SY19 - Rapport de projet

Engel CALON, Bastien CUVILLIER, Camille MILON

10 Décembre 2025

1 Introduction

Ce document présente le raisonnement et les méthodes appliquées, ainsi que les résultats obtenus dans le cadre du projet de SY19. Ce rapport se découpe en trois parties, une pour chaque jeu de données étudié : TP5_reg, TP5_clas et Heart Failure Prediction.

2 Regression : TP5_reg

Ce jeu de données, destiné à de la regression, se compose de 500 observations, de 100 variables descriptives numériques (X1, X2,..., X100) et d'une variable cible continue à valeurs réelles. L'objectif, dans la suite de cette partie, sera de prédire la variable y à partir d'observations des variables descriptives (X1 à X100).

2.1 Analyse Descriptive

Afin de proposer un modèle de prédiction correct, il est important de connaître les caracteristiques des variables étudiées, ainsi que les relations entre celles-ci mais aussi avec la variable cible. C'est pour cette raison que nous avons réalisé une analyse descriptive.

Dans un premier temps, nous nous sommes intéressés à la structure des variables en étudiant leurs distributions univariées. Comme illustré sur la figure 1, les 100 variables ont des formes de distribution extrêmement similaires : les médianes se situent toutes autour de 5, les étendues interquartiles sont presque identiques et la quasi-totalité des observations se trouve dans un intervalle compris entre 0 et 10. Cette homogénéité des distributions nous a permis de former l'hypothèse que les transformations simples (log, racine, carré) auraient probablement un impact limité sur la qualité des modèles, puisqu'aucune variable ne présente de distribution fortement asymétrique ou étendue.

Par ailleurs, aucune variable ne présente de valeurs extrêmes susceptibles de perturber l'ajustement ou de nécessiter un traitement particulier. L'absence d'outliers massifs en plus des distributions quasi-identiques renforce l'idée que les transformations globales appliquées à toutes les variables auront probablement peu d'impact. Cela sera vérifié par la suite.
L'absence de valeurs manquantes sur l'ensemble du jeu de données confirme également que les modèles pourront être entraînés sans amputation supplémentaire.

D'après la matrice de corrélation (Figure 2), les coefficients de corrélation entre les différentes variables explicatives sont globalement très proches de zéro. Il n'existe ni blocs fortement corrélés, ni redondance manifeste entre les variables. Cette absence de structure corrélée réduit l'intérêt potentiel de méthodes basées sur la réduction de dimension telles que l'Analyse en Composantes Principales (PCA) ou la régression PCR, qui s'appuient précisément sur l'existence de directions fortes dans l'espace des données. De même, elle suggère que la régression Ridge, conçue pour traiter la multicolinéarité, n'apporterait pas d'avantage significatif par rapport à LASSO ou même une régréssion linéaire simple. Cela sera vérifié par la suite.

Enfin, l'exploration des relations univariées entre y et chaque X_j n'a pas révélé de variable possédant un signal linéaire fort et isolé capable d'expliquer une large fraction de la variance de y. Autrement dit, le signal prédictif disponible par variable est relativement faible et réparti sur plusieurs variables. Cela oriente naturellement vers des méthodes qui contrôlent la variance (régularisation) plutôt que des méthodes non régularisées.

Au vu des résultats de l'analyse descriptive, nous avons postulé que les modèles linéaires régularisés (LASSO ou Elastic Net) seraient plus adaptés que les méthodes non linéaires (arbres, forêts, réseaux, etc), dû à l'absence de structures complexes (interactions fortes, non-linéarités marquées) dans les données.

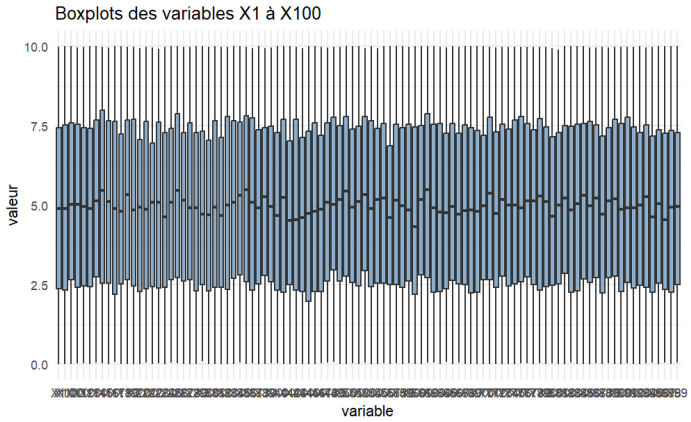

Figure 1: Boxplots des variables explicatives. X_j

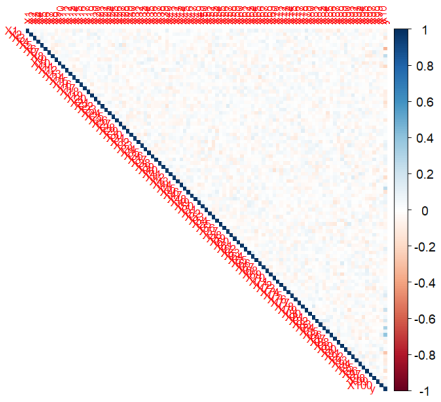

Figure 2: Matrice de corrélation.

2.2 Sélection de variables

Nous avons sélectionné les variables les plus pertinentes pour nos modèles de régression en appliquant les méthodes itératives (stepwise) forward et backward, ainsi que les critères AIC et BIC via une recherche bidirectionnelle. AIC et BIC ont tout d'abord été appliqués à un modèle de régression logistique complet (incluant toutes les variables) afin d'identifier les variables les plus pertinentes. Nous avons ensuite comparé les différentes méthodes par validation classique. La formule proposée par la méthode AIC s'est révélée particulièrement efficace pour les modèles de régression linéaire, offrant une grande performance prédictive.

2.3 Transformations

Afin d'améliorer les performances du modèle de régression, nous avons exploré plusieurs transformations classiques des variables explicatives. L'objectif était d'augmenter le pouvoir prédictif en révélant d'éventuelles non-linéarités simples ou en stabilisant la variabilité des prédicteurs.

Transformation quadratique : l'application de notre modèle LASSO sur les variables transformées X_j^2 a produit une MSE significativement plus élevée que n'importe quel modèle testé utilisant les variables non transformées. Cela confirme que les variables X_j ne présentent pas de courbure notable et que la relation avec y est probablement proche d'une linéarité faible.

Transformation logarithmique : la transformation logarithme étant majoritairement utilisée pour atténuer l'effet de queues lourdes ou réduire l'influence des valeurs extrêmes, ce qui n'est pas le cas ici d'après l'analyse descriptive, nous ne nous attendions pas à des résultats particulièrement probants. Cela a été confirmé par les essais : aucune amélioration n'a été observée, avec des MSE 6 fois supérieures au LASSO sur données standardisées brutes.

Transformation racine carrée : la transformation racine carré est utile lorsque les données croissent rapidement ou présentent une dispersion croissante avec leur valeur, ce qui ne semblait pas être le cas d'après l'analyse descriptive et a été confirmé lors des tests. Nous avons obtenu des MSE 5 fois supérieures au LASSO sur données standardisées brutes.

Mélanges de transformations : Nous avons également testé des combinaisons telles que :

- concaténation des données originales et transformées,
- choix sélectif des transformations par validation croisée,
- ajout d'interactions de type pairwise. $X_i X_j$

Cependant, aucun de ces mélanges de transformations n'a créé de structure exploitable par les modèles linéaires pénalisés, ce qui confirme que le signal est faible, diffus et essentiellement linéaire.

Enfin, en transformation appliquée, nous avons normalisé les variables explicatives, étape indispensable pour pouvoir appliquer certains modèles (régressions pénalisées notamment).

2.4 Régresseur

Compte tenu des conclusions de l'analyse descriptive et des nombreux essais réalisés, notre objectif était de sélectionner un modèle robuste face à un signal diffus sur de nombreuses variables, capable de limiter le surapprentissage et apte à effectuer une sélection automatique de variables.

Plusieurs modèles ont été évalués : Régression linéaire simple, Ridge, LASSO, Elastic Net, PCR, PLS, GAM/BAM et Réseau de neurones simple. Conformément à nos attentes (explicitées dans l'analyse descriptive), de très bons résultats ont été obtenus avec LASSO, Elastic Net mais aussi la régression linéaire avec sélection de variable AIC.

L'évaluation de BAM avec sélection LASSO sur la plateforme de test aboutit à une MSE de **158.3115**. Ainsi, le meilleur modèle final combine la sélection automatique du LASSO et la flexibilité contrôlée d'un modèle additif généralisé (estimé via `bam()`). Le LASSO retire les nombreuses variables bruitées, tandis que les splines du GAM capturent de faibles effets non linéaires sans surapprentissage. Ce compromis entre parcimonie et souplesse permet d'obtenir la meilleure MSE parmi tous les modèles testés.

2.5 Résultats et interprétations

D'après les résultats obtenus, voici les conclusions que nous pouvons faire sur les différents modèles testés.
Méthodes de réduction de dimension (PCR, PLS) : Conformément aux conclusions de l'analyse descriptive, ces méthodes ne donnent pas de bons résultats dû à l'absence de blocs corrélés, à l'absence de directions dominantes dans l'espace des X et au fait que le signal est faible et réparti. La réduction de dimension entraîne donc une perte d'information utile.
Méthodes non linéaires : Les réseaux de neurones simples donnent de mauvais résultats car :

- Les données ne contiennent pas de non-linéarités marquées,
- Le signal est faible donc les modèles flexibles surapprennent,
- Les relations entre X et y semblent essentiellement linéaires.

Les arbres et forêts n'ont pas été retenus pour la même raison.

Ridge : Ridge n'apporte pas de gain car il n'y a pas de multicolinéarité.
Régression linéaire avec sélection de variables AIC : Les méthodes de sélection ainsi que les procédures pas à pas (forward, backward) ont permis d'identifier des sous-ensembles de variables pertinentes. Cette approche a permis d'améliorer les performances du modèle linéaire, qui n'a cependant pas excédé le modèle selectionné.

3 Classification : TP5_clas

Ce jeu de données, destiné à de la classification, se compose de 500 observations, de 50 variables explicatives (X1 à X50) et d'une variable cible composée de trois modalités (1, 2 et 3). L'objectif, dans la suite de cette partie, sera de prédire la classe y à partir d'observations des variables descriptives (X1 à X50).

3.1 Analyse descriptive

D'après l'analyse descriptive, 3 types de variables explicatives se distinguent :

La première famille de variables (Figure 3) est composée des variables numériques continues **X1 à X20**, dont les valeurs se situent majoritairement dans l'intervalle $[0, 10]$, avec des moyennes proches de 5. Ces variables présentent des distributions globalement symétriques, bien centrées, sans valeurs extrêmes marquées. Leur comportement homogène et leur caractère apparemment aléatoire suggèrent qu'elles contiennent peu de structure exploitable pour la séparation des classes, hypothèse qui sera vérifiée lors de la phase

de modélisation.

Les variables **X21 à X45** (Figure 4) se distinguent par une plage de valeurs nettement plus large, incluant des valeurs négatives, et par une forte variabilité. Les intervalles interquartiles sont élevés et plusieurs variables (notamment X22, X32 et X35) présentent des valeurs atypiques marquées. Ces caractéristiques traduisent une forte hétérogénéité des données et rendent nécessaire l'application d'une étape de standardisation (centrage-réduction).

Les variables **X46 à X50** (Figure 5) sont de nature discrète entières, avec des valeurs comprises entre 0 et 13. Elles peuvent être interprétées comme des variables ordinales. Leurs distributions sont relativement centrées autour des valeurs 3 à 5, ce qui indique une absence de dispersion extrême mais impose néanmoins un traitement cohérent avec les autres variables continues lors de la phase de prétraitement.

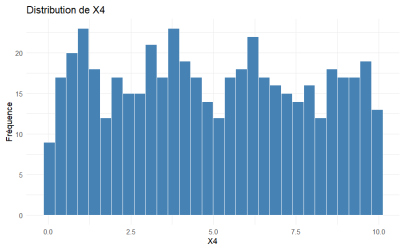

Figure 3: Distribution de X4 (première famille).

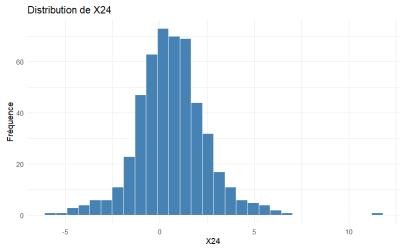

Figure 4: Distribution de X24 (deuxième famille).

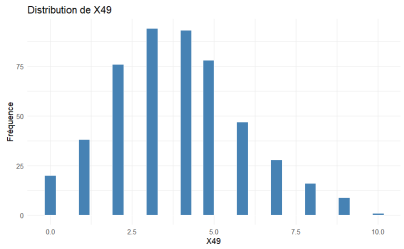

Figure 5: Distribution de X49 (troisième famille).

Concernant la variable cible y, celle-ci prend trois modalités. Les statistiques descriptives (moyenne de 2,20, médiane égale à 2 et premier quartile égal à 2) indiquent un déséquilibre des classes, la classe 1 étant deux fois moins représentée par rapport aux classes 2 et 3.
Aucun cas de valeurs manquantes n'a été observé, aucune étape d'imputation ne sera donc nécessaire.

L'analyse des distributions montre que les variables X1 à X20 suivent des distributions proches d'une loi uniforme grandement bruitée par un bruit aléatoire, sans asymétrie marquée, tandis que les variables X21 à X50 présentent des distributions plus proches de lois gaussiennes (très bruitées tout de même). Ces dernières se prêtent davantage à des méthodes paramétriques linéaires. Néanmoins, les graphiques de densité par classe révèlent un fort chevauchement entre les distributions des différentes classes pour la majorité des variables. Aucune variable prise isolément ne permet une séparation nette des classes, bien que certaines variables (notamment X9, X12, X45, X46, X47, X48 et X49) présentent un potentiel discriminant légèrement supérieur.

Enfin, l'étude de la matrice de corrélation met en évidence l'absence de corrélations fortes entre les variables explicatives. Cette faible redondance, combinée à la présence de valeurs aberrantes et au déséquilibre des classes, suggère que la performance des modèles reposera davantage sur des combinaisons multivariées que sur des effets univariés marqués.

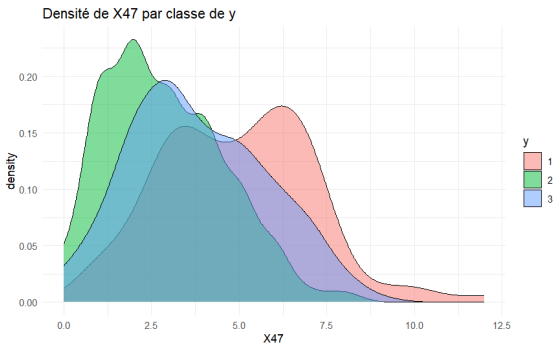

Figure 6: Densité par classe de la variable X47.

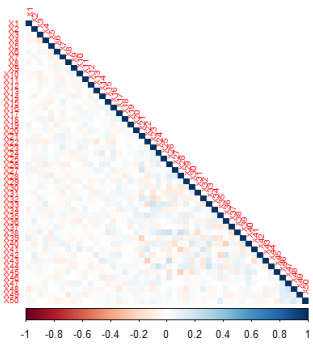

Figure 7: Matrice de corrélation.

3.2 Sélection de variables

Nous avons testé des procédures de sélection automatique telles que la sélection forward, ainsi que les critères d'information AIC et BIC. Cependant, aucune de ces approches n'a permis d'améliorer les performances par rapport aux modèles. Cela confirme que, dans ce problème, ces méthodes ne parviennent pas à un sous-ensemble de variables réellement plus informatif que simplement l'utilisation des variables non bruitées.

3.3 Transformations

D'après l'analyse descriptive, les variables **X1 à X20**, toutes comprises dans l'intervalle $[0, 10]$, présentent des distributions quasi uniformes fortement bruitées. Leurs densités conditionnelles montrent un chevauchement presque total entre les modalités de y et aucune structure discriminante exploitable. Elles ne contribuent donc que très faiblement à la séparation des classes. Par ailleurs, conserver ces variables non informatives risque d'augmenter le bruit dans les modèles et de dégrader les performances. Pour ces raisons, les variables `X1 à X20 ont été exclues de la phase de classification`. Cette suppression de variables a aussi permis de réduire la dimension, aspect particulièrement intéressant puisque nous utilisons une QDA. En effet, la QDA n'est pas performante voire peut échouer dans le cas où p est grand par rapport à n (matrices de covariance par classe non inversibles).

Les variables restantes **X21 à X50** présentent quant à elles une grande hétérogénéité, avec des amplitudes très différentes, des valeurs négatives pour certaines et la présence de valeurs atypiques. Afin de rendre ces variables directement comparables et de limiter l'influence des échelles différentes dans les méthodes basées sur les distances, une `standardisation par centrage-réduction` a été appliquée sur chaque variable.

3.4 Classifieur

La tâche de classification a été réalisée à l'aide d'un modèle de **Quadratic Discriminant Analysis (QDA)**. Ce choix se justifie par l'observation, issue de la phase descriptive, que les variables standardisées $X21$ à $X50$ présentent des formes proches de distributions gaussiennes, bien que bruitées, ce qui rend les modèles discriminants paramétriques adaptés au problème.

Contrairement à la méthode LDA, QDA autorise une **matrice de covariance différente pour chaque classe**, introduisant ainsi des frontières de décision potentiellement non linéaires. Ce choix permet de capturer des effets de variance propres à chacune des trois classes, particulièrement pertinents au vu de l'hétérogénéité observée.

3.5 Résultats et interprétations

L'évaluation du classifieur QDA sur la plateforme de test aboutit à une accuracy de **0,7219**. Cette performance valide le choix méthodologique effectué lors de la phase de préparation et confirme la pertinence de la réduction de dimension appliquée. En effet, l'exclusion des variables **X1 à X20**, identifiées comme très

faiblement informatives et fortement bruitées, a eu un impact notable : la performance est passée d'environ 0,62 à 0,72, soit un gain de près de 0,10 point d'accuracy. Ce résultat illustre deux aspects importants :

- Réduction du bruit : en supprimant des variables quasi uniformes et peu discriminantes, le modèle n'est plus perturbé par une information aléatoire qui dégradait sa capacité à estimer correctement les matrices de covariance propres à la QDA.
- Stabilisation de l'estimation : avec 30 variables au lieu de 50, le rapport p/n est plus favorable, ce qui améliore la stabilité numérique des matrices de covariance estimées par classe.

Au regard des autres modèles testés, cette performance place la QDA parmi les meilleurs classifieurs explorés. Les méthodes linéaires classiques (régression logistique, LDA) se sont révélées moins adaptées.

Les méthodes non linéaires, tel que SVM à noyau RBF, n'apportent pas non plus de gain. Les données ne présentent pas de non-linéarités marquées, et la faible informativité de plusieurs variables conduit ces modèles flexibles à surapprendre rapidement, dégradant leurs performances en généralisation. Les approches basées sur les arbres ont également été écartées. Elles ont tendance à surexploiter le bruit présent dans les variables X1 à X20, même après tuning, et n'ont pas réussi à dépasser les performances de la QDA. Enfin, les techniques de réduction de dimension (ACP) n'ont pas apporté d'amélioration sur la performance

Ainsi, la QDA apparaît comme le meilleur compromis entre flexibilité et robustesse, ce qui explique sa supériorité sur les modèles testés dans ce contexte.

4 Jeu de données réel

4.1 Présentation du jeu de données

Le jeu de données réelles a été téléchargé depuis le site OpenML et se nomme "Heart Failure Prediction - Clinical Records". Il comprend les dossiers cliniques de patients atteints d'une maladie cardiovasculaire, détaillant divers attributs médicaux susceptibles d'entrainer une insuffisance cardiaque. On retrouve les données de 5000 patients (donc 5000 observations) pour 13 variables cliniques (dont la variable à prédire). Nous détaillerons dans la prochaine section à quoi correspond chacune des variables. La variable à prédire est un facteur binaire représentant la mort du patient. Il s'agit donc d'un problème de classification binaire.

4.2 Analyse descriptive

Chacune des variables correspond à une grandeur médicale mesurée directement sur le patient:
- age: Age du patient (en années)
- anaemia: Etat d'anémie (diminution du nombre de globules rouges) (0: Non, 1: Oui)
- creatinine_phosphokinase: Niveau de l'enzyme CPK dans le sang (mcg/L)
- diabetes: Si le patient est atteint de diabète (0: Non, 1: Oui)
- ejection_fraction: Pourcentage de sang quittant le cœur à chaque contraction.
- high_blood_pressure: Si le patient est atteint d'hypertension (0: Non, 1: Oui)
- platelets: Plaquettes dans le sang (kiloplaquettes/mL)
- serum_creatinine: Niveau de creatinine dans le sang (mg/dL) (correspond à une insuffisance rénale)
- serum_sodium:Niveau de sodium dans le sang (mEq/L) (facteur de la pression artérielle notamment)
- sex: Sexe biologique du patient (0: Femme, 1: Homme)
- smoking: Si le patient fume (0: Non, 1: Oui)
- time: Période de suivi (jours)
- DEATH_EVENT: Si le patient est mort durant la période de suivi (0: Non, 1: Oui)

Nous avons affiché pour chacune des variables : leur moyenne, médiane, éléments extrémaux; leur distribution et:
- leur densité par rapport à la valeur de DEATH_EVENT pour les variables numériques.
- la relation entre les variables catégorielles et la variable DEATH_EVENT (analyse bivariée).

Voici ce que nous avons remarqué:

Concernant la distribution des variables:

Plusieurs variables catégorielles sont déséquilibrées dont DEATH_EVENT (31% de décès). Il s'agit des variables high_blood_pressure (36% oui), sex (65% d'hommes) et smoking (31% oui).
Pour les variables numériques, plusieurs d'entre elles révèlent une distribution très asymétrique ou encore une très forte amplitude. Il faudra donc le prendre en compte dans notre modèle (voir la partie de transformation des variables).

L'analyse de densité pour les variables numériques ne donne rien de bien concret. Seules les variables age, ejection_fraction et time semblent entraîner une légère discrimination par rapport à DEATH_EVENT. Pour l'âge, cela semble logique, il y a une tendance au décès plus le patient est agé. Nous verrons plus tard pourquoi time est à exclure de notre modèle malgré une bonne discrimination. Enfin, ejection_fraction semble montrer une bonne discrimination:

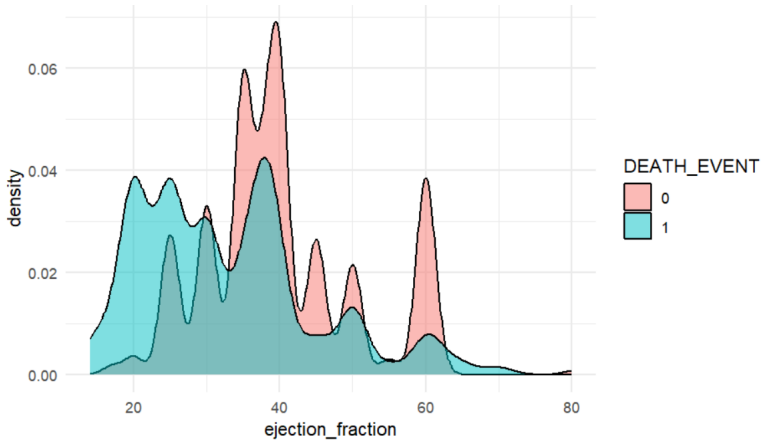

Figure 8: Densité de ejection_fraction par classe de DEATH_EVENT

De même pour les variables catégorielles : Aucune d'entre elles ne semble déterminante à pour la prédiction de DEATH_EVENT lors de l'analyse bivariée.

Enfin, nous n'observons aucune forte corrélation entre variables:

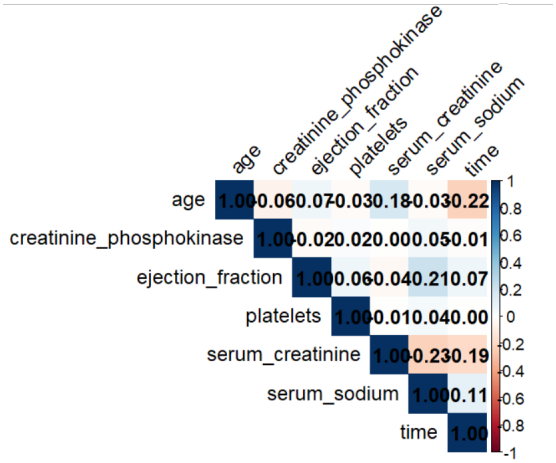

Figure 9: Matrice de corrélation

4.3 Sélection de variables

Etant donné que la variable time correspond au temps de suivi avant la mort, il s'agit d'une variable obtenue à-posteriori et ne correspond donc pas à une situation de prédiction. Il faut donc exclure cette variable de notre modèle si l'on veut qu'il soit réellement utilisable.

4.4 Transformations

Plusieurs variables numériques ont eu besoin d'une transformation log du fait de leur grande asymétrie:
- La variable platelets.
- La variable creatinine_phosphokinase.
- La variable serum_creatinine.

4.5 Classifieur

Comme précédemment, nous avons testé plusieurs classifieurs via une validation croisée. Cette fois-ci, la validation croisée présente 5 folds. Nous avons testé la regression logistique, la Random Forest, l'Extreme Gradient Boosting (XGBoost) et un modèle additif généralisé avec smooth splines pour la classification.

4.6 Résultats et interprétations

Nous avons utilisé plusieurs métriques pour évaluer la qualité de chacun des classifieurs. La moyenne et la déviation standard de la précision ainsi que la moyenne et la déviation standard de l'aire sous la courbe ROC, voici le tableau des résultats obtenus:

	Accuracy_Mean <dbl>	Accuracy_SD <dbl>	AUC_Mean <dbl>	AUC_SD <dbl>
Logistic	0.7798	0.018965759	0.8191267	0.014556954
RF	0.9902	0.004207137	0.9985189	0.001871553
XGB	0.9276	0.008234076	0.9711592	0.002992234
GAM	0.7974	0.019919839	0.8677537	0.010061129

4 rows

Figure 10: Métriques de qualités pour chacun des modèles.

Nous remarquons directement l'efficacité déconcertante de la Random Forest sur ce jeu de données. De plus, la Random Forest n'est pas (ou quasi pas) sujette au surapprentissage, nous nous en sommes assuré en changeant le nombre d'arbres générés de 100 à 500 et l'erreur ne passait pas en dessous de 95%.

Outre la validation croisée, nous avons aussi testé directement sur 4/5ème du jeu de données en entrainement et 1/5 en test et toutes les prédictions étaient bonnes.

Au vu de la taille du dataset, obtenir un aussi grand nombre de données sur un pool de patients aussi grand semble difficile. En effet, il est possible que ce dataset soit basé sur un plus petit dataset réel et que celui-ci ait été artificiellement grandis, renforçant ainsi les dynamiques internes. Ce renforcement a pu aussi améliorer les performances des modèles qui bénéficiaient déjà de ces dynamiques. C'est pourquoi nous faisons l'hypothèse que le jeu de données pourrait être biaisé en faveur de la Random Forest ou des arbres de décision en général.

5 Annexe - Code makeEnv

```r
# Installer les packages nécessaires
packages_needed <- c("MASS", "glmnet", "dplyr", "here")
to_install <- packages_needed[!(packages_needed %in% installed.packages()[, "Package"])]
if(length(to_install)) install.packages(to_install)

library(MASS)
library(glmnet)
library(dplyr)
library(here)
library(mgcv)

# ========================================
# 1. Lecture des jeux d'apprentissage

# Modifier les chemins si nécessaire
clas_data_path <- here("src", "data", "TP5_a25_clas_app.txt")
reg_data_path <- here("src", "data", "TP5_a25_reg_app.txt")

# Lecture des données
X.reg <- read.table(
  reg_data_path,
  header = TRUE,
  sep = " ",
  quote = "\"",
  stringsAsFactors = FALSE
)
X.clas <- read.table(
  clas_data_path,
  header = TRUE,
  sep = " ",
  quote = "\"",
  stringsAsFactors = FALSE
)

# ========================================
# 2. Préparation classification (QDA simple sans X1 à X20)

# Cible en facteur
X.clas$y <- as.factor(X.clas$y)

# Sélection des variables X21:X50 + y
vars_clas <- c(paste0("X", 21:50), "y")
X_clas_sel <- X.clas[, vars_clas]

# Standardisation des X
X_clas_scaled_mat <- scale(X_clas_sel[, -ncol(X_clas_sel)])

# Sauvegarde des paramètres de standardisation
X_mean_clas <- attr(X_clas_scaled_mat, "scaled:center")
X_sd_clas   <- attr(X_clas_scaled_mat, "scaled:scale")

# Data frame final pour l'entraînement
X_clas_scaled <- data.frame(X_clas_scaled_mat, y = X_clas_sel$y)

# Entraînement QDA
clas <- qda(y ~ ., data = X_clas_scaled)

# Fonction classifieur pour la plateforme
```

```r
classifieur <- function(test_set) {
  library(MASS)

  # Garder X21:X50
  test_sub <- test_set[, paste0("X", 21:50), drop = FALSE]

  # Standardisation avec les bons paramètres
  X_test_scaled <- sweep(as.matrix(test_sub), 2, X_mean_clas, FUN = "-")
  X_test_scaled <- sweep(X_test_scaled, 2, X_sd_clas, FUN = "/")

  test_scaled_df <- data.frame(X_test_scaled)

  # Prediction
  preds <- predict(clas, newdata = test_scaled_df)$class
  return(preds)
}

# ========================================
# 3. Préparation régression LASSO OPTI + BAM

# Préparation des données
X <- as.matrix(reg_data[, setdiff(names(reg_data), "y")])
y <- reg_data$y
X_scaled <- scale(X)

# LASSO pour la sélection
cv_lasso <- cv.glmnet(X_scaled, y, alpha = 1, nfolds = 10)
best_lambda <- cv_lasso$lambda.min

# Coefficients
lasso_coef <- coef(cv_lasso, s = best_lambda)
selected_vars <- rownames(lasso_coef)[lasso_coef[,1] != 0]
selected_vars <- selected_vars[selected_vars != "(Intercept)"]
cat("Variables sélectionnées par LASSO :\n")
print(selected_vars)

# 3. Construire un GAM/BAM sur les variables sélectionnées
X_scaled_df <- as.data.frame(X_scaled)
colnames(X_scaled_df) <- colnames(X)
df_gam <- data.frame(y = y, X_scaled_df[, selected_vars, drop = FALSE])

form_str <- paste0("y ~ ", paste0("s(", selected_vars, ", bs='ps', k=5)", collapse = " + "))
bam_model <- bam(as.formula(form_str), data = df_gam, family = gaussian())
# On stocke la moyenne et l'écart type pour reproduire la standardisation sur le test

X_mean_reg <- attr(X_scaled, "scaled:center")
X_sd_reg   <- attr(X_scaled, "scaled:scale")

# Fonction regresseur pour la plateforme
regresseur <- function(test_set) {
  library(mgcv)
  # Convertir en dataframe et rescale avec les variables sélectionnées.
  X_test_scaled <- sweep(as.matrix(test_set[,selected_vars]), 2, X_mean_reg[selected_vars], FUN="-")
  X_test_scaled <- sweep(X_test_scaled, 2, X_sd_reg[selected_vars], FUN="/")
  X_test <- data.frame(X_test_scaled)
  # Prédiction
  preds <- as.numeric(predict(bam_model, newdata = X_test))
  return(preds)
}

```

```r
# ========================================
# 4. Sauvegarder l'environnement minimal
save(
  clas,
  bam_model,
  classifieur,
  regresseur,
  X_mean_reg,
  X_sd_reg,
  X_mean_clas,
  X_sd_clas,
  selected_vars,
  file = "env.Rdata"
)

cat("Fichier env.Rdata créé avec succès !\n")
cat("Contenu :", ls()[ls() %in% c('clas','reg','classifieur','regresseur')], "\n")
```