Fouille de Données et Media Sociaux
Cours 3
Méthodes d'ensemble
Master DAC Data Science
UPMC - LIP6

Ludovic Denoyer

28 septembre 2015

Rappel (ou pas) sur les arbres de décision

Contexte

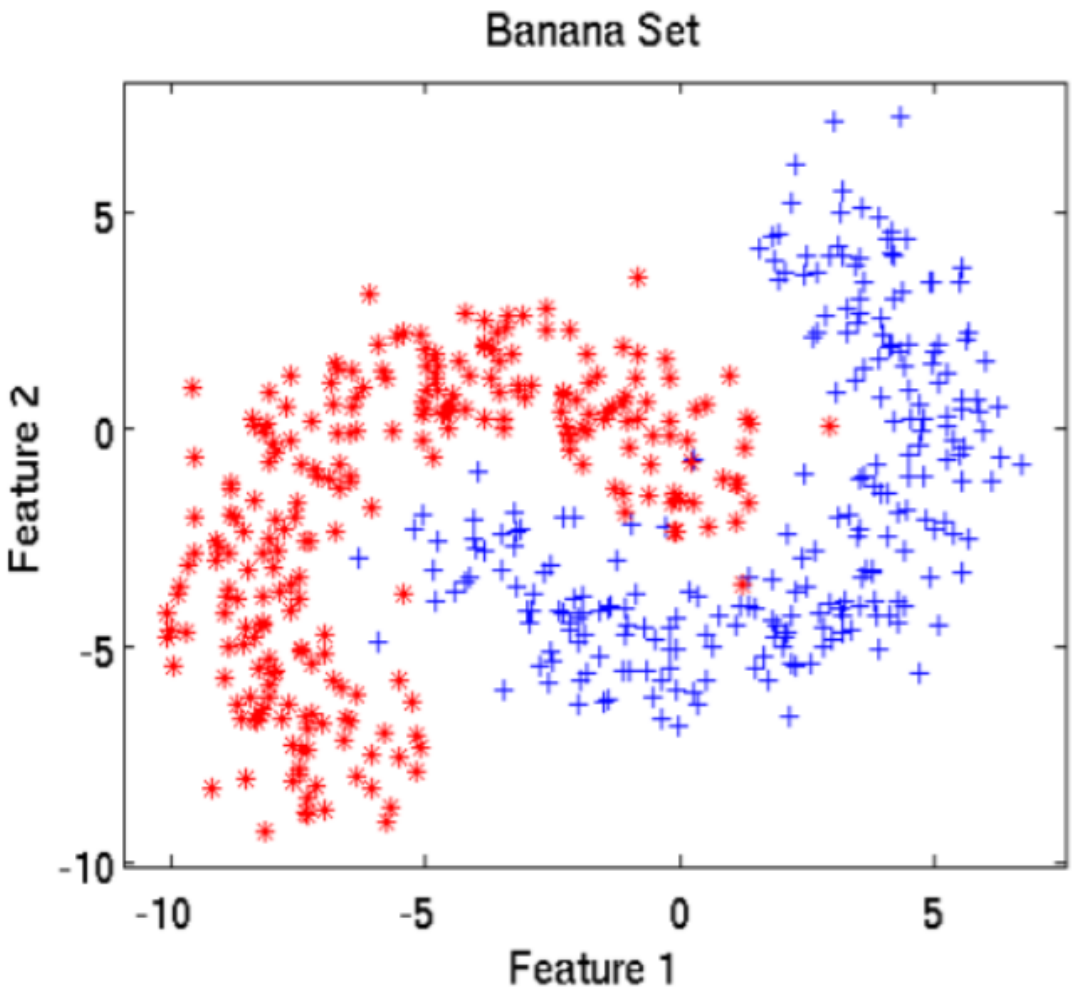
- Données numériques dans $\mathbb{R}^N$
- Sortie $=$ 1 classe parmi K
- Problème d'apprentissage : trouver la fonction $f_\theta(x)$ qui associe une catégorie à un vecteur x
- **Classification binaire :** $K = 2$
  - Classification de *spam*
- Multiples modèles

Rappel (ou pas) sur les arbres de décision

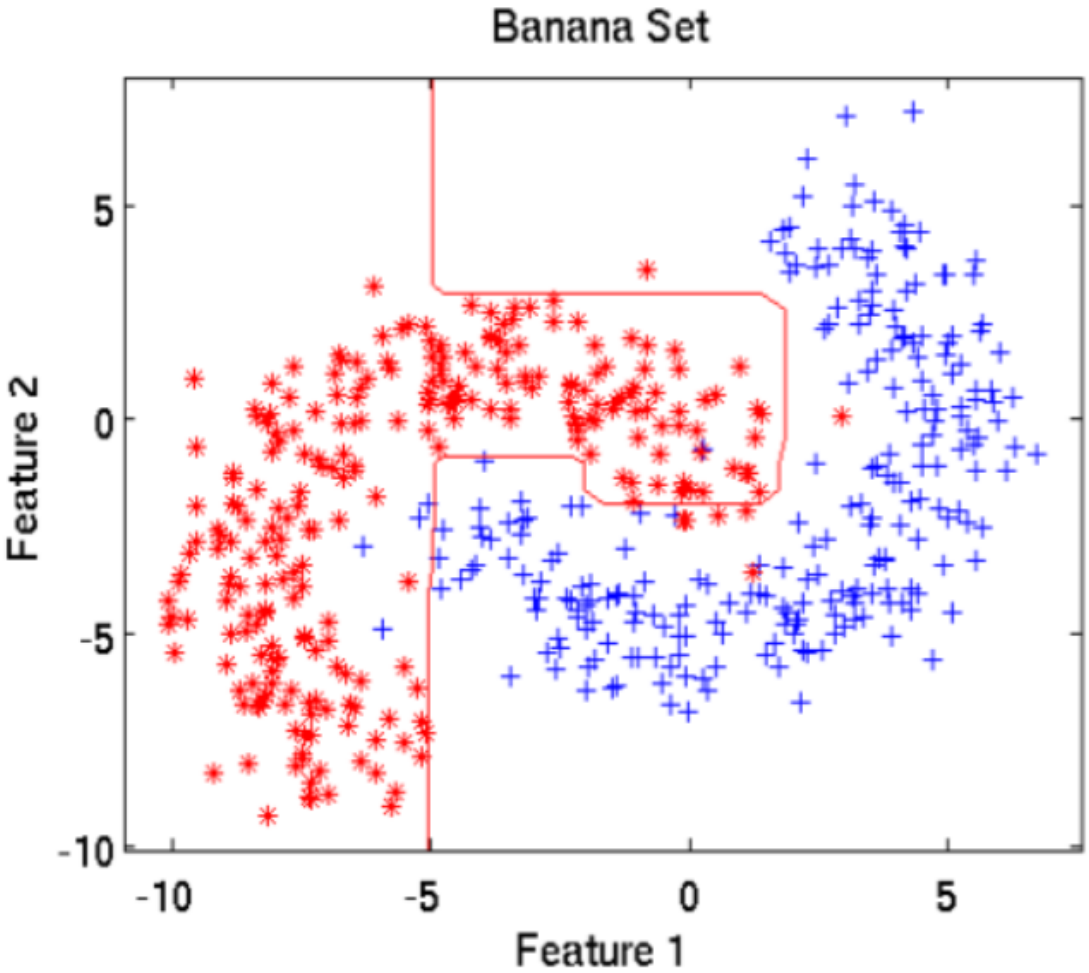
Arbres de décision

- Le modèle est représenté par un ensemble de décisions binaires organisées sous forme d'arbre
- Chaque noeud de l'abre est un **test sur une features** :$x_3 > 0.6$
- Les feuilles sont des décisions - dans $1, 2, ..., K$
- Le noeud est construit de la racine vers les feuilles par l'utilisation d'un algorithme itératif
  - Pleins d'algorithmes existent - CART, C4.5, ID3, ...
  - Ils diffèrent principalement par le critère de **split** qui permet la création des noeuds
- L'arbre est souvent élagé (**pruning**) afin d'améliorer sa performance en généralisation
- L'inférence est faite par la "descente" de l'exemple dans l'arbre

Rappel (ou pas) sur les arbres de décision



Rappel (ou pas) sur les arbres de décision

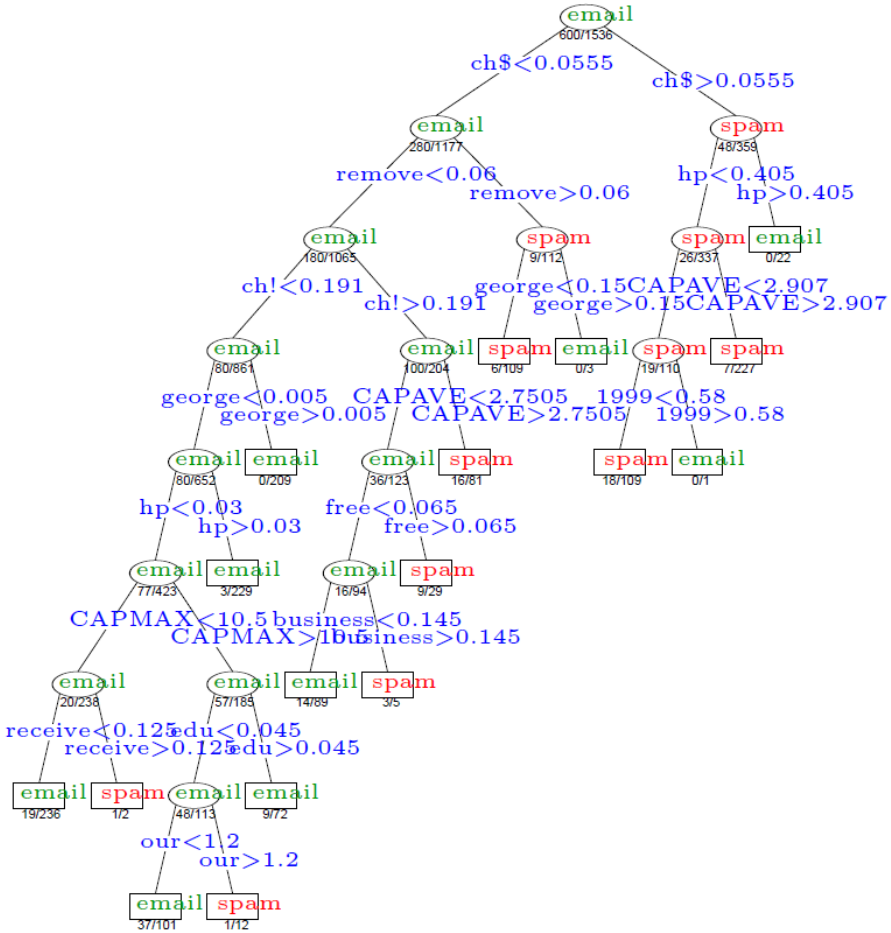


Rappel (ou pas) sur les arbres de décision

Avantages

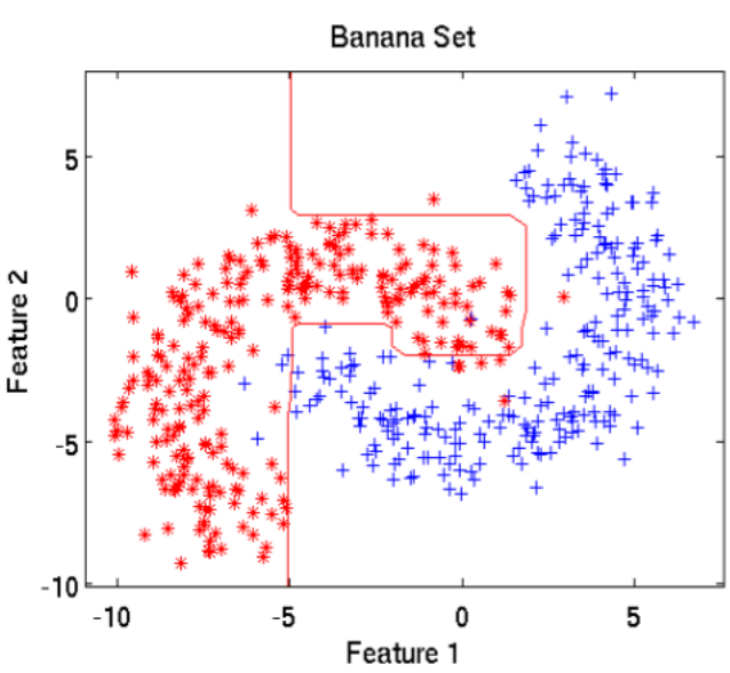
- Capable de digérer de grands jeux de données
- Interprétables
- Autres avantages : variables manquantes, variables redondantes, entrées qualitatives et quantitatives

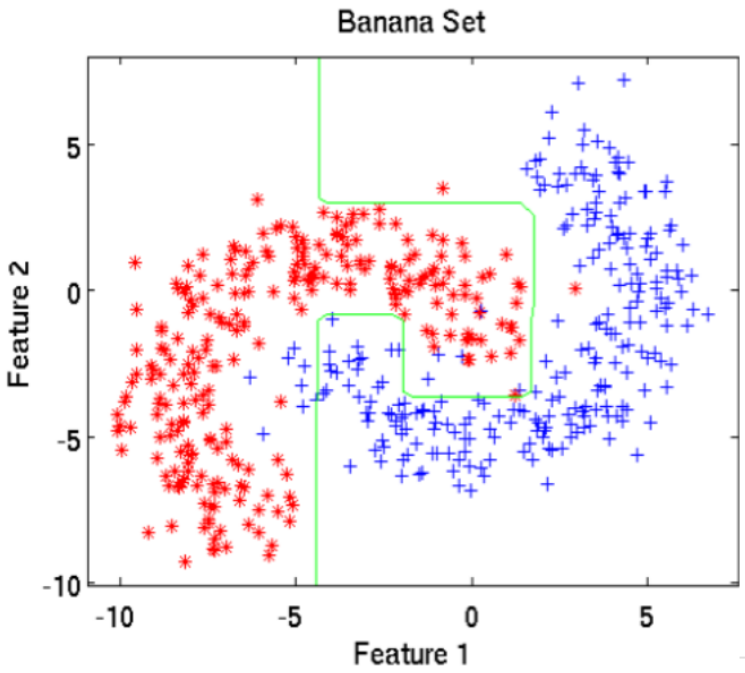
Rappel (ou pas) sur les arbres de décision



Inconvénient : Pas si interprétable que cela

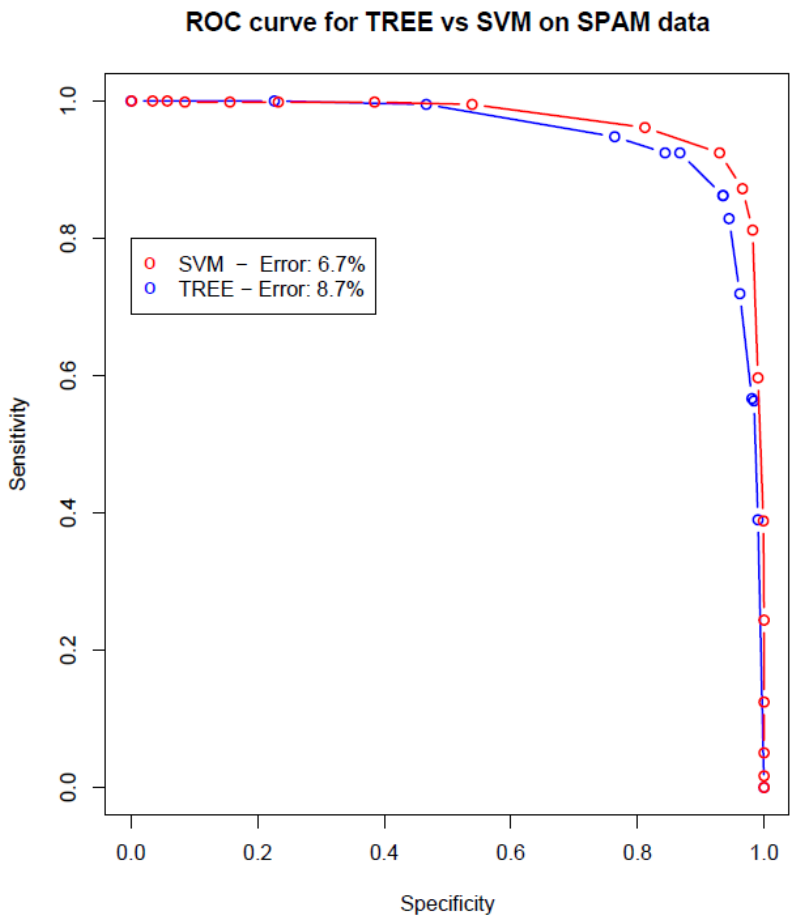
Rappel (ou pas) sur les arbres de décision





Inconvénient : Instable

Rappel (ou pas) sur les arbres de décision

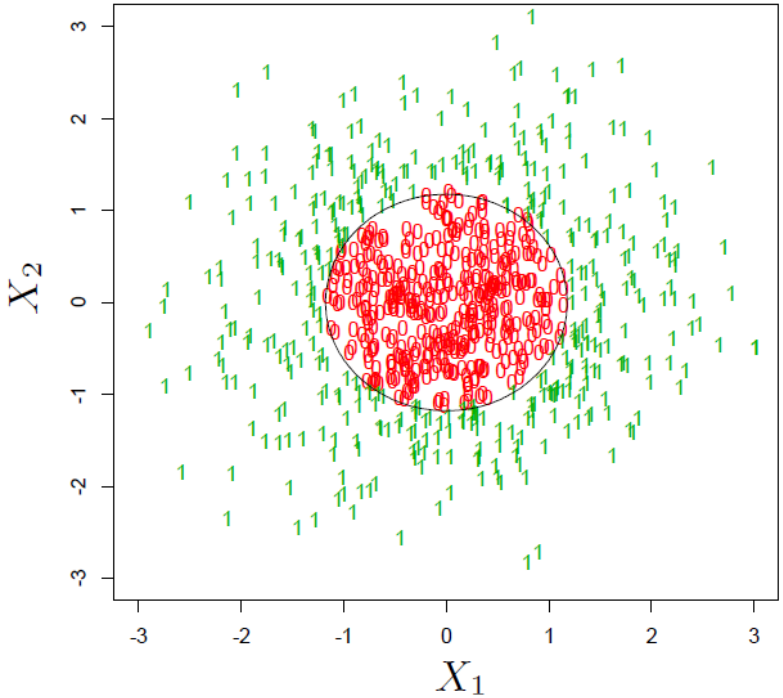


Inconvénient : Pas toujours très performant

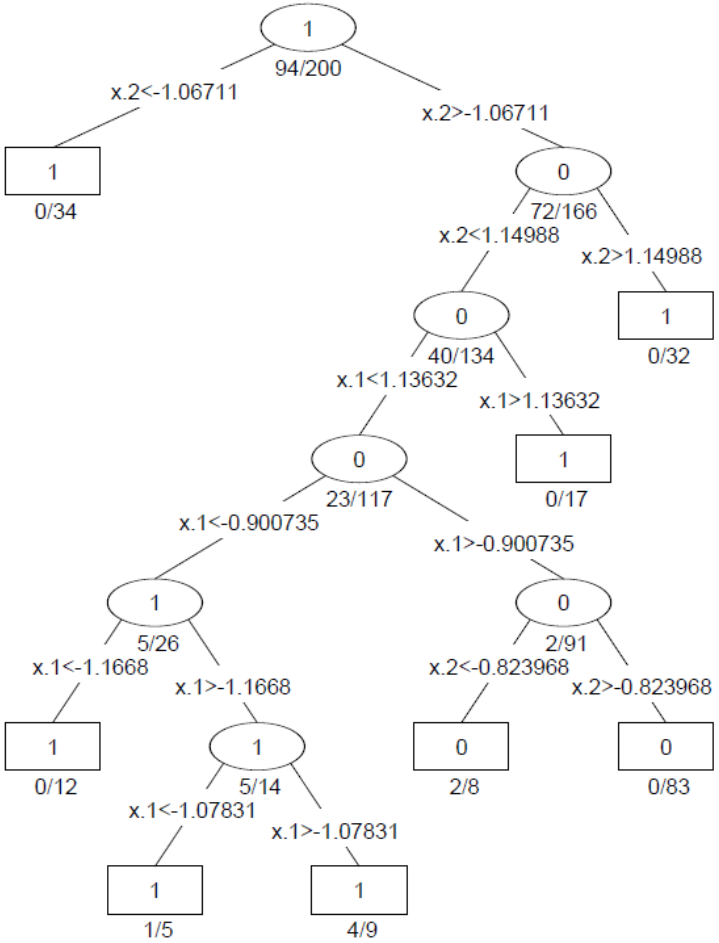
Rappel (ou pas) sur les arbres de décision

Sphères imbriquées

- Deux sphère l'une dans l'autre, en n dimension
- Sans bruit
- $\Rightarrow$ Erreur de Bayes $= 0$
- Rappel : l'erreur de Bayes est l'erreur minimale pour une certaine classe de classifieur



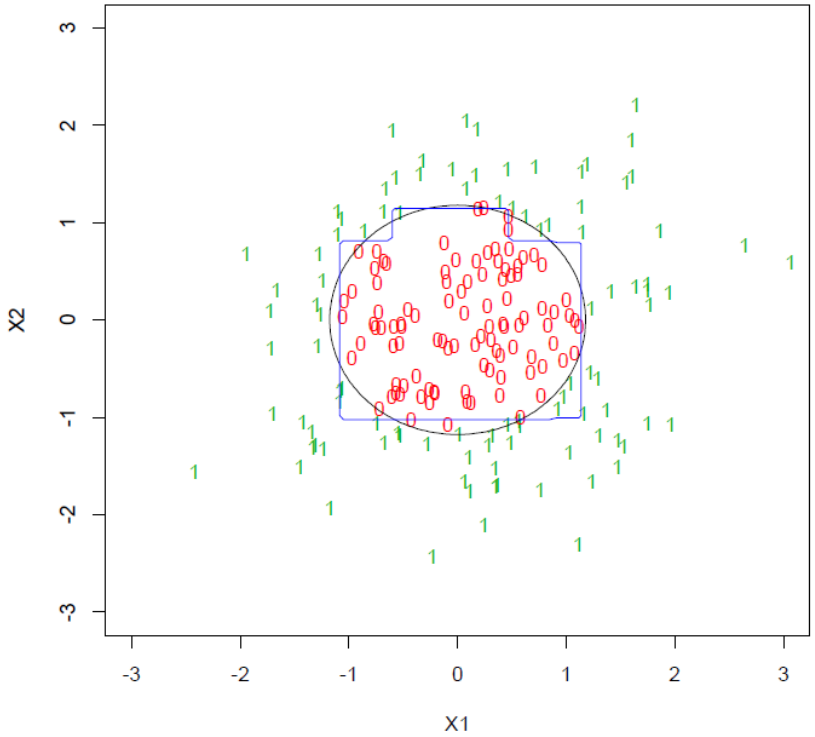
Rappel (ou pas) sur les arbres de décision



Rappel (ou pas) sur les arbres de décision

Sphères imbriquées

- Dans le cas où $n = 10$, les arbres de décision produisent une erreur > 0.3



Décomposition Biais-Variance

Biais

Le **Biais** mesure la qualité d'un prédicteur. Un grand biais signifie que le modèle n'est pas performant, et provient souvent du fait de mauvaises hypothèses dans la classe de fonctions utilisées. Haut bias $\Rightarrow$ sous-apprentissage

Variance

La **Variance** mesure la sensibilité du classifieur à de petites fluctuations dans l'ensemble d'apprentissage. Une variance élevé signifie que le système généralise mal (sur-apprentissage)

Compromis Biais-Variance

Idéalement, on souhaite minimiser les deux simultanément. **Mais** il y a un compromis à trouver ! $\Rightarrow$ problème clef de l'apprentissage statistique

Décomposition Biais-Variance

- On considère un ensemble de points d'apprentissage $S = x^1, ..., x^n$ et les sorties associées y^i.
- On considère la relation suivante $y = f_\theta(x) + \epsilon$ où ϵ est un bruit gaussien de moyenne 0 et de variance σ^2
- On cherche la fonction $\hat{f}$ qui approxime f au sens de l'erreur des moindres carrées $(y - \hat{f}(x))^2$
- Etant donné un nouveau point x^*, y^*, nous voudrions comprendre le comporetement du modèle sur ce nouveau point.
- Si l'on considère que S est tiré selon une loi $P(S)$, nous allons calculer la valeur suivante :

$$E_P[(y^* - \hat{f}(x^*))^2] \tag{1}$$

Décomposition Biais-Variance

- On considère un ensemble de points d'apprentissage $S = x^1, ..., x^n$ et les sorties associées y^i.
- On considère la relation suivante $y = f_\theta(x) + \epsilon$ où ϵ est un bruit gaussien de moyenne 0 et de variance σ^2
- On cherche la fonction $\hat{f}$ qui approxime f au sens de l'erreur des moindres carrées $(y - \hat{f}(x))^2$
- Etant donné un nouveau point x^*, y^*, nous voudrions comprendre le comporetement du modèle sur ce nouveau point.
- Si l'on considère que S est tiré selon une loi $P(S)$, nous allons calculer la valeur suivante :

$$E_P[(y^* - \hat{f}(x^*))^2] \tag{2}$$

Décomposition Biais-Variance

Soit Z une variable aléatoire et $\bar{Z} = E_P[Z]$ sa moyenne. On a :

$$\begin{aligned} E[(Z-\bar{Z})^2] &= E[Z^2 - 2Z\bar{Z} + \bar{Z}^2] \\ &= E[Z^2] - 2E[Z]\bar{Z} + \bar{Z}^2 \\ &= E[Z^2] - 2\bar{Z}^2 + \bar{Z}^2 \\ &= E[Z^2] - \bar{Z}^2 \end{aligned} \tag{3}$$

Et donc $E[Z^2] = E[(Z-\bar{Z})^2] + \bar{Z}^2$

Décomposition Biais-Variance

$$
\begin{aligned}
E[((\hat{f}(x^*) - y^*)^2] &= E[\hat{f}(x^*)^2 - 2\hat{f}(x^*)y^* + y^{*2}] \\
&= E[\hat{f}(x^*)^2] - 2E[\hat{f}(x^*)]E[y^*] + E[y^{*2}] \\
&= E[(\hat{f}(x^*) - \bar{\hat{f}}(x^*))^2] + \bar{\hat{f}}(x^*))^2 \\
&- 2\bar{\hat{f}}(x^*)f(x^*) \\
&+ E[(y^* - f(x^*))^2] + f(x^*)^2 \\
&= E[(\hat{f}(x^*) - \bar{\hat{f}}(x^*))^2] + (\bar{\hat{f}}(x^*) - f(x^*))^2 \\
&+ E[(y^* - f(x^*))^2]
\end{aligned} \tag{4}
$$

$$= Variance(\hat{f}(x^*)) + Biais(\hat{f}(x^*))^2 + Bruit\ (\sigma^2)$$

Décomposition Biais-Variance

Comment mesurer le bias et la variance ?

- Un seul S disponible
- On peut simuler plusieurs S par bootstraping = tirage avec remise

Bagging

Bagging

Bagging = Bootstrap AGGregatING est une procédure de moyennage qui permet de réduire la variance d'un modèle de Machine Learning

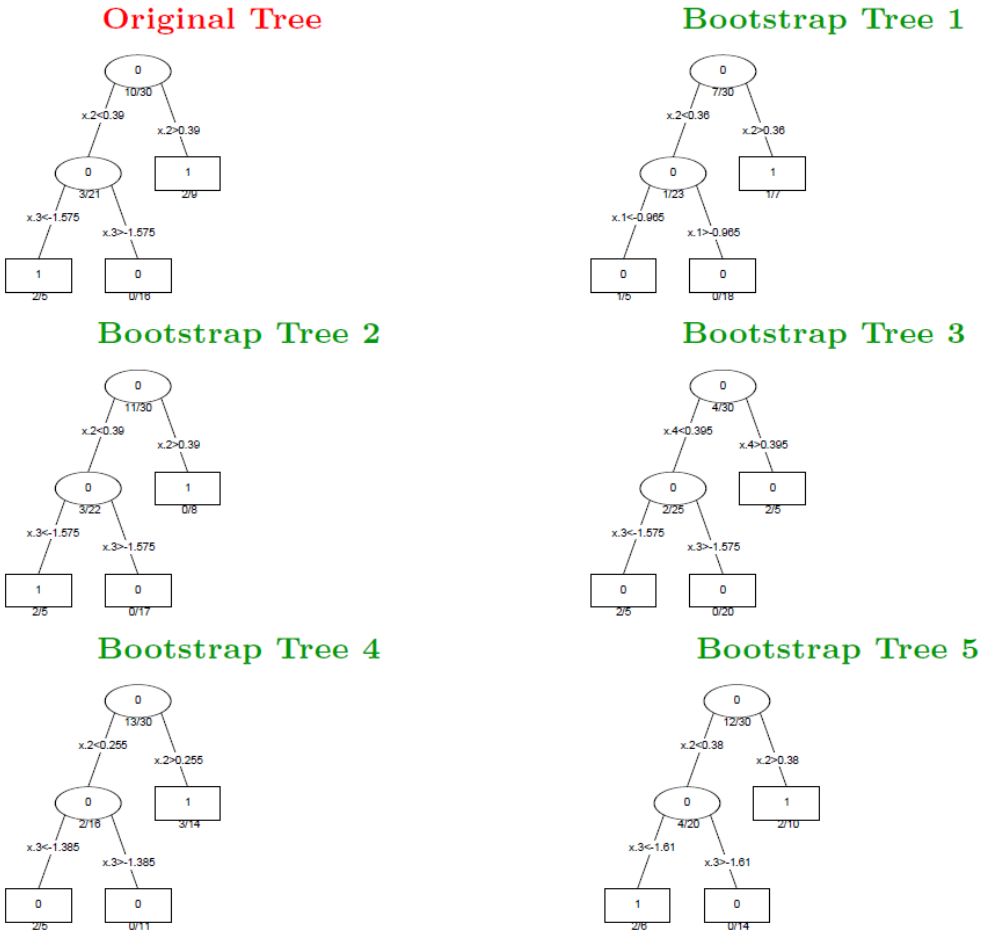
Combinaisons de modèles

Soit $\hat{f}_1, ..., \hat{f}_B$ un ensemble de modèles, on peut constuire un modèle aggrégé par :

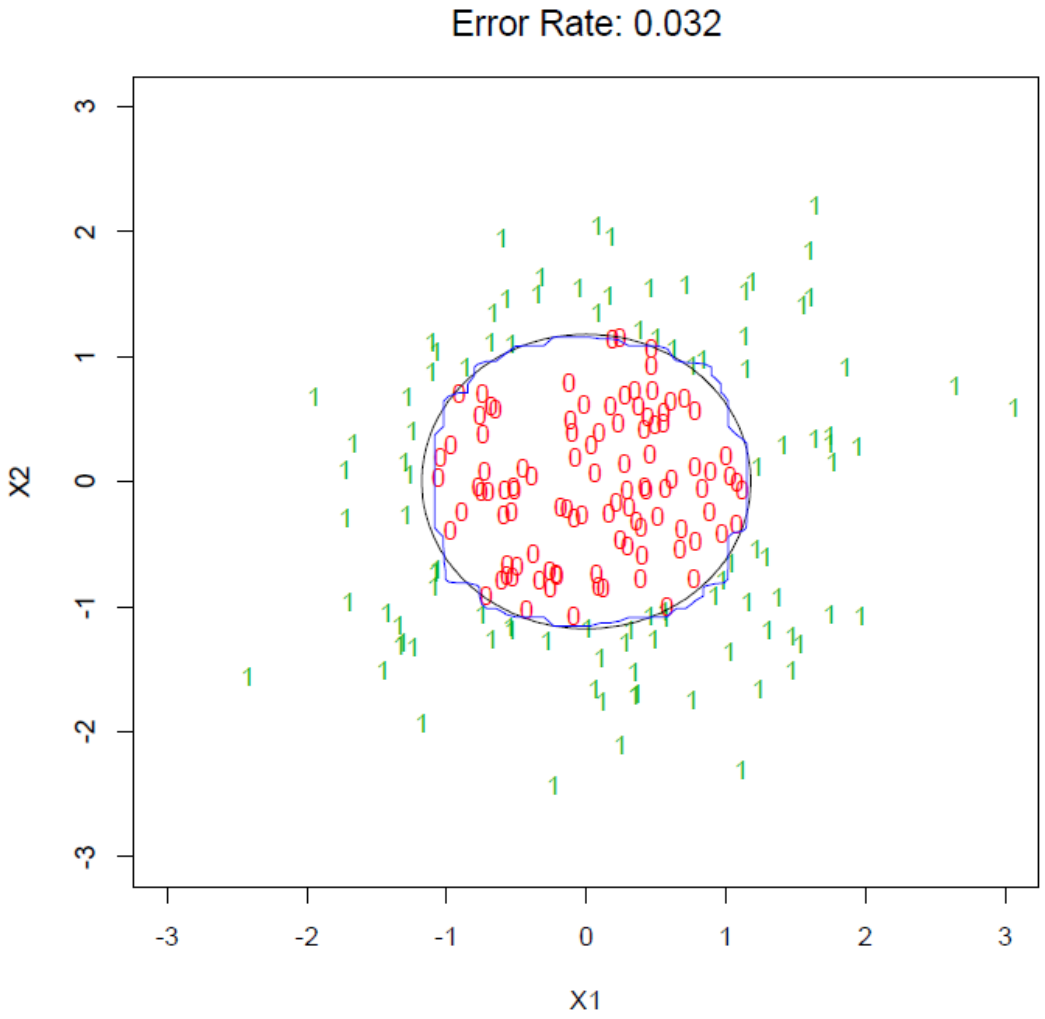
- Moyenne des prédiction des modèles (régression)
- Vote majoritaire (classification)

Dans le Bagging, les différents modèles sont construits par apprentissage sur un ensemble de B ensemble d'apprentissage générés en suivant la procédure du bootstrap.

Bagging


Original Tree
0
10/30
x.2<0.39
x.2>0.39
0
3/21
1
2/9
x.3<-1.575
x.3>-1.575
1
2/5
0
0/16
Bootstrap Tree 1
0
7/30
x.2<0.36
x.2>0.36
0
1/23
1
1/7
x.1<-0.965
x.1>-0.965
0
1/5
0
0/18
Bootstrap Tree 2
0
11/30
x.2<0.39
x.2>0.39
0
3/22
1
0/8
x.3<-1.575
x.3>-1.575
1
2/5
0
0/17
Bootstrap Tree 3
0
4/30
x.4<0.395
x.4>0.395
0
2/25
0
2/5
x.3<-1.575
x.3>-1.575
0
2/5
0
0/20
Bootstrap Tree 4
0
13/30
x.2<0.255
x.2>0.255
0
2/16
1
3/14
x.3<-1.385
x.3>-1.385
0
2/5
0
0/11
Bootstrap Tree 5
0
12/30
x.2<0.38
x.2>0.38
0
4/20
1
2/10
x.3<-1.61
x.3>-1.61
1
2/6
0
0/14

Bagging



La frontière de décision est plus lisse

Bagging

Si l'on considère que le modèle appris par Bagging est :

$$\hat{f}(x) = \frac{1}{B} \sum_i \hat{f}_i(x) \tag{5}$$

Sachant que :

- Biais $= (\bar{\hat{f}}(x^*) - f(x^*))^2$
- Variance $= E[(\hat{f}(x^*) - \bar{\hat{f}}(x^*))^2]$

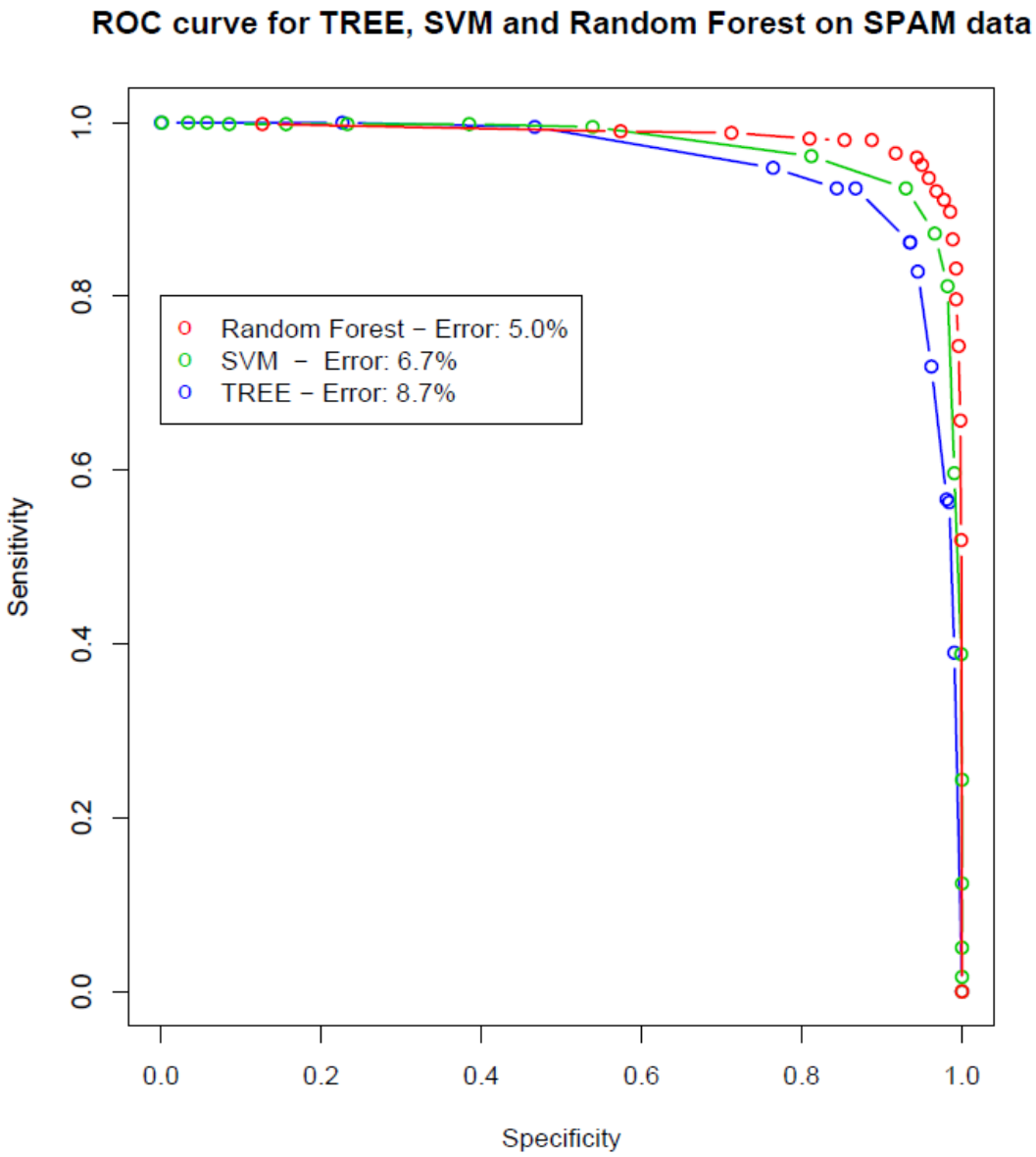
Le Bagging supprime la **variance** d'un classifieur. En réalité, le Bagging réduit la variance, et augmente le biais légèrement.

Fôrêts aléatoires

Random Forests

- Bagging d'arbres de décision
- At chaque *split*, a échantillon aléatoire de m features est tiré (décorrélation des arbres). Typiquement, $m = \sqrt{n}$ ou $\log_2(n)$.
- Chaque arbre est appris sur un *bootstrap* de l'échantillon original. L'erreur est évaluées sur les points qui n'ont pas été pris dans les échantillons samplés

Fôrêts aléatoires



Boosting

Définition

Le boosting est un domaine de l'apprentissage automatique (branche de l'intelligence artificielle). C'est un principe qui regroupe de nombreux algorithmes qui s'appuient sur des ensembles de classifieurs binaires : le boosting optimise leurs performances. Le principe est issu de la combinaison de classifieurs (appelés également hypothèses).
wikipedia

Schapire a montré que la combinaisons de *weak classifiers* (acccuracy $> 50\%$) permet d'obtenir un classifieur très performant.

Idée

- Apprentissage succesif de modèles
- Pondération des exemples d'apprentissage :
  - Points bien prédits $\Rightarrow$ poids faible
  - Points mal prédits $\Rightarrow$ poids fort
- Focalisation sur les parties de l'espace mal prédits.

Boosting

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in Y = \{-1, +1\}$
Initialize $D_1(i) = 1/m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : X \to \{-1, +1\}$ with error

$$\epsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1 - \epsilon_t}{\epsilon_t}\right)$.
- Update:

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases} \\ &= \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t} \end{aligned}$$

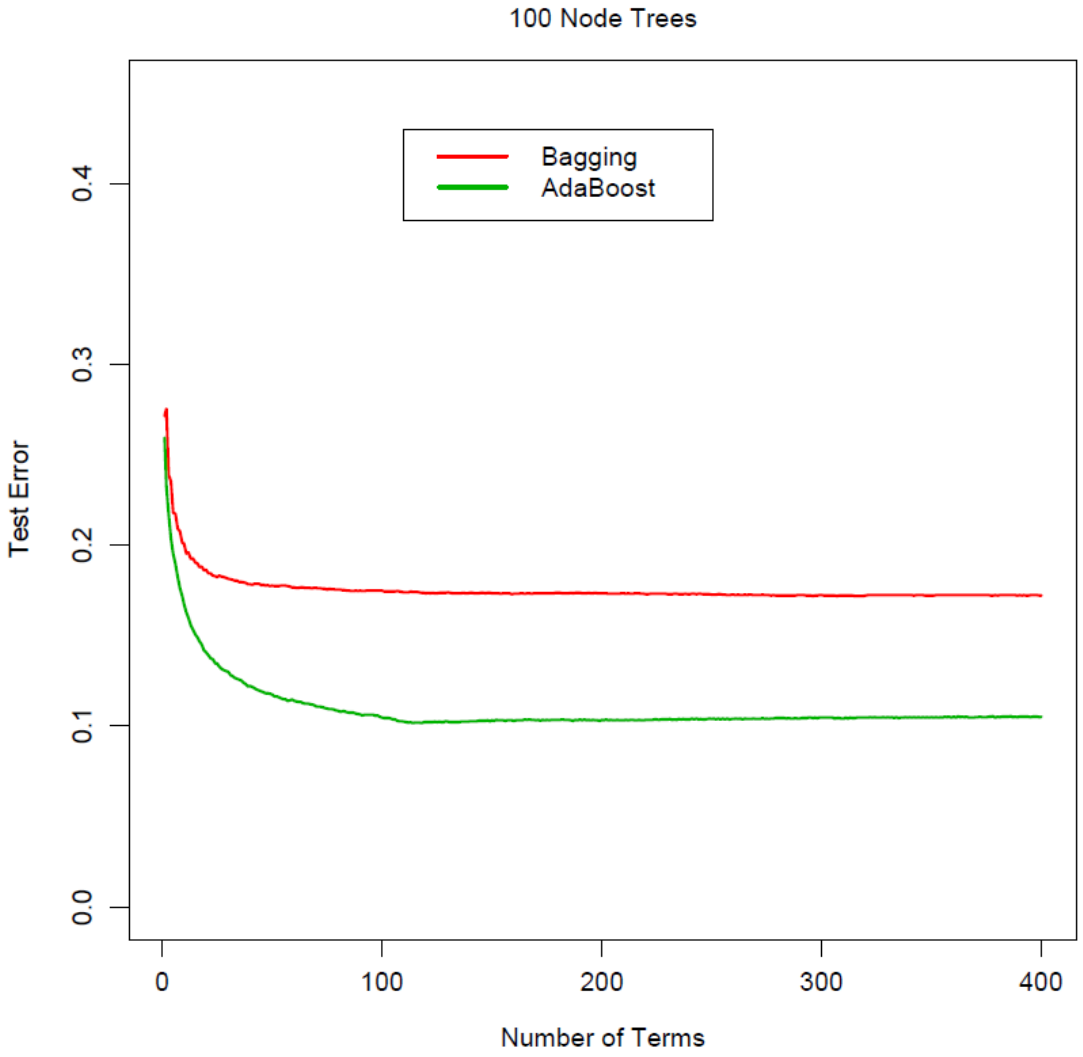
where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

Figure 1: The boosting algorithm AdaBoost.

Boosting


100 Node Trees
Bagging
AdaBoost
Test Error
0.0
0.1
0.2
0.3
0.4
0
100
200
300
400
Number of Terms

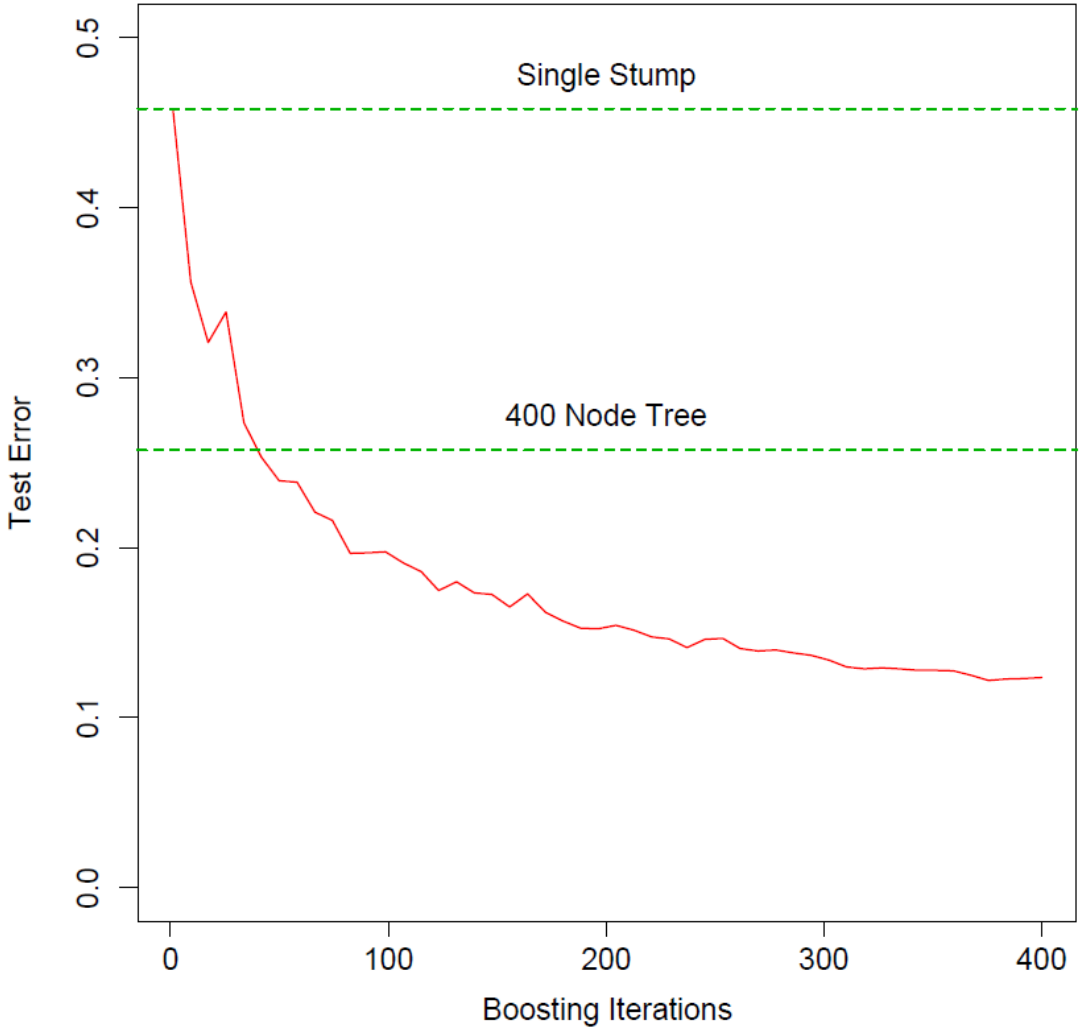
Boosting

Soit l'erreur ϵ_t de h_t, et notons $\epsilon_t = \frac{1}{2} - \gamma_t$ tel que γ_t mesure l'amélioration obtenu par l'utilisation de h_t par rapport a l'aléatoire. Freund et Schapire (1996) ont montré que :

$$\epsilon_H \leq \prod_t \left[2\sqrt{\epsilon_t(1-\epsilon_t)}\right] = \prod_t \sqrt{1-4\gamma_t^2} \leq exp\left(-2\sum_t \gamma_t^2\right)$$

Si $\gamma_t > 0$, l'erreur empirique diminue exponetiellement rapidement avec t.

Boosting - Stumps



Gradient-Boosting

- Fonction de décision : $f_t(x) = \sum_t \alpha_t h_t(x)$
- Fonction objective $L(f(x), y) = -e^{y.f(x)}$
- Minimisation du risque empirique : $R(f) = \sum_{i=1}^{N} L(f(x^i), y^i)$

Comment peut-on améliorer R ? En suivant le gradient de R !

Gradient-Boosting

$$\begin{aligned}\nabla R(f) &= (\frac{\partial R}{\partial f(x^1)}, ..., \frac{\partial R}{\partial f(x^N)}) \\ &= (-y^1 e^{-y^1 f(x^1)}, ..., -y^N e^{-y^1 f(x^N)})\end{aligned} \tag{6}$$

Pour cela, on va cherche un vecteur une fonctio h qui correspond au vecteur $h(x^1), ..., h(x^N)$ qui soit le plus dans la direction opposée à ce gradient.

$$h_{t+1} = argmax_h[\sum_{i=1}^{N} h(x^i) y^i e^{-y^i f(x^i)}] \tag{7}$$

Gradient-Boosting

Si l'on considère le cas de la classification binaire :

$$\begin{aligned}
h_{t+1} &= \text{argmax}_h \left[\sum_{i=1}^{m} \left(1 - 2\,\delta_{y_i \neq h(x_i)}\right) e^{-y_i f_t(x_i)} \right] \\
&= \text{argmax}_h \left[\left(\sum_{i=1}^{m} e^{-y_i f_t(x_i)} \right) - 2 \left(\sum_{i=1}^{m} \delta_{y_i \neq h(x_i)} e^{-y_i f_t(x_i)} \right) \right] \\
&= \text{argmax}_h \left[-\sum_{i=1}^{m} \delta_{y_i \neq h(x_i)} e^{-y_i f_t(x_i)} \right] \\
&= \text{argmin}_h \left[\sum_{i=1}^{m} \delta_{y_i \neq h(x_i)} D_{t+1}(i) \right]
\end{aligned}$$

avec $D_{t+1}(i) = e^{-y_i f(x_i)}/Z \Rightarrow$ distribution. On a :

$$h_{t+1} = argmin E_{D_{t+1}}[\delta_{y^i \neq h(x_i)}] \tag{8}$$

Gradient-Boosting

Pour trouver les α_{t+1}, il suffit de chercher celui qui minimise

$$R(f_{t+1}) = R(f_t + \alpha_{t+1} h_{t+1}) \tag{9}$$

On a :

$$\alpha_{t+1} = argmin_\alpha \frac{1}{N} \sum_i L(f_t(x_i) + \alpha_{t+1} h_{t+1}(x_i), y_i) \tag{10}$$

qui s'obtient dans le cas présent en annulant le gradient. On retombe dans ce cas sur Adaboost.

Peut-on généraliser cette approche à d'autres fonctions de coûts ? et à des apprenants qui ne sont pas uniquement des classifieurs binaires ?

Gradient-Boosting

Pour trouver les α_{t+1}, il suffit de chercher celui qui minimise

$$R(f_{t+1}) = R(f_t + \alpha_{t+1} h_{t+1}) \tag{11}$$

On a :

$$\alpha_{t+1} = argmin_\alpha \frac{1}{N} \sum_i L(f_t(x_i) + \alpha_{t+1} h_{t+1}(x_i), y_i) \tag{12}$$

qui s'obtient dans le cas présent en annulant le gradient. On retombe dans ce cas sur Adaboost.

Peut-on généraliser cette approche à d'autres fonctions de coûts ? et à des apprenants qui ne sont pas uniquement des classifieurs binaires ?

Conclusion

Some characteristics of different learning methods.
Key: ●= good, ●=fair, and ●=poor.

Characteristic	Neural Nets	SVM	CART	GAM	KNN, Kernel	Gradient Boost
Natural handling of data of "mixed" type	●	●	●	●	●	●
Handling of missing values	●	●	●	●	●	●
Robustness to outliers in input space	●	●	●	●	●	●
Insensitive to monotone transformations of inputs	●	●	●	●	●	●
Computational scalability (large N)	●	●	●	●	●	●
Ability to deal with irrelevant inputs	●	●	●	●	●	●
Ability to extract linear combinations of features	●	●	●	●	●	●
Interpretability	●	●	●	●	●	●
Predictive power	●	●	●	●	●	●

Conclusion

Autres méthodes d'ensemble

- Classifieurs en cascade
- Hiérarchies d'experts
- ...

Sources

- *A short introduction to boosting* - Yoav Freund and Robert E. Schapire
- *Trees, Bagging, Random Forests and Boosting* - Trevor Hastier - Standford University
- *Bias-Variance Tradeoff and Ensemble Methods* - Tom Dietterich, Rich Maclin
- ...