Alternatives au modèle polynomial

Classification et régression

Astrid Jourdan

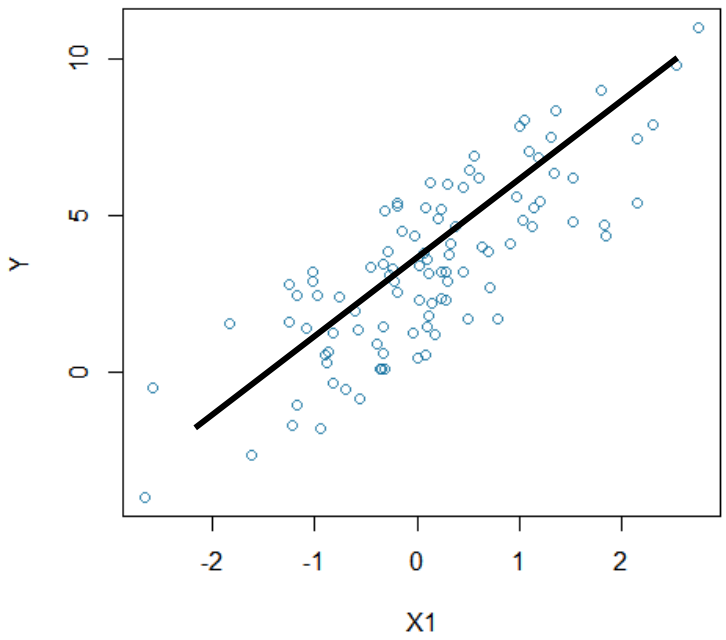
CYTech – CY Cergy Paris université

Campus de Pau

Workshop Plans d'expériences
APEX XV – Aix en Provence – 4-6 octobre 2023

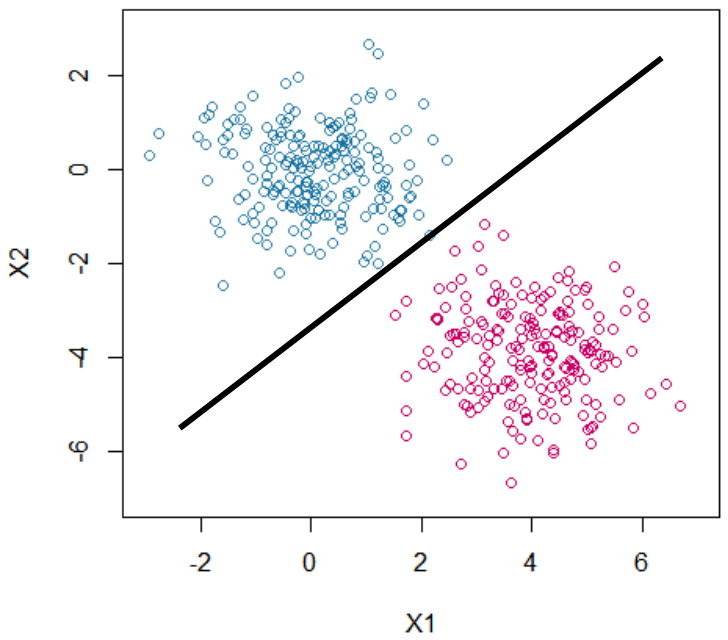
Modèle polynomial

Régression linéaire polynomiale



Variable cible quantitative :
Méthodes de régression

Régression logistique



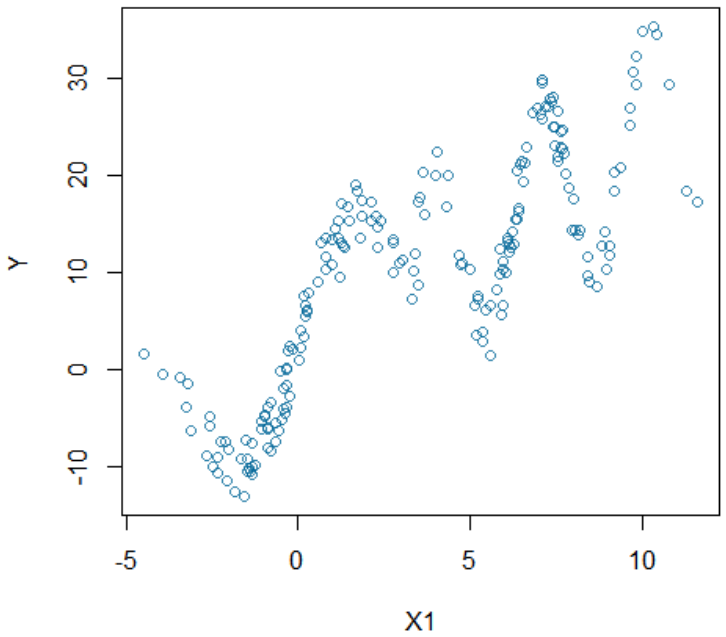
Variable cible catégorielle :
Méthodes de classification

Dans les deux cas, il s'agit de déterminer une droite (hyperplan)

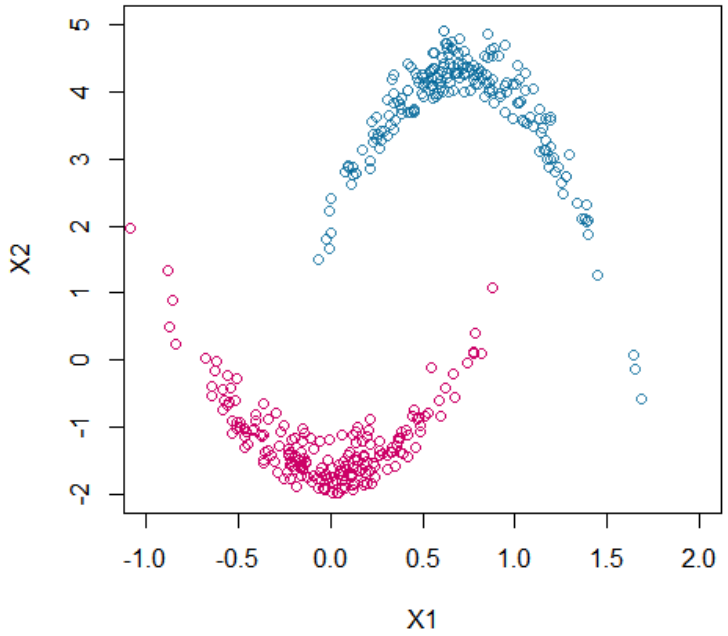
$$f(X_1, \dots, X_d) = a_0 + a_1 X_1 + \cdots + a_d X_d$$

Modèle polynomial

Variable cible quantitative :
Méthodes de régression



Variable cible catégorielle :
Méthodes de classification



Quelles sont les alternatives quand les données nécessitent un modèle plus complexe?

Sommaire

Régression (variable cible quantitative)

- Modèle polynomial : la régression linéaire polynomiale et plans d'expériences
- Alternative : Krigeage et space-filling designs

Classification (variable cible catégorielle)

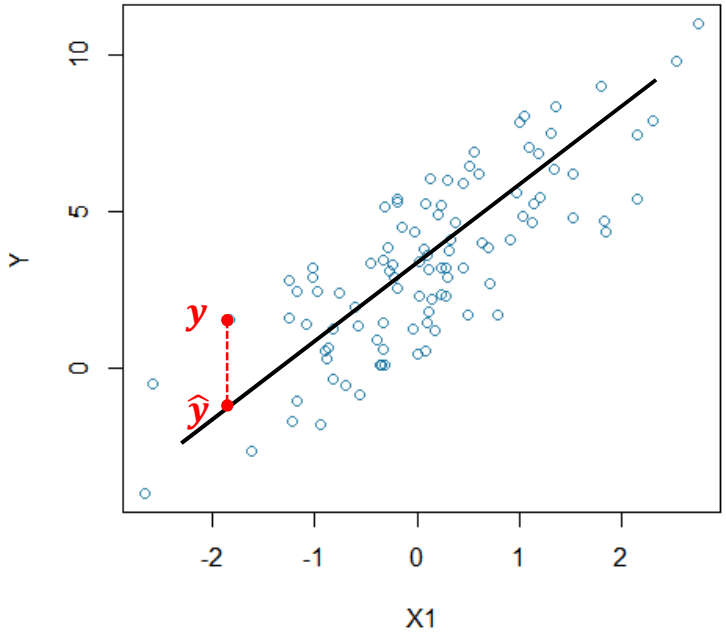
- Généralités sur la classification
- Modèle polynomial : la régression logistique
- Alternative : Arbres de décision et forêts aléatoires

Et les réseaux de neurones…

Régression (variable cible quantitative)

- Rappels sur la régression polynomiale
- Krigeage
- Space-filling designs

Rappel sur la régression polynomiale



La régression linéaire polynomiale :

$$y = a_0 + a_1 X_1 + \cdots + a_d X_d + \varepsilon(x)$$

où $\varepsilon(x)$ erreurs indépendantes N(0,σ^2).

Toute l'information est contenue dans la partie polynomiale.

Pour déterminer les coefficients a_i , on minimise le carré de l'écart entre l'observation et la valeur du modèle (erreur quadratique moyenne) :

$$\sum_i (y_i - \hat{y}_i)^2$$

Il existe une solution analytique à ce problème de minimisation. Pas besoin d'un algorithme!

Rappel sur la régression polynomiale

Estimation des coefficients

X_1	X_2		X_d	Y
x_{11}	x_{12}	…	x_{1d}	Y_1
x_{21}	x_{22}	…	x_{2d}	Y_2
…	…	…	…	…
x_{n1}	x_{n2}	…	x_{nd}	Y_n

Dataset

Le modèle:

$$y = a_0 + a_1 X_1 + \cdots + a_d X_d$$
$$+ a_{12} X_1 X_2 + \cdots + \varepsilon(x)$$

Estimation des coefficients : Solution exacte et très peu couteuse en temps de calcul

1	X_1	…	X_d	X_1X_2	…
1	x_{11}	…	x_{1p}	$x_{11}x_{12}$	…
1	x_{21}	…	x_{2p}	$x_{21}x_{22}$	…
…	…	…	…	…	…
1	x_{n1}	…	x_{np}	$x_{n1}x_{n2}$	…

a
a_1
…
a_d
a_{12}
…

X : Matrice du modèle

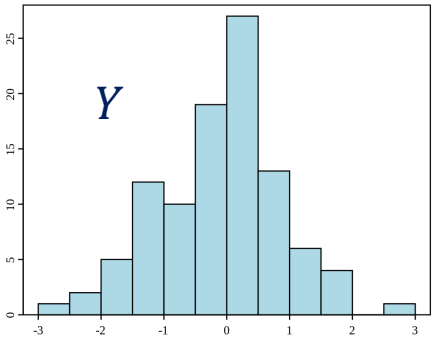
a : Vecteur des coefficients

Les coefficients :

$$\hat{a} = (X^T X)^{-1} X^T Y$$

Rappel sur la régression polynomiale

Analyse du modèle



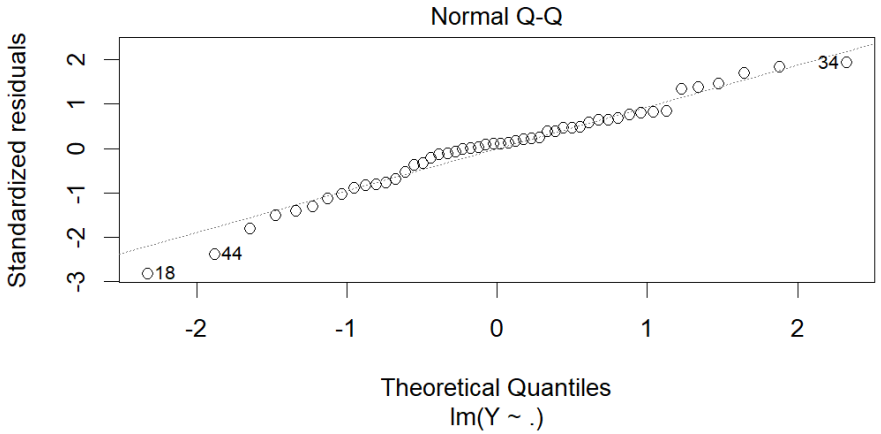
- R^2
- Test de Student
- Outliers

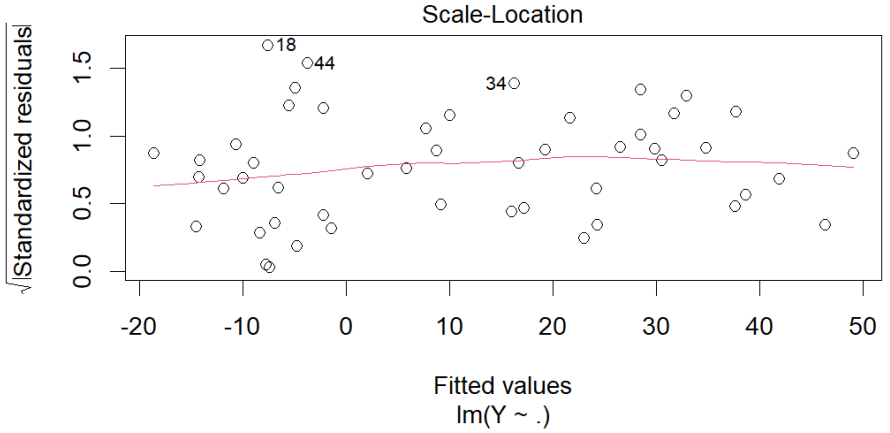
```
Call:
lm(formula = Y ~ ., data = Mydata)

Residuals:
    Min      1Q  Median      3Q     Max
-25.376  -5.997   1.005   5.849  18.087

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  37.1342    21.6761   1.713   0.0936 .
X1            2.3331     3.6004   0.648   0.5203
X2           -0.2620     1.5477  -0.169   0.8663
X3           -1.4002     0.5621  -2.491   0.0165 *
X4           -3.1610     0.2444 -12.936   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9.743 on 45 degrees of freedom
Multiple R-squared:  0.8083,    Adjusted R-squared:  0.7913
F-statistic: 47.44 on 4 and 45 DF,  p-value: 1.386e-15
```





Rappel sur la régression polynomiale

Stratégie de planification d'expériences

L'objectif de la planification des expériences est d'obtenir un maximum d'information (pour construire un modèle précis) en un minimum d'expériences (problème de la dimension).

Le fait d'avoir une expression analytique de l'estimation des coefficients permet de prévoir un plan d'acquisition des données optimal

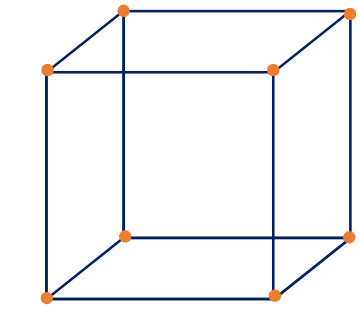
$$\hat{a} = (X^T X)^{-1} X^T Y \qquad \text{et} \qquad \hat{\sigma}^2 = \frac{\|y - X\hat{a}\|^2}{n-p-1}$$

- Plans orthogonaux

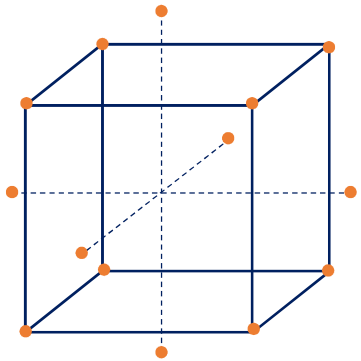
Fraction de toutes les combinaisons des niveaux des variables, choisie afin de rendre indépendants les termes du modèle (chaque terme contribue aux variations de la réponse sans être perturbé par les autres)

- Plans optimaux

Ensemble de points construit pour optimiser la précision du modèle (en général, minimiser la variance de prévision)



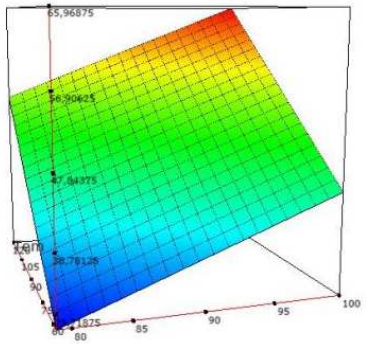
Plans factoriels

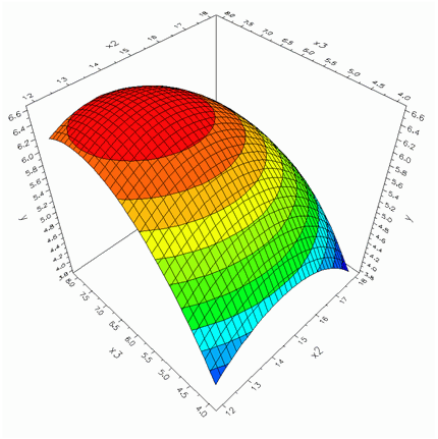


Plans composites

Première alternative: le krigeage

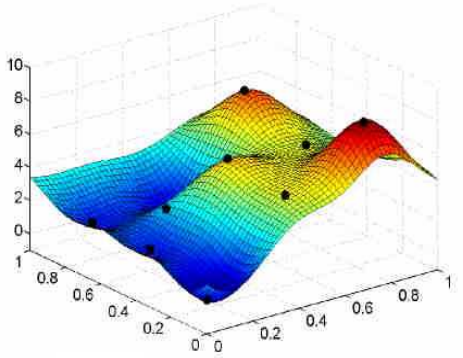
Le modèle polynomial permet d'obtenir des surfaces de réponse simple.





Le krigeage va permettre de représenter des formes plus irrégulières.

$$Y(x) = \underbrace{X(x)a}_{\text{Partie polynomiale}} + \underbrace{\Gamma(x)}_{\text{Processus aléatoire}}$$



On remplace les erreurs gaussiennes indépendantes par un processus aléatoire centré mais avec une structure de covariance qui dépend de la distance entre les points,

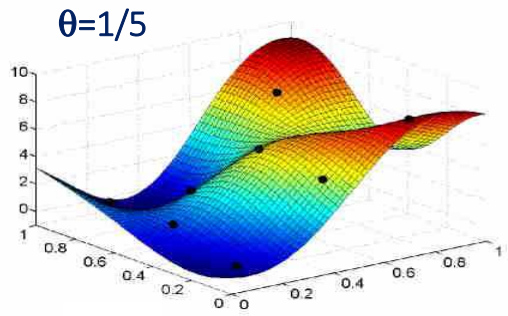
$$cov[\Gamma(x), \Gamma(u)] = \sigma^2 R_\theta(x - u)$$

où R_θ est une fonction de corrélation et θ est un vecteur de paramètres inconnus (et donc à estimer).

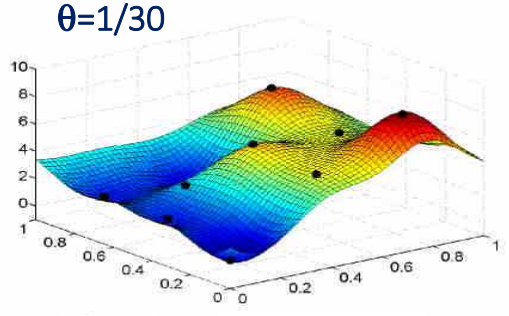
Krigeage : Fonction de corrélation

Exemple de la fonction de correlation gaussienne: $cov[\Gamma(x), \Gamma(u)] = \sigma^2 \exp(-\theta \|x - u\|^2)$

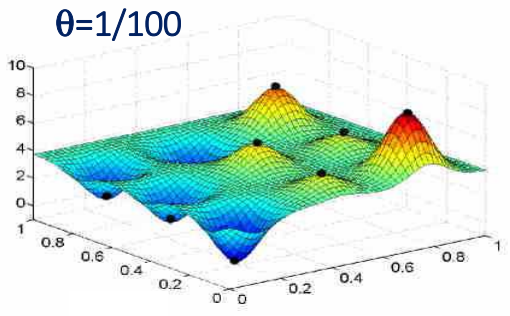
θ=1/5



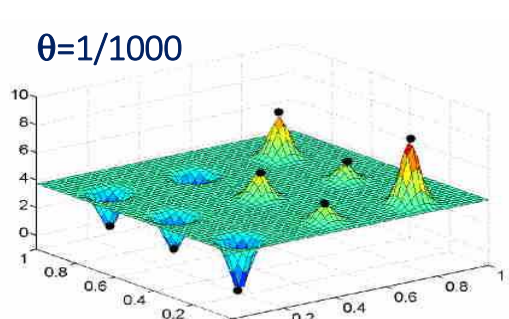
θ=1/30



θ=1/100

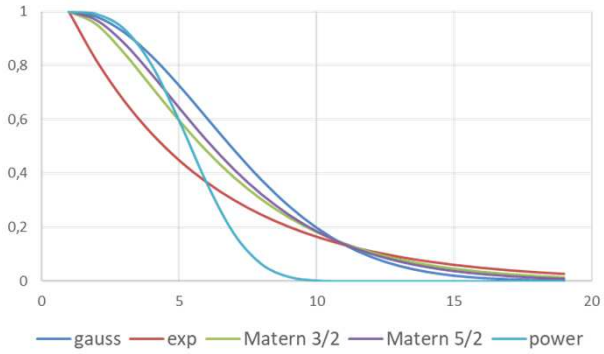


θ=1/1000



- La distance de correlation augmente avec θ
- Plus θ est grand, plus la surface est lisse

- Il existe plusieurs fonctions de corrélation (Matern, exponentielle,…)
- Le paramètre peut être vectoriel de façon à prendre en compte des distances de corrélation différentes suivant la dimension

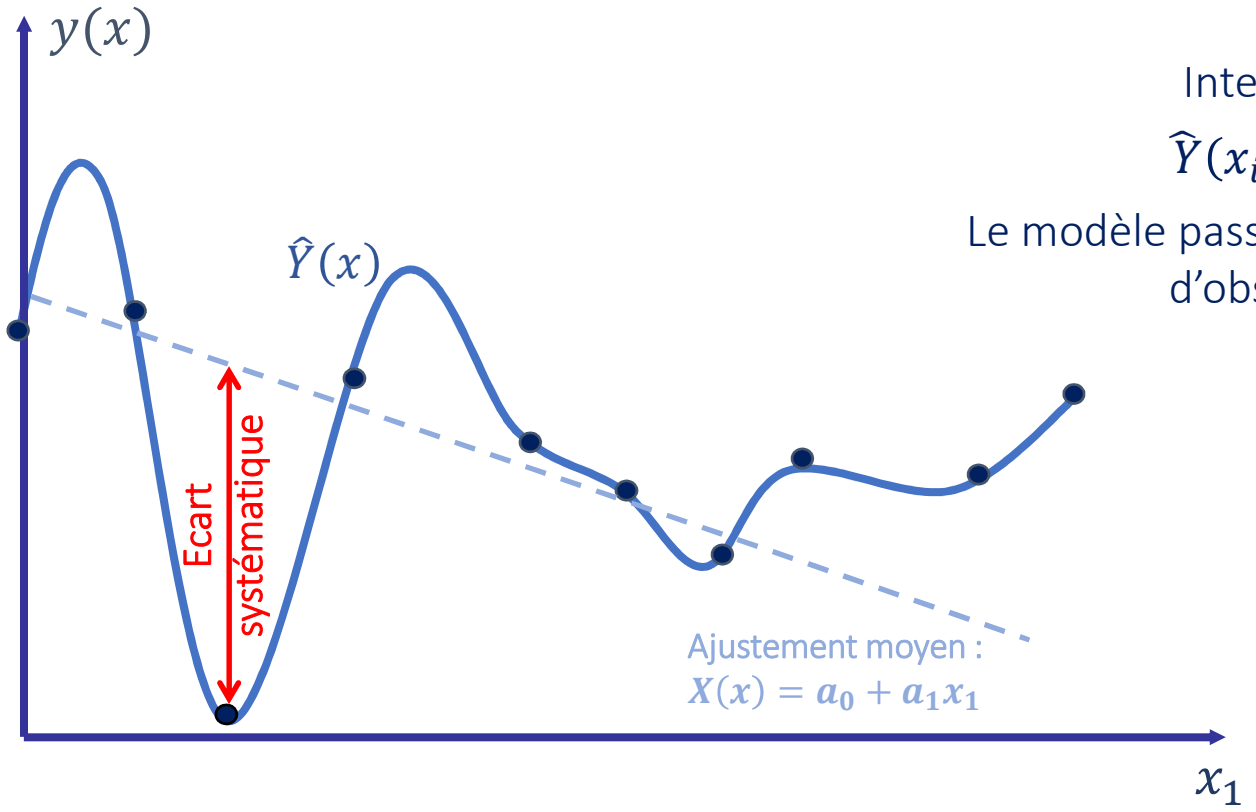


Krigeage : Modèle d'interpolation

Le prédicteur est de la forme

$$\hat{Y}(x) = \underbrace{X(x)\hat{a}}_{\text{Ajustement moyen}} + \underbrace{r^T(x)R^{-1}(Y - X\hat{a})}_{\text{Ecart systématique}}$$

où $\hat{a} = (X^T R^{-1} X)^{-1} X^T R^{-1} Y$



Interpolation

$$\hat{Y}(x_i) = \mathrm{Y}(x_i)$$

Le modèle passe par tous les points d'observations

Krigeage

	X_1	X_2		X_d
x_1	x_{11}	x_{12}	…	x_{1d}
x_2	x_{21}	x_{22}	…	x_{2d}
…	…	…	…	…
x_n	x_{n1}	x_{n2}	…	x_{nd}

Dataset

$$\hat{Y}(x) = X(x)\hat{a} + r^T(x)R^{-1}(Y - X\hat{a})$$

où $\hat{a} = (X^T R^{-1} X)^{-1} X^T R^{-1} \mathrm{Y}$

	x_1	x_2	…	x_n
x_1	1	$R_\theta(x_1,x_2)$	…	$R_\theta(x_1,x_n)$
x_2		1	…	$R_\theta(x_2,x_n)$
…		sym	…	…
x_n				1

Matrice de corrélation R

	x
x_1	$R_\theta(x_1,x)$
x_2	$R_\theta(x_2,x)$
…	…
x_n	$R_\theta(x_n,x)$

Vecteur de corrélation $r(x)$

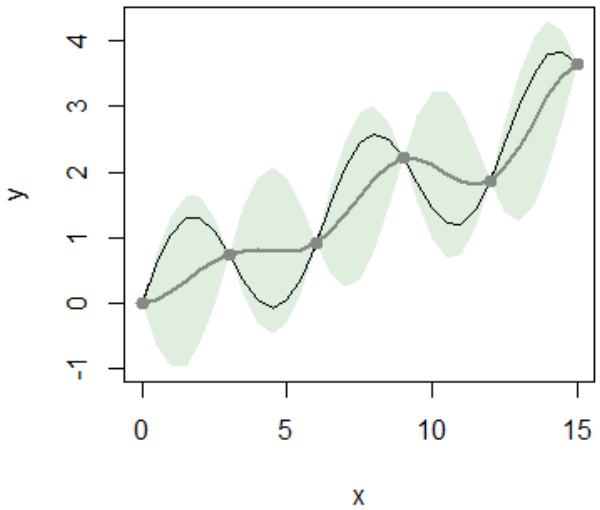
Krigeage : Stratégie expérimentale

L'erreur quadratique moyenne et l'estimation de la variance dependent de la fonction de correlation,

$$MSE(x) = E\left[\left(Y(x) - \hat{Y}(x)\right)^2\right]$$
$$= \sigma^2(1 - r^T(x)R^{-1}r(x) + K(x)(X^T R^{-1}X)^{-1}K^T(x))$$

où $K(x) = X(x) - r^T(x)R^{-1}\mathrm{X}$.

$$\hat{\sigma}^2 = \frac{1}{n}(Y - X\hat{a})^T R^{-1}(Y - X\hat{a})$$



Il est impossible de mettre en place un plan d'expériences minimisant la MSE et la variance car le paramètre θ est inconnu. Il est estimé *a posteriori* par maximum de vraisemblance. La stratégie expérimentale est itérative :

- Un plan exploratoire pour une première estimation du modèle et de θ.
- Ajout de points pour optimiser un critère statistique calculé sur le modèle obtenu.

Space-filling designs

L'objectif du plan d'experiences initial est de détecter les éventuelles irrégularités de la surface de réponse

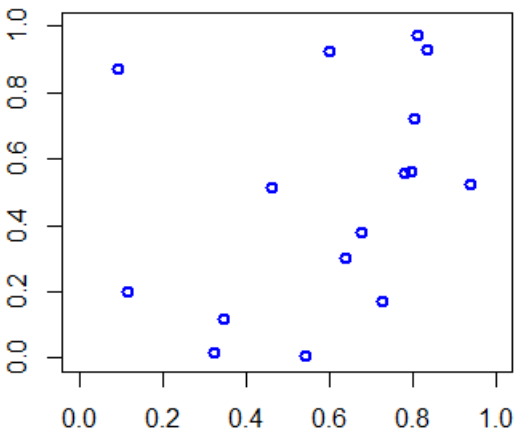
- Remplir au mieux le domaine experimental
- Tester beaucoup de niveaux pour chaque variable

Un simple tirage aléatoire ne remplit les objectifs (points redondants et espaces vides)

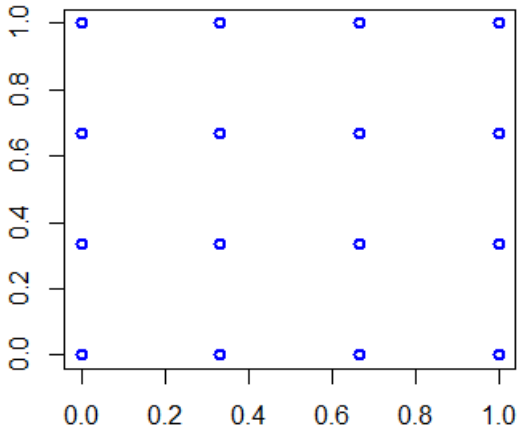
Une grille nécessite beaucoup d'experiences pour peu de niveaux testés ($n = p^d$)

$\Rightarrow$ Space-filling design

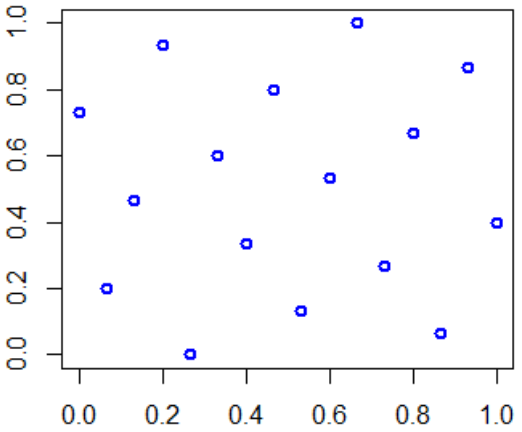
Tirages aléatoires



Grille



Space-filling design



Exemples pour $n = 16$

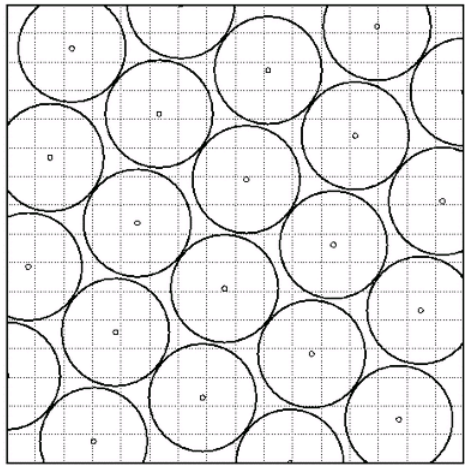
Space-filling design

Détecter les éventuelles irrégularités de la réponse

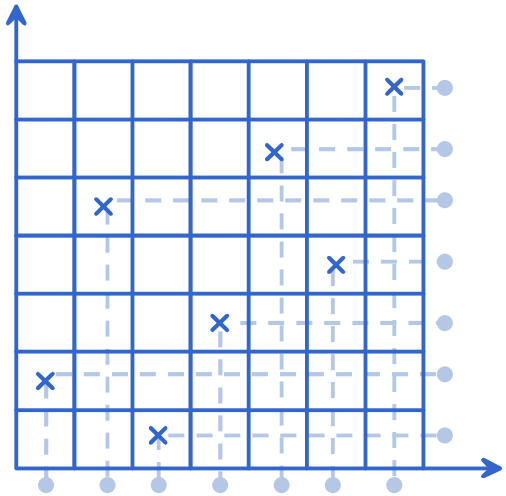
Tester toutes les parties du domaine

Tester beaucoup de niveaux

Critères de remplissage : distance Maximin, Audze-Eglais, discrépance, …

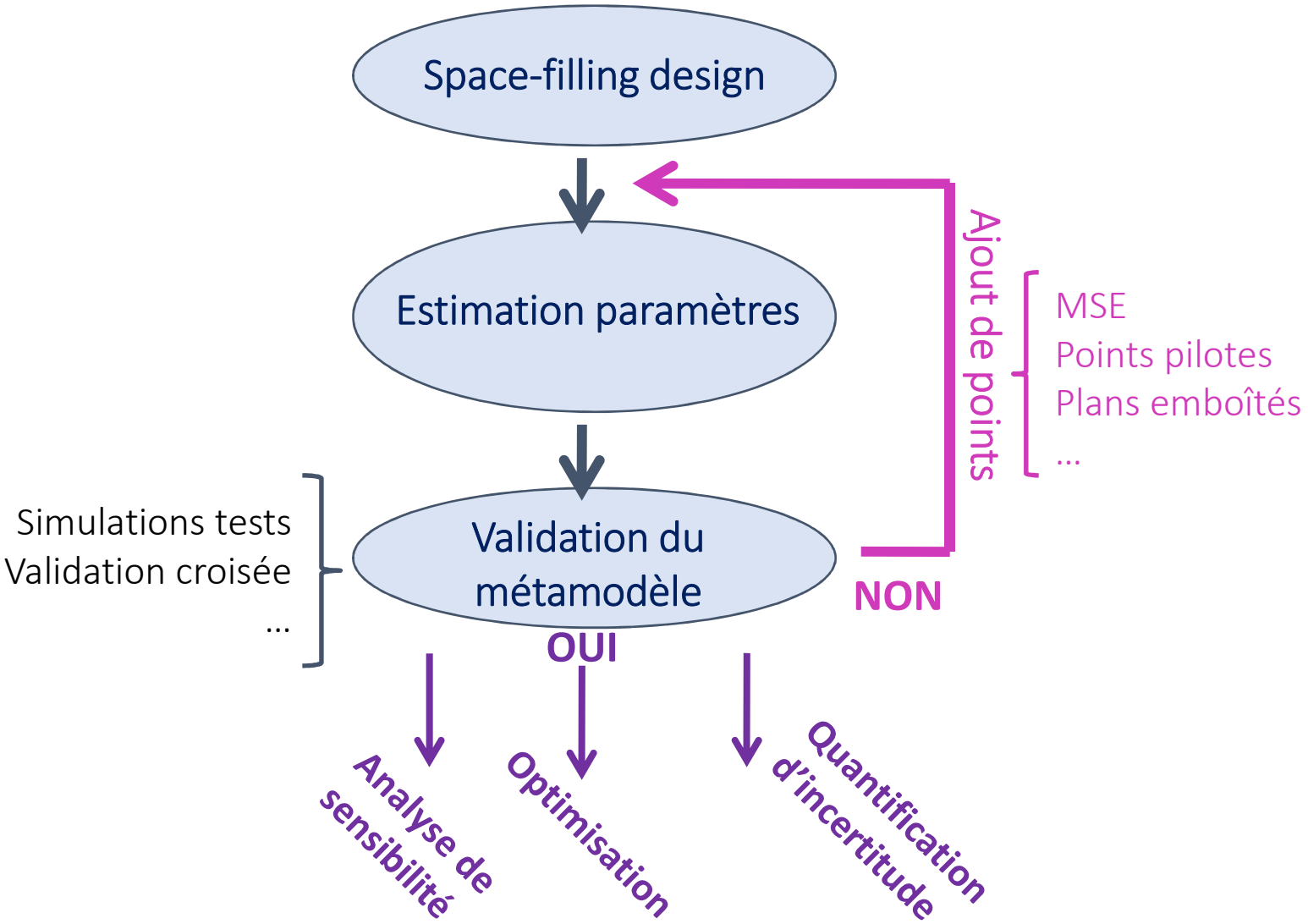


Hypercubes latins
n points = n niveaux

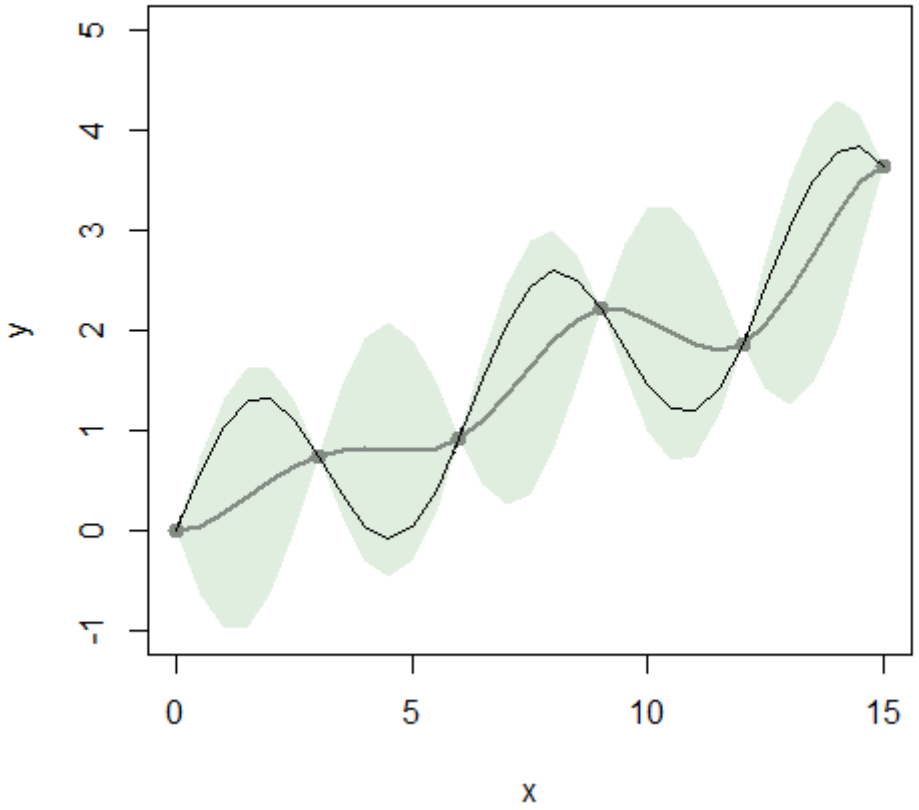


https://spacefillingdesigns.nl/

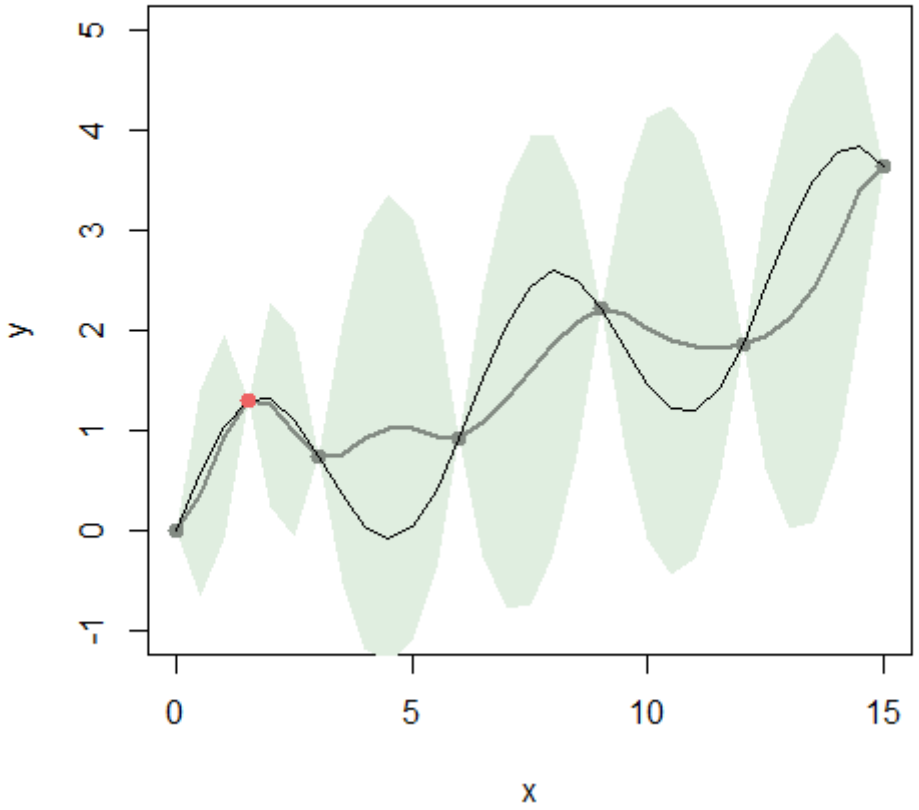
Méthodologie



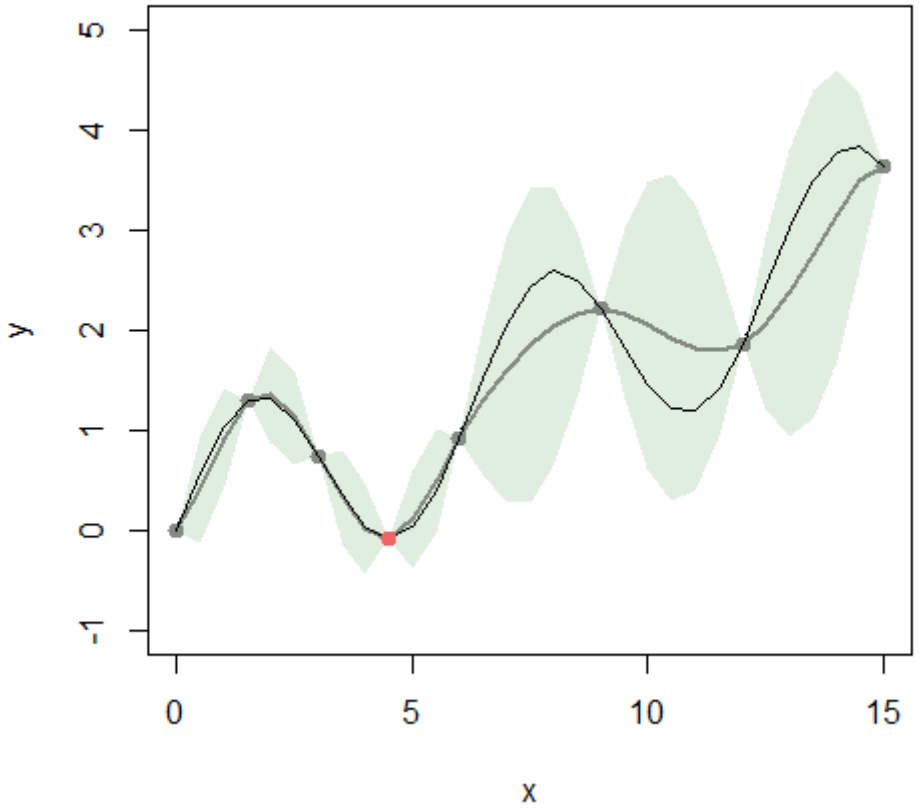
Exemple



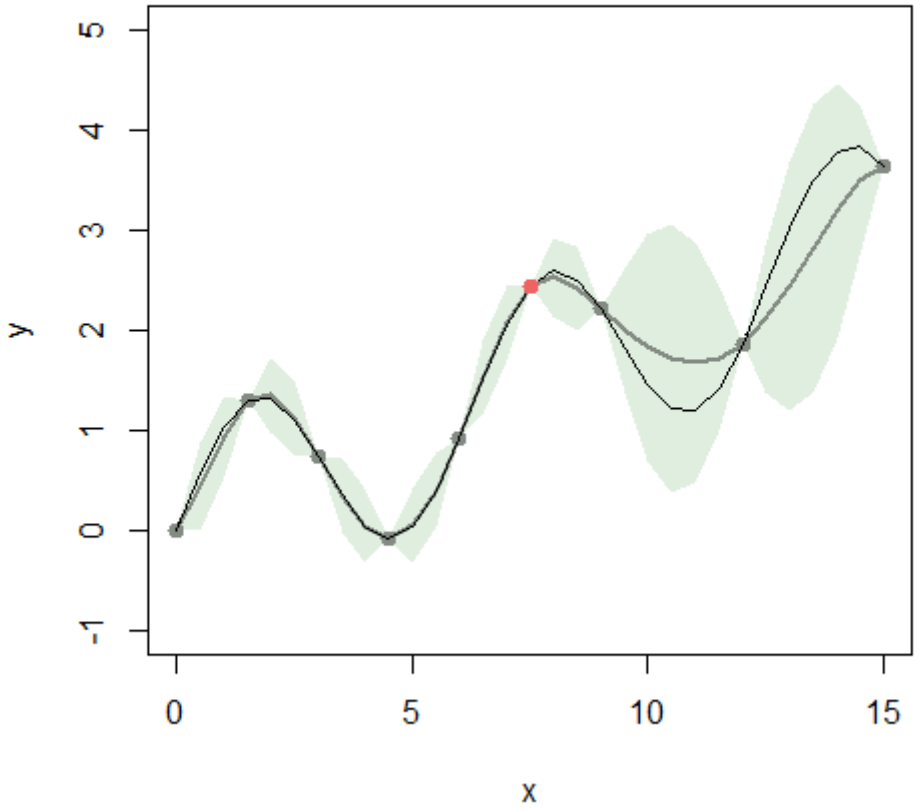
Exemple



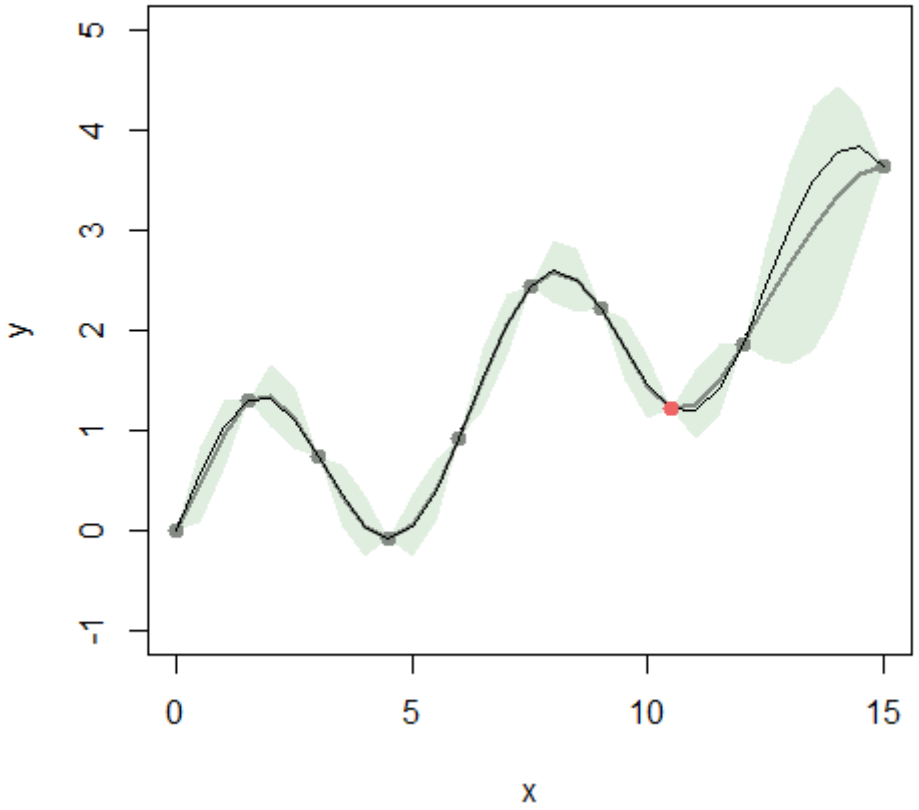
Exemple



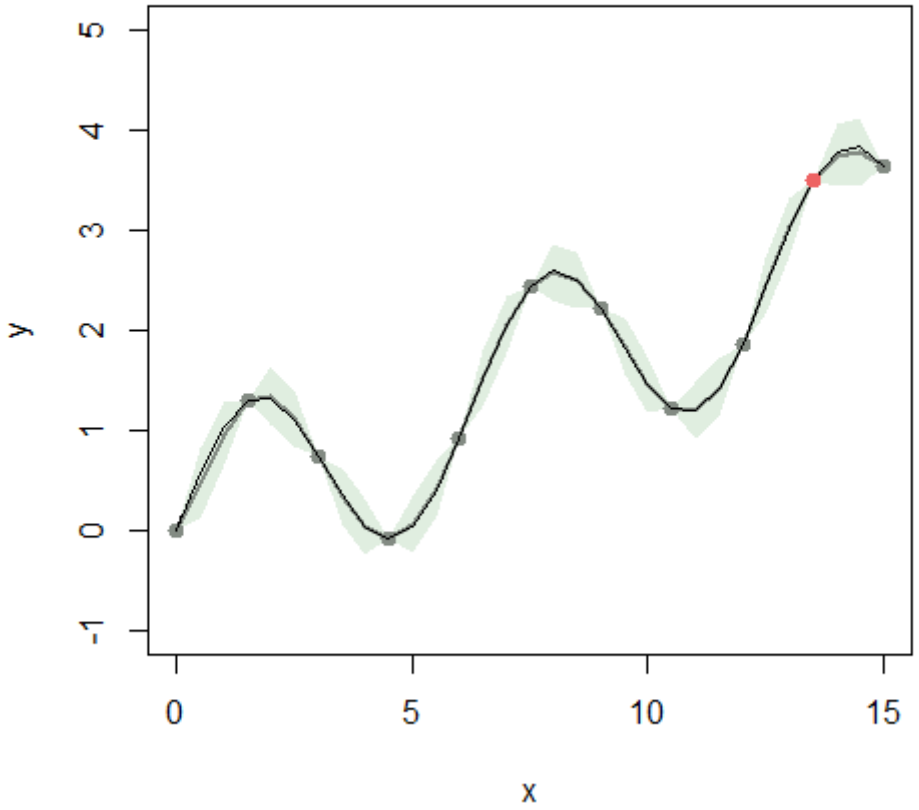
Exemple



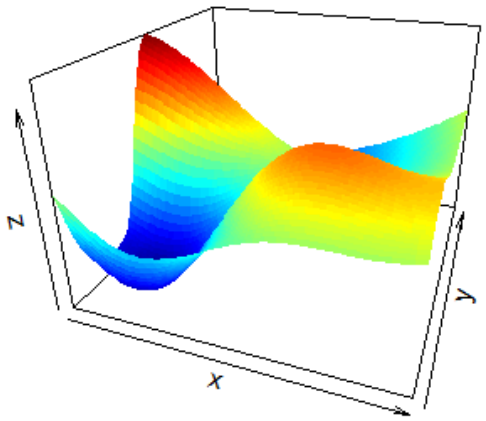
Exemple

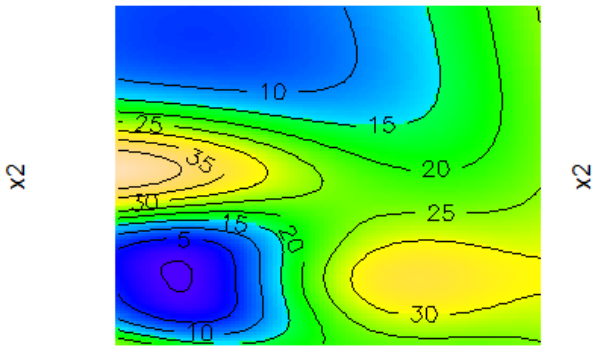


Exemple

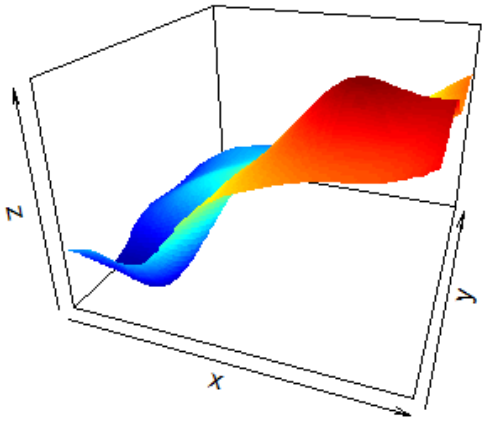


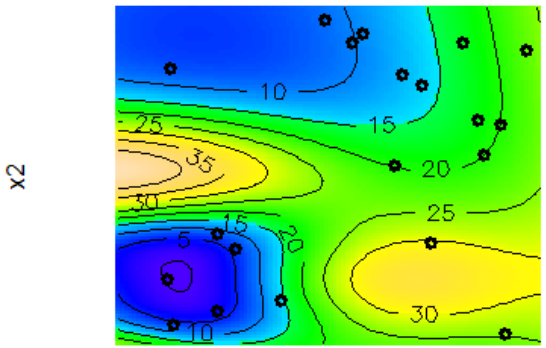
Exemple en 2D



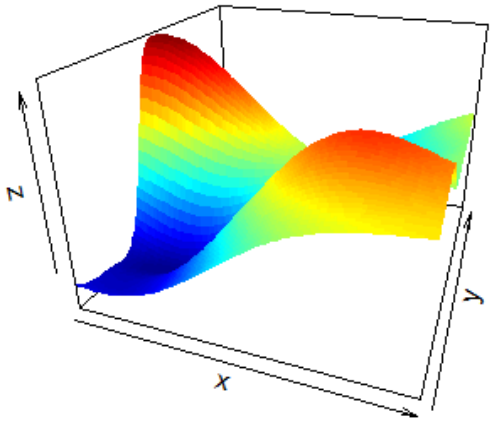


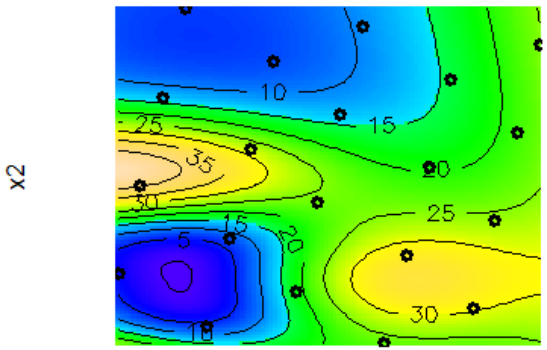
Vraie surface





Surface obtenue à par de 20 points choisis aléatoirement





Surface obtenue à par de 20 points choisis suivant un SFD

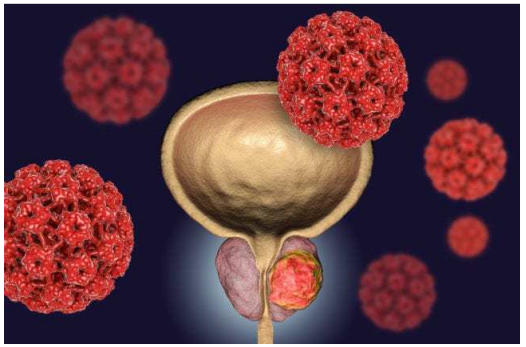
Classification (variable cible catégorielle)

- Généralités sur la classification
- Modèle polynomial : la régression logistique
- Alternative : Arbres de décision et forêts aléatoires

Rappels sur la classification

	Variables explicatives			Variable cible
Radius	…	Compactness	Symmetry	Diagnosis
23	…	0.278	0.242	Malignant
9	…	0.079	0.181	Benign
21	…	0.16	0.207	Malignant
14	…	0.284	0.26	Malignant
9	…	0.133	0.181	Malignant
25	…	0.17	0.209	Benign
21	**…**	**0.161**	**0.225**	??? Nouvelle donnée

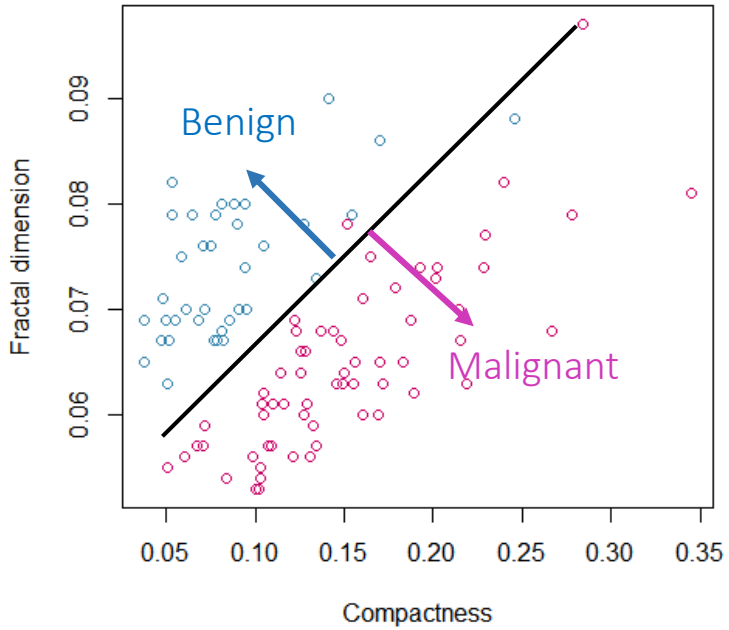
Données labelisées



https://www.santenatureinnovation.com

Les données labélisées servent à construire le modèle et à le valider.

Le modèle est ensuite utilisé pour prédire la variable cible de nouvelles données.



Rappels sur la classification
Précision du modèle

La précision d'un modèle se mesure en comptabilisant le nombre d'erreurs qu'il commet.

Radius	…	Compactness	Symmetry	Diagnosis
23	…	0.278	0.242	Malignant
9	…	0.079	0.181	Benign
21	…	0.16	0.207	Malignant
14	…	0.284	0.26	Malignant
9	…	0.133	0.181	Malignant
25	…	0.17	0.209	Benign
9	…	0.079	0.181	Benign
21	…	0.161	0.225	Malignant

Modèle

Prévision
Malignant
Malignant
Malignant
Malignant
Benign
Benign
Malignant
Malignant

Erreurs du modèle

Les erreurs sont comptabilisées par classe dans un tableau appelé ***matrice de confusion***.

Vraie valeur \ Valeur prédite	Malignant	Benign	%
Malignant	**4**	1	80%
Benign	2	**1**	33%

Recall = taux de vrais positifs

Specificity = taux de vrais négatifs

Taux de bien classés = 62,5%

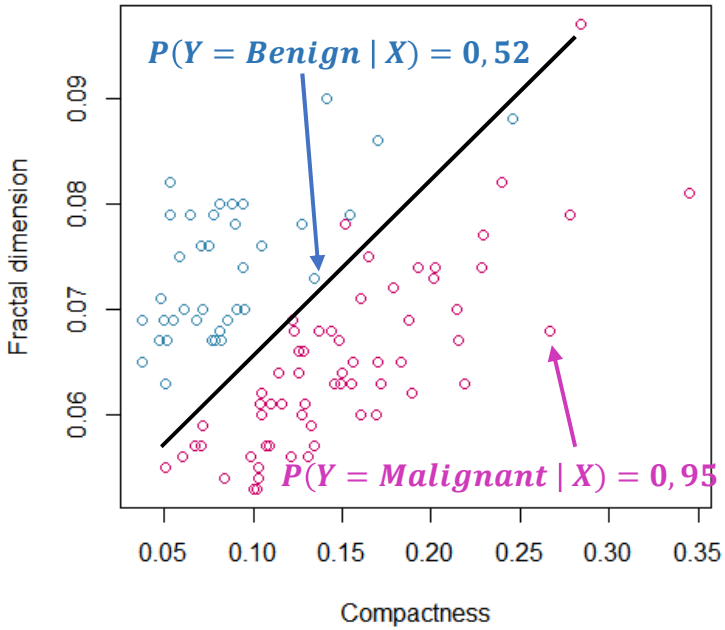
Attention aux classes déséquilibrées

Rappel sur la classification
Que modélise le modèle?

Le modèle ne retourne pas directement la classe de la variable cible Y mais la probabilité d'appartenir aux classes, connaissant les valeurs des variables explicatives X,

$$P(Y = 1|X)$$

Inutile de modéliser $P(Y = 0|X)$ car $P(Y = 0|X) = 1 - P(Y = 1|X)$. Dans le cas d'une variable cible à plus de deux classes, il faudra modéliser chaque probabilité.



Plus un nouvel individu est éloigné de la frontière, plus la probabilité d'appartenir à la classe est grande. En revanche un individu proche de la frontière aura une probabilité proche de 0,5 d'appartenir à la classe.

En général, on adopte la règle,

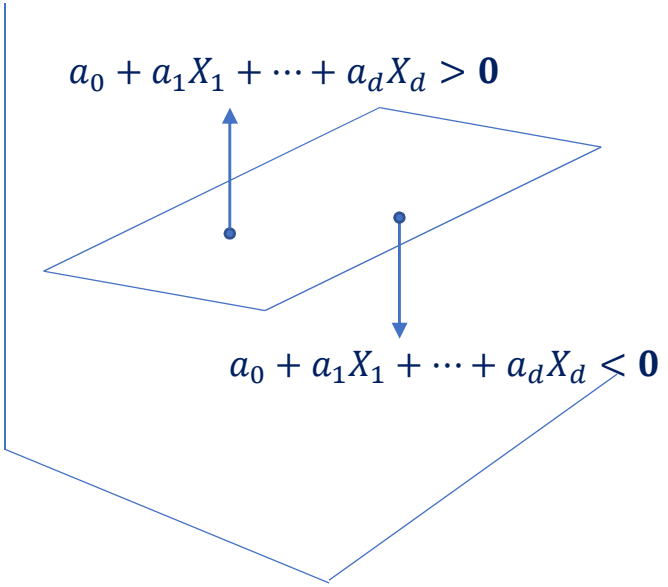
- Si $P(Y = 1|X) > 0{,}5$ on prédit la classe 1
- Si $P(Y = 1|X) < 0{,}5$ on prédit la classe 0

Quelques mots sur la régression logistique

Pour modéliser la probabilité de la classe positive, la régression logistique utilise un modèle polynomial

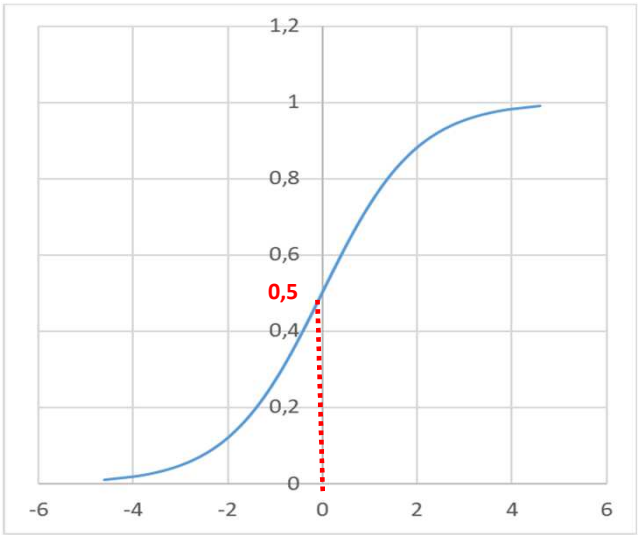
$$f(X_1, \dots, X_d) = a_0 + a_1 X_1 + \cdots + a_d X_d$$

Ce modèle définit un plan qui sépare l'espace en deux classes suivant la position en dessous ou en dessus du plan.



On applique une fonction de seuil pour ramener le résultat entre 0 et 1 (probabilité oblige),

$$\text{LOGIT}^{-1} : \quad \text{R} \rightarrow]0,1[$$
$$z \rightarrow \frac{\exp(z)}{1+\exp(z)}$$



Sur le plan, la probabilité d'appartenir à une classe est 0,5. Plus on s'éloigne en positif, plus la probabilité augmente (et vice-versa).

Quelques mots sur la régression logistique

Au final un modèle d'une forme assez simple,

$$p(x) = P(Y = 1|x) = \frac{\exp(a_0 + a_1 x_1 + \cdots + a_d x_d)}{1 + \exp(a_0 + a_1 x + \cdots + a_d x_d)}$$

Le modèle peut se résumer à un ensemble de coefficients $a_0, a_1, \ldots, a_d$.

Ces coefficients sont déterminés de façon à optimiser une fonction de coût qui mesure l'écart entre les prévisions $p(x_i)$ et les valeurs observées y_i.

Optimisation numérique pas de formule analytique contrairement à la régression linéaire

Les points forts de la régression logistique sont

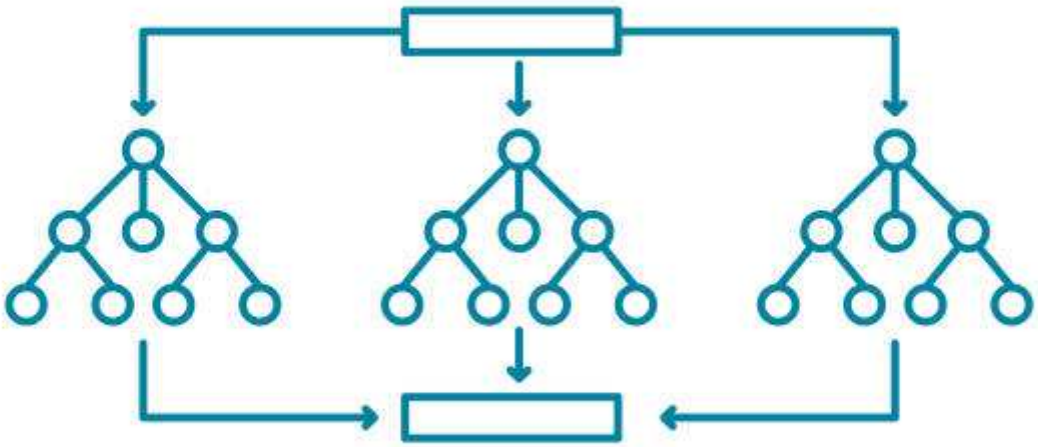
- son interprétabilité (déterminer les variables pertinentes dans le modèle + odds-ratio),
- son faible en temps de calcul (passe bien l'échelle des données massives),
- quasiment pas d'hyperparamètres à fixer.

```
Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      8.783e-01  1.470e+01   0.060   0.9524
Radius          -2.006e-02  6.969e-02  -0.288   0.7735
Texture          7.915e-02  6.970e-02   1.136   0.2561
Perimeter        9.481e-02  2.057e-01   0.461   0.6448
Area            -3.468e-03  1.324e-02  -0.262   0.7934
Smoothness      -2.014e+01  2.915e+01  -0.691   0.4897
Compactness      4.622e+01  2.324e+01   1.989   0.0467 *
Symmetry        -4.738e+00  1.911e+01  -0.248   0.8042
Fractal_dimension -1.615e+02  1.271e+02  -1.270   0.2040
```

Coefficients a_i

P-valeur

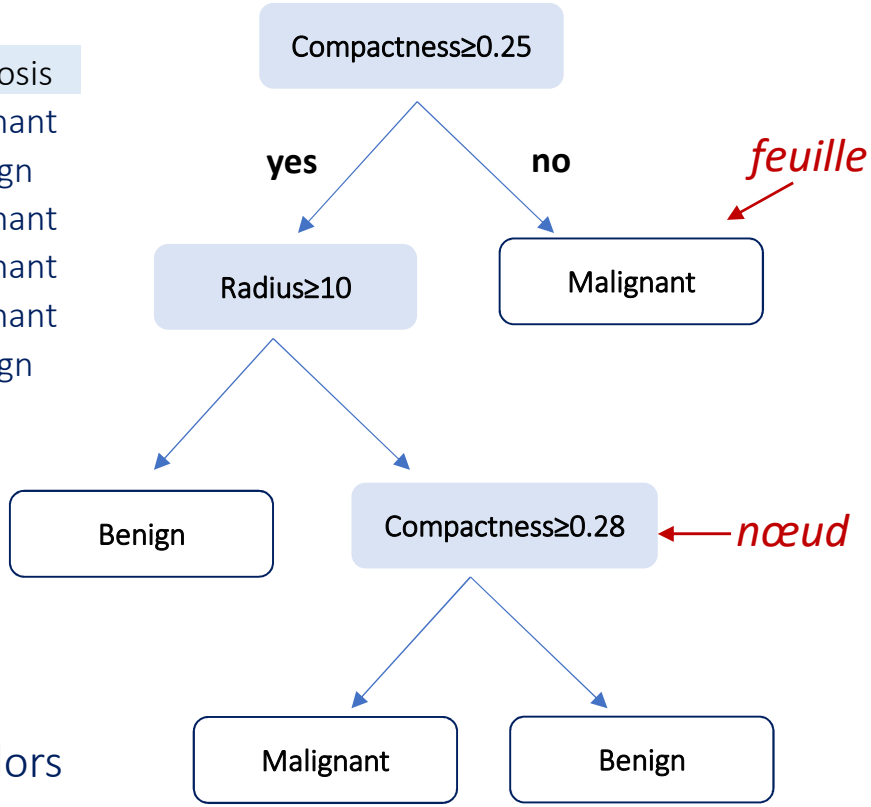
Première alternative :
Arbre de décision et forêts aléatoires



Qu'est-ce qu'un arbre de décision?

Un arbre de décision est une méthode d'apprentissage supervisée très intuitive. Son résultat est un graphe où chaque nœud intermédiaire représente un *test* et chaque nœud final (feuille) représente une *décision* (classe).

Radius	…	Compactness	Symmetry	Diagnosis
23	…	0.278	0.242	Malignant
9	…	0.079	0.181	Benign
21	…	0.16	0.207	Malignant
14	…	0.284	0.26	Malignant
9	…	0.133	0.181	Malignant
25	…	0.17	0.209	Benign



Le modèle est un ensemble de règles :

- Si Compactness < 0,25 alors Malignant
- Si Compactness > 0,25 et si Radius > 10 alors Benign

….

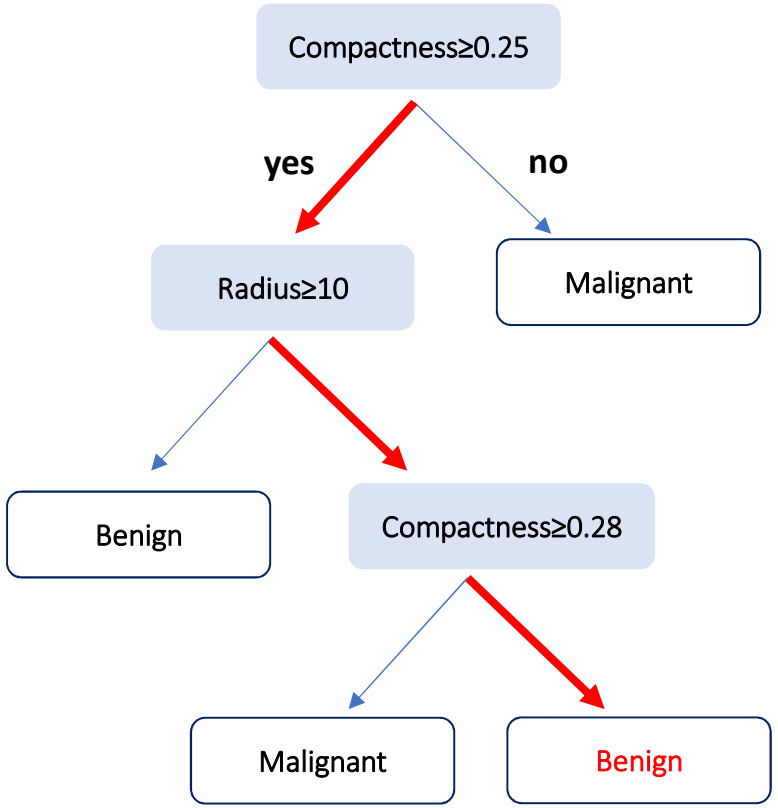
Comment utilise-t-on un arbre de décision?

Nouvel individu :

- ✓ Radius = 9
- ✓ …
- ✓ Compactness = 0.27
- ✓ Symmetry = 0.21

Diagnosis = Benign !

Rapide et interprétable



Comment construit-on un arbre?

A chaque nœud, on choisit la variable et le test qui permettra d'isoler le plus rapidement des deux classes.

Bad split

Compactness	Diagnosis
0.278	Malignant
0.079	Benign
0.16	Malignant
0.284	Malignant
0.153	Malignant
0.13	Benign

yes — Compactness≥0.20 — no

Compactness	Diagnosis
0.278	Malignant
0.284	Malignant
0.23	Malignant

Compactness	Diagnosis
0.079	Benign
0.16	Malignant
0.153	Malignant
0.13	Benign

Good split

Compactness	Diagnosis
0.278	Malignant
0.079	Benign
0.16	Malignant
0.284	Malignant
0.153	Malignant
0.13	Benign

yes — Compactness≥0.15 — no

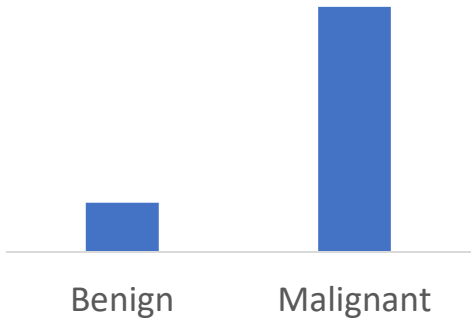
Compactness	Diagnosis
0.278	Malignant
0.16	Malignant
0.284	Malignant
0.153	Malignant

Compactness	Diagnosis
0.079	Benign
0.13	Benign

Comment construit-on un arbre?

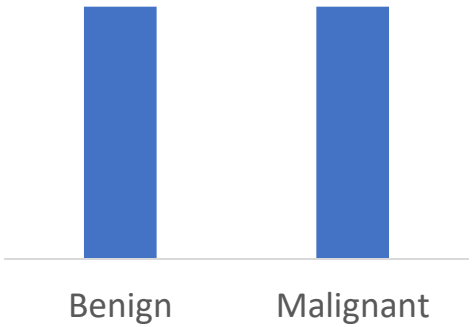
Deux indices pour mesurer la répartition des classes : Gini ou entropie

Diagnosis
Malignant
Benign
Malignant
Malignant
Malignant
Malignant



Gini	0,28
Entropie	0,65

Diagnosis
Malignant
Benign
Malignant
Benign
Benign
Benign

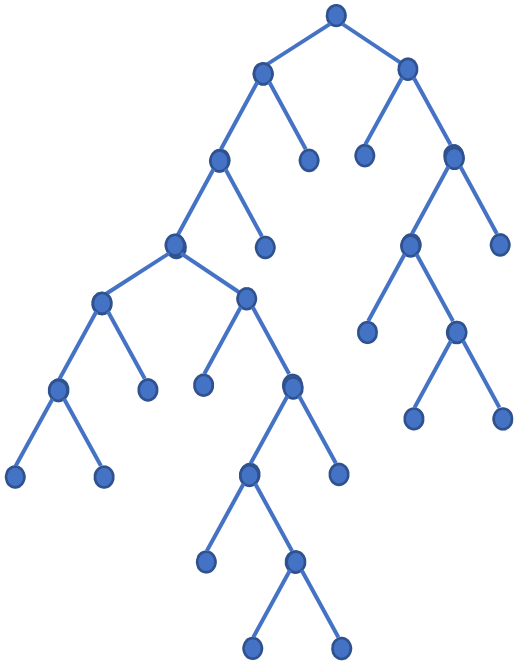


Gini	0,50
Entropie	1,00

Choisir entre Gini ou entropie dans les hyperparamètres du modèle.

Comment construit-on un arbre?

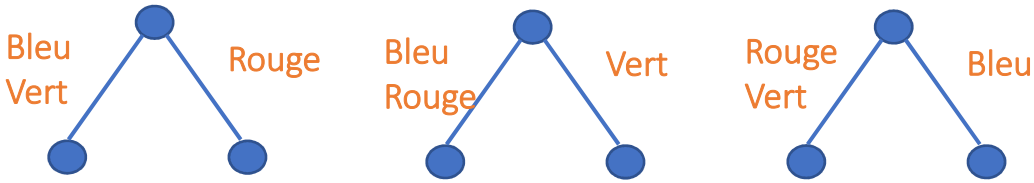
Les arbres sont binaires (2 branches à chaque noeud)



Les tests :

- Si X est numérique : $X \geq a$ avec différentes valeurs pour a
- Si X est binaire : X=0 ou X=1
- Si X est catégorielle : combinaison de modalités

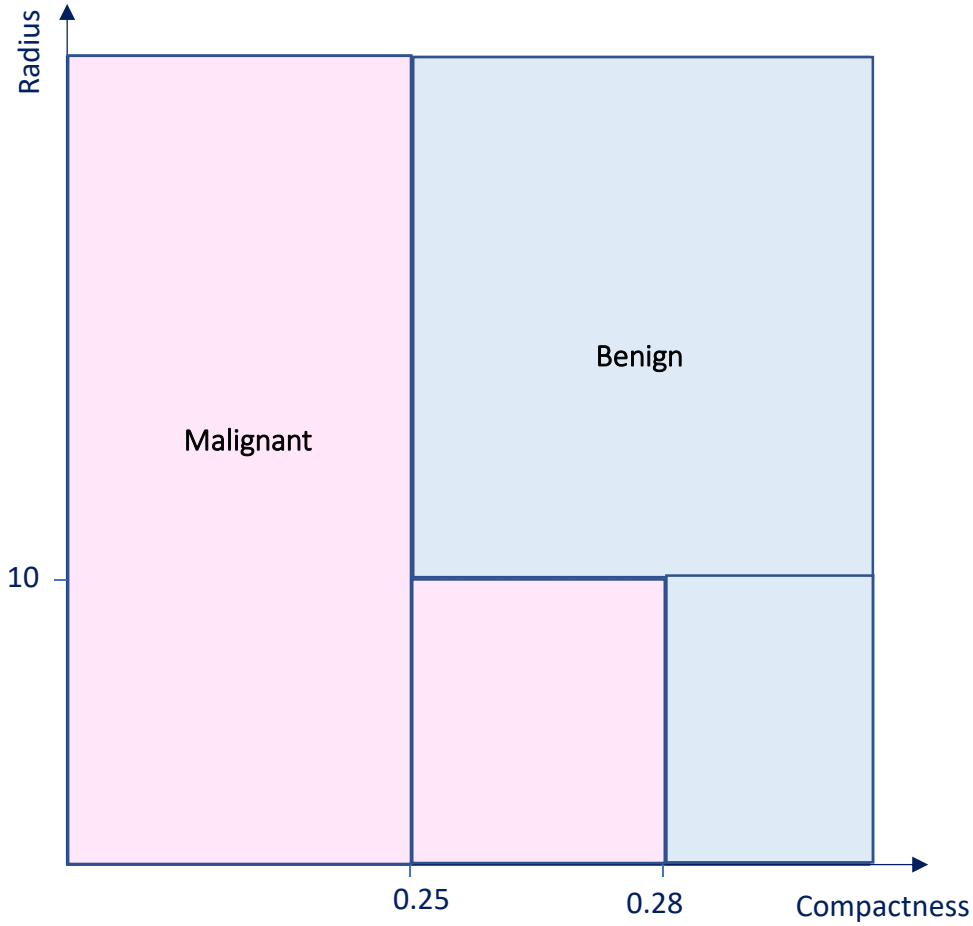
Ex. Bleu, Vert, Rouge

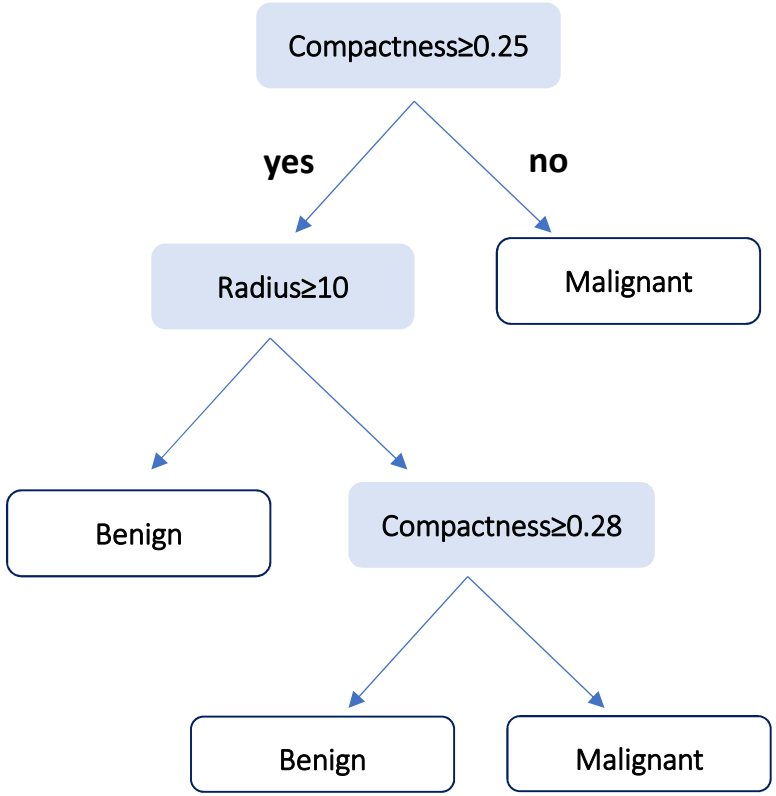


A chaque noeud, il faut mettre en compétition toutes les variables et tous les tests,

$\Rightarrow$ Complexité de l'algorithme

Pourquoi peut-on avoir des frontières plus complexes?



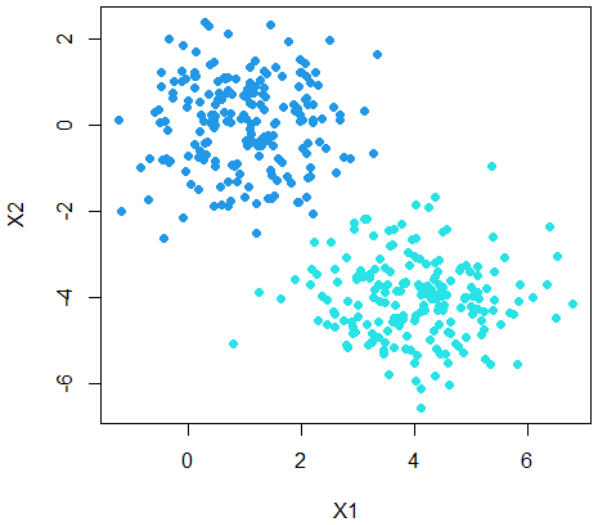


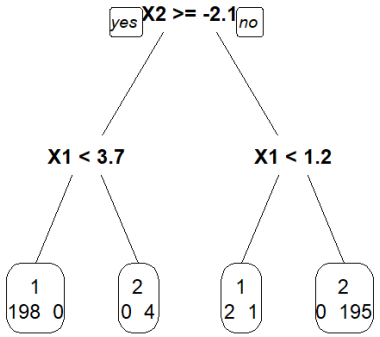
Frontières horizontales ou verticales. Plus l'arbre est profond plus les frontières seront de forme complexe ⇒ sur-apprentissage.

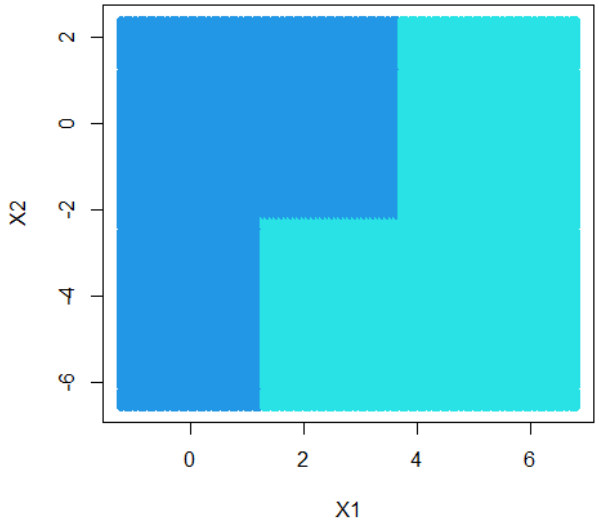
Pourquoi peut-on avoir des frontières plus complexes?

Frontières horizontales ou verticales. Plus l'arbre est profond plus les frontières seront de forme complexe.

Data → Modèle → Prévision





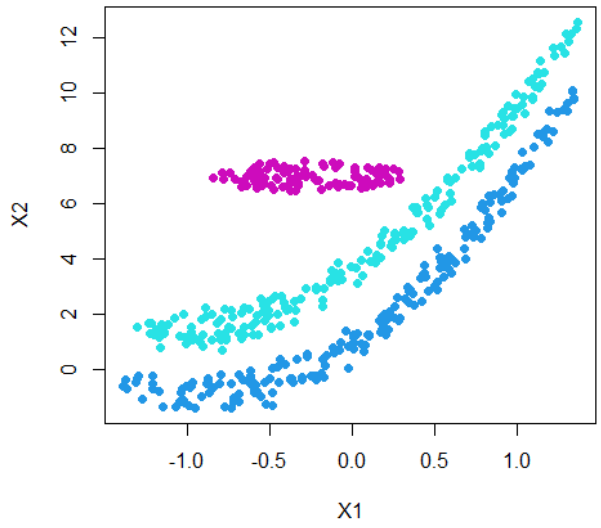


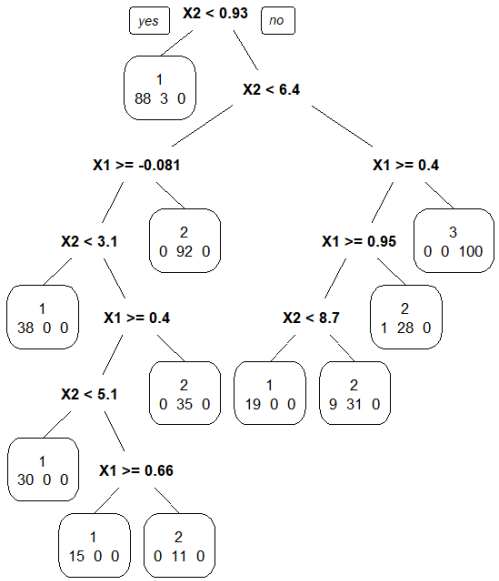
Modèle trop complexe?
Sur-apprentissage

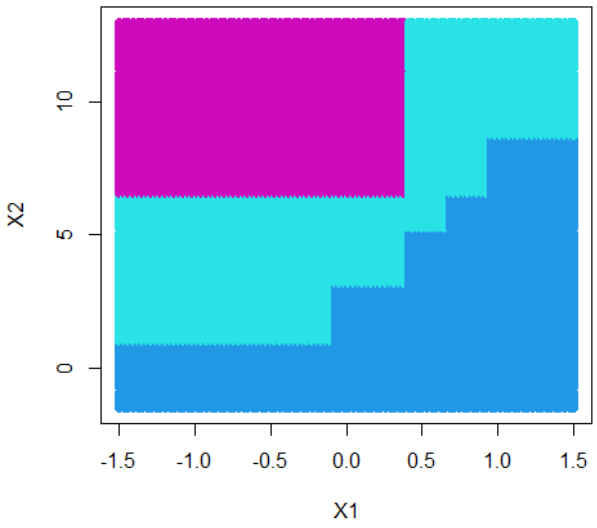
Pourquoi peut-on avoir des frontières plus complexes?

Frontières horizontales ou verticales. Plus l'arbre est profond plus les frontières seront de forme complexe.

Data → Modèle → Prévision





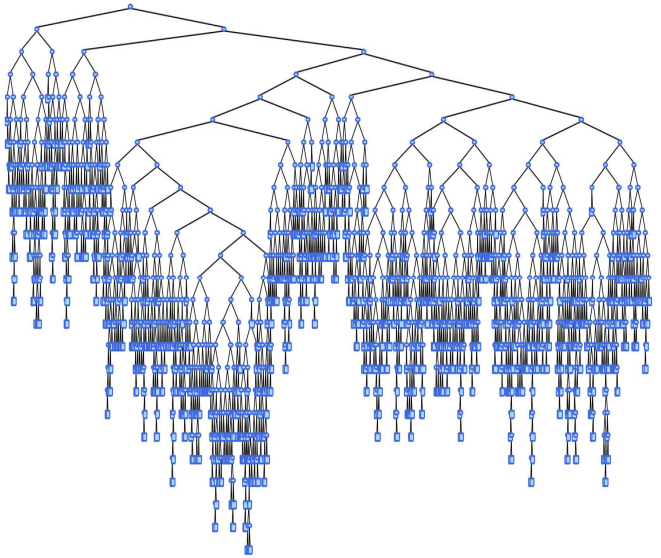


Modèle trop complexe?
Sur-apprentissage

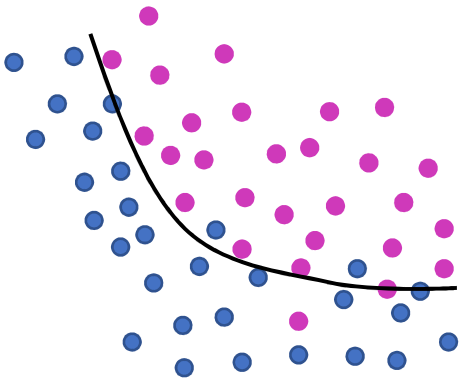
Sur-apprentissage

Le revers de la médaille avec les modèles de machine learning permettant d'avoir des frontières complexes est le sur-apprentissage.

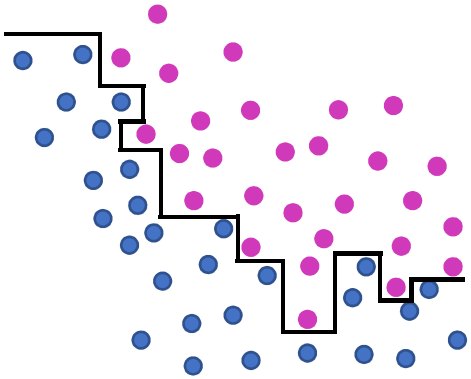
Le modèle devient suffisamment complexe pour apprendre les cas particuliers des données d'entrainement du modèle. Il sera alors trop contraint par ces données et il sera incapable de généraliser, c-a-d de prédire de nouvelles données.



Bon apprentissage



Sur-apprentissage

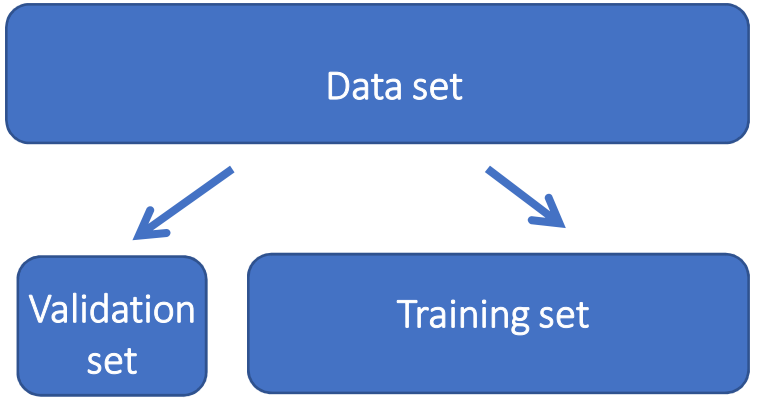


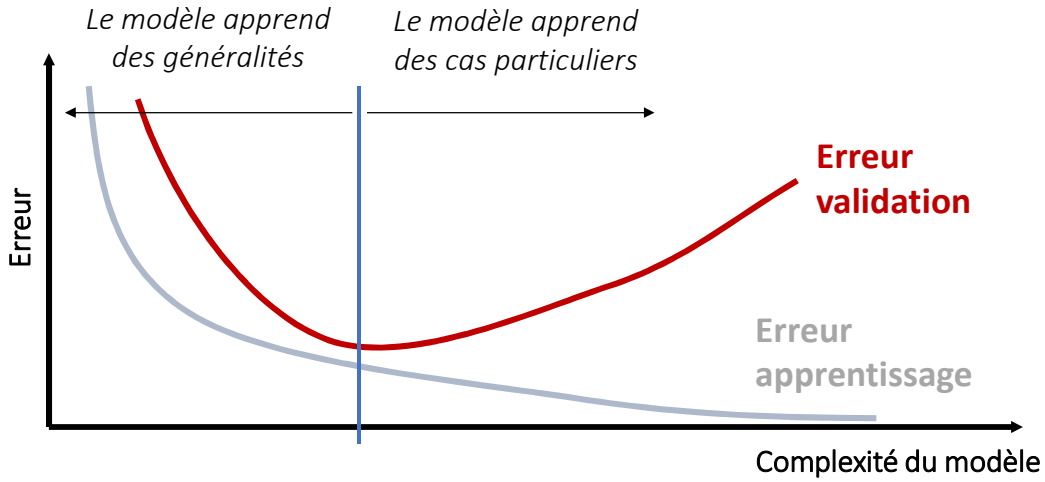
- Elagage
- Forêts aléatoires

Comment détecte-t-on le sur-apprentissage?

Le jeu de données se divise en deux:

- La ***base d'apprentissage*** est utilisée pour ajuster les paramètres du modèle (les coefficients d'une régression linéaire, les tests de partage dans un arbre de décision, …).
- La ***base de validation*** est utilisée pour définir les hyperparamètres d'un modèle (le degré d'un polynôme, la profondeur d'un arbre,…).

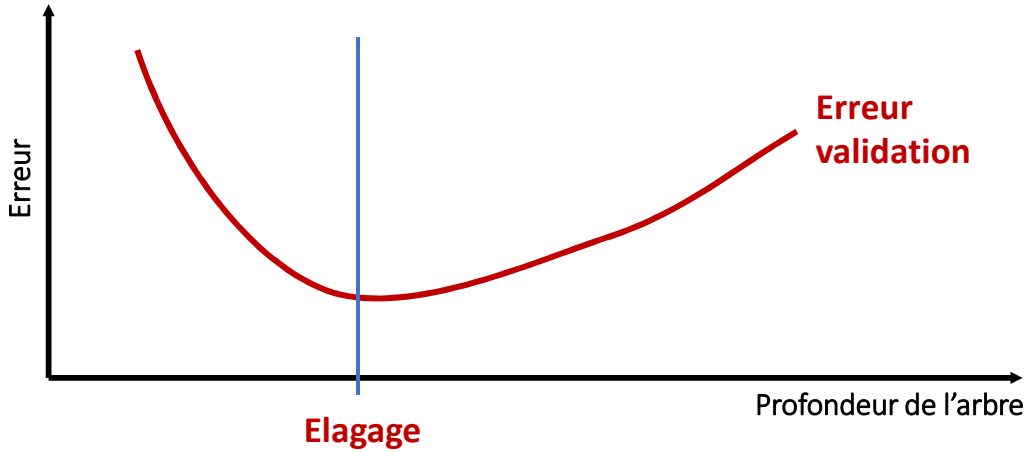


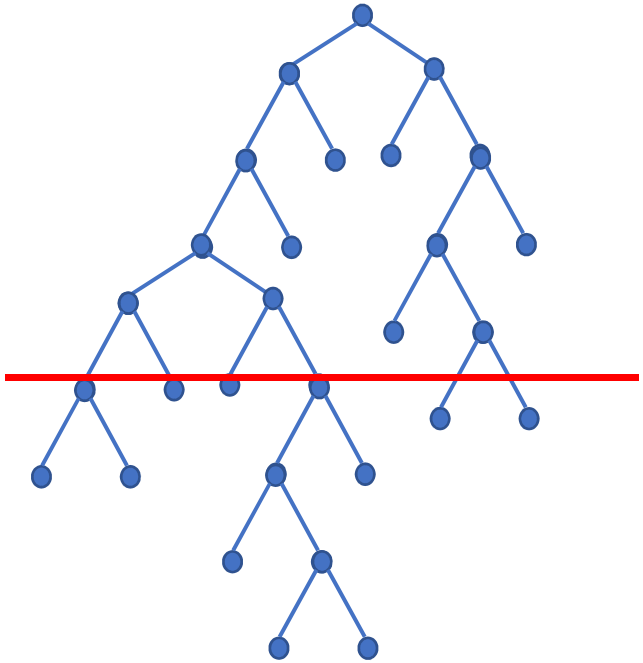


Un écart entre les deux erreurs suggère un sur-apprentissage.

Elagage d'un arbre

L'élagage consiste à couper les branches les plus profondes d'un arbre pour éviter quelles n'apprennent des cas particuliers.





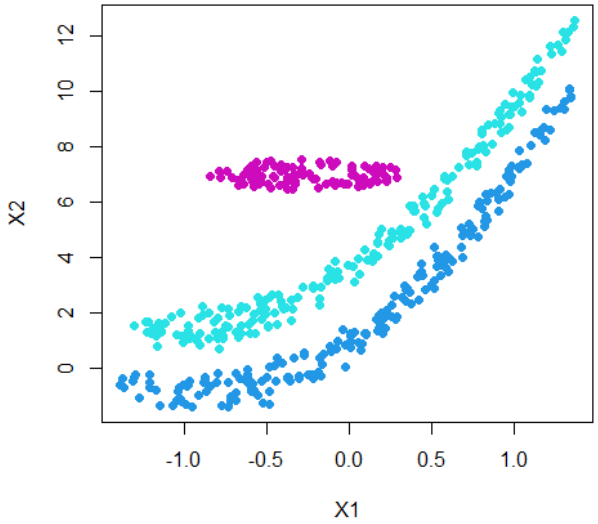
L'élagage se gère avec des hyperparamètres :

- Nombre min d'observations dans une feuille
- Nombre min d'observations dans un nœud
- Nombre max de nœuds

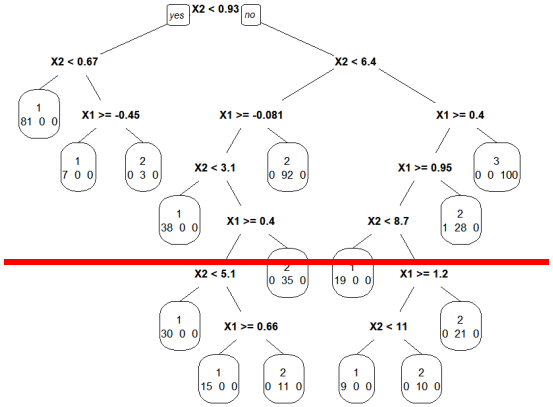
⇒ Problème du choix des hyperparamètres

Elagage d'un arbre

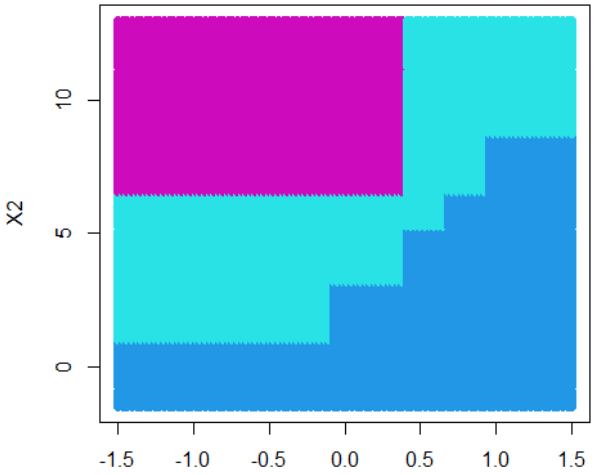
Dataset



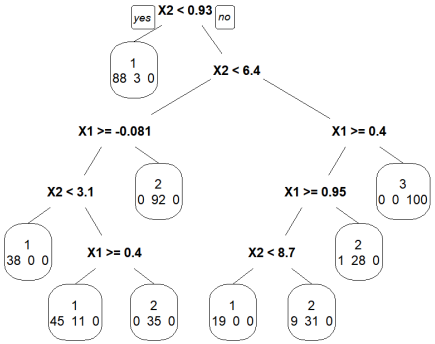
Modèle

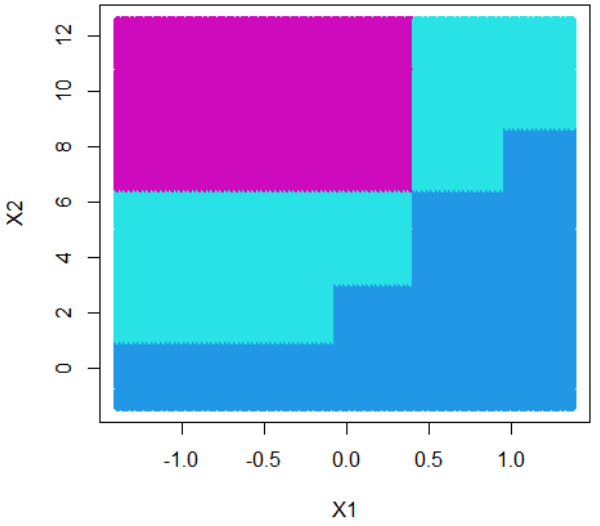


Frontières



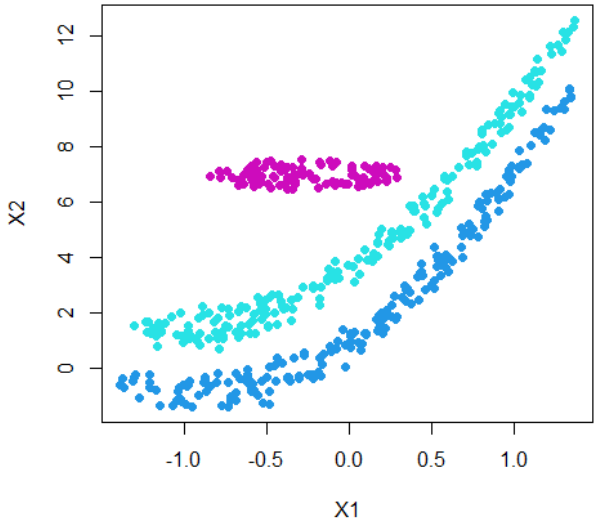
Plus on élague l'arbre, plus les frontières seront simples.



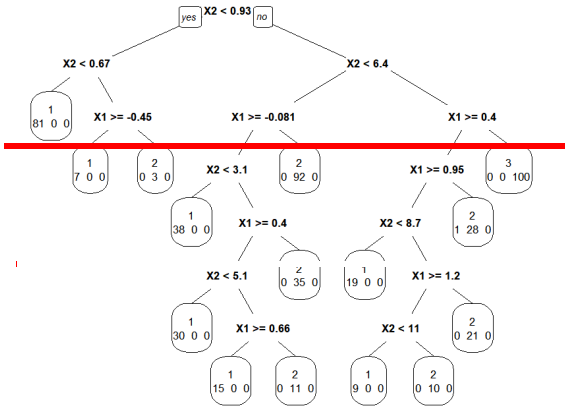


Elagage d'un arbre

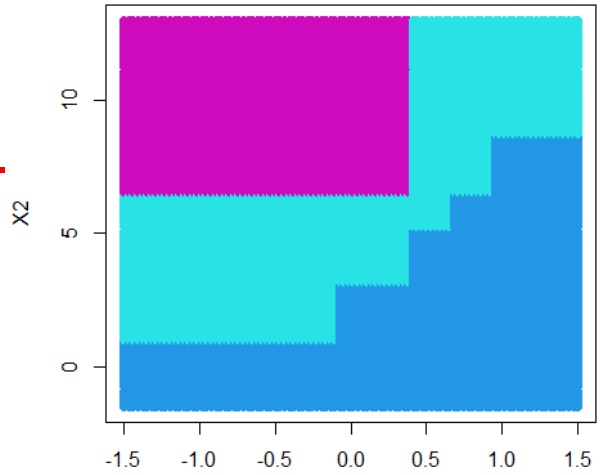
Dataset



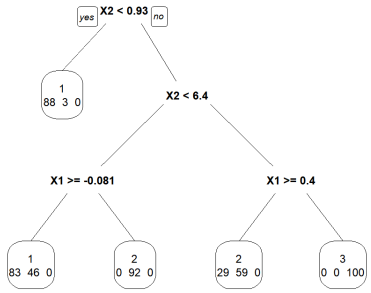
Modèle

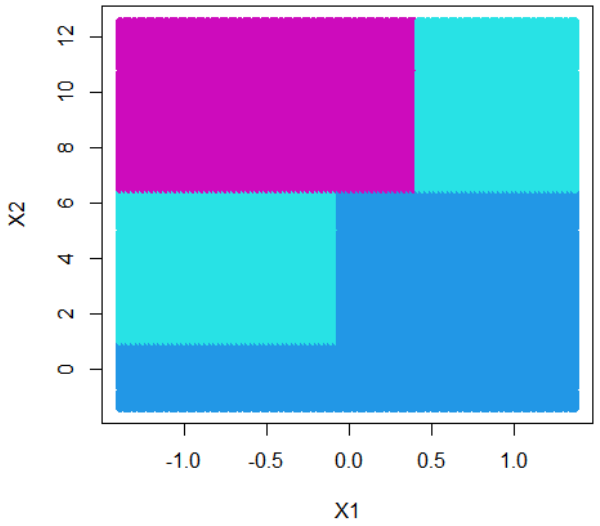


Frontières

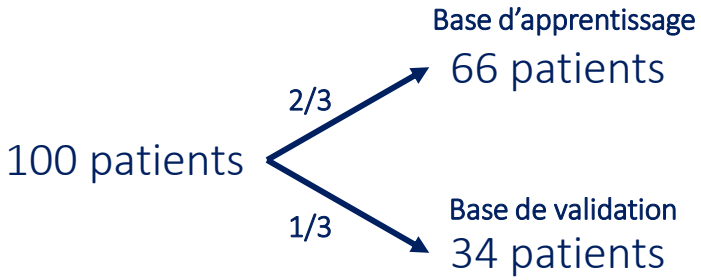


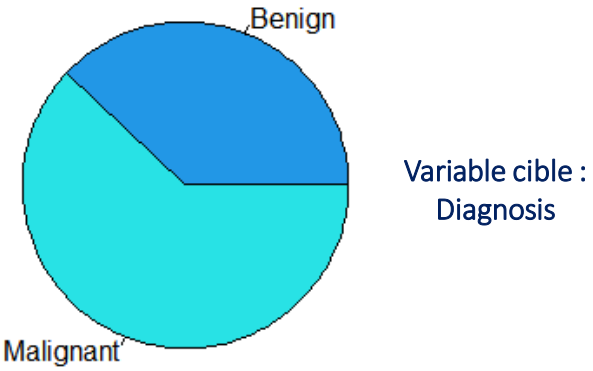
Plus on élague l'arbre, plus les frontières seront simples.





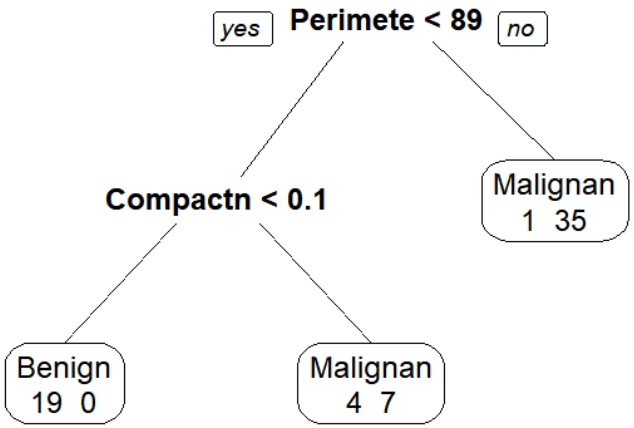
Application aux données sur le cancer de la prostate





Variable cible : Diagnosis

Résultats après élagage



Erreur sur la base validation

	prev.v	
	Benign	Malignant
Benign	9	5
Malignant	1	19

Le modèle commet 18% d'erreur sur les nouvelles données. La classe *Malignan*t est mieux apprise que la classe *Benign*.

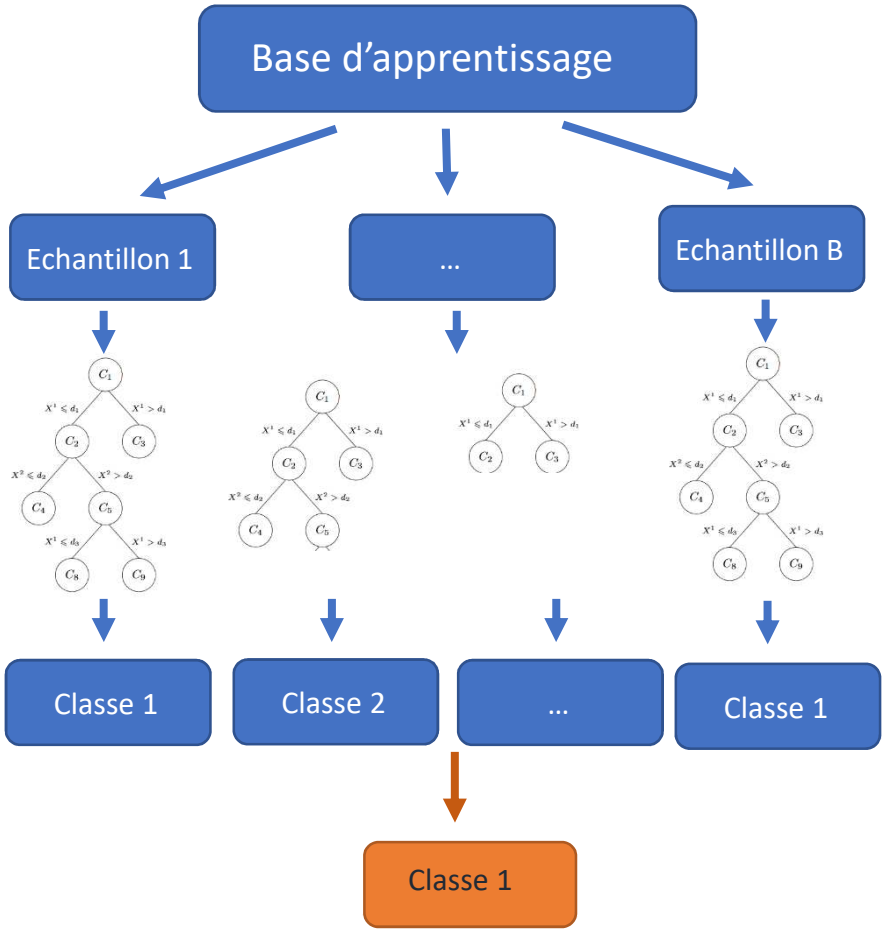
Les variables qui permettent la classification sont *Compactness* et *Perimeter*.

Forêts aléatoires

Une forêt aléatoire est une agrégation d'arbres.

Chaque arbre est construit sur un échantillon bootstrap (tirage aléatoire avec remise) de la base d'apprentissage. Une observation peut apparaitre dans plusieurs échantillons et ne pas être considérée dans les autres.

Chaque arbre prédit une classe et la classe finale est obtenue par vote majoritaire.



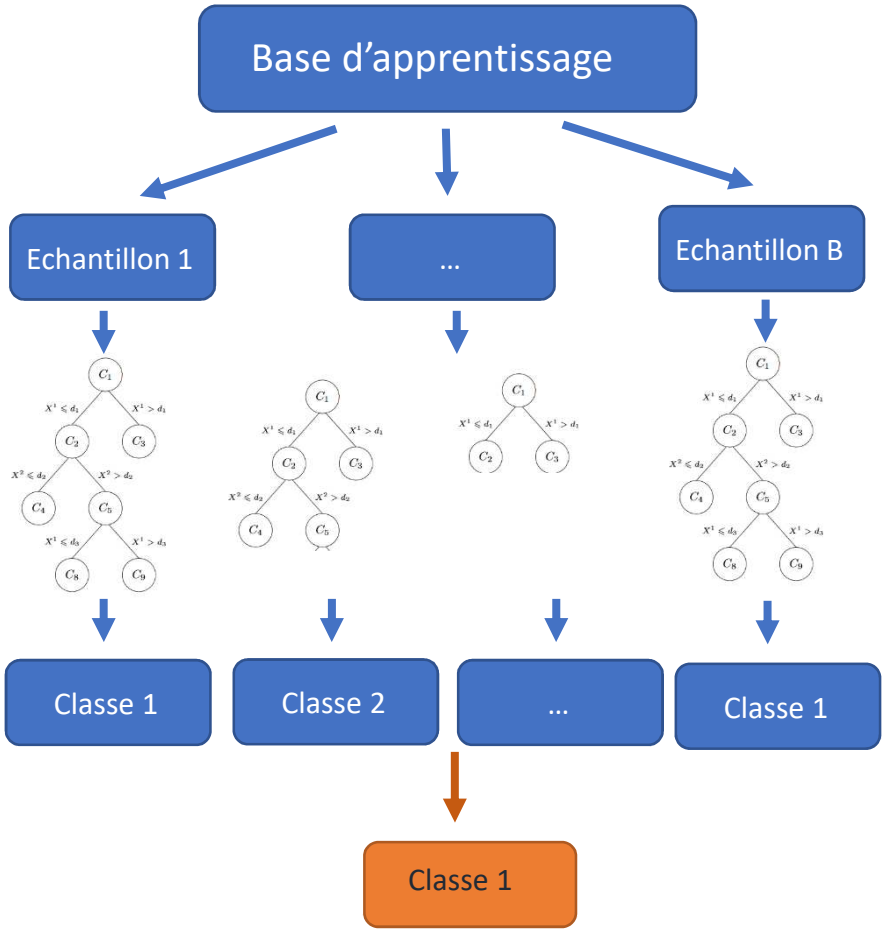
Forêts aléatoires

Pour que le vote majoritaire soit représentatif, il faut que les arbres soient différents.
L'échantillonnage bootstrap ne suffit pas.

⇒ Choix aléatoire d'un sous-ensemble de variables de split à chaque nœud

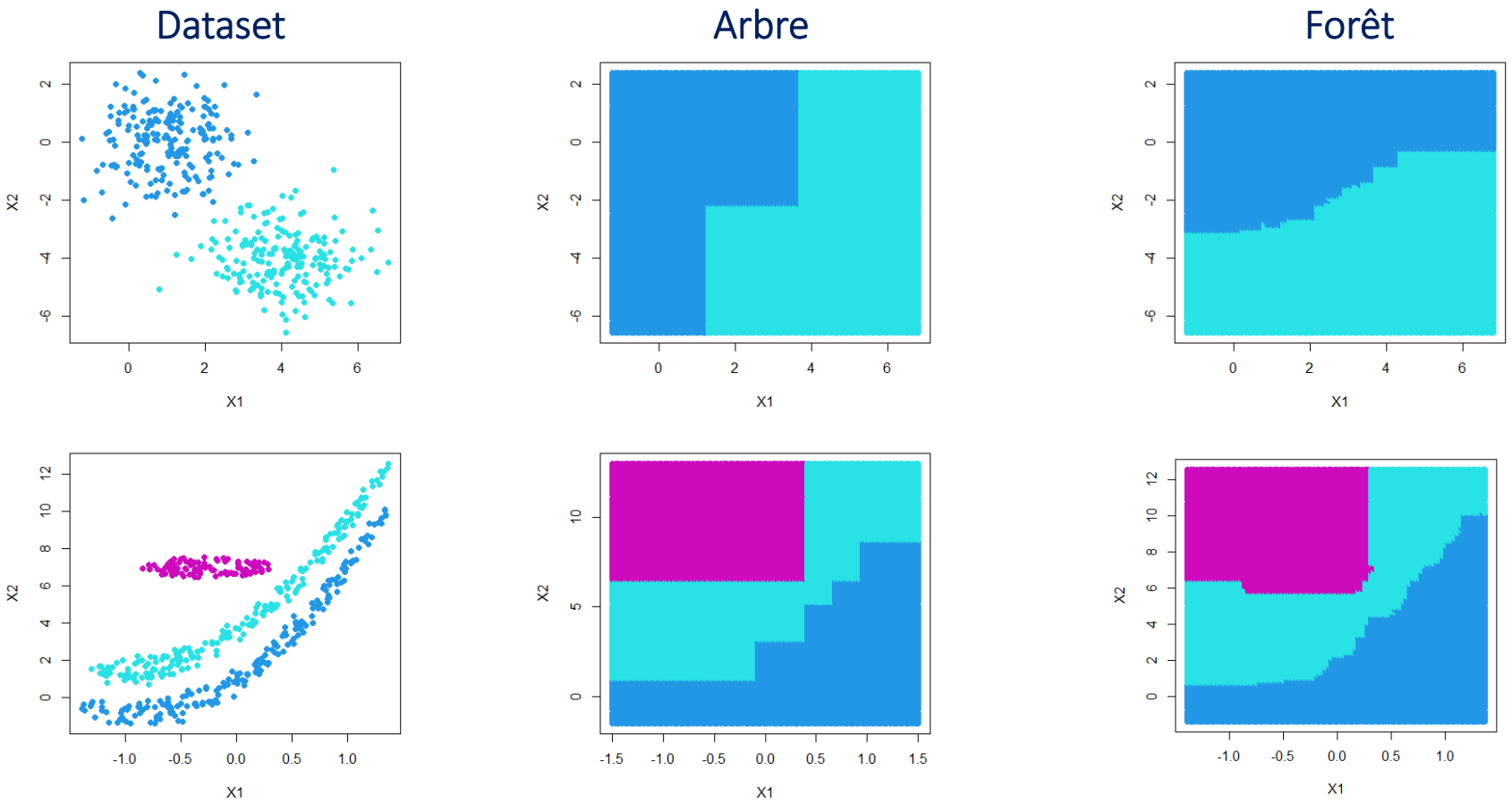
Donc 2 niveaux d'aléa dans l'algorithme.

Chaque arbre est très simple (peu profond et pas toutes les variables) et donc peu précis.
La force de la forêt réside dans le grand nombre d'arbres.



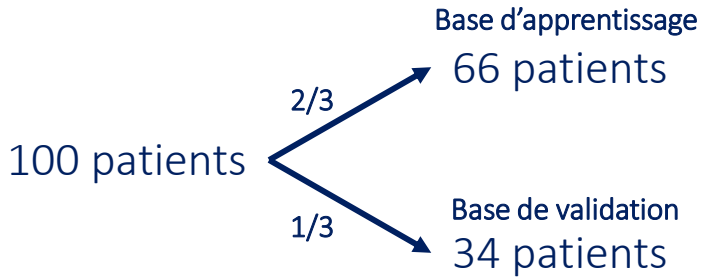
Les arbres sont construits indépendamment donc possibilité de paralléliser les calculs

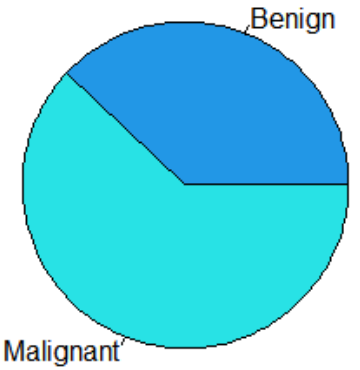
Quelles frontières avec une forêt?



Frontières plus précises mais modèle plus complexe avec un algorithme stochastique et plus d'hyperparamètres à gérer (nombre d'arbres, nombre de variables par split,…)

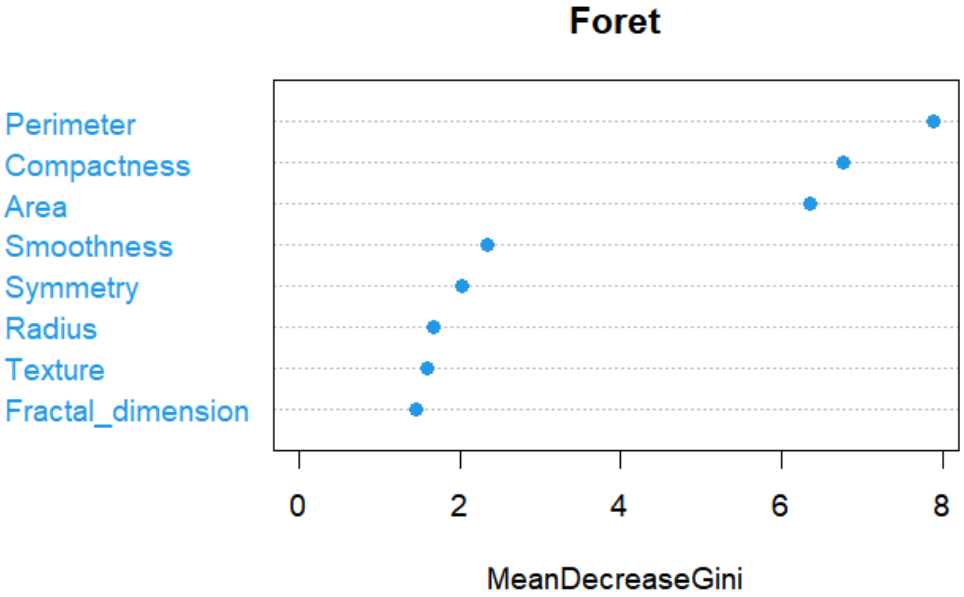
Application aux données sur le cancer de la prostate





Variable cible : Diagnosis

Résultats avec 500 arbres



Sur base validation

	prev.v	
	Benign	Malignant
Benign	9	5
Malignant	2	18

Le modèle commet 20% d'erreur sur les nouvelles données. La classe *Malignant* est mieux apprise que la classe *Benign*.

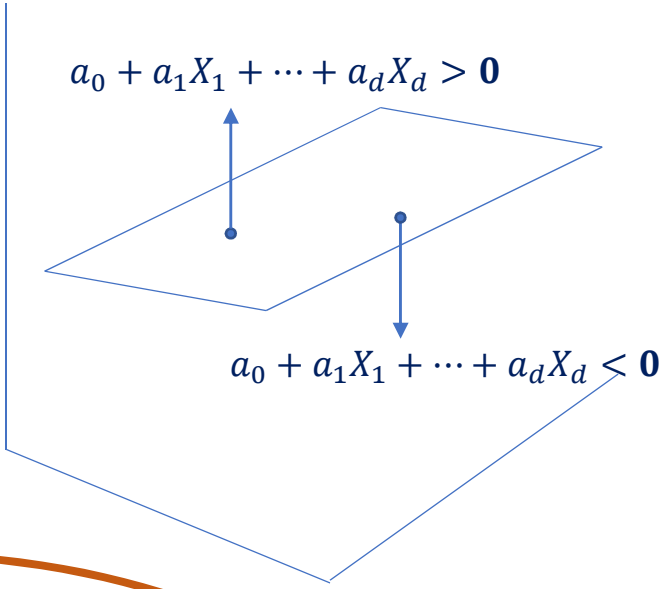
Les variables qui permettent la classification sont *Compactness*, *Perimeter* et *Area*.

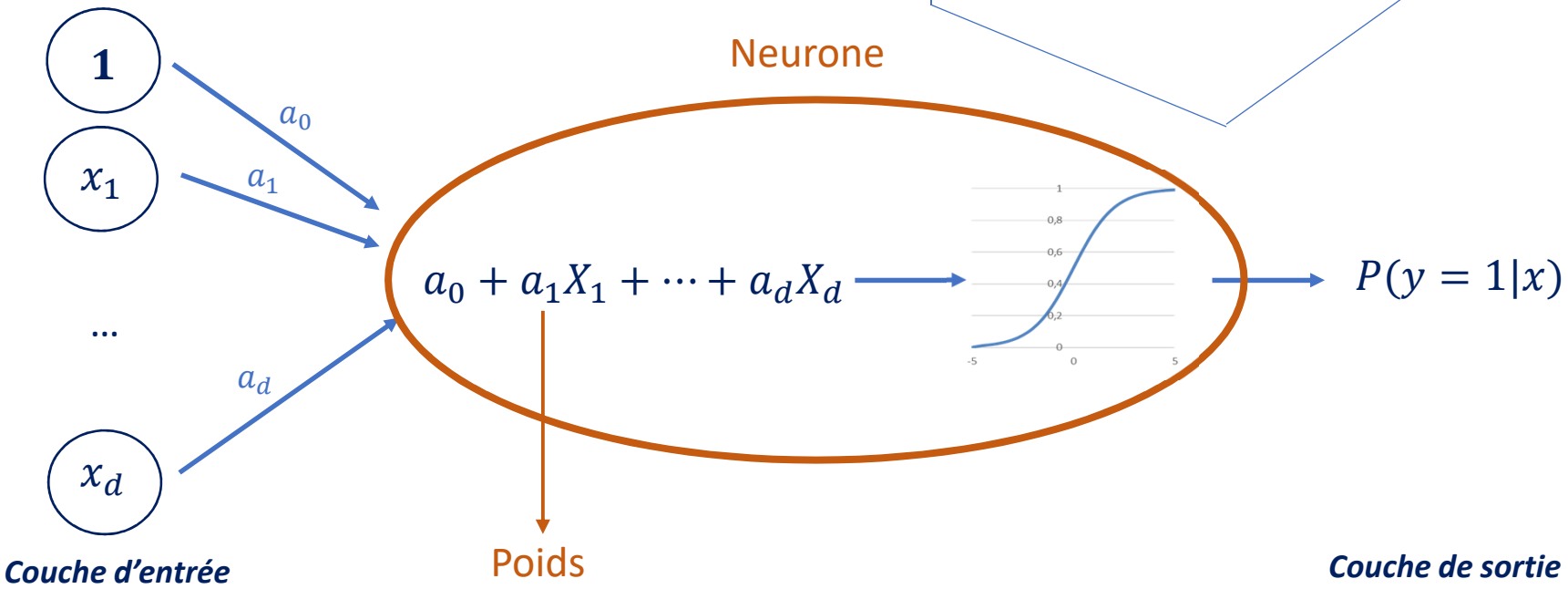
Et les réseaux de neurones

Réseau de neurones

La régression logistique est un réseau de neurones avec un seul neurone!

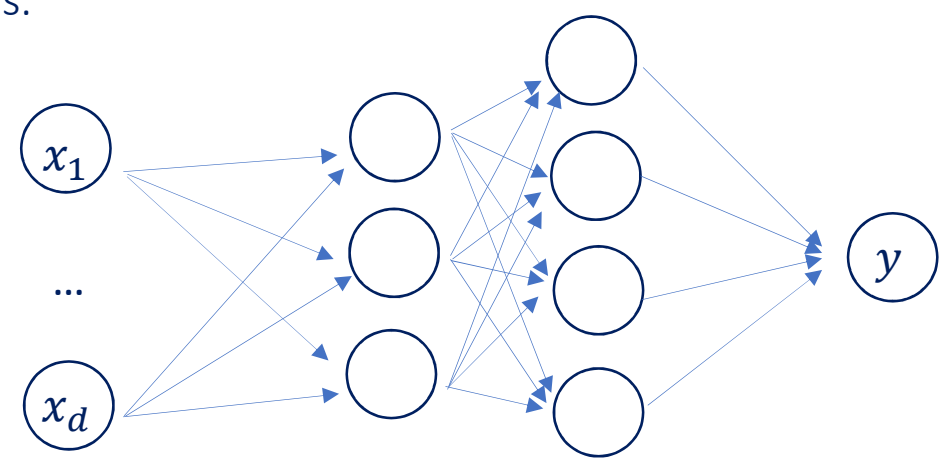
Neurone = combinaison linéaire des entrées
+ fonction de seuil



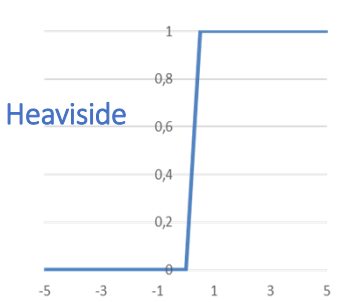


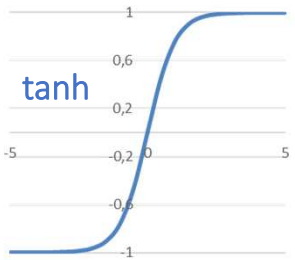
Réseau de neurones

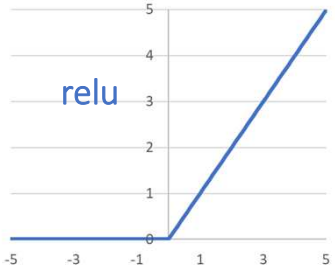
Un réseau de neurones est un empilement couches intermédiaires constituées de plusieurs neurones.

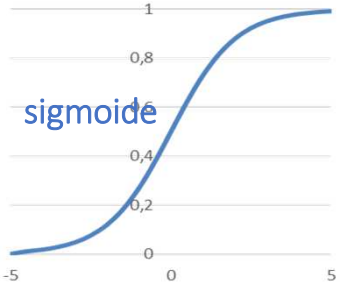


Sur la couche de sortie, la fonction d'activation est toujours la logit inverse (sigmoide) pour avoir une probabilité (ou softmax si plus de 2 classes).
Sur les couches cachées, les fonctions de seuil sont diverses:









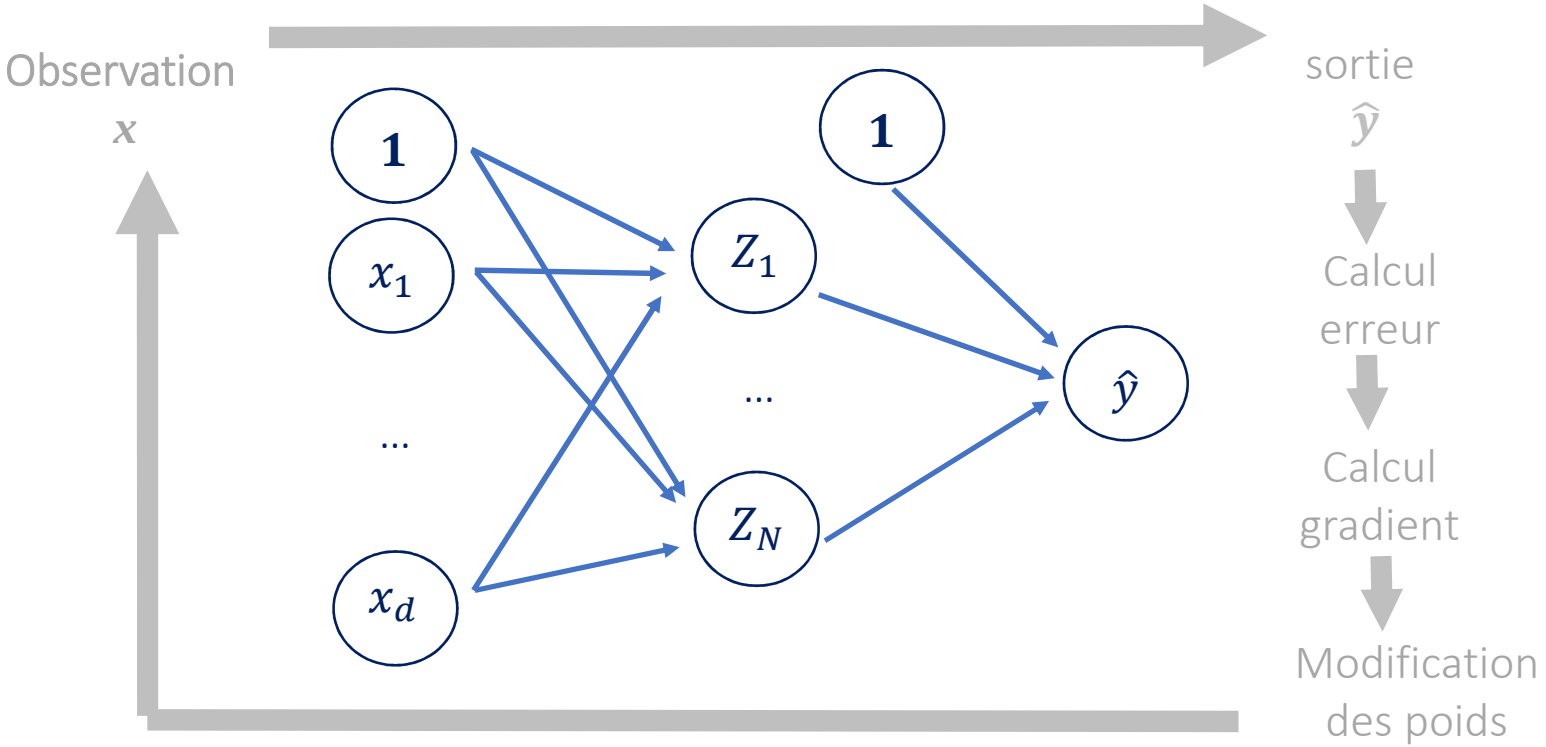
Ajustement d'un réseau de neurones

Les paramètres inconnus d'un réseau de neurones sont les poids (coefficients des combinaisons linéaires).

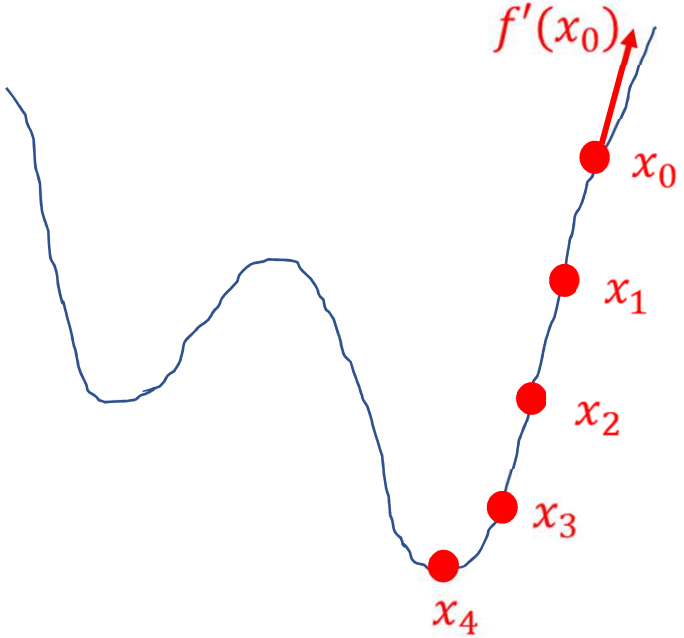
GPT3 = 175 milliards de poids!

Les poids sont déterminés en minimisant une erreur entre ce que prédit le réseau et la classe observées des observations de la base d'apprentissage.

Algorithme de rétro-propagation du gradient



Descente du gradient



Algorithme pour minimiser une fonction f

- Choisir un point au hasard x_0
- Répéter jusqu'à convergence :
  - Calculer la pente (gradient) $f'(x_0)$
  - Avancer dans le sens opposé à la pente

$$x_1 = x_0 - \alpha * f'(x_0)$$

où α est le taux d'apprentissage

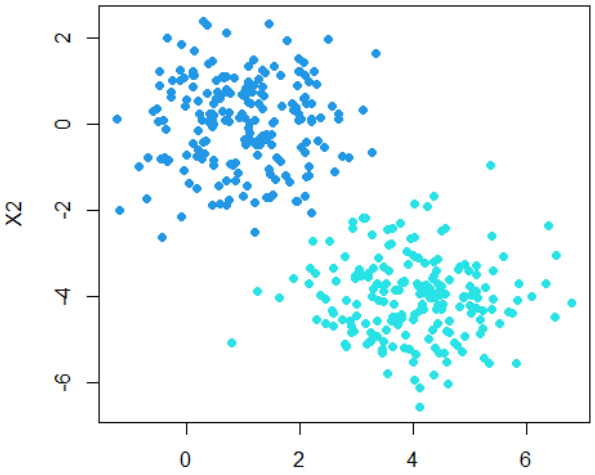
La convergence de l'algorithme dépend :

- du taux d'apprentissage α
- de l'initialisation aléatoire

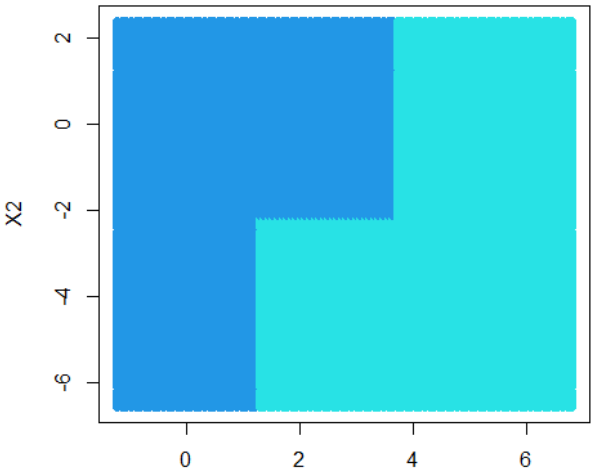
https://www.charlesbordet.com/fr/gradient-descent/#

Quelles frontières avec un réseau de neurones?

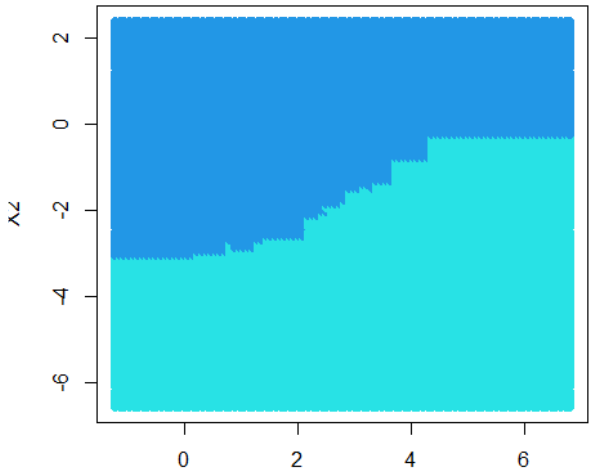
Dataset



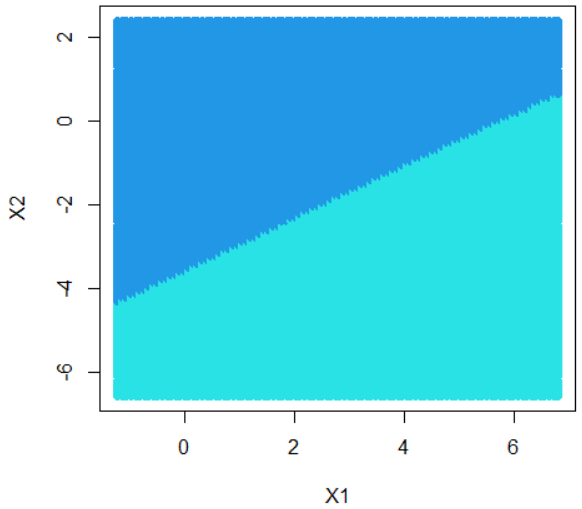
Arbre



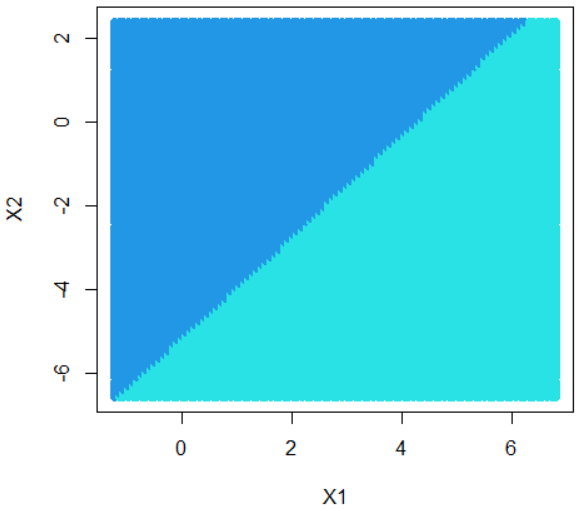
Forêt



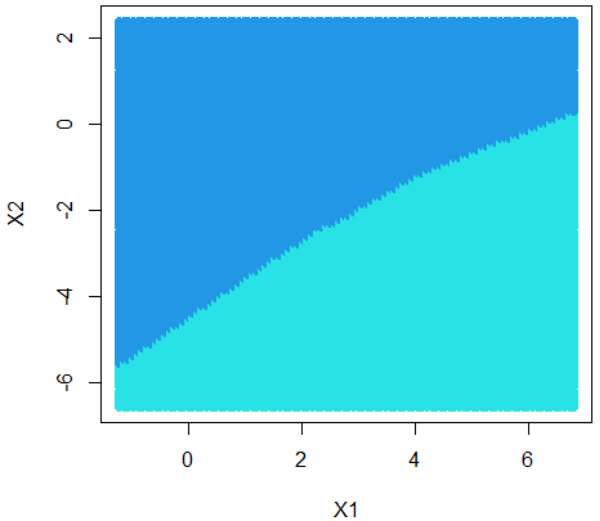
2 neurones



5 neurones

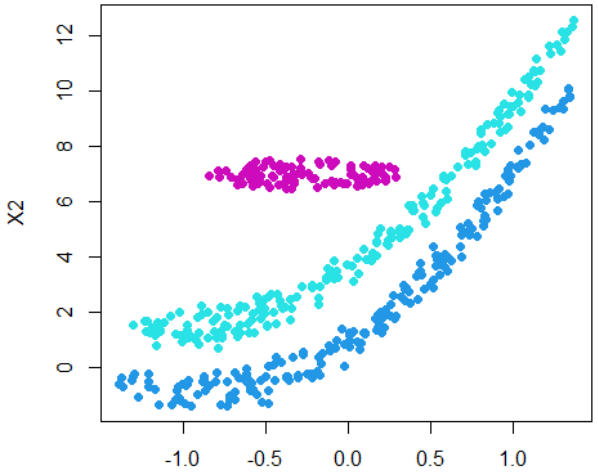


10 neurones

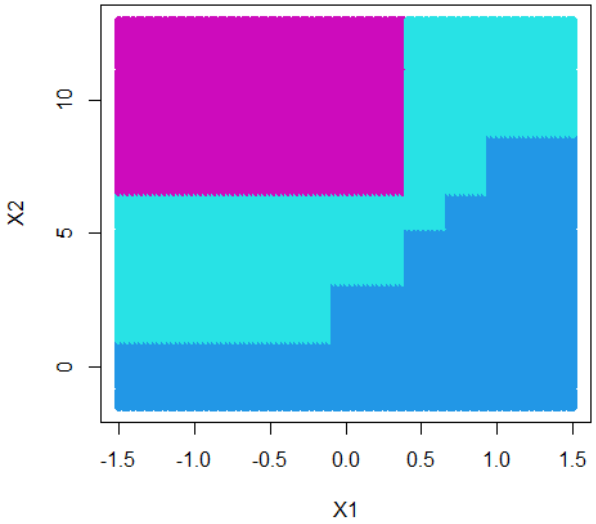


Quelles frontières avec un réseau de neurones?

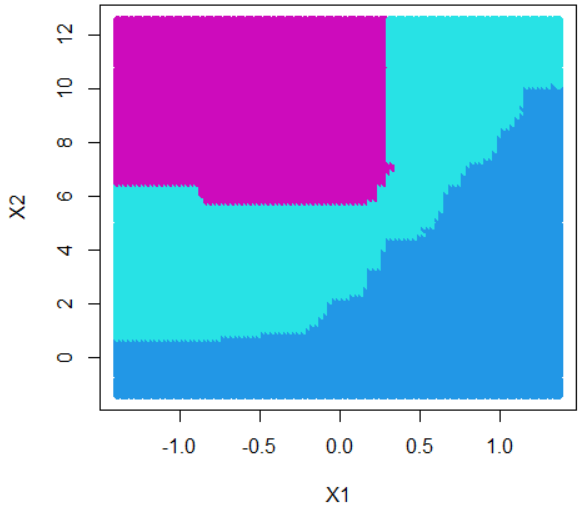
Dataset



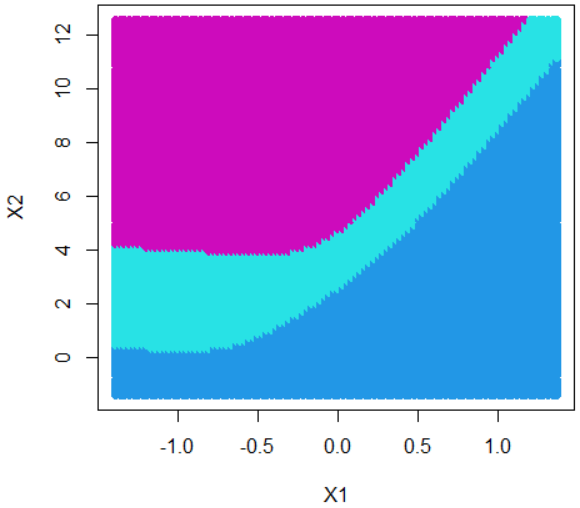
Arbre



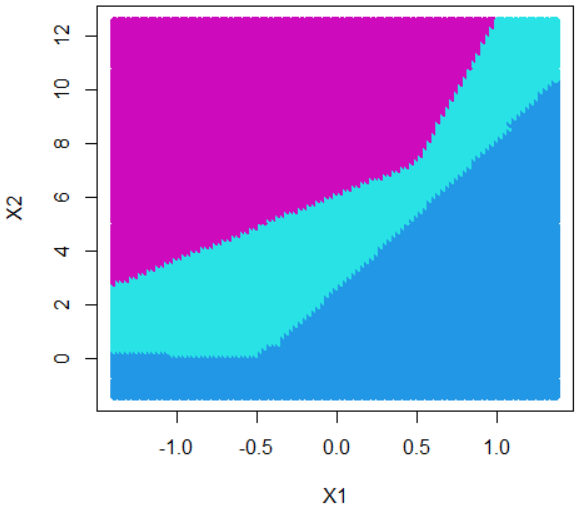
Forêt



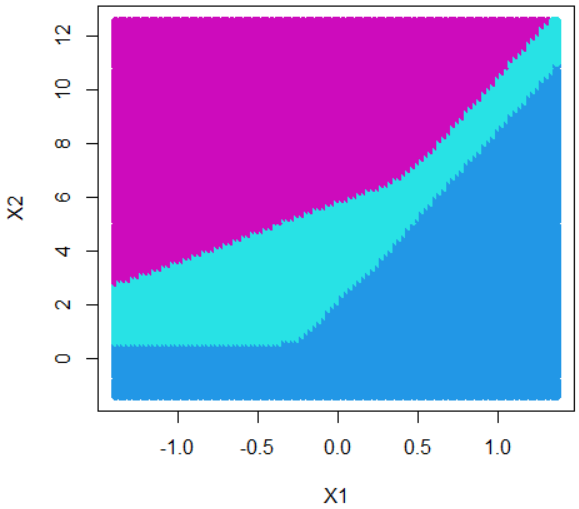
2 neurones



5 neurones

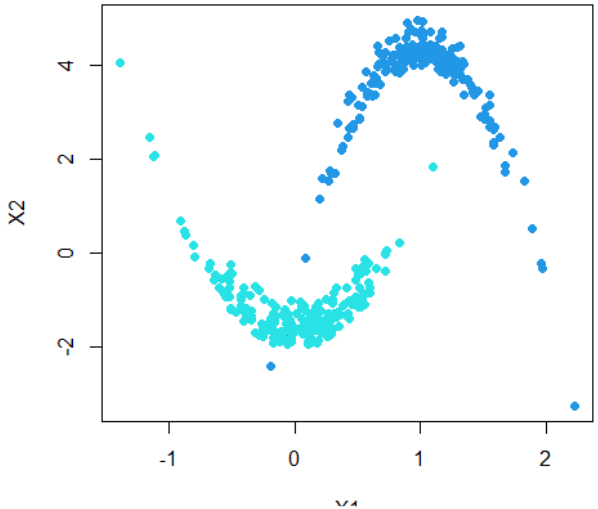


10 neurones

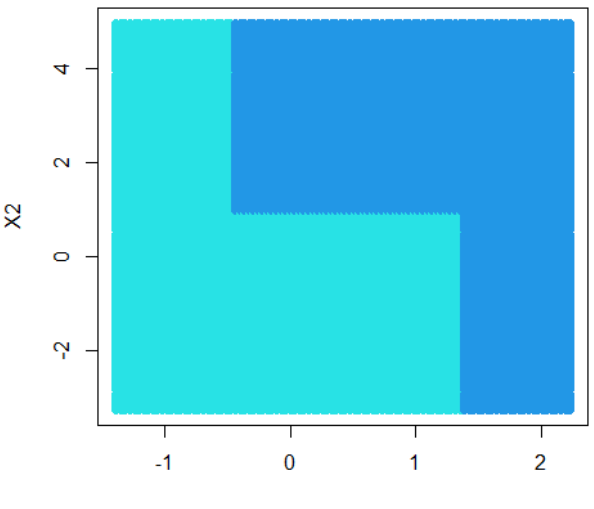


Quelles frontières avec un réseau de neurones?

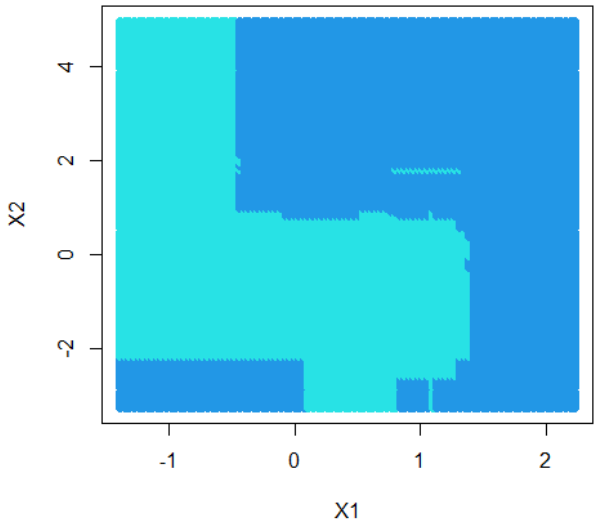
Dataset



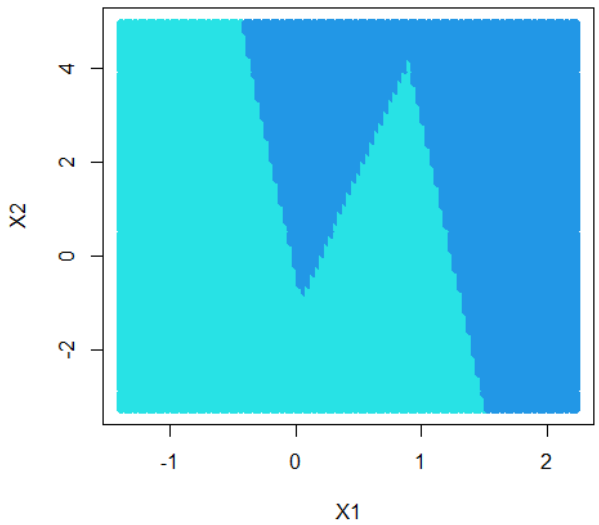
Arbre



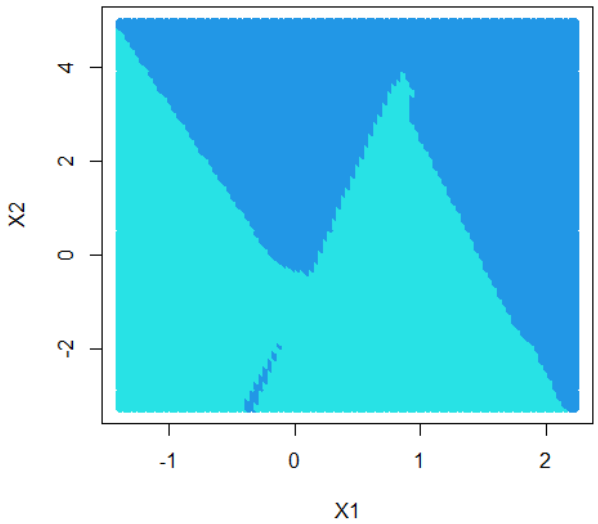
Forêt



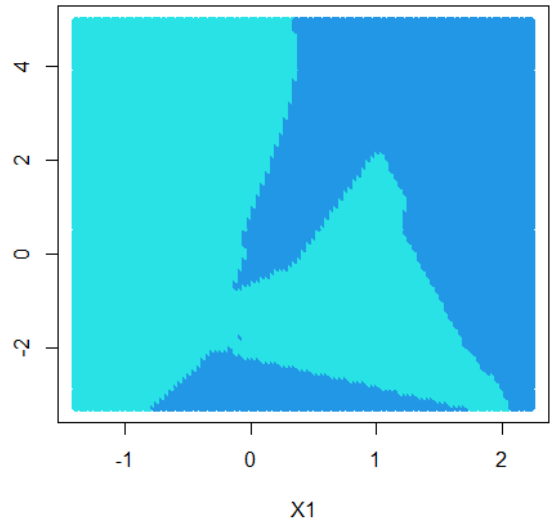
2 neurones



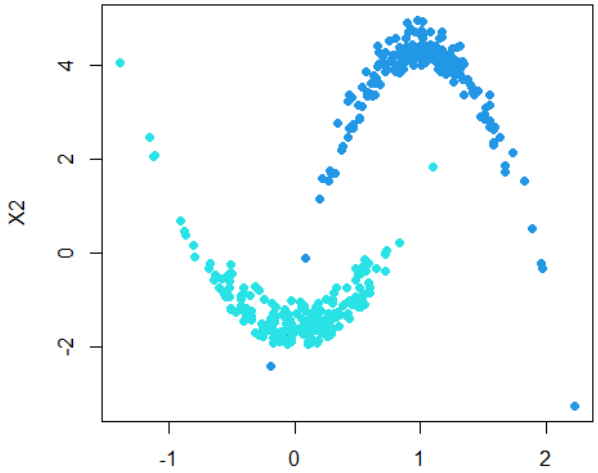
5 neurones



10 neurones



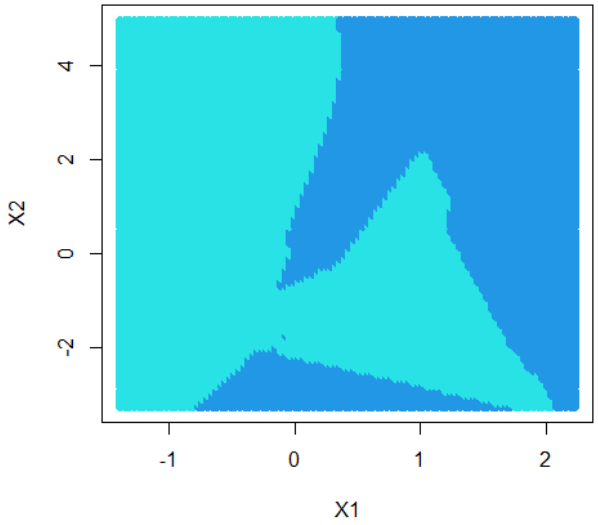
Effet de l'initialisation aléatoire

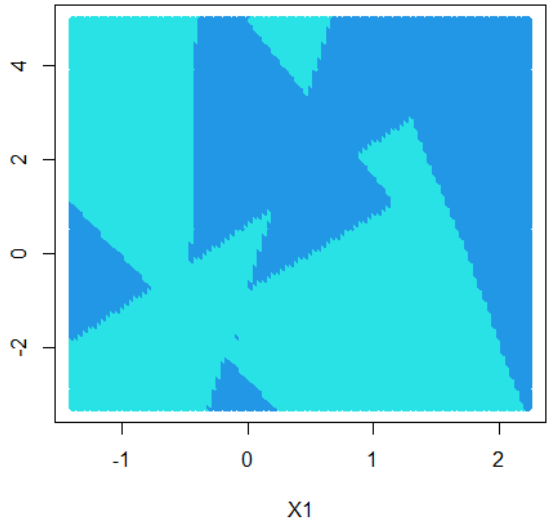


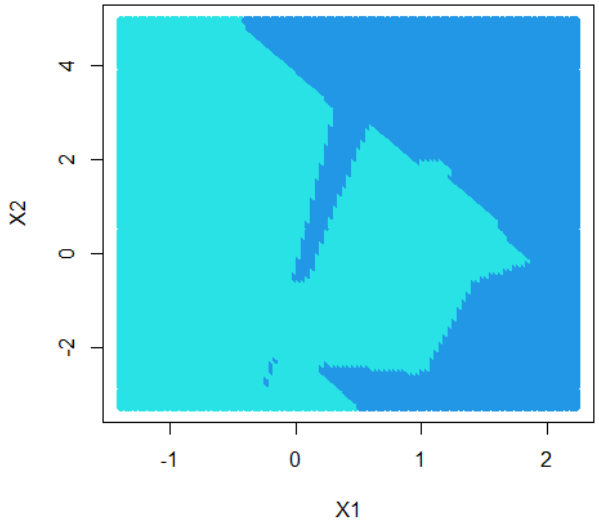
Le résultat de l'algorithme d'apprentissage d'un réseau de neurones est dépendant de l'initialisation aléatoire des poids.

⇒ Plusieurs restart

Résultats avec 10 neurones





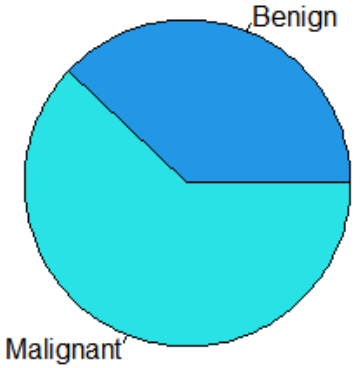


Application aux données sur le cancer de la prostate

100 patients

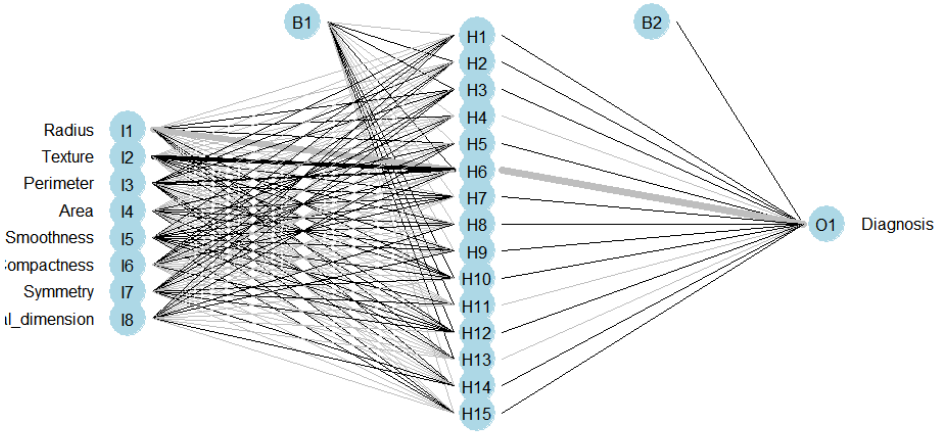
2/3 → Base d'apprentissage
66 patients

1/3 → Base de validation
34 patients



Variable cible :
Diagnosis

Résultats avec 5 neurones



Sur base validation

	prev.v	
	Benign	Malignant
Benign	11	3
Malignant	5	15

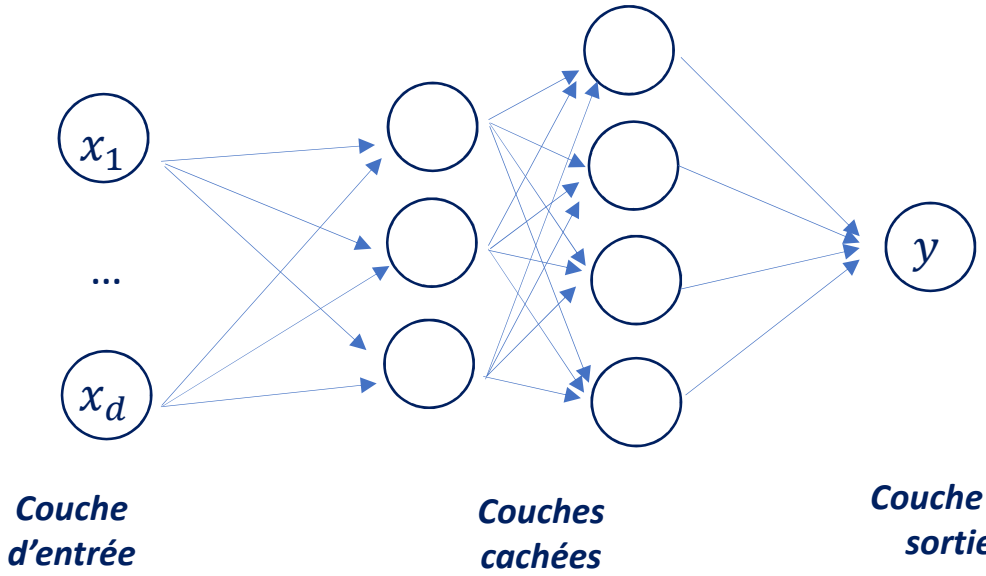
Le modèle commet 23% d'erreur sur les nouvelles données.

Pas d'interprétation possible

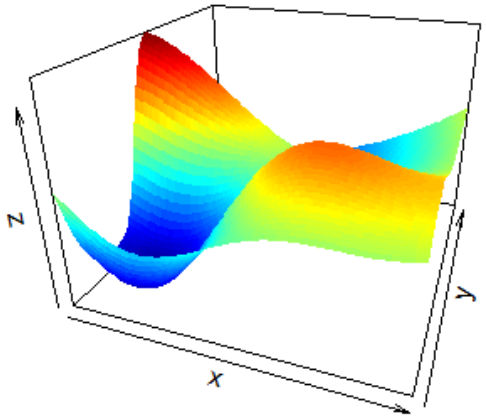
Réseaux de neurones pour la régression

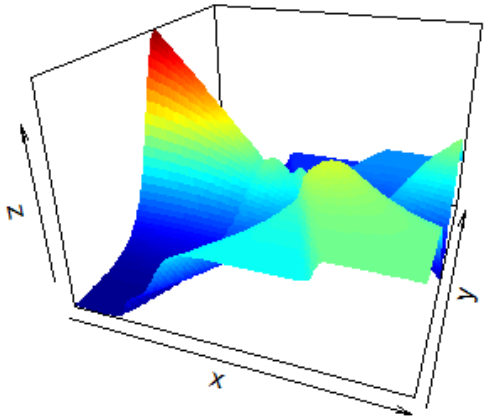
Pour utiliser un réseau de neurones en régression, on change la fonction de seuil de la couche de sortie.

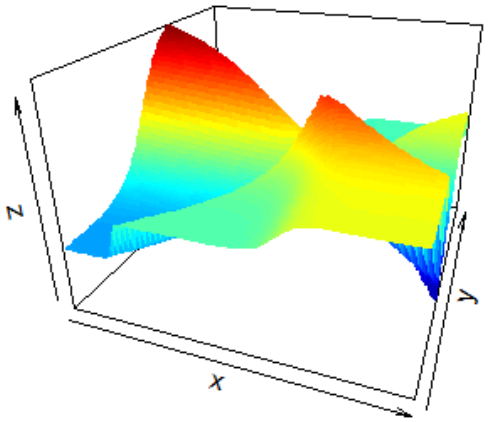
Les poids sont déterminés pour minimiser l'erreur quadratique moyenne.



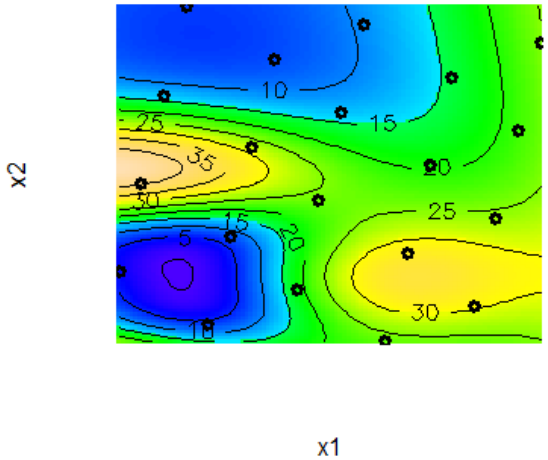
Exemple en 2D



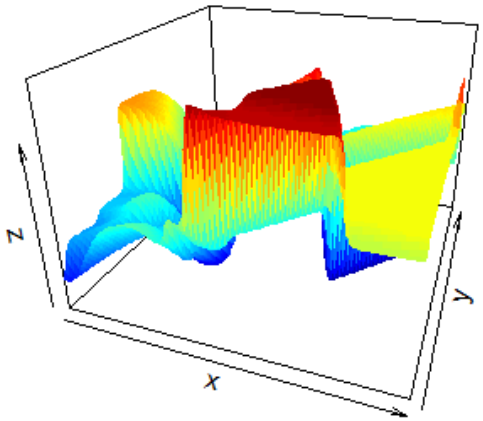


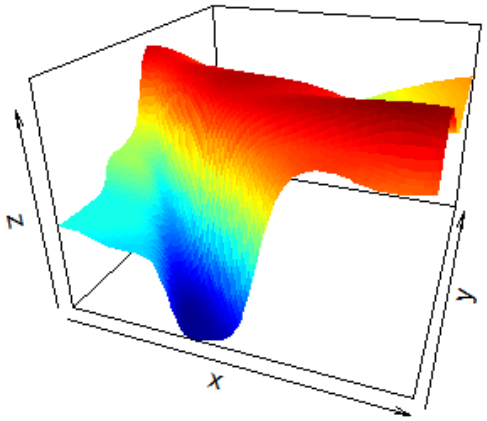


Surfaces obtenues avec 5 neurones



Vraie surface échantillonnée avec un SDF de 20 points





Surfaces obtenues avec 10 neurones

Bilan des réseaux de neurones

Le réseau de neurones peut produire des frontières très complexes mais :

- Les résultats sont sensibles à la base d'apprentissage et à l'initialisation des poids.
- Le modèle n'est pas interprétable.
- Il y a beaucoup d'hyperparamètres à gérer
  - Structure du réseau (nombre de couches, nombre de neurones, fonctions d'activation,…)
  - Algorithme d'optimisation (taux d'apprentissage, taille des batch, nombre d'epochs,…)

Obligatoirement centrer et réduire les variables

En conclusion, les alternatives au modèle polynomial permettent de répondre à des problèmes complexes mais avec plus d'observations.

Merci pour votre écoute attentive.

Je vous invite à passer à la pratique avec R ou Python.