DEVOIR_S2 DATA_MINING

SABAYE Fried-Junior, Caleb KASHALA, Mohamad EL KAWASS, Hicham AZOUD

Table des matières

1 Contexte et problématique

Cette étude fait suite à l'analyse factorielle éffectuée au **semestre précédent**.

La base de données renseigne sur diverses variables et paramètres concernant des accidents de vélo.

Les dix premières lignes et colonnes de la base de données figurent dans le tableau suivant :

	Bike_Age	Bike_Alc_D	Bike_Dir	Bike_Injur	Bike_Pos	Bike_Race	Bike_Sex	Developmen	Drvr_Alc_D	Drvr_Injur
1	6	No	Not Applicable	Non	Driveway / Alley	Black	Female	Residential	No	O: No Injury
2	51	No	With Traffic	Non	Travel Lane	Black	Male	Commercial	No	O: No Injury
3	10	No	With Traffic	Oui	Travel Lane	Black	Male	Residential	No	O: No Injury
6	52	No	With Traffic	Oui	Travel Lane	White	Male	Commercial	No	O: No Injury
9	6	No	Facing Traffic	Oui	Travel Lane	White	Male	Residential	No	O: No Injury
12	30	No	With Traffic	Oui	Travel Lane	Black	Male	Commercial	No	O: No Injury
13	17	No	With Traffic	Oui	Travel Lane	White	Male	Residential	No	O: No Injury
14	20	No	With Traffic	Oui	Travel Lane	White	Male	Residential	No	O: No Injury
15	14	No	Facing Traffic	Non	Travel Lane	White	Male	Residential	No	B: Evident Injury
17	19	No	With Traffic	Oui	Travel Lane	Black	Male	Residential	No	O: No Injury

Le but de cette étude est de prédire la variable : *Bike_Injur*. Pour cela nous allons utiliser diverses méthodes d'apprentissage supervisé.

Table 1 – Effectif de la variable à prédire

Bike_Injur	Nombre
Non	2725
Oui	2991

1.1 Méthodologie

Nous allons séparer la base de données en deux échantillons : Apprentissage et Test.

La base de données sera séparée comme suit : 2/3 des données constitueront l'*échantillon d'Apprentissage* qui sera utilisé pour construire les modèles ; les 1/3 restant qui constitueront l'*échantillon Test* seront utilisés pour tester nos modèles.

Pour chacunes des méthodes d'apprentissage supervisé qui sera utilisée : on commencera par construire le(s) modèle(s) en fonction des différents paramètres propres à chacun d'eux et ce, sur l'échantillon d'apprentissage. Ensuite, dans le but de déterminer le meilleur paramétrage de chaque méthode : nous appliquerons ce(s) modèle(s) ainsi construit(s) sur l'échantillon test, puis nous comparerons les diverses mesures de performance. Ce qui nous permettra de choisir, si il le faut, le meilleur modèle de chaque méthode. Enfin, nous comparerons les résultats des meilleurs modèles de chaque méthode afin de déterminer celui permettant de mieux prédire.

Quelques précisions

1) Après la transformation, la sélection des variables et l'affectation des effectifs réalisée **au premier semestre**, les variables restantes pouvant être utilisées sont : *Bike_Age, Bike_Alc_D, Bike_Dir, Bike_Injur, Bike_Pos, Bike_Race, Bike_Sex, Crash_Hour, Crash_Loc, Crash_Time, Crash_Type, Crash_Ty_1, Developmen, DrvrAge_Gr, Drvr_Age, Drvr_Alc_D, Drvr_Injur, Drvr_Race, Drvr_Sex, Hit_Run, Light_Cond, Num_Lanes, Num_Units, Rd_Charact, Rd_Class, Rd_Conditi, Region, Rural_Urba et Workzone_I.*

2) En apprentissage automatique supervisé, une matrice de confusion est une matrice qui mesure la qualité d'un système de classification. Elles s'interprétent comme suit :

TABLE 2 – Effectif de la variable à prédire

Classe réelle	Classe estimée par le classificateur	
	Non	Oui
Non	Vrais négatifs	Faux négatifs
Oui	Faux positifs	Vrais positifs

Un vrai positif(VP) est un résultat où le modèle prédit correctement la classe positive. De façon analogue, un vrai négatif(VN) est un résultat où le modèle prédit correctement la classe négative.

Un faux positif(FP) est un résultat où le modèle prédit incorrectement la classe positive et un faux négatif(FN) est un résultat où le modèle prédit incorrectement la classe négative.

3) À partir de la matrice de confusion on peut dériver tout un tas de critères de performance. Parmi lesquels :

• **La sensibilité** : le taux de vrais positifs, c'est à dire la proportion de bléssés que l'on a correctement identifiés. C'est la capacité de notre modèle à détecter toutes les blessures.

$$Sensibilité = \frac{VP}{VP + FN}$$

• **La spécificité** : le taux de vrais négatifs, autrement dit la capacité à détecter toutes les situations où il n'y a pas de bléssures. C'est une mesure complémentaire de la sensibilité.

$$Spécificité = \frac{VN}{FP + VN}$$

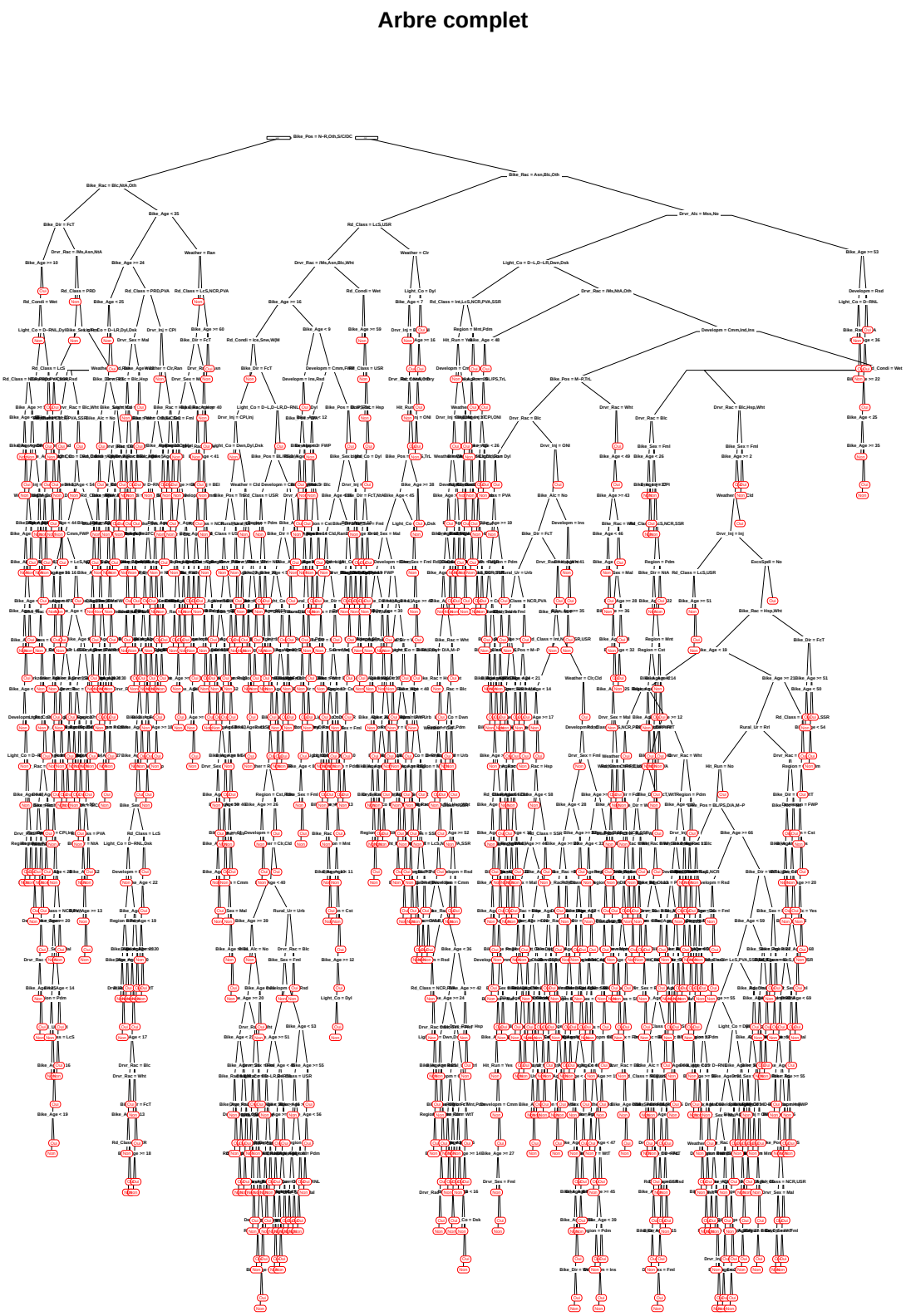
• **La précision** : la proportion de prédictions correctes parmi les blessures que l'on a prédites positives. C'est la capacité de notre modèle à prédire qu'un individu à été bléssé que si il à été réellement bléssé.

$$Précision = \frac{VP}{VP + FP}$$

2 Arbres

Nous allons commencer par réaliser un modèle d'arbre complet sur *l'échantillon d'apprentissage* et ce, pour avoir une vue d'ensemble.

2.1 Arbre complet



Nous savons qu'un arbre complet n'est généralement pas utilisé pour les prédictions car il ne commet aucune

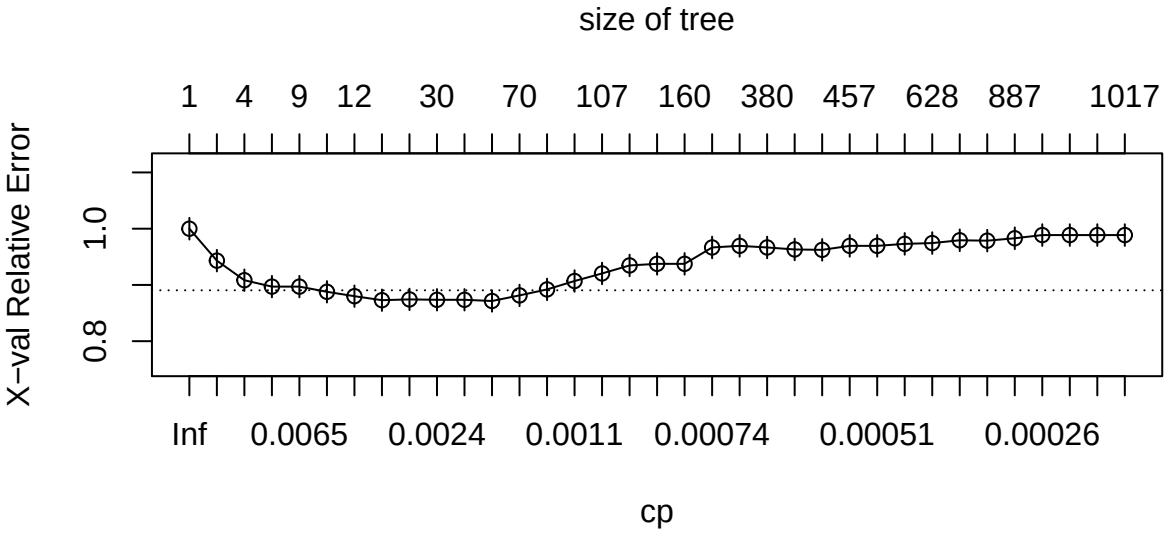
erreur sur l'échantillon sur lequel il est construit, puisqu'il en épouse toutes les caractéristiques. Par conséquent il est difficilement généralisable.

Nous devons donc élaguer notre arbre selon un certain niveau de complexité afin de palier à ce problème de surraprentissage.

2.2 Arbre élagué

Cherchons le niveau de complexité qui minimise l'erreur estimée.

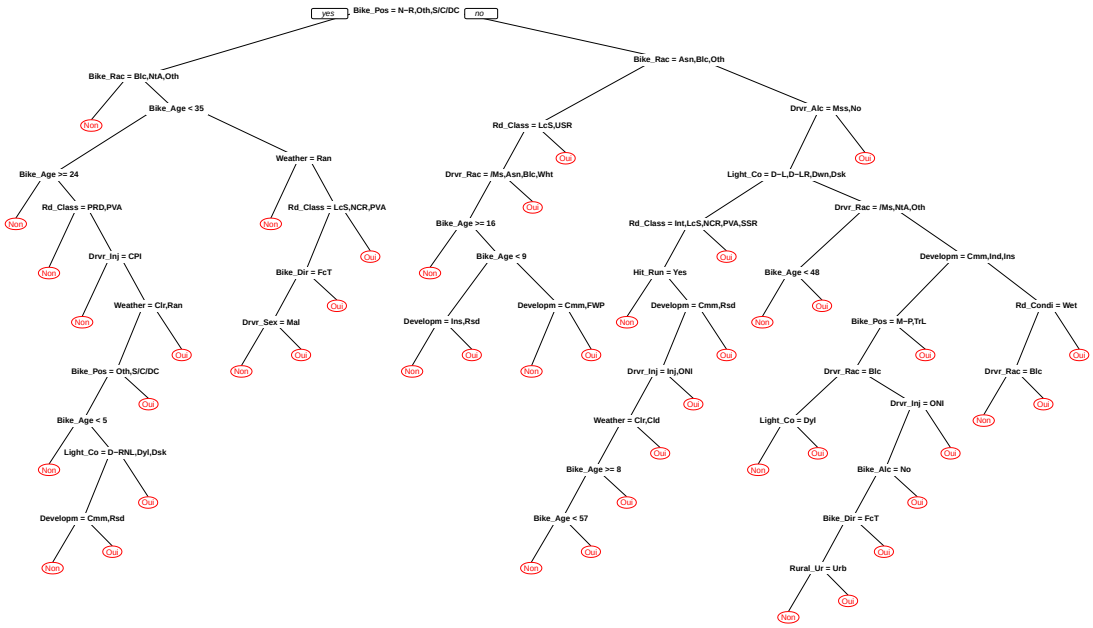
Nous allons réalisé un graphique qui nous montre le taux d'erreur en fonction de la complexité.



Ce graphique nous montre que la complexité qui permet de minimiser l'erreur estimée est de 0.001420455 avec une erreur égale à 0.88 environ.

En appliquant ce niveau de complexité on obtient l'arbre élagué suivant :

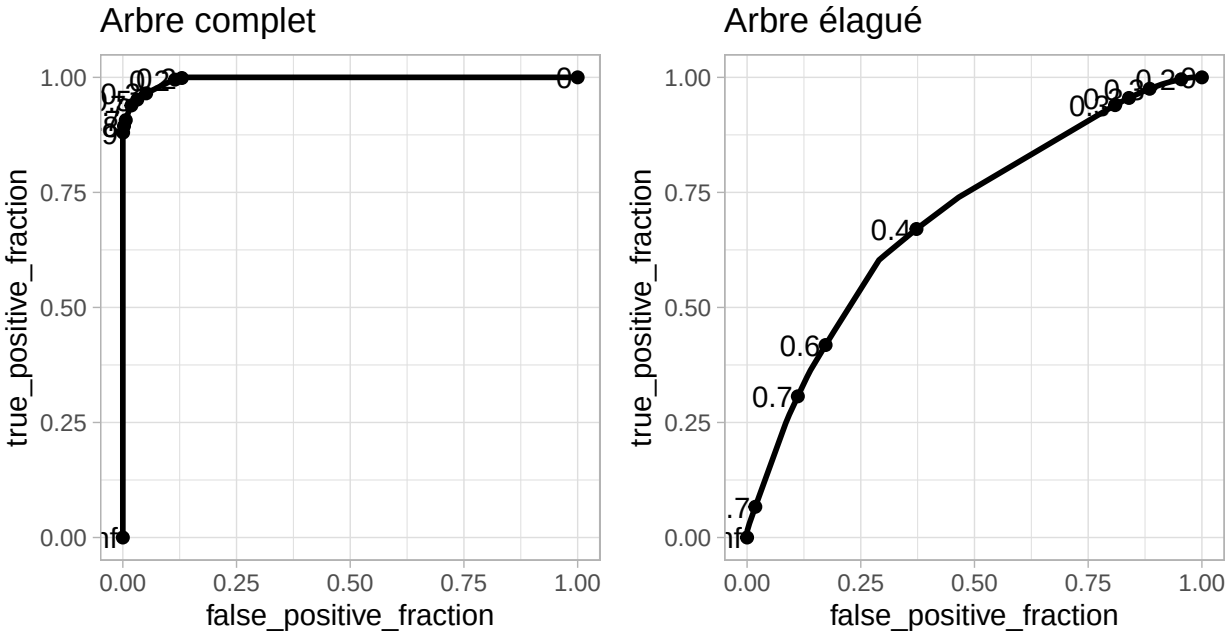
Arbre élagué



Pour se convaincre de l'utilisation d'un arbre élagué par rapport à l'arbre complet on peut représenter les **courbes ROC**.

Une courbe ROC (receiver operating characteristic) est une représentation graphique de la relation qu'il existe

entre la sensibilité (qui mesure sa capacité à donner un résultat positif lorsqu'une hypothèse est vérifiée) et la spécificité (qui mesure la capacité d'un test à donner un résultat négatif lorsque l'hypothèse est vérifiée) d'un test pour chaque valeur seuil considérée.



Comme prévu, la courbe ROC de l'arbre complet est parfaite à cause du surapprentissage. La courbe ROC de l'arbre élagué est près de la bissectrice mais suffisament au dessus pour conclure que l'utilisation de l'arbre élagué est préférable à celui d'un classificateur aléatoire .

Nous poursuivrons la comparaison en appliquant ces deux modèles à l'échantillon d'apprentissage

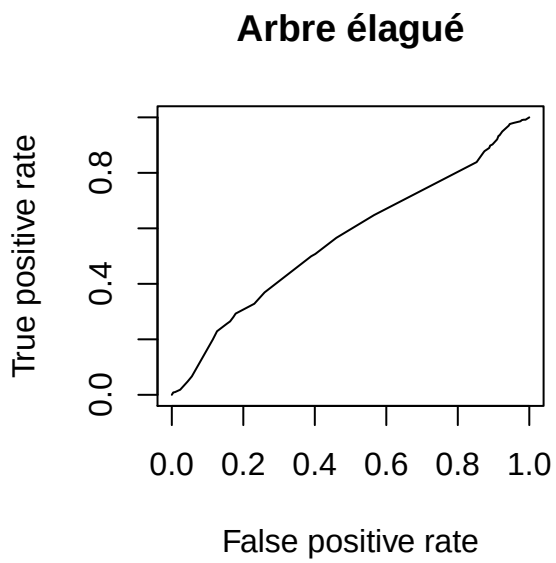
2.3 Application à l'échantillon test

En appliquant ces modèles à *l'échantillon Test* on obtient la matrice de confusion suivante :

Table 3 – Matrice de confusion Arbres : test

Prédiction	Référence			
	Arbre complet		Arbre élagué	
	Non	Oui	Non	Oui
Non	355	311	353	299
Oui	352	456	354	468

Courbe ROC



Les mesures de performance de nos arbres appliqués à l'échantillon test sont renseignés dans le tableau suivant :

TABLE 4 – mesures de performance

	Sensitivity	Specificity	Precision	AUC
Arbre complet	0.5021216	0.5945241	0.533033	0.5457836
Arbre élagué	0.4992928	0.6101695	0.541411	0.5609402

Conclusion: Il semblerait, dans le cas particulier de notre base de données, que l'arbre complet et l'arbre élagué aient des performances qui se valent.

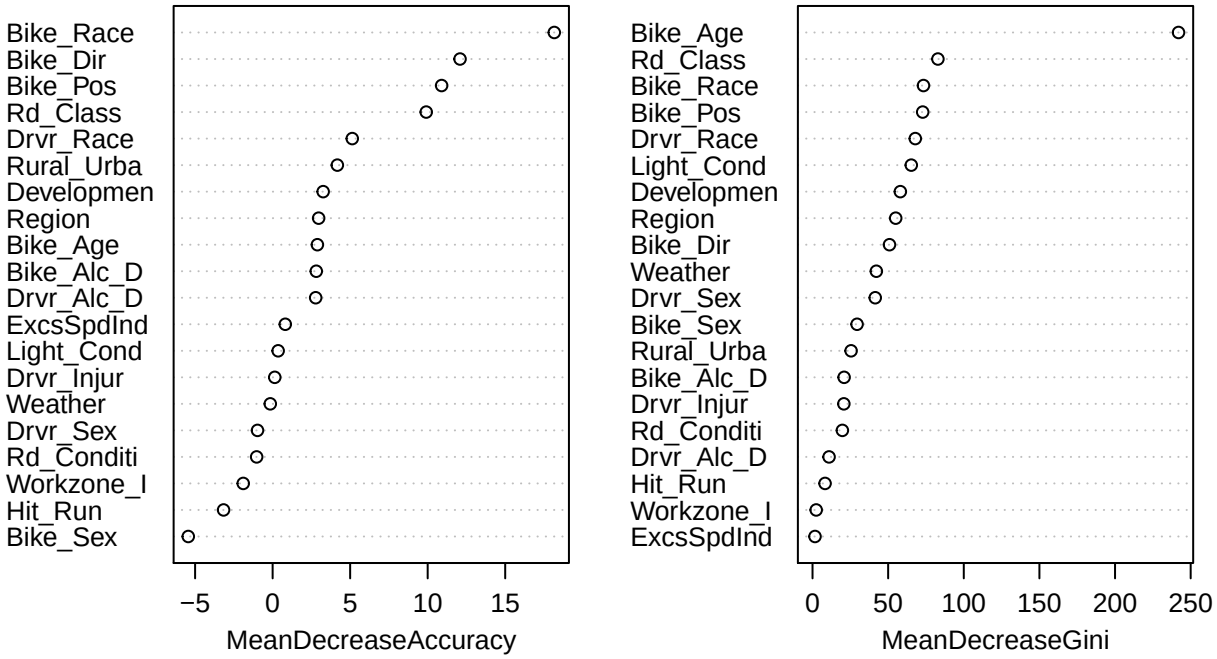
3 Bagging et Random forest

Dans cette partie nous allons comparer la méthode *RandomForest* classique avec celle du *Bagging*. Nous allons d'abord construire ces modèles sur les données d'apprentissage, comparer les résultats et enfin nous les appliquerons sur nos données test avant de conclure.

3.1 Random forest

L'algorithme Random Forest, est l'un des plus couramment utilisé, il s'agit d'un type spécial de bagging appliqué aux arbres de décision. Cet algorithme de forêt aléatoire de Breiman(basé sur le code Fortran original de Breiman et Cutler) combine les concepts de sous-espaces aléatoires et de bagging. L'algorithme des forêts d'arbres décisionnels effectue un apprentissage sur de multiples arbres de décision entraînés sur des sous-ensembles de données légèrement différents.

Commencons par déterminer l'importance des variables selon la méthode de *Mean Decrease Accuracy*.



Ce graphique montre l'importance qu'a chaque variable dans la construction de notre forêt. On peut noter que les variables : *Bike_Race,Bike_Dir, Bike_Pos* et *Rd_Class* sont les plus importantes.

En construisant la forêt aléatoire sur nos données d'apprentissage on obtient :

Table 5 – Matrice de confusion Random Forest : apprentissage

Prédiction	Référence		
	Non	Oui	class.error
Non	713	695	0.4936080
Oui	572	969	0.3711875

Cette méthode permet d'obtenir une erreur OOB (Out Of Bag) de 42.96%.

3.2 Bagging

Bagging signifie bootstrap aggregating. C'est un méta-algorithme d'ensemble d'apprentissage automatique conçu pour améliorer la stabilité et la précision des algorithmes d'apprentissage automatique utilisés dans la classification statistique et la régression.

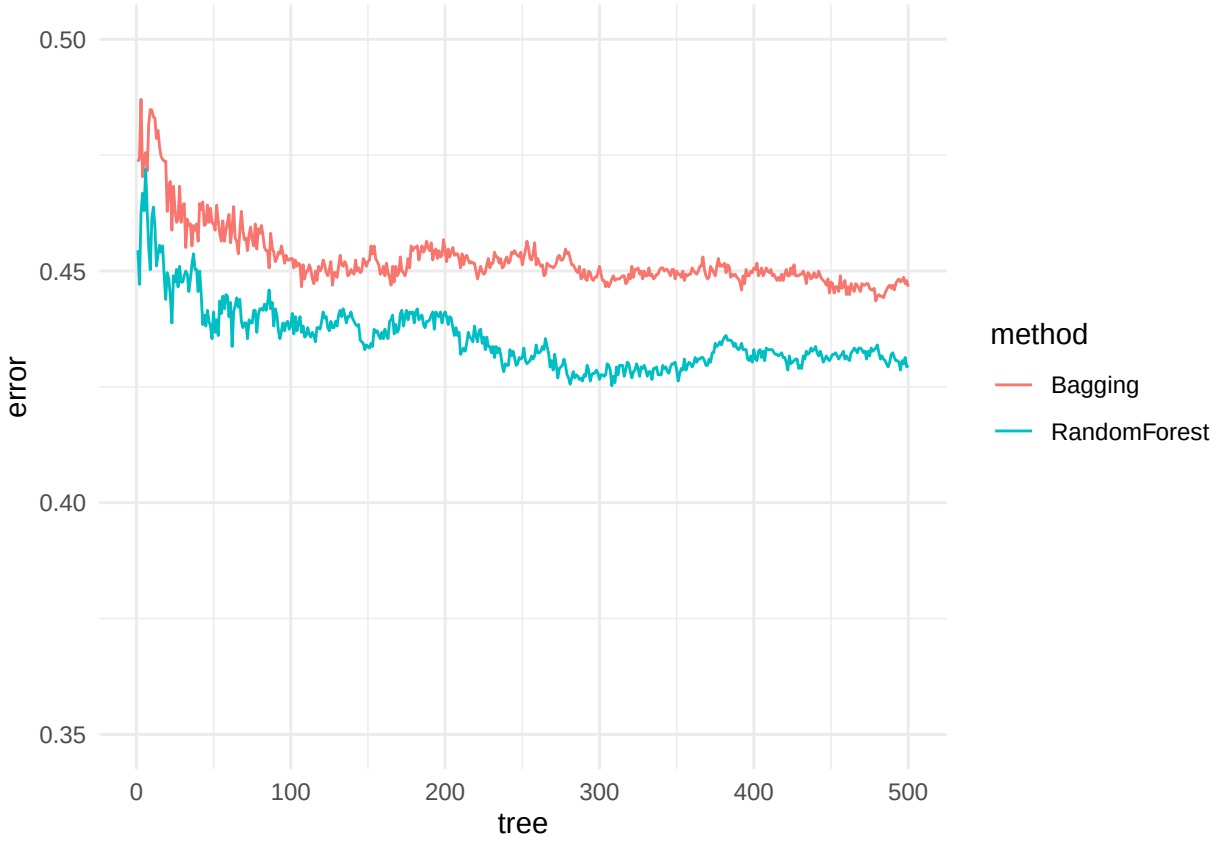
Table 6 – Matrice de confusion bagging : apprentissage

Prédiction	Référence		
	Non	Oui	class.error
Non	700	708	0.5028409
Oui	609	932	0.3951979

En utilisant le Bagging : on obtient une erreur OOB (Out Of Bag) égale à 44.6% sur nos données d'entraînement.

3.3 Comparaison bagging et Random forest

Comparons les niveaux d'erreurs des deux modèles. Pour 500 arbres, on a :



En comparant les erreurs sur les données d'entraînement des 2 modèles pour 500 arbres, on remarque que l'erreur provenant du *Random Forest* est plus faible que l'erreur du bagging quelque soit le nombre d'arbre considéré.

3.4 Application des modèles à l'échantillon Test

TABLE 7 – Matrice de confusion test

Prédiction	Référence			
	Random Forest		Bagging	
	Non	Oui	Non	Oui
Non	358	277	356	294
Oui	349	490	351	473

Les mesures de performances de ce modèle sont les suivantes :

TABLE 8 – Mesures de performance

	Sensitivity	Specificity	Precision
Random Forest	0.5063649	0.6388527	0.5637795
Bagging	0.5035361	0.6166884	0.5476923

Conclusion : Le Bagging permet un taux de précision superieur à celui du Random Forest.

4 Modèle randomForest automatisé

En laissant le logiciel déterminer les paramètres optimaux pour le modèle Random Forest, on obtient les paramètres suivants : 100 noeuds, une variable et 250 arbres.

En appliquant ces paramètres et en construisant le modèle sur notre échantillon d'apprentissage, on obtient la matrice de confusion :

Table 9 – Matrice de confusion Random Forest automatisé : apprentissage

Prédiction	Référence		
	Non	Oui	class.error
Non	542	866	0.6150568
Oui	333	1208	0.2160934

4.1 Application à l'échantillon test

En appliquant ce modèle à l'échantillon test on obtient :

Table 10 – Matrice de confusion Random Forest automatisé : test

Prédiction	Référence	
	Non	Oui
Non	267	160
Oui	440	607

Les mesures de performances de ce modèle sont les suivantes :

Table 11 – Mesures de performance

Sensitivity	Specificity	Precision
0.3776521	0.791395	0.6252927

Conclusion : Les performances de ce modèle semblent bien meilleures que tous les précédents.

5 Boosting

Le *Boosting* génère une séquence de modèles de classification, chaque modèle de classification successif dans la séquence permettant de mieux prévoir la classification des observations qui était mal classée par les modèles de classification précédents. Lors du déploiement, les prévisions issues des différents modèles de classification pourront alors être combinées afin d'obtenir la meilleure prévision ou classification. Le Boosting, est similaire à la méthode bagging. En revanche, les étapes se produisent de manière séquentielles et non pas simultanées.

En construisant le boosting, on obtient la matrice de confusion suivante :

Table 12 – Matrice de confusion apprentissage : Boosting

Référence	Prédiction	
	Non	Oui
Non	1387	21
Oui	20	1521

L'erreur Out of bag est de 0.019.

On constate, comme on pouvait s'y attendre, un problème de surapprentissage. On décide donc d'appliquer différentes pénalisations à notre modèle et ce afin de trouver le modèle optimal.

Les pénalisations généralement utilisées sur le Boosting sont : 0.1, 0.01 et 0.001.

Pénalisation = 0.1

L'erreur Out of bag est de 0.398.

Table 13 – Matrice de confusion apprentissage : Boosting p=0.1

Référence	Prédiction	
	Non	Oui
Non	724	684
Oui	477	1064

Pénalisation = 0.01

Table 14 – Matrice de confusion apprentissage : Boosting p=0.01

Référence	Prédiction	
	Non	Oui
Non	450	958
Oui	245	1296

L'erreur Out of bag est de 0.408.

Pénalisation = 0.001

Table 15 – Matrice de confusion apprentissage : Boosting p=0.001

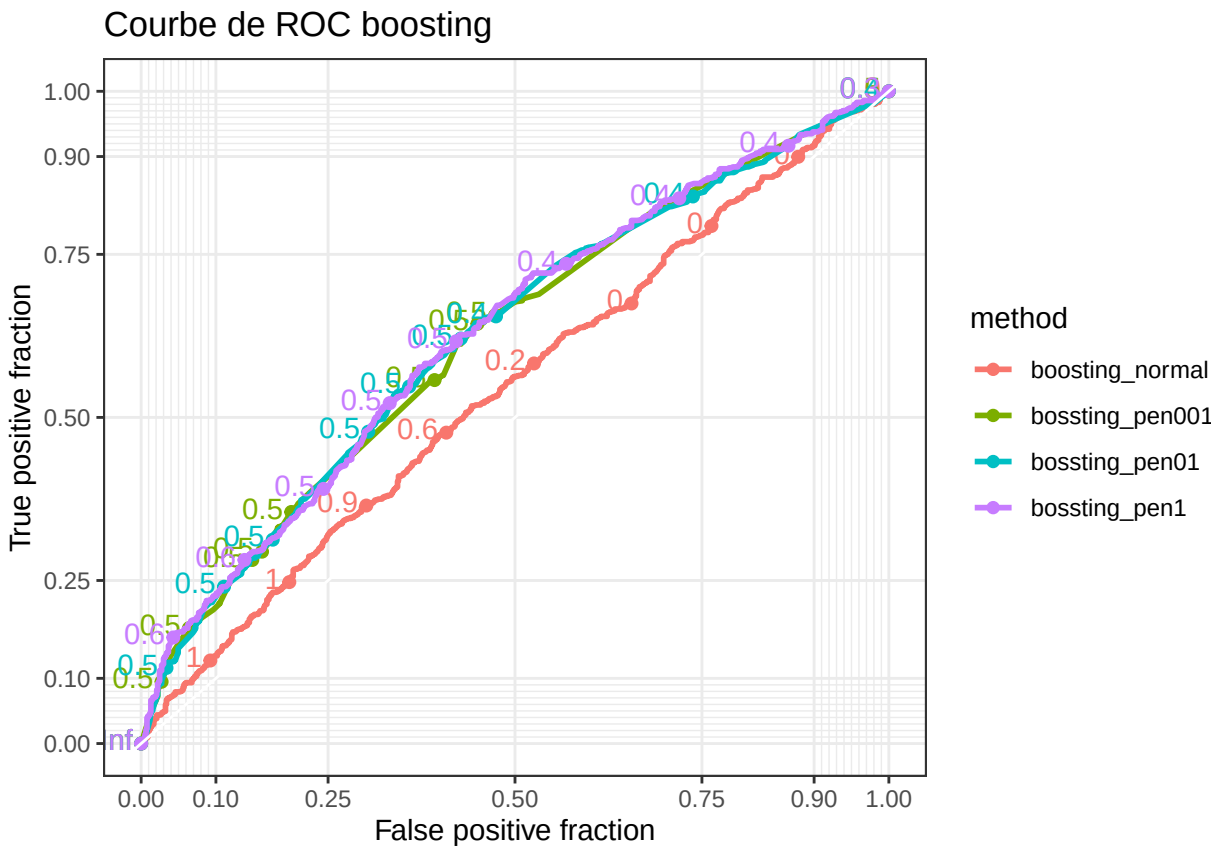
Référence	Prédiction	
	Non	Oui
Non	451	957
Oui	247	1294

L'erreur Out of bag est de 0.409.

Afin de déterminer le modèle boosting optimal, on va appliquer chacun d'entre eux à l'échantillon test. Le meilleur modèle sera celui permettant la meilleure prévision.

5.1 Application à l'échantillon test

Pour déterminer le meilleur modèle Boosting on va comparer les courbes ROC ainsi que les mesures de performance.



On doit, avant de conclure, comparer les mesures de performance des tests :

TABLE 16 – Comparaions des boosting

	Sensibilité	Spécificité	Précision	AUC
Boosting	0.4992928	0.5697523	0.5168375	0.5429685
Boosting p=0.1	0.4865629	0.6923077	0.5931034	0.6283772
Boosting p=0.01	0.2842999	0.8487614	0.6340694	0.6240178
Boosting p=0.001	0.2842999	0.8474576	0.6320755	0.6208118

Conclusion : Les AUC étant très similaires, pour le choix du meilleur modèle Boosting on se base sur la Précision : c'est donc le Boosting pénalisé à 0.01 que nous considérerons comme étant le meilleur modèle Boosting.

6 Scoring

Comme toute bonne démarche de modélisation, la construction d'un bon **score** se fait par une succession d'étapes : nous commencerons par vérifier la liaison entre les descripteurs, ensuite nous construirons les modèles sur l'*échantillon d'apprentissage* et enfin nous les appliquerons sur l'*échantillon test*. Nous pourrons ensuite comparer les mesures de performance afin de déterminer le meilleur modèle.

Nous comparerons plusieurs modèles et retiendrons le modèle le plus adéquat selon l'objectif de l'étude.

6.1 Variable à expliquer

Revenons sur nos données initiales. La variable **Bike_Injur** est séparée en plusieurs modalités, comme suit :

A: Disabling Injury	B: Evident Injury	C: Possible Injury	Injury	K: Killed	O: No Injury
291	2405	2199	172	123	526

Nous allons affecter à ces modalités les valeurs 0 et 1.

La valeur 1 : à celles qui concernent les blessures avérées et graves (Killed, Disabling Injury, Evident Injury et Injury) et la valeur 0 à celles qui concernent l'absence, évidente ou non, de blessure (No Injury et Possible Injury).

On obtient donc :

	Bike_Injur	
	0	1
Effectif	2725	2991

On souhaite déterminer la probabilité qu'un cycliste soit bléssé ou non compte tenu des paramètres de son accident.

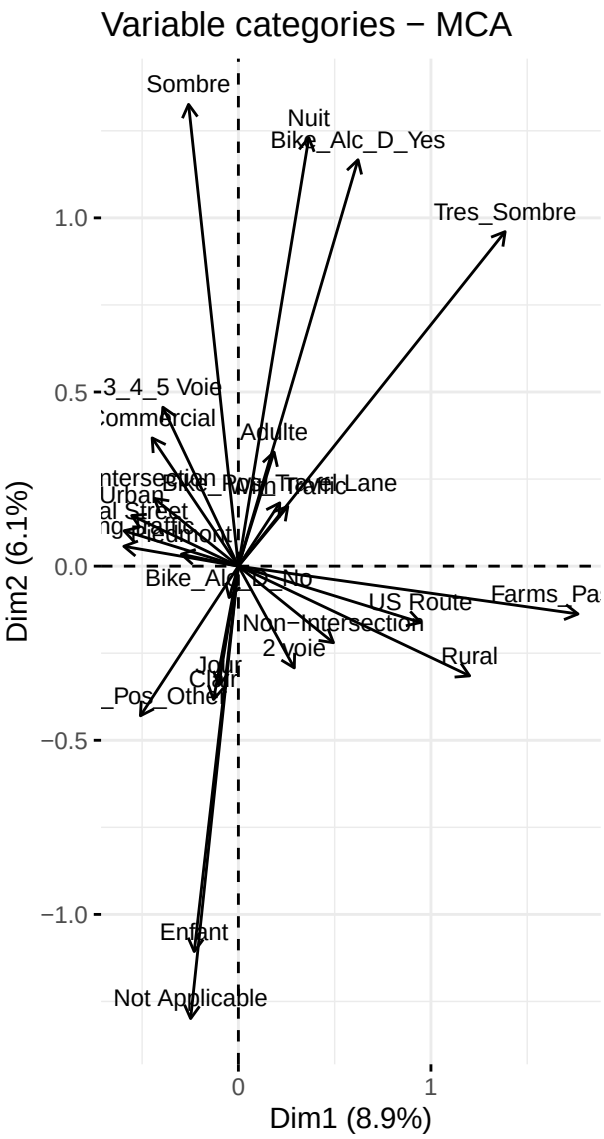
Pour cela, nous utiliserons des ***Régression logistique***.

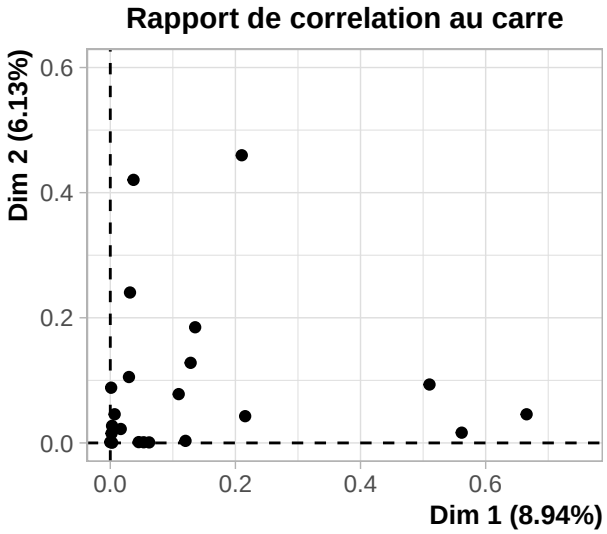
Nous allons effectuer une sélection de variables et ce pour plusieurs raisons.

D'abord parce que certaines des variables de notre base de données sont inutilisables en l'état et d'autre part parce qu'un modèle avec peu de variables sera plus facilement généralisable en terme de robustesse : *Principe du rasoir d'Occam*.

6.2 Analyse exploratoire

Construisons une ACM afin d'essayer d'identifier les corrélations éventuelles entre les variables explicatives.





Plusieurs variables semblent très corrélées, ce qui pourrait entraîner des problèmes de colinéarité lors de la régression.

6.3 Construction des modèles

Le **modèle général** que nous allons construire est un modèle naïf. Il prend en compte toutes les variables, sans aucune spécification particulière :

$$Bike\_Injur_i = \beta_0 + \beta_1\ Bike\_Age_i + \beta_2\ Bike\_Alc\_D_i + \beta_3\ Bike\_Dir_i + \beta_4\ Bike\_Pos_i + \beta_5\ Bike\_Race_i + \beta_6\ Bike\_Sex_i + \beta_7\ Crash\_Hour_i + \beta_8\ Crash\_Loc_i + \beta_9\ Developmen_i + \beta_{10}\ Drvr\_Alc\_D_i + \beta_{11}\ Drvr\_Injur_i + \beta_{12}\ Drvr\_Race_i + \beta_{13}\ Drvr\_Sex_i + \beta_{14}\ Hit\_Run_i + \beta_{15}\ Light\_Cond_i + \beta_{16}\ Num\_Lanes_i + \beta_{17}\ Rd\_Class_i + \beta_{18}\ Rd\_Conditi_i + \beta_{19}\ Region_i + \beta_{20}\ Rural\_Urba_i + \beta_{21}\ Workzone\_I_i + \varepsilon_i$$

Nous savons que ce modèle sera très mauvais étant donné le nombre de variable utilisée. Pour obtenir un modèle pertinent nous devons effectuer une sélection des variables et ce pour plusieurs raisons, la principale étant que : un modèle avec peu de variables sera plus facilement généralisable en terme de robustesse *Principe du rasoir d'Occam.*

Le second modèle, le **modèle AIC** : sera obtenu en faisant une sélection automatique de variables, sur le critère d'Akaïke (AIC). Ce dernier s'écrit comme suit: $AIC = 2k - 2\ln(L)$; où k est le nombre de paramètres à estimer du modèle et L est le maximum de la fonction de vraisemblance du modèle. Si l'on considère un ensemble de modèles candidats, le modèle choisi sera celui qui aura la plus faible valeur d'AIC.

Le dernier modèle, le **modèle AIC modifié** sera issue du second. On supprimera toutes les variables non significatives du second modèle.

Les différents résultats des régréssions obtenus sont renseignés dans le tableau çi-dessous :

TABLE 17 – Résultats

	Dependent variable:		
	Bike_Injur		
	(1)	(2)	(3)
Bike_AgeEnfant			0.001
			(0.127)
Bike_AgeJeune			−0.009
			(0.089)
Bike_Alc_DYes			0.024
			(0.163)
Bike_DirNot Applicable	0.676***	0.709***	0.673***
	(0.124)	(0.123)	(0.165)
Bike_DirWith Traffic	0.456***	0.486***	0.512***
	(0.082)	(0.081)	(0.103)
Bike_PosTravel Lane	0.301***	0.316***	0.262***
	(0.080)	(0.080)	(0.101)
Bike_RaceOther	0.470***	0.473***	0.436***
	(0.119)	(0.119)	(0.146)
Bike_RaceWhite	0.495***	0.480***	0.450***
	(0.071)	(0.070)	(0.093)
Bike_SexMale			−0.001
			(0.111)
Crash_HourNuit			0.083
			(0.099)
Crash_LocNon-Intersection			0.077
			(0.087)
DevelopmenFarms_Pastures	0.195		0.077
	(0.121)		(0.151)
DevelopmenInstitutional	0.263		0.330
	(0.195)		(0.236)
DevelopmenResidential	0.206***		0.228**
	(0.071)		(0.091)
Drvr_Alc_DNo	0.408	0.393	0.111
	(0.675)	(0.670)	(0.717)
Drvr_Alc_DYes	1.300*	1.288*	1.223
	(0.722)	(0.717)	(0.795)
Drvr_Injur1			−0.369
			(0.259)
Drvr_RaceBlack			−0.404
			(0.632)
Drvr_RaceOther			−0.424
			(0.644)
Drvr_RaceWhite			−0.371
			(0.628)
Drvr_SexMale			0.078
			(0.079)
Rd_ConditiWet			0.136
			(0.150)
RegionMountains			0.196
			(0.160)
RegionPiedmont			0.140
			(0.090)
Rural_UrbaUrban	−0.185**	−0.242***	−0.208*
	(0.083)	(0.070)	(0.110)
Workzone_IYes	0.062		−1.207*
	(0.473)		(0.698)
Constant	−1.218*	−1.066	−0.687
	(0.684)	(0.677)	(0.971)
Observations	4,191	4,191	2,794
Log Likelihood	−2,796.461	−2,801.077	−1,855.710
Akaike Inf. Crit.	5,618.923	5,620.154	3,765.419
Note:			*p<0.1; **p<0.05; ***p<0.01

Modèle AIC:

$$Bike\_Injur_i = \beta_0 + \beta_1\ Bike\_Alc\_D_i + \beta_2\ Bike\_Dir_i + \beta_3\ Bike\_Pos_i + \beta_4\ Bike\_Race_i + \beta_9\ Developmen_i + \beta_5\ Drvr\_Alc\_D_i + \beta_6\ Rural\_Urba_i + \beta_7\ Workzone\_I_i + \varepsilon_i$$

Modèle AIC modifié:

$$Bike\_Injur_i = \beta_0 + \beta_1\ Bike\_Alc\_D_i + \beta_2\ Bike\_Dir_i + \beta_3\ Bike\_Pos_i + \beta_4\ Bike\_Race_i + \beta_5$$

$$Drvr\_Alc\_D_i + \beta_6\ Rural\_Urba_i + \varepsilon_i$$

6.4 Validation des modèles : Indicateurs de qualité et de robustesse

On exclu le modèle général car la plupart des coefficients ne sont pas significatifs.

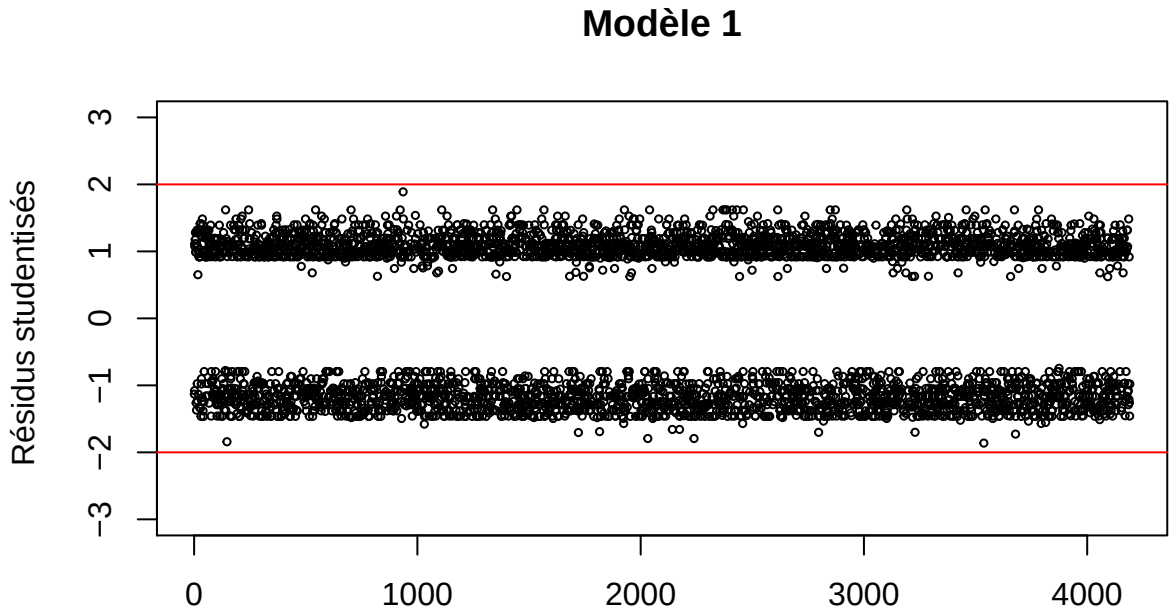
On s'intéressera donc exclusivement aux modèles AIC (Régréssion 1) et AIC modifié (Régréssion 2).

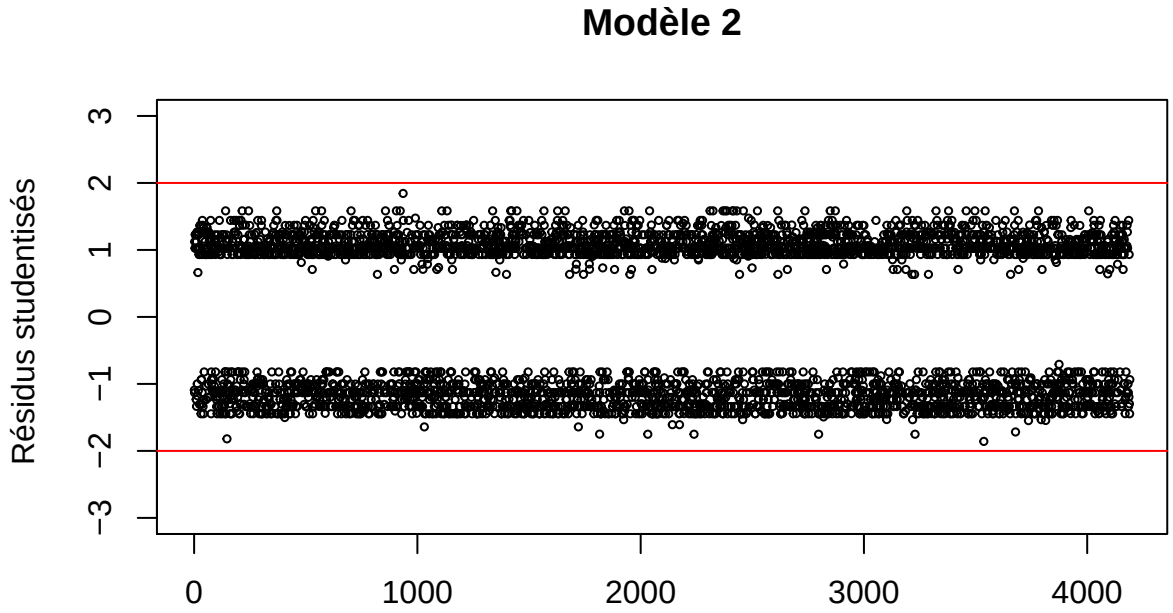
TABLE 18 – Résultats des modèles 2 et 3

	Dependent variable:	
	Bike_Injur	
	(1)	(2)
Bike_DirNot Applicable	0.676***	0.709***
	(0.124)	(0.123)
Bike_DirWith Traffic	0.456***	0.486***
	(0.082)	(0.081)
Bike_PosTravel Lane	0.301***	0.316***
	(0.080)	(0.080)
Bike_RaceOther	0.470***	0.473***
	(0.119)	(0.119)
Bike_RaceWhite	0.495***	0.480***
	(0.071)	(0.070)
DevelopmenFarms_Pastures	0.195	
	(0.121)	
DevelopmenInstitutional	0.263	
	(0.195)	
DevelopmenResidential	0.206***	
	(0.071)	
Drvr_Alc_DNo	0.408	0.393
	(0.675)	(0.670)
Drvr_Alc_DYes	1.300*	1.288*
	(0.722)	(0.717)
Rural_UrbaUrban	−0.185**	−0.242***
	(0.083)	(0.070)
Workzone_IYes	0.062	
	(0.473)	
Constant	−1.218*	−1.066
	(0.684)	(0.677)
Observations	4,191	4,191
Log Likelihood	−2,796.461	−2,801.077
Akaike Inf. Crit.	5,618.923	5,620.154
Note:	*p<0.1; **p<0.05; ***p<0.01	

Résidus de déviances

Pour les régressions logistiques, on s'intéresse la plupart du temps aux résidus de déviance. Ils prennent généralement des valeurs qui oscillent entre -2 et 2.





Il semblerait qu'il n'y ait pas de valeurs aberrantes.

Les deux modèles sont donc utilisables dans l'état.

6.5 Application à l'échantillon test

En appliquant les deux modèles à l'*échantillon Test* on a :

Table 19 – Matrice de confusion Scoring : test

Prédiction	Référence				
	Modèle AIC			Modèle AIC modifié	
	0	1		0	1
0	349	255		325	222
1	318	475		342	508

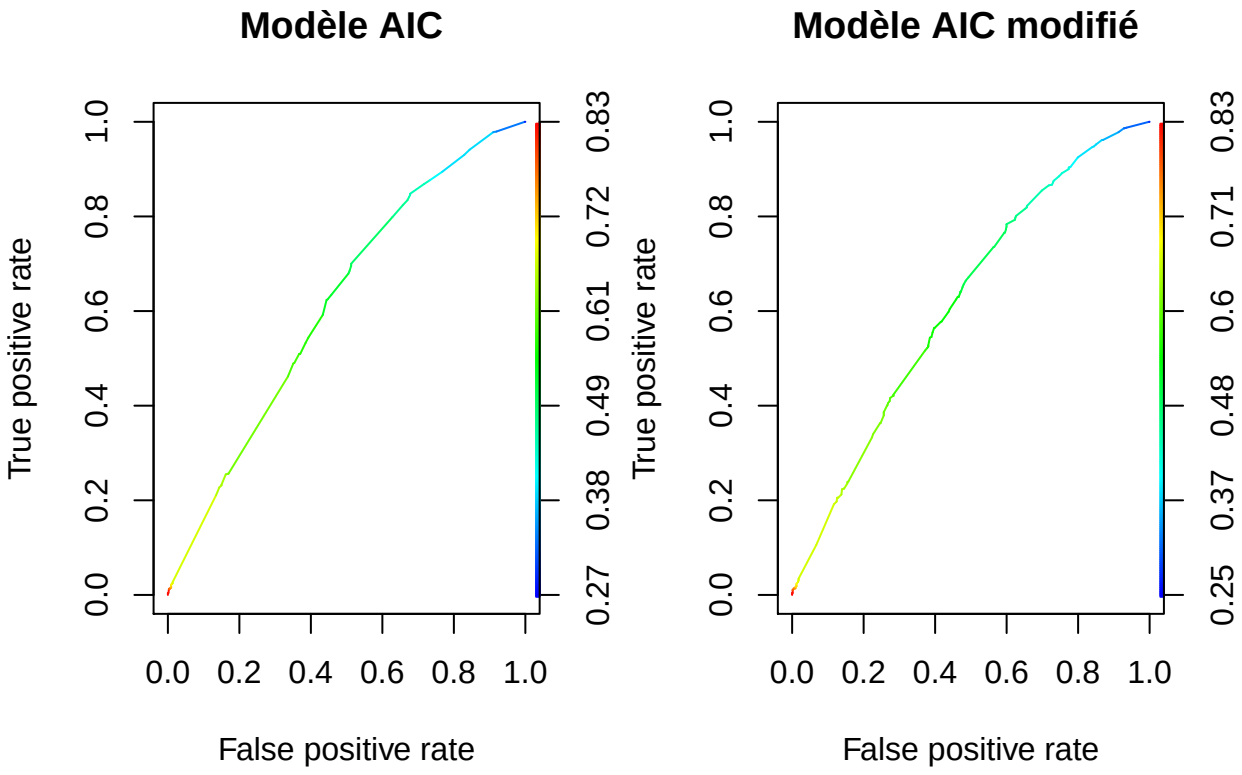
Qu'en est-il du taux d'erreur ?

Table 20 – Taux d'erreur test

Modèle AIC	Modèle AIC modifié
0.4101646	0.4037223

Le taux d'erreur est relativement le même pour les deux modèles. Il est inférieur à 0.5, ce qui est suffisant pour conclure que les modèles sont pertinents.

Courbes ROC



Aire sous les courbes

Table 21 – Aire sous les courbes

Modèle AIC	Modèle AIC modifié
0.6167362	0.6136391

L'aire sous les courbes est relativement le même lui aussi.

Mesure de performance :

Table 22 – Mesure de performance : Scoring

	Sensibilité	Spécificité	Précision
Modèle AIC	0.5232384	0.6506849	0.5778146
Modèle AIC modifié	0.4872564	0.6958904	0.5941499

Conclusion : Les deux modèles sont très similaires et ont des mesures de performances qui se valent mais on décide de privilégié celui étant le plus précis : le modèle AUC modifié.

7 Choix du meilleur modèle

7.1 Rappel et comparaison

Nous avons, au cours de cette étude, construit différents modèles sur la base de divereses méthodes d'apprentissage supervisé. Pour chacune de ces méthodes nous avons construit différents modèles sur l'*échantillon apprentissage* en fonction de différents paramêtres. Nous avons ensuite appliqué ces modèles sur l'*échantillon test* afin de déterminer le meilleur modèle de chaque famille.

Ces applications ayant été éffecuées sur le même échantillon, on peut en comparer les résultats et déterminer, en fonction des mesures de performances, quel modèle permet de mieux prédire.

Les mesures de performance des différents modèles que nous avons construit sont résumé dans le tableau suivant :

Table 23 – Comparaison des mesures de performance des modèles

	Sensibilité	Spécificité	Précision
Arbre complet	0.5021216	0.5945241	0.5330330
Arbre élagué	0.4992928	0.6101695	0.5414110
Random Forest	0.5063649	0.6388527	0.5637795
Bagging	0.5035361	0.6166884	0.5476923
RandomForest automatisé	0.3776521	0.7913950	0.6252927
Boosting p=0.01	0.2842999	0.8487614	0.6340694
Scoring	0.4872564	0.6958904	0.5941499

7.2 Discussion

La sélection des mesures de performance les plus pertinentes se fait en fonction de la problématique à traiter. La notre pourrait être soit de déterminer la probabilité que l'accident ait causé une blessure soit de déterminé la probabilité que l'accident n'ait pas causé de blessure. Il n'y a pas grand intérêt à déterminer la probabilité que l'accident ait causé une blessure pour les hôpitaux par exemple car ces derniers envoient systématiquement une ambulance. En revanche il serait très intéressant pour eux de déterminer la probabilité que l'accident n'ait pas causé de blessures et ce afin de gérer le flux des ambulances ou simplement de faire un choix prioritaire parmi deux situations par exemple. En ce sens la sensibilité ne donne pas une mesure très pertinente. La spécificité en revanche, qui mesure le taux de vrais négatifs (dans notre cas qu'il n'y ait pas de blessure) semble plus pertinente.

Aussi, La précision est de fait une mesure très intéressante. Dans le cadre de notre étude : c'est la capacité de nos modèles à ne prédire non à une blessure si l'accident n'a effectivement pas entraîner une blessure. Pour évaluer un compromis entre sensibilité et précision, on peut calculer la "F-mesure", qui est leur moyenne harmonique.

Le calcul du F-mesure est le suivant:

$$F - mesure = \frac{2 \times Précision \times sensibilité}{précision + sensibilité}$$

On obtient donc :

Table 24 – F-mesure des différents modèles

	F-mesure
Arbre complet	0.5171158
Arbre élagué	0.5194996
Random Forest	0.5335320
Bagging	0.5246868
Random Forest automatisé	0.4708995
Boosting p=0.01	0.3925781
Scoring	0.5354201

Toutes les mesures de performances étant définies et calculées, quel sont les meilleurs modèles ?

Meilleur modèle

Ces nombreuses considérations ainsi prises en compte nous permettent de conclure que les modèles : *RandomForest automatisé* (qui pour rappel est un modèle Random Forest ayant les paramètres suivant : 250 arbres, une variable et 50 nœuds) et *Boosting* pénalisé à 0.01 sont les modèles permettant d'obtenir les meilleures résultats pour la problématique : quelle est la probabilité que l'accident n'est pas engendré de blessures.