Banking

BOUSSENGUI François - LEGER Aline - BARRE Nicolas

Table des matières

1 Présentation de la base de données

1.1 Description de la base

- `Age` : Age du client ;
- `Job` : emploi du client ;
- `Statut_matrimonial` : Statut matrimonial du client ;
- `Education`: Niveau d'étude ;
- `Défaut` : Vaut Oui i le client est en défaut de paiment et non sinon ;
- `Balance`: Solde du compte courant;
- `Prêt immobilier`: Vaut Oui si l'individu a contracté un prêt immobilier et non sinon ;
- `Crédit` : Vaut Oui si l'individu a un crédit et non sinon ;
- `Contact` : Type de contact ;
- `Jour` : Jour de la semaine durant lequel est contacté le client ;
- `Mois` : Mois durant lequel est contacté le client ;
- `Durée` : ;
- `Campagne` : Nombre de personnes contactées durant la campagne ;
- `Jour_suivant` : Nombre de jours passés après le dernier appel ;
- `Jour_précédent` : Nombre d'appels concluant passés avant la campagne ;
- `Dépôt` : Vaut yes si le client a souscrit à un compte dépôt et non sinon ;

Dans un premier temps, la variable que nous allons tenter de prédire est la variable "défaut". C'est une variable binaire valant "oui" si l'individu est en défaut de paiement et "non" sinon. Dans notre base de données, nous comptabilisons 98.1514836 % de personnes non en défaut de paiement contre seulement 1.8485164 % étant en défaut de paiement. La variable à prédire présente donc un fort déséquilibre ce qui influera probablement sur les modèles construits.

Nous allons donc dans un premier temps appliquer les méthodes de prédiction sur les données brutes et ensuite tenter d'améliorer sesprédictions en prenant le compte le déséquilibre entre les classes.

De plus, dans nos interprétations nous prendrons en compte les critères de performances non biaisés par le déséquilibre des classes.

Dans un second temps, nous allons étudier la variables concernant le fait d'avoir contracté un prêt immobilier ou pas.

Les critères de perfomances utilisés seront les suivants :

$$Precision = \frac{TP}{Prédictions\ positives}$$

$$Recall = \frac{TP}{Données\ positives}$$

$$F_1\ score = 2(\frac{Précision * sensibilité}{Précision\ + sensibilité})$$

2 Première application des méthodes

2.1 Analyse linéaire discriminante

Tout au long de notre exposé, nous allons utiliser les variables : âge, balance, durée, campagne, jour_précédent pour l'analyse linéaire discriminante et l'analyse quadratique discrimiante.

Table 1 – Matrice de confusion LDA

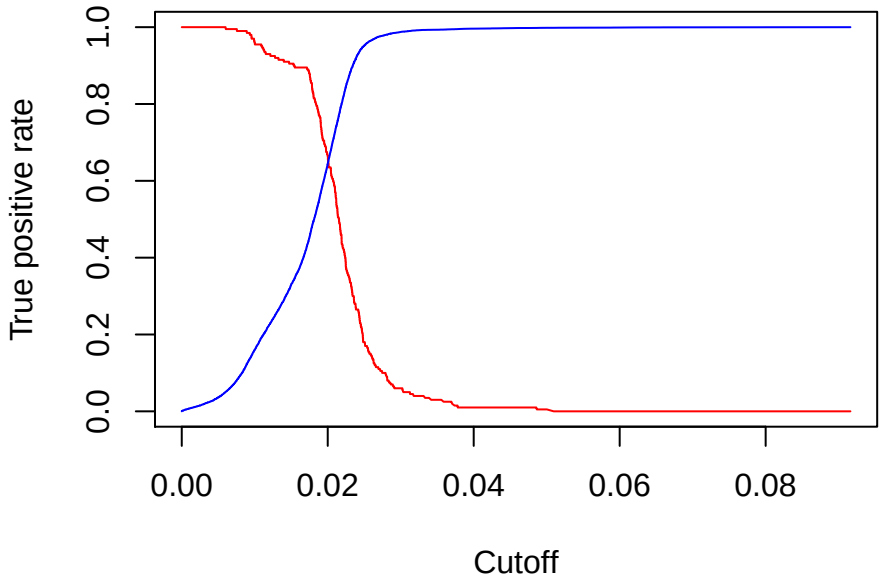
	no	yes	Sum
no	10349	0	10349
yes	200	0	200
Sum	10549	0	10549

Le taux de bonne affection de 98,1 %. L'erreur globale de classement est 1,9%. La précision du modèle est 0 % et le recall est de 0 %.

La méthode LDA permet d'obtenir un bon taux d'affectation global et une erreur globale très faible. Mais elle ne détecte pas du tout les individus en défaut de paiement tandis que c'est une classe que nous souhaitons également prédire au mieux.

Par défaut, l'affectation dans la catégorie "oui" a été effectué lorsque :

$$\mathbb{P}(défaut = oui \mid X = x) > 0.5$$



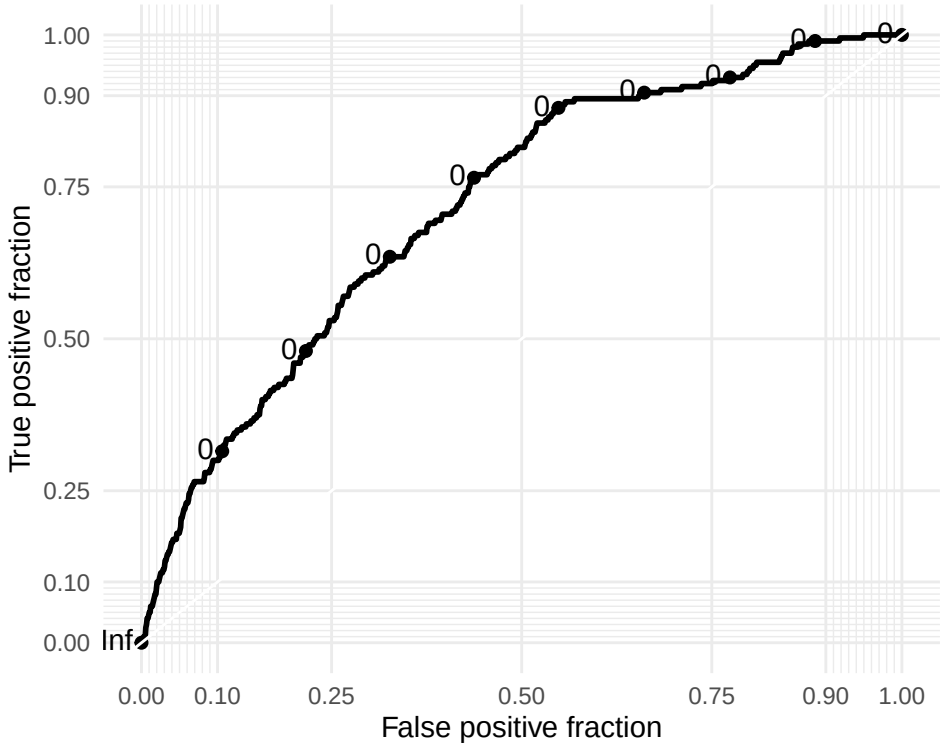
Au vue de ce graphique , nous décidons donc de baisser le seuil afin de mieux détecter les individus en défaut de paiment :

$$\mathbb{P}(défaut = oui \mid X = x) > 0.02$$

Table 2 – Matrice de confusion LDA

	No	Yes	Sum
no	6668	3681	10349
yes	67	133	200
Sum	6735	3814	10549

On voit que globalement le taux de bonne affectaion baisse et passe à 64.5 %. Le taux d'erreur global augmente pour atteindre 35.5 %. La précision est à présent 3.4 % et le recall, qui représente le taux de vrais positifs que l'on a correctement identifés est de 66.5 %. L'objectif est de trouver un compromis entre précision et recall. En effet, on voit qu'en abaissant le seuil, le recall augmente effectivement, mais la précision augmente très faiblement. Cela pourrait sans doute s'expliquer par le fait que nous ayons des classes déséquilibrées. Par conséquent, notre modèle prédit mal.



L'aire sous de la courbe ROC est de 0.74.

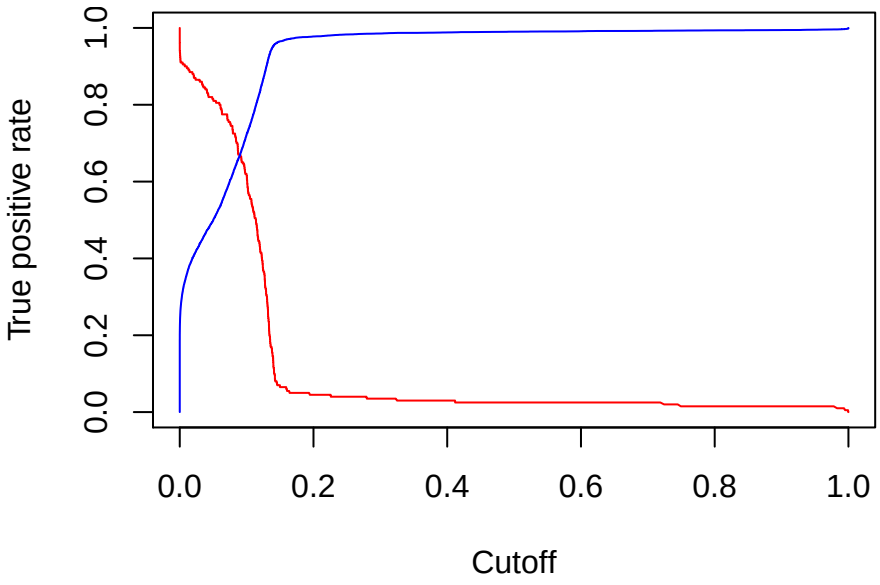
2.2 Analyse quadratique discriminante

TABLE 3 – Matrice de confusion QDA

	no	yes	Sum
no	10248	101	10349
yes	195	5	200
Sum	10443	106	10549

L'erreur globale est de 2.8 %. Le taux de bonne affectation est 98,2 %. Dans ce modèle la précision est de 4.7 % et le recall est de 2.5 % sachant que le seuil d'affection par défaut est :

$$\mathbb{P}(défaut = oui \mid X = x) > 0.5$$



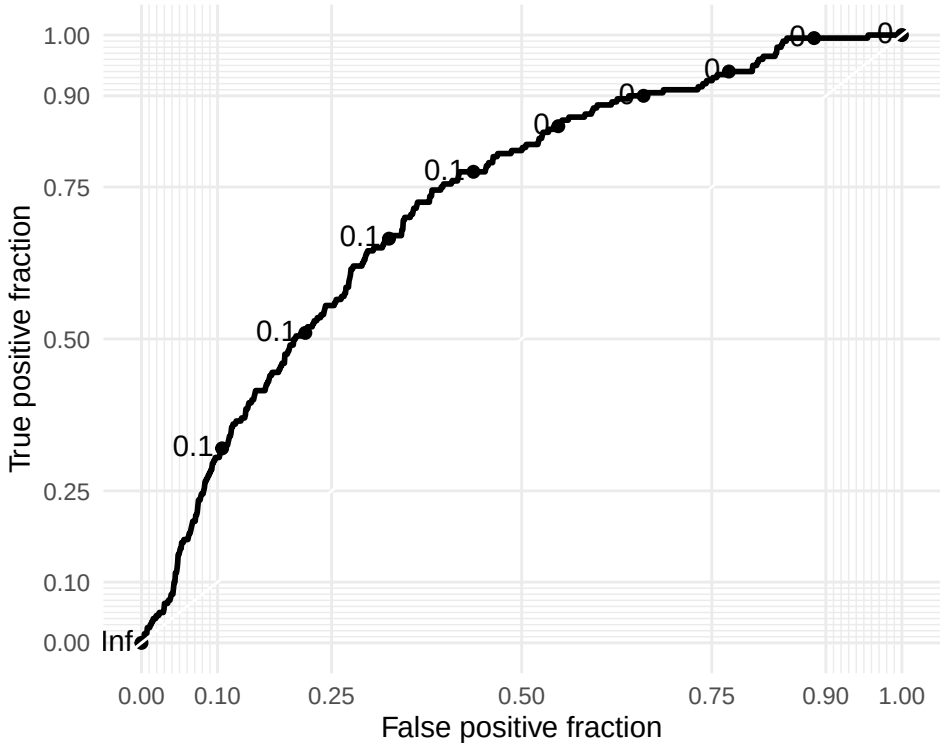
Au vue de ce graphique, nous allons à présent baisser le seuil à

$$\mathbb{P}(défaut \ = \ oui \ | X \ = \ x) > 0.08$$

Table 4 – Matrice de confusion QDA

	No	Yes	Sum
no	6454	3895	10349
yes	55	145	200
Sum	6509	4040	10549

L'erreur globale est maintenant de 37. 44 %. le taux de bonnne affectation est de 62.5 %. Dans ce modèle la précision est de 3.5% et la recall est de 72.5 %. De même qu'en analyse linéaire discriminante, l'objecfif est de trouver un compris entre précision et recall. On voit que le recall est élevé, ce qui est une bonne chose. Cependant, la précision est très faible. Comme précédemment, le fait que notre modèle ne soit pas du tout précis peut s'expliquer par le fait que nous ayons des classes déséquilibrées.



L'aire sous la courbe ROC est de 0.72.

Les variables utilisées pour les arbres de décisions, les forêts aléatoires et le boosting sont les suivantes : “age”,“job”,“marié”,“education”, immobilier”,“prêt”,“contact”,“mois”,“durée”,“campagne”,“jour_précédent” et “souscription”.

2.3 Arbres

2.3.1 Construction du modèle

Nous allons commencer par réaliser le premier arbre à savoir l’abre maximal. Ce n’est pas vraiment l’arbre maximal car nous avons ajouté des contraintes afin que cet arbre ne soit pas trop profond. Nous avons modifié différents paramètres :

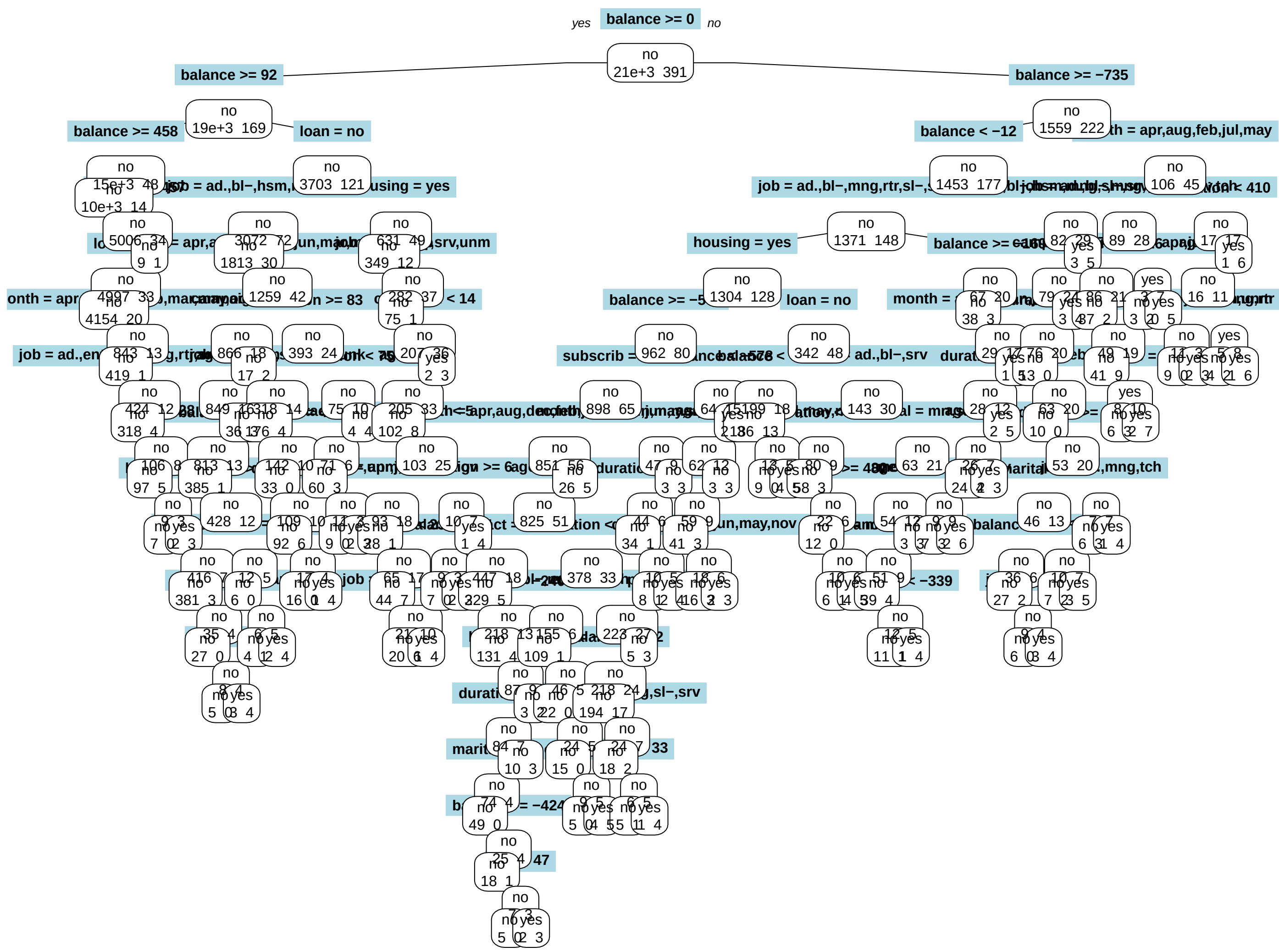
— minsplit : construit un arbre en continuant les découpages dans les feuilles qui contiennent au moins trente observations,
— minbucket : refuse segmentation ou un des noeuds enfants aurait moins de dix observations.

Nous avons mis aucune contrainte sur la qualité du découpage (cp).

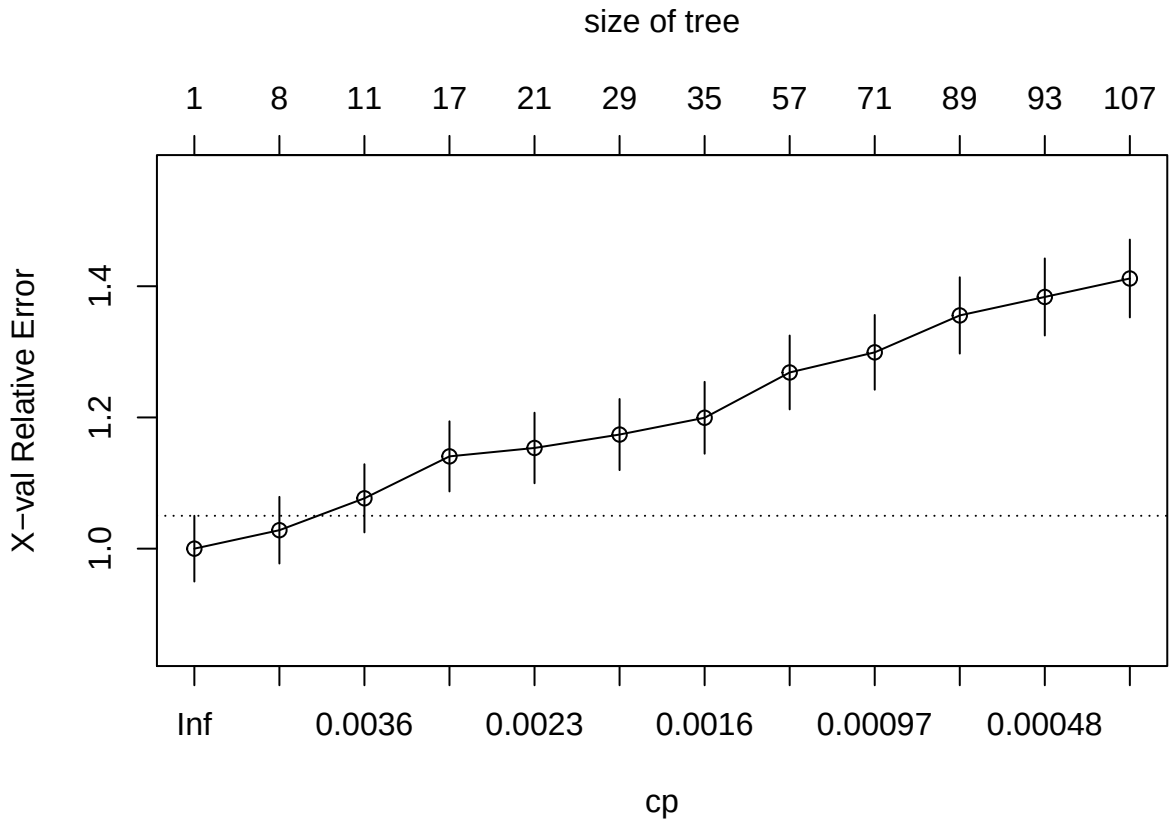
De plus, nous avons ajouté un argument pour la validation croisée :

— xval : nombre de blocs pour la validation croisée

Nous obtenons l’arbre suivant :



Le nombre de feuilles est trop important. Pour des soucis de lisibilité et d’over-fitting, nous allons élaguer cet arbre. Un élagage judicieux se fait ou l’on atteint un compromis entre complexité et précision de la prédiction. Ce compromis peut se fait grâce à la validation croisée testant différents versions élaguées de l’arbre. On peut utiliser le paramètre de complexité qui rend l’erreur de validation croisée petite.



Au vu de ce graphique, il est difficile de connaître la taille de l'arbre qui minimiserait l'erreur en validation croisée.

Afin de savoir, nous allons regarder le tableau suivant. On y trouve la valeur du paramètre de complexité, le nombre de segmentations, l'erreur sur l'échantillon apprentissage, l'erreur de validation croisée et l'écart type de cette erreur.

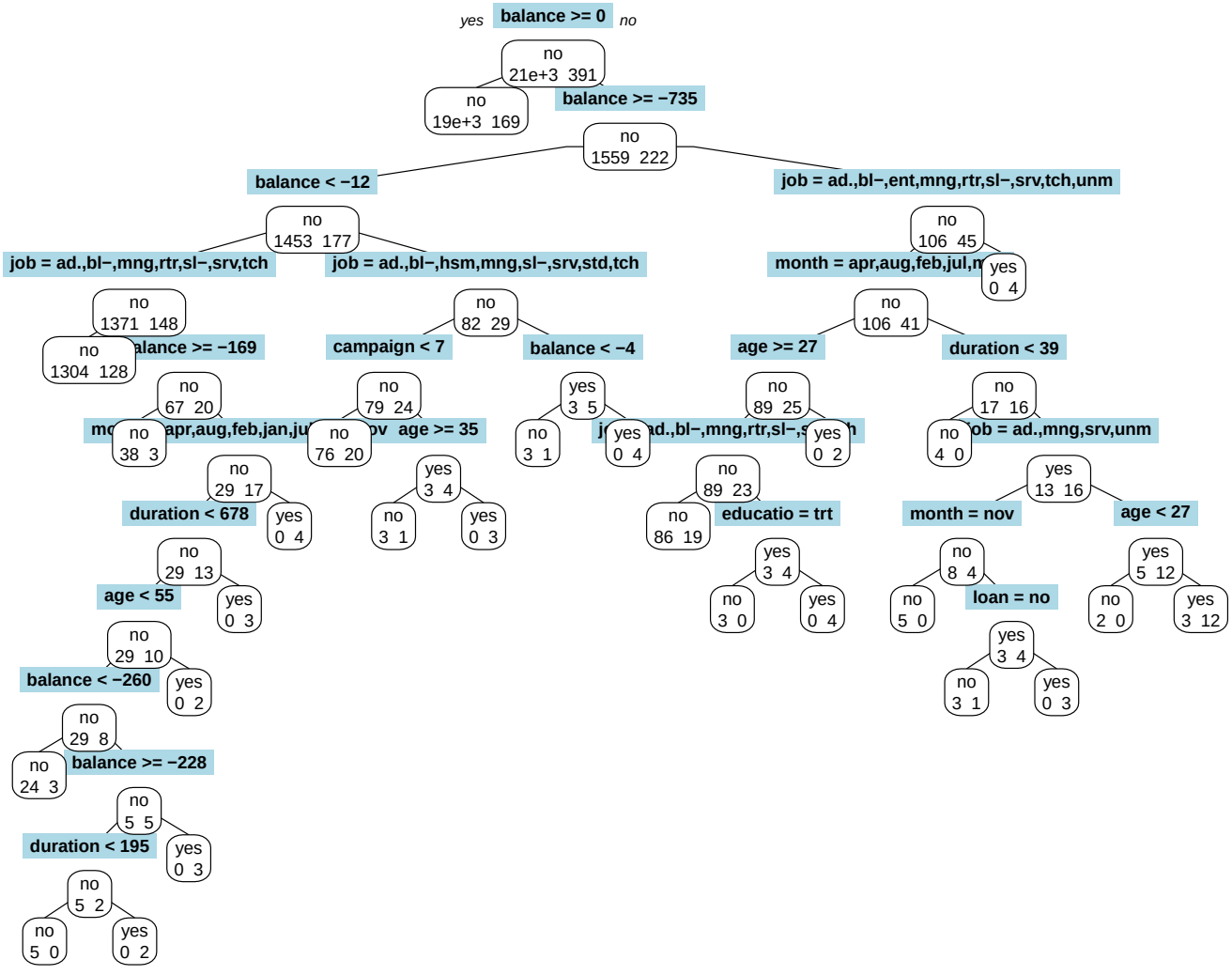
```
##              CP nsplit rel error   xerror       xstd
## 1  0.0046035806      0 1.0000000 1.000000 0.05010137
## 2  0.0042625746      7 0.9641944 1.028133 0.05078773
## 3  0.0030690537     10 0.9514066 1.076726 0.05195022
## 4  0.0025575448     16 0.9283887 1.140665 0.05343811
## 5  0.0020460358     20 0.9181586 1.153453 0.05373031
## 6  0.0019181586     28 0.9002558 1.173913 0.05419426
## 7  0.0012787724     34 0.8874680 1.199488 0.05476815
## 8  0.0010960906     56 0.8542199 1.268542 0.05628571
## 9  0.0008525149     70 0.8388747 1.299233 0.05694592
## 10 0.0006393862     88 0.8235294 1.355499 0.05813485
## 11 0.0003653635     92 0.8209719 1.383632 0.05871933
## 12 0.0000000000    106 0.8158568 1.411765 0.05929742
```

On peut voir que l'erreur de validation croisée est optimale pour un arbre composé d'un seul noeud ce qui n'est pas l'idéal pour nous car il ne nous donnerait aucune information. Or, on peut voir que pour un nombre de noeud égal à 6 l'erreur est plus élevée mais reste relativement faible. Il serait donc optimal d'élaguer à 6 noeuds. Le paramètre de complexité correpsondant est : 0.0030690537.

Table 5 –

	CP	nsplit	rel error	xerror	xstd
1	0.0046035806	0	1	1	0.050
2	0.0042625746	7	0.964	1.028	0.051
3	0.0030690537	10	0.951	1.077	0.052
4	0.0025575448	16	0.928	1.141	0.053
5	0.0020460358	20	0.918	1.153	0.054
6	0.0019181586	28	0.900	1.174	0.054
7	0.0012787724	34	0.887	1.199	0.055
8	0.0010960906	56	0.854	1.269	0.056
9	0.0008525149	70	0.839	1.299	0.057
10	0.0006393862	88	0.824	1.355	0.058
11	0.0003653635	92	0.821	1.384	0.059
12	0	106	0.816	1.412	0.059

L'arbre élagué est le suivant :



Concernant la construction de cet arbre, nous pouvons dire qu'une variable contribue très fortement c'est la variable "balance" suivie de la variable "job", "age" "month" et "duration". On retrouve effectivement que ces 5 variables dans l'arbre.

2.3.2 Prévisions et estimation de l'erreur

Notre arbre étant maintenant construit, nous allons estimer l'erreur sur l'échantillon test et observer les prédictions.

Table 6 – Importance des variables

	Importance
balance	74.937187
job	16.876209
age	15.802844
duration	14.098117
month	12.111665
education	4.355690
campaign	4.343302
loan	3.071429
contact	2.843105
housing	1.293747
subscribed	1.247583
poutcome	1.136235
marital	1.050668

Table 7 – Matrice de confusion

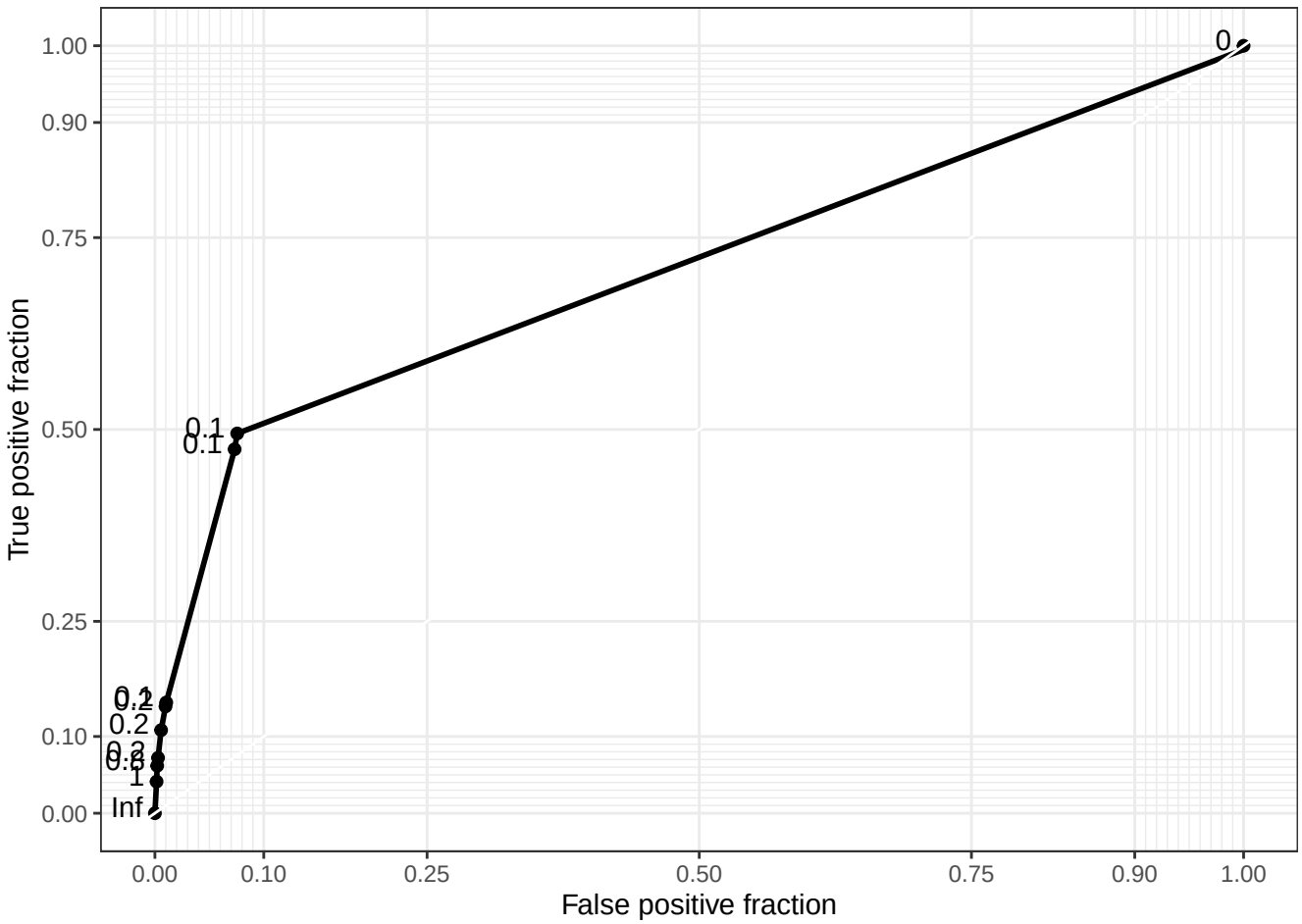
	No	Yes	Total
No	10333	182	10515
Yes	22	12	34
Total	10355	194	10549

La matrice de confusion nous donne les résultats suivants :

- taux de mauvaises prédictions : 1.8 %
- taux de bonnes prédictions : 98.1 %
- précision : 40.7 %
- recall : 5.6 %

Les non défauts de paiement sont très biens prédits (98% bien prédits) alors que les défauts de paiement le sont beaucoup moins (40% bien prédits).

L'erreur sur l'échantillon est de 0.02 soit 2 %.



L'aire sous la courbe est de 0.71.

2.4 Forêt aléatoire

2.4.1 Construction du modèle

TABLE 8 – Matrice de confusion

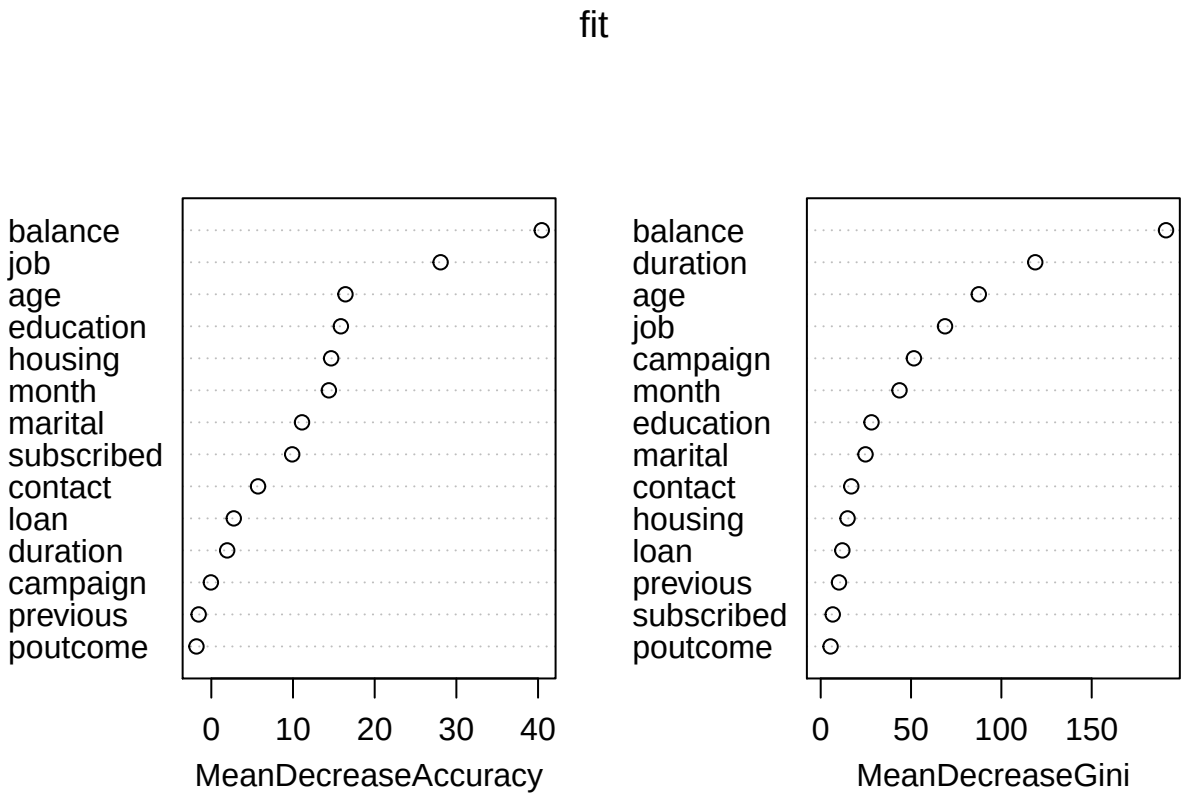
	No	Yes	Total
No	10354	194	10548
Yes	1	0	1
Total	10355	194	10549

La matrice donne les résultats suivants:

- taux de mauvaises prédictions : 1.8 %
- taux de bonnes prédictions : 98.1 %
- précision : 0 %
- recall : 0 %

Le modèle prédit très bien les non défauts de paiements mais pas du tout les défauts des paiements. Cela est surement dû à nos données déséquilibrées.

2.4.2 Importance des variables

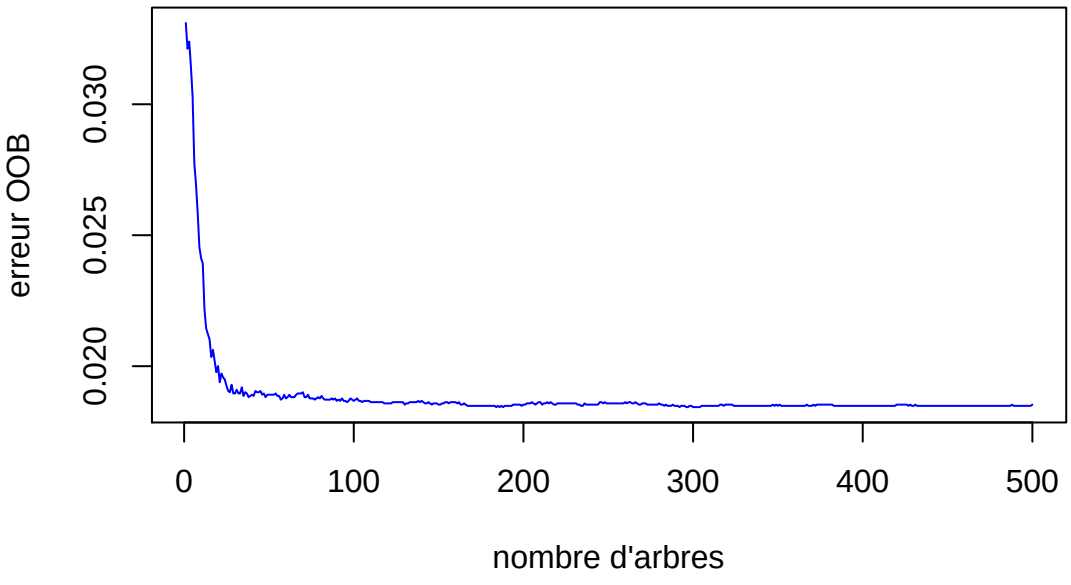


A l'aide du premier graphique, nous pouvons dire que les variables les plus importantes sont "balance" et "job". On remarque que deux variables ne sont pas du tout importante : poutcome et previous.

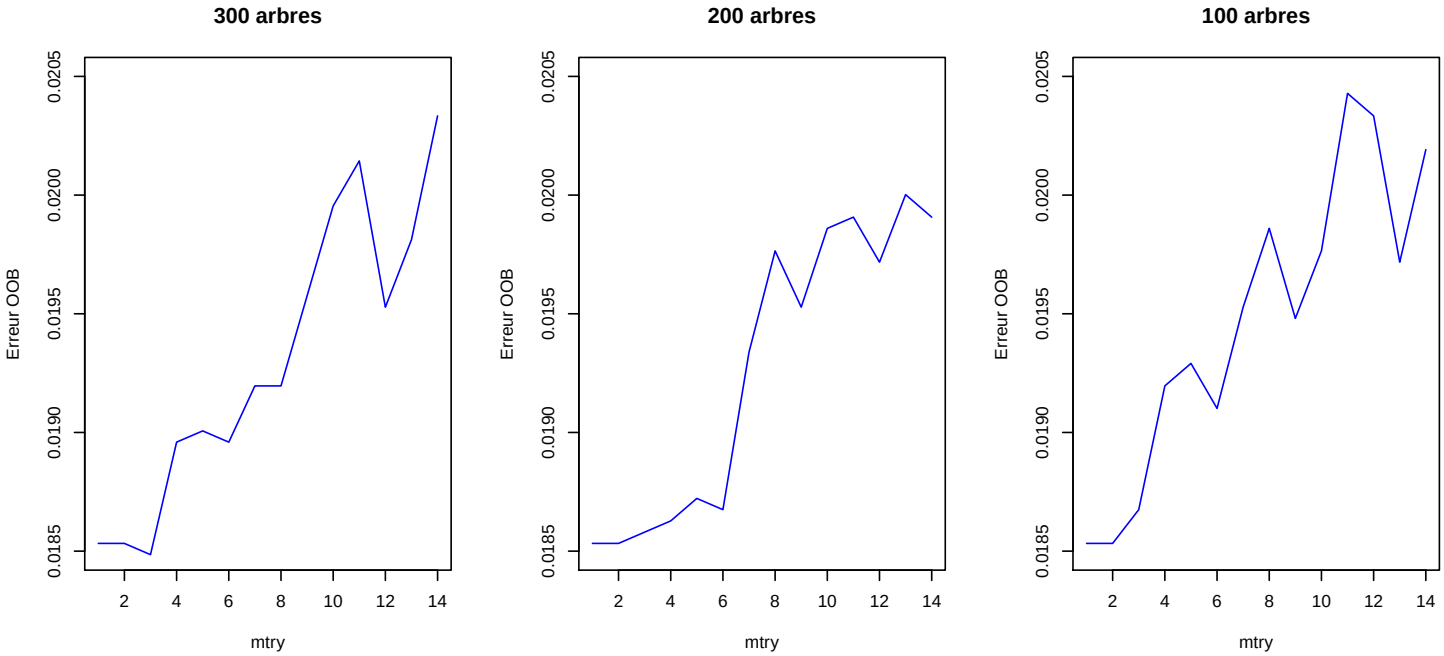
A l'aide du deuxième graphique, nous pouvons dire que les variables les plus importantes sont "balance" et "duration".

Aussi, nous avons utilisé le package VSURF afin de voir quelles variables sont les plus importantes à l'issue des trois étapes. Cette fonction a sélectionné les variables 2 et 5 à savoir job et balance comme étant les plus importantes et cela pour un nombre d'arbres égal à 100, 200 et 300.

2.4.3 Paramètres optimaux du modèle



Suite à ce graphique, nous pouvons voir que l'erreur out-of-bag se stabilise aux alentours de 200 arbres.



Sur les graphiques précédents, nous pouvons observer l'évolution de l'erreur OOB en fonction du nombre de variables :

- pour 300 arbres, le nombre de variables qui minimise l'erreur OOB est 3.
- pour 200 arbres, le nombre de variables qui minimise l'erreur OOB est 2.
- pour 100 arbres, le nombre de variables qui minimise l'erreur OOB est 2.

Suite à toutes ces observations, nous allons paramétrer notre modèle afin de pouvoir améliorer les prédictions.

2.4.4 Paramétrage du modèle

Afin d'améliorer les prévisions de notre modèle, nous allons jouer sur deux paramètres à savoir: le nombre de variables utilisées à chaque scission et le nombre d'arbres.

Concernant le nombre de variables, nous allons tester les valeurs : 2,3,6,8,9,12. Concernant le nombre d'arbres, nous allons tester 200 et 300.

TABLE 9 – Forêt aléatoire

	Paramètres		Résultats	
	Nombre de variables	Nombre d'arbres	Erreur OOB	Erreur test
1er modèle	2	100	1.85 %	1.08 %
2ème modèle	2	200	1.85 %	1.08 %
3ème modèle	**2**	**300**	**1.85 %**	**1.08 %**

Après avoir testé les différents nombre de variables, nous avons obtenu les résultats suivants. Pour un nombre d'arbre égal à 200 et 300, le nombre de variables optimal est de 2.

Notre meilleur arbre sera donc obetenu pour un nombe d'arbres égal à 300 et un nombre de variables à chaque scission égal à 2.

2.5 Boosting

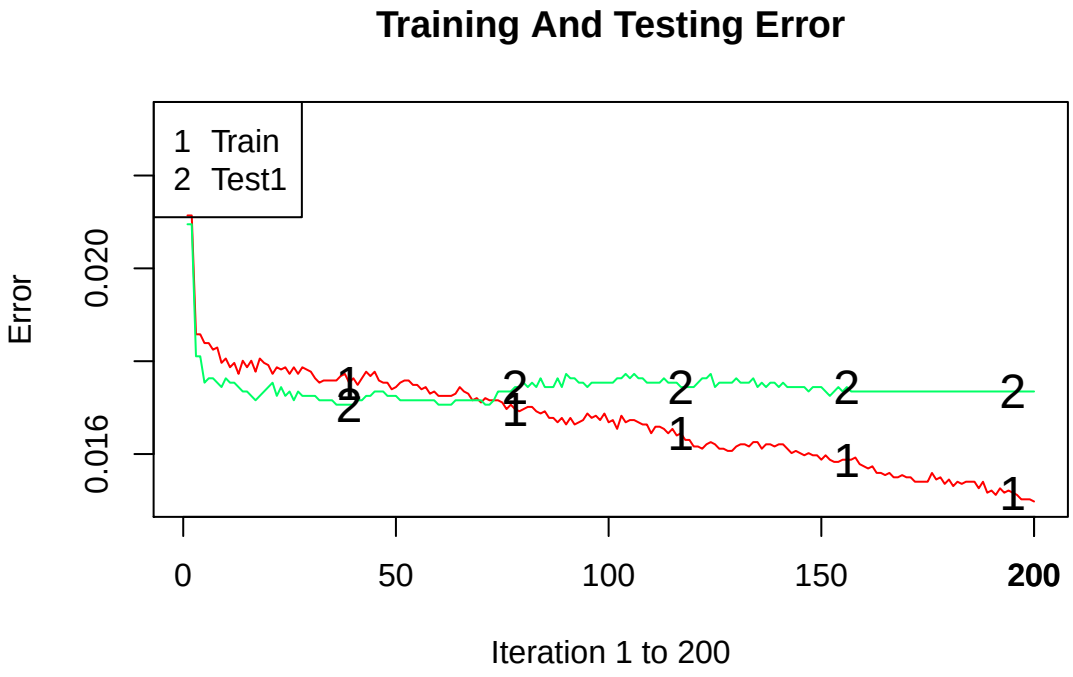
Nous avons testé différents modèles en faisant varier le nombre d'arbres, stumps ou no, et différentaux taux de pénalisation. Nous avons obtenu les résultats suivants :

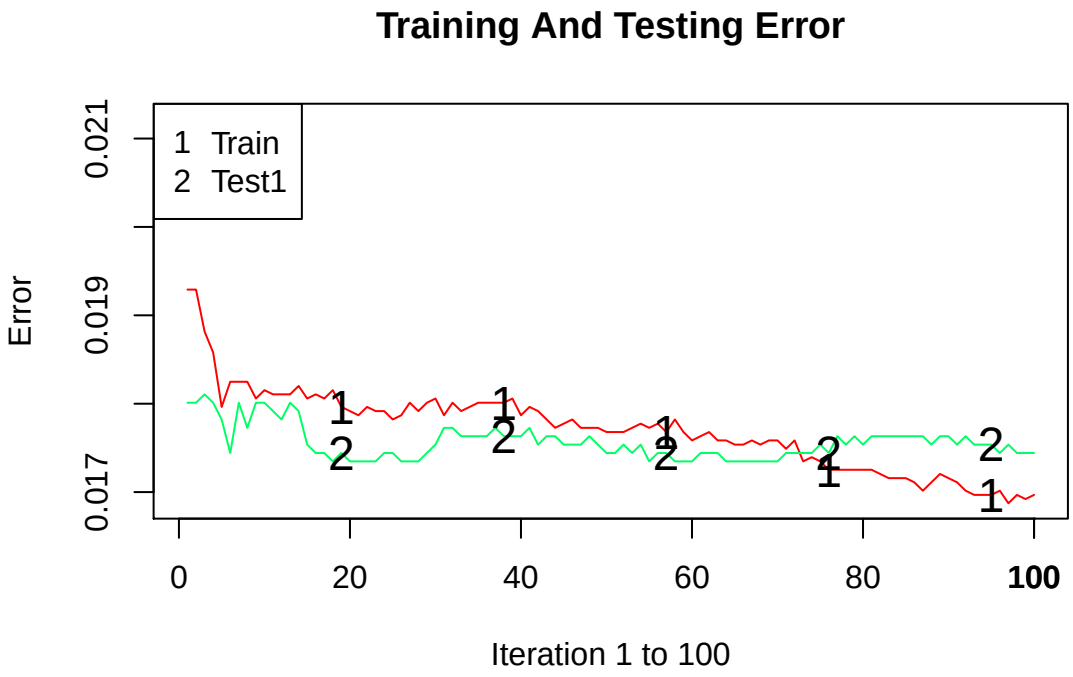
Après avoir tracé l'évolution de l'erreur en apprentissage et en test en fonction du nombre d'abres, les modèles les meilleurs et n'étant pas en su-apprendtissage sont les suivants :

TABLE 10 – Résumé erreur en test - Boosting

	100	200	500
Abres complets sans pénalisation	0.178	0.187	0.175
Arbres complets avec pénalisation de 0.1	0.177	0.178	0.180
Arbres complets avec pénalisation de 0.05	0.172	0.175	0.177
Arbres complets avec pénalisation de 0.01	**0.172**	**0.173**	**0.174**
Stumps avec pénalisation 0.1	0.173	0.173	0.173
Stumps avec pénalisation 0.05	0.173	0.173	0.173
Stumps avec pénalisation 0.01	0.173	0.173	0.173

Ces modèles ont été construits sur des arbres complets avec une pénalisation de 0.01.

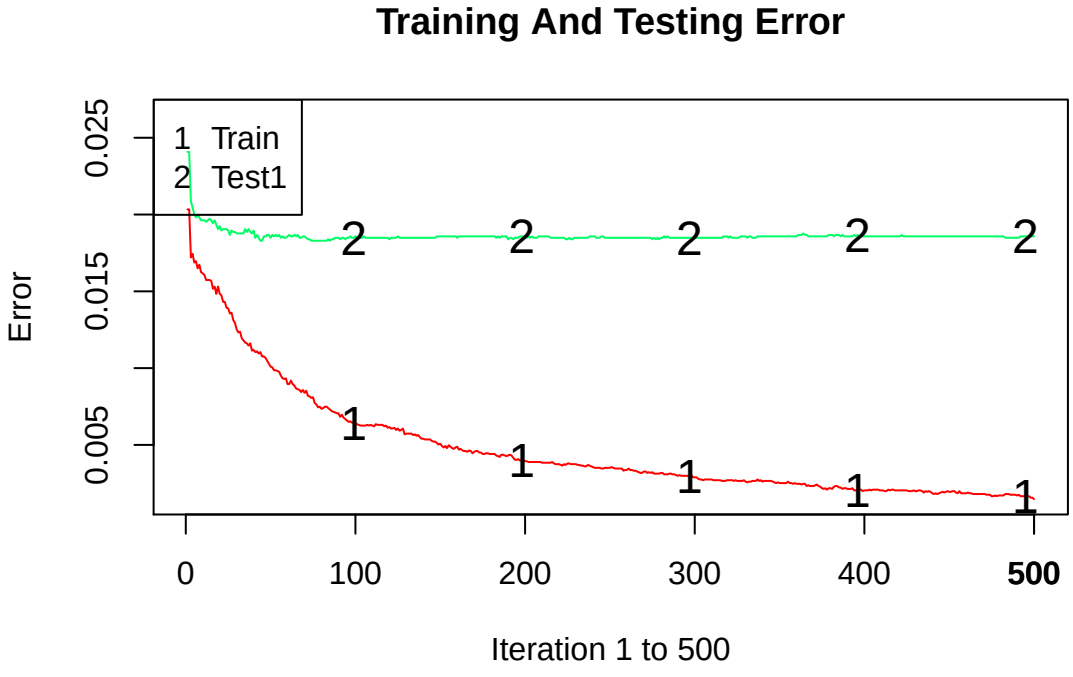


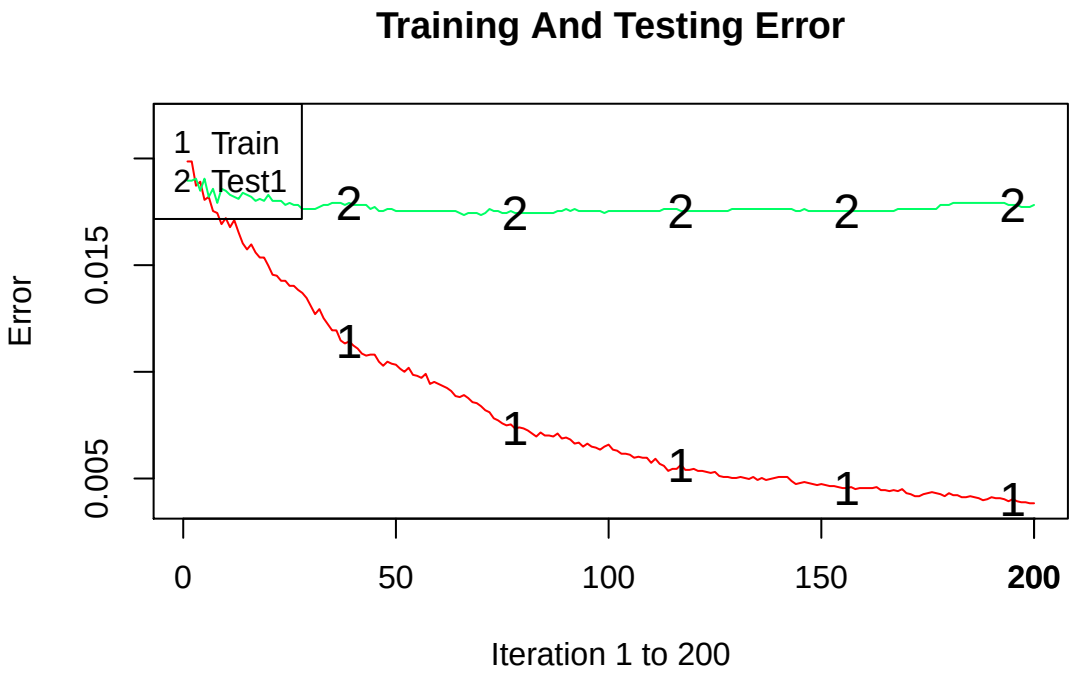


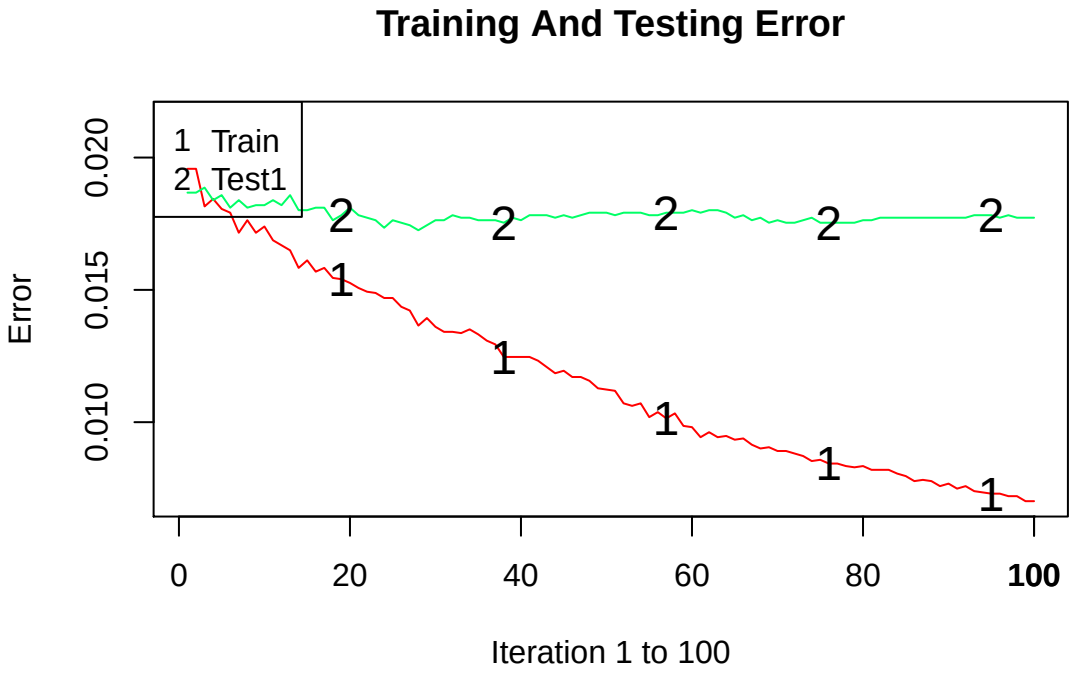
Concernant le modèle avec 500 itérations : durant les 100 premières itérations on observe que l'erreur en test est inférieure à celle en apprentissage (le modèle n'a pas encore appris suffisamment). ensuite, l'erreur en apprentissage continue de diminuer (le modèle apprend de plus en plus les données) pendant que l'erreur en test reste relativement constante.

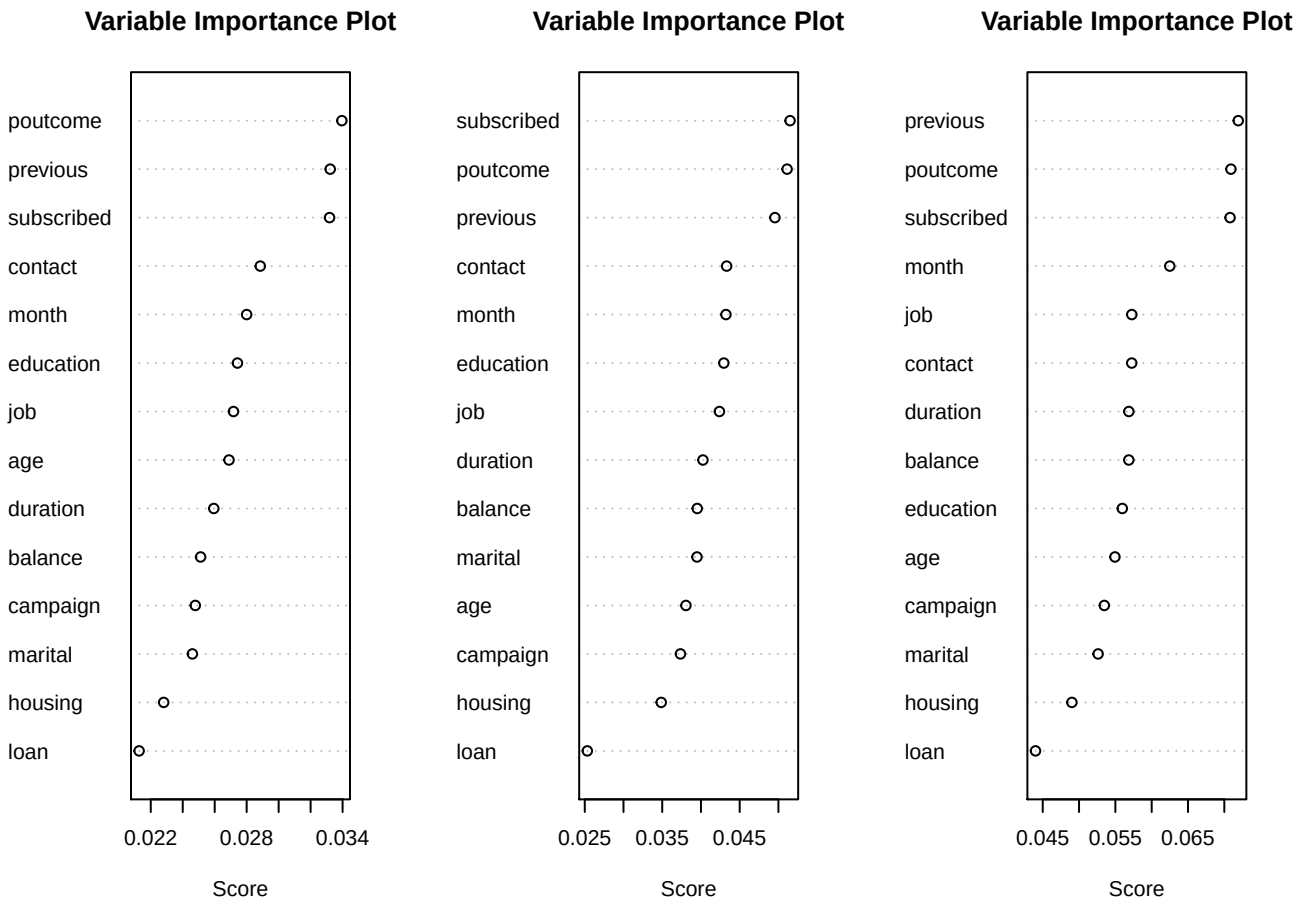
Concernant les modèles avec 200 et 100 itérations: on voit que l'erreur en apprentissage n'est pas très éloigné de l'erreur en test. Limiter le nombre d'itérations serait préférable afin d'empêcher le modèle de sur-apprendre.

Ci-dessous ce sont les graphqiues obtenus avec une pénalisation plus grande (0.1). On voit bien que les résultats sont nettement moins bons.




Training And Testing Error
1 Train
2 Test1
Error
0.005
0.015
0
50
100
150
200
Iteration 1 to 200


Training And Testing Error
1 Train
2 Test1
Error
0.010
0.015
0.020
0
20
40
60
80
100
Iteration 1 to 100



De par les graphiques ci-dessus, on observe que toutes les variables semblent plus importantes moins il y a d'itérations. Aussi, on observe une différence d'importance des variables comparé à la forêt aléatoire. Les variables peu voire pas importantes pour la forêt aléatoire sont ici les variables les plus importantes. Cela est peut-être dû au fait que le bossting accorde plus de poids aux apprenants faibles.

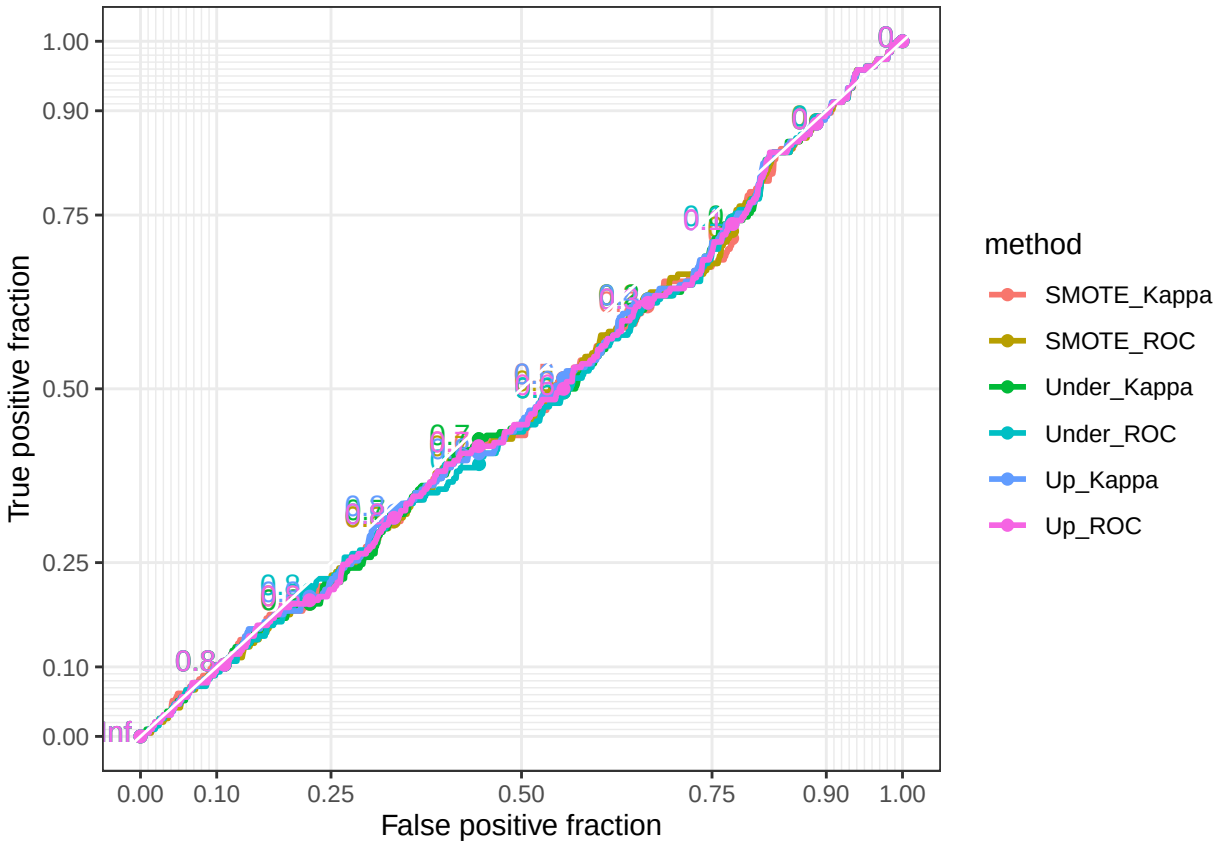
3 Prise en compte des données déséquilibrées : amélioration des modèles

Comme nous l'avons vu précédememnt, nous avons des données déséquilibrées et cela influe beaucoup sur les prédiction de nos modèles. Afin de les rendre plus performants nous allons utiliser des méthodes de rééchantillonnage afin de palier aux soucis qu'engendre le déséquilibre entre les deux classes. Pour ce faire, nous allons utiliser une technique de sous-échantillonnage, sur-échantillonnage et la méthode SMOTE.

3.1 Résultats de l'analyse quadratique discriminante

Nous avons décidé d'utiliser deux métriques, ROC et Kappa, pour la construction de nos modèles. La métrique Kappa étant particulièrement adaptée pour les données déséquilibrées.

3.2 Comparaison selon les méthodes



Grâce aux courbes ROC de nos différents modèles, on observe très bien que ces derniers n'ont aucun pouvoir prédicitif.

Table 11 – QDA sous échantillonnée avec la métrique ROC

	Prédictions	
	Non	Oui
Non	4134	6221
Oui	86	108

Table 12 – QDA SMOTE avec la métrique ROC

	Prédictions	
	Non	Oui
Non	4534	5821
Oui	95	99

Table 13 – QDA sur échantillonnée avec la métrique ROC

	Prédictions	
	Non	Oui
Non	4014	6341
Oui	84	110

Table 14 – QDA sous échantillonnée avec la métrique Kappa

	Prédictions	
	Non	Oui
Non	4517	5838
Oui	97	97

Table 15 – QDA SMOTE avec la métrique Kappa

	Prédictions	
	Non	Oui
Non	4756	5599
Oui	98	96

Table 16 – QDA SMOTE avec la métrique Kappa

	Prédictions	
	Non	Oui
Non	3990	6365
Oui	83	111

Nous allons à présent calculer la pécision de nos modèles en fonction de la métrique choisie à partir des résultats de nos matrices de confusion dont le seuil d'appartenance est :

$$\mathbb{P}(défaut \ = \ oui \ | \ X \ = \ x) > 0.5$$

Avec la métrique ROC :

— QDA sous échantillonnée : 1.9 %

— QDA SMOTE : 2.26 %
— QDA sur échnatillonée : 1.92 %

Avec la métrique Kappa :

— QDA sous échantillonnée : 1.8 %
— QDA SMOTE : 1.9 %
— QDA sur échnatillonée : 1.9 %

Ainsi, on se rend bien compte, en plus de ce qu'illustre nos courbes ROC, que nos modèles ne sont pas du tout précis.

3.3 Résultats pour la forêt aléatoire

Table 17 – Métrique ROC

	Paramètres		Résultats	
	Nombre de variables	Nombre d'arbres	Erreur OOB	Erreur test
Sous-echantillonné				
1er modèle	9	500	18.03 %	19.9 %
2ème modèle	**9**	**300**	**18.67 %**	**21.88 %**
3ème modèle	12	200	19.67 %	22.6 %
4ème modèle	12	300	21.23 %	21.92 %
SMOTE				
5ème modèle	12	300	10.52 %	8.08 %
6ème modèle	12	200	10.85 %	9.30 %
7ème modèle	9	300	11.18 %	8.18 %
8ème modèle	**9**	**200**	**11.29 %**	**7.77 %**

Pour chaque taille d'arbres (500,300 et 200) nous avons tester un nombre de variables égal à 2,3,6,8,9 et 12.

Concernant les modèles sous-échantillonnées :

— une forêt de 500 arbres nous renvoie 9 variables.
— une forêt de 300 et 200 arbres nous renvoie 12 variables.

Nous avons aussi testé une forêt de 300 arbres et 9 variables.

Le modèle sous-échantillonné retenu est le 2ème modèle. Il a une faible erreur OOB et une erreur test qui reste assez proche voire inférieure aux autres modèles. L'erreur OOB est ici un peu optimiste. Le nombre de variables n'est pas maximal, ce qui évite de l'over-fitting. Concernant les modèles SMOTE :

— une forêt de 300 et 200 arbres nous renvoie 12 variables.

Nous avons aussi testé pour 9 variables.

Le modèle SMOTE retenu est le 8ème modèle. Il possède une erreur OOB relativement plsu élevée que les autres mais une erre en test relativement plus faible. Il ne possède pas non plus un nombre trop important de variables.

Table 18 – Métrique kappa

	Paramètres		Résultats	
	Nombre de variables	Nombre d'arbres	Erreur OOB	Erreur test
Sous-echantillonné				
1er modèle	12	200	19.57 %	22.73 %
2ème modèle	**9**	**300**	**19.82 %**	**21.15 %**
3ème modèle	9	200	19.82 %	21.99 %
4ème modèle	12	300	20.22 %	22.90 %
5ème modèle	12	100	21.87 %	21.95 %
SMOTE				
6ème modèle	**9**	**200**	**10.56 %**	**8.2 %**
7ème modèle	9	300	11.03 %	8.1 %
8ème modèle	9	100	11.33 %	7.1 %

Pour chaque taille d'arbres (500,300 et 200) nous avons tester un nombre de variables égal à 2,3,6,8,9 et 12.

Concernant les modèles sous-échantillonnées :

— une forêt de 300 arbres nous renvoie 9 variables.
— une forêt de 200 et 100 arbres nous renvoie 12 variables.

Nous avons aussi testé une forêt de 300 arbres et 12 variables et 200 arbres et 9 variables.

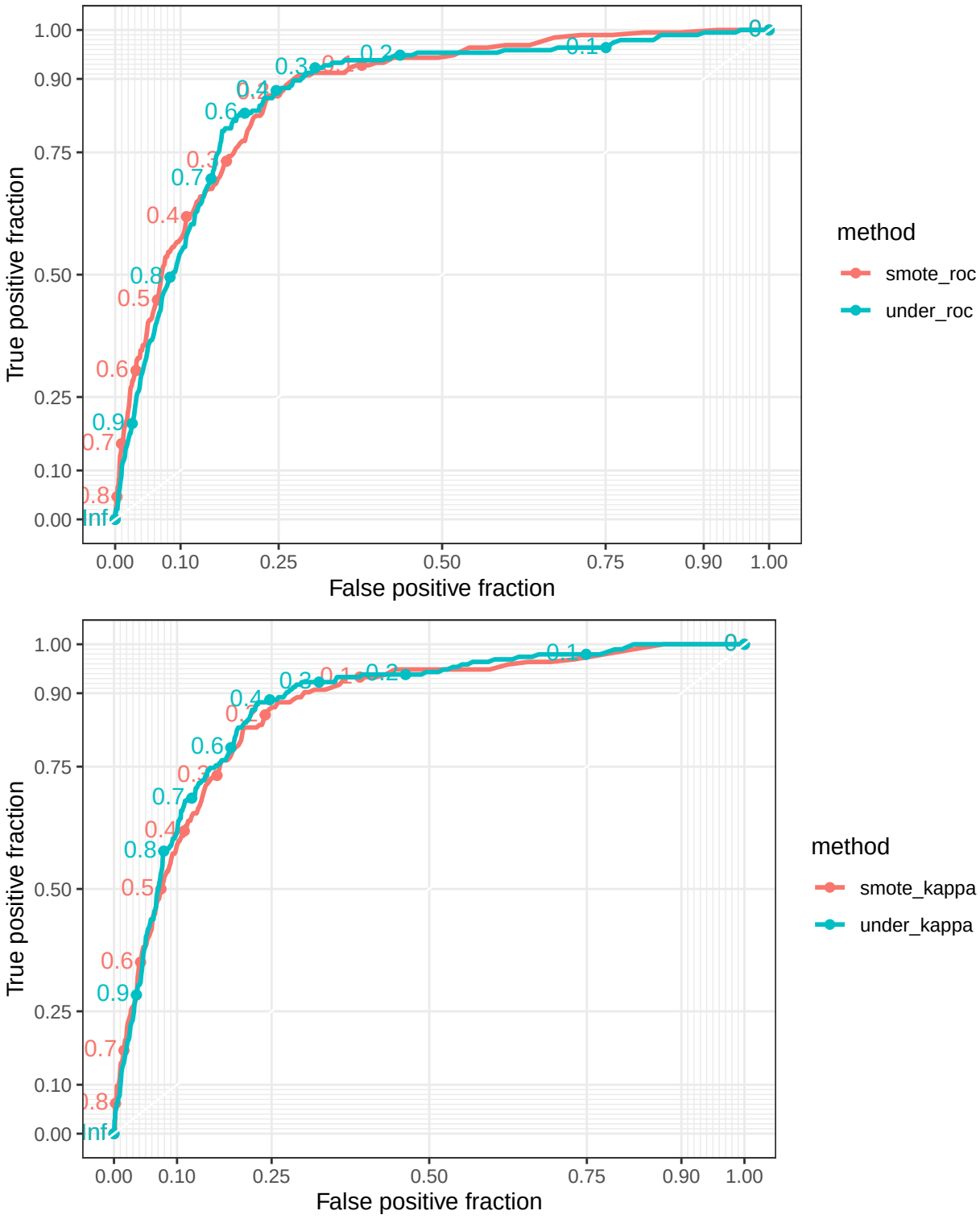
Le modèle sous-échantillonné retenu est le 2ème. Il possède une erreur OOB et une erreur test assez faible par rapport aux autres. De plus, le nombre de variables n'est pas extrêmement élevé ce qui évite de l'over-fitting.

Concernant les modèles SMOTE :

Une forêt à 300,200 et 100 arbre snous renvoie 9 variables.

Nous avons décidé de sélectionner le modèle 6 car il possède les erreurs les plus faibles.

3.4 Comparaison selon les méthodes



Au vu des graphique ci-dessus, on peut dire que le modèle sous-échantillonné est un peu meilleur que le modèle SMOTE quelque soit la métrique.

Un seuil optimal serait de 0.4 pour le modèle sous-échantillonné et de 0.2 pour le modèle SMOTE.

Table 19 – Métrique ROC

	Seuil 0.6		Seuil 0.5	
	under_roc	**smote_roc**	under_roc	**smote_roc**
Precision	0.995465643529822	**0.987161761801304**	0.996056199161942	**0.989633583085292**
Recall	0.826846933848382	**0.96533075808788**	0.780492515692902	**0.931144374698213**
F1_Score	0.903355138214813	**0.976124212684927**	0.875196274838919	**0.959498457557966**

Table 20 – Métrique ROC

	Seuil 0.4		Seuil 0.2	
	under_roc	**smote_roc**	under_roc	**smote_roc**
Precision	0.996989528795811	**0.992101276779918**	0.998364231188659	**0.996783324755533**
Recall	0.73558667310478	**0.885465958474167**	0.530468372766779	**0.748140994688556**
F1_Score	0.846568491247569	**0.935755472776445**	0.692817052405878	**0.854747062393115**

Table 21 – Métrique Kappa

	Seuil 0.6		Seuil 0.5	
	under_kappa	**smote_kappa**	under_kappa	**smote_kappa**
Precision	0.994438138479001	**0.987455197132617**	0.996336996336996	**0.989979338842975**
Recall	0.846064703042009	**0.957798165137615**	0.788025108643168	**0.925446644133269**
F1_Score	0.914270806157057	**0.972400607872935**	0.880021569156107	**0.956625904666833**

Table 22 – Métrique Kappa

	Seuil 0.4		Seuil 0.2	
	under_kappa	**smote_kappa**	under_kappa	**smote_kappa**
Precision	0.99723429474516	**0.992020703040759**	0.997873985311171	**0.996455247499684**
Recall	0.731240946402704	**0.888459681313375**	0.498599710284887	**0.760115886045389**
F1_Score	0.843770893692891	**0.937388557746192**	0.664949449417219	**0.86238632628465**

Pour les trois seuils et pour les trois critères de performance, le modèle SMOTE semble meilleur. Cela s'oppose à ce que nous avions trouvé au vue de la courbe ROC.

On peut voir que le recall et le F1-score baisse suite à l'abaissement du seuil. Les modèles sont meilleurs à un seuil de 0.6.

Table 23 – Under - SMOTE (seuil 0.6 - ROC)

	0	1
FALSE	8562	1793
TRUE	39	155

	0	1
FALSE	9996	359
TRUE	130	64

La matrice de confusion pour le modèle sous-échantillonné nous donne les résultats suivants :

— taux de bonnes prédictions : 82.6 %
— taux de mauvaises prédictions : 17.4 %.

La matrice de confusion pour le modèle SMOTE nous donne les résultats suivants :

— taux de bonnes prédictions : 95.3 %
— taux de mauvaises prédictions : 4.7 %

Table 24 – Under - SMOTE (seuil 0.6 - Kappa)

	0	1		0	1
FALSE	8761	1594	FALSE	9918	437
TRUE	49	145	TRUE	126	68

La matrice de confusion pour le modèle sous-échantillonné nous donne les résultats suivants :

— taux de bonnes prédictions : 84.4 %
— taux de mauvaises prédictions : 15.6 %.

La matrice de confusion pour le modèle SMOTE nous donne les résultats suivants :

— taux de bonnes prédictions : 94.6 %
— taux de mauvaises prédictions : 5.4 %

Cela vient confirmer le fait que le modèle SMOTE est plus performant que le modèle sous-échantillonné. De plus, on remarque qu'une légère différence entre les deux métriques.

4 Etude de la variable prêt immobilier

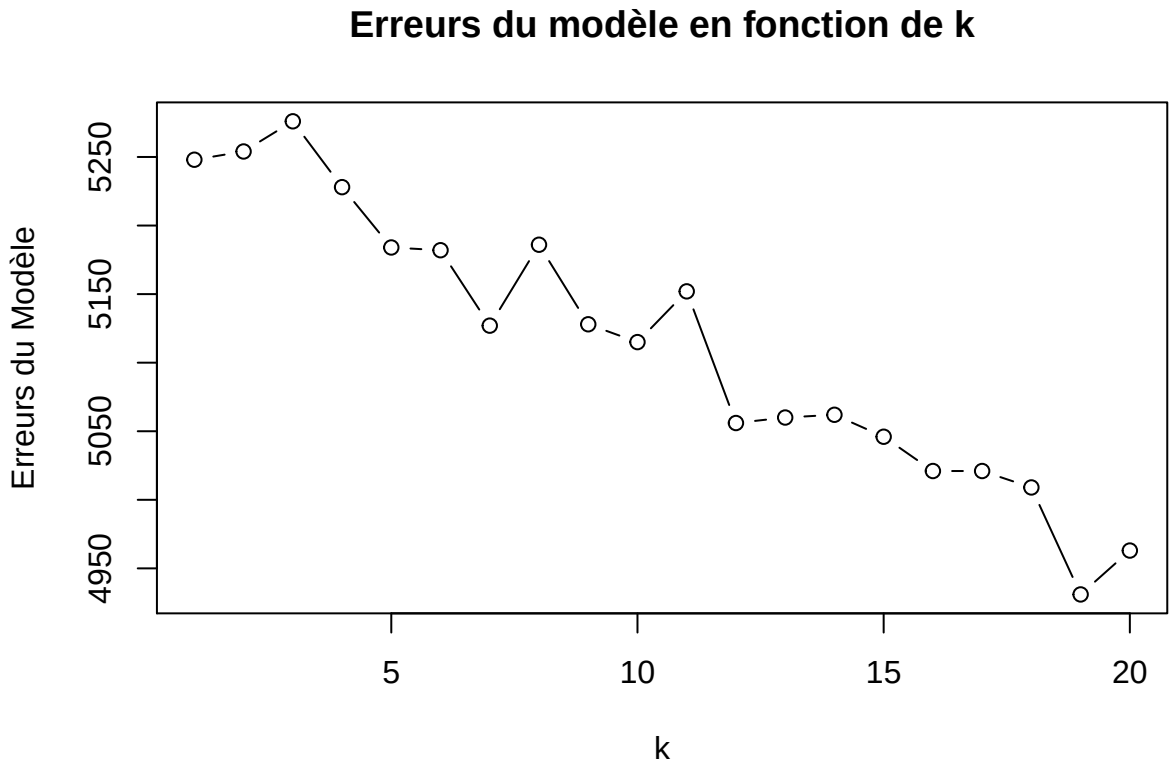
— La variable housing indique si l'individu à contracté un prêt immobilier. C'est une variable binaire.

La proportion empirique de personnes ayant contracté un prêt immobilier est de 0.5.

→ On gardera cette valeur comme un critère de performance minimale, en partant du principe que si un modèle ne fait pas mieux que 55% de taux de prédiction, nous aurions pu avoir le même résultat en devinant simplement le classement par la proportion empirique. Nous rejetterons donc les modèles moins performants.

4.1 Méthode des K-plus proches voisins

— Ici, nous ne gardons que les variables quantitatives.



→ On voit bien que le modèle va vite devenir ininterprétable pour un faible gain en prédiction.

TABLE 25 – QDA sous échantillonnée avec la métrique ROC

	no	yes
no	1557	3158
yes	1843	3991

Dans ce modèle le taux d'erreur est de 47 % et le taux de prédiction est de 68 %.

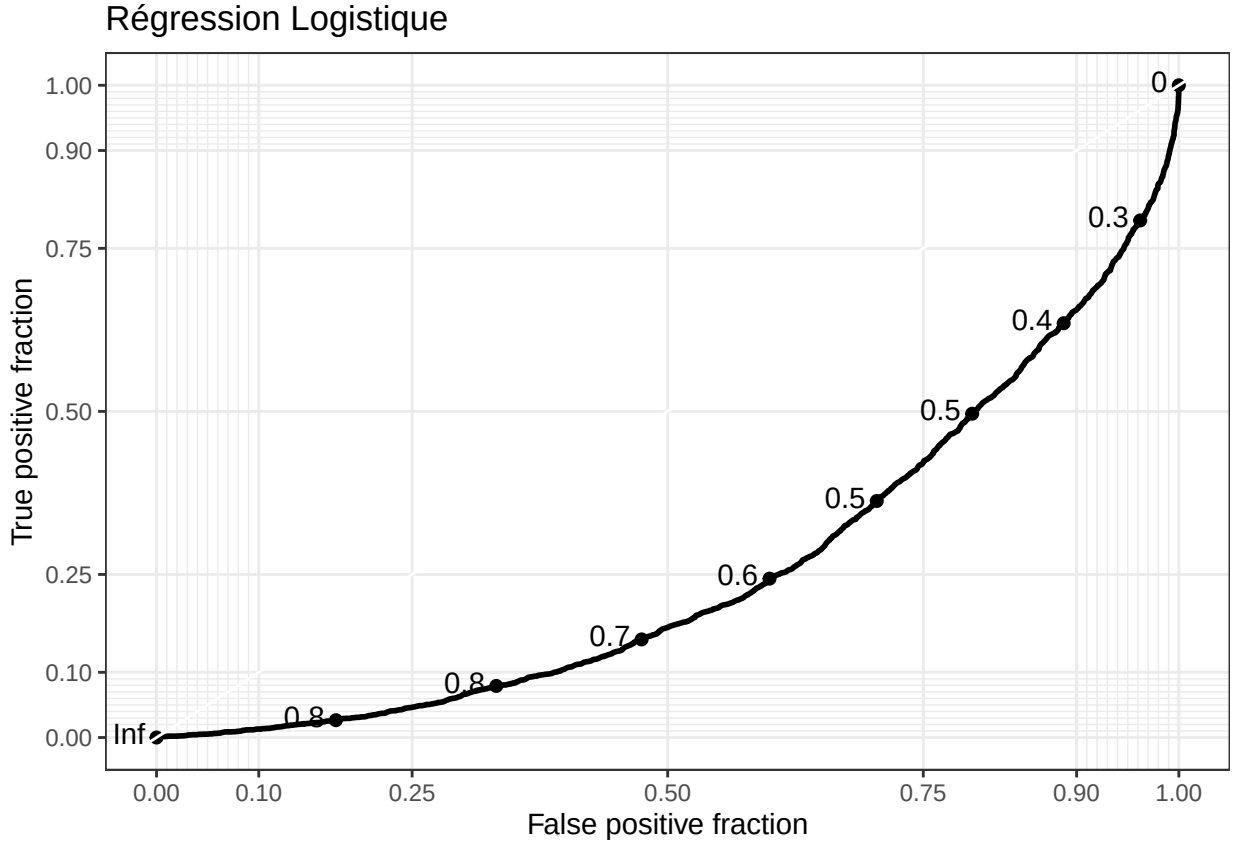
$\Longrightarrow$ Ce modèle fait mieux qu'une prédiction naîve, mais semble peu performant ici. Le taux d'erreur semble trop élevé, le pouvoir prédictif de ce modèle semble très limité.

→ Peut être dû à l'ommision des variables qualitatives.

4.2 Régression Logistique

— On pourrait utiliser les modèles à réponses binaires pour estimer notre variable qualitative, cela nous permetterait de conserver l'information contenue dans les variables catégorielles.

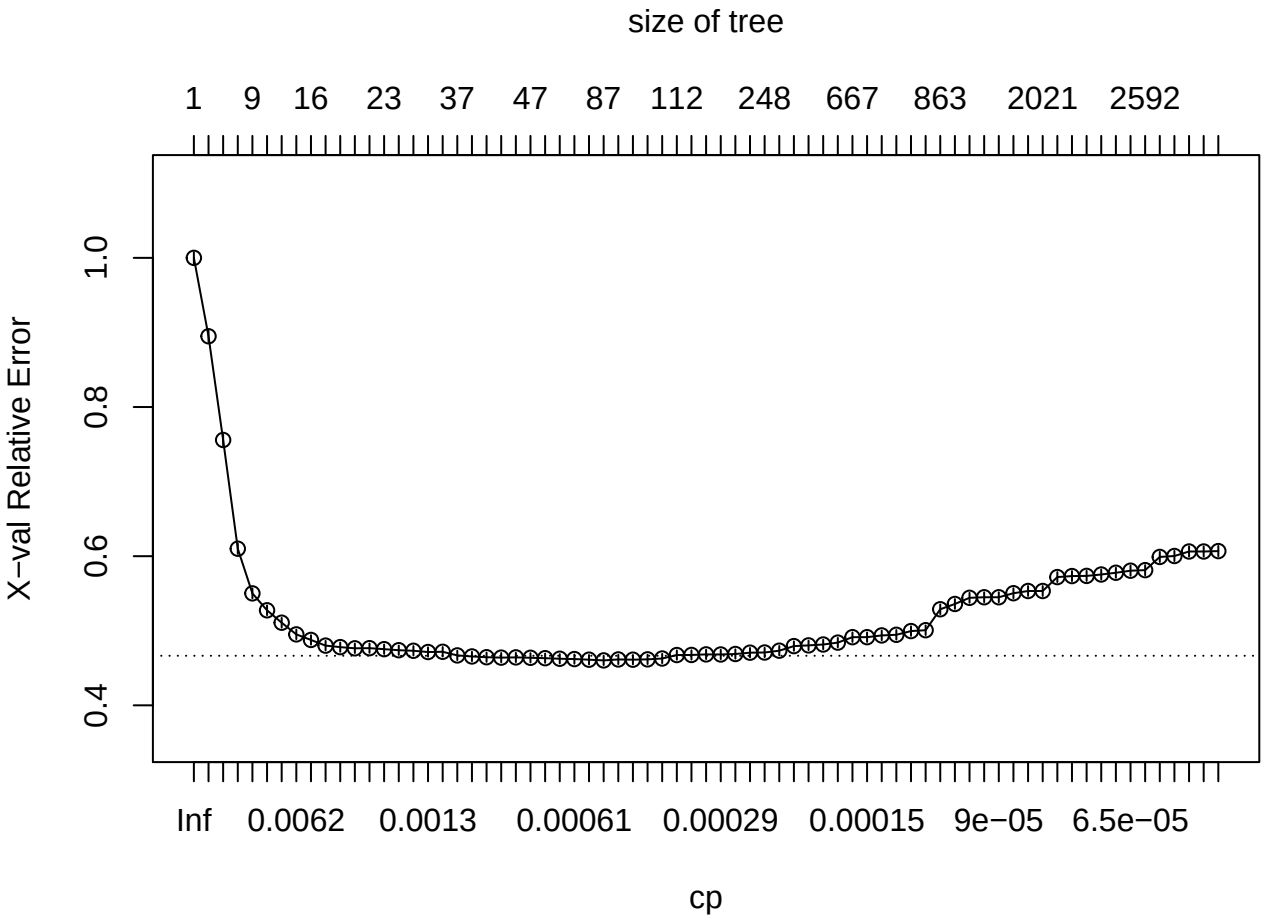
Le taux d'erreur de ce modèle est de 36 % et le taux de bonnes prédiction est de 70 %. - Légèrement mieux que la méthode des k plus proches voisins, cependant le taux de prédiction n'est pas vraiment mieux.



— On voit vite les limites de notre modèle..

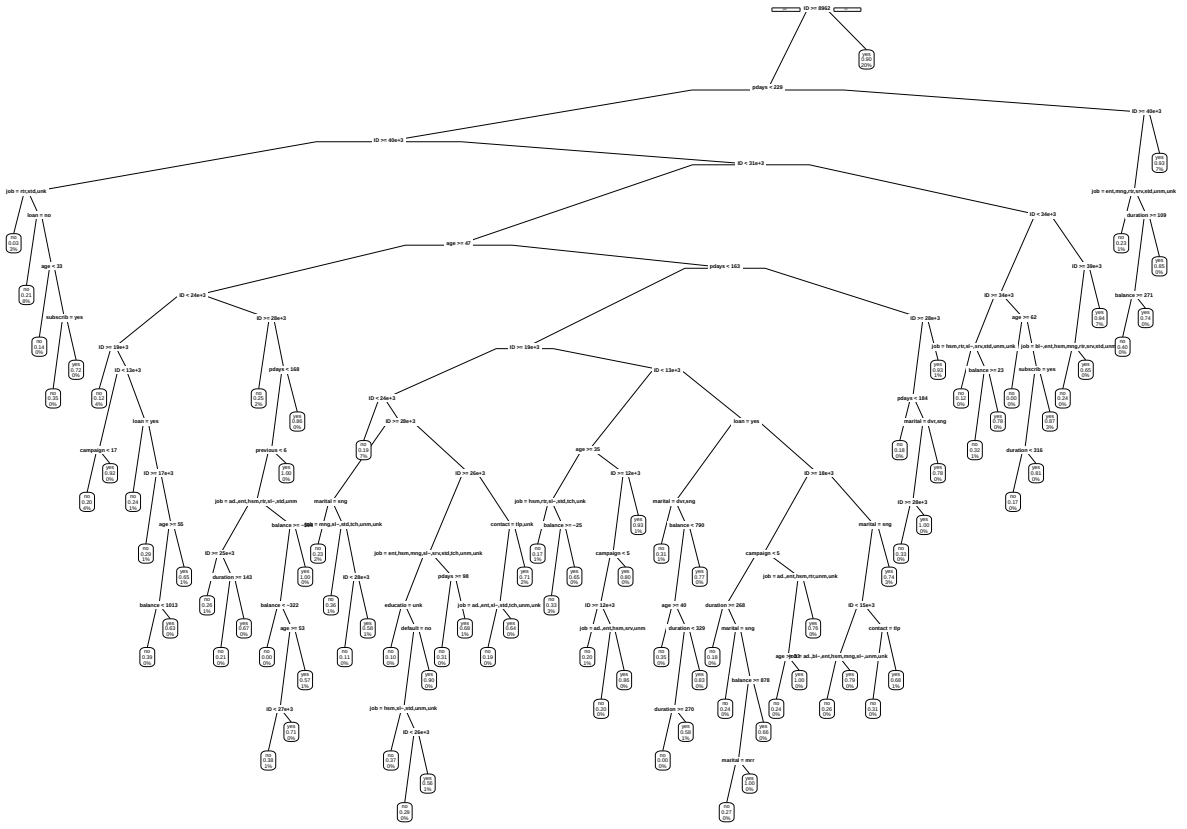
4.3 Arbre de Décision

— On enlève les variables Month et Day car très peu interprétables, et nous gardons ID car cette variable nous informe sur l'ordre d'inscription des clients, cette variable est interprétable. De plus, nous récupèrons les variables catégorielles pour notre estimation.



4.3.1 Elaguage

— Elaguons cet arbre à son niveau de complexité minimisant l'erreur :



→ L'arbre élagué est très gros. Le niveau de complexité optimal conduit à un arbre relativement profond, ce qui implique beaucoup de variabilité et un arbre pas assez interprétable.

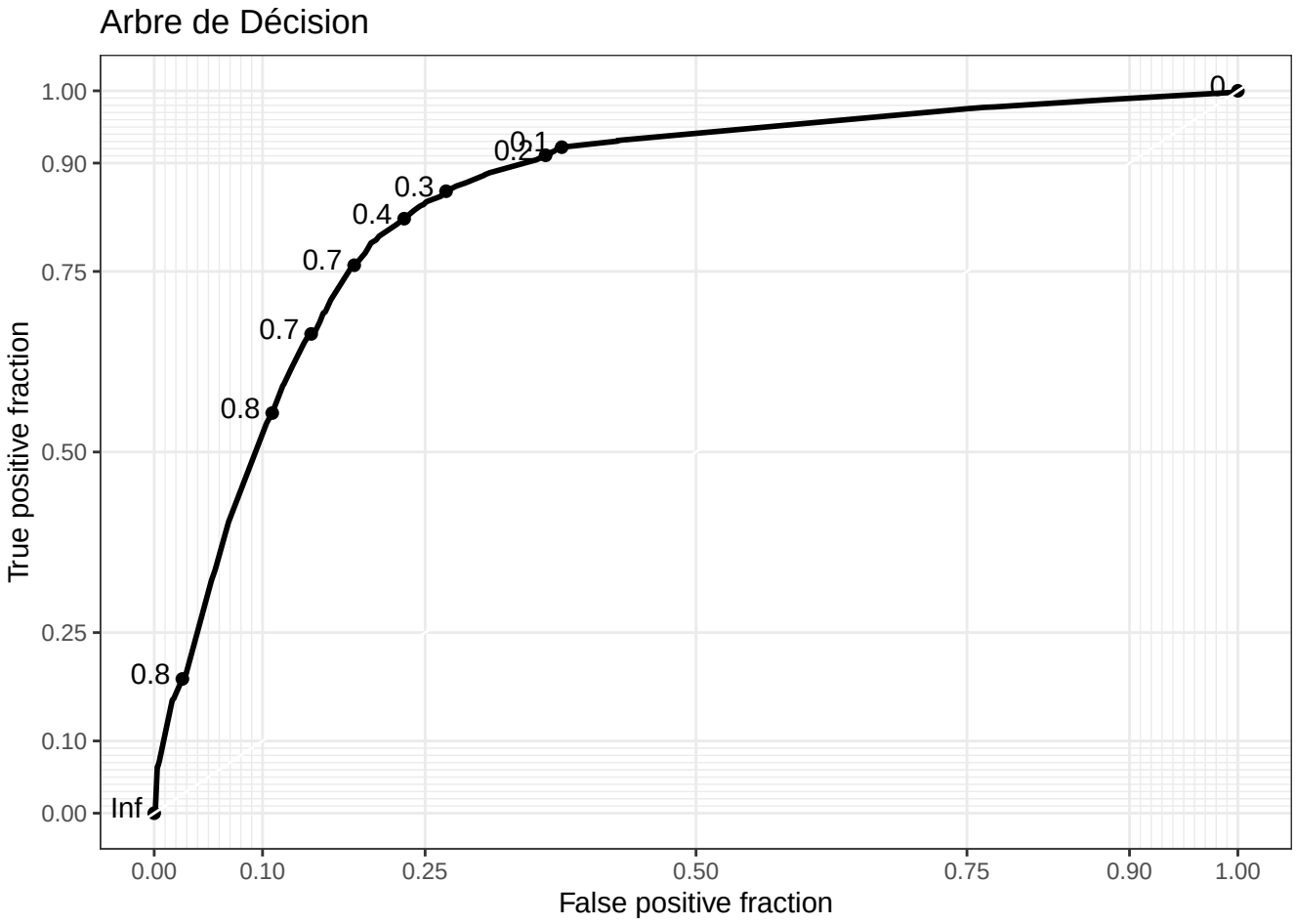
— Testons maintenant le modèle d'entrainement sur l'échantillon test.

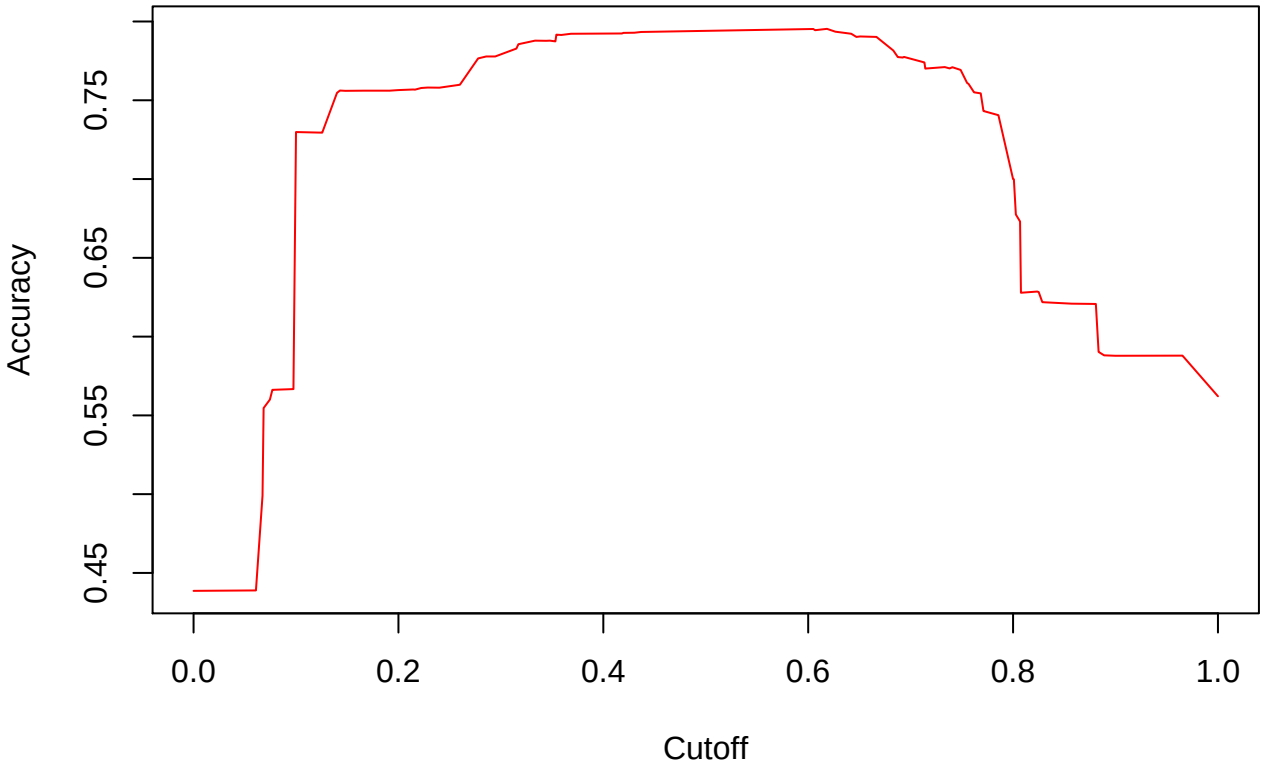
TABLE 26 – QDA sous échantillonnée avec la métrique ROC

	no	yes
no	3696	931
yes	1229	4693

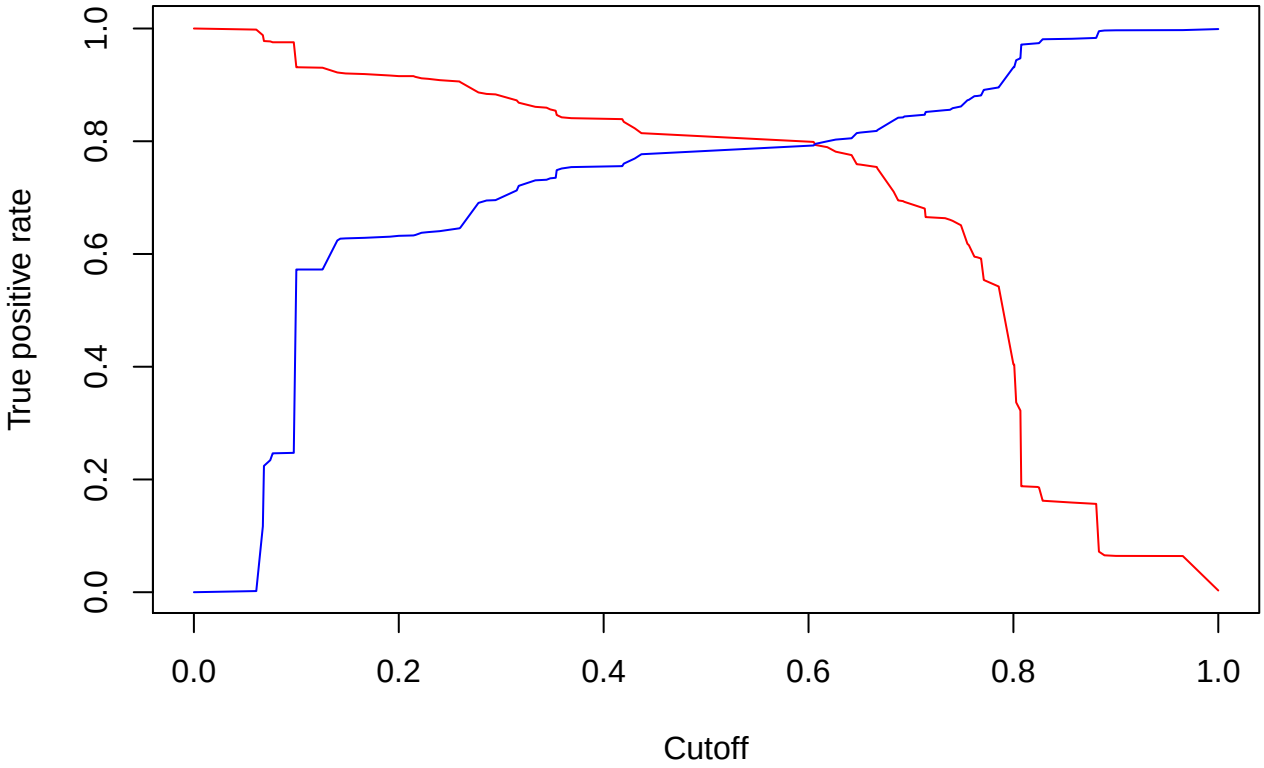
Le taux d'erreur est de 22 % et le taux de prédiction de 77 %. → 20% de mals classés, et environ 79 % de prédiction. Cet arbre à un pouvoir prédictif certain.

4.3.2 Courbe ROC

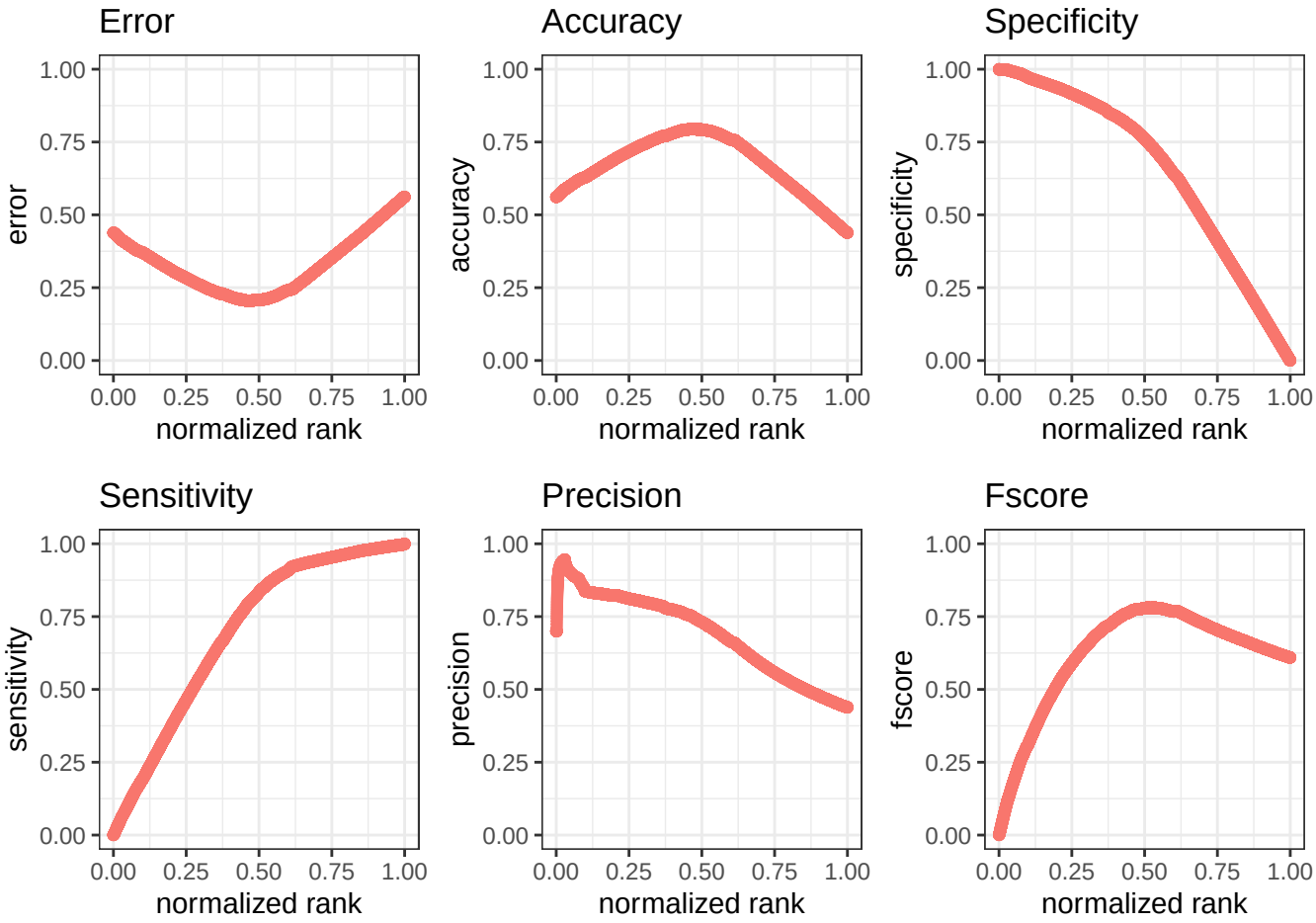




→ On peut atteindre jusqu'à presque 80% de précision

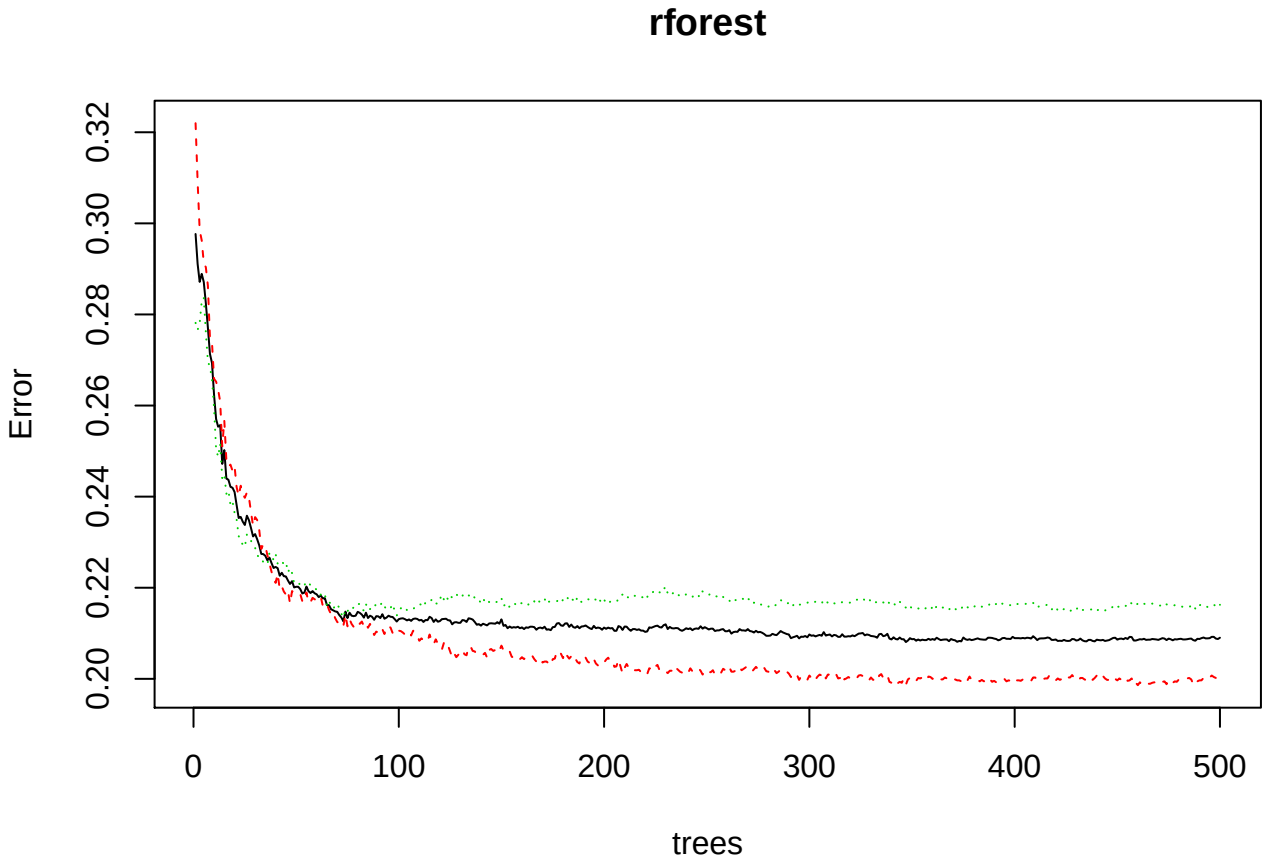


→ Le seuil optimal semble etre autour de 0,6 et maximise les taux de vrais positifs/négatifs vers 80% .



4.4 Forêt Aléatoire

— Conscients du caractère peu adaptable des arbres de décisions, nous allons construire une forêt aléatoire en prennant des sous-échantillons aléatoires de notre échantillon d'entrainement pour y construire 500 arbres à partir de sous-jeux de variables aléatoires. Les 500 arbres aléatoires formeront la forêt aléatoire dont nous testerons l'efficacité.



→ On peut descendre vers 20 % d'erreurs avec moins de 500 arbres !

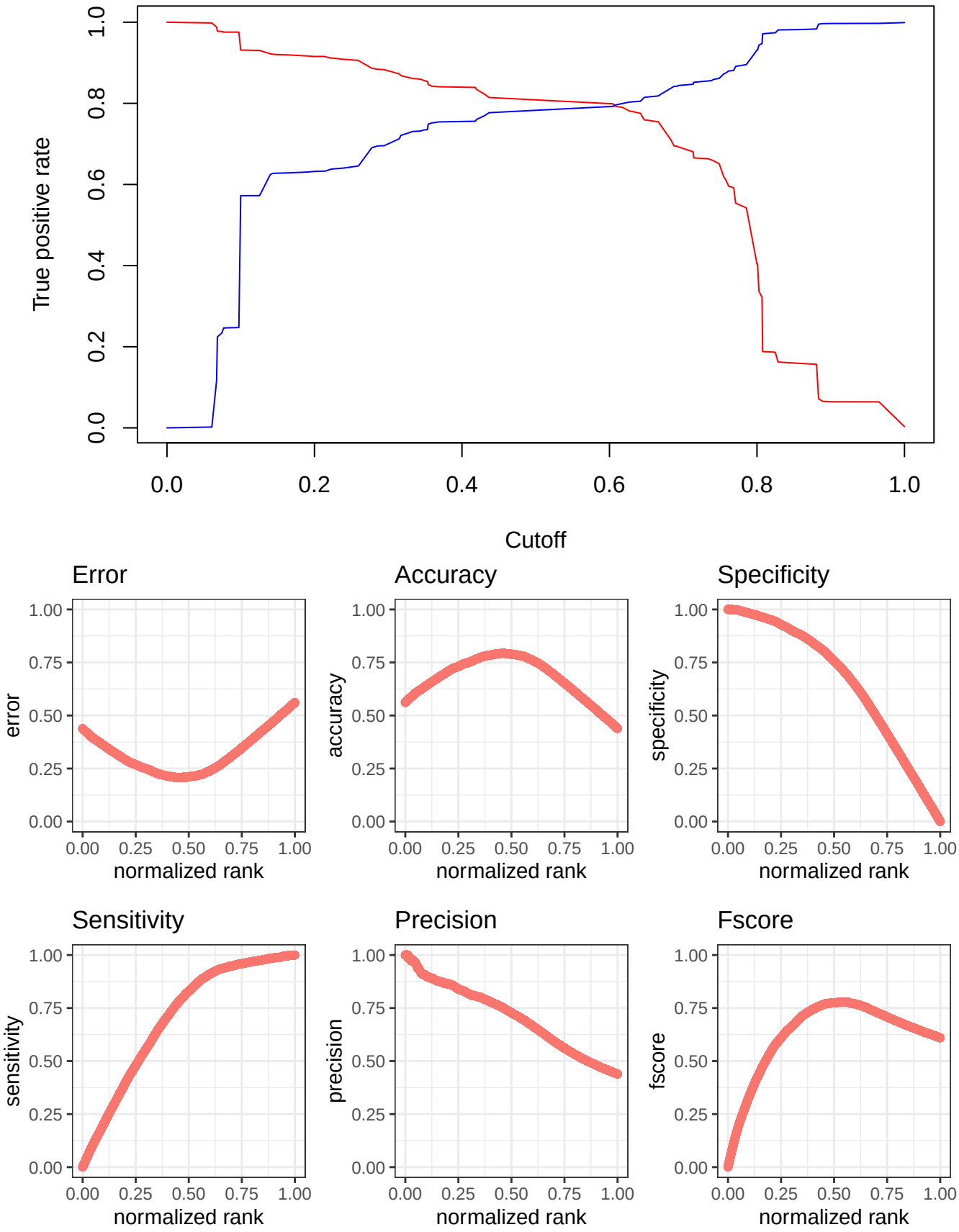
TABLE 27 – QDA sous échantillonnée avec la métrique ROC

	no	yes
no	3733	894
yes	1311	4611

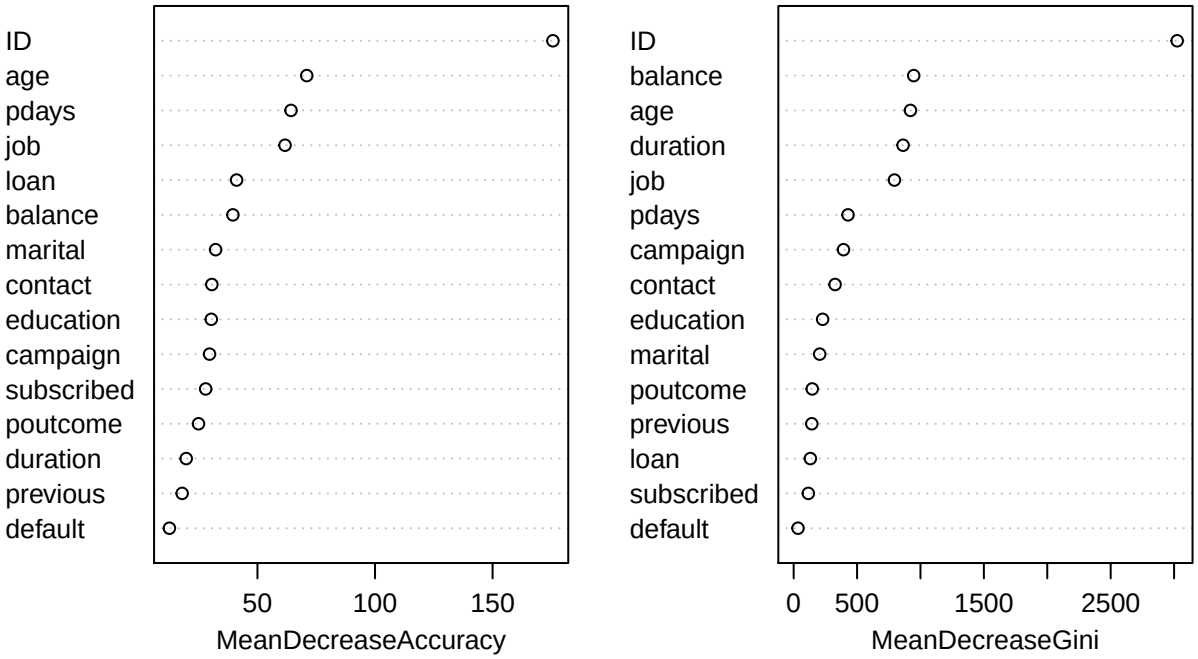
On voit que dans ce modèle le taux de prédiction est de 78 % et le taux d'erreur est de 21 %.

— Les taux d'erreur et de prédiction ne varient pas sensiblement.
— On cherchera par la suite à optimiser les paramètres de la construction de la forêt.

4.4.1 Courbe Roc



→ Les résultats obtenus sont sensiblement les mêmes, il ne semble pas y avoir d'intérêt certain a préferer une forêt aléatoire. Cependant, elle nous permet de mesurer l'importance des variables dans la constructions des arbres :



→ On voit que la variable ID a le plus d'importance. On pourrait interpréter ce fait comme ceci : ID représente l'ordre chronologique d'inscription, et semble être la plus importante dans la construction des arbres pour expliquer Housing. Ainsi, on peut se dire que Housing est expliquée en partie par un effet temporel sur ce comportement, car la position dans l'ordre d'inscription semble être un facteur explicatif de Housing. On retrouve Age, Balance, Duration et Job ensuite, ce qui semble plus intuitif.