Paris, le 8 novembre 2016

INSTITUT DES ACTUAIRES

Atelier 100% Data Science

Quels algos pour quels usages en assurance auto ?

Sommaire

1/ Base de donnée

- Approche traditionnelle
- Tarification auto

2/ Mener un projet Data Science

- Partitionnement de la base
- Importance des variables

3/ Algorithmes de Machine Learning

- Sous et sur-apprentissage
- Validation croisée
- Qualité de prévision

4/ Analyse des résultats et synthèse

Base de données auto

- **Objectif** : A partir de deux bases de données (base de contrat et base sinistres), challenger la méthode traditionnelle de tarification basée sur le GLM sur la garantie RC matérielle :
  - ✓ Garantie Responsabilité Civile (matérielle et corporelle)
  - ✓ 100 000 polices sur un historique de deux ans dont 14 748 ont au moins un sinistre RC matérielle, soit environ 50 000 polices par année observée
  - ✓ Variables liées aux caractéristiques :
    - du conducteur :
      - Sexe, âge, statut professionnel …
    - du véhicule :
      - Type, catégorie, puissance …
    - du contrat :
      - Année de souscription, exposition au risque, valeur assurée

Tarification auto

- Méthode « fréquence x coût moyen »
- Fréquence suit généralement une loi de Poisson (ou binomiale négative)
- En assurance IARD, la tarification a priori modélise généralement la variable montant de sinistre à l'aide d'une loi Gamma (ou log normale)

Approche traditionnelle : Modèles linéaires généralisés

- Rappel sur les GLM : Unifier certains modèles statistiques comme la régression linéaire, la régression logistique et la régression de Poisson.

Forme de la fonction : glm(formula, family=familytype(link = linkfonction), data=)

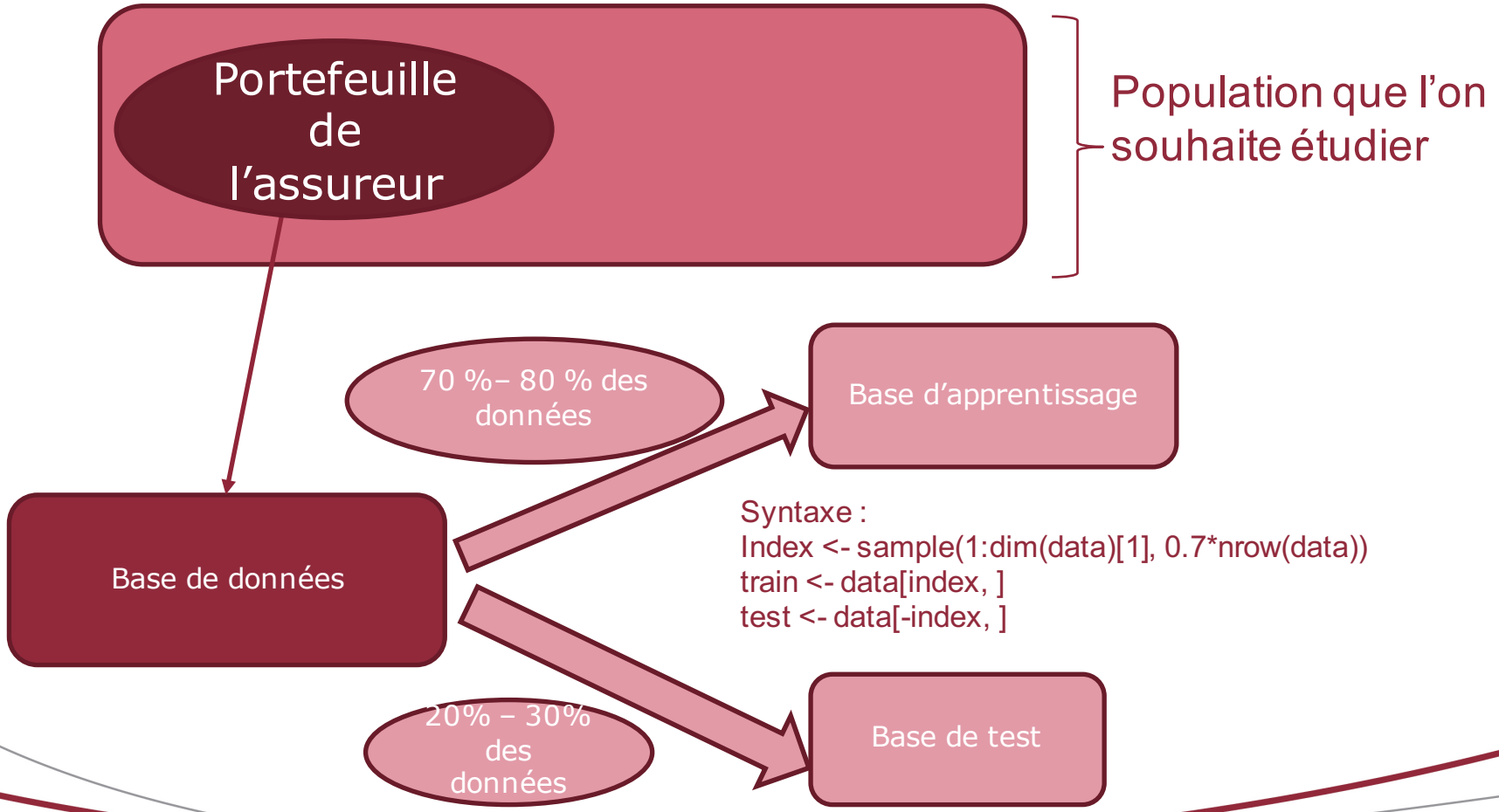
Family	Default Link Function
Binomial	(link = "logit")
Gaussian	(link = "identity")
Gamma	(link = "inverse")
Poisson	(link = "log")

GLM

- GLM faciles à interpréter et faciles à implémenter
- GLM ne capture pas la complexité dans les données
- GLM tendance à « sous-apprendre »
- Approches statistiques traditionnelles de plus en plus challenger par des méthodes de machine learning

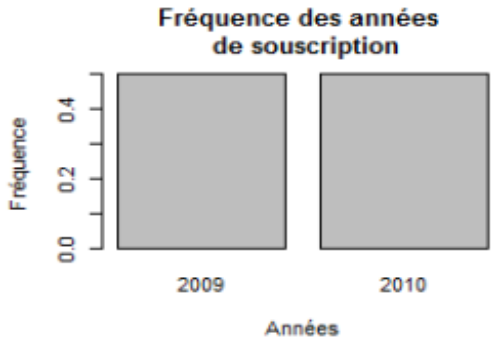
Partitionnement de la base

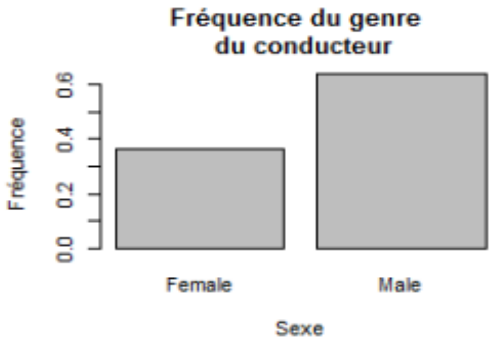
Base d'apprentissage et Base de test

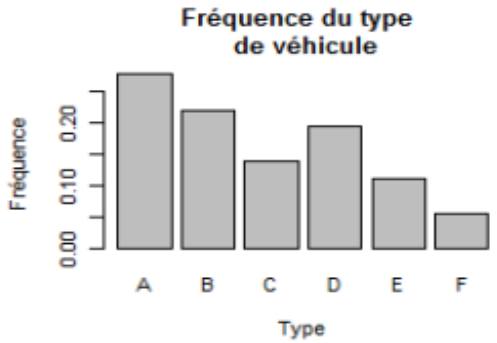


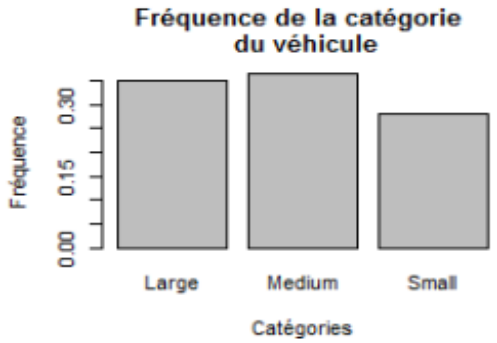
4, rue Chauveau-Lagarde - 75008 Paris - Tél : 01 44 51 72 72 - Fax : 01 44 51 72 73
www.institutdesactuaires.com

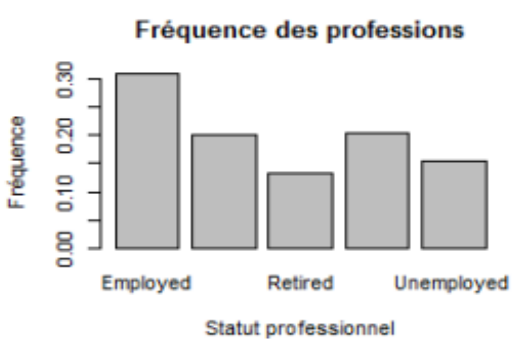
Importance des variables I

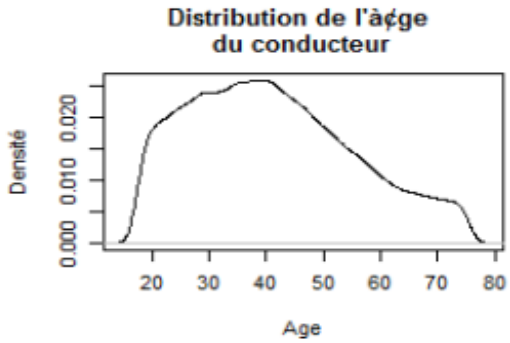


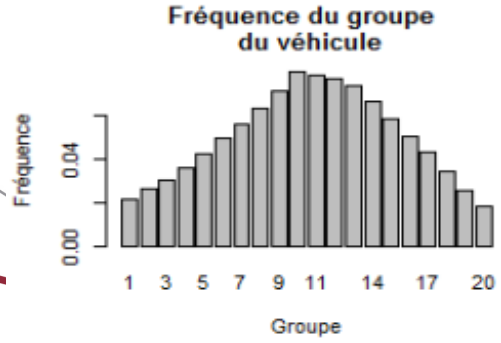


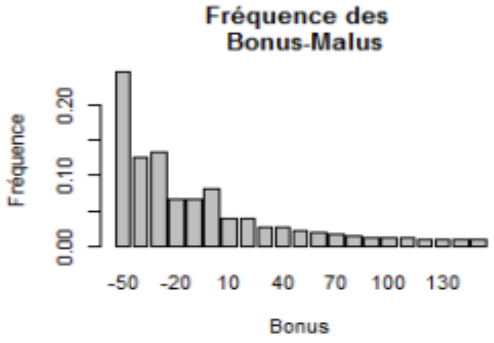


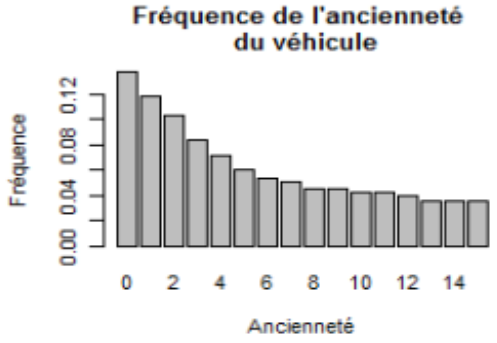






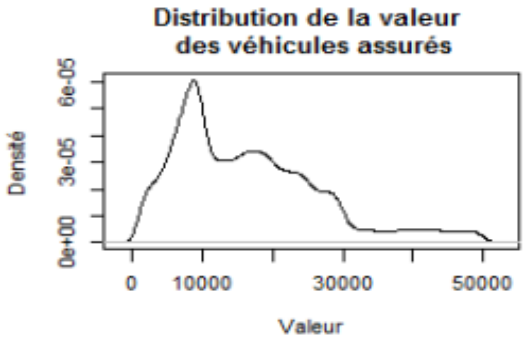


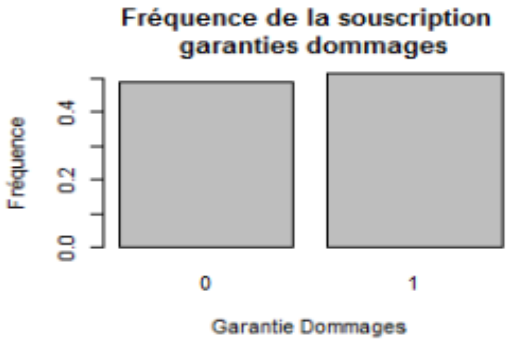


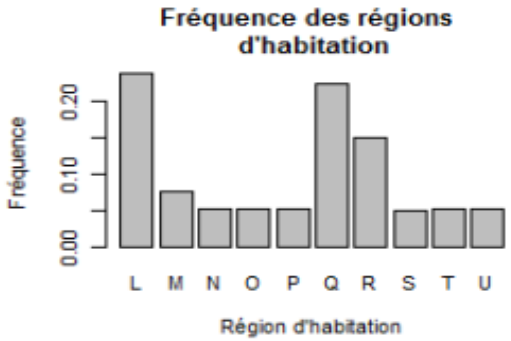


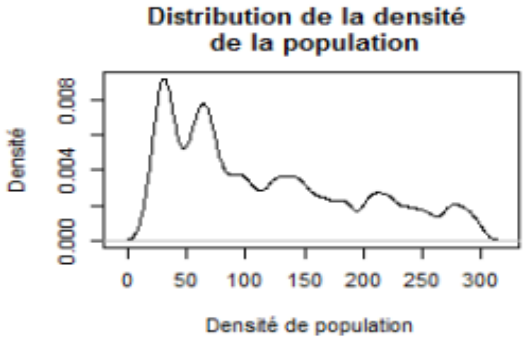
4, rue Chauveau-Lagarde - 75008 Paris - Tél : 01 44 51 72 72 - Fax : 01 44 51 72 73
www.institutdesactuaires.com

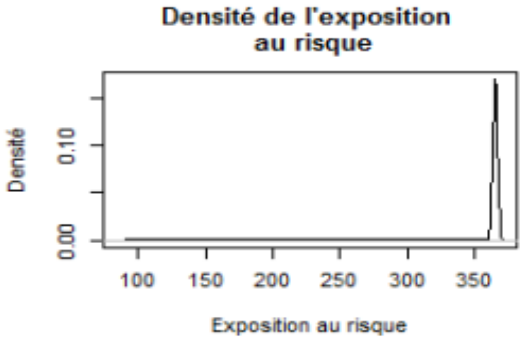
Importance des variables II

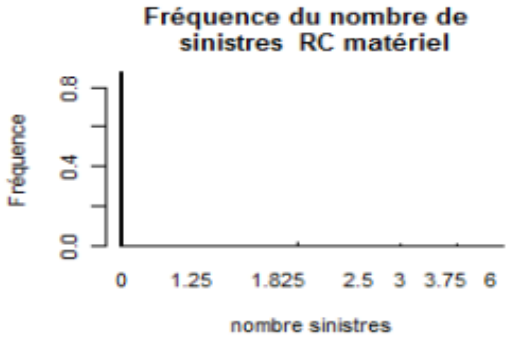


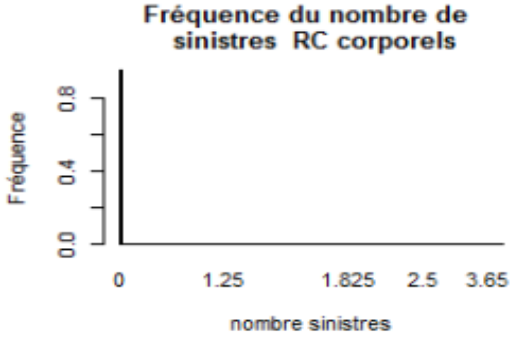


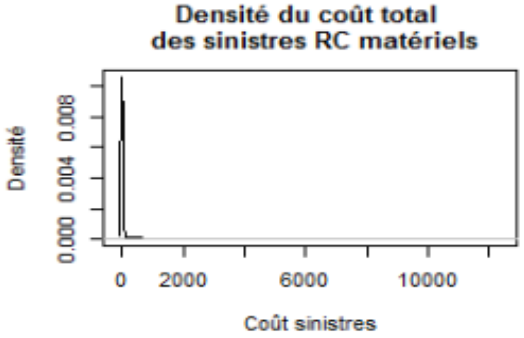


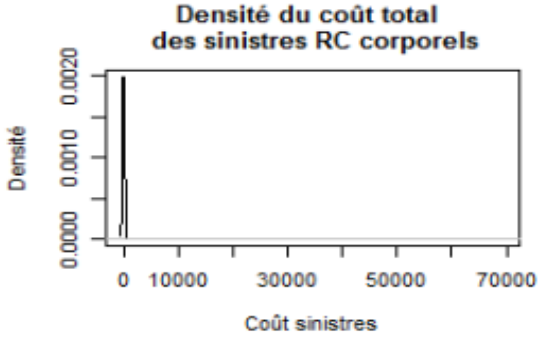












Garantie : RC matérielle

- ✓ 14 475 sinistres au titre de la garantie « RC matérielle » sur les deux années d'observations
- ✓ Sinistres max : 12 325€
- ✓ Pas de voiture « haut de gamme » dans la base

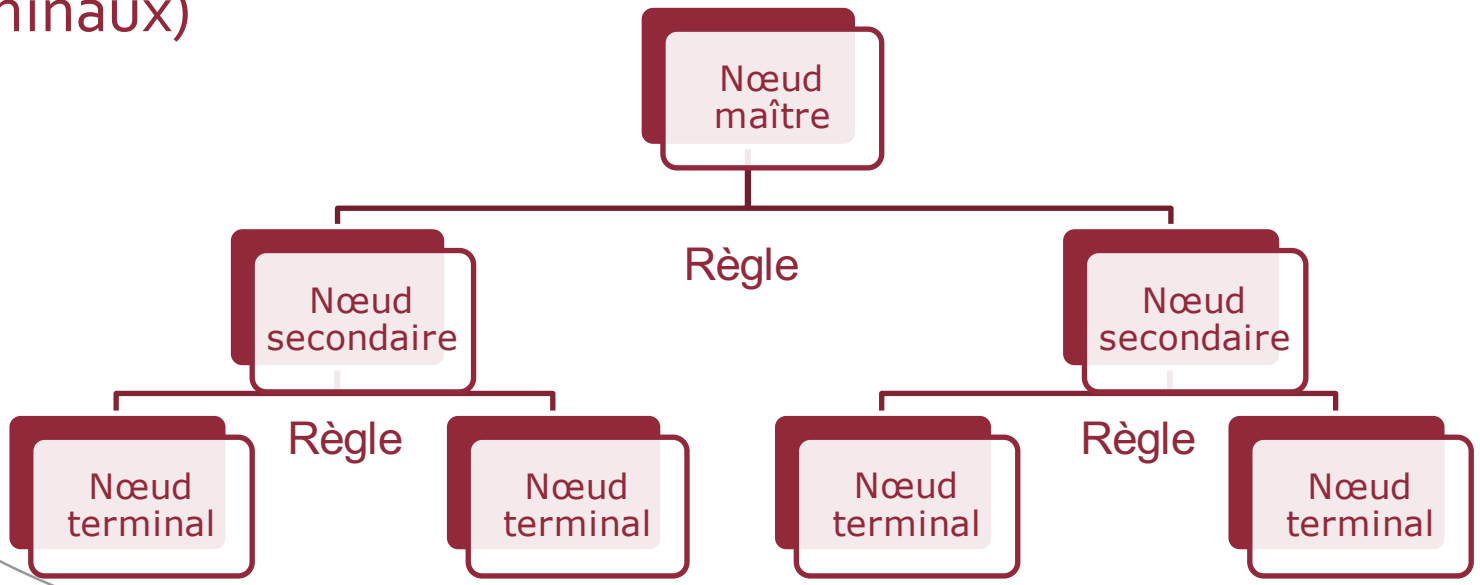
Algorithmes de machine learning

Trois méthodes d'apprentissage supervisé en régression :

- ✓ Arbres de décision / CART
- ✓ Forêts aléatoires (RF)
- ✓ Gradient Boosting Machine (GBM)

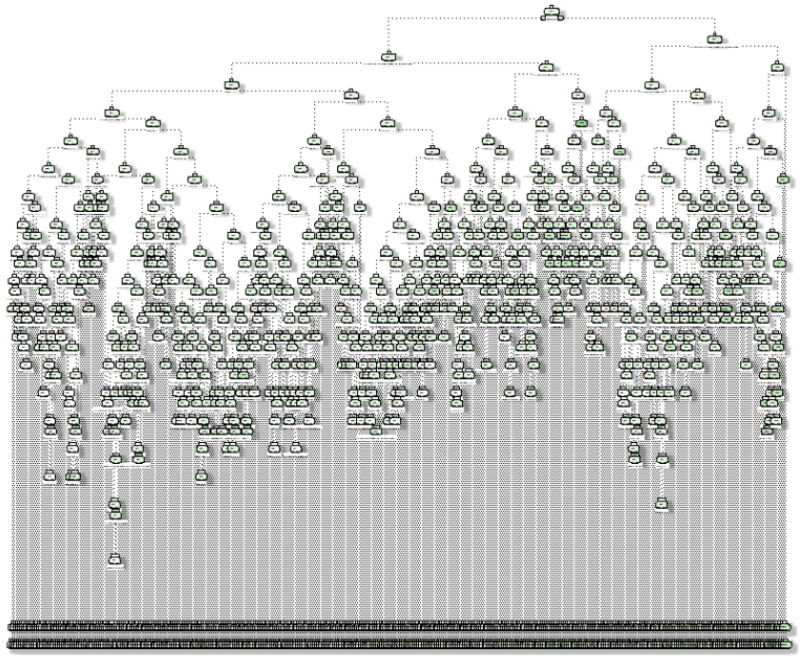
Construction d'un arbre

- ✓ Un nœud est l'intersection d'un ensemble de règle
- ✓ Les sous populations créées sont disjointes
- ✓ Un arbre se lit de la racine (nœud maître) vers les feuilles (nœuds terminaux)



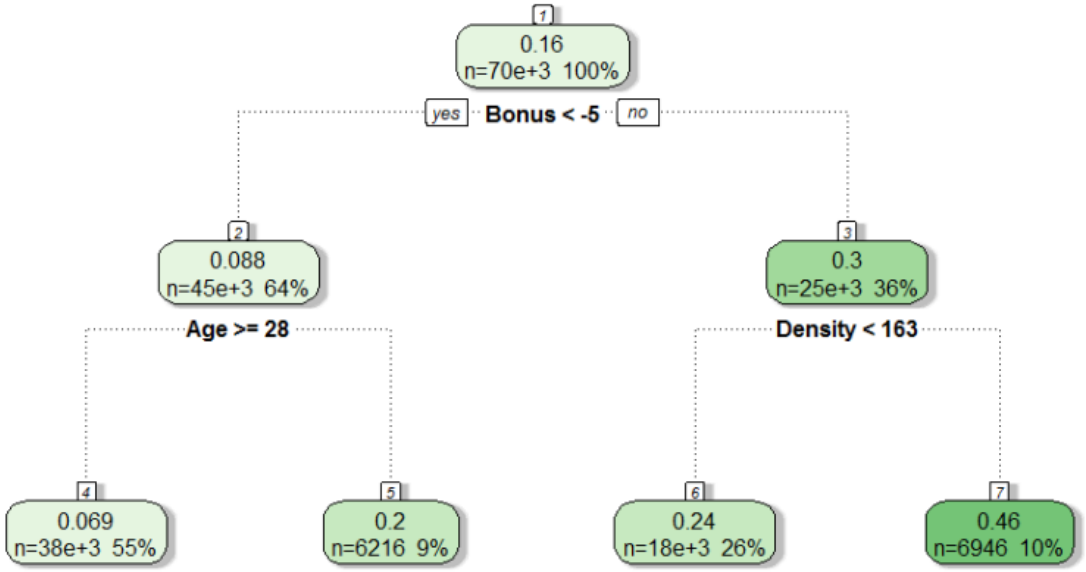
CART

- ✓ Classification And Regression Trees
- ✓ Arbre de décision constitué de :
  - Un nœud maître
  - De nœuds secondaires
  - De feuilles (nœuds terminaux)
- ✓ Classifieurs faibles



CART

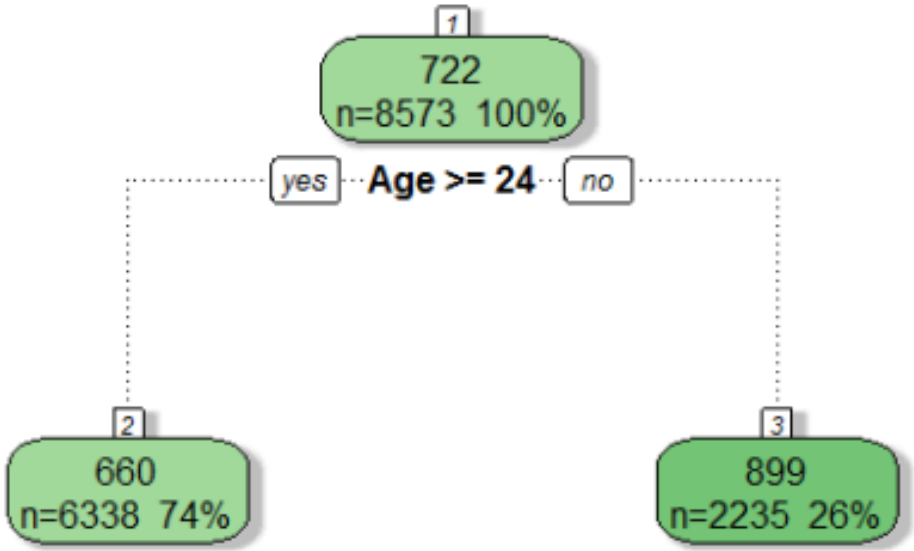
- Variable fréquence :



- Variable explicative : « Bonus-Malus »
- Un nœud maître, deux nœuds secondaires et quatre nœuds terminaux

CART

- Variable montant de sinistres :



- Un nœud maître et deux nœuds terminaux
- Variable explicative : «Age »

Méthodes ensemblistes

- **Agrégation de modèles** basée sur des stratégies aléatoires « bagging ou forêts aléatoires » ou adaptatives « boosting »
  - Permettent d'améliorer l'ajustement par une combinaison ou une agrégation d'un grand nombre de modèle tout en évitant le sur-apprentissage
  - Permettent de construire une famille de modèles qui sont agrégés par une moyenne pondérée des estimations ou d'un vote
  - Principes s'appliquent à toute méthode de modélisation (régression, CART, …)
  - Plus efficaces sur les modèles linéaires

Bagging

- **Pour bootstrap aggregating** (Breiman 1996) et les forêts aléatoires (Breiman 2001) qui améliorent le bagging spécifique aux modèles définis par des arbres binaires (CART)

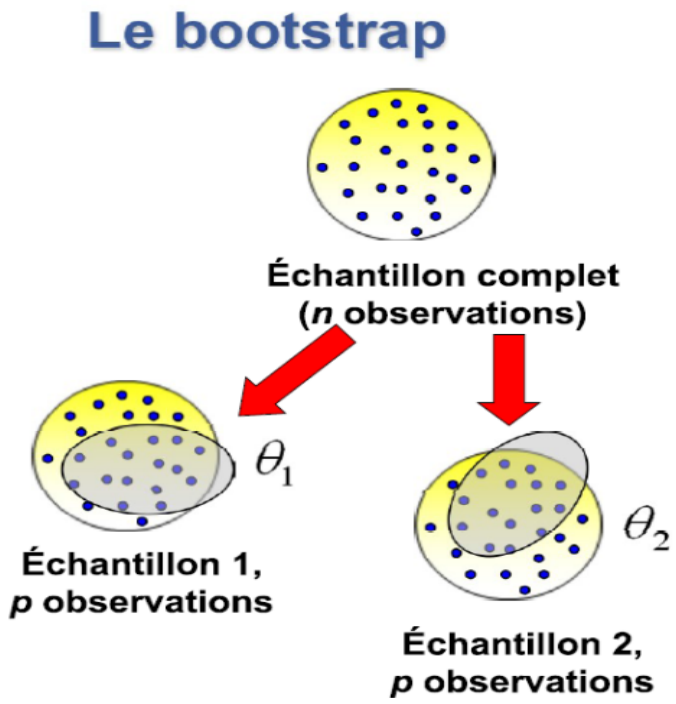
  - Algorithme 1 : Bagging

Soit $\mathbf{x}_0$ *à prévoir et*
$\mathbf{z} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ *un échantillon*
for $b = 1$ à B ***do***
  Tirer un échantillon bootstrap $\mathbf{z}_b^*$.
  Estimer $\widehat{\phi}_{\mathbf{z}_b}(\mathbf{x}_0)$ *sur l'échantillon bootstrap.*
end for
Calculer l'estimation moyenne $\widehat{\phi}_B(\mathbf{x}_0) = \frac{1}{B}\sum_{b=1}^{B} \widehat{\phi}_{\mathbf{z}_b}(\mathbf{x}_0)$ *ou le résultat du vote.*

Méthode du bootstrap

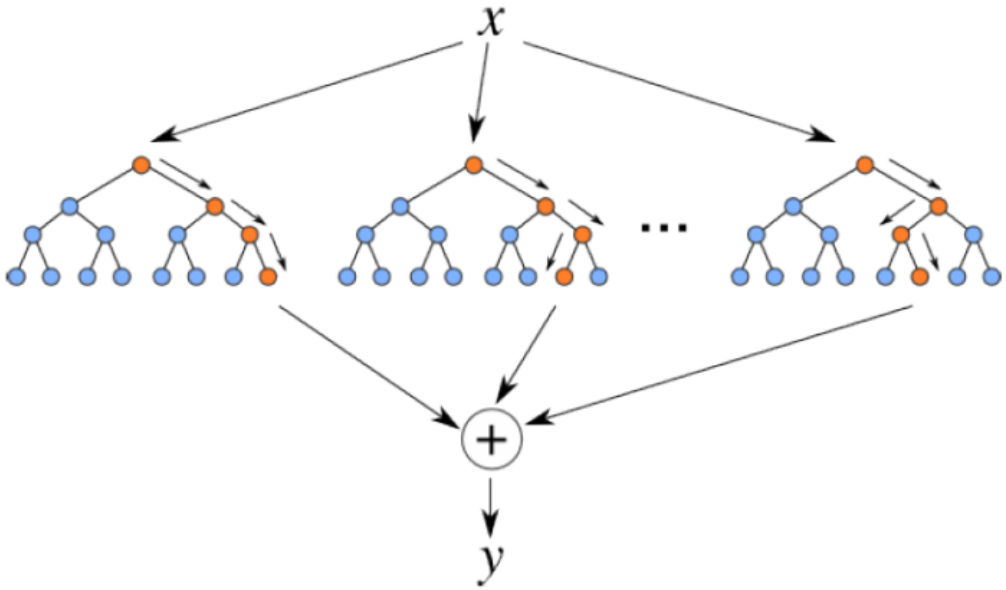
➢ Rappel sur la méthode du bootstrap :

- *n* observations $x_1, x_2, x_3, \ldots, x_n$
- créer m échantillons de p $\leq$ n observations, échantillons avec remises



Bagging

➢ **Principe du bagging :** agréger plusieurs modèles.



- cas de la régression : $\hat{f}_B(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{\phi}_b(x)$
- cas de la classification : $\hat{f}_B(x) = \underset{j\in(1,2,\ldots,J)}{argmax} \frac{1}{B}\sum_{b=1}^{B}1_{\{\hat{\phi}_b(x)==j\}}$

Forêts aléatoires

- **Dans le cas spécifique des modèles d'arbres binaires,** Breiman (2001) propose une amélioration du bagging avec l'ajout d'une « randomisation » des variables.

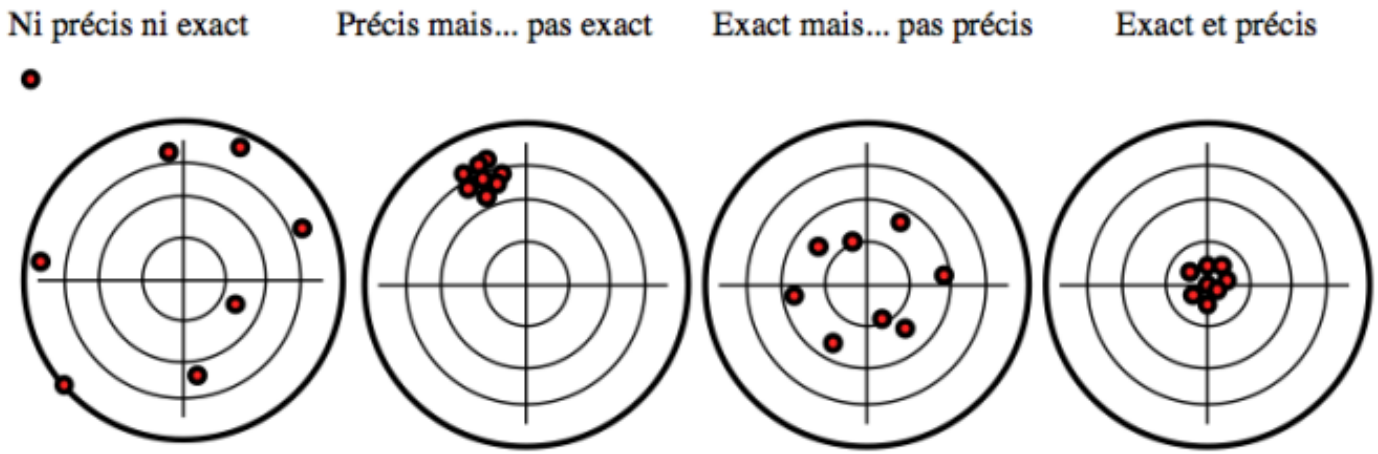
  - Algorithme 2 : Forêts aléatoires

Soit $\mathbf{x}_0$ *à prévoir et*
$\mathbf{z} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ *un échantillon*
for $b = 1$ à B ***do***
  Tirer un échantillon bootstrap $\mathbf{z}_b^*$
  Estimer un arbre sur cet échantillon avec **randomisation** *des variables : la recherche de chaque nœud optimal est précédé d'un tirage aléatoire d'un sous-ensemble de* m *prédicteurs.*
end for
Calculer l'estimation moyenne $\widehat{\phi}_B(\mathbf{x}_0) = \frac{1}{B}\sum_{b=1}^{B} \widehat{\phi}_{\mathbf{z}_b}(\mathbf{x}_0)$ *ou le résultat du vote.*

Boosting

- **Freund and Shapire, 1999 :** Algorithme de base Adaboost (adaptative boosting)
  - Le boosting adopte le même principe que le bagging
- **Principe du boosting :**
  - Combiner des classifieurs faibles en classifieurs agrégés robustes
  - Classifieurs ayant prédit correctement ont un poids plus fort
  - Construire une séquence de modèles de sorte qu'à chaque étape chaque modèle apparaisse comme un pas vers une meilleure solution

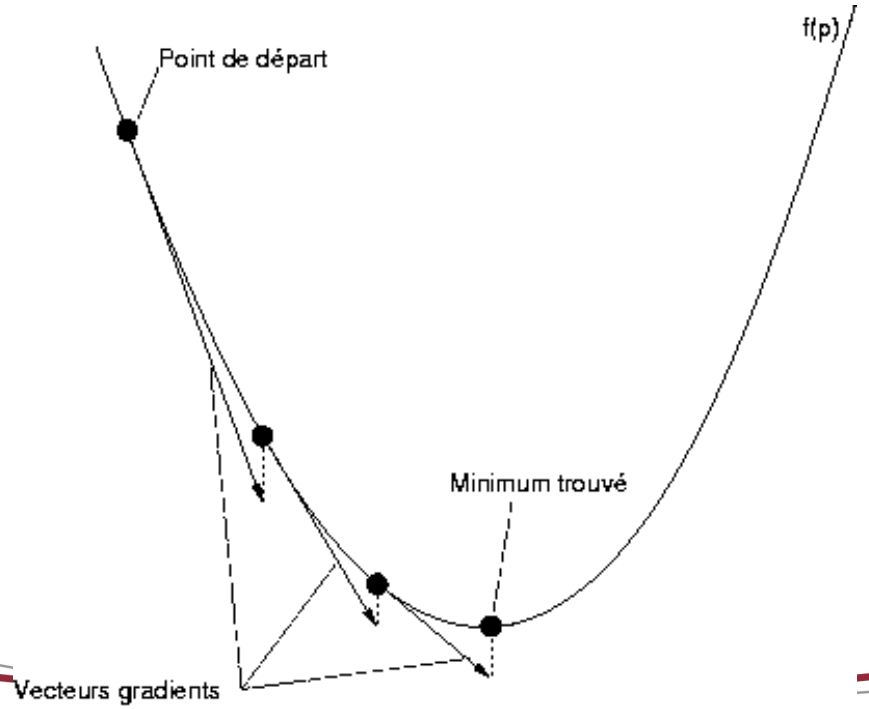
Equilibre « biais/variance »



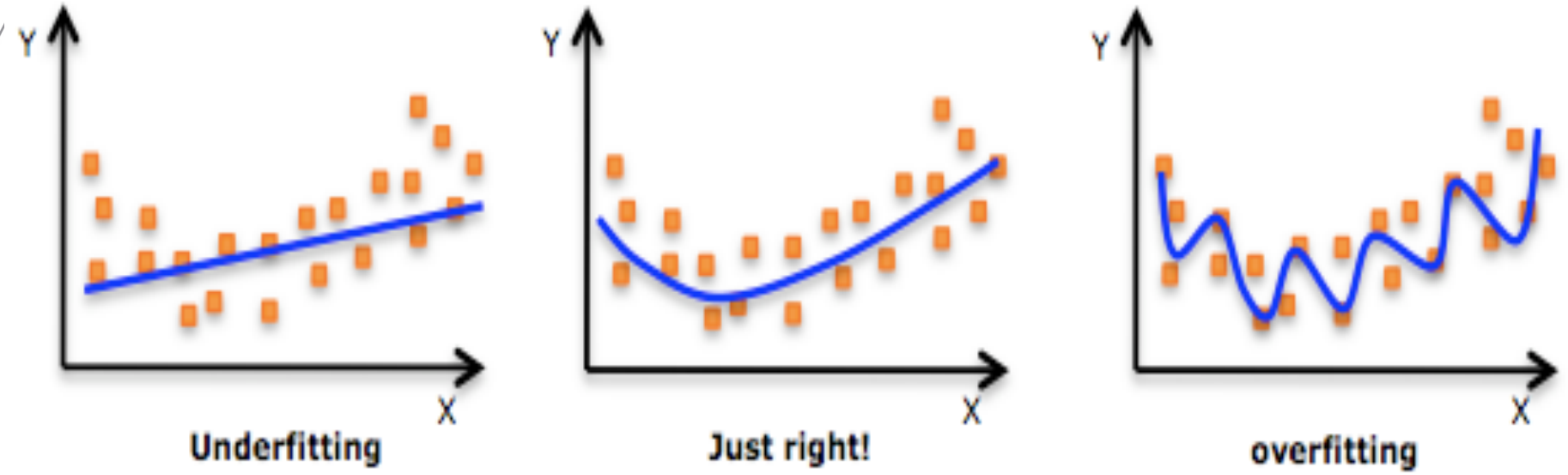
- Le bagging réduit la variance des modèles à faible biais
- Le boosting réduit le biais des modèles à forte variance

Gradient Boosting Model

- Méthode de descente de gradient pour trouver le minimum de la fonction de perte
- Le gradient indique la direction dans laquelle se produit la plus grande décroissance de la fonction de perte



« Sous et sur-apprentissage »



Méthode de validation croisée

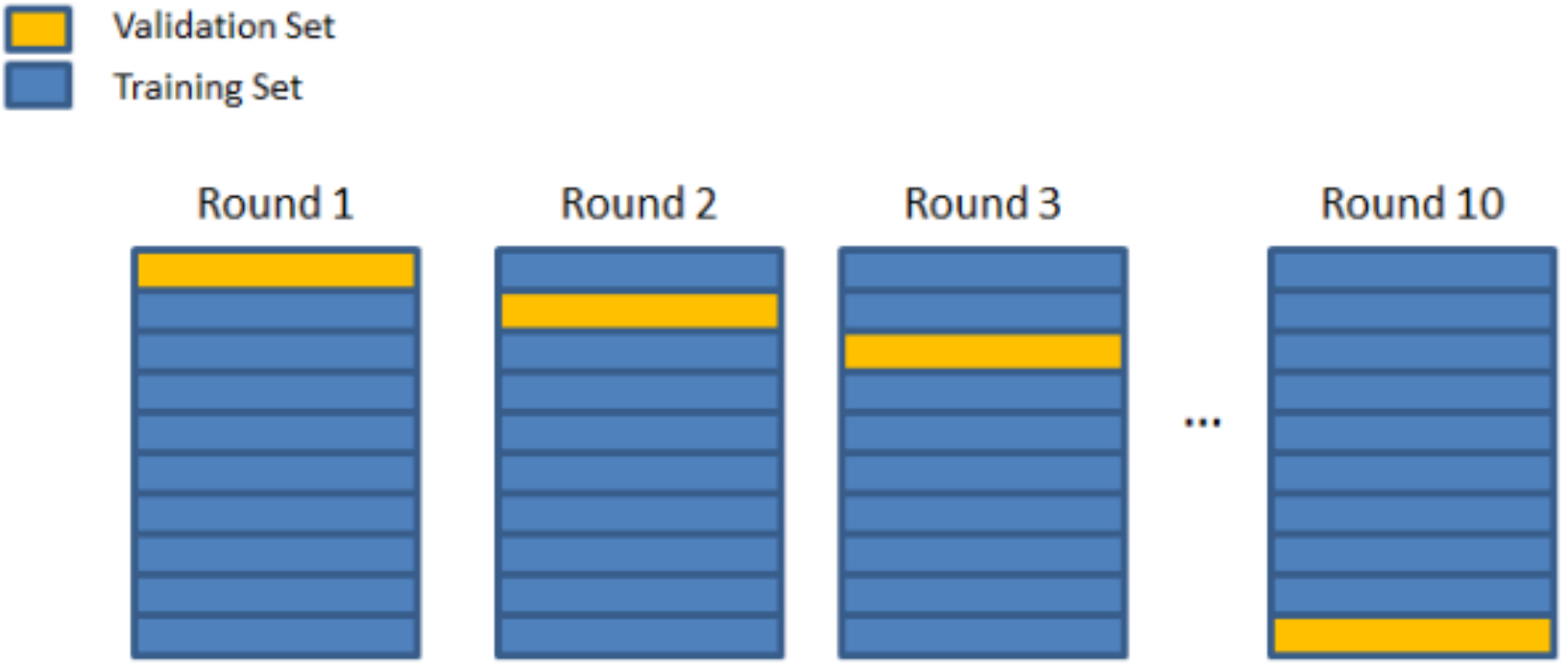
- Principe simple, efficace et très utilisée
  - Itérations de l'estimation de l'erreur sur plusieurs échantillons plusieurs échantillons de *validation* puis d'en calculer la moyenne,
  - Indispensable pour réduire la variance et améliorer la précision

Algorithme 3 :

Découper aléatoirement l'échantillon en K parts (K-fold) de tailles approximativement égales selon une loi uniforme ;
for *$k=1$ à K* **do**
    mettre de côté l'une des partie,
    estimer le modèle sur les $K-1$ parties restantes,
    calculer l'erreur sur chacune des observations qui n'ont pas participé à l'estimation
end for
moyenner toutes ces erreurs pour aboutir à l'estimation par validation croisée.

Validation croisée

Exemple avec 10 k-folds



Qualité de prévision

- Métrique permet de mesurer l'efficacité d'un algo
- Métrique utilisée dans notre étude

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

- Cette métrique représente l'erreur moyenne élevée au carré

Résumé de la procédure suivie dans notre étude

1. Partitionnement de la base de donnée (70% train et 30% test)
2. Description des variables
3. Algos : GLM, CART, RF, et GBM
4. Validation croisée sur les algos avec k-folds = 5
5. Utilisation de la métrique MSE pour les comparer

Résultats

MSE	Fréquence Sinistre	Montant Sinistre	Prime Pure
GBM	0,209	850 508	177 756
GLM	0,212	860 168	182 356
RF	0,214	871 277	186 453
CART	0,231	905 689	209 214

Gain de 1,03% sur la garantie RC matérielle de notre portefeuille

100% DATA SCIENCE

Algos VERSUS THE COMPETITION IS ON GLM

INSTITUT DES ACTUAIRES

	GLM	CART	RF	GBM
FORCES	Facile à interpréter	Facile à interpréter	L'un des meilleurs nécessitant quasiment aucun réglage	Souvent plus précis que les RF
	Facile à implémenter	Rapide à entraîner, rapide à prédire	Contrôle de la variance à travers la randomisation qui donne de meilleure performance	Structures souples qui s'adaptent aux fonctions de pertes
	Reconnu comme des standards en assurance	Méthode non-paramétrique	Méthode non-paramétrique	Contrôle du "sous ou sur" apprentissage
FAIBLESSES	Méthode paramétrique	Faible biais mais généralement variance élévée (solution : combiner les solutions de plusieurs arbres en un seul modèle)	Modérement rapide à entraîner, mais rapide pour prédire	Lent à entraîner, mais rapide pour prédire
	Ne capturent pas la complexité automatiquement dans les données		Moins interprétables que les arbres de décision	Réglages minutieux requis
	Ont tendances à "sous" apprendre			Difficilement interprétable

Synthèse

- ✓ GBM fait un peu mieux que le GLM
- ✓ Utiliser les GBM pour les garanties mal interprétées par les GLM (ex : la garantie Vol)
- ✓ Résultats dépendent des données utilisées :
  - Les algos s'adaptent aux données et non l'inverse
- ✓ Explicabilité des algorithmes :
  - Pour les assurés, les actionnaires, le régulateur

→ Choix des algos : CART, RF et GBM pour l'assurance auto.

100% DATA SCIENCE

INSTITUT DES ACTUAIRES

QUESTIONS ?

4, rue Chauveau-Lagarde - 75008 Paris - Tél : 01 44 51 72 72 - Fax : 01 44 51 72 73
www.institutdesactuaires.com

100% DATA SCIENCE

INSTITUT DES ACTUAIRES

MERCI DE VOTRE ATTENTION !

4, rue Chauveau-Lagarde - 75008 Paris - Tél : 01 44 51 72 72 - Fax : 01 44 51 72 73
www.institutdesactuaires.com

100% DATA SCIENCE

INSTITUT DES ACTUAIRES

ANNEXES

4, rue Chauveau-Lagarde - 75008 Paris - Tél : 01 44 51 72 72 - Fax : 01 44 51 72 73
www.institutdesactuaires.com

Algorithme CART sous R

Rpart: Recursive Partitioning and Regression Trees

Recursive partitioning for classification, regression and survival trees. An implementation of most of the functionality of the 1984 book by Breiman, Friedman, Olshen and Stone.

Version:	**4.1-10**
Priority:	recommended
Depends:	R (≥ 2.15.0), graphics, stats, grDevices
Suggests:	survival
Published:	2015-06-29
Author:	Terry Therneau [aut], Beth Atkinson [aut], Brian Ripley [aut, trl, cre] (author of initial R port)
Maintainer:	Brian Ripley <ripley at stats.ox.ac.uk>
License:	GPL-2 | GPL-3
NeedsCompilation:	yes
Materials:	NEWS ChangeLog
In views:	Environmetrics, MachineLearning, Multivariate, Survival
CRAN checks:	rpart results

Downloads:

Reference manual:	rpart.pdf
Vignettes:	Introduction to Rpart User Written Split Functions
Package source:	rpart_4.1-10.tar.gz
Windows binaries:	r-devel: rpart_4.1-10.zip, r-release: rpart_4.1-10.zip, r-oldrel: rpart_4.1-10.zip

Bagging sous R

Adabag: Applies Multiclass AdaBoost.M1, SAMME and Bagging

Version:	**4.1**
Depends:	rpart, mlbench, caret
Published:	2015-10-14
Author:	Alfaro, Esteban; Gamez, Matias and Garcia, Noelia; with contributions from Li Guo
Maintainer:	Esteban Alfaro <Esteban.Alfaro at uclm.es>
License:	GPL-2 | GPL-3 [expanded from: GPL (≥ 2)]
NeedsCompilation:	no
Citation:	adabag citation info
CRAN checks:	adabag results

Downloads:

Reference manual:	**adabag.pdf**
Package source:	adabag_4.1.tar.gz
Windows binaries:	r-devel: adabag_4.1.zip, r-release: adabag_4.1.zip, r-oldrel: adabag_4.1.zip

Forêts aléatoires sous R

RandomForest: Breiman and Cutler's Random Forests for Classification and Regression

Classification and regression based on a forest of trees using random inputs.

Version:	**4.6-12**
Depends:	R (≥ 2.5.0), stats
Suggests:	RColorBrewer, MASS
Published:	2015-10-07
Author:	Fortran original by Leo Breiman and Adele Cutler, R port by Andy Liaw and Matthew Wiener.
Maintainer:	Andy Liaw <andy_liaw at merck.com>
License:	GPL-2 | GPL-3 [expanded from: GPL (≥ 2)]
URL:	https://www.stat.berkeley.edu/~breiman/RandomForests/
NeedsCompilation:	Yes
Citation:	randomForest citation info
Materials:	NEWS
In views:	Environmetrics, MachineLearning
CRAN checks:	randomForest results

Download

Reference manual:	randomForest.pdf
Package source:	randomForest_4.6-12.tar.gz
Windows binaries:	r-devel: randomForest_4.6-12.zip, r-release: randomForest_4.6-12.zip, r-oldrel: randomForest_4.6-12.zip