TP - Abres de décision et forêts aléatoires

Enzo Munoz - Hugo Munier - Maxime Guilbaud

2025-10-21

Importation des données

```
##   age marital education default balance housing loan   contact poutcome class
## 1  33 married  tertiary      no     882      no   no telephone  failure    no
## 2  42  single secondary      no    -247     yes  yes telephone    other   yes
## 3  33 married secondary      no    3444     yes   no telephone  failure   yes
## 4  36 married  tertiary      no    2415     yes   no telephone    other    no
## 5  36 married  tertiary      no       0     yes   no telephone  failure   yes
## 6  44 married secondary      no    1324     yes   no telephone    other    no
```

```
##       age         marital           education           default
##  Min.   :18.0   Length:7864        Length:7864        Length:7864
##  1st Qu.:32.0   Class :character   Class :character   Class :character
##  Median :38.0   Mode  :character   Mode  :character   Mode  :character
##  Mean   :40.8
##  3rd Qu.:47.0
##  Max.   :89.0
##     balance          housing              loan             contact
##  Min.   :-1884.0   Length:7864        Length:7864        Length:7864
##  1st Qu.:  162.8   Class :character   Class :character   Class :character
##  Median :  596.0   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 1553.2
##  3rd Qu.: 1740.0
##  Max.   :81204.0
##    poutcome            class
##  Length:7864        Length:7864
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

Le jeu de données étudié comprend 8 variables de type catégoriel (caractère) et 2 variables numériques (float). L'objectif de notre travail est de développer un modèle prédictif pour la variable class, qui indique si un client souscrit ou non à un dépôt à terme.

Valeurs manquantes

```
## [1] "=== Nombre de NA par variable ==="
```

```
##       age   marital education   default   balance   housing      loan   contact
##         0         0         0         0         0         0         0         0
##  poutcome     class
##         0         0
```

```
## [1] "=== Nombre de 'unknown' par variable ==="
```

```
##       age   marital education   default   balance   housing      loan   contact
##         0         0         0         0         0         0         0         0
##  poutcome     class
##         0         0
```

Le jeu de données ne présente aucune valeur manquante, ce qui nous dispense d'effectuer des étapes de traitement ou d'imputation liées aux données manquantes.

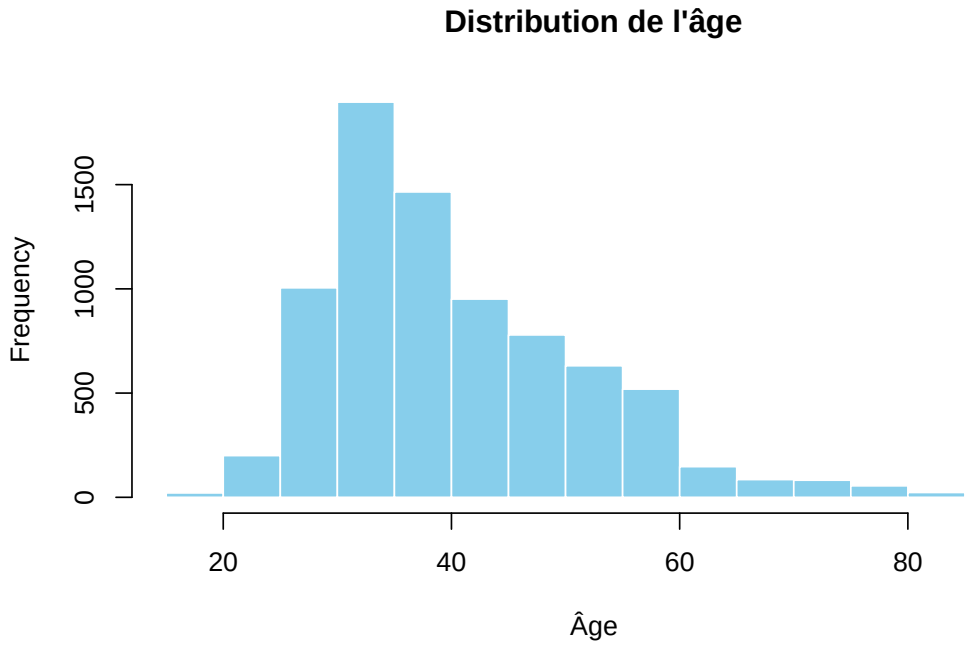
Transformation des variables

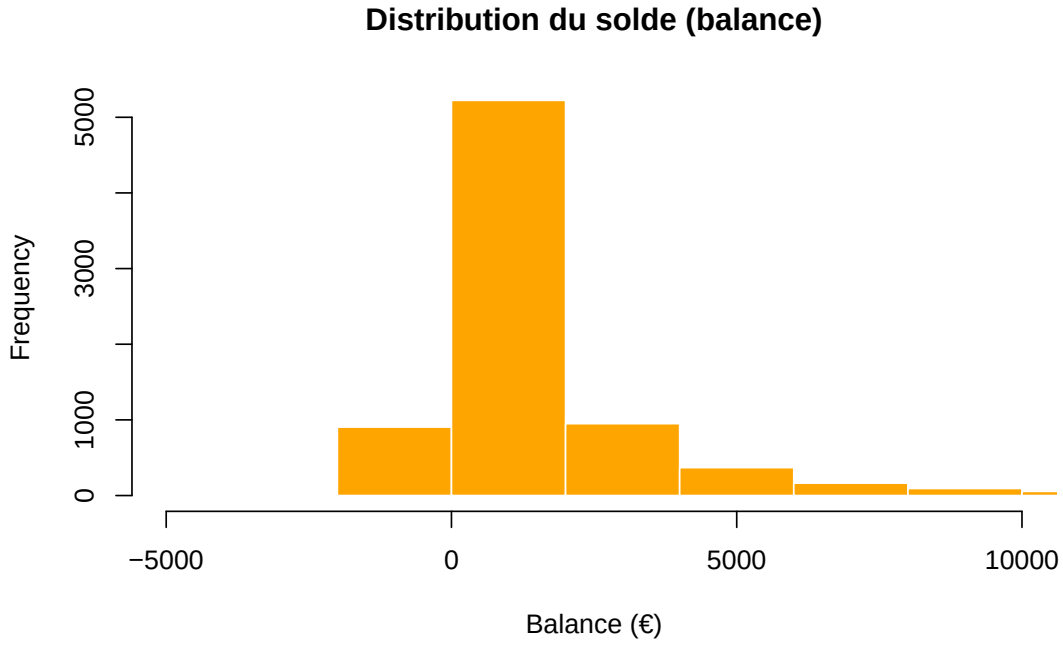
```
## 'data.frame':    7864 obs. of  10 variables:
##  $ age      : num  33 42 33 36 36 44 26 51 33 30 ...
##  $ marital  : Factor w/ 3 levels "divorced","married",..: 2 3 2 2 2 2 3 3 1 2 ...
##  $ education: Factor w/ 3 levels "primary","secondary",..: 3 2 2 3 3 2 3 2 2 2 ...
##  $ default  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ balance  : num  882 -247 3444 2415 0 ...
##  $ housing  : Factor w/ 2 levels "no","yes": 1 2 2 2 2 2 1 1 2 2 ...
##  $ loan     : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 2 1 1 1 ...
##  $ contact  : Factor w/ 2 levels "cellular","telephone": 2 2 2 2 2 2 2 2 2 2 ...
##  $ poutcome : Factor w/ 3 levels "failure","other",..: 1 2 1 2 1 2 2 1 1 3 ...
##  $ class    : Factor w/ 2 levels "no","yes": 1 2 2 1 2 1 1 1 1 1 ...
```

```
##       age           marital         education    default       balance
##  Min.   :18.0   divorced: 890   primary  :1015   no :7808   Min.   :-1884.0
##  1st Qu.:32.0   married :4515   secondary:4210   yes:  56   1st Qu.:  162.8
##  Median :38.0   single  :2459   tertiary :2639              Median :  596.0
##  Mean   :40.8                                              Mean   : 1553.2
##  3rd Qu.:47.0                                              3rd Qu.: 1740.0
##  Max.   :89.0                                              Max.   :81204.0
##  housing     loan           contact        poutcome    class
##  no :2917   no :6774   cellular :7274   failure:4694   no :6068
##  yes:4947   yes:1090   telephone: 590   other  :1751   yes:1796
##                                         success:1419
##
##
##
```

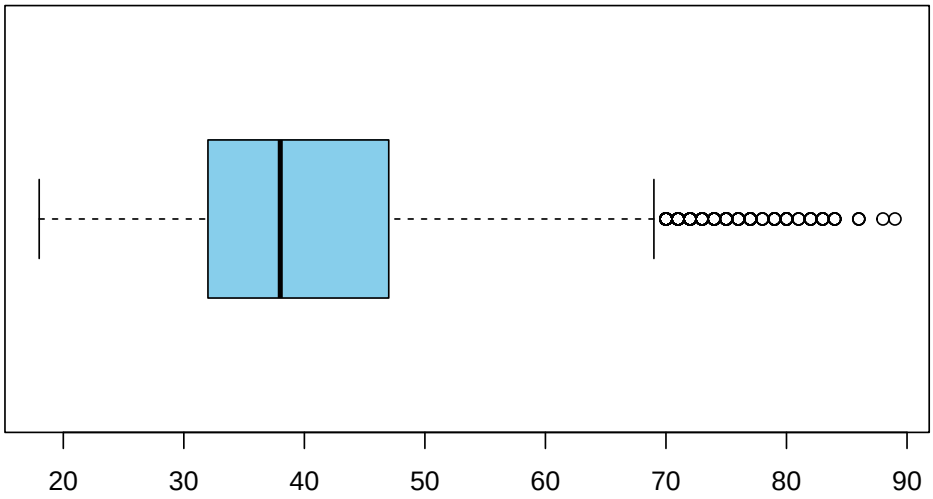
Afin de pouvoir exploiter correctement l'ensemble des variables dans R, les variables catégorielles ont été converties en facteurs, tandis que les variables quantitatives ont été transformées en numériques.

Analyse univariée

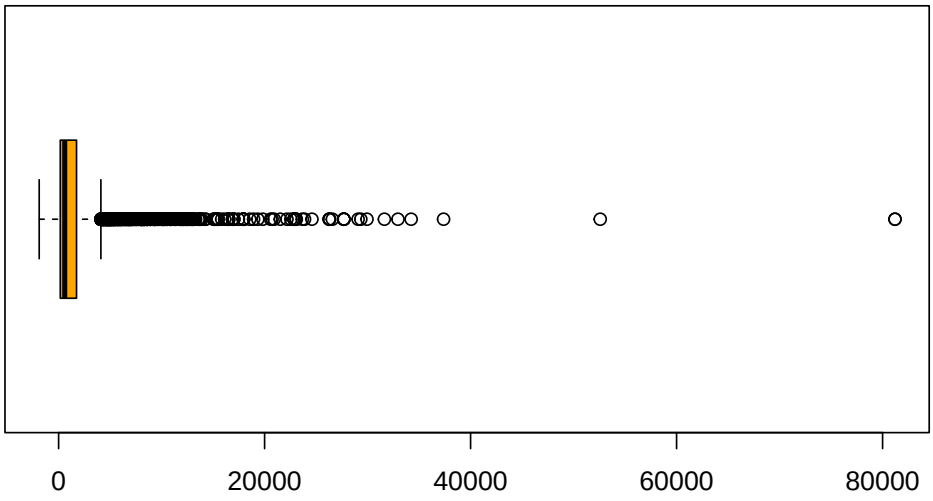


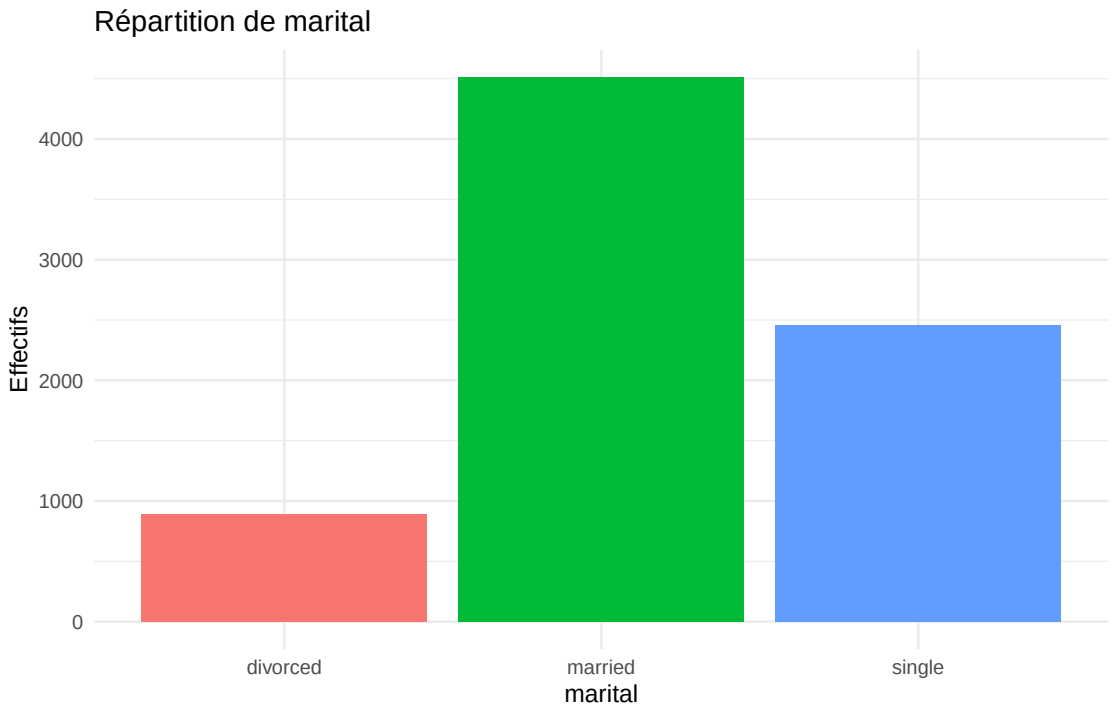


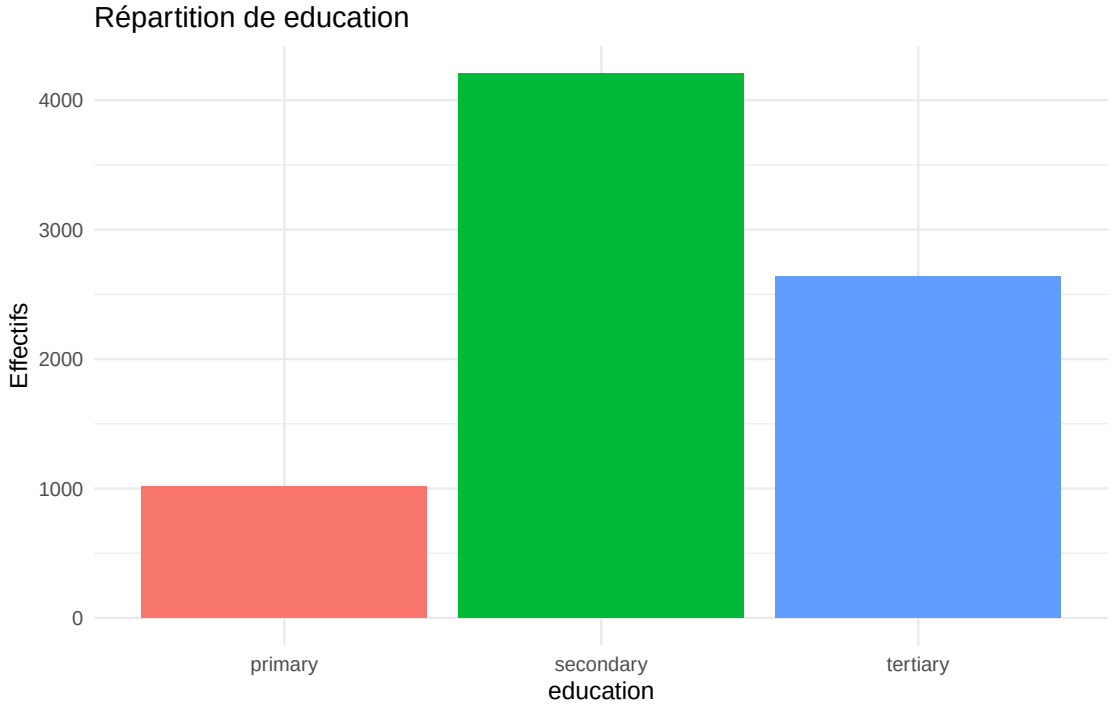
Boxplot de l'âge

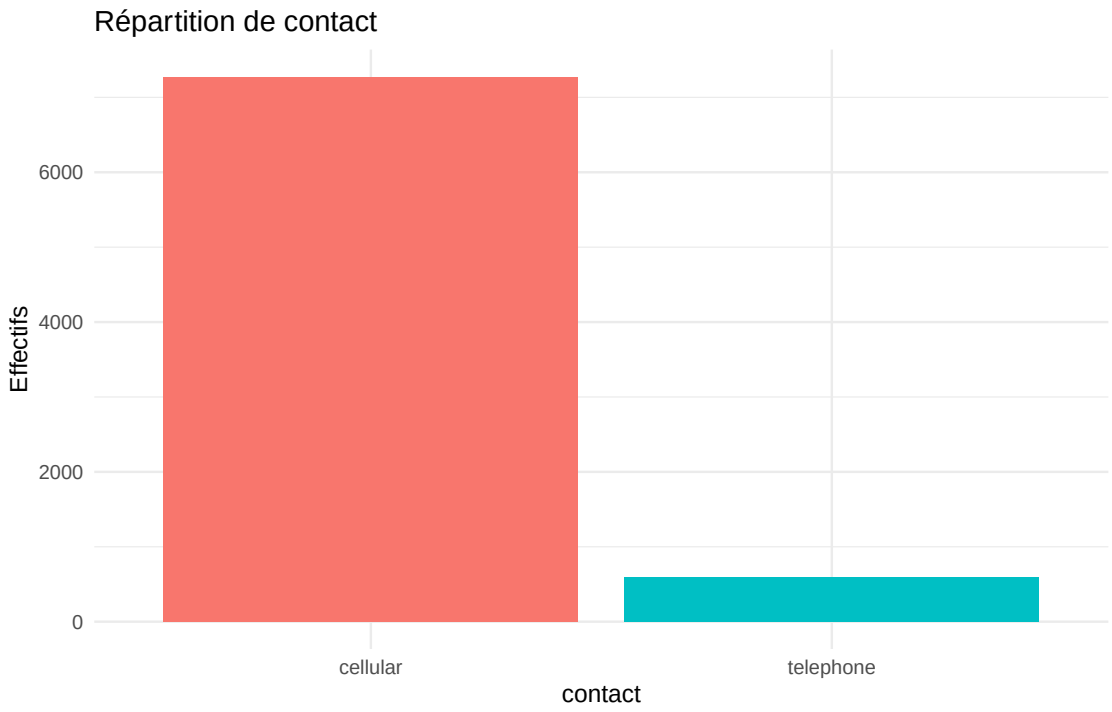


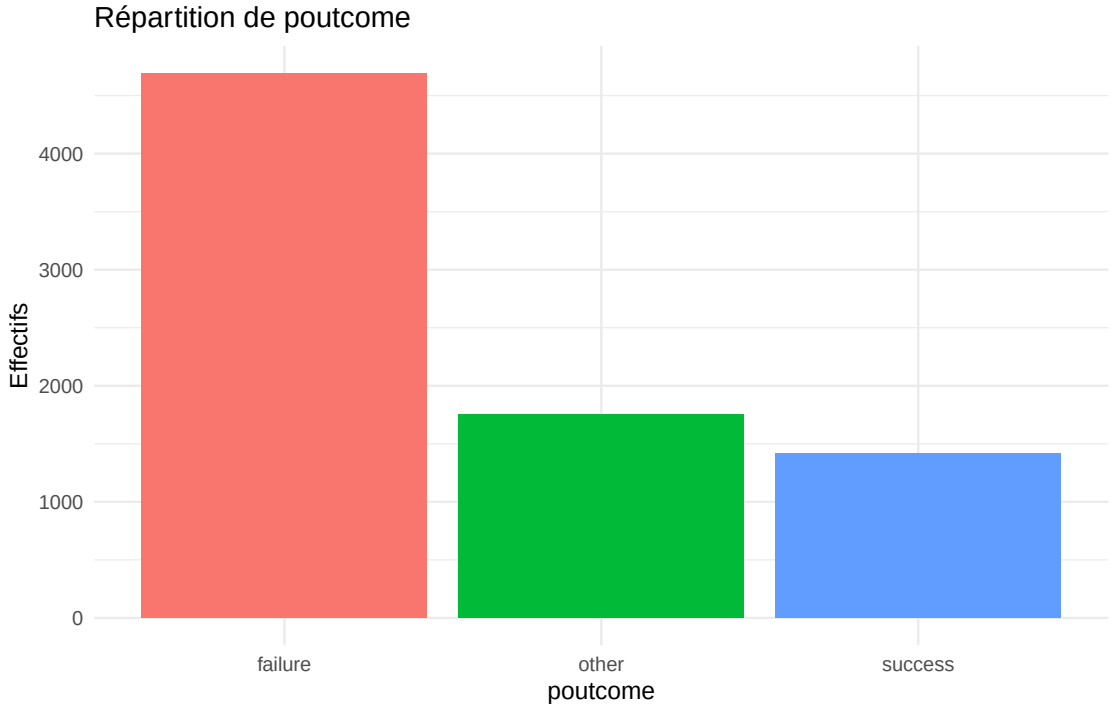
Boxplot du solde (balance)

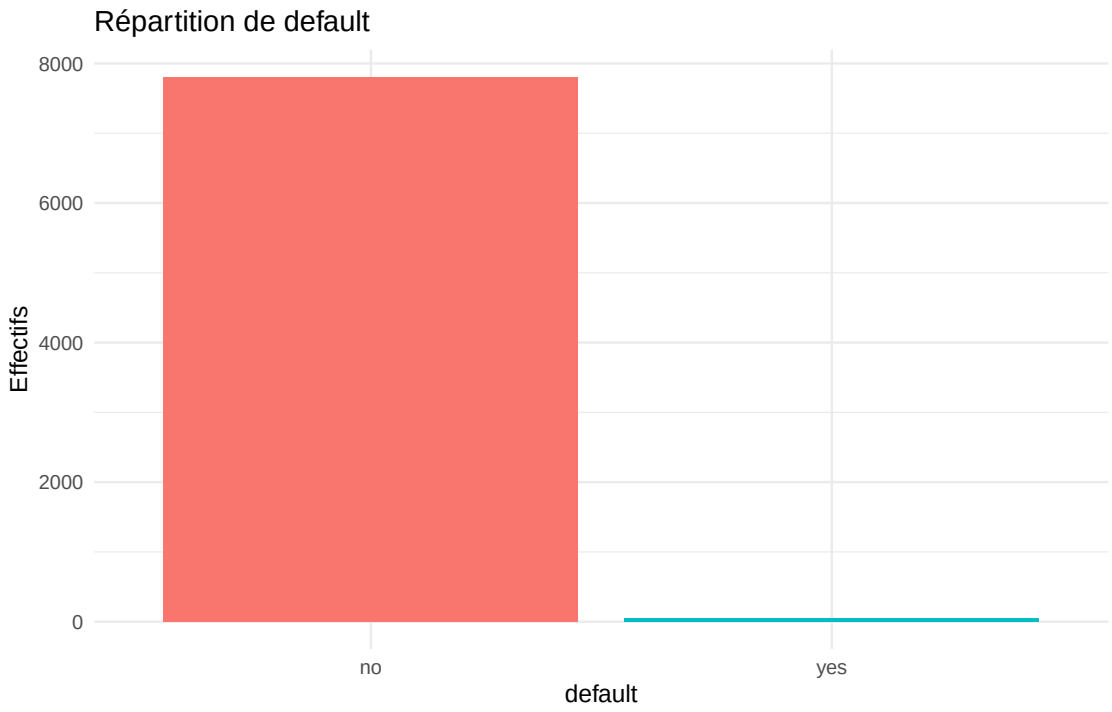


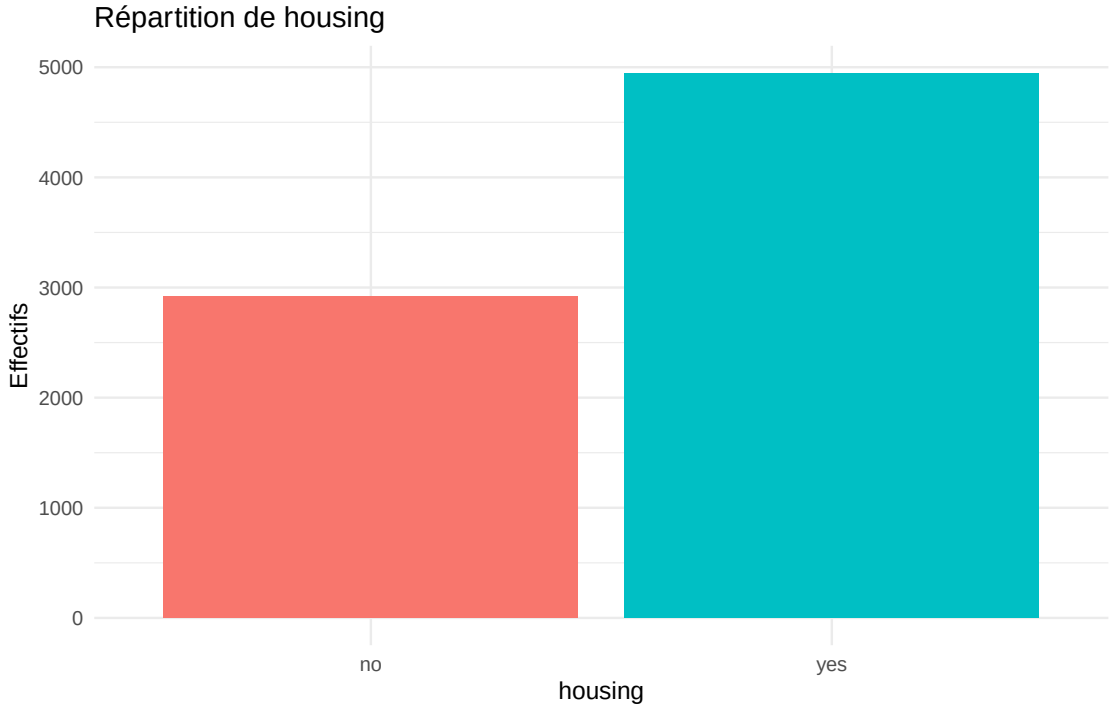

Répartition de marital
Effectifs
4000
3000
2000
1000
0
divorced
married
single
marital

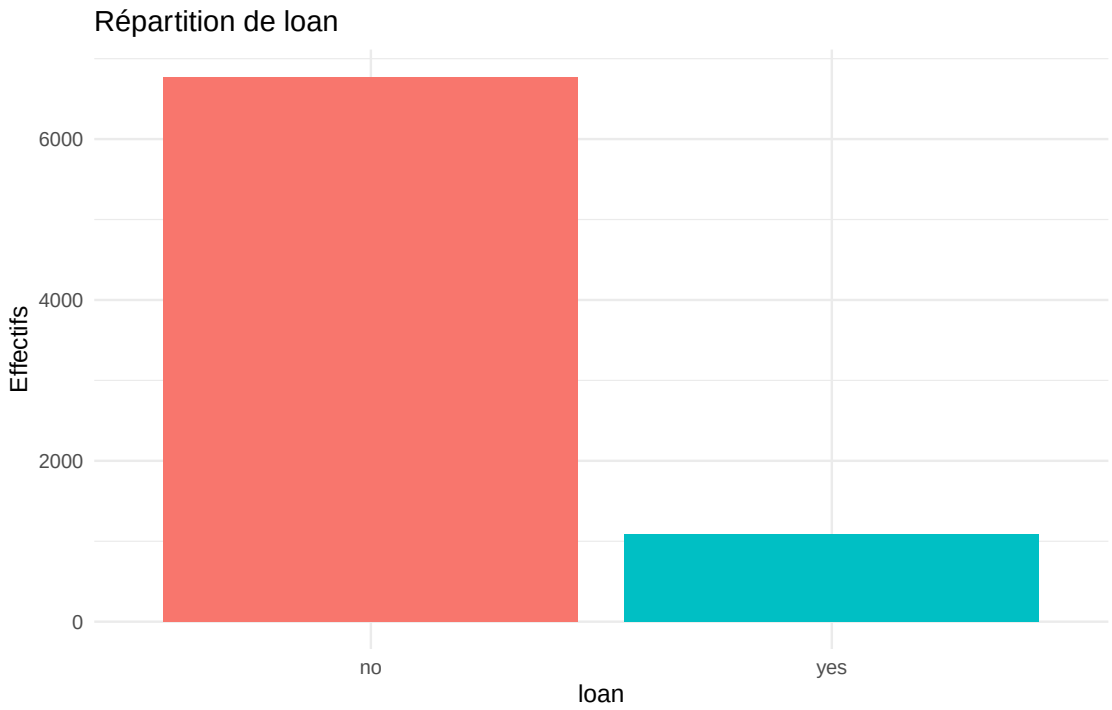

Répartition de education
Effectifs
4000
3000
2000
1000
0
primary
secondary
tertiary
education

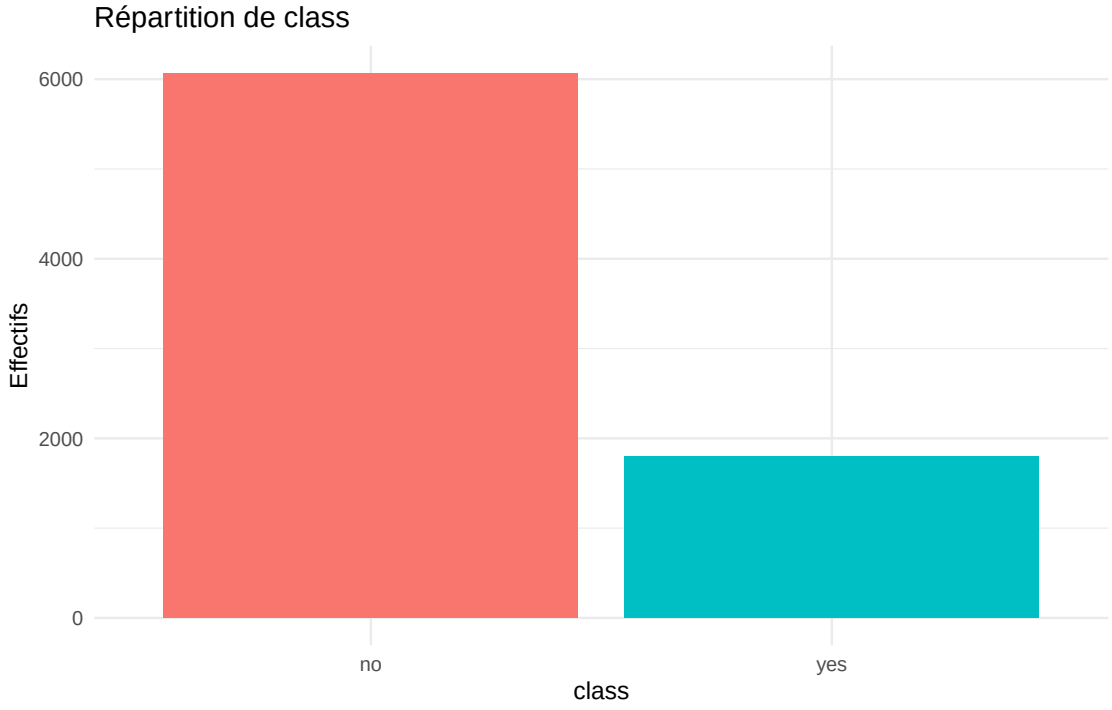

Répartition de contact
Effectifs
6000
4000
2000
0
cellular
telephone
contact


Répartition de poutcome
Effectifs
4000
3000
2000
1000
0
failure
other
success
poutcome


Répartition de default
8000
6000
4000
2000
0
Effectifs
no
yes
default


Répartition de housing
5000
4000
3000
2000
1000
0
Effectifs
no
yes
housing



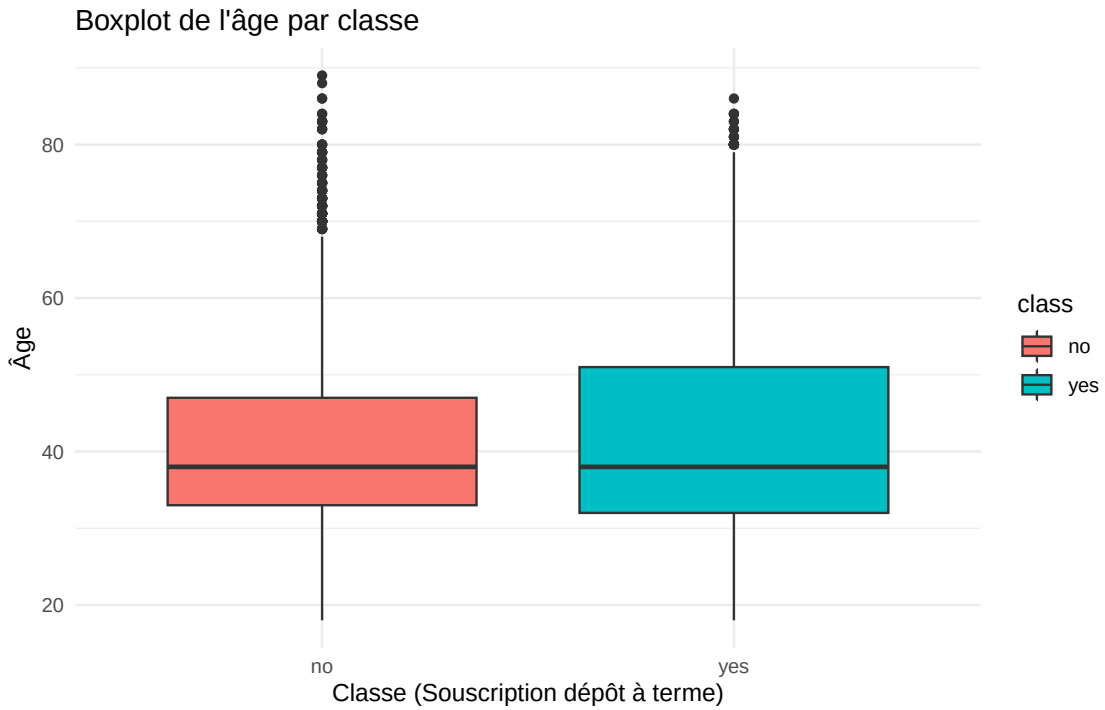


L'analyse univariée met en évidence un fort déséquilibre dans les données : la majorité des clients n'ont pas souscrit à un dépôt à terme (class = no), et la variable poutcome est dominée par la modalité failure. Du côté des variables numériques, l'âge est principalement concentré entre 30 et 40 ans, tandis que le solde des comptes présente une distribution asymétrique, avec la majorité des valeurs proches de zéro mais quelques soldes très élevés. Ces constats soulignent à la fois un déséquilibre de classes et la présence de valeurs extrêmes. Les valeurs extrêmes ne posent pas de problèmes car les arbres de décision et les forêts aléatoires

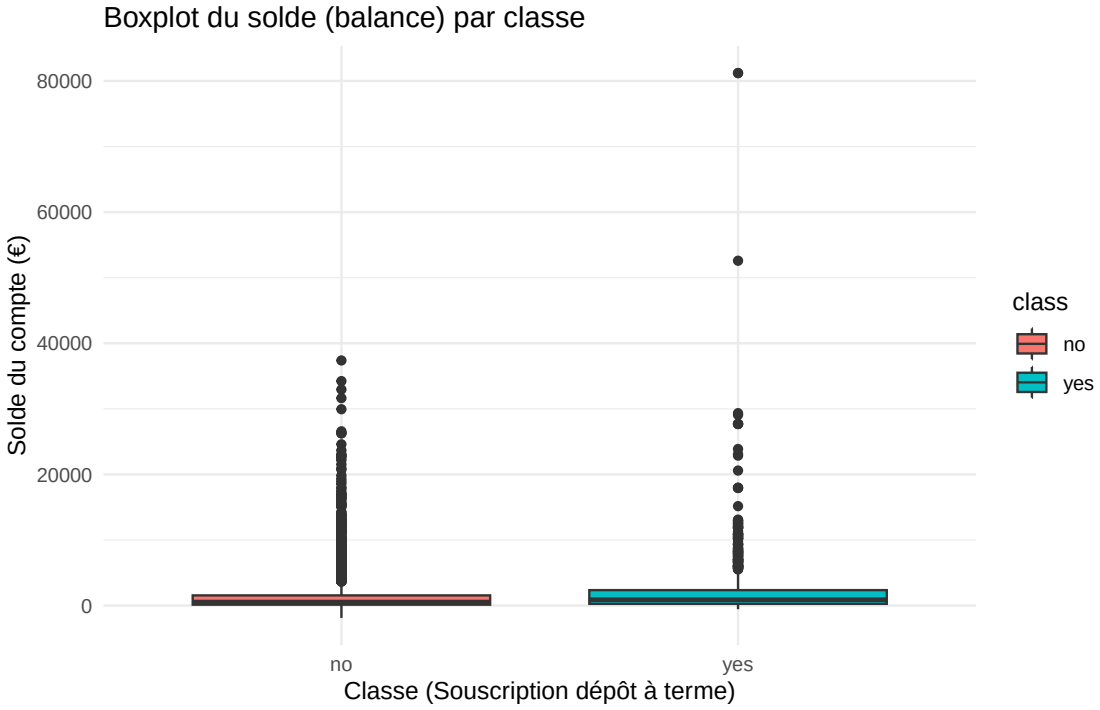
n'y sont pas sensibles. En revanche, ce déséquilibre issu de la variable class est à prendre en compte pour la modélisation.

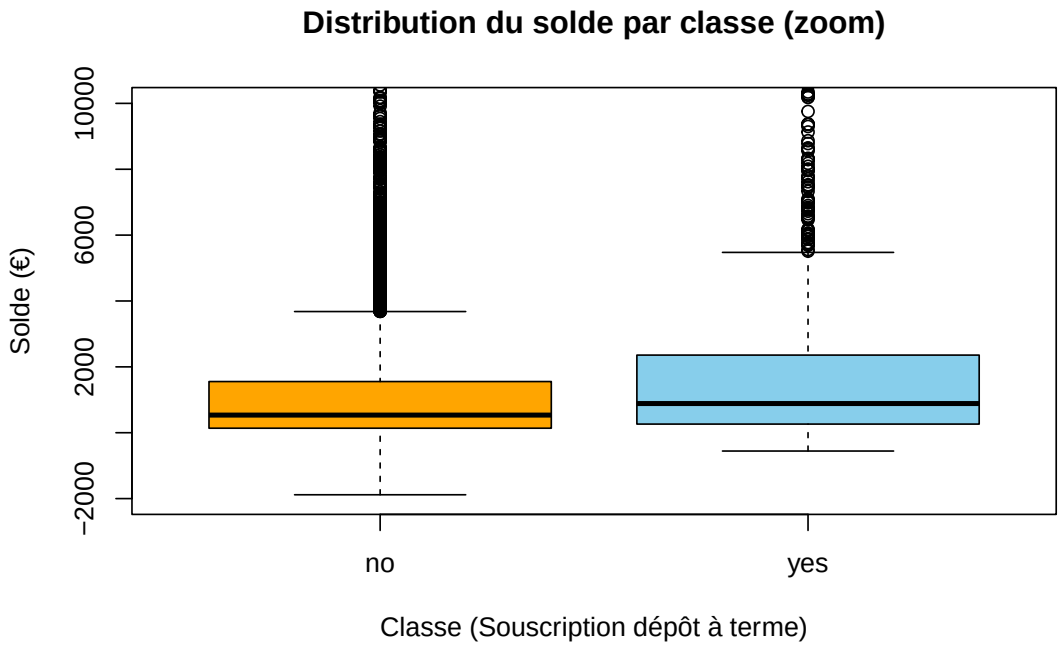
Cette première analyse univariée permet dès le départ de visualiser la distribution des variables et d'en obtenir une première compréhension.

Analyse bivariée



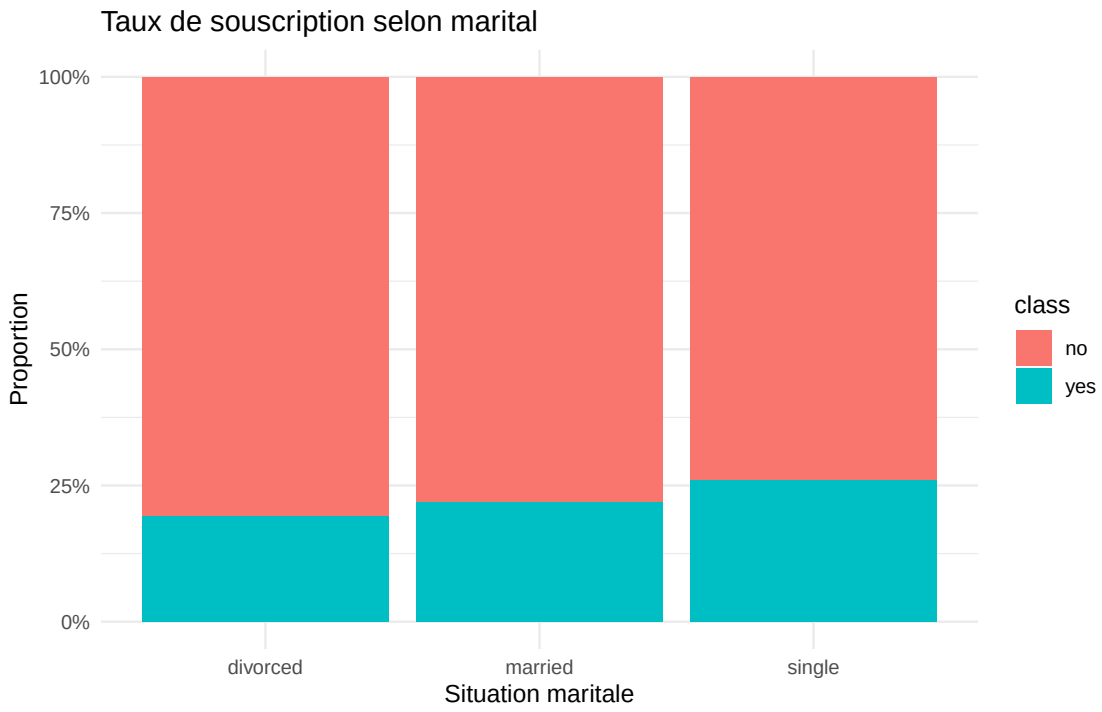

Densité de l'âge selon la classe
0.05
0.04
0.03
0.02
0.01
0.00
Densité
20
40
60
80
Âge
class
no
yes


Boxplot du solde (balance) par classe
80000
60000
40000
20000
0
Solde du compte (€)
no
yes
Classe (Souscription dépôt à terme)
class
no
yes



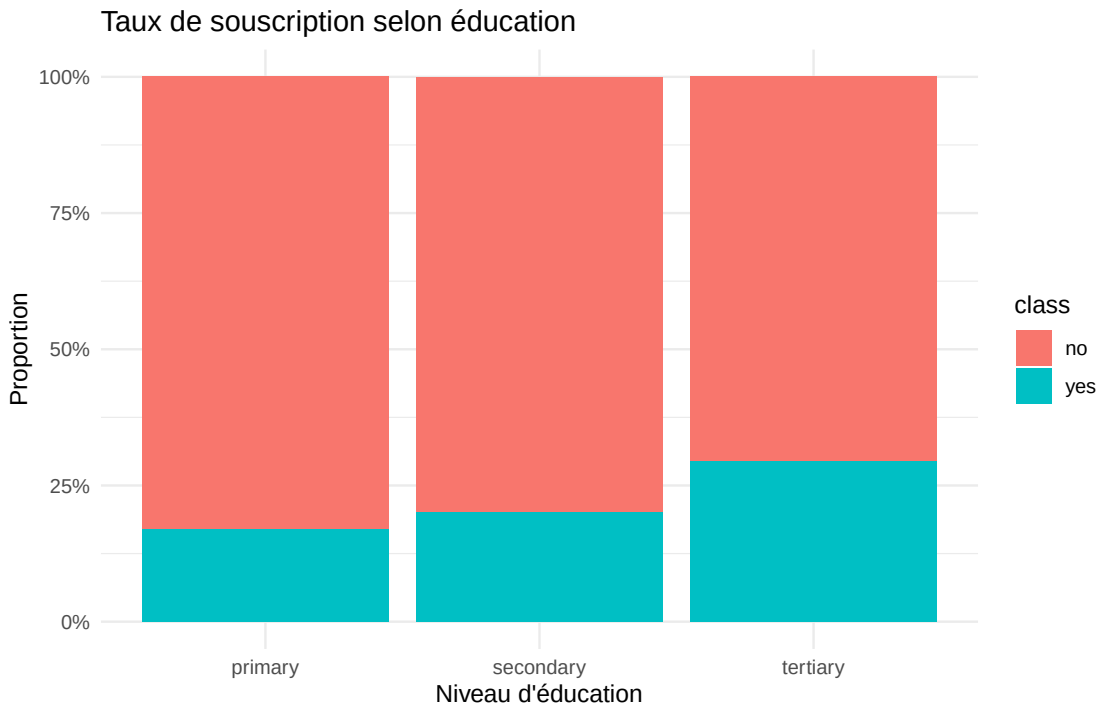

```
##
##              no  yes
##   divorced  718  172
##   married  3528  987
##   single   1822  637
```

```
##
##                    no       yes
##   divorced 0.8067416 0.1932584
##   married  0.7813953 0.2186047
##   single   0.7409516 0.2590484
```

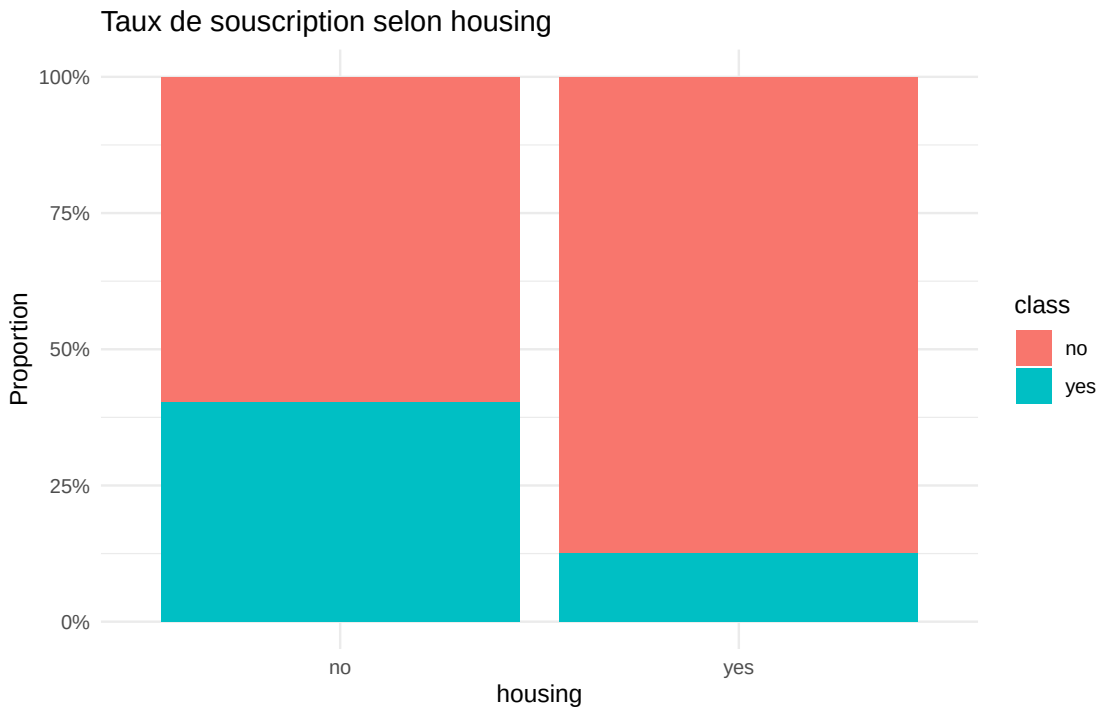



```
##
##                no  yes
##   primary     842  173
##   secondary  3365  845
##   tertiary   1861  778
```

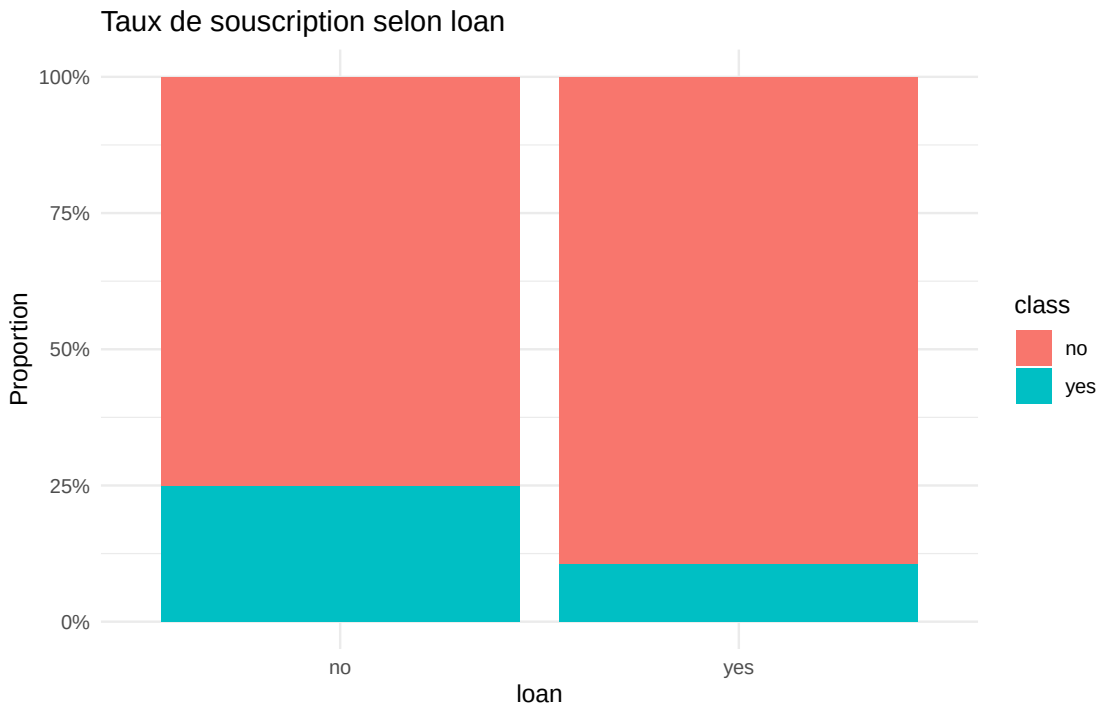
```
##
##                     no       yes
##   primary   0.8295567 0.1704433
##   secondary 0.7992874 0.2007126
##   tertiary  0.7051914 0.2948086
```



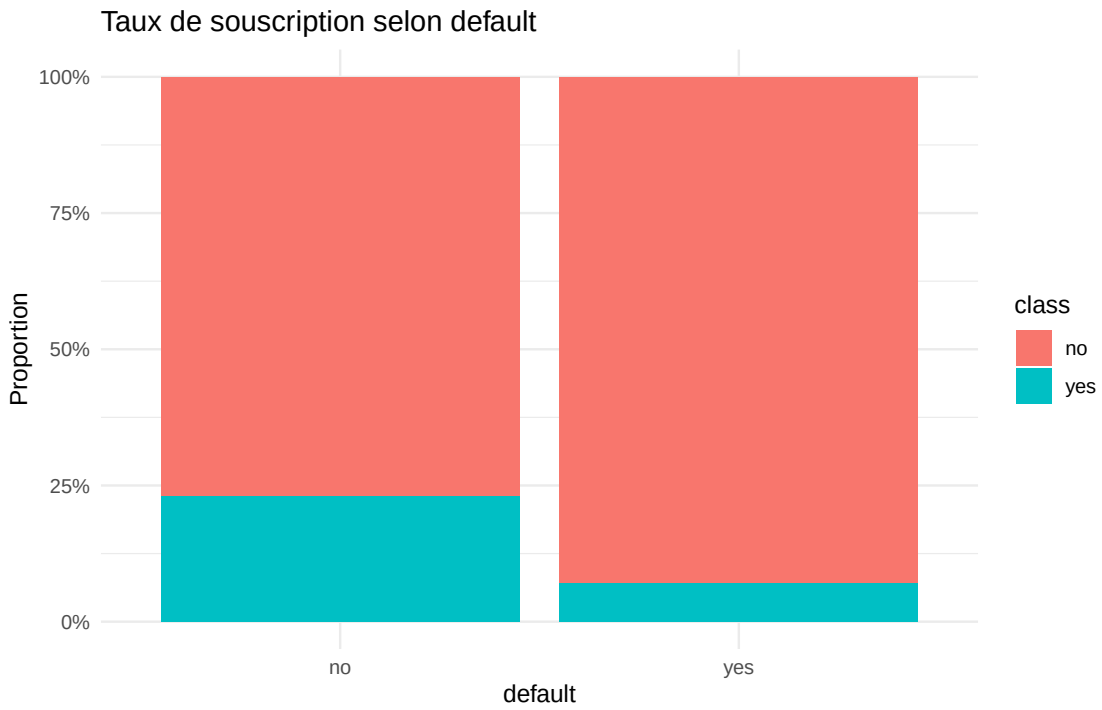

```
##
##         no  yes
##    no  1742 1175
##    yes 4326  621
##
##              no       yes
##    no  0.5971889 0.4028111
##    yes 0.8744694 0.1255306
```




```
##
##         no  yes
##    no  5093 1681
##    yes  975  115
##
##              no       yes
##    no  0.7518453 0.2481547
##    yes 0.8944954 0.1055046
```

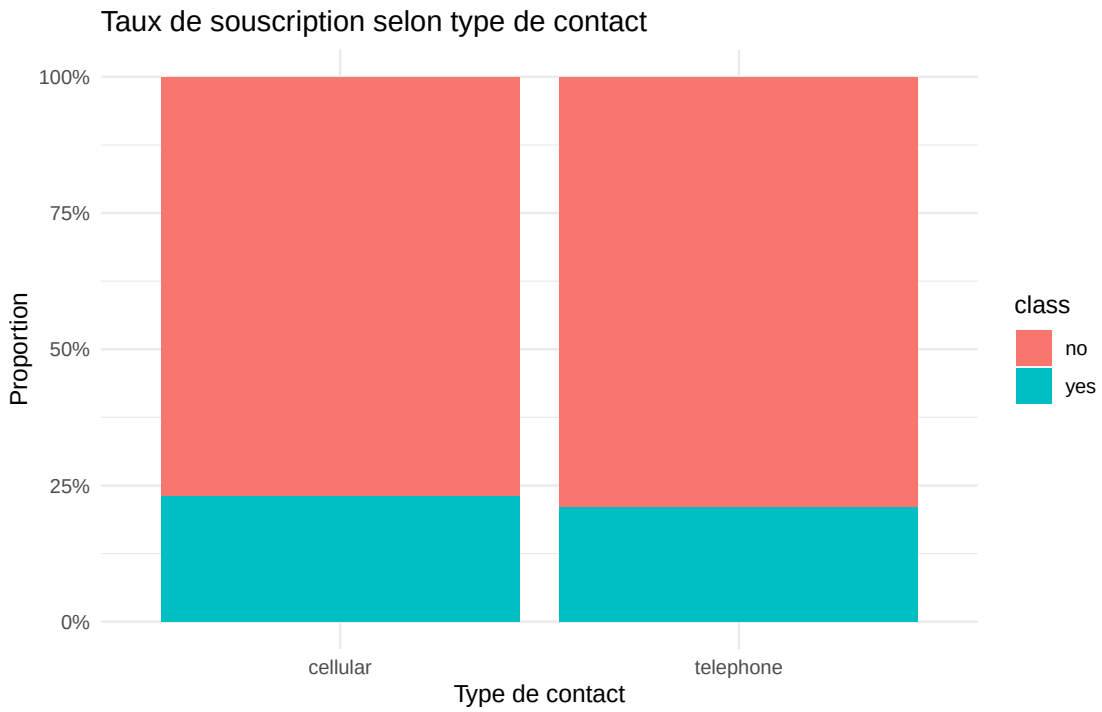



```
##
##         no  yes
##    no  6016 1792
##    yes   52    4
##
##                no        yes
##    no  0.77049180 0.22950820
##    yes 0.92857143 0.07142857
```



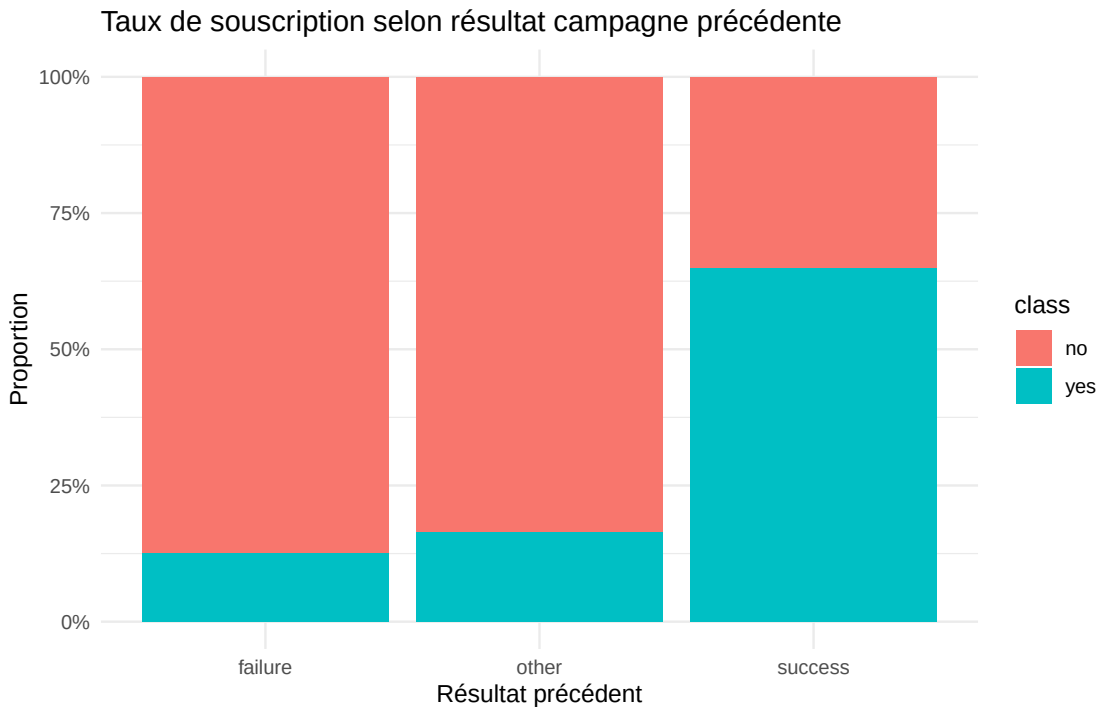

```
##
##              no  yes
##   cellular  5602 1672
##   telephone  466  124
```

```
##
##                    no       yes
##   cellular  0.7701402 0.2298598
##   telephone 0.7898305 0.2101695
```



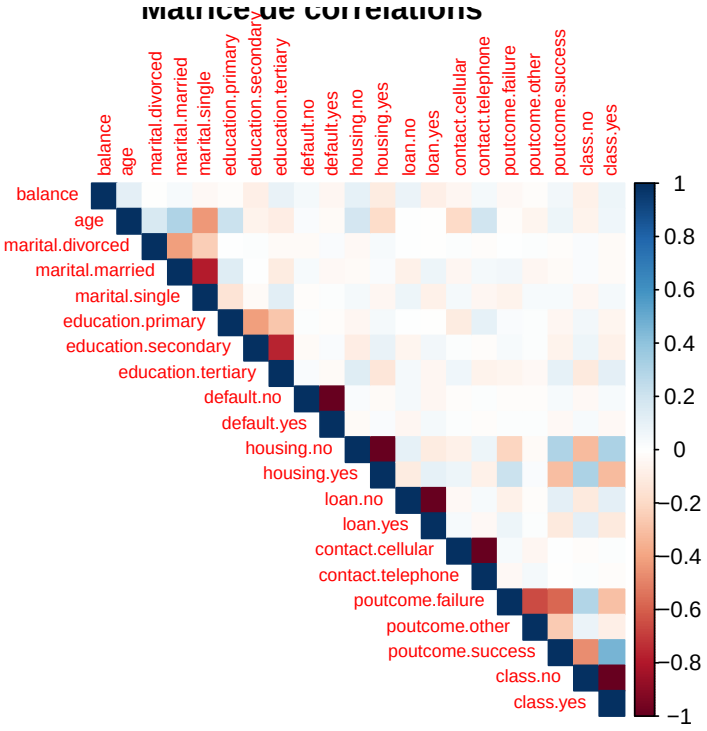

```
##
##              no  yes
##   failure  4107  587
##   other    1462  289
##   success   499  920
```

```
##
##                   no       yes
##   failure  0.8749467 0.1250533
##   other    0.8349515 0.1650485
##   success  0.3516561 0.6483439
```



L'analyse bivariée met en évidence plusieurs relations intéressantes avec la variable cible. Nous allons illustré notre porpos avec quelques exemples. Tout d'abord, la densité de l'âge selon la classe montre que les clients plus âgés ont tendance à souscrire davantage à un dépôt à terme, contrairement aux plus jeunes chez qui les refus sont plus fréquents. Ensuite, la variable poutcome illustre une relation claire : les clients dont la campagne précédente s'est soldée par un succès présentent un taux de souscription nettement plus élevé que ceux associés aux modalités failure ou other. Enfin, la variable housing révèle que les clients sans crédit logement sont proportionnellement plus enclins à souscrire, tandis que ceux ayant un prêt immobilier souscrivent beaucoup moins.

Analyse multidimensionnelle



La matrice de corrélations permet d'identifier les relations linéaires entre les variables. On observe que la variable cible class.yes est positivement corrélée avec poutcome.success, confirmant que les clients ayant connu un succès lors de la campagne précédente ont davantage tendance à souscrire. À l'inverse, elle est négativement corrélée avec poutcome.failure et housing.yes, ce qui reflète les tendances mises en évidence dans l'analyse bivariée. Concernant les variables explicatives entre elles, les corrélations restent globalement faibles, ce qui limite le risque de forte multicolinéarité.

En résumé, l'analyse descriptive du jeu de données révèle à la fois un déséquilibre de la variable cible, des relations significatives entre certaines caractéristiques (âge, résultat de campagne, crédit logement) et la probabilité de souscription, ainsi que des corrélations globalement faibles entre variables explicatives, ce qui constitue une base solide pour la phase de modélisation.

PARTIE I

Split Apprentissage

```
## [1] "no"  "yes"
```

```
## [1] 6292   10
```

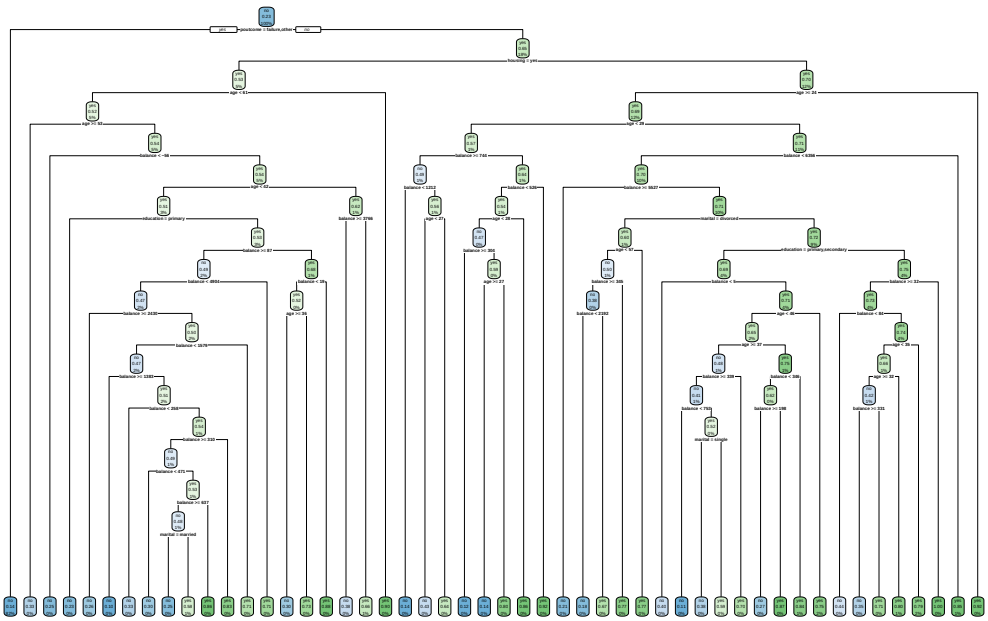
```
## [1] 1572   10
```

```
##
##        no       yes
## 0.7716147 0.2283853
```

```
##
##        no       yes
## 0.7716285 0.2283715
```

Dans cette première partie “Split apprentissage”, nous avons séparé l’échantillon en deux jeux de données : 80 % pour l’apprentissage et 20 % pour le test. Après cette séparation, on retrouve dans chaque jeu environ 77 % de “no” et 23 % de “yes”. Cela montre que le jeu de test reste représentatif de l’ensemble des données et pourra donc servir à évaluer correctement la capacité de généralisation des modèles. Ce point est important, car la classe “yes” étant minoritaire, il faut s’assurer que cette proportion soit bien conservée dans le jeu de test afin d’éviter un biais et d’obtenir des indicateurs de performance fiables.

Construction de notre arbre de décisions



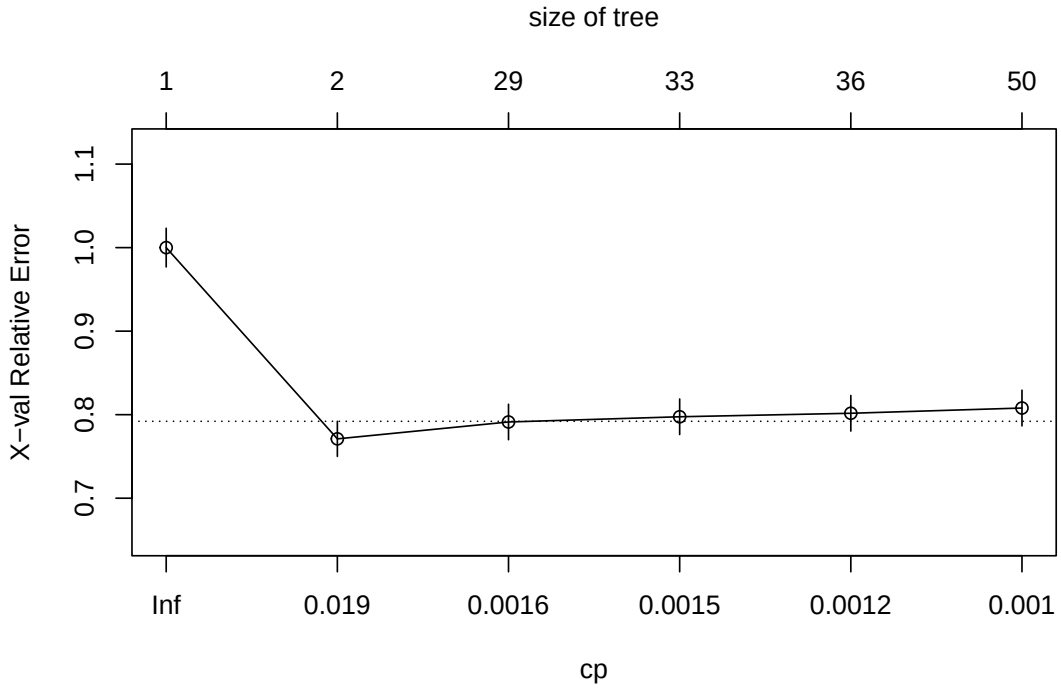
Nous avons construit un arbre de décision en fixant un paramètre de complexité petit. Ce choix permet à l’algorithme de conserver un grand nombre de divisions. Nous obtenons ainsi à un arbre relativement profond et ramifié, comme le montre la figure ci-dessus. Ce type de modèle peut s’avérer performant sur les données d’apprentissage car il capture de nombreuses nuances, mais il présente également un risque élevé de surapprentissage. En effet, l’arbre devient trop spécifique aux données du train et généralise moins bien sur de nouvelles observations.

Pour pallier ce problème, nous allons par la suite ajuster le paramètre de complexité par un processus d’élagage. La méthode consiste à tracer la courbe d’erreur en fonction de la valeur de cp et à retenir le cp optimal. Cette étape permettra de simplifier l’arbre et d’améliorer sa capacité de généralisation.

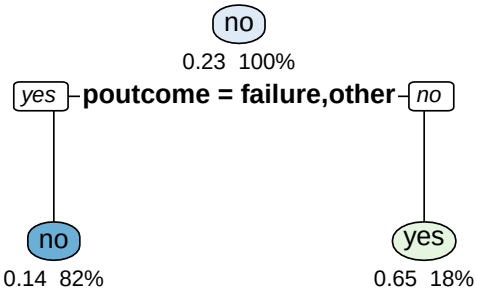
Optimisation de notre arbre de décisions

```
##
## Classification tree:
## rpart(formula = class ~ ., data = train, method = "class", control = rpart.control(cp = 0.001))
```

```
##
## Variables actually used in tree construction:
## [1] age       balance   education housing   marital   poutcome
##
## Root node error: 1437/6292 = 0.22839
##
## n= 6292
##
##          CP nsplit rel error  xerror     xstd
## 1 0.2289492      0   1.00000 1.00000 0.023172
## 2 0.0016594      1   0.77105 0.77105 0.021026
## 3 0.0016238     28   0.71608 0.79123 0.021239
## 4 0.0013918     32   0.70842 0.79749 0.021305
## 5 0.0010438     35   0.70424 0.80167 0.021348
## 6 0.0010000     49   0.68615 0.80793 0.021412
```




```
##
## Paramètre cp optimal retenu : 0.00165944
```

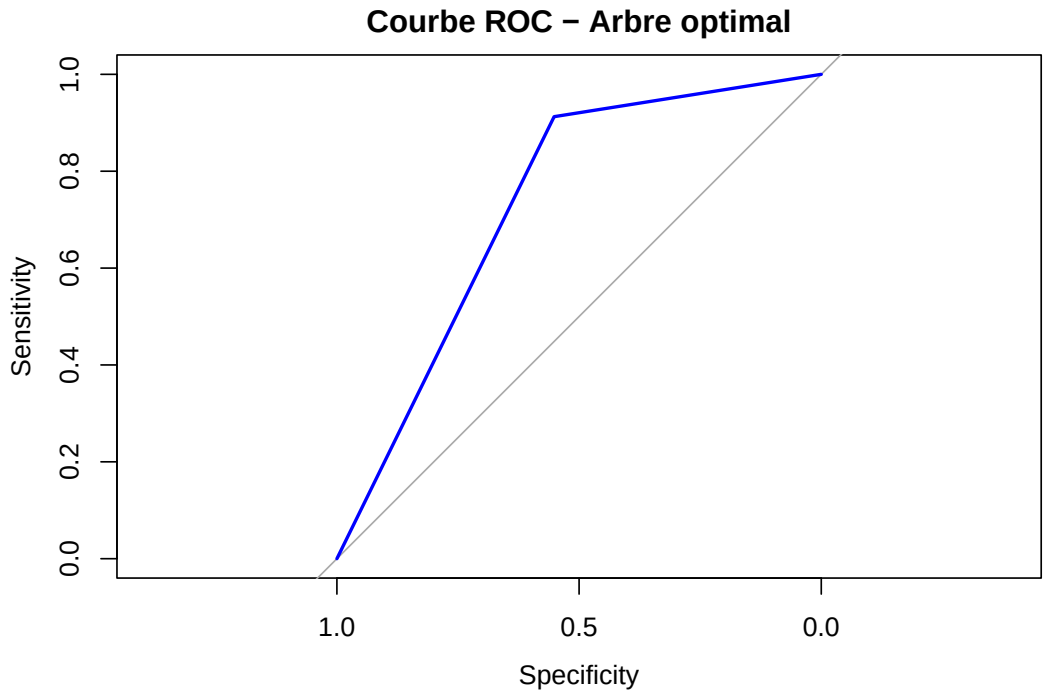


La racine de l'arbre présente une erreur de classification de 22,8 %, ce qui signifie qu'environ un quart des observations sont mal classées au départ. L'analyse de la table de complexité et de la courbe d'erreur montre que l'erreur de validation croisée diminue fortement dès le premier split, puis se stabilise et repart légèrement à la hausse lorsque l'arbre devient plus complexe. Le paramètre cp optimal retenu est donc 0.00165, car il correspond au minimum de l'erreur de validation croisée. Cela permet d'élaguer l'arbre pour conserver un modèle plus simple et robuste, évitant ainsi le surapprentissage tout en maintenant une bonne capacité de généralisation.

Evaluation sur le jeu de test

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  1107  161
##        yes  106  198
##
##                Accuracy : 0.8302
##                  95% CI : (0.8107, 0.8484)
##     No Information Rate : 0.7716
##     P-Value [Acc > NIR] : 6.713e-09
##
##                   Kappa : 0.4906
##
##  Mcnemar's Test P-Value : 0.0009506
##
##             Sensitivity : 0.5515
##             Specificity : 0.9126
##          Pos Pred Value : 0.6513
```

```
##          Neg Pred Value : 0.8730
##              Prevalence : 0.2284
##          Detection Rate : 0.1260
##    Detection Prevalence : 0.1934
##       Balanced Accuracy : 0.7321
##
##        'Positive' Class : yes
##

## AUC = 0.7320727
```



L'évaluation du modèle optimal sur les données de test montre une accuracy globale de 83 %, significativement supérieure au taux de référence (No Information Rate = 77 %). La matrice de confusion met en évidence une bonne capacité du classifieur à reconnaître les clients n'ayant pas souscrit, avec une spécificité élevée de 91,3 %. En revanche, la sensibilité reste plus limitée (55,1 %), ce qui signifie qu'un peu moins de six clients souscripteurs sur dix sont correctement identifiés. Ce déséquilibre entre précision sur les "no" et rappel sur les "yes" est courant dans des jeux de données où la classe positive est minoritaire. La valeur prédictive positive (65,1 %) et la valeur prédictive négative (87,3 %) confirment cette tendance. Les prédictions "no" sont plus fiables que les prédictions "yes". Le coefficient de Kappa (0,49) traduit un accord modéré entre les prédictions et la réalité. L'analyse de la courbe ROC indique une bonne capacité de discrimination globale du modèle, supérieure à un classifieur aléatoire (AUC = 0,5), mais encore perfectible pour atteindre des standards élevés supérieurs à 0.8. La courbe ROC présente une montée rapide dès les premiers seuils, ce qui traduit une aptitude du modèle à distinguer efficacement les deux classes dès les probabilités les plus faibles.

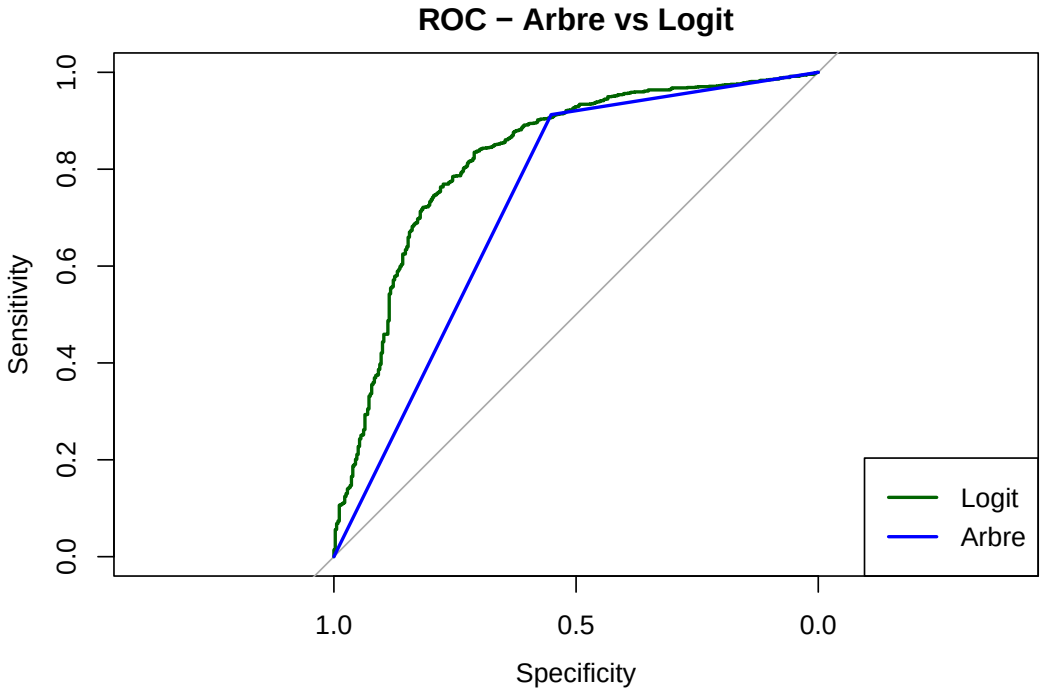
En conclusion, le modèle présente une performance satisfaisante avec une bonne précision globale et une capacité discriminante correcte, mais il privilégie la détection des non-souscripteurs au détriment des souscripteurs. Selon les objectifs de l'étude, il pourrait être pertinent de diminuer le seuil de décision (actuellement fixé à 0,5) afin d'augmenter le rappel sur la classe "yes", quitte à introduire davantage de faux positifs. Cette approche permettrait de maximiser la détection des clients susceptibles de souscrire, ce qui constitue souvent l'objectif prioritaire dans un contexte marketing.

Modèle de régression logistique

```
##
## Call:
## glm(formula = class ~ ., family = binomial, data = train)
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -2.273e+00  2.556e-01  -8.894  < 2e-16 ***
## age                 8.613e-03  3.528e-03   2.441 0.014646 *
## maritalmarried      2.790e-01  1.214e-01   2.298 0.021579 *
## maritalsingle       4.283e-01  1.365e-01   3.138 0.001700 **
## educationsecondary  2.831e-01  1.242e-01   2.280 0.022625 *
## educationtertiary   5.398e-01  1.286e-01   4.199 2.68e-05 ***
## defaultyes         -7.133e-01  6.225e-01  -1.146 0.251863
## balance             2.401e-05  1.064e-05   2.257 0.024000 *
## housingyes         -9.556e-01  7.305e-02 -13.081  < 2e-16 ***
## loanyes            -5.765e-01  1.234e-01  -4.671 2.99e-06 ***
## contacttelephone   -3.062e-01  1.377e-01  -2.224 0.026154 *
## poutcomeother       3.305e-01  8.868e-02   3.727 0.000194 ***
## poutcomesuccess     2.170e+00  8.351e-02  25.986  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6761.6  on 6291  degrees of freedom
## Residual deviance: 5289.3  on 6279  degrees of freedom
## AIC: 5315.3
##
## Number of Fisher Scoring iterations: 5
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  1141  198
##        yes   72  161
##
##                Accuracy : 0.8282
##                  95% CI : (0.8087, 0.8466)
##     No Information Rate : 0.7716
##     P-Value [Acc > NIR] : 2.041e-08
##
##                   Kappa : 0.444
##
##  Mcnemar's Test P-Value : 2.800e-14
##
##             Sensitivity : 0.4485
##             Specificity : 0.9406
##          Pos Pred Value : 0.6910
##          Neg Pred Value : 0.8521
##              Prevalence : 0.2284
##          Detection Rate : 0.1024
```

```
##    Detection Prevalence : 0.1482
##       Balanced Accuracy : 0.6946
##
##        'Positive' Class : yes
##

## AUC régression logistique = 0.8247938
```



L'arbre optimal atteint une accuracy de 83 % et une AUC de 0,73, avec une bonne capacité à identifier les non-souscripteurs (spécificité de 91 %) mais une sensibilité plus limitée (55 %). La régression logistique présente une accuracy comparable (82,8 %) mais une AUC supérieure (0,82), traduisant une meilleure capacité globale de discrimination entre les classes. On observe toutefois que la sensibilité du modèle logistique (44,8 %) est inférieure à celle de l'arbre, alors que sa spécificité (94 %) est plus élevée. Il privilégie ainsi fortement la détection des "no" au détriment des "yes".

De plus, la régression logistique présente l'avantage d'être plus interprétable d'un point de vue statistique. Les coefficients estimés permettent de quantifier l'effet de chaque variable explicative. Par exemple, un résultat positif à la campagne précédente augmente la probabilité de souscription, tandis qu'un prêt immobilier la réduit. L'arbre de décision est quant à lui plus intuitif visuellement et offre une segmentation opérationnelle claire des clients.

En conclusion, la régression logistique se distingue par sa meilleure calibration probabiliste et son pouvoir discriminant global avec un AUC plus élevé, tandis que l'arbre reste un outil de décision pratique et lisible pour une segmentation marketing.

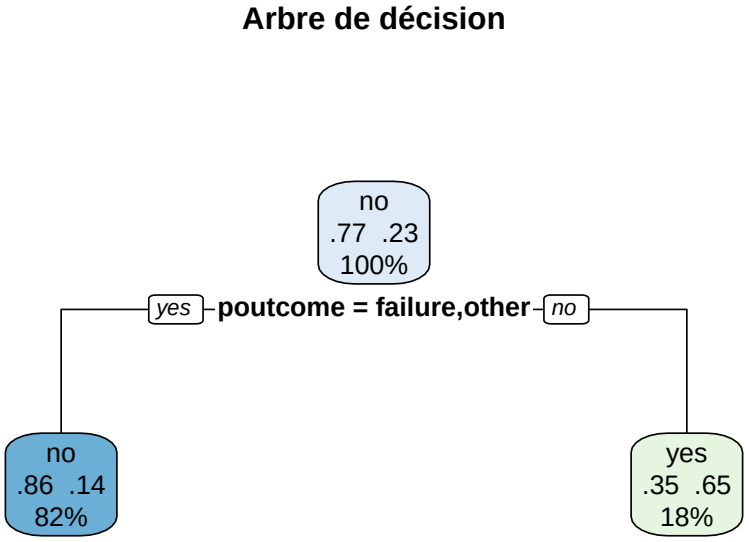
PARTIE II

Schéma de validation et métriques

Nous avons ensuite mis en place un schéma de validation pour évaluer et comparer les deux modèles de forêts aléatoires. Le jeu d'apprentissage est soumis à une validation croisée à 5 folds, répétée deux fois afin

de limiter la variance des estimations. Comme la variable “class” est déséquilibrée, nous appliquons un down sampling à chaque fold afin d’éviter que le modèle n’apprenne à prédire uniquement “no”. Les performances sont alors mesurées à l’aide de plusieurs indicateurs : l’AUC/ROC, la sensibilité et la spécificité. Enfin, une évaluation finale est réalisée sur le jeu de test pour estimer la capacité de généralisation de nos deux modèles.

Arbre de décision illustratif

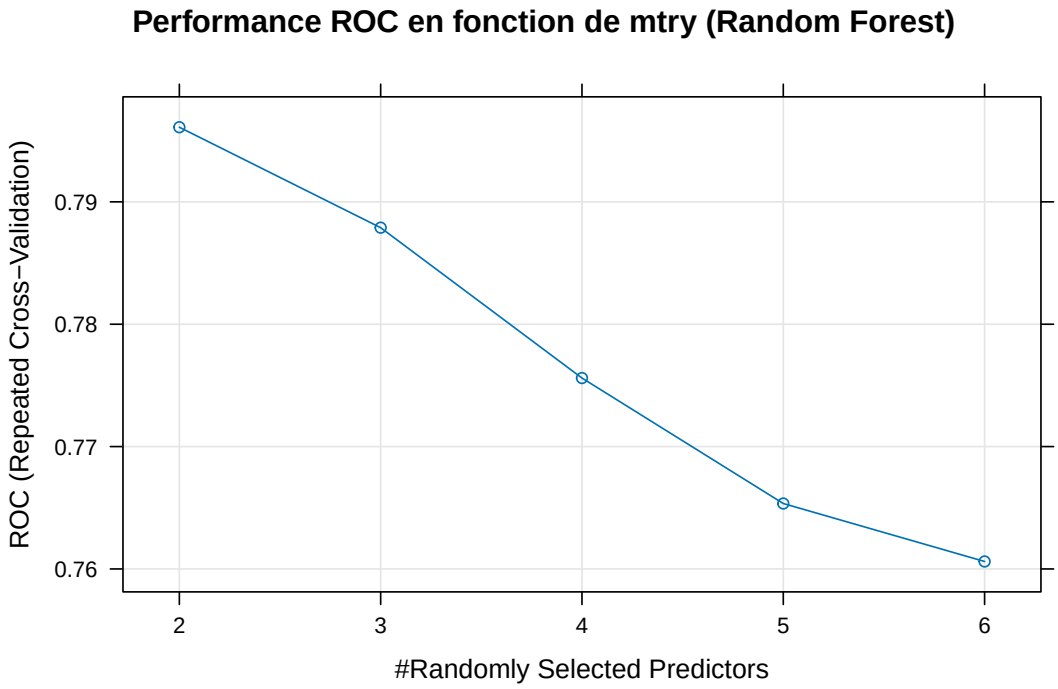


Avant de passer à la modélisation avec les deux forêts aléatoires, nous avons ajouté un arbre de décision à titre de rappel. Celui-ci met en évidence que la variable “poutcome” joue un rôle central dans la prédiction : lorsque poutcome vaut “failure” ou “other”, la probabilité de non-souscription atteint 86 %, tandis qu’en cas de “success”, 65 % des clients souscrivent à nouveau. Cet arbre illustre bien la logique de segmentation, mais il reste limité puisqu’il ne repose que sur une règle principale et demeure instable. Pour obtenir un modèle plus robuste et généralisable, il est donc nécessaire d’utiliser une forêt aléatoire, issue de la combinaison de nombreux arbres.

Implémentation A

```
## Random Forest
##
## 6292 samples
##    9 predictor
##    2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 5033, 5034, 5034, 5033, 5034, 5034, ...
## Addtional sampling using down-sampling
##
```

```
## Resampling results across tuning parameters:
##
##   mtry  ROC        Sens       Spec
##   2     0.7960999  0.7609681  0.7212979
##   3     0.7878895  0.7740474  0.6969440
##   4     0.7756015  0.7548919  0.6837193
##   5     0.7653465  0.7355304  0.6795332
##   6     0.7606105  0.7316169  0.6746625
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

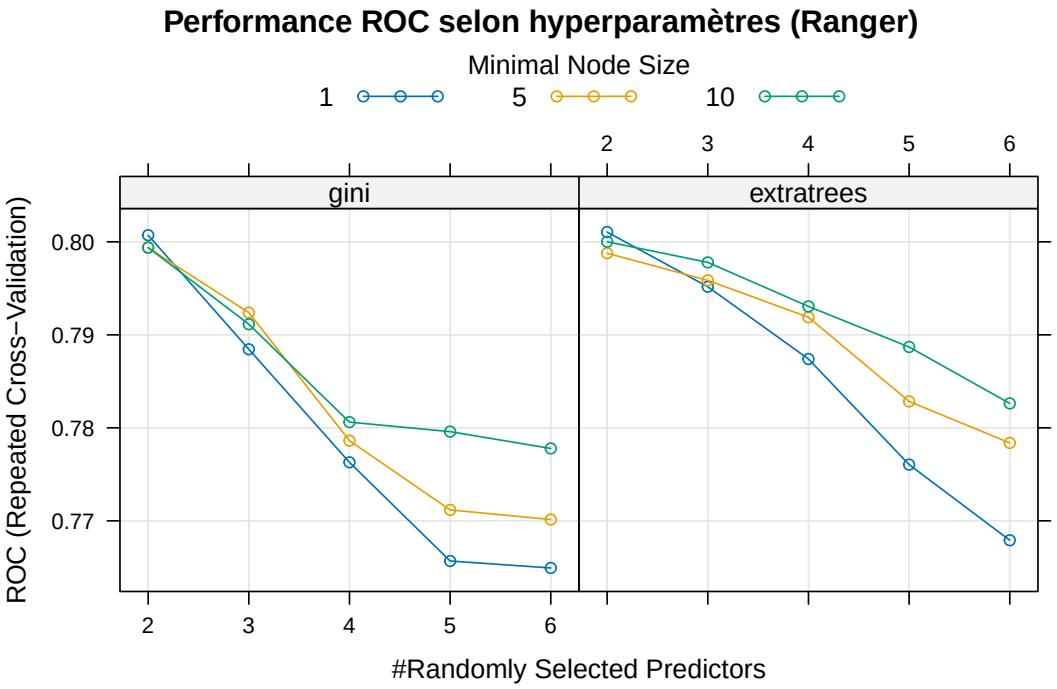


Notre premier modèle repose sur l'implémentation classique du Random Forest. Nous avons testé différentes valeurs de l'hyperparamètre mtry, qui correspond au nombre de variables candidates sélectionnées aléatoirement à chaque division de nœud. L'évaluation a été réalisée selon le schéma de validation décrit précédemment. Les résultats montrent que la performance, mesurée par l'AUC (ROC), diminue lorsque mtry augmente. Le maximum est atteint pour mtry = 2 avec une AUC de 0,796, puis elle baisse progressivement jusqu'à 0,761 pour mtry = 6. En effet un nombre trop important de variables à chaque split réduit la diversité des arbres et favorise le sur-apprentissage, d'pù cette évolution de l'AUC. Le modèle optimal retenu est donc celui avec mtry = 2. Il offre également le meilleur compromis entre sensibilité (0,761) et spécificité (0,721). Ce réglage maximise la capacité discriminante de la forêt aléatoire et améliore la détection des souscripteurs, objectif principal de notre étude.

Imlplémentation B

```
## Random Forest
##
## 6292 samples
##    9 predictor
```

```
##    2 classes: 'no', 'yes'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 5033, 5034, 5034, 5033, 5034, 5034, ...
## Addtional sampling using down-sampling
##
## Resampling results across tuning parameters:
##
##   mtry  splitrule   min.node.size  ROC        Sens       Spec
##   2     gini         1             0.8007139  0.7642636  0.7115600
##   2     gini         5             0.7993618  0.7645726  0.7108510
##   2     gini        10             0.7993878  0.7647786  0.7098166
##   2     extratrees   1             0.8010513  0.7400618  0.7411126
##   2     extratrees   5             0.7987748  0.7393409  0.7376307
##   2     extratrees  10             0.8000054  0.7461380  0.7331252
##   3     gini         1             0.7884455  0.7573635  0.6948497
##   3     gini         5             0.7923848  0.7670443  0.7007658
##   3     gini        10             0.7911377  0.7731205  0.7004126
##   3     extratrees   1             0.7951706  0.7617920  0.7087604
##   3     extratrees   5             0.7958545  0.7631308  0.7042441
##   3     extratrees  10             0.7977894  0.7680742  0.7059826
##   4     gini         1             0.7762893  0.7403708  0.6955369
##   4     gini         5             0.7786187  0.7502575  0.6885768
##   4     gini        10             0.7806119  0.7610711  0.6896281
##   4     extratrees   1             0.7874000  0.7597322  0.6885876
##   4     extratrees   5             0.7918877  0.7574665  0.7028383
##   4     extratrees  10             0.7930536  0.7665294  0.7021717
##   5     gini         1             0.7656787  0.7329557  0.6857808
##   5     gini         5             0.7711736  0.7455201  0.6889385
##   5     gini        10             0.7795900  0.7601442  0.6896184
##   5     extratrees   1             0.7760431  0.7507724  0.6795393
##   5     extratrees   5             0.7828391  0.7579815  0.6924289
##   5     extratrees  10             0.7886899  0.7686921  0.6858050
##   6     gini         1             0.7649321  0.7247168  0.6812827
##   6     gini         5             0.7701391  0.7423275  0.6844210
##   6     gini        10             0.7777837  0.7455201  0.6972706
##   6     extratrees   1             0.7679127  0.7320288  0.6798829
##   6     extratrees   5             0.7783757  0.7497425  0.6850973
##   6     extratrees  10             0.7826265  0.7640577  0.6805906
##
## ROC was used to select the optimal model using the largest value.
## The final values used for the model were mtry = 2, splitrule = extratrees
##  and min.node.size = 1.
```



Notre second modèle repose sur l'implémentation Ranger, choisie car elle permet un réglage plus fin des hyperparamètres, même si son coût de calcul est plus élevé que celui du Random Forest. Elle offre notamment la possibilité de contrôler la structure des arbres via différents paramètres : min.node.size (taille minimale des nœuds terminaux, jouant un rôle d'élagage implicite), splitrule (critère de séparation, gini ou extratrees), ainsi que mtry (nombre de variables candidates par split). Ces paramètres ont été optimisés à l'aide de notre schéma de validation. Les résultats montrent que la performance est maximale pour la configuration suivante : mtry = 2, splitrule = extratrees et min.node.size = 1, avec une AUC de 0,801. Cette combinaison favorise la diversité des arbres (faible mtry), des séparations plus aléatoires (extratrees) et une granularité plus fine (nœuds très petits), ce qui améliore la capacité de discrimination du modèle. À l'inverse, lorsque mtry augmente ou que min.node.size devient trop élevé, l'AUC décroît car la forêt perd en diversité et en précision. Le modèle optimal retenu permet ainsi de maximiser la détection des souscripteurs tout en maintenant un bon équilibre entre sensibilité (0,740) et spécificité (0,741).

Sélection du modèle le plus optimal

```
## AUC RandomForest : 0.7961
```

```
## AUC Ranger       : 0.8011
```

```
## ==> Modèle retenu : Ranger (package ranger)
```

```
## Hyperparamètres optimaux :
```

```
##   mtry  splitrule min.node.size
## 4    2 extratrees             1
```

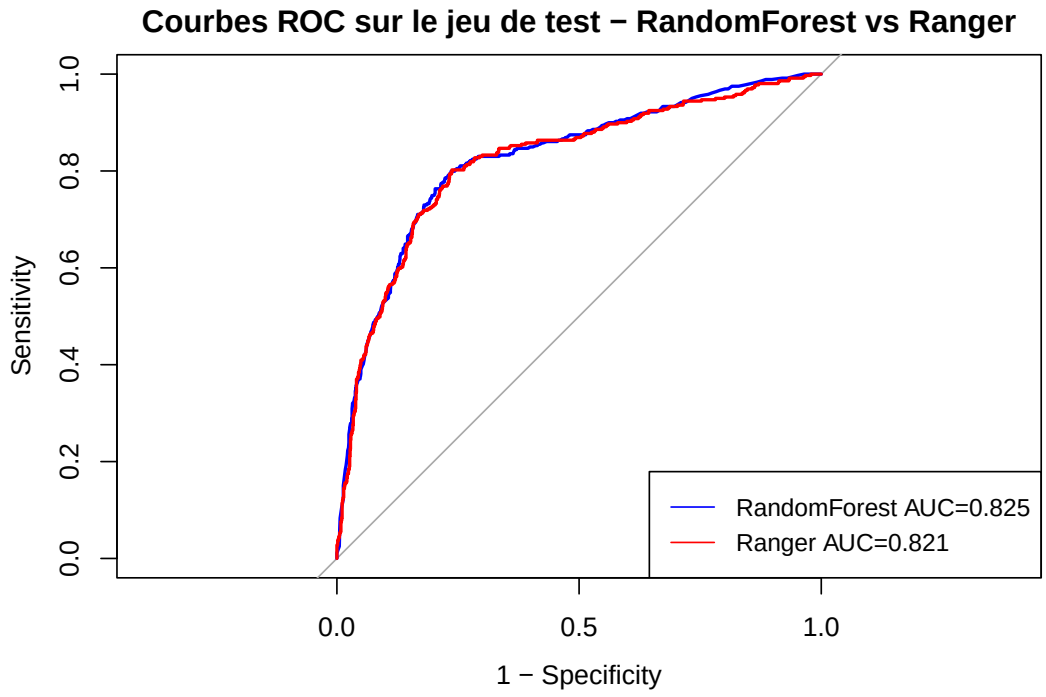
La comparaison des deux implémentations met en évidence que la forêt aléatoire (randomForest) atteint une AUC maximale de 0,796, tandis que l'implémentation Ranger obtient un score légèrement supérieur,

à 0,802. Bien que l'écart soit limité, il souligne l'intérêt de Ranger, qui offre une plus grande flexibilité dans l'ajustement des hyperparamètres. Le modèle optimal retenu correspond à la configuration mtry = 2, splitrule = extratrees et min.node.size = 1. Ce paramétrage favorise une forte diversité entre les arbres et améliore la capacité discriminante de la forêt sur un jeu de données déséquilibré. Ainsi, le modèle Ranger est jugé plus performant et mieux adapté pour la suite de l'analyse, puisqu'il maximise la détection des souscripteurs tout en conservant une bonne généralisation.

Evaluation sur le jeu de test

```
## === RandomForest ===

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  926  72
##        yes 287 287
##
##                Accuracy : 0.7716
##                  95% CI : (0.7501, 0.7922)
##     No Information Rate : 0.7716
##     P-Value [Acc > NIR] : 0.5141
##
##                   Kappa : 0.4648
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7994
##             Specificity : 0.7634
##          Pos Pred Value : 0.5000
##          Neg Pred Value : 0.9279
##              Prevalence : 0.2284
##          Detection Rate : 0.1826
##    Detection Prevalence : 0.3651
##       Balanced Accuracy : 0.7814
##
##        'Positive' Class : yes
##


##
## === Ranger ===

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  915  71
##        yes 298 288
##
##                Accuracy : 0.7653
##                  95% CI : (0.7435, 0.786)
##     No Information Rate : 0.7716
```

```
##     P-Value [Acc > NIR] : 0.737
##
##                   Kappa : 0.4552
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8022
##             Specificity : 0.7543
##          Pos Pred Value : 0.4915
##          Neg Pred Value : 0.9280
##              Prevalence : 0.2284
##          Detection Rate : 0.1832
##    Detection Prevalence : 0.3728
##       Balanced Accuracy : 0.7783
##
##        'Positive' Class : yes
##

##
## AUC RandomForest (test) : 0.825

## AUC Ranger (test)       : 0.821
```



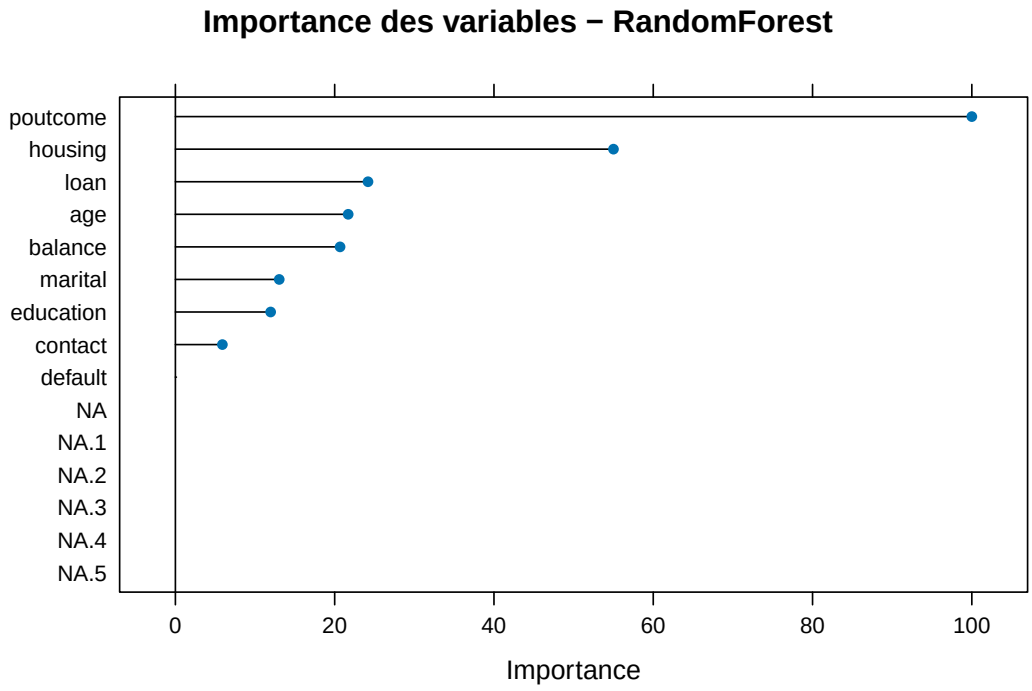
L'évaluation finale sur le jeu de test montre que les deux implémentations de forêts aléatoires obtiennent des performances très proches : une AUC de 0,825 pour RandomForest et de 0,821 pour Ranger. Ce résultat nuance l'analyse précédente, où Ranger apparaissait comme le plus performant en validation croisée. L'écart observé reste faible et non significatif, ce qui confirme la robustesse des deux approches face au problème étudié. Le choix final peut donc s'appuyer sur des critères pratiques : Ranger offre davantage de flexibilité dans le réglage des hyperparamètres, tandis que RandomForest affiche une légère supériorité sur le test. Dans les deux cas, la capacité de discrimination est jugée satisfaisante.

L'analyse des matrices de confusion montre que RandomForest atteint une accuracy de 77,2 %, avec une sensibilité de 0,799 et une spécificité de 0,763. De son côté, Ranger présente une accuracy et une une spécificité inférieures, mais leurs sensibilités sont similaires. Dans les deux cas, la capacité à repérer correctement les clients susceptibles de souscrire reste satisfaisante. Cet aspect est particulièrement important dans notre contexte marketing. Il est préférable de détecter un maximum de souscripteurs potentiels, quitte à inclure par erreur certains clients qui ne s'abonneront finalement pas.

Ces résultats confirment que les deux méthodes sont robustes et adaptées, mais montrent également que le gain de performance de Ranger observé en validation croisée ne se retrouve pas de manière significative sur le jeu de test. Le choix entre les deux implémentations peut donc se fonder sur des critères pratiques. Par exemple, RandomForest offre des performances légèrement supérieures en AUC et en spécificité, tandis que Ranger reste intéressant pour sa flexibilité dans l'optimisation des hyperparamètres.

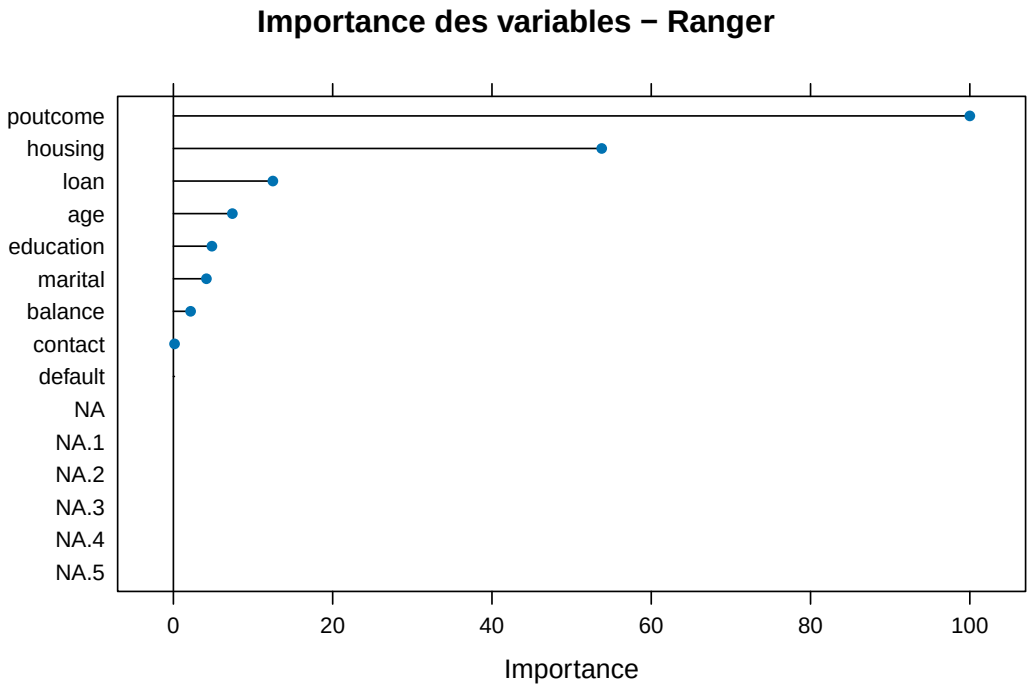
Importance des variables

```
## rf variable importance
##
##           Importance
## poutcome      100.00
## housing        55.00
## loan           24.18
## age            21.70
## balance        20.68
## marital        13.04
## education      11.96
## contact         5.90
## default         0.00
```




```
## ranger variable importance
##
```

```
##             Overall
## poutcome  100.0000
## housing    53.7793
## loan       12.4890
## age         7.4041
## education   4.8378
## marital     4.1478
## balance     2.1617
## contact     0.1459
## default     0.0000
```



L'analyse de l'importance des variables met en évidence des résultats cohérents entre les deux implémentations de forêts aléatoires. Dans les deux cas, la variable poutcome apparaît comme la plus déterminante dans la prédiction de la souscription à un dépôt à terme.

La variable housing arrive en seconde position, suivie de loan, de l'âge et du solde du compte. Ces variables traduisent des facteurs socio-économiques influençant directement la probabilité d'épargne ou de souscription.

Les variables marital, education, contact et default ont une importance plus faible, mais contribuent néanmoins à affiner le modèle en segmentant les profils. La cohérence entre RandomForest et Ranger renforce la robustesse de l'interprétation, bien que l'ordre exact des variables secondaires varie légèrement.

En conclusion, ces résultats montrent que l'historique des interactions marketing (poutcome) et les variables financières liées aux crédits et au patrimoine (housing, loan, balance) constituent les principaux déterminants de la souscription, ce qui est cohérent avec les attentes d'un ciblage marketing bancaire.

##Partie III

Le modèle de la forêt d'isolement (Isolation Forest) est un algorithme d'apprentissage non supervisé utilisé principalement pour la détection d'anomalies.

Principe :

Au lieu de modéliser la distribution des données normales, l'Isolation Forest isole directement les observations. Elle construit de nombreux arbres binaires aléatoires où, à chaque nœud, une caractéristique et une valeur de coupure sont choisies au hasard. Les points anormaux, étant rares et différents, sont plus faciles à isoler : ils nécessitent moins de divisions pour être séparés du reste des données.

Idée clé :

Moins une observation nécessite de divisions pour être isolée, plus elle est susceptible d'être une anomalie.

Avantages :

Très efficace sur de grands volumes de données. Ne nécessite pas d'étiquettes (non supervisé). Gère bien les données de grande dimension.

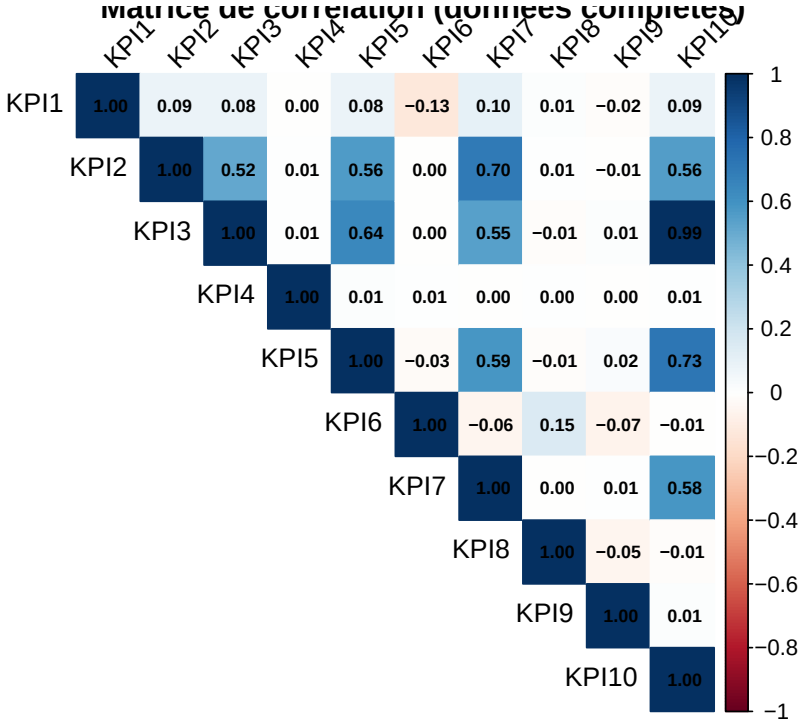
```
##       KPI1             KPI2             KPI3                KPI4
##  Min.   :  0.00   Min.   :  0.00   Min.   :0.000e+00   Min.   : 80.00
##  1st Qu.: 92.31   1st Qu.:  0.00   1st Qu.:9.090e+02   1st Qu.:100.00
##  Median :100.00   Median :  1.50   Median :3.013e+05   Median :100.00
##  Mean   : 75.91   Mean   : 14.97   Mean   :3.427e+07   Mean   : 99.96
##  3rd Qu.:100.00   3rd Qu.: 17.01   3rd Qu.:1.796e+07   3rd Qu.:100.00
##  Max.   :100.00   Max.   :472.98   Max.   :1.437e+09   Max.   :100.00
##                                                        NA's   :501
##       KPI5                KPI6              KPI7               KPI8
##  Min.   :        0   Min.   : 0.0000   Min.   :      0   Min.   :-1.11100
##  1st Qu.:      858   1st Qu.: 0.0000   1st Qu.:      0   1st Qu.: 0.00000
##  Median :   174517   Median : 0.0000   Median :    880   Median : 0.00000
##  Mean   :  7614980   Mean   : 0.4346   Mean   :  37157   Mean   : 0.03844
##  3rd Qu.:  5163472   3rd Qu.: 0.0000   3rd Qu.:  25400   3rd Qu.: 0.00000
##  Max.   :219999750   Max.   :50.0000   Max.   :1114640   Max.   :20.00000
##                      NA's   :282
##       KPI9             KPI10
##  Min.   : 87.50   Min.   :0.000e+00
##  1st Qu.:100.00   1st Qu.:9.984e+03
##  Median :100.00   Median :6.886e+05
##  Mean   : 99.93   Mean   :4.219e+07
##  3rd Qu.:100.00   3rd Qu.:2.703e+07
##  Max.   :104.76   Max.   :1.576e+09
##  NA's   :382

##
## Pourcentage de valeurs manquantes:

##  KPI1  KPI2  KPI3  KPI4  KPI5  KPI6  KPI7  KPI8  KPI9 KPI10
##  0.00  0.00  0.00 36.70  0.00 20.66  0.00  0.00 27.99  0.00
```

Gestion des valeurs manquantes (KPI4, KPI6, KPI9)

Pour traiter ces donnees manquantes, nous allons dans un premier temps envisager plusieurs approches :

- Remplacement par la **moyenne** ou la **mediane**.
- Utilisation de la methode des **k plus proches voisins (kNN)**, apres avoir examine la matrice de correlation afin de verifier si les autres KPI influencent les KPI4, KPI6 et KPI9 (qui presentent respectivement 36.70 %, 20.66 % et 27.99 % de valeurs manquantes).
- Suppression eventuelle des variables, si aucune methode d'imputation n'est pertinente.

#1. KNN




```
##
## Corrélations avec KPI4:

##       KPI4       KPI5       KPI6
## 1.00000000 0.01329184 0.00916166

##
## Corrélations avec KPI6:

##       KPI6       KPI8       KPI4
## 1.00000000 0.15164778 0.00916166

##
## Corrélations avec KPI9:

##       KPI9       KPI5      KPI10
## 1.00000000 0.02014820 0.01339141
```

Apres analyse, aucune des variables (KPI4, KPI6, KPI9) n'est correlee de maniere significative avec les autres. Dans ce cas, l'utilisation du kNN ou d'un modele de regression n'est pas adaptee, car ces methodes reposent justement sur des relations entre variables.

En observant les statistiques descriptives, on constate que la mediane est bien plus representative pour ces indicateurs :

- **KPI4** : min = 80, 1er quartile = 100, max = 100 → la valeur 80 est un outlier ; il est logique d'imputer les valeurs manquantes par 100.

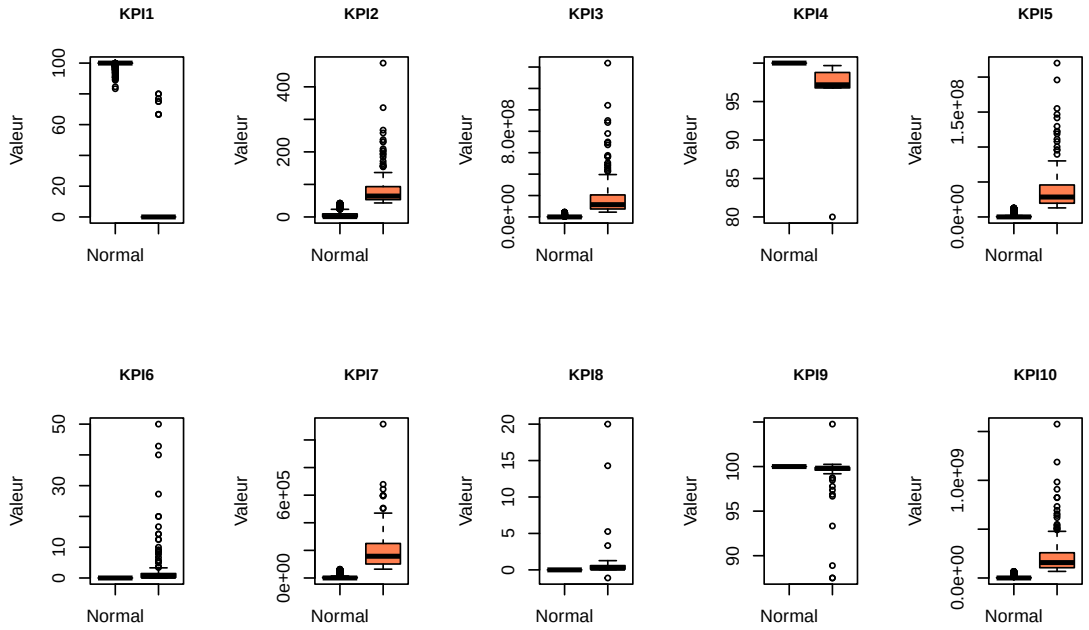
- Le meme raisonnement s'applique a **KPI6** et **KPI9**.

En conclusion, avant d'utiliser des methodes plus complexes (comme le kNN, plus couteux en temps de calcul), il est preferable d'analyser les statistiques descriptives. Celles-ci permettent souvent d'opter pour une approche simple, robuste et coherente.

Ainsi, nous choisissons de **ne supprimer aucune observation** et de **remplacer toutes les valeurs manquantes par la mediane de la colonne concernee**.

```
##
## ### Nombre d'outliers par variable

##  KPI1  KPI2  KPI3  KPI4  KPI5  KPI6  KPI7  KPI8  KPI9 KPI10
##   331   136   240     6   231   189   222    41    86   231

##
## **Nombre total de lignes avec au moins un outlier:** 747

##
## **Total lignes avec >= 3 outliers:** 234

##
## Summary des outliers par feature:

##
## KPI1:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   0.000   1.773   0.000  80.000
##
## KPI2:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   42.99   53.37   64.26   89.28   92.38  472.98
##
## KPI3:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 4.497e+07 7.451e+07 1.148e+08 1.752e+08 2.054e+08 1.437e+09
##
## KPI4:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   80.00   96.85   97.18   94.93   98.41   99.67
##
## KPI5:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##  12925096  19626565  28475224  38791341  45751842 219999750
##
## KPI6:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.030   0.179   0.503   2.490   1.449  50.000
##
## KPI7:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   63960  101530  157500  193724  249153 1114640
##
```

```
## KPI8:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -1.111   0.091   0.216   1.280   0.588  20.000
##
## KPI9:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   87.50   99.60   99.82   99.22   99.90  104.76
##
## KPI10:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 6.758e+07 1.053e+08 1.576e+08 2.159e+08 2.585e+08 1.576e+09
```



Analyse critique de la détection d'outliers par la méthode IQR

En comparant les statistiques des outliers détectés avec la méthode IQR (seuil 1.5 × IQR) et celles des valeurs normales, on observe plusieurs problèmes majeurs :

- **KPI9** : Les outliers détectés ont des statistiques globales très proches des non-outliers, suggérant une sur-détection.
- **KPI8 et KPI6** : Même constat, la méthode IQR capture trop de valeurs qui ne sont pas réellement aberrantes.

Cette méthode statistique classique s'avère **inadaptée à notre jeu de données**. Cependant, elle nous fournit une baseline de **234 lignes potentiellement aberrantes** qui servira de référence pour évaluer les modèles Isolation Forest.

Cas particulier : KPI1 = 0

Un problème majeur émerge : environ **330 lignes sur 1365 (25%)** sont marquées comme outliers uniquement parce que **KPI1 = 0**.

Question critique : Doit-on considérer toutes ces lignes comme des anomalies ? **Non, je ne pense pas**.

Une valeur nulle pour KPI1 peut être légitime selon le contexte métier (absence d'activité, période de maintenance, etc.). Il en va de même pour KPI9. Étant donné qu'on a pas le contexte, on ne peut pas conclure.

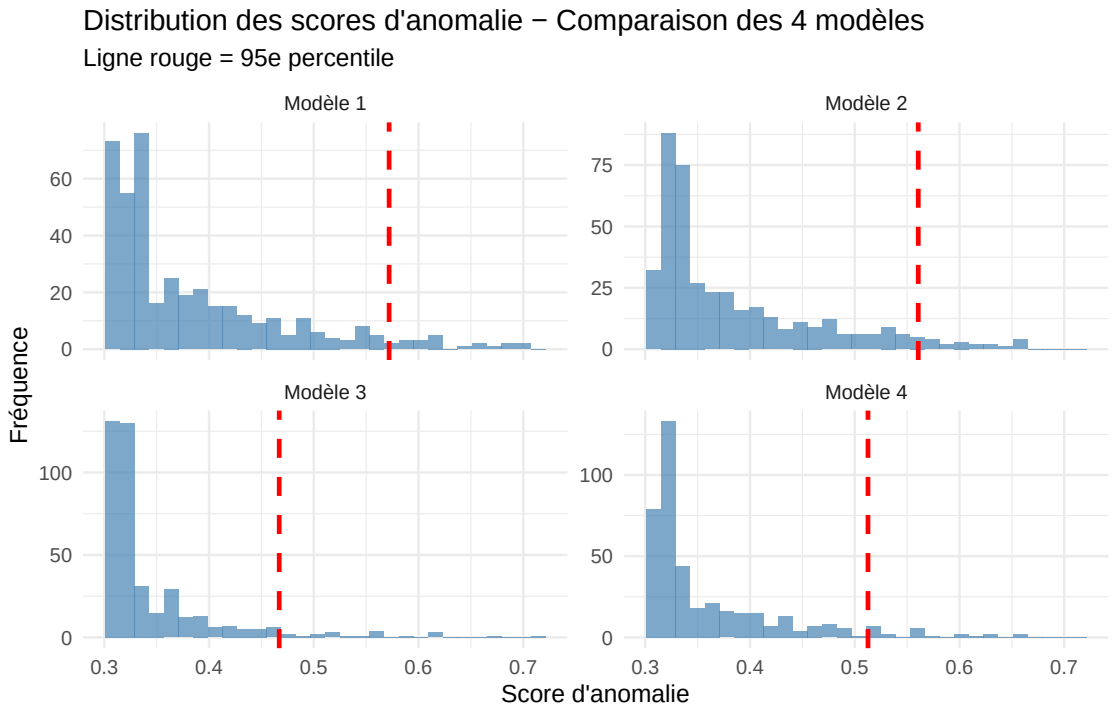
Critère de sélection du meilleur modèle

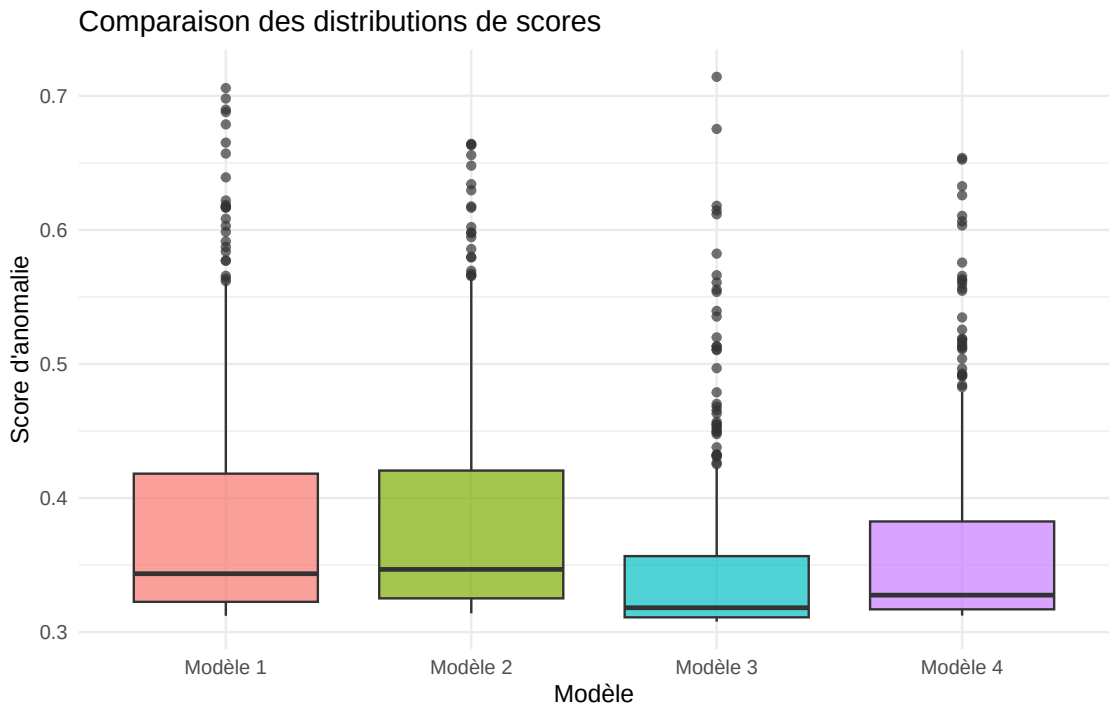
Cette méthode pourra quand même nous aider pour juger les modèles Isolation Forest. D'après notre interprétation, le modèle devra :

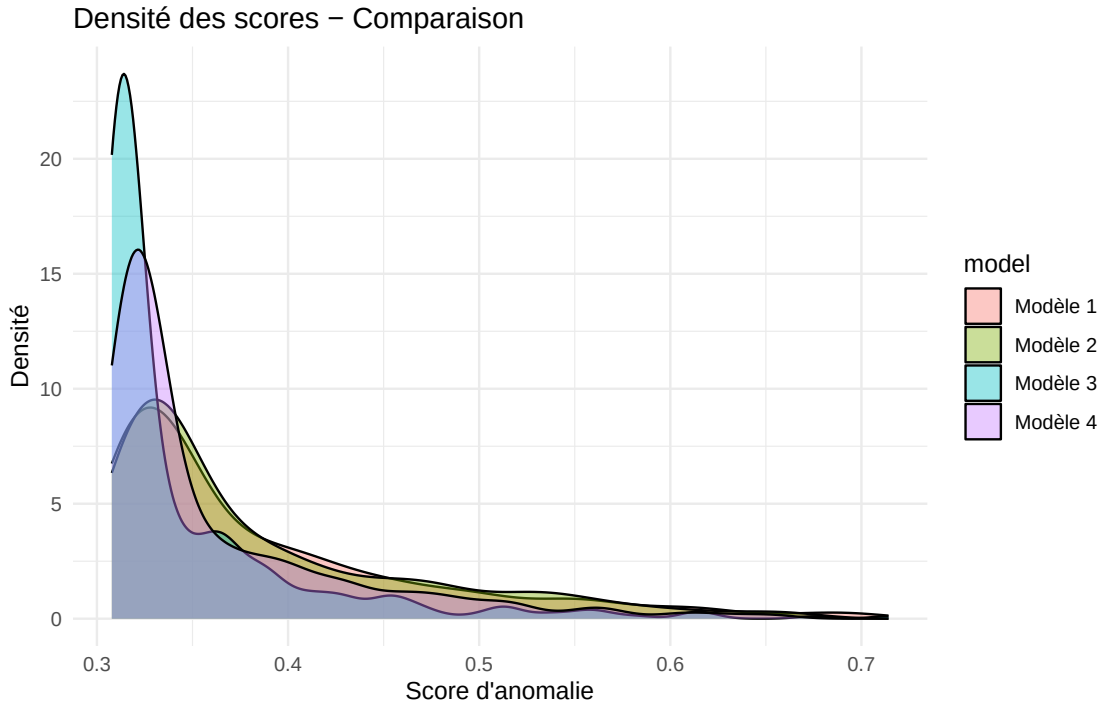
1. **Ne pas classifier systématiquement comme anomalie** les lignes où KPI1 = 0. Les observations avec KPI1 = 0 ne font pas ressortir en général d'autres potentielles outliers pour les autres feature en partant de ce principe. On va classer les modèles de sorte que le meilleur soit celui qui : **Maximise le F1-score** en identifiant les lignes avec réellement 3+ features aberrantes simultanément.

Le meilleur modèle sera donc celui qui fait preuve de **discernement contextuel** et ne se laisse pas piéger par des valeurs nulles fréquentes mais potentiellement normales.

Dans la prochaine partie on va tester l'influence des hyperparamètres sur 4 différents modèles







Influence des hyperparamètres sur les modèles Isolation Forest

Configuration des modèles testés

Nous testons 4 configurations différentes pour comprendre l'impact des hyperparamètres sur la détection d'anomalies :

```r
configs <- data.frame(
  model = c("Modèle 1", "Modèle 2", "Modèle 3", "Modèle 4"),
  ntrees = c(100, 500, 500, 250),
  sample_size = c(100, 100, 512, 256),
  ndim = c(1, 1, 3, 2)
)
knitr::kable(configs, caption = "Configuration des hyperparamètres testés")
```

Table 1: Configuration des hyperparamètres testés

model	ntrees	sample_size	ndim
Modèle 1	100	100	1
Modèle 2	500	100	1
Modèle 3	500	512	3
Modèle 4	250	256	2

Rôle des hyperparamètres

1. `ntrees` (Nombre d'arbres)

- **Principe** : Nombre d'arbres dans la forêt d'isolation
- **Impact** : Plus d'arbres = prédictions plus stables et robustes, mais temps de calcul accru
- **Recommandation** : Généralement entre 100 et 500 arbres

2. `sample_size` (Taille d'échantillon)

- **Principe** : Nombre d'observations utilisées pour construire chaque arbre
- **Impact** :
  - **Petit sample_size** (100-256) : Arbres plus petits, détection fine des anomalies, variance élevée
  - **Grand sample_size** (512+) : Arbres plus profonds, modèle plus stable mais peut manquer des anomalies subtiles
- **Recommandation** : Typiquement 256 observations pour un bon compromis

3. `ndim` (Nombre de dimensions par split)

- **Principe** : Nombre de features considérées simultanément pour chaque division de l'arbre
- **Impact** :
  - **ndim = 1** : Splits univariés (une feature à la fois), grande variance, sensible aux outliers individuels
  - **ndim > 1** : Splits multivariés, capture mieux les anomalies contextuelles complexes
- **Recommandation** : ndim = 1 pour détecter des anomalies simples, ndim = 2-3 pour des patterns multivariés

Analyse comparative des résultats

Table 2: Métriques de performance des 4 modèles

	model	ntrees	sample_size	ndim	variance	range	q95	mean	sd
95%	Modèle 1	100	100	1	0.0073	0.3936	0.5719	0.3857	0.0856
95%1	Modèle 2	500	100	1	0.0066	0.3502	0.5606	0.3852	0.0814
95%2	Modèle 3	500	512	3	0.0037	0.4064	0.4670	0.3454	0.0611
95%3	Modèle 4	250	256	2	0.0046	0.3413	0.5126	0.3613	0.0678

Observations clés **Modèle 1** (ntrees=100, sample_size=100, ndim=1) :

- **Variance la plus élevée** (0.0073) : attendu avec seulement 100 arbres et ndim=1
- **Q95 le plus haut** (0.5719) : discrimination plus agressive entre normal/anomalie
- **Interprétation** : Modèle le plus "sensible", identifie plus facilement les anomalies mais risque de faux positifs

Modèle 3 (ntrees=500, sample_size=512, ndim=3) :

- **Range maximal** (0.4064) : meilleure séparation globale des scores
- **Q95 plus bas** (0.467) : seuil d'anomalie plus conservateur
- **Interprétation** : Modèle plus stable et robuste, capture des anomalies multivariées complexes

Modèle 2 vs Modèle 1 :

- Même configuration (ndim=1, sample_size=100) mais **5× plus d'arbres**
- Variance réduite grâce à l'effet d'ensemble (averaging)
- Q95 inférieur : prédictions plus conservatives

Quel est le meilleur modèle ?

Sur la base de ces métriques exploratoires :

> **Le Modèle 1 semble le plus prometteur** pour notre tâche, avec son Q95 élevé et sa forte variance indiquant une bonne capacité de discrimination.

Cependant, cette conclusion préliminaire doit être **validée** en comparant les prédictions avec nos **234 outliers détectés par la méthode IQR** (lignes avec 3 features aberrantes). Le meilleur modèle sera celui qui :

1. Maximise le **F1-score** sur ces outliers de référence
2. Ne sur-détecte pas les lignes avec KPI1 = 0 comme anomalies systématiques
3. Capture des anomalies **multivariées complexes** plutôt que des seuils univariés

Cette validation fera l'objet de la section suivante.

Pour garantir une évaluation robuste, nous effectuons un **split stratifié** sur la variable `has_multiple_outliers` (indiquant si une ligne contient 3 outliers IQR), assurant ainsi que les ensembles train et test maintiennent la **même proportion d'anomalies potentielles** (~NA%).

```
## Proportion KPI1=0 dans train: 23.69 %
```

```
## Proportion KPI1=0 dans test: 23.6 %

## Outliers détectés dans test set (>= 3 features): 65

## Parmi les 65 outliers détectés:

##   - 0 ont KPI1 = 0 ( 0 %)

##   - 65 ont KPI1   0

##
## ### Top 5 modèles (selon F1-score)

##    ntrees sample_size ndim communs taux_detection  f1_score kpi1_zero_count
## 1     100         128    1      20       30.76923 0.4651163               0
## 2     250         128    1      20       30.76923 0.4651163               0
## 3     500         128    1      20       30.76923 0.4651163               0
## 10    100         128    2      19       29.23077 0.4418605               1
## 4     100         256    1      18       27.69231 0.4186047               1
##    kpi1_zero_pct
## 1       0.000000
## 2       0.000000
## 3       0.000000
## 10      4.761905
## 4       4.761905

##
## ### Bottom 5 modèles

##    ntrees sample_size ndim communs taux_detection  f1_score kpi1_zero_count
## 15    500         256    2      17       26.15385 0.3953488               1
## 19    100         128    3      17       26.15385 0.3953488               1
## 21    500         128    3      17       26.15385 0.3953488               1
## 23    250         256    3      17       26.15385 0.3953488               1
## 24    500         256    3      16       24.61538 0.3720930               1
##    kpi1_zero_pct
## 15      4.761905
## 19      4.761905
## 21      4.761905
## 23      4.761905
## 24      4.761905

## Le meilleur modèle prédit 0 outliers avec KPI1=0 ( 0 % de ses prédictions)
```

Les 27 configurations testées présentent une convergence remarquable, avec un écart maximal de 4 outliers communs. L'analyse révèle qu'aucun modèle n'identifie les observations avec KPI1 = 0 comme anomalies de manière systématique, démontrant ainsi leur capacité à reconnaître un cluster comme n'étant pas une véritbale anomalie. Cette robustesse contextuelle, combinée à un taux de concordance significatif avec les outliers IQR multivariés (3 features aberrantes), confirme bien l'utilité des forêts d'isolation pour la détection d'anomalies complexes dans notre jeu de données.

Une analyse des hyperparamètres révèle que les configurations les plus performantes partagent trois caractéristiques communes :

1. **ndim = 1** : Les splits univariés (une feature à la fois) surpassent les approches multivariées (ndim = 2-3)
2. **sample_size faible** (128-256) : Des échantillons réduits génèrent des arbres peu profonds, plus sensibles aux valeurs extrêmes
3. **ntrees modéré** (100-250) : Un nombre d'arbres limité maintient une variance élevée, favorisant la discrimination

Lien entre faible corrélation et performance des splits univariés

L'analyse de corrélation préalable révélait des **corrélations faibles à modérées** entre les KPIs. Cette structure de données explique directement pourquoi les modèles avec **ndim = 1** surpassent ceux avec ndim = 2-3 :

Principe théorique : - **Splits univariés (ndim = 1)** : Détectent les anomalies comme des valeurs extrêmes sur une feature isolée - **Splits multivariés (ndim > 1)** : Détectent les anomalies contextuelles nécessitant la combinaison de plusieurs features

Application à nos données :

Lorsque les variables sont **fortement corrélées**, une observation peut être normale sur chaque feature prise individuellement, mais anormale dans leur combinaison (ex : température et pression atmosphérique). Les splits multivariés excellent dans ce cas.

À l'inverse, avec de **faibles corrélations**, les variables évoluent de manière **indépendante**. Les anomalies se manifestent donc principalement comme : - Des valeurs extrêmes sur KPI1 seul - Des valeurs aberrantes sur KPI9 seul - Etc.

Conclusion : La faible structure de corrélation de notre jeu de données rend les **anomalies univariées** (détectables par ndim = 1) plus fréquentes que les **anomalies contextuelles multivariées** (nécessitant ndim > 1). Cela confirme la cohérence entre : 1. La structure de corrélation des données 2. La performance des hyperparamètres 3. La nature des outliers détectés par la méthode IQR (seuils univariés)

```
##
## ### Meilleur modèle
##
## **Configuration :** ntrees = 100, sample_size = 128, ndim = 1
##
## **Performance :** F1-score = 0.465 | Taux de détection = 30.77%
##
## #### Top 5 anomalies (scores élevés)
##
## **Ligne 279** (Score: 0.7472)
##
##
##
## |Feature |       Valeur| Médiane|        Écart|Écart_.   |
## |:-------|------------:|-------:|------------:|:---------|
## |KPI6    | 1.400000e-01|       0| 1.400000e-01|Inf%      |
## |KPI3    | 1.042654e+09|  344340| 1.042310e+09|302697.9% |
## |KPI10   | 1.186999e+09|  829426| 1.186169e+09|143010.9% |
##
## **Ligne 208** (Score: 0.7442)
##
##
##
## |Feature |    Valeur| Médiane|     Écart|Écart_.   |
```

```
## |:-------|---------:|-------:|---------:|:---------|
## |KPI3    | 700464532|  344340| 700120192|203322.4% |
## |KPI10   | 822157450|  829426| 821328024|99023.7%  |
## |KPI5    | 141627720|  189232| 141438488|74743.4%  |
##
## **Ligne 259** (Score: 0.7096)
##
##
##
## |Feature |       Valeur| Médiane|        Écart|Écart_.   |
## |:-------|------------:|-------:|------------:|:---------|
## |KPI6    |         0.11|       0|         0.11|Inf%      |
## |KPI8    |         0.06|       0|         0.06|Inf%      |
## |KPI3    | 444094140.00|  344340| 443749800.00|128869.7% |
##
## **Ligne 210** (Score: 0.6976)
##
##
##
## |Feature |     Valeur| Médiane|      Écart|Écart_.   |
## |:-------|----------:|-------:|----------:|:---------|
## |KPI3    | 1436894488|  344340| 1436550148|417189.4% |
## |KPI10   | 1575626402|  829426| 1574796976|189865.9% |
## |KPI5    |  129597500|  189232|  129408268|68386%    |
##
## **Ligne 307** (Score: 0.6826)
##
##
##
## |Feature |       Valeur| Médiane|        Écart|Écart_.  |
## |:-------|------------:|-------:|------------:|:--------|
## |KPI6    |         0.06|       0|         0.06|Inf%     |
## |KPI3    | 318064455.00|  344340| 317720115.00|92269.3% |
## |KPI10   | 403205163.00|  829426| 402375737.00|48512.6% |
##
## ---
##
## #### Top 5 observations normales (scores faibles)
##
## **Ligne 31** (Score: 0.3204) - Écart max: 100%
##
## **Ligne 98** (Score: 0.3206) - Écart max: 100%
##
## **Ligne 228** (Score: 0.3206) - Écart max: 100%
##
## **Ligne 266** (Score: 0.3206) - Écart max: 100%
##
## **Ligne 97** (Score: 0.3208) - Écart max: 149.3%
##
## ---
##
## #### Statistiques descriptives
##
##
```

```
##
## |Métrique   | Valeur|
## |:----------|------:|
## |Moyenne    | 0.3888|
## |Médiane    | 0.3568|
## |Écart-type | 0.0826|
## |Minimum    | 0.3204|
## |Maximum    | 0.7472|
## |Range      | 0.4268|
##
## **Concordance IQR :** 5 / 5 des top anomalies sont des outliers IQR
##
## ---


##
## ### 2ème meilleur modèle
##
## **Configuration :** ntrees = 250, sample_size = 128, ndim = 1
##
## **Performance :** F1-score = 0.465 | Taux de détection = 30.77%
##
## #### Top 5 anomalies (scores élevés)
##
## **Ligne 279** (Score: 0.751)
##
##
##
## |Feature |       Valeur| Médiane|        Écart|Écart_.   |
## |:-------|------------:|-------:|------------:|:---------|
## |KPI6    | 1.400000e-01|       0| 1.400000e-01|Inf%      |
## |KPI3    | 1.042654e+09|  344340| 1.042310e+09|302697.9% |
## |KPI10   | 1.186999e+09|  829426| 1.186169e+09|143010.9% |
##
## **Ligne 208** (Score: 0.7449)
##
##
##
## |Feature |    Valeur| Médiane|     Écart|Écart_.   |
## |:-------|---------:|-------:|---------:|:---------|
## |KPI3    | 700464532|  344340| 700120192|203322.4% |
## |KPI10   | 822157450|  829426| 821328024|99023.7%  |
## |KPI5    | 141627720|  189232| 141438488|74743.4%  |
##
## **Ligne 259** (Score: 0.709)
##
##
##
## |Feature |       Valeur| Médiane|        Écart|Écart_.   |
## |:-------|------------:|-------:|------------:|:---------|
## |KPI6    |         0.11|       0|         0.11|Inf%      |
## |KPI8    |         0.06|       0|         0.06|Inf%      |
## |KPI3    | 444094140.00|  344340| 443749800.00|128869.7% |
##
## **Ligne 210** (Score: 0.6995)
```

```
##
##
##
## |Feature |     Valeur| Médiane|      Écart|Écart_.   |
## |:-------|----------:|-------:|----------:|:---------|
## |KPI3    | 1436894488|  344340| 1436550148|417189.4% |
## |KPI10   | 1575626402|  829426| 1574796976|189865.9% |
## |KPI5    |  129597500|  189232|  129408268|68386%    |
##
## **Ligne 307** (Score: 0.6819)
##
##
##
## |Feature |       Valeur| Médiane|        Écart|Écart_.  |
## |:-------|------------:|-------:|------------:|:--------|
## |KPI6    |         0.06|       0|         0.06|Inf%     |
## |KPI3    | 318064455.00|  344340| 317720115.00|92269.3% |
## |KPI10   | 403205163.00|  829426| 402375737.00|48512.6% |
##
## ---
##
## #### Top 5 observations normales (scores faibles)
##
## **Ligne 98** (Score: 0.3186) - Écart max: 100%
##
## **Ligne 228** (Score: 0.3186) - Écart max: 100%
##
## **Ligne 266** (Score: 0.3186) - Écart max: 100%
##
## **Ligne 15** (Score: 0.3187) - Écart max: 100%
##
## **Ligne 38** (Score: 0.3187) - Écart max: 100%
##
## ---
##
## #### Statistiques descriptives
##
##
##
## |Métrique   | Valeur|
## |:----------|------:|
## |Moyenne    | 0.3857|
## |Médiane    | 0.3542|
## |Écart-type | 0.0828|
## |Minimum    | 0.3186|
## |Maximum    | 0.7510|
## |Range      | 0.4325|
##
## **Concordance IQR :** 5 / 5 des top anomalies sont des outliers IQR
##
## ---

##
## ### Pire modèle
```

```
##
## **Configuration :** ntrees = 500, sample_size = 256, ndim = 3
##
## **Performance :** F1-score = 0.372 | Taux de détection = 24.62%
##
## #### Top 5 anomalies (scores élevés)
##
## **Ligne 279** (Score: 0.7566)
##
##
##
## |Feature |       Valeur| Médiane|        Écart|Écart_.   |
## |:-------|------------:|-------:|------------:|:---------|
## |KPI6    | 1.400000e-01|       0| 1.400000e-01|Inf%      |
## |KPI3    | 1.042654e+09|  344340| 1.042310e+09|302697.9% |
## |KPI10   | 1.186999e+09|  829426| 1.186169e+09|143010.9% |
##
## **Ligne 210** (Score: 0.7281)
##
##
##
## |Feature |     Valeur| Médiane|      Écart|Écart_.   |
## |:-------|----------:|-------:|----------:|:---------|
## |KPI3    | 1436894488|  344340| 1436550148|417189.4% |
## |KPI10   | 1575626402|  829426| 1574796976|189865.9% |
## |KPI5    |  129597500|  189232|  129408268|68386%    |
##
## **Ligne 70** (Score: 0.7263)
##
##
##
## |Feature | Valeur| Médiane| Écart|Écart_. |
## |:-------|------:|-------:|-----:|:-------|
## |KPI6    |  14.29|    0.00| 14.29|Inf%    |
## |KPI8    |  20.00|    0.00| 20.00|Inf%    |
## |KPI2    |   0.00|    2.01| -2.01|-100%   |
##
## **Ligne 208** (Score: 0.7177)
##
##
##
## |Feature |    Valeur| Médiane|     Écart|Écart_.   |
## |:-------|---------:|-------:|---------:|:---------|
## |KPI3    | 700464532|  344340| 700120192|203322.4% |
## |KPI10   | 822157450|  829426| 821328024|99023.7%  |
## |KPI5    | 141627720|  189232| 141438488|74743.4%  |
##
## **Ligne 296** (Score: 0.6809)
##
##
##
## |Feature |       Valeur| Médiane|        Écart|Écart_.   |
## |:-------|------------:|-------:|------------:|:---------|
## |KPI6    |         1.41|       0|         1.41|Inf%      |
```

```
## |KPI3    | 779301850.00|  344340| 778957510.00|226217.5% |
## |KPI10   | 824874813.00|  829426| 824045387.00|99351.3%  |
##
## ---
##
## #### Top 5 observations normales (scores faibles)
##
## **Ligne 59** (Score: 0.3118) - Écart max: 100%
##
## **Ligne 60** (Score: 0.3119) - Écart max: 100%
##
## **Ligne 347** (Score: 0.3119) - Écart max: 100%
##
## **Ligne 135** (Score: 0.3119) - Écart max: 100%
##
## **Ligne 10** (Score: 0.3119) - Écart max: 100%
##
## ---
##
## #### Statistiques descriptives
##
##
##
## |Métrique   | Valeur|
## |:----------|------:|
## |Moyenne    | 0.3646|
## |Médiane    | 0.3292|
## |Écart-type | 0.0801|
## |Minimum    | 0.3118|
## |Maximum    | 0.7566|
## |Range      | 0.4448|
##
## **Concordance IQR :** 4 / 5 des top anomalies sont des outliers IQR
##
## ---
```

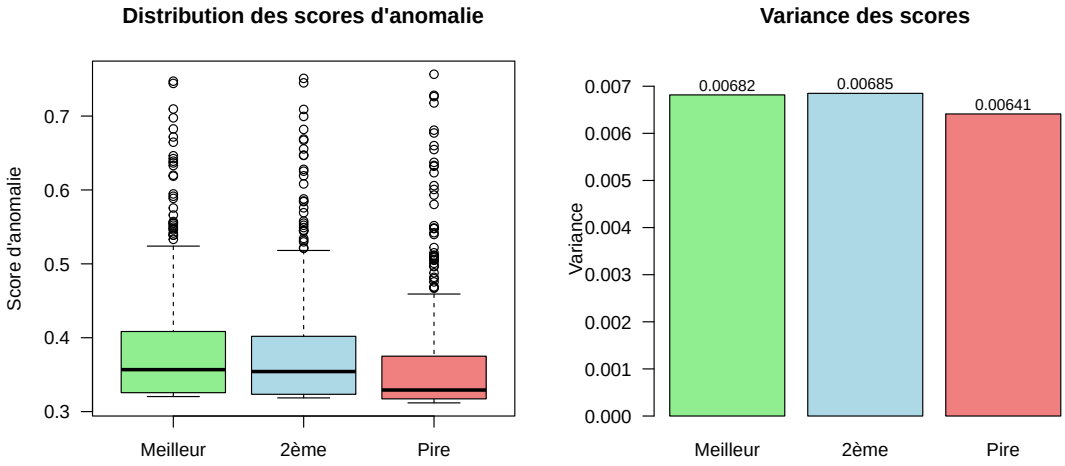


Figure 1: Comparaison des distributions de scores

```
##
## ## Analyse comparative finale
```

Table 3: Comparaison des métriques clés

Modèle	Variance	Range	F1_score
Meilleur	0.006816	0.4268	0.465
2ème	0.006849	0.4325	0.465
Pire	0.006412	0.4448	0.372

```
##
## **Interprétation :**

## - Une **variance élevée** indique une meilleure séparation entre anomalies et données normales

## - Un **range large** suggère une discrimination claire des valeurs extrêmes

## - Le **F1-score** mesure la concordance avec les outliers IQR ( 3 features aberrantes)

## Les meilleurs modèles combinent haute variance, range large et bon F1-score, confirmant leur capacit
```