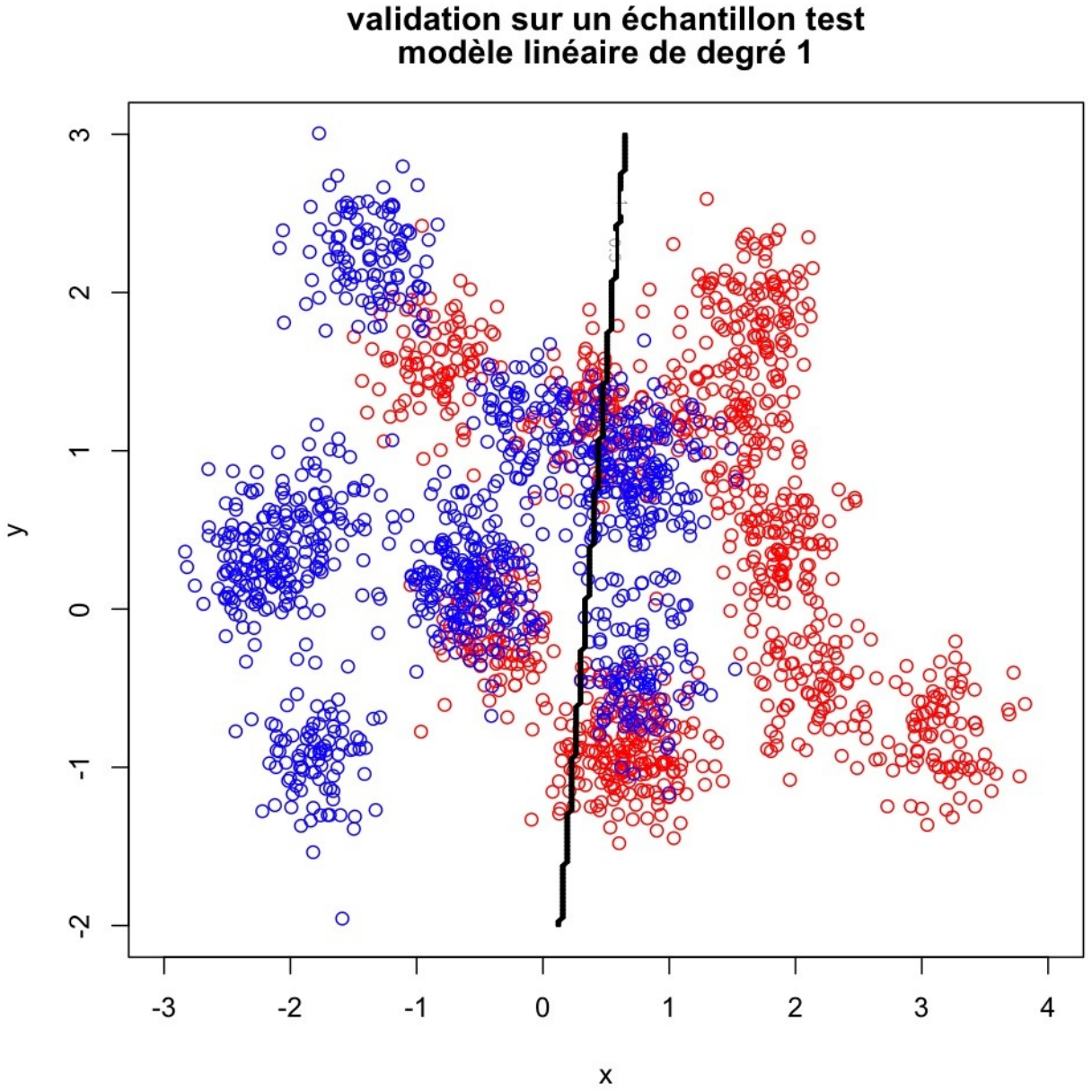
Retour sur l'exemple

72,8% de bien classés

rappel :

73,5% de bien classés

en apprentissage



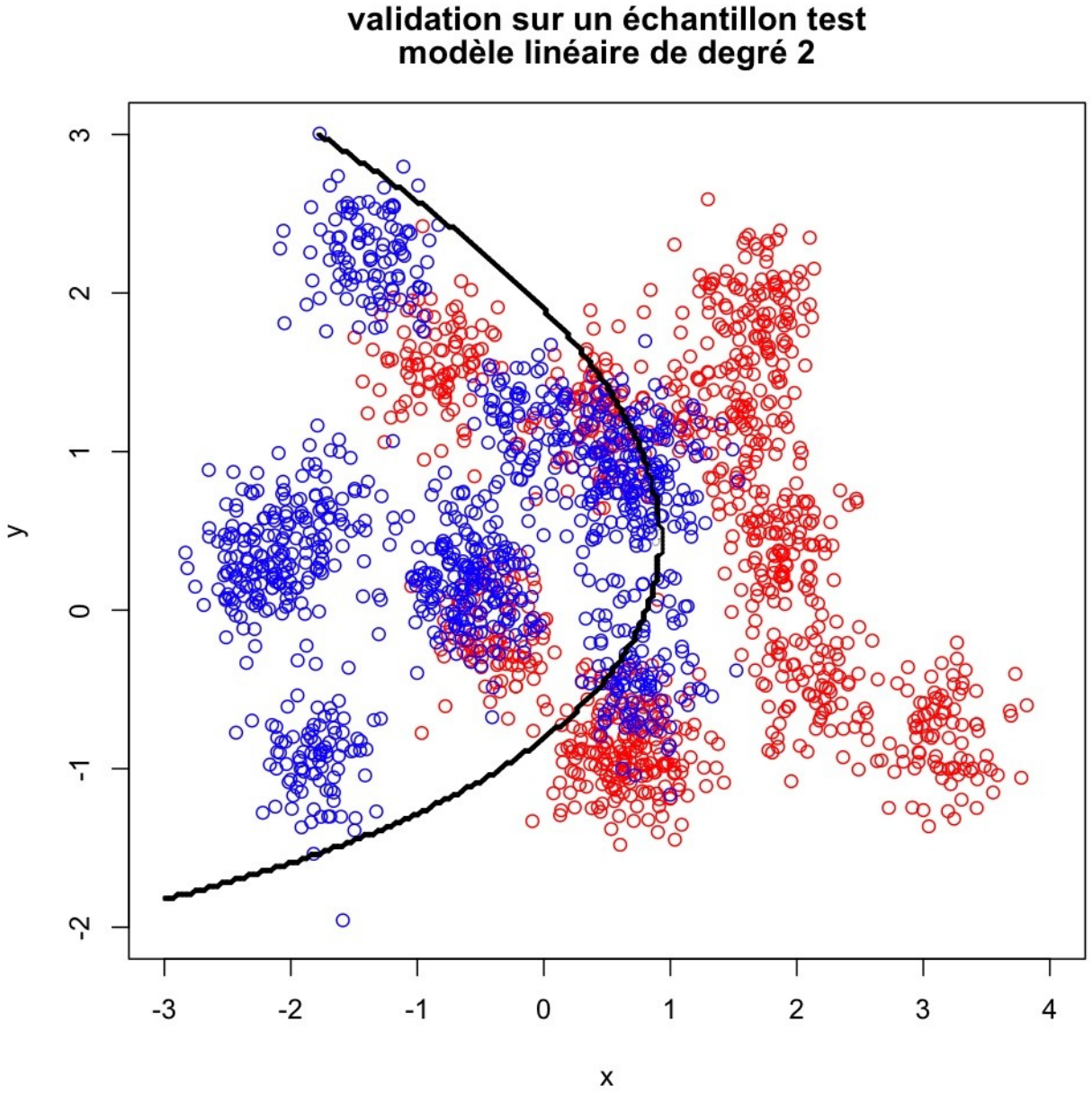
Retour sur l'exemple

77,5% de bien classés

rappel :

79,5% de bien classés

en apprentissage



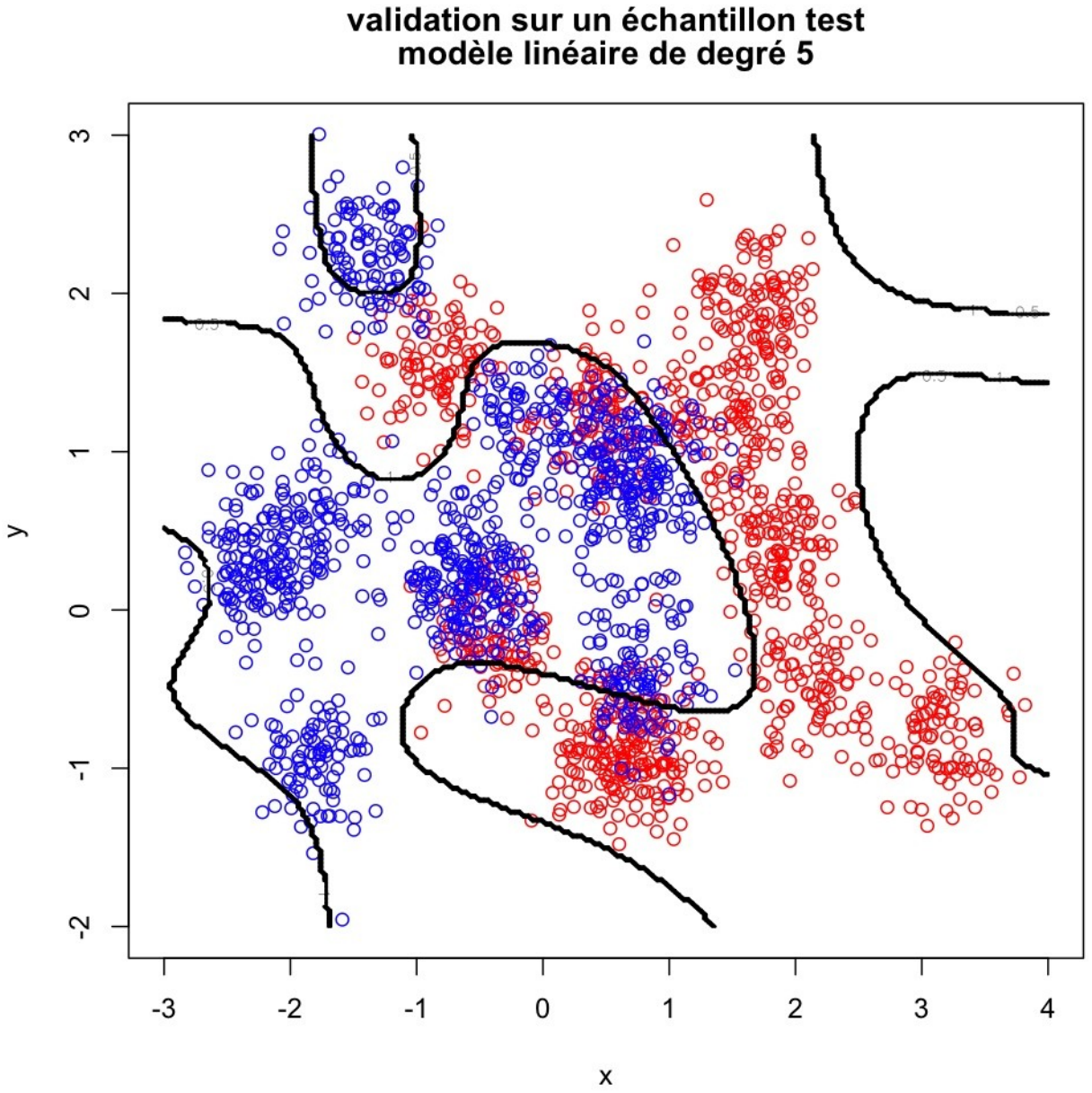
Retour sur l'exemple

84,5% de bien classés

rappel :

88% de bien classés

en apprentissage



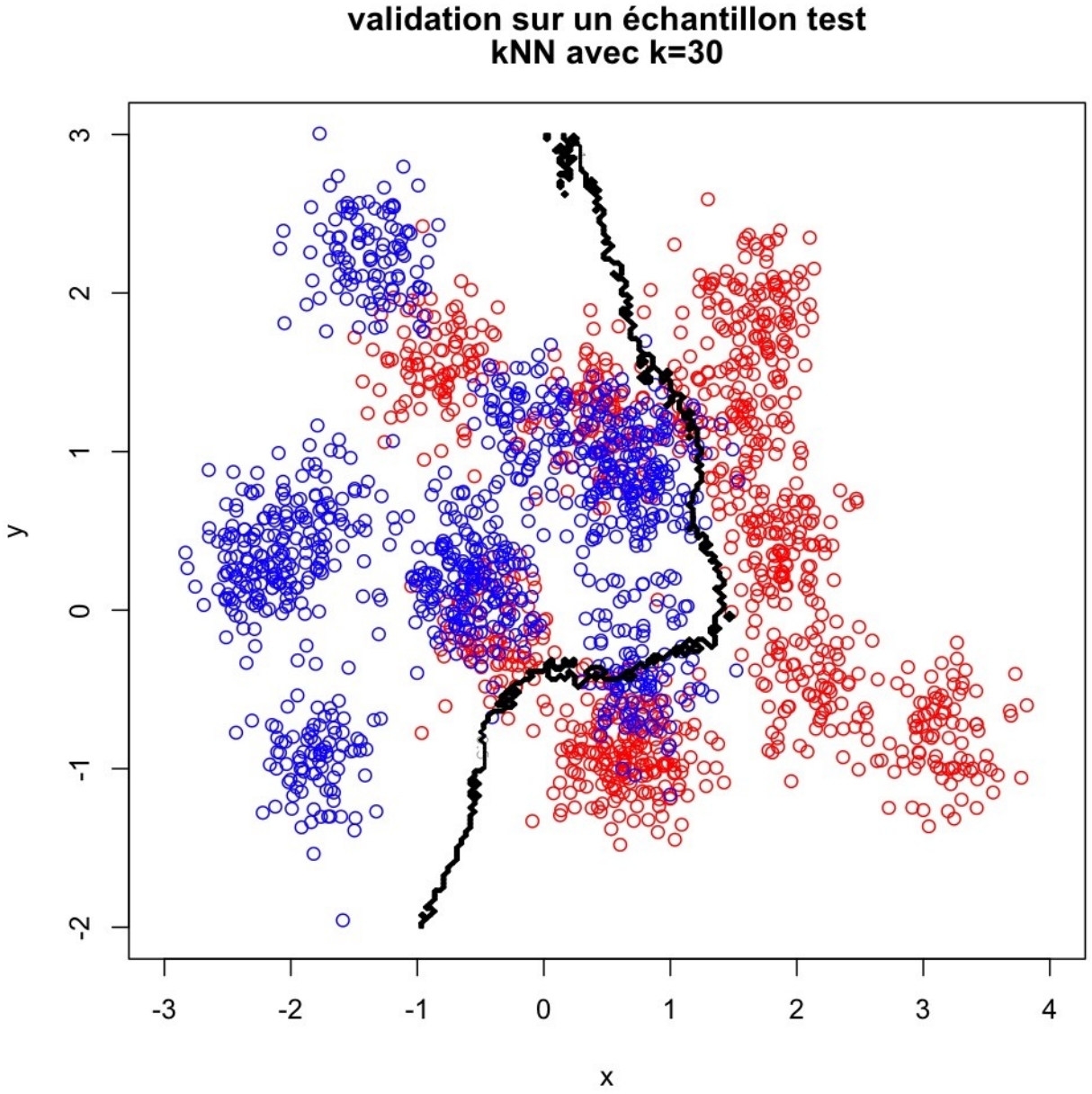
Retour sur l'exemple

80,2% de bien classés

rappel :

84% de bien classés

en apprentissage



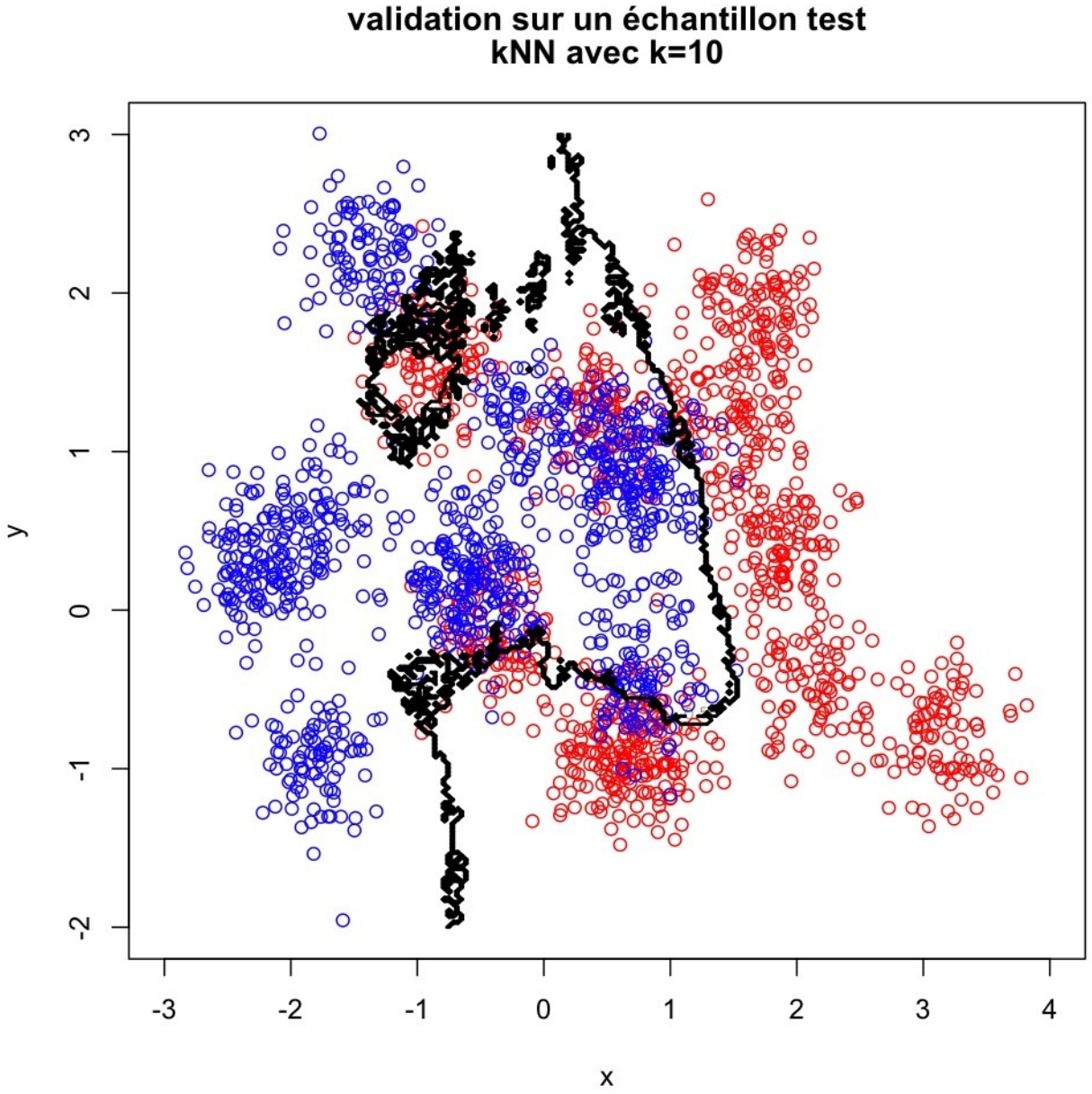
Retour sur l'exemple

84,9% de bien classés

rappel :

88% de bien classés

en apprentissage



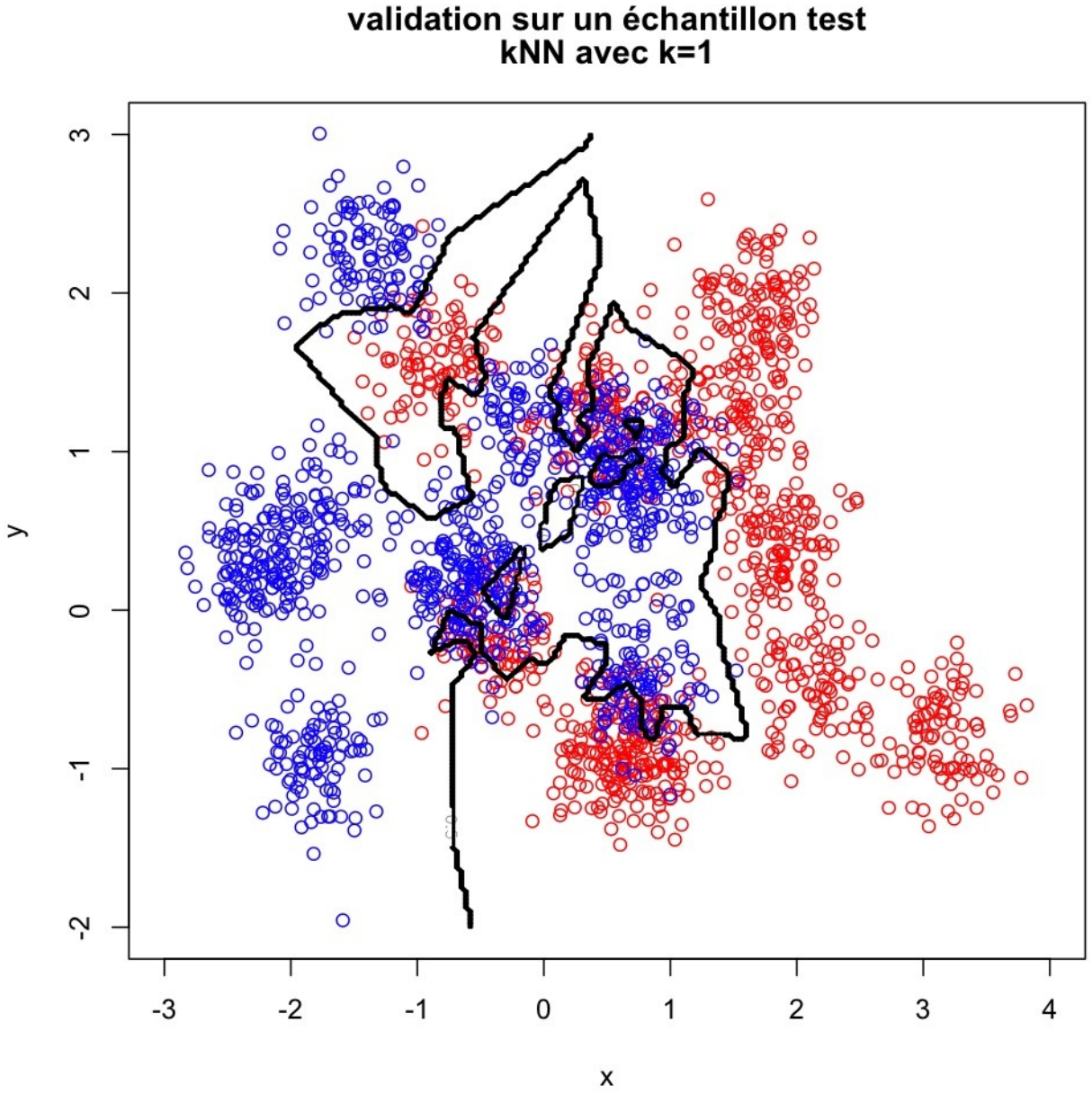
Retour sur l'exemple

82% de bien classés

rappel :

100% de bien classés

en apprentissage



Conclusions

- La différence entre écart en apprentissage et écart en validation augmente quand la complexité du modèle augmente.
- L'écart en validation n'augmente pas forcément quand la complexité augmente.
- Des modèles complexes prennent des décisions absurdes :
  - modèle linéaire de degré 5 (points pour x petit ou x grand)
  - kNN pour k=1 (îlots au centre)

La validation croisée

- Idéal :
  - ensemble d'apprentissage (≈50%)
  - ensemble de validation et de choix de modèle (≈25%)
  - ensemble de test (≈25%)
- Mais :
  - les données coûtent très cher
  - l'idéal consiste mène à n'utiliser que la moitié des données pour construire le modèle !!

Idée de faire bouger les sous-ensembles...

Cross-validation

- Découper les données en K parties de tailles approximativement égales : $A_1,...,A_K$
- Pour k=1 to K, faire :
  - apprendre un modèle sur $\bigcup_{j \neq k} A_j$, noté $\hat{Y}^{-k}(x)$
  - valider le modèle sur A_k
  - évaluer le critère $L_k = \frac{1}{n/K} \sum_{i \in A_k} L(y_i, \hat{Y}^{-k}(x_i))$
- Evaluer alors :

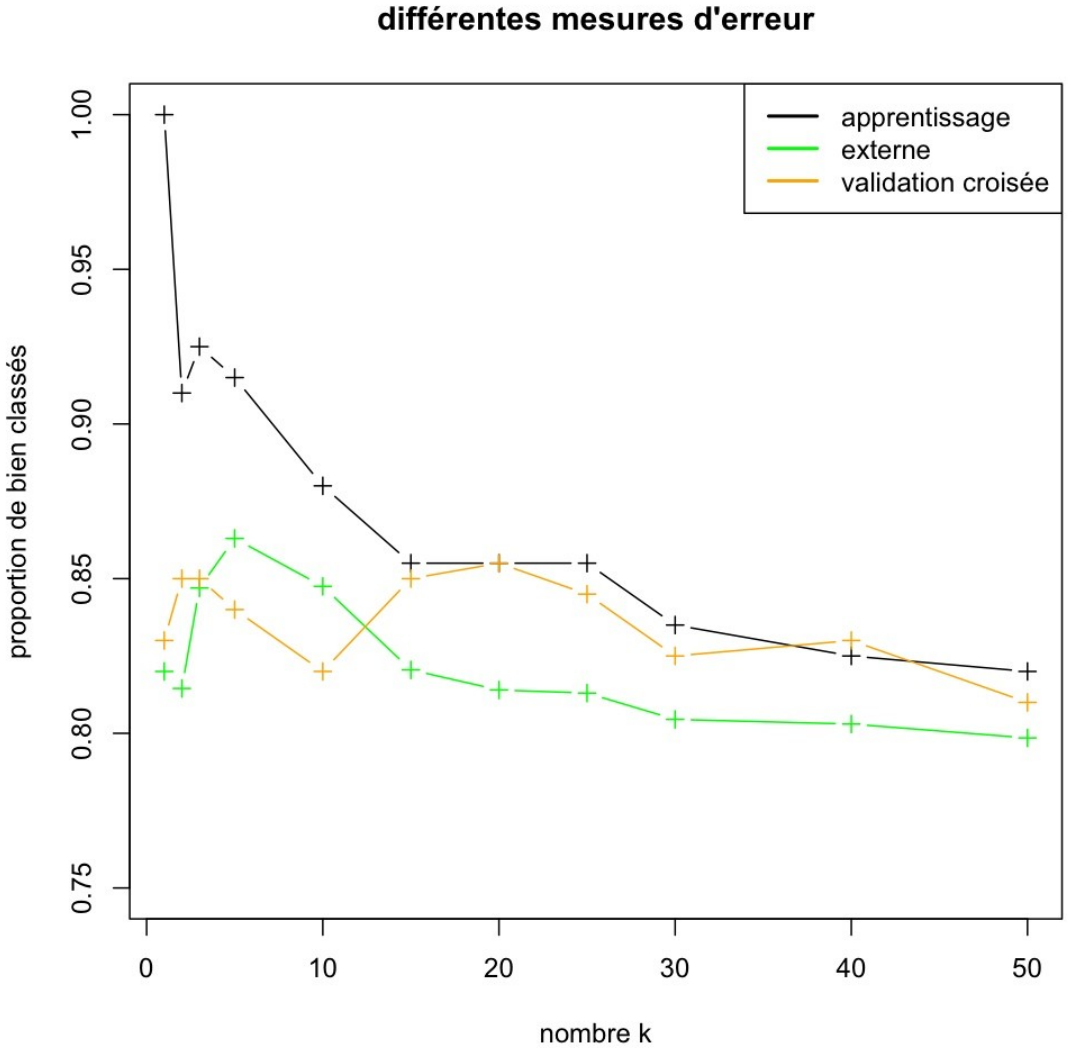
$$CV = \frac{1}{K} \sum_{k=1}^{K} L_k = \frac{1}{n} \sum_{i=1}^{n} L(y_i, \hat{Y}^{-k(i)}(x_i))$$

où k(i) désigne le numéro de la partie auquel appartient (x_i, y_i)

Utilisation

- Sous hypothèses à préciser, CV s'approche de EPL
- En pratique :
  - K = 5 à 10
  - ou K = n : leave-one-out
- Choix de modèles basé sur CV :
  - Exemple pour kNN, choix de k minimisant CV
  - En régression linéaire, on verra le lien avec les résidus studentisés.

Retour sur l'exemple - kNN



k	1	2	3	5	10	15	20	25	30	40	50
apprentissage	1,00	0,91	0,93	0,92	0,88	0,86	0,86	0,86	0,85	0,84	0,82
validation externe	0,82	0,82	0,85	0,86	0,85	0,82	0,81	0,81	0,81	0,80	0,80
validation croisée	0,83	0,85	0,85	0,84	0,82	0,85	0,86	0,85	0,83	0,83	0,81

Le bootstrap

- Méthode générale en statistiques
- Exemple élémentaire : intervalle de confiance
  - Soit $(x_1,...,x_n)$ un n-échantillon de loi normale N(m,1)
  - Problème : intervalle de confiance pour m basé sur l'échantillon
  - Ici, calcul théorique :

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \text{ est de loi } N\left(m, \frac{1}{n}\right)$$

$$\left[\overline{X} - \frac{1.96}{\sqrt{n}}, \overline{X} + \frac{1.96}{\sqrt{n}}\right]$$ est un intervalle de confiance

à 95% pour m

Cas général

- Un intervalle de confiance de niveau doit vérifier :

$$P_{\theta_0}\left(\theta_0 \in [l(X_1,\ldots,X_n), u(X_1,\ldots,X_n)]\right) = 1-\alpha$$

Pour l'exemple qui précède, on trouve un intervalle valable pour tout m= et non seulement pour le « bon » θ_0

- Bootstrap :

Remplacer P_{θ_0} par une estimation notée P_n^*

choix le plus courant $P_n^* = \frac{1}{n}\sum_{i=1}^{n} \delta_{X_i}$: loi empirique

Algorithme

Pour b=1 à B, faire :

simuler un échantillon de taille n parmi $\{x_1,...,x_n\}$, noté X^{*b}

stocker l'estimation de obtenue, notée ^{*b}

Fin faire

Evaluer les quantiles d'ordre /2 et 1- /2 de l'échantillon des ^{*b}, b=1,...,B

$$IC_{bootstrap} = [q_{\alpha/2}, q_{1-\alpha/2}]$$

Résultats : voir simulations R

Le bagging

- Bagging = Boostraping and Aggregating

- Idée :
  - rééchantillonner parmi les (x_i, y_i)
  - construire un modèle sur cet échantillon
  - moyenner les modèles obtenus

Algorithme

- Soit Z = (X,Y) ou (X,G) l'ensemble d'apprentissage

Pour b=1 à B, faire :

simuler un échantillon de taille n parmi $\{z_1,...,z_n\}$, noté Z^{*b}

estimer un modèle à partir de Z^{*b}

stocker le modèle obtenu, noté $\hat{Y}^{*b}$

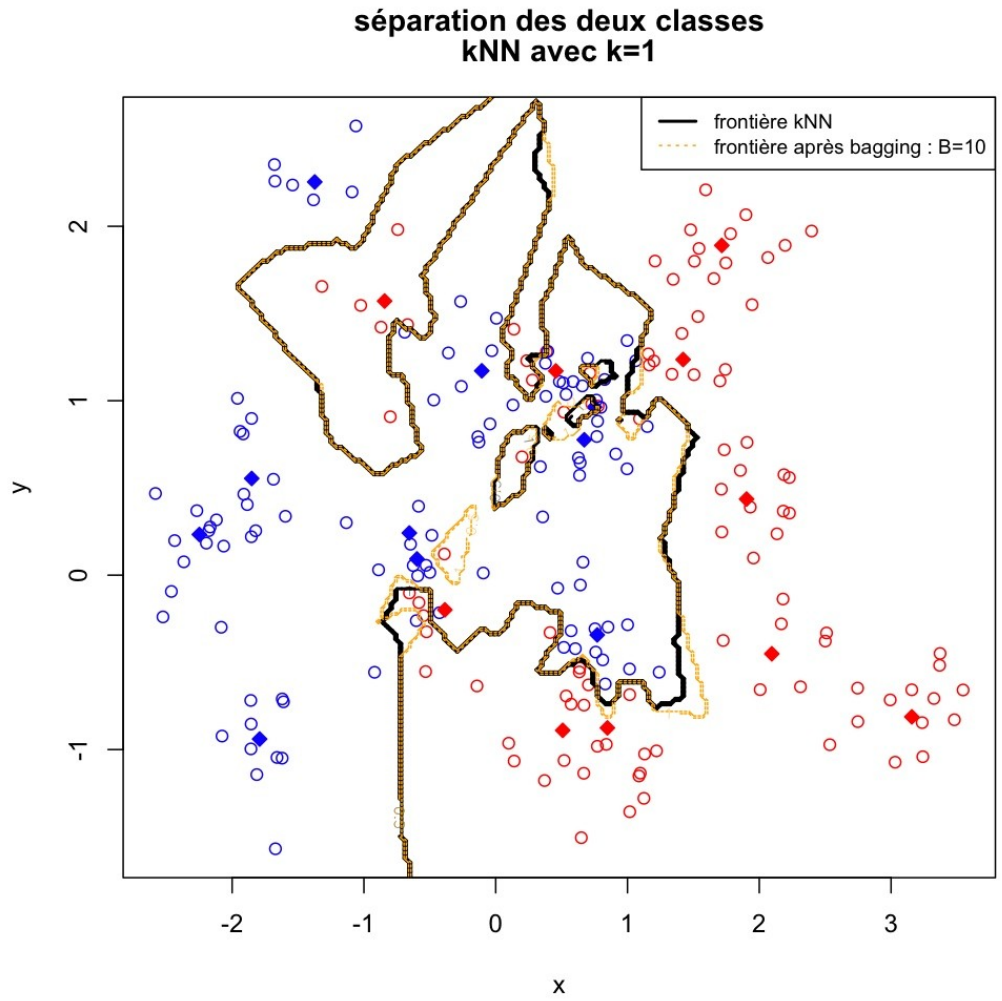
Fin faire

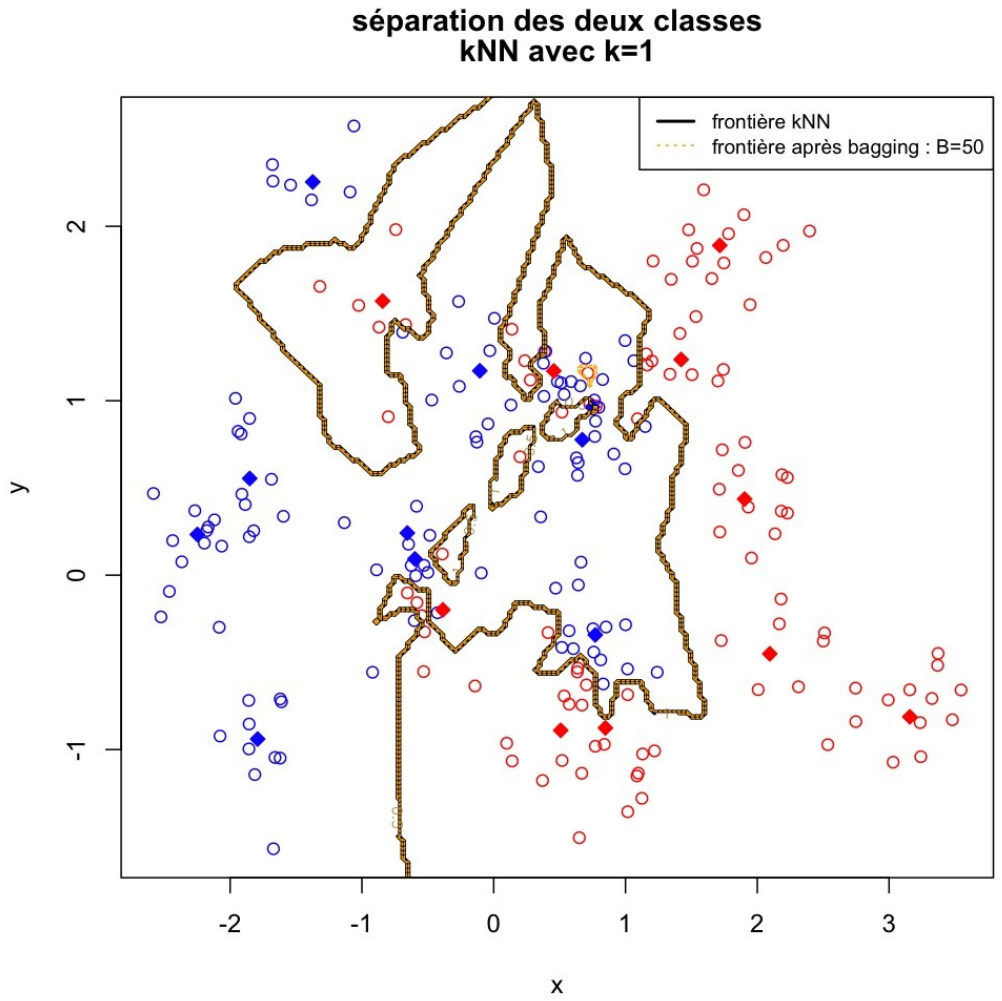
Poser :

$$\hat{Y}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{Y}^{*b}(x)$$

 inutile pour un modèle linéaire en les réponses y_i

Exemple avec kNN





Voir plus loin : CART, NN...