UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN MATHEMATIQUES ET INFORMATIQUE APPLIQUÉES

PAR
CHEIKH SALL

Modèles prédictifs par apprentissage automatique pour la survie : application à la base de données clinique et génétique pour les tumeurs du cancer du sein.

Mai 2024

Université du Québec à Trois-Rivières

Service de la bibliothèque

Avertissement

L'auteur de ce mémoire, de cette thèse ou de cet essai a autorisé l'Université du Québec à Trois-Rivières à diffuser, à des fins non lucratives, une copie de son mémoire, de sa thèse ou de son essai.

Cette diffusion n'entraîne pas une renonciation de la part de l'auteur à ses droits de propriété intellectuelle, incluant le droit d'auteur, sur ce mémoire, cette thèse ou cet essai. Notamment, la reproduction ou la publication de la totalité ou d'une partie importante de ce mémoire, de cette thèse et de son essai requiert son autorisation.

Résumé

L'analyse de survie du cancer du sein est essentielle pour comprendre les facteurs qui influent sur la longévité des patients, en mettant l'accent sur l'identification des gènes clés dans l'évolution de la maladie. Cette recherche est complexe en raison de la diversité des types de cancer du sein, chacun ayant ses propres caractéristiques génétiques uniques. Pour identifier les gènes pertinents, les scientifiques utilisent des techniques avancées telles que l'apprentissage automatique, une branche de l'intelligence artificielle qui permet aux ordinateurs d'apprendre à partir de données, ainsi que l'analyse moléculaire pour repérer les variations génétiques associées à la maladie. Dans ce mémoire, l'objectif est de mettre en place des techniques de sélection de caractéristiques, telles que la corrélation de Pearson, la corrélation de Spearman et l'information mutuelle, pour choisir les gènes les plus pertinents liés à la durée de survie du cancer du sein. Ces caractéristiques sélectionnées seront ensuite utilisées dans différents modèles d'apprentissage automatique, tels que la forêt de décision, le SVM (Support Vector Machine) avec des cas linéaires et non linéaires, la régression linéaire, la régression Ridge, Lasso et elastic net. L'objectif final est de comparer la capacité prédictive de ces modèles pour déterminer les variables offrant la meilleure explication de la durée de survie du cancer du sein, fournissant ainsi aux médecins des informations cruciales pour prendre des décisions éclairées. Pour mener à bien cette tâche, nous avons combiné des ensembles de données génétiques et cliniques contenant la variable days_to_death indiquant la durée de vie avant le décès.

Cheikh Sall

Professeure Nadia Ghazzali

Abstract

The analysis of breast cancer survival is crucial for understanding the factors influencing patients' longevity, with a focus on identifying key genes in the disease's progression. This research is complex due to the diversity of breast cancer types, each with its own unique genetic characteristics. To identify relevant genes, scientists employ advanced techniques such as machine learning, a branch of artificial intelligence enabling computers to learn from data, as well as molecular analysis to pinpoint genetic variations associated with the disease. In this thesis, the goal is to implement feature selection techniques, such as Pearson correlation, Spearman correlation, and mutual information, to choose the most pertinent genes related to breast cancer survival duration. These selected features will then be utilized in various machine learning models, including decision trees, Support Vector Machine (SVM) with linear and non-linear kernels, linear regression, Ridge regression, Lasso, and elastic net. The ultimate objective is to compare the predictive capacity of these models to determine the variables providing the best explanation of breast cancer survival duration, thereby providing crucial information for informed decision-making by physicians. To accomplish this task, we have combined genetic and clinical datasets containing the days_to_death variable indicating the duration of life before death.

Avant-propos

Ce projet de recherche vise principalement à prédire la survie du cancer du sein en utilisant à la fois des données génétiques et cliniques. Nous appliquons des méthodes de sélection de caractéristiques telles que la corrélation de Pearson, de Spearman et l'information mutuelle pour identifier les caractéristiques les plus pertinentes pour la survie du cancer du sein. Ensuite, nous utilisons plusieurs modèles d'apprentissage automatique, notamment la forêt de décision, la régression linéaire, la régression Ridge, Lasso et Élastique Net, ainsi que la SVR linéaire et non linéaire, afin de fournir une meilleure explication de la survie du cancer du sein aux professionnels de santé. L'objectif est de fournir des outils de prédiction précis et fiables pour aider les médecins à prendre des décisions éclairées concernant le traitement et la gestion des patients atteints de cancer du sein.

Je tiens à exprimer mes sincères remerciements à ma directrice de recherche, Mme Nadia Ghazzali, pour avoir accepté de diriger mon projet de recherche. Je souhaite également la remercier pour sa disponibilité et son soutien, tant du côté financier que pédagogique. Je tiens aussi à exprimer ma gratitude envers mon codirecteur, Monsieur Venkata Satya Kumar Manem, pour toute l'aide qu'il m'a apportée durant ce travail, également du côté financier que pédagogique.

Je souhaite également exprimer ma reconnaissance envers le laboratoire d'Intelligence Artificielle Appliquée (LI2A) de l'UQTR et tout le personnel de l'UQTR, en particulier celui du département de mathématiques et d'informatique, pour tous les

services rendus.

Mes plus sincères remerciements vont également à mes parents, plus particulièrement à ma mère, Khadi Diop, pour tous les sacrifices qu'elle a faits pour moi afin que j'atteigne ce stade. Je tiens également à remercier mon oncle, Pape Memedou Gueye, pour m'avoir orienté dans le domaine des sciences et pour toute l'aide qu'il m'a apportée durant ce travail.

Je souhaite aussi exprimer ma gratitude envers tous les membres de ma famille, mes camarades de classe, mes amis et tous ceux qui m'ont encouragé durant mes études. Votre soutien a été précieux et m'a grandement aidé à réussir dans ce projet de recherche.

Table des matières

Résumé **ii**

Abstract **iv**

Avant-propos **v**

Table des matières **vii**

1 Introduction **1**

1.1 Mise en contetxe 1

1.2 Cancer du sein 2

1.3 Objectifs et structure du mémoire 4

2 Réduction de la dimensionnalité **5**

2.1 Sélection de caractéristique par l'information mutuelle 6

2.1.1 Propriétés de l'information mutuelle 7

2.1.2 Information mutuelle en terme d'entropie 7

2.1.3 Relation entre l'information mutuelle et la divergence de Kullback-Leibler 8

2.2 Sélection de caractéristiques de Pearson 9

2.2.1 Interprétation de coefficient de Pearson 10

2.3 Sélection de caractéristiques de Spearman 10

3 Apprentissage automatique **12**

3.1 ML supervisé 13

3.1.1 Classification 14
3.1.2 Régression 15
3.2 ML non supervisé 15
3.2.1 Clustering non supervisé 16
3.2.2 Réduction de dimensionnalité 16
3.3 Autres méthodes d'apprentissage automatique 17
3.4 Les algorithmes de ML supervisé 18
3.4.1 Régression linéaire simple 18
3.4.1.1 Modèle 18
3.4.1.2 Critére des moindres Carrés Ordinaires (MCO) 19
3.4.1.3 Calcul des estimateurs $\hat{\beta}_0$ et $\hat{\beta}_1$ 20
3.4.1.4 Coefficient de détermination R^2 21
3.4.2 Régression linéaire multiple 22
3.4.2.1 Modèle 22
3.4.2.2 Estimation des moindres carrés ordinaires 23
3.4.2.3 Prévision 23
3.4.2.4 Erreur de prévision 24
3.4.2.5 Coefficient de détermination R^2 24
3.4.3 Régression Ridge, Lasso et ElastiNet 25
3.4.4 Machines à Vecteurs de Support (SVM) 27
3.4.4.1 Notion de base : Hyperplan, marge et support vecteur 28
3.4.4.2 Régression par Machines à Vecteurs de Support (SVR) 30
3.4.4.2.1 SVR cas linéaire 31
3.4.4.2.2 SVR cas non linéaire 34
3.4.5 Les arbres et forêts aléatoire (Random Forest) 35
3.4.5.1 Les arbres de décision 36
3.4.5.1.1 Arbres de régression (CART) 37
3.4.5.2 Forêt aléatoire 41
3.4.6 Prédiction de la probabilité de survie globale 43
3.4.6.1 Modélisation de la probabilité de survie 43

3.4.6.2 Modèle de régression de Cox 44
3.4.6.2.1 Régression de Cox à risque proportionnel . . . 44
3.4.6.3 Indice de concordance IC 45

4 Analyse statistique des données 47
4.1 Description des données 47
4.2 Analyse exploratoire 48
4.2.1 Modèle univarié : Association entre un gène et la survie 50
4.3 Modèle multivarié 53
4.4 Méthode basée sur la corrélation et approches en apprentissage automatique 54
4.5 Étude comparative 55
4.5.1 Méthode de sélection de caractéristique de Pearson 55
4.5.2 Méthode de sélection de caractéristique de Spearman 61
4.5.3 L'information mutuelle 63
4.6 Synthèse des résultats obtenus 67

5 Conclusion et perspectives 71

Bibliographie 73

A Importation et Transformation des données 80

B Analyse exploratoire des données 83

C Sélection de caractéristique de Pearson et modèle d'apprentissage automatique 87

D Sélection de caractéristiques de Spearman, de l'information mutuelle et les modèles d'apprentissage automatique. 131

Liste des tableaux

4.1 Données de patients atteints de cancer. 48
4.2 Statistiques sur la Durée de Survie (days_to_death) pour le Cancer du Sein 49
4.3 Résultats du modèle de régression de Cox pour les génes les plus significatifs dans la survie 50
4.4 Identification des gènes les plus significatifs dans la survie du cancer du sein par leur symbole 52
4.5 Statistiques sur la Durée de Survie (days_to_death) pour le Cancer du Sein 54
4.6 Tableau comparatif des performances des modèles de régression pour la méthode de sélection de caractéristiques de Pearson 58
4.7 Corrélations de Pearson maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles 59
4.8 Indice de concordance maximale entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles 60
4.9 Tableau comparatif des performances des modèles de régression pour la méthode de sélection de caractéristiques de Spearman 61
4.10 Corrélations de Spearman maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles 62

4.11 Indice de concordance maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles 63
4.12 Tableau des pourcentages des scores d'information mutuelle 64
4.13 Tableau comparatif des performances des modèles de régression pour la méthode de sélection de caractéristiques de mutuelle information . . 64
4.14 Corrélations maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles 65
4.15 Indice de concordance maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles 66
4.16 Comparaison des performances des différents modèles de régression dans la prédiction de la survie des patients atteints de cancer du sein selon les méthodes de sélection de caractéristiques 67
4.17 Indices de concordance entre les valeurs réelles et prédites pour les différents modèles de régression dans la prédiction de la survie des patients atteints de cancer du sein selon les méthodes de sélection de caractéristiques 67

Liste des figures

3.1 Hyperplan qui sépare deux classes de points. 28
3.2 Support Vecteur 29
3.3 SVM exemples séparables et non séparables. 30
3.4 Excemple d'arbre de décision 37

4.1 Distribution de *days_to_death* 50
4.2 Courbe de survie 52
4.3 Performance du modèle de régression linéaire pour la méthode de sélection de caractéristiques de Pearson 56
4.4 Corrélation entre les valeurs réelles et les valeurs prédites en fonction du nombre de variables 57
4.5 indice de concordance entre les valeurs réelles et les valeurs prédites en fonction du nombre de variables 57
4.6 Performance du modèle de random Forest pour la méthode de sélection de caractéristiques de Pearson 68
4.7 Courbe entre valeurs réelles et prédites 69

Chapitre 1

Introduction

1.1 Mise en contetxe

L'analyse de survie est une méthodologie statistique visant à explorer l'occurrence et le calendrier d'événements, en se concentrant sur des entités ayant une durée de vie définie, comme des organismes vivants, des équipements ou des systèmes politiques. Cette approche est largement utilisée dans divers domaines tels que la médecine, la biologie, les sciences sociales, l'ingénierie, la gestion d'entreprise et l'économie. Les données de survie présentent des défis uniques, tels que la censure et les covariables dépendantes du temps[1, 2]. La survie ne se restreint pas à la biologie ; elle englobe une variété de phénomènes caractérisés par des processus temporels jusqu'à un événement. Les chercheurs utilisent l'analyse de survie pour décrire, mesurer et analyser les caractéristiques des événements, permettant des prédictions sur la survie et le délai avant un changement de statut[1]. Les applications de l'analyse de survie sont diverses. En médecine, elle est employée dans les essais cliniques pour évaluer l'efficacité de nouveaux médicaments. En biologie, elle permet d'étudier les perspectives évolutives de la sénescence. Dans les sciences sociales, elle s'applique à des sujets tels que le chômage, la

récidive liée à la consommation de drogue, les ruptures conjugales. Elle est également utilisée en ingénierie pour des tests de durée de vie et en économie pour analyser des processus comme la migration internationale[1, 2].

Il est donc intéressant de prédire la survie en utilisant des données cliniques et génétiques, et de générer des modèles pronostiques pour comprendre les processus pathologiques, explorer les interactions entre les facteurs de risque, et prédire le comportement des nouveaux patients dans le contexte des données connues[3]. L'utilisation de la prédiction de survie permet d'identifier des patients présentant des pronostics comparables, ce qui pourrait être utile pour la stratification des risques dans les essais cliniques. De plus, cela permettrait aux médecins et aux patients d'être mieux informés sur le pronostic, et ils pourraient en tenir compte dans un processus décisionnel partagé[4]. À titre d'exemple, lors d'une discussion avec un patient de 80 ans, le médecin pourrait expliquer, basé sur les analyses de données et les modèles de survie, qu'il y a une forte probabilité de maintenir une bonne santé au cours de la première année suivant le diagnostic. En revanche, les perspectives à trois ans indiquent une diminution probable de la stabilité de la santé, avec une probabilité réduite de maintenir le bien-être initial. Les résultats d'une telle analyse, à leur tour, peuvent fournir des informations importantes ayant des implications politiques[1].

1.2 Cancer du sein

Le cancer est un fardeau dans l'échelle mondiale. En effet, le nombre de nouveaux cas diagnostiqués en 2020 était 19,3 millions (19 300 000), et le nombre de personnes qui en sont décédées était de 10 millions (10 000 000)[5]. À l'avenir, on prévoit une augmentation continue de ces chiffres, car on estime que le nombre de nouveaux dignostiques atteindra 28,4 millions d'ici 2040[6]. De nombreux facteurs peuvent expliquer cette augmentation, notamment les influences environnementales, le stress interne ou l'hérédité, le vieillissement et la croissance de la population anisi

que l'évolution de la prévalence de certains facteurs explicatifs liés au développement économique et social. Le cancer le plus fréquent et la principale cause de décés varient selon les pays et les facteurs socio-économiques. Les facteurs responsables varient d'un patient à l'autre en fonction du type de cancer et de la situation géographique[7]. Les cancers les plus couramment diagnostiqués dans le monde sont le cancer du sein chez les femmes (2,26 millions de cas, $11,7\%$), le cancer du poumon (2,21 millions de cas, $11,4\%$) et le cancer du prostate (1,41 millions de cas, $7,3\%$). En ce qui concerne les décés dus au cancer, les principales causes sont le cancer du poumon (1,79 million de décés, 18% du total de décés liés au cancer), le cancer du foie (830 000, $8,3\%$), le cancer de l'estomac (769 000, environs $7,7\%$) et le cancer du sein (680 000, $6,9\%$)[5]. Après avoir examiné les statistiques mondiales sur le cancer et ses tendances, il est intéressant de se concentrer sur un type spécifique de cancer, à savoir le cancer du sein.

Le cancer du sein est caractérisé par une croissance incontrôlée de cellules malignes dans le tissu épithélial mammaire[8]. Le cancer du sein est l'un des défis majeurs de santé publique au niveau mondial. C'est une maladie complexe qui touche principalement les femmes[9]. Le cancer du sein est causé non seulement par des facteurs génétiques, mais aussi par différentes variables, comme le sexe, l'âge, le tabagisme, consommation d'alcool, ménopause [10]. Au fil des décennies, des efforts constants ont été déployés pour caractériser l'hétérogénéité de cette maladie et identifier les facteurs qui influent sur la réponse au traitement et la survie.

Cependant, en raison de la diversité des caractéristiques associées au cancer du sein, cibler de manière précise les éléments susceptibles d'influencer la survie du cancer du sein demeure difficile, voire impossible.

1.3 Objectifs et structure du mémoire

Pour répondre à tous ces problèmes évoqués plus haut, l'objectif de notre étude est de mettre en place des techniques statistiques de sélection de caractéristiques telles que la corrélation de Pearson, la corrélation de Spearman et l'information mutuelle afin de choisir les variables les plus pertinentes liées à la durée de survie du cancer du sein. Ensuite, nous utiliserons ces caractéristiques sélectionnées pour comparer différents modèles d'apprentissage automatique tels que la régression linéaire, le SVM, la forêt aléatoire, la régression Ridge, le Lasso et l'Elastic Net pour déterminer les variables permettant une meilleure explication de la durée de survie du cancer du sein. L'objectif est de fournir aux médecins des informations précises pour prendre des décisions éclairées.

Notre travail est composé de cinq chapitres. Le premier chapitre est consacré à l'introduction. Le deuxième présente les techniques de réduction de la dimensionnalité. Le troisième aborde l'apprentissage automatique. Le quatrième est dédié à la présentation des résultats, et le cinquième est consacré à la conclusion et aux perspectives.

Chapitre 2

Réduction de la dimensionnalité

La réduction de la dimensionnalité des ensembles de données devient de plus en plus essentielle en raison de la prolifération des données. Dans de nombreux domaines, la résolution d'un probléme repose sur un ensemble de variables (caractéristiques). Ainsi, l'ajout de nouvelles variables pour modéliser le problème engendre des difficultés à divers niveaux tels que la complexité, le temps de calcul et la détérioration de la résolution du système en présence de données bruitées. Une méthode courante de réduction de la dimensionnalité vise à trouver une représentation plus réduite des données initiales dans un espace adapté. Les méthodes de réduction de la dimentionnalité sont genéralement classées en deux catégories [11].

Une réduction basée sur la sélection de caractéristiques qui consiste à sélectionner les caractéristiques les plus pertinentes parmi l'ensemble des variables décrivant le phénoméne étudié. Cette méthode vise à identifier les éléments les plus significatifs pour représenter de manière concise et précise les données en question.

Une méthode de réduction alternative repose sur la transformation des données, également appelée extraction des caractéristiques. Cette approche implique de rempla-

cer l'ensemble initial de données par un nouvel ensemble réduit, construit à partir des caractéristiques de départ. L'objectif est de créer un nouvel espace de représentation qui capture l'essentiel des informations tout en diminuant la complexité et la dimensionnalité de l'ensemble de données initial[11]. Parmi ces méthodes de sélection de caractéristiques, on peut citer la méthode de sélection de caractérisques basée sur l'information mutuelle, celle de Pearson et celle de Spearman. Dans la suite de ce chapitre, nous allons brièvement présenter ces trois méthodes de sélection de caractéristiques.

2.1 Sélection de caractéristique par l'information mutuelle

La théorie mathématique de l'information mutuelle (MI) est un concept issu de la théorie de l'information, développée par Claude Shannon dans les années 1948[12]. La MI est une mesure de la dépendance entre deux variables aléatoires continues ou discrètes [13]. En apprentissage automatique et en statistique, l'information mutuelle est souvent utilisée comme critère pour sélectionner les caractéristiques les plus informatives dans un ensemble de données. Soient X et Y deux variables aléatoires discrètes, avec leurs distributions de probabilité respectives $P(X)$ et $P(Y)$. L'information mutuelle entre X et Y est définie comme suit[13, 14, 15, 16] :

$$I(X,Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \log \left(\frac{P(x,y)}{P(x)P(y)} \right) \quad (1)$$

où

$I(X,Y)$ représente l'information mutuelle entre X et Y.

$P(x,y)$ est la probabilité conjointe de l'occurrence des événements x dans X et y dans Y.

$P(x)$ et $P(y)$ sont les probabilités marginales des événements de x dans X et y dans Y, respectivement.

L'information mutuelle mesure à quel point la connaissance d'une variable (X ou Y) réduit l'incertitude associée à l'autre variable. Si l'information mutuelle est élevée, cela signifie que les deux variables sont étroitement liées et que la connaissance de l'une fournit des information utiles pour prédire l'autre [15].

2.1.1 Propriétés de l'information mutuelle

L'information mutuelle (MI) est toujours non négative : $I(X, Y) \geq 0$.
Elle est nulle si et seulement si les variables X et Y sont indépendantes.
La MI est symétrique : $I(X, Y) = I(Y, X)$
Si X et Y sont indépendantes, alors $I(X, Y) = 0$ [16]
Dans la suite, nous allons expliquer comment la MI peut être comprise en relation avec l'entropie. Ensuite, nous montrerons comment ces concepts sont liés à la divergence de Kullback-Leibler.

2.1.2 Information mutuelle en terme d'entropie

La MI est basée sur le concept d'entropie, qui mesure l'incertitude ou le désordre dans une variable aléatoire. L'entropie $H(X)$ d'une variable aléatoire discréte X est définie comme [17, 18] :

$$H(X) = -\sum_{x \in X} P(x) \log P(x) \quad (2)$$

où
$H(X)$ est l'entropie de la variable aléatoire X.
$P(x)$ est la probabilité de l'événement x se produise dans X. L'entropie mesure le nombre moyen de bits nécessaires pour représenter les résultats possibles d'une va-

riable aléatoire X. Plus l'entropie est élevée, plus la variable est incertaine, car il y a plus d'incertitude sur les résultats possibles.
La MI entre deux variables aléatoires X et Y peut être exprimée en termes d'entropie comme suit[17, 18] :

$$I(X,Y) = H(X) + H(Y) - H(X,Y) \quad (3)$$

où :
$I(X,Y)$ représente l'information mutuelle entre les variables X et Y.
$H(X)$ est l'entropie de la variable aléatoire X.
$H(Y)$ est l'entropie de la variable aléatoire Y.
$H(X,Y)$ est l'entropie conjointe des variables X et Y, définie comme :

$$H(X,Y) = -\sum_{x\in X}\sum_{y\in Y} P(x,y)\log P(x,y). \quad (4)$$

2.1.3 Relation entre l'information mutuelle et la divergence de Kullback-Leibler

L'information mutuelle est liée à la divergence de Kullback-Leibler(KL), également appelée divergence relative ou divergence de l'information. La divergence de KL mesure la différence entre deux distributions de probabilité. Pour deux variables aléatoires X et Y, la divergence de KL de la distribution $P(Y)$ par rapport à la distribution P(X,Y) est définie comme suit [19] :

$$D_{KL}(P(Y)||P(X,Y)) = \sum_{y\in Y} P(Y)\log\left(\frac{P(Y)}{P(X,Y)}\right). \quad (5)$$

La divergence de KL est toujours non négative et est égale à zéro si et seulement si les distributions P(Y) et P(X,Y) sont identiques. L'information mutuelle (MI) est liée

à la divergence de KL par la formule suivante :

$$I(X,Y) = D_{KL}(P(X,Y)||P(X)P(Y)). \quad (6)$$

Cela montre que MI mesure la similitude entre la distribution jointe $P(X,Y)$ et le produit des distributions marginales $P(X)$ et P(Y). Plus MI est élevée, plus les distributions jointes et marginales sont similaires, indiquant une forte dépendance entre les variables X et Y.

2.2 Sélection de caractéristiques de Pearson

Historiquement, la corrélation de Pearson a joué un rôle pionnier en tant que première mesure formelle de la corrélation et est rapidement devenue la méthode dominante pour évaluer l'association entre deux variables aléatoires. Initialement conçue par le statisticien Carl Pearson[20], ce coefficient a été introduit comme un outil statistique permettant d'analyser et d'explorer le degré de corrélation entre des variables. Le coefficient de corrélation de Pearson de deux variables X et Y est formellement défini comme suite [21, 22, 23] :

$$r_p = \frac{\text{cov}(X,Y)}{\sqrt{\text{var}(X)\text{var}(Y)}} \quad (7)$$

où

cov() désigne la covariance entre X et Y

var() désigne la variance entre les deux caractérisques X et Y

De manière équivalente, l'estimation du coefficient de corrélation d'un échantillon est défini dans l'équation suivante :

$$r_p = \frac{\sum_{j=1}^{N}(x_j - \bar{X})(y_j - \bar{Y})}{\sqrt{\sum_{j=1}^{N}(x_j - \bar{X})^2 \sum_{j=1}^{N}(y_j - \bar{Y})^2}} \quad (8)$$

où : $\bar{X} = \sum_{j=1}^{N} x_j$ et $\bar{Y} = \sum_{j=1}^{N} y_j$, représentent respectivement la valeur moyenne de X et Y. Le coefficient de corrélation de Pearson est calculé à l'aide de la méthode des produits croisés. Cette méthode implique tout d'abord le calcul des moyennes individuelles des deux variables, puis la détermination des écarts entre chaque valeur de variable et sa moyenne respective. Ensuite, ces écarts sont multipliés entre eux pour obtenir les termes de la corrélation, qui en fin de compte reflètent le degré de corrélation entre les variables.

2.2.1 Interprétation de coefficient de Pearson

La valeur de r_p se situe dans l'intervalle -1 à $+1$. Si $r_p > 0$, cela indique une corrélation positive entre les deux variables, signifiant que lorsque la valeur d'une variable augmente, la valeur de l'autre variable tend également à augmenter. Si $r_p < 0$, cela suggére une corrèlation négative, où l'augmentation de la valeur d'une variable est associée à une diminution de la valeur de l'autre variable. La magnitude absolue de r_p est indicative de la force de la corrélation. Une valeur absolue de r_p plus élevée reflète une corrélation plus forte. Lorsque le coefficient de corrélation se rapproche de 1 ou -1, la corrélation est considérée comme étant plus intense. En revanche, lorsque le coefficient de corrélation est proche de 0, cela suggère une corrélation faible et enfin si $r_p = 0$, cela signifie il n y aucune corrélation entre les variables.

2.3 Sélection de caractéristiques de Spearman

La corrélation de Spearman est fréquemment utilisée pour évaluer l'association entre des variables mesurées sur une échelle d'intervalles ou ordinales. Le calcul du coefficient de corrélation de Spearman se fait de la même manière que celui du coefficient de corrélation de Pearson [21, 23, 24].

Cependant, une distinction fondamentale existe : tandis que le calcul de r_p implique l'utilisation des valeurs réelles de l'échantillon, le calcul de r_s nécessite la transformation des valeurs de l'échantillon en rangs, qui sont ensuite attribuées dans la plage $[1, N]$. Pour effectuer cette transformation en rang, les valeurs de X et Y sont triées par ordre croissant, puis se voient assigner un entier $r \in [1, N]$. Si X et Y ont des rangs différents le coefficient de corrélation de Spearman peut être calculé par[25] :

$$\rho_S = 1 - \frac{6\, d_{ij}^2}{n\,(n^2 - 1)} \quad (9)$$

où

d_{ij} représente la différence entre les rangs de chaque paire de valeurs appariées entre X et Y.

n est le nombre total de paires appariées de valeurs X et Y.

Si X et Y ont des rangs identiques, le coefficient de corrélation de Spearman se calcule de la même manière que celui de Pearson, exprimé par la formule suivante[24, 25] :

$$r_s = \frac{\sum_{j=1}^{N}(x_j - \bar{X})(y_j - \bar{Y})}{\sqrt{\sum_{j=1}^{N}(x_j - \bar{X})^2 \sum_{j=1}^{N}(y_j - \bar{Y})^2}}. \quad (10)$$

Interprétation et hypothèses : En convertissant les valeurs en rangs, la corrélation de Spearman se focalise sur l'amplitude relative des valeurs, ce qui lui permet d'évaluer la force de la relation monotone générale entre X et Y. De plus, elle se révèle robuste face aux valeurs aberrantes dans les données. Par conséquent, les hypothèses sont moins restrictives : X et Y doivent être des variables continues ou ordinales et présenter une relation monotone[23]. La gamme de valeurs, ainsi que son interprétation, demeurent cohérentes avec celles de la corrélation de Pearson.

Chapitre 3

Apprentissage automatique

L'apprentissage automatique désigné par ML, constitue une branche de l'intelligence artificielle qui se fonde sur des algorithmes d'apprentissage statistique[26]. Il englobe la capacité d'un systéme informatique à assimiler son environnement et à améliorer automatiquement sa performance grâce à l'expérience, sans nécessiter une programmation explicite[27]. L'objectif de l'apprentissage automatique est de permettre aux algorithmes d'apprendre à partir des données fournies, d'extraire des informations et de formuler des prédictions sur des données non traitées préalablement, en se basant sur les connaissances acquises[28].

Dans le domaine d'apprentissage automatique, diverses techniques peuvent être mises en œuvre pour des tâches telles que la prédiction, la classification ou la régression. Parmi les nombreuses méthodes disponibles, on peut citer la régression linéaire simple, les Machines à Vecteurs de Support(SVM), les Arbres de Classification et de Régression (CART) ainsi que la méthode des k Plus Proches Voisins (k-NN).

Les algorithmes de ML se regroupent généralement en trois grandes catégories :

3.1 ML supervisé

Dans le cadre de l'apprentissage supervisé, l'algorithme dispose d'un accés à la variable que l'on cherche à prédire, également appelée variable cible[27, 29, 30]. Cette variable cible peut représenter divers aspects tels que la présence ou l'absence d'une maladie, la sévérité des symptômes ou encore un résultat clinique futur. L'objectif fondamental ici est de mettre en œuvre un algorithme qui puisse apprendre de manière optimale la fonction qui capture au mieux la relation entre les données d'entrée et la variable cible[26, 27].

Le qualificatif "supervisé" attribué à ce type d'apprentissage provient du fait que l'algorithme bénéficie d'une connaissance préalable des valeurs attendues en sortie[27]. Par exemple, dans le cas d'un patient souffrant de dépression par rapport à un groupe de contrôle sain, ces valeurs sont préalablement établies. L'algorithme est formé au moyen de multiples exemples et a la possibilité de recevoir des commentaires durant son processus d'apprentissage, en fonction de la proximité de ses prédictions par rapport à la véritable variable cible.

Cette méthode d'apprentissage est souvent comparée à l'idée d'apprentissage avec un enseignant, où l'enseignant possède les réponses correctes et peut corriger l'algorithme en cas d'erreur[27]. Ainsi, l'apprentissage devient le processus itératif de prédiction et d'ajustement successif, visant à minimiser autant que possible l'écart entre les prédictions en sortie et les valeurs cibles réelles. L'évaluation des performances se réalise en comparant les prédictions de l'algorithme aux véritables valeurs cibles, mais en utilisant des données qui n'ont pas été utilisées durant l'apprentissage[27, 29].

Selon que la varible cible est de nature catégorielle ou continue, cela définit respectivement une tâche d'apprentissage supervisé comme un problème de classification ou de régression[27, 30, 31].

3.1.1 Classification

Les algorithmes de classification ont pour objectif de prédire l'inclusion dans un groupe donné, souvent désigné sous les termes d'étiquettes ou de classes, pour un ensemble spécifique d'observations. Ces types d'algorithmes sont fréquemment employés dans la recherche portant sur les troubles cérébraux. L'engouement pour cette catégorie d'algorithmes découle en grande partie de la capacité à simplifier la plupart des problèmes cliniques en décisions catégoriques. Par exemple, il s'agit de déterminer si un patient donné devrait être traité avec le médicament A, B ou C.

La classification diagnostique représente l'application la plus élémentaire d'un classificateur dans le contexte des troubles cérébraux. Dans cette tâche, l'algorithme est instruit pour différencier, par exemple, les patients atteints d'une maladie spécifique des individus en bonne santé. Jusqu'à présent, une majorité écrasante d'études portant sur la classification diagnostique se sont appuyées sur des données de neuroimagerie. Un nombre moindre d'études ont recouru à d'autres types de données, tels que les symptômes moteurs pour l'identification des personnes souffrant de la maladie de Parkinson, ou les informations génétiques pour repérer les patients schizophrènes.

Cette approche peut être aisément étendue au domaine du diagnostic différentiel, où l'algorithme est en mesure de prédire la probabilité qu'un patient particulier appartienne à chacune des diverses catégories diagnostiques. De plus, la classification peut être adaptée pour distinguer entre différents sous-types à l'intérieur du cancer du sein. Une autre utilisation des classificateurs dans le contexte des cancers se situe dans la classification pronostique, impliquant la prévision de résultats longitudinaux tels que l'évolution de la maladie, la transition d'un état à un autre, ou encore la réaction au traitement pour une patiente spécifique. À titre d'exemple, quelques études ont récemment cherché à anticiper la progression du cancer du sein, ainsi que la transition de la phase précoce à une forme plus avancée, en exploitant des données d'imagerie médicale et de paramètres cliniques[27].

3.1.2 Régression

La régression est une technique d'apprentissage supervisé utilisée pour prédire des valeurs continues. Elle peut être appliquée dans des domaines similaires à ceux des classificateurs, en modélisant les relations entre variables. Par exemple, Mechelli et al. (2017)[32] ont utilisé à la fois une classification et une régression pour prédire le niveau de fonctionnement des individus à haut risque de psychose, en utilisant des scores continus pour mesurer le fonctionnement.
En régression, on modélise les valeurs d'une variable de sortie en fonction des variables d'entrée, et cela peut être appliqué à diverses situations, telles que la prédiction des prix immobiliers, des ventes, des résultats d'examens, ou encore des mouvements boursiers. La forme la plus simple est la régression linéaire, qui tente d'ajuster une ligne droite aux données lorsque la relation entre les variables est linéaire.
La régression trouve sa place dans deux domaines. Tout d'abord, elle est utilisée pour la prévision et la prédiction, ce qui la rapproche de l'apprentissage automatique. De plus, elle peut servir à déterminer des relations causales entre les variables indépendantes et dépendantes, bien que cela ne montre que les relations entre une variable dépendante et un ensemble fixe de variables indépendantes[27].

3.2 ML non supervisé

L'apprentissage non supervisé implique des méthodes où les algorithmes opèrent sans réponses préétablies ni supervision directe[33, 34, 35]. Ils travaillent indépendamment pour découvrir des schémas significatifs dans les données, sans objectifs prédéfinis. Contrairement à l'apprentissage supervisé, il n'y a pas de valeurs cibles à prédire, mais plutôt une recherche de structures sous-jacentes[35]. Les algorithmes apprennent des caractéristiques à partir de données non étiquetées et les utilisent pour catégoriser de nouvelles données, souvent pour le regroupement ou la réduction de dimension. Cette

approche, fondée sur des principes statistiques, permet de reconnaître des modèles non identifiés dans les données et est particulièrement adaptée lorsque les catégories sont inconnues. Elle est pertinente pour la recherche sur les troubles cérébraux, notamment en identifiant des similitudes entre données et en simplifiant leur complexité. L'apprentissage est fréquemment utilisé pour effectuer du regroupement (clustering) et pour réduire la dimensionnalité des données.

3.2.1 Clustering non supervisé

Le clustering non supervisé est une tâche qui, sans supervision, vise à regrouper un ensemble d'objets en classes caractérisées par leur similarité. Chaque classe représente un groupe d'objets partageant des similitudes internes et se distinguant des objets des autres classes. Ainsi, le clustering non supervisé repose sur la notion essentielle de similarité (ou de distance) [36]. Il existe de multiples approches de clustering non supervisé qui utilisent différents algorithmes sous-jacents pour regrouper les points de données en fonction de leur similarité. Une approche simple du clustering non supervisé est le k-means. Ici, dans sa version de base le nombre de classes à identifier est prédéfini par un paramètre k fixé à l'avance. Chaque classe est représentée par un centre de classe, qui est un point de données artificielles représentant la valeur moyenne (ou médiane) de tous les points attribués à cette classe [28, 34].

3.2.2 Réduction de dimensionnalité

La réduction de dimensionnalité consiste à réduire la dimensionnalité des modèles tout en préservant les informations importantes. De nombreuses méthodes de réduction de dimensionnalité se concentrent sur la recherche de représentations de faible dimension de modèles de haute dimension, appelées variables latentes, représentations la-

tentes ou plongements latents. Cette méthode utilise une transformation orthogonale pour changer les variables qui sont liées entre elles en un nouvel ensemble de variables qui n'ont plus de liens, et on les appelle les composantes principales. La réduction de dimensionnalité peut être utilisée pour diverses tâches, par exemple la visualisation, le prétraitement pour les méthodes de reconnaissance de formes ou pour les algorithmes symboliques. Pour permettre la compréhension et l'interprétation humaine des données de grande dimension, la réduction aux espaces à 2 et 3 dimensions est une tâche importante[37].

3.3 Autres méthodes d'apprentissage automatique

Il existe d'autres types d'apprentissage automatique tels que l'apprentissage semi-supervisé et l'apprentissage par renforcement. L'apprentissage semi-supervisé (SSL) se positionne entre l'apprentissage supervisé et non supervisé en traitant des ensembles de données comprenant à la fois des données étiquetées et non étiquetées. Contrairement à l'apprentissage supervisé, où toutes les données sont étiquetées, et à l'apprentissage non supervisé, où aucune étiquette n'est fournie, l'apprentissage semi-supervisé utilise partiellement des données étiquetées pour former le modèle tout en exploitant les données non étiquetées[27, 38, 39]. Cette approche équilibrée est particulièrement adaptée aux situations où toutes les caractéristiques sont présentes, mais certaines n'ont pas de cibles associées. Ces circonstances se présentent fréquemment lorsque l'étiquetage d'images est chronophage ou devient prohibitif. L'apprentissage semi-supervisé est couramment appliqué aux images médicales, où un médecin peut annoter un petit sous-ensemble d'images pour former le modèle. Ce dernier est ensuite utilisé pour classer le reste des images non étiquetées de l'ensemble de données. L'ensemble de données étiqueté résultant est ensuite utilisé pour entraîner un modèle opérationnel qui devrait théoriquement surpasser les modèles non supervisés[38]. En ce qui concerne l'apprentissage par renforcement, l'objectif est de développer un système

capable d'apprendre en interagissant avec son environnement, de manière similaire au conditionnement opérant. Dans ce type d'apprentissage, le comportement de l'algorithme est façonné par une séquence de récompenses et de pénalités, dépendant de la précision de ses décisions par rapport à un objectif défini par le chercheur. Contrairement à l'apprentissage supervisé, où l'algorithme utilise des exemples pour modéliser le comportement, l'apprentissage par renforcement permet à l'algorithme d'explorer librement, c'est-à-dire par essais et erreurs, afin de déterminer quelles actions maximisent les récompenses et minimisent les pénalités. L'apprentissage par renforcement représente l'un des domaines les plus prometteurs de l'apprentissage automatique dans de nombreuses disciplines[27].

3.4 Les algorithmes de ML supervisé

3.4.1 Régression linéaire simple

3.4.1.1 Modèle

La régression linéaire simple permet de modéliser une variable quantitative y en fonction d'un autre autre variable quantitative x. Les deux variables n'ont pas le même rôle : la variable y est dite variable à expliquer. Elle est aussi appelée variable réponse, ou encore variable dépendante. La variable x, quant à elle, est appelée prédicteur ou variable indépendante[40]. Le modéle étudié s'écrit :

$$\forall i \in \{1, \ldots, n\}, \quad y_i = \beta_1 + \beta_2 x_i + \epsilon_i \quad (11)$$

où (x_i, y_i) sont n observations des variables x et y. Les quantités ϵ_i viennent du fait que les points ne sont jamais parfaitement alignés sur une droite. On les appelle les

erreurs (ou bruits) et elles sont supposées aléatoires. Le modèle de regression linéaire simple peut encore s'écrit sous la forme vectorielle :

$$Y = \beta_1 + \beta_2 X + \epsilon \quad (12)$$

où :
- le vecteur $Y = [y_1, \ldots, y_n]^T$ est aléatoire de dimension n,
- $1 = [1, \ldots, 1]^T$ est le vecteur de $\mathbb{R}^n$ dont les n composantes valent toutes 1,
- $X = [x_1, \ldots, x_n]^T$ est un vecteur de dimension n donné (non aléatoire), - les coefficients β_1 et β_2 sont les paramétres inconnus (mais non aléatoire) du modèle,
- le vecteur $\epsilon = [\epsilon_1, \ldots, \epsilon_n]^T$ est aléatoire de dimension n[41].

3.4.1.2 Critére des moindres Carrés Ordinaires (MCO)

Les points (x_i, y_i) étant donnés, le but est maintenant de trouver une fonction affine f telle que la quantité $\sum_{i=1}^{n} L(y_i - f(x_i))$ soit minimale. Pour pouvoir déterminer la fonction f, encore faut-il préciser la fonction de coût L. Deux fonctions sont classiquement utilisées :
- le coût absolu $L(u) = |u|$;
- le coût quadratique $L(u) = u^2$.

Le but est d'estimer les paramétres β_1 et β_2. Pour ce faire, on minimise la quantité :

$$S(\beta_1, \beta_2) = \sum_{i=1}^{n} (y_i - \beta_1 - \beta_2 x_i)^2. \quad (13)$$

Autrement dit, la droite des moindres carrées minimise la somme des carrées des distances verticales des points (x_i, y_i) du nuage à la droite ajustée $y = \hat{\beta}_1 + \hat{\beta}_2 x$[41, 42].

3.4.1.3 Calcul des estimateurs $\hat{\beta}_0$ et $\hat{\beta}_1$

La fonction de deux variables S est une fonction quadratique et sa minimisation ne pose aucun problème.
Les estimateurs des MCO ont pour expressions :

$$\hat{\beta}_1 = \bar{y} - \hat{\beta}_2 \bar{x} \quad (14)$$

avec :

$$\hat{\beta}_2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})y_i}{\sum_{i=1}^{n}(x_i - \bar{x})^2}. \quad (15)$$

On obtient aisément la valeur prédite définie par :

$$\hat{y}_{n+1} = \hat{\beta}_1 + \hat{\beta}_2 x_{n+1} + \hat{\epsilon}_{n+1} \quad (16)$$

Deux types d'erreurs vont entacher notre prévision : la première est due à la non-connaissance de ϵ_{n+1}, la seconde à l'incertitude sur les estimateurs $\hat{\beta}_1$ et $\hat{\beta}_2$.
On va donc exprimer l'erreur de prévision définie par $\hat{\epsilon}_{n+1} = (y_{n+1} - \hat{y}_{n+1})$ qui satisfait les propriétés suivantes[41] :

$$\begin{cases} \mathrm{E}[\hat{\epsilon}_{n+1}] & = 0 \\ \mathrm{Var}(\hat{\epsilon}_{n+1}) & = \sigma^2(1 + \frac{1}{n} + \frac{(x_{n+1}-\bar{x})^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}). \end{cases} \quad (17)$$

Ainci la variance augmente lorsque x_{n+1} s'éloigne du centre de gravité du nuage $\bar{x}$.

3.4.1.4 Coefficient de détermination R^2

On a défini la valeur prédite ou ajustée par $\hat{y}_{n+1} = \hat{\beta}_1 + \hat{\beta}_2 x_{n+1} + \hat{\epsilon}_{n+1}$. Le résidu est alors défini comme la différence entre la valeur observée et la valeur prédite, à savoir $\hat{\epsilon} = Y - \hat{\hat{Y}} = [\hat{\epsilon}_1, \ldots, \hat{\epsilon}_n]^T$ le vecteur des résidus. On définit aussi la somme des carrées résiduelle par :

$$\text{SSE} = \sum_{i=1}^{n} (Y_i - \hat{Y})^2. \quad (18)$$

Pour évaluer la qualité de la prévision, on peut utiliser différents critéres. Dans un premier temps, on définit la variation expliquée par la régression ainsi que la variation totale respectivement par

$$\text{SSR} = \sum_{i=1}^{n} (\hat{Y}_i - \bar{Y})^2 \quad (19)$$

et

$$\text{SCT} = \sum_{i=i}^{n} (Y_i - \bar{Y})^2. \quad (20)$$

Le coefficient de détermination est alors défini par le quotient :

$$R^2 = \frac{\text{SSR}}{\text{SCT}} = \frac{\|\hat{Y}_i - \bar{Y}\|^2}{\|Y_i - \bar{Y}\|^2} = 1 - \frac{\|\epsilon\|^2}{\|Y_i - \bar{Y}\|^2} = 1 - \frac{\text{SSE}}{\text{SCT}}. \quad (21)$$

Le R^2 peut être interprété de manière suivante :
- Si $R^2 = 1$, le modèle explique tout, c'est-à-dire que $y_i = \beta_1 + \beta_2 x_i$ pour tout i : les points de l'échantillons sont parfaitement alignés sur la droite des moindres carrés ;
- Si $R^2 = 0$, cela veut dire que $\sum(\hat{y}_i - \bar{y})^2 = 0$, donc $\hat{y}_i = \bar{y}$ pour tout i . Le modèle de régression linéaire est inadapté puisqu'on ne modélise rien de mieux que la moyenne ;
- Si R^2 est proche de zéro, le modéle de régression linéaire est inadapté, la variable x n'explique pas bien la variable réponse y (du moins pas de façons affine). De façon générale, l'interprétation est la suivante : le modèle de régression linéaire permet d'expliquer $100 \times R^2\%$ de la variance totale des données[42].

3.4.2 Régression linéaire multiple

3.4.2.1 Modèle

Le modèle de régression linéaire multiple est une généralisation du modèle de régression simple lorsque les variables explicatives sont en nombre quelconque. Nous supposons donc que les données collectées suivent le modèle suivant :

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i, \quad i = 1, \ldots, n, \qquad j = 1, \ldots, p \quad (22)$$

où :

- les x_{ij} sont des nombres connus, non aléatoires ;
- les paramétres β_j du modèle sont inconnus, mais non aléatoires ;
- les ϵ_i sont des variables aléatoires inconnues.

Soit de façon matricielle, un modéle de régression linéaire multiple est défini par une équation de la forme :

$$Y = X\beta + \varepsilon \quad (23)$$

où :

- Y est un vecteur de dimension n,
- X est une matrice de taille $n \times p$ connue, appelée matrice du plan d'expérience,
- β est le vecteur de dimension p des paramètres inconnus du modéle,

- ϵ est le vecteur de dimension n des erreurs[43].

$$Y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}, \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}, X = \begin{bmatrix} 1 & X_{11} & X_{12} & \cdot & X_{1n} \\ 1 & X_{21} & X_{22} & \cdot & X_{2n} \\ \vdots & \vdots & \vdots & \vdots & \\ 1 & X_{n1} & X_{n2} & \cdot & X_{nn} \end{bmatrix}. \quad (24)$$

3.4.2.2 Estimation des moindres carrés ordinaires

Commme pour la regression linéaire simple, le critére permet de trouver le «meilleur modèle» permettant de prédire Y à l'aide de combinaison linéaire des variables de X. L'estimateur des MCO $\hat{\beta}$ à pour expression[41] :

$$\hat{\beta} = (X'X)^{-1}X'Y. \quad (25)$$

Il est construit de sorte que l'erreur d'estimation entre $X\hat{\beta}$ et Y soit la plus petite possible au sens $\|.\|^2$:

$$\beta = \arg \min_{\beta \in \mathbb{R}^{p+1}} \|\mathbf{Y} - \mathbf{X}\beta\|^2. \quad (26)$$

Ainsi, $\beta \in \mathbb{R}^{p+1}$ signifie que le vecteur β appartient à l'espace des vecteurs de dimension $(p+1)$,où p est le nombre de variables indépendantes, et que chaque composante de ce vecteur est un nombre réel. En d'autres termes, β est un vecteur dont chaque élément est un nombre réel, et sa dimension est $(p+1)$.

3.4.2.3 Prévision

Un des buts de la régression est de proposer des prédictions pour la variable à expliquer y lorsque nous avons de nouvelles valeurs de x. Soit donc $x'_{n+1} = [x_{n+1,1}, \ldots, x_{n+1,p}]$

une nouvelle valeur pour laquelle nous voudrions prédire y_{n+1}. Cette variable réponse est définie par[41] : $y_{n+1} = x'_{n+1}\beta + \varepsilon_{n+1}$, avec $\mathrm{E}[\epsilon_{n+1}] = 0$, $\mathrm{Var}(\epsilon_{n+1}) = \sigma^2$, et $\mathrm{Cov}(\epsilon_{n+1}, \epsilon_i) = 0$, pour $\forall i \in \{1, \dots n\}$.

La méthode naturelle est de prédire la valeur correspondante grâce au modèle ajusté, soit : $\hat{y}_{n+1} = x'_{n+1}\hat{\beta}$. L'erreur de prévision est à nouveau définie par $\hat{\epsilon}_{n+1} = y_{n+1} - \hat{y}_{n+1} = x'_{n+1}(\beta - \hat{\beta}) + \varepsilon_{n+1}$. Deux types d'erreur vont entacher notre prévision : la première due à l'incertitude sur ϵ_{n+1}, l'autre à l'incertitude inhérente à l'estimation $\hat{\beta}$.

3.4.2.4 Erreur de prévision

L'erreur de prévision $\hat{\epsilon_{n+1}} = (y_{n+1} - \hat{y}_{n+1})$ satisfait les propriétés suivantes[41] :

$$\begin{cases} \mathrm{E}[\hat{\epsilon}_{n+1}] & = 0 \\ \mathrm{Var}(\hat{\epsilon}_{n+1}) & = \sigma^2(1 + x'_{n+1}(X'X)^{-1}x_{n+1}). \end{cases} \quad (27)$$

3.4.2.5 Coefficient de détermination R^2

Le coefficient de détermination R^2, également connu sous le nom de coefficient de détermination multiple (CDM), est couramment utilisé pour évaluer la qualité de l'ajustement d'un modèle de régression linéaire. Comme pour la regression linéaire simple,

$$R^2 = 1 - \frac{\mathrm{SSE}}{\mathrm{SCT}} \quad (28)$$

R^2 peut parfois être trompeur, en particulier lorsque le nombre de variables indépendantes est important par rapport à la taille de l'échantillon. Dans de tels cas, R^2 peut sures-

timer la qualité de l'ajustement et donner une vision optimiste de la performance du modèle. Pour remédier à ce problème, on utilise souvent le coefficient de détermination ajusté (CDA) R_a^2, qui tient compte du nombre de variables dans le modèle ainsi que de la taille de l'échantillon. Le CDA a pour expression[41, 42] :

$$R_a^2 = 1 - \frac{n}{n-p}\frac{\|\hat{\epsilon}\|^2}{\|Y\|^2} = 1 - \frac{n}{n-p}\frac{\text{SSE}}{\text{SCT}} = 1 - \frac{n}{n-p}(1-R^2) \quad (29)$$

R_a^2 prend en considération le risque de surajustement et fournit une évaluation plus équilibrée de l'ajustement du modèle.

En somme, R^2 et R_a^2 sont des mesures importantes pour évaluer la pertinence d'un modèle de régression linéaire, mais il est crucial de les interpréter avec prudence, en particulier lorsque le modèle contient un grand nombre de variables par rapport à la taille de l'échantillon[41].

3.4.3 Régression Ridge, Lasso et ElastiNet

La régression Ridge (RR) et la régression Lasso (RL) sont des variantes régularisées de la régression des moindres carrés, qui appliquent respectivement des pénalités de type L2 et L1 sur le vecteur de coefficients[44]. Considérons le modèle de régression linéaire multiple défini par $Y = X\beta + \varepsilon$.
Dans le contexte de la régression Ridge, l'objectif est de minimiser par rapport au vecteur de coefficients β, un critère formulé comme suit[44, 45] :

$$\text{L}_2 = \sum_{i=1}^{n}(y_i - \mu - x_i^T\beta)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 \quad (30)$$

où n représente le nombre d'observations, p est la dimension du vecteur de coeffi-

cients β, y_i est la valeur observée de la variable dépendante pour la i-ème observation, x_i est le vecteur des prédicteurs pour la i-ème observation, μ est l'intercept, β_j est le j-ème coefficient, et λ est le paramètre de régularisation de Ridge qui varie dans l'intervalle $[0, \infty]$.

À mesure que la valeur de λ évolue, la solution $\beta(\lambda)$ décrit un parcours dans l'espace des coefficients $\mathbb{R}^p$. L'incorporation de la pénalité $\lambda \sum_{j=1}^{p} \beta_j^2$ a pour effet de réduire la variance de l'estimation $\beta(\lambda)$, mais cela peut également introduire un certain biais. Il est important de noter que le terme intercept μ ne figure pas dans la partie de la pénalité quadratique.

En introduisant $\varepsilon_i = y_i - \mu - x_i^T \beta$, la formulation de la régression Ridge peut également être interprétée comme la minimisation de la somme des carrés des résidus $\|\varepsilon\|_2^2$ tout en imposant une contrainte de borne supérieure sur la norme L2 du vecteur de coefficients $\|\beta\|_2$. Cette contrainte conduit au même chemin dans l'espace $\beta(\lambda)$, bien que chaque point le long de ce chemin puisse correspondre à une valeur différente de λ[44].

Le Lasso modifie le critère (30) en utilisant la formulation suivante[44, 45] :

$$\mathrm{L}_1 = \sum_{i=1}^{n} (y_i - \mu - x_i^T \beta)^2 + \lambda \sum_{j=1}^{p} |\beta_j|. \quad (31)$$

Dans cette formulation, le Lasso remplace la pénalité L2 $\|\beta\|_2$ par une pénalité L1 $\|\beta\|_1$. L'avantage principal du Lasso réside dans sa capacité à identifier des solutions parcimonieuses, c'est-à-dire celles où certains, voire la majorité, des coefficients β_j sont nuls. Cette propriété de parcimonie est particulièrement souhaitable pour l'interprétation des résultats[44, 46].

Cependant, le Lasso présente une limitation en imposant une certaine quantité de parcimonie. Il peut y avoir au plus p coefficients β_j non nuls. Cette limitation peut devenir significative lorsque le nombre de prédicteurs p dépasse le nombre d'observations n. De plus, il est souvent évoqué que la parcimonie induite par la pénalité L1

pourrait entraîner une moindre précision par rapport à une pénalité L2. Dans les cas où plusieurs caractéristiques sont corrélées et ont des effets importants sur la réponse, le Lasso a tendance à annuler certaines de ces caractéristiques, voire toutes sauf une. En revanche, la régression Ridge n'effectue pas une telle sélection, mais plutôt tend à «partager» la valeur des coefficients entre les prédicteurs corrélés[44].
La régression ElasticNet est une combinaison linéaire des régression L1 et L2, qui tire parti des avantages des deux méthodes simultanément. Le fait que les variables ne soient pas forcées à devenir nulles, comme c'est le cas avec la régularisation L1, permet de créer des conditions favorables à un effet de groupe présentant une forte corrélation entre les variables[45].
La formule générale de la régularisation ElasticNet est la suivante [45] :

$$\mathrm{L_{EN}} = \sum_{i=1}^{n}(y_i - \mu - x_i^T\beta)^2 + \lambda_1 \sum_{j=1}^{p}|\beta_j| + \lambda_2 \sum_{j=1}^{p}\beta_j^2. \quad (32)$$

Cette méthode est principalement utilisée lorsque le modèle comporte un grand nombre de paramètres, mais il n'est pas certain de leur nécessité ou de la possibilité de les négliger préalablement[45].

3.4.4 Machines à Vecteurs de Support (SVM)

Les Machines à Vecteurs de Support (SVM) ont été évoquées pour la première fois en 1992. Elles constituent un ensemble de méthodes d'apprentissage supervisé utilisées pour la classification et la régression. Elles font partie d'une famille de classificateurs linéaires généralisés. En d'autres termes, SVM est un outil de prédiction pour la classification et la régression qui exploite la théorie de l'apprentissage automatique afin de maximiser la précision prédictive tout en évitant automatiquement le surajustement aux données. Elles peuvent être définies comme des systèmes qui opèrent dans un espace d'hypothèses composé de fonctions linéaires, au sein d'un espace de

caractéristiques de grande dimension. Elles sont élaborées à l'aide d'un algorithme d'apprentissage basé sur la théorie de l'optimisation, lequel met en œuvre un biais d'apprentissage dérivé de la théorie statistique de l'apprentissage[47].

3.4.4.1 Notion de base : Hyperplan, marge et support vecteur

Hyperplan

Pour deux classes d'exemples donnés, l'objectif des SVM consiste à découvrir un moyen de les différencier en trouvant un séparateur qui maximise la marge entre ces ensembles. L'outil utilisé par les SVM pour accomplir cela est un modèle de classification linéaire connu sous le nom d'hyperplan[48].

Dans le schéma présenté à la figure 3.1, nous identifions un hyperplan spécifique qui agit comme une barrière entre les deux groupes de points distincts. Ce faisant, les SVM visent à créer un espace aussi large que possible entre ces deux groupes, permettant ainsi une meilleure séparation et classification des données[48].
Le schéma ci-dessous montre un hyperplan qui sépare les ensembles de points de deux classes[48].

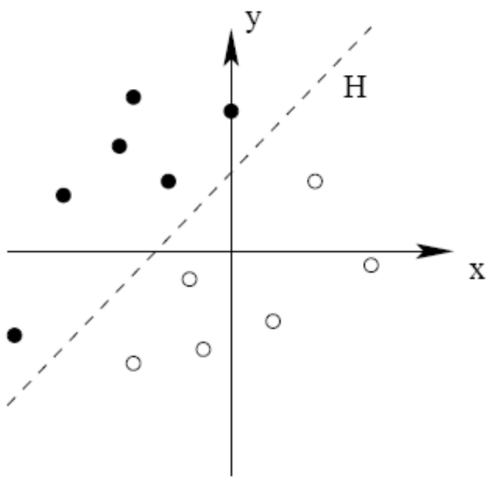

FIGURE 3.1 – Hyperplan qui sépare deux classes de points.

Support Vecteur

Les points les plus proches, qui seuls sont utilisés pour la détermination de l'hyperplan, sont appelés vecteurs de support.
Le schéma de la figure (3.2) montre un hyperplan (celui du milieu) optimal avec une marge maximale.

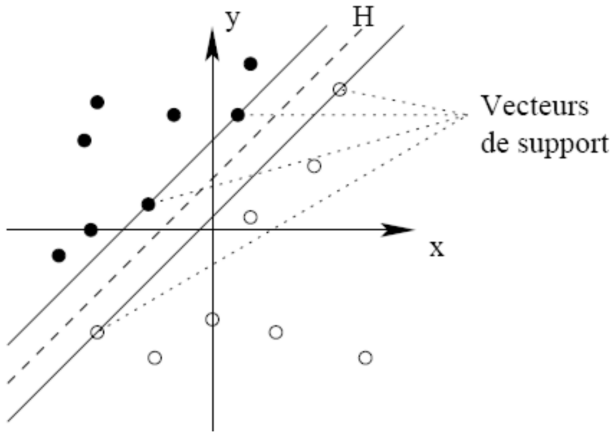

FIGURE 3.2 – Support Vecteur

Marge

Il est clair qu'il existe plusieurs façons possibles de placer un plan qui sépare les points, mais ce qui rend les SVM particulièrement intéressants, c'est qu'elles cherchent le meilleur plan possible. Pour cela, nous voulons trouver un plan entre les points des deux groupes, comme s'il était au milieu d'eux. Cela équivaut à chercher le plan le plus sûr. Si un point n'est pas parfaitement expliqué, de petites variations ne le feront pas changer de groupe si sa distance par rapport au plan est grande. En d'autres termes, nous cherchons un plan où la plus petite distance entre les exemples d'apprentissage et le plan est la plus grande possible. Cette distance est appelée « marge » entre le plan et les exemples. L'objectif est de trouver le plan qui maximise cette marge. Étant donné que nous cherchons à maximiser la marge, nous parlons de plans avec une grande marge.
De manière intuitive, une marge plus large offre une meilleure sécurité lors de la clas-

sification de nouveaux exemples. De plus, en trouvant le classificateur qui fonctionne le mieux avec les données d'apprentissage, il est évident qu'il fonctionnera également de manière optimale pour classer de nouveaux exemples. Dans le schéma ci-dessous, la partie de droite illustre qu'avec un hyperplan optimal, un nouvel exemple est correctement classé même s'il se trouve dans la marge. En revanche, dans la partie de gauche où la marge est plus étroite, cet exemple serait mal classé[48].

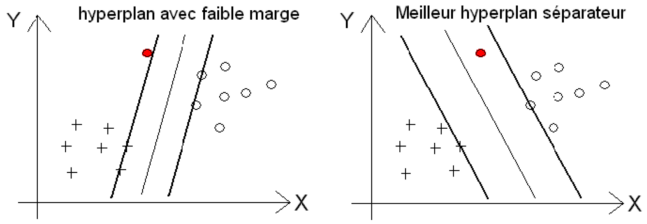

FIGURE 3.3 – SVM exemples séparables et non séparables.

3.4.4.2 Régression par Machines à Vecteurs de Support (SVR)

La régression par Machines à Vecteurs de Support (SVR) vise à déterminer la fonction $f(x)$ qui présente au maximum une déviation ϵ par rapport aux exemples d'apprentissage (xi, yi) pour $i = 1, \cdots, N$, tout en cherchant à minimiser la pente de la fonction. Cette approche implique de ne pas tenir compte des erreurs inférieures à ϵ, tout en limitant celles qui dépassent ϵ. L'objectif est d'optimiser la planéité de la fonction, ce qui se traduit par une réduction de la complexité du modèle, influant positivement sur ses performances en généralisation. En effet, selon la théorie de l'apprentissage, l'erreur de généralisation peut être bornée par une combinaison de deux termes : l'un dépendant de la complexité du modèle et l'autre dépendant de l'erreur sur les données d'apprentissage. Les méthodes SVM reposent sur la régulation de la complexité du modèle pendant le processus d'apprentissage[49]. Dans la suite, nous allons présenter l'algorithme de SVR pour un modèle linéaire, puis pour le cas non-linéaire.

3.4.4.2.1 SVR cas linéaire

Nous commencerons d'abord d'écrire une fonction f prise sous la forme :

$$f(x) = \langle w, x \rangle + b \quad (33)$$

avec $\langle ., . \rangle$ représente le produit scalaire dans $\mathbb{R}^p$, $x \in \mathbb{R}^p$ le vecteur d'entrée et $w \in \mathbb{R}^p$ le vecteur des paramétres (ou poids) et b une constante à déterminer. Afin de résoudre le probléme de régression, on cherche à minimiser

$$\min \frac{1}{2}\|w\|^2 \quad (34)$$

sous la contrainte

$$\begin{cases} y_i - \langle w, x \rangle - b \leq \epsilon \\ y_i - \langle w, x \rangle - b \geq -\epsilon. \end{cases} \quad (35)$$

L'objectif principal de SVR est de garantir une fonction très précise sans aucune erreur, et la minimisation de l'erreur est imposée strictement sous forme de contraintes. Cela suppose qu'une fonction linéaire parfaite qui approxime tous les exemples avec une précision ϵ est possible, ce qui n'est pas toujours réaliste dans la pratique. Dans des situations où il y a beaucoup de bruit ou des données aberrantes, il est parfois plus important de permettre certaines erreurs. C'est là qu'intervient le concept de "marge souple" (soft margin). Il consiste à introduire des variables de relâchement (slack variables) ξ_i et ξ_i^* pour rendre les contraintes du problème d'optimisation plus flexibles, ce qui signifie qu'il est acceptable d'avoir quelques erreurs tout en cherchant toujours à minimiser l'erreur dans la mesure du possible. Ainsi la régression SVR est formulée comme une minimisation de la fonctionnelle suivante[49, 50, 51] :

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{N} (\xi_i + \xi_i^*) \quad (36)$$

sous la contrainte

$$\begin{cases} y_i - \langle w, x_i \rangle - b & \leq \varepsilon + \xi_i \\ \langle w, x_i \rangle + b - y_i & \leq \varepsilon + \xi_{i^*} \\ \xi_i, \xi_{i^*} & \geq 0, \quad i = 1, \ldots, N \end{cases} \tag{37}$$

où

ξ_i et ξ_i^* représentent respectivement les erreurs positive et négative. La constante $C > 0$ est un hyperparamètre permettant de régler le compromis entre la quantité d'erreur autorisée et la platitude de la fonction f. Cette formulation du problème revient à utiliser une fonction d'erreur $|\xi|_\epsilon$ appelée ϵ-insensible (ϵ-insensitive) de la forme[49, 51] :

$$|y - f(x)|_\epsilon = \begin{cases} 0, & \text{pour } |y - f(x)| \leq \epsilon \\ |y - f(x)| - \epsilon, & \text{pour } |y - f(x)| > \epsilon \end{cases} \tag{38}$$

où ϵ est un paramètre de précision de la régression, $f(x)$ la fonction de régression recherchée et y la valeur prise en x_i.

La résolution du problème implique la minimisation du Lagrangien L défini comme suit[49] :

$$\begin{aligned} L = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) - \sum_{i=1}^{N}(\eta_i\xi_i + \eta_i^*\xi_i^*) \\ - \sum_{i=1}^{N}\alpha_i(\epsilon + \xi_i - y_i + \langle w, x_i \rangle + b) \\ - \sum_{i=1}^{N}\alpha_i^*(\epsilon + \xi_i^* + y_i - \langle w, x_i \rangle - b), \end{aligned} \tag{39}$$

où les $\eta_i, \eta_i^*, \alpha_i, \alpha_i^* > 0$ désignent les multiplicateurs de Langrange.

La minimisation de cette fonction par rapport à w, b, ξ_i et ξ_i^* conduit aux équations :

$$w = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) x_i \quad (40)$$

qui représente le poids du modèle

$$\sum_{i=1}^{N} (\alpha_i - \alpha_i^*) = 0 \quad (41)$$

et

$$\alpha_i^* + \eta_i^* = C, \quad \alpha_i + \eta_i = C. \quad (42)$$

La fonction s'écrit :

$$f(x) = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b. \quad (43)$$

La valeur de biais b peut être calculée en appliquant les conditions de Karush-Kuhn-Tucker (KKT), qui impliquent que les produits des variables duales et des contraintes sont nuls à l'optimum[49].

$$\alpha_i (\epsilon + \xi_i - y_i + \langle w, x_i \rangle + b) = 0 \quad (44)$$

$$\alpha_i^* (\epsilon + \xi_i^* + y_i - \langle w, x_i \rangle - b) = 0 \quad (45)$$

$$\eta_i \xi_i (C - \alpha_i) \xi_i = 0 \quad (46)$$

$$\eta_i^* \xi_i^* (C - \alpha_i^*) \xi_i^* = 0 \quad (47)$$

b peut être determiné par (44) sur un point particulier pour lequel x_i, y_i, ξ_i sont connus et α_i peut être déterminé à partir de l'algorithme de maximisation du Lagrangien dual $\mathrm{L_D}$. L'injection des résultats (41), (42) et (43) dans (39) donne le Lagrangien dual qui doit être maximiser :

$$\mathrm{L_D} = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \langle x_i, x_j \rangle \quad (48)$$

$$-\epsilon \sum_{i=1}^{N}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i - \alpha_i^*)$$

sous la contrainte

$$\sum_{i=1}^{N}(\alpha_i - \alpha_i^*) = 0, \quad \alpha_i, \alpha_i^* \in [0, C]. \quad (49)$$

3.4.4.2.2 SVR cas non linéaire

L'adaptation des SVM au contexte non linéaire est assez directe et implique la transformation des données dans un espace de dimensions plus élevées, où le problème se simplifie en un problème linéaire[49]. Après l'application d'un algorithme similaire à celui des SVM linéaires dans cet espace de caractéristiques, on peut trouver un hyperplan de séparation linéaire. Les fonctions de noyau couramment utilisées incluent les fonctions linéaires, polynomiales, à base radiale et sigmoïde définie ainsi[49, 52, 53] :

$$\text{linéaire} \quad \mathrm{k}(x, x') = \langle x, x' \rangle \quad (50)$$

$$\text{polynomial} \quad \mathrm{k}(x, x') = (\gamma \langle x, x' \rangle + c)^d \quad (51)$$

$$\text{sigmoïde} \quad \mathrm{k}(x, x') = \tanh(\gamma \langle x, x' \rangle + c) \quad (52)$$

où

x et x' représentent deux vecteurs dans l'espace des caractéristiques,

$\langle x, x' \rangle$ est le produit scalaire entre x et x',

γ est un paramètre qui contrôle l'influence du produit scalaire,

d est un entier qui représente le degré du polynôme,

c est une constante,

tanh est la fonction tangente hyperbolique. Le problème non-linéaire peut donc être

formulé ainsi :

$$\max \mathrm{L_D} = -2(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\mathrm{k}(x_i, x_j) \quad (53)$$
$$-\epsilon \sum_{i=1}^{N}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i - \alpha_i^*)$$

sous la contrainte

$$\begin{cases} \sum_{i=1}^{N}(\alpha_i - \alpha_i^*) & = 0 \\ 0 \leq \alpha_i, \alpha_i^* \leq \mathrm{c} & (i = 1, 2, \cdots, \mathrm{L}) \end{cases} \quad (54)$$

Où α_i et α_i^* sont des vecteurs de multiplicateurs de Lagrange, et $\mathrm{k}(x_i, x_j) = \langle \Phi(x_i), \Phi(x) \rangle$ est la fonction de noyau. La valeur du noyau est le produit scalaire des deux vecteurs x_i et x_j dans l'espace des caractéristiques. Dans cette étude, la fonction de noyau gaussien à base radiale est utilisée comme fonction de noyau, et elle est écrite comme suit[54] :

$$\mathrm{k}(x_i, x_j) = \exp(-\frac{\|x - x_i\|^2}{\sigma^2}). \quad (55)$$

De même, les poids sont donnés par :

$$\mathrm{w} = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)\Phi(x_i); \quad (56)$$

et le modèle par[49, 54] :

$$f(x) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)k\langle x_i, x \rangle + b. \quad (57)$$

3.4.5 Les arbres et forêts aléatoire (Random Forest)

Les Random Forests (Forêts Aléatoires), créées par Breiman en 2001[55], sont initialement nées de l'idée de combiner plusieurs arbres de décision de style CART

(Classification and Regression Trees) à l'aide de la méthode de bagging (Bootstrap Aggregating)[56]. Leur évolution initiale a été façonnée par diverses approches, dont la méthode du sous-espace aléatoire, la sélection aléatoire de points de division, ainsi que la sélection de caractéristiques.[56, 57].

Ce qui a grandement contribué à la popularité des forêts aléatoires, c'est leur capacité à être appliquées à une vaste gamme de problèmes de prédiction tout en nécessitant peu de paramètres à ajuster. En plus d'être simples à utiliser, cette méthode est généralement reconnue pour sa précision et sa capacité à traiter des échantillons de petite taille ainsi que des espaces de caractéristiques de haute dimension. De plus, elle peut être facilement parallélisée, ce qui lui confère le potentiel de traiter efficacement de vastes systèmes du monde réel[57].

Au fil des années depuis leur création, les Random Forests sont devenues un cadre de modèles complet et ont connu un succès considérable dans de nombreux domaines.[56, 57].

Bien que largement utilisées en pratique, les fondements mathématiques du succès des Random Forests demeurent mal compris. Les premiers travaux théoriques de Breiman (2004)[58] étaient principalement basés sur l'intuition et des heuristiques mathématiques, et leur formalisation rigoureuse n'a été entreprise que récemment (Biau, 2016)[56, 57].
Dans la suite, nous allons entreprendre la présentation des algorithmes des arbres de décision et des forêts de décision.

3.4.5.1 Les arbres de décision

Un arbre de décision est un modèle prédictif exprimé sous la forme d'une partition récursive de l'espace des variables explicatives en sous-espaces qui servent de base

pour les prédictions. Les arbres de décision sont fréquemment employés pour des tâches de classification. La classification vise à attribuer un objet ou une instance à un ensemble préalablement défini de classes en fonction des valeurs de leurs attributs (caractéristiques)[59].

Par exemple, l'arbre de la figure 3.4 décide une réponse booléenne (classification dans l'ensemble oui, non) en fonction des valeurs discrètes des attributs difficile, durée, motivation, surprenant[60].

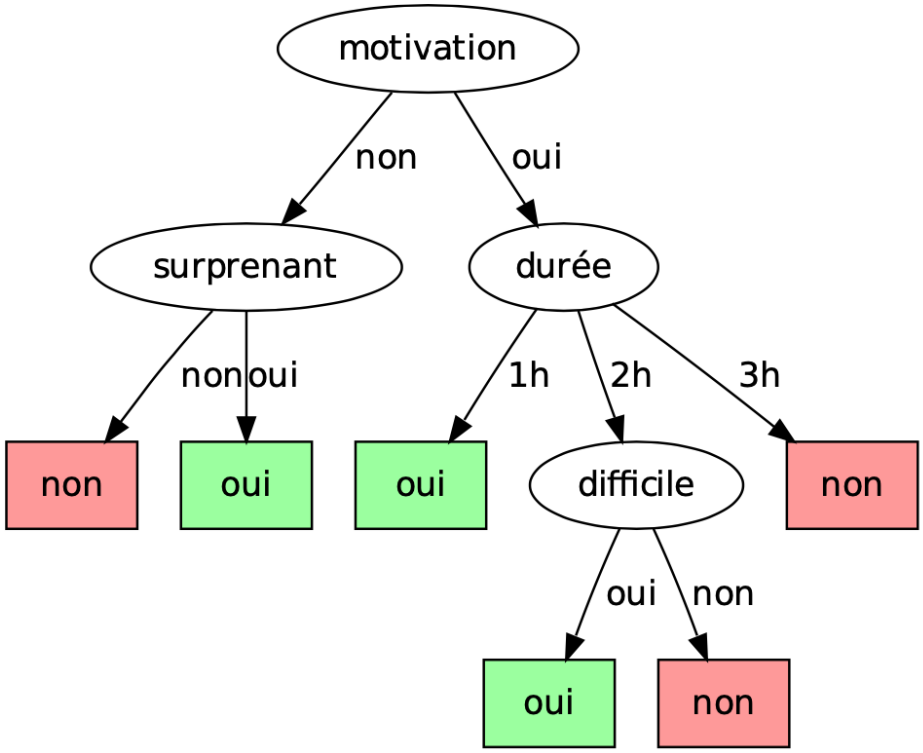

FIGURE 3.4 – Excemple d'arbre de décision

3.4.5.1.1 Arbres de régression (CART)

CART est un type d'arbre de décision qui peut être appliqué à la fois à des problèmes de classification et de régression. Cet algorithme de faible complexité utilise un arbre binaire pour sa construction, ce qui signifie que chaque nœud de l'arbre

CART se divise en deux branches. La sélection de la meilleure caractéristique pour le partitionnement au niveau d'un nœud implique l'évaluation de toutes les variables d'entrée[61]. Soit x la matrice d'entrée contenant p caractéristiques, définie par[62] :

$$x = \left(x^1, x^2, \cdots, x^p, \cdots, x^P\right) = \begin{pmatrix} {x_1}^1 & {x_1}^2 & \dots & {x_1}^p & \dots & {x_1}^P \\ {x_2}^1 & \ddots & & & & \vdots \\ \vdots & & \ddots & & & \vdots \\ {x_n}^1 & & & & & \\ \vdots & & & & & \\ {x_N}^1 & {x_N}^2 & \dots & {x_N}^p & \dots & {x_N}^P \end{pmatrix} \quad (58)$$

où P est le nombre de variables d'entrée x^p, également appelées caractéristiques ; N est le nombre d'échantillons ; et chaque ligne de l'espace d'entrée est associée à un vecteur d'entrée :

$$x_n = \left({x_n}^1, {x_n}^2, \cdots, {x_n}^p \cdots, {x_n}^P\right). \quad (59)$$

Les variables de sortie correspondantes peuvent être définies comme suit :

$$Y = \left(y_1, y_2, \cdots, y_n, \cdots, y_N\right)^T \quad (60)$$

où y_n est la valeur de sortie associée au vecteur d'entrée x_n ; dans le problème de régression, l'espace d'entrée est partitionné en M sous-espaces $\{R_1, \dots, R_m, \dots, R_M\}$, et dans chaque sous-espace R_m, il y a une valeur de sortie c_m.

La méthode d'apprentissage automatique supervisé doit être formée en utilisant les

données de mesures passées, qui forment un ensemble de données d'entraînement[62] :

$$T = (X, Y) = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n), \ldots, (x_N, y_N)\}^T \quad (61)$$

où (x_n, y_n) est une paire constituée d'un vecteur d'entrée et d'une valeur de sortie. Le but de l'arbre de décision est de trouver une fonction de mapping à partir de l'ensemble de données de formation :

$$\phi : X \to Y \quad (62)$$

et les prédictions obtenues à partir de la fonction de mapping peuvent être écrites comme suit[62] :

$$\hat{Y} = \phi(X). \quad (63)$$

Le modèle de régression CART peut être formulé comme suit :

$$\phi(X) = \sum_{m=1}^{M} c_m I(X \in R_m) \quad (64)$$

où $I(\cdot)$ est une fonction indicatrice définie comme suit :

$$\begin{cases} 1, & \text{si } X \in R_m \\ 0, & \text{si } X \notin R_m. \end{cases} \quad (65)$$

Dans le contexte de l'analyse statistique, X et Y sont des variables aléatoires. L'erreur de prédiction attendue peut être exprimée comme suit :

$$\mathrm{Err}(\phi_T) = \mathrm{E}_{X,Y}\{L[Y, \phi_T(X)]\} \approx \frac{1}{N} \sum_{n=1}^{N} L[y_n, \phi_T(x_n)] \quad (66)$$

où ϕ_T représente le modèle d'arbre de décision CART appris à partir de l'ensemble de données T, et $L(\cdot)$ est la fonction de perte. Pour un problème de régression, la

fonction de perte peut être exprimée comme suit[62] :

$$L[Y, \phi(X)] = \sum_{n=1}^{N} (y_n - \phi(X_n))^2 = \sum_{n=1}^{N} (y_n - \hat{y}_n)^2. \quad (67)$$

En minimisant la fonction de perte attendue ainsi que la mesure d'impureté pour le modèle de régression, la valeur de sortie c_m dans le sous-espace partitionné peut être exprimée comme suit[62] :

$$\hat{c}_m = \arg \min_{x_i \in R_m} E\{L[Y, \phi(X)]$$

$$= \frac{1}{N} \sum_{n=1}^{N} y_n. \quad (68)$$

La construction d'un arbre de décision vise à trouver une partition optimale, maximisant la pureté de chaque sous-ensemble en regroupant des échantillons de la même classe. Le processus se déroule de manière descendante, avec un nœud racine divisant l'ensemble de données T en deux sous-ensembles via une division binaire, typique de la méthode CART. À chaque nœud interne t, les divisions s_t segmentent l'ensemble de données T_t en deux sous-ensembles plus petits. Ce processus de partitionnement se répète jusqu'à ce qu'un critère d'arrêt soit satisfait. Ce critère peut être l'atteinte d'une profondeur d'arbre prédéfinie ou la pureté maximale des échantillons dans chaque partition, empêchant une réduction ultérieure de l'impureté. Une fois le partitionnement terminé, un nœud devient une feuille s'il ne peut plus être divisé. Chaque nœud feuille est associé à une valeur de sortie estimée $\hat{y}_t$, représentant la classe la plus probable. La phase cruciale de la construction d'un arbre de décision réside dans le choix de la règle de division s_t. Cela implique la sélection des caractéristiques x^p et des points de division a[62]. Dans la méthode CART pour la régression, l'espace d'entrée sur chaque nœud t est binaire, générant deux sous-espaces, R_1 et R_2 définis par[61, 62] :

$$R_1(p, a) = \{x \mid x^p \leq a\} \quad \text{et} \quad R_2(p, a) = \{x \mid x^p > a\} \quad (69)$$

où x^p désigne la p-ème variable d'entrée et a est le seuil de partitionnement. La caractéristique optimale et le point de division sont déterminés en minimisant la fonction suivante[61, 62, 63] :

$$\min_{p,a}\left[\min_{c_1}\sum_{x_n\in R_1(p,a)}(y_n-c_1)^2+\min_{c_2}\sum_{x_n\in R_2(p,a)}(y_n-c_2)^2\right]. \quad (70)$$

Les valeurs de sortie pour le nœud divisé binaire sont déterminées comme suit[62] :

$$\hat{c_1}=\frac{1}{N_m}\sum_{x_n\in R_1(p,a)}y_n \quad \text{et} \quad \hat{c_2}=\frac{1}{N_m}\sum_{x_n\in R_2(p,a)}y_n \quad (71)$$

où les valeurs $\hat{c_1}$ et $\hat{c_2}$ représentent les moyennes des valeurs cibles y_n dans les sous-ensembles R_1 et R_2 respectivement, où N_m représente le nombre total d'échantillons dans le nœud divisé.
Le processus de division binaire se poursuit jusqu'à ce que le critère d'arrêt soit satisfait, et l'espace d'entrée est ainsi divisé de manière optimale et gourmande (c'est-à-dire en sélectionnant les divisions maximisant un critère à chaque étape) en $\{R_1,\ldots,R_m,\ldots,R_M\}$. Les résultats de sortie finaux pour l'arbre CART construit peuvent être calculés à l'aide de l'équation (64), laquelle peut être reformulée comme suit :

$$\hat{Y}=\sum_{m=1}^{M}c_m\mathbb{I}(X\in R_m). \quad (72)$$

3.4.5.2 Forêt aléatoire

Les arbres de décision sont utiles en apprentissage automatique, mais ils présentent deux problèmes. Tout d'abord, bien qu'ils aient généralement une faible prédiction biaisée, leur variance de prédiction peut être élevée car ils sont sensibles aux petites perturbations dans l'ensemble d'entraînement. Deuxièmement, même si les règles de division dans chaque nœud sont optimales, l'approche gourmande ne garantit pas que

l'arbre de décision global sera optimal. Les méthodes d'ensemble peuvent atténuer ces problèmes en formant plusieurs arbres simultanément et en transformant plusieurs apprenants faibles en un apprenant fort. La forêt aléatoire est l'une de ces méthodes basée sur l'arbre de décision CART et la méthode de bagging. Le bagging consiste à construire plusieurs arbres de décision de manière indépendante, puis à agréger les prédictions en moyennant pour réduire la variance de prédiction. Contrairement à l'approche séquentielle du boosting, où les poids des résultats précédents influent sur les poids actuels, le bagging construit les arbres de manière parallèle, exploitant ainsi efficacement les capacités des ordinateurs modernes.

La forêt aléatoire est un ensemble d'arbres CART, et les randomisations interviennent dans deux aspects de l'algorithme. Dans la méthode de la forêt aléatoire, on sélectionne de manière aléatoire l'ensemble d'entraînement $T_b(b = 1, \ldots, B)$ à partir de l'ensemble d'entraînement total T avec remplacement (c'est-à-dire, échantillonnage bootstrap) pour entraîner chaque arbre CART. Les données laissées de côté lors de ce processus de sélection aléatoire sont appelées échantillons "out-of-bag". Lors de la construction de chaque arbre CART, la méthode de la forêt aléatoire sélectionne aléatoirement M caractéristiques ou variables d'entrée parmi les P caractéristiques ($M < P$). La division optimale pour chaque arbre CART est calculée en fonction de T_b et des P caractéristiques sélectionnées[62]. Les arbres de l'ensemble obtenu peuvent être notés comme suit :

$$\{\phi_{T_b,m} \quad \text{pour} \quad b = 1, \ldots, B\}.$$

Dans la méthode de régression, les résultats finaux sont obtenus par un vote majoritaire[62, 64] :

$$\hat{Y} = \phi_{T,P}(X) = \frac{1}{B}\sum_{b=1}^{B} \phi_{T_b,m}(X). \quad (73)$$

3.4.6 Prédiction de la probabilité de survie globale

3.4.6.1 Modélisation de la probabilité de survie

En analyse de survie, le temps jusqu'à l'événement (décès, défaillance) t est généralement modélisé comme une variable aléatoire, qui suit une certaine distribution de densité de probabilité $p(t)$. La densité peut être caractérisée par la fonction de survie définie comme :

$$S(t) = \Pr(T > t) = \int_t^\infty p(x)dx \text{ pour } t > 0. \quad (74)$$

La fonction de survie capture la probabilité que l'événement ne se produise pas avant le temps t. Un concept étroitement lié à la fonction de survie est la fonction de risque $h(t)$ définie comme :

$$h(t) = \lim_{\Delta t \to 0} \frac{\Pr(t < T < t + \Delta t | T > t)}{\Delta t} = \frac{p(t)}{S(t)} \quad (75)$$

qui mesure le taux d'événements au temps t conditionné par la survie jusqu'à t. On peut ensuite montrer que :

$$S(t) = e^{-\int_0^t h(\tau)d\tau} \quad (76)$$

où $h(\tau)$ est la fonction de risque.

3.4.6.2 Modèle de régression de Cox

Le modèle de régression de Cox, souvent utilisé dans la recherche médicale, est une méthode statistique qui vise à prédire la durée de survie des patients en évaluant l'impact de différentes caractéristiques sur le taux de risque de décès. Il est considéré comme un exemple de modèle semi-paramétrique, ce qui signifie qu'il ne fait pas d'hypothèses strictes sur la distribution des temps de survie. La fonction de risque $h(t)$ dans le modèle de Cox est utilisée pour exprimer la probabilité de décès au temps t. La formulation générale de la fonction de risque est donnée par[65] :

$$h(t) = h_0(t)\exp\sum_{i=1}^{p}\beta_i x_i \quad (77)$$

où

$h(t)$ est la fonction de hasard représentant le taux de risque instantané au temps t,

$h_o(t)$ est la fonction de hasard de base, qui est le risque de base au temps t lorsque toutes les covariables sont égales à zéro,

β_i sont les coefficients de régression de Cox associés aux covariables x_i,

x_i sont les valeurs des covariables,

p est le nombre de covariables.

3.4.6.2.1 Régression de Cox à risque proportionnel

Le modèle de Cox, également appelé modèle de régression de Cox, est caractérisé par l'hypothèse des risques proportionnels. Cette hypothèse est cruciale car elle stipule que le rapport des risques instantanés entre deux individus, x_i^* et x_i, reste constant au fil du temps[66]. En d'autres termes, le modèle suppose que le rapport des risques entre deux individus ne dépend pas du temps, ce qui signifie que le risque pour un individu donné reste proportionnel au risque pour tout autre individu. Cette propriété simplifie l'analyse en permettant une interprétation constante des effets des covariables sur le

risque de l'événement étudié, indépendamment de la durée de l'observation. Ainsi, l'hypothèse des risques proportionnels est une fondation importante du modèle de Cox dans le domaine de la survie et de l'analyse des temps jusqu'à l'événement[1]. Soient $x^* = (x_1^*, x_2^*, \ldots, x_p^*)$ et $x = (x_1, x_2, \ldots, x_p)$ les covariables de deux individus. Le rapport de risque est donné comme suit :

$$\frac{h(t \mid x_i^*, t)}{h(t \mid x_i, t)} = \frac{h_0(t)\exp(\hat{\beta} \cdot x_{i,t}^*)}{h_0(t)\exp(\hat{\beta} \cdot x_{i,t})} = \exp(\hat{\beta} \cdot (x_{i,t}^* - x_{i,t})) \quad (78)$$

où $\hat{\beta}$ est l'estimation du coefficient de régression pour la covariable x_i[1].

3.4.6.3 Indice de concordance IC

L'indice de concordance IC est une mesure de performance couramment utilisée pour les modèles de survie. Fondamentalement, c'est la fraction de toutes les paires de patients dont les prédictions sont correctement ordonnées parmi les paires qui peuvent être ordonnées. En d'autres termes, il quantifie la capacité du modèle à prédire avec précision l'ordre des événements pour les paires d'individus comparables. Formellement, le IC est donné par :

$$IC = \frac{1}{|P|} \sum_{(i,j) \in P} \mathbb{I}(F(x_i) < F(x_j))$$

$$= \frac{1}{|P|} \sum_{i \in E} \sum_{j : t_j > t_i} \mathbb{I}(F(x_i) < F(x_j)) \quad (79)$$

où P est l'ensemble des paires validement ordonnables, $t_i < t_j$; $|P|$ est le nombre de paires dans P ; $F(x)$ est la prédiction du temps de survie ; $\mathbb{I}$ est la fonction indicatrice indiquant si la condition entre parenthèses est satisfaite ou non. Dans le contexte du modèle de Cox à risque proportionnel, le temps de survie prédit peut être représenté de manière équivalente par le logarithme négatif du risque relatif. L'indice IC estime la probabilité que l'ordre des prédictions d'une paire de patients comparables soit

cohérent avec leurs informations de survie observées.

Chapitre 4

Analyse statistique des données

4.1 Description des données

Le jeu de données utilisé dans notre étude comprend des mesures génétiques pour un groupe de patients atteints de cancer du sein, ainsi qu'une variable indiquant la durée de vie avant le décès, nommée days_to_death. L'objectif de ce mémoire est de développer un modèle de prédiction de la durée de survie des patients atteints de cancer du sein en utilisant deux ensembles de données distincts : un ensemble de découverte clinique et génétique, et un ensemble de validation clinique et génétique. L'ensemble de découverte clinique se compose de 997 patients atteints de cancer du sein, chacun est caractérisé par des données sur l'âge initial au diagnostic pathologique, le stade de la tumeur, les traitements administrés, l'état vital, la taille de la tumeur et le temps jusqu'au décès. L'ensemble de découverte génétique contient le même nombre de patients que l'ensemble de découverte clinique, et comprend également l'expression génique mesurée pour 24 925 gènes. Les données de validation clinique et génétique sont utilisées pour évaluer la performance du modèle de prédiction et sont distinctes de l'ensemble de découverte clinique et génétique, comprenant un groupe

de patients indépendants. La découverte de gènes utilisables pour prédire le pronostic en termes de durée de survie chez les patients atteints de cancer est essentielle pour améliorer les options de traitement et faire progresser notre compréhension des bases moléculaires de la croissance tumorale. Cependant, l'hétérogénéité des échantillons de patients constitue l'un des défis majeurs qui rendent difficile l'identification de gènes pronostiques et la prédiction des résultats du cancer. Les données initialement récupérées étaient structurées sous forme de dictionnaire. La première transformation en un dataframe présente les identifiants des patients en lignes et les gènes en colonnes. Une opération de transposition a été effectuée pour obtenir la disposition correcte du dataframe, produisant le tableau 4.1 :

	473	645218	494470	...	54862	57549	149647	days_to_death
MB_0002	3,32393	2,65618	2,75461	...	2,87343	2,98890	2,72346	1484
MB_0008	3,17794	2,68623	2,70721	...	2,87258	2,86246	2,59231	1241
MB_0010	3,23047	2,73777	2,69004	...	2,80483	3,10419	2,72564	234
MB_0035	3,40778	2,66999	2,66431	...	2,90272	3,01533	2,57227	1088
MB_0036	3,37776	2,65210	3,09730	...	2,84853	2,94067	2,59523	2314
MB_0050	3,38419	2,64250	3,02683	...	2,83563	3,19891	2,68107	1784
MB_0059	3,28197	2,66845	2,64462	...	2,88160	3,02224	2,76386	1718
MB_0060	3,22127	2,64982	3,08717	...	2,90174	3,00281	2,61676	1846
MB_0066	3,21344	2,65065	3,02908	...	2,91755	2,90534	2,63725	1543
MB_0101	3,26782	2,67301	2,78095	...	2,90261	3,04610	2,65785	2713

TABLE 4.1 – Données de patients atteints de cancer.

4.2 Analyse exploratoire

Les statistiques fournies nous guident à travers divers aspects de ces données, soulignant la gamme de situations que vivent ces patients.

En moyenne, les patients atteints du cancer du sein dans l'échantillon ont une durée de survie d'environ 2877 jours, soit environ 8 ans. Cela offre une perspective

Duréé de survie (jours)	
Moyenne (jours)	2877
Écart type	1737
Minimum (jours)	8
Maximum (jours)	8220
Médiane (jours)	2578
Premier quartile (Q1)	1439
Troisième quartile (Q3)	4364

TABLE 4.2 – Statistiques sur la Durée de Survie (days_to_death) pour le Cancer du Sein

générale de la survie pour ce groupe de patients. L'écart type de 1737 jours indique une grande variabilité dans la durée de survie. Cela suggère que les expériences des patients peuvent différer considérablement, probablement en raison de facteurs tels que le stade du cancer, le type de traitement et d'autres variables individuelles. L'intervalle de survie allant de 8 à 8220 jours met en évidence la diversité des expériences des patients atteints du cancer du sein. Certains patients ont des survies relativement courtes (8 jours), tandis que d'autres vivent beaucoup plus longtemps (8220 jours), illustrant la complexité de cette maladie. Cependant, la médiane de 2578 jours indique également que de nombreux patients font face à une période plus longue, offrant un certain espoir et une qualité de vie malgré le diagnostic. Les quartiles fournissent des informations sur la répartition des données. Le premier quartile (Q1) à 1439 jours et le troisième quartile (Q3) à 4364 jours montrent comment la majorité des patients se situent dans ces intervalles, mettant en évidence la variabilité mais aussi les tendances centrales.

L'histogramme de la durée de survie offre un aperçu détaillé de la distribution des données, montrant la variabilité, la tendance centrale et la diversité des expériences des patients atteints du cancer du sein dans notre échantillon. Dans la Figure 4.1, on observe que les patients se trouvant dans les intervalles [500, 2000] jours et [3500, 5500] jours ont une durée de survie beaucoup plus longue que les patients situés dans les intervalles [0, 500] jours, [2500, 3000] jours et [5500, 8220] jours.

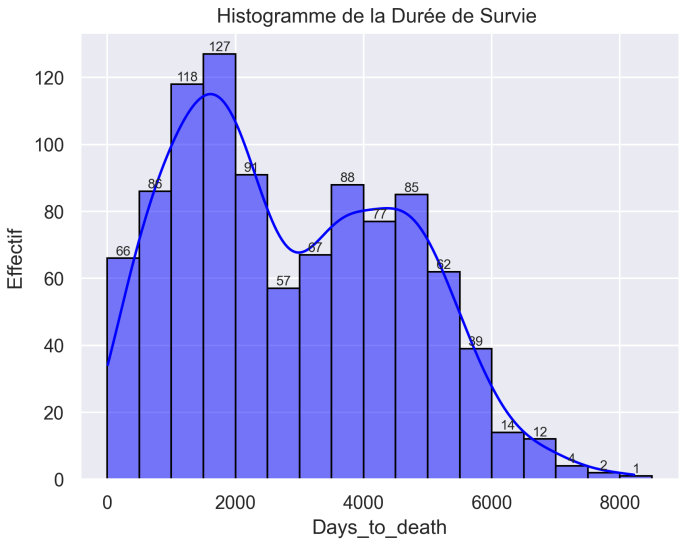

FIGURE 4.1 – Distribution de *days_to_death*

4.2.1 Modèle univarié : Association entre un gène et la survie

Le modèle de régression de Cox a été mis en place pour évaluer l'impact des différents gènes sur la survie. Comme mentionné dans le chapitre 3, la régression de Cox est une méthode statistique visant à prédire la durée de survie des patients en évaluant l'impact de différentes caractéristiques. L'utilisation du modèle de régression donne les résultats du tableau 4.3.

covariate	coef	exp(coef)	se(coef)	coef lower 95%	coef upper 95%	exp(coef) lower 95%	exp(coef) upper 95%	cmp to	z	p	-log2(p)
121260	3,66	38,74	0,71	2,26	5,05	9,6	156,29	0	5,14	2×10^{-7}	21,78
55719	-3,26	0,04	0,91	-5,05	-1,48	0,01	0,23	0	-3,58	3×10^{-4}	11,51
80042	3,94	51,28	1,05	1,87	6	6,51	404,04	0	3,74	1.8×10^{-4}	12,4
5359	1,3	3,65	0,5	0,31	2,28	1,37	9,73	0	2,59	0,01	6,7
3491	1,1	3,01	0,44	0,24	1,97	1,27	7,14	0	2,5	0,01	6,34
5127	2,68	14,62	1	0,73	4,63	2,08	102,94	0	2,69	0,01	7,15
126069	-2,15	0,12	0,83	-3,79	-0,52	0,02	0,59	0	-2,59	0,01	6,69
56829	-2,14	0,12	0,93	-3,97	-0,32	0,02	0,73	0	-2,3	0,02	5,54
23512	-1,86	0,16	0,8	-3,43	-0,3	0,03	0,74	0	-2,34	0,02	5,69
116254	1,67	5,33	0,69	0,32	3,03	1,38	20,67	0	2,42	0,02	6,02
11269	-1,92	0,15	0,84	-3,57	-0,28	0,03	0,76	0	-2,29	0,02	5,51
10724	2,48	11,95	1,06	0,39	4,57	1,48	96,24	0	2,33	0,02	5,66
10445	2,15	8,59	1,02	0,15	4,15	1,17	63,14	0	2,11	0,03	4,85
715	-1,14	0,32	0,57	-2,25	-0,03	0,11	0,97	0	-2,01	0,04	4,48

TABLE 4.3 – Résultats du modèle de régression de Cox pour les génes les plus significatifs dans la survie

Sur l'ensemble des 100 variables les plus corrélées retenues dans l'étude de l'as-

sociation avec la survie, représentée par la variable "days_to_death", on dénote 14 variables significatives. Les 5 plus importantes sont les suivantes :

Covariable 121260 :

Coefficient (coef) : Le coefficient représente la variation attendue dans le logarithme du taux de risque pour une unité d'augmentation de la variable 121260. Dans ce cas, un coefficient de 3,66 suggère une augmentation significative du risque.

Exp(coef) : L'exponentielle du coefficient est interprétée comme le facteur multiplicatif par lequel le taux de risque change pour une unité d'augmentation de la variable 121260. Ici, exp(3,66) équivaut à environ 38,74. Cela signifie que, pour une unité d'augmentation dans la variable 121260, le risque est multiplié par environ 38,74.

Valeur de p : La valeur de p (p-value) est très faible 2×10^{-7}, indiquant une significativité statistique. Cela suggère que le coefficient n'est probablement pas nul, et la relation entre la variable 121260 et le résultat est statistiquement significative.

De manière analogue pour les variables :

Covariable 80042 :

Une augmentation d'une unité de cette covariable entraîne une augmentation significative de 51,28 fois du risque de l'événement. La faible valeur de p suggère une forte signification statistique, renforçant l'impact.

Covariable 5127 :

Une augmentation d'une unité de cette covariable correspond à une augmentation considérable de 14,62 fois du risque de l'événement. La valeur de p est assez faible, indiquant un soutien statistique significatif pour cet effet.

Covariable 5359 :

Une augmentation d'une unité de cette covariable est associée à une augmentation de 3,65 fois du risque de l'événement. La valeur de p est faible, signifiant une signification statistique, bien que moins forte que les précédentes.

Covariable 10724 :

Une augmentation d'une unité de cette covariable entraîne une augmentation de 11,95 fois du risque de l'événement. La valeur de p est modérément faible, soutenant la signification statistique de cet effet. Ces interprétations mettent en évidence l'ampleur

des changements des rapports de risques et la signification statistique associée.

Gène ID	Symbole
121260	SLC15A4
10724	OGA
5359	PLSCR1
5127	CDK16
80042	FLJ12078

TABLE 4.4 – Identification des gènes les plus significatifs dans la survie du cancer du sein par leur symbole

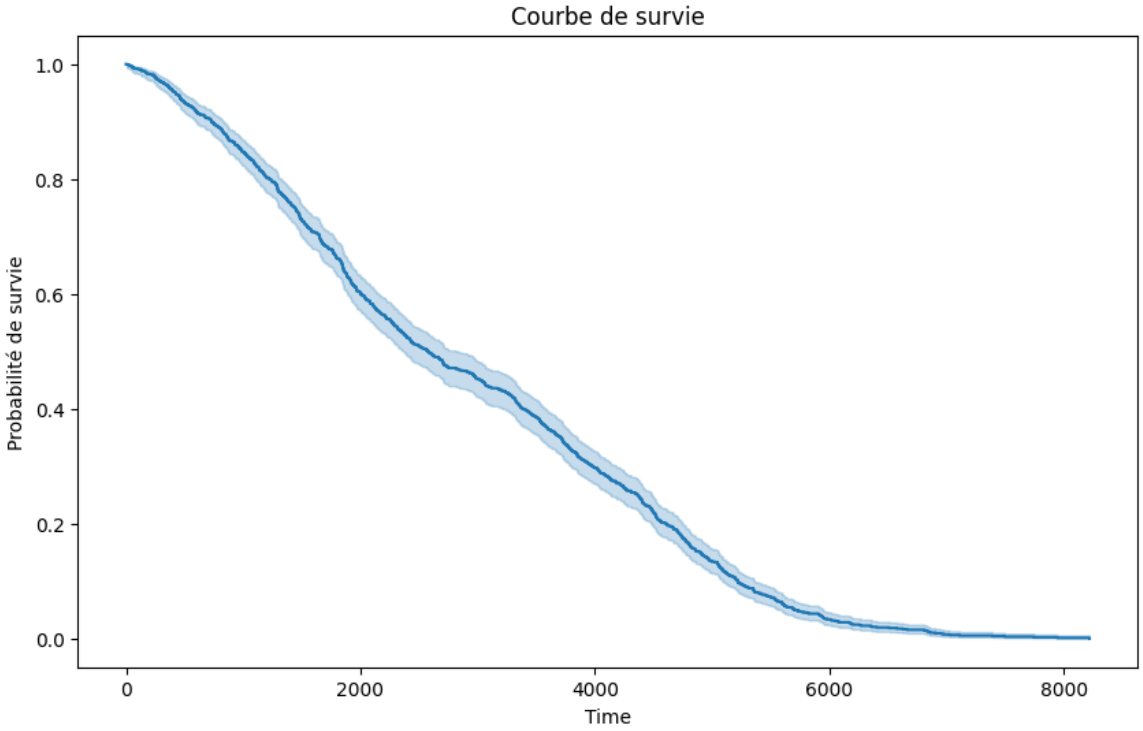

FIGURE 4.2 – Courbe de survie

La courbe de survie est un outil d'analyse de survie largement utilisé pour vi-

sualiser la probabilité de survie au fil du temps. La courbe commence à 1 (100 % de probabilité de survie) à l'instant initial et diminue à mesure que le temps avance. Chaque descente de la courbe représente un événement de décès ou de défaillance. Les paliers horizontaux entre les descentes indiquent des périodes où aucun événement n'a eu lieu. La forme de la courbe de survie peut fournir des informations précieuses sur les données de survie. Une courbe qui descend rapidement indique une diminution rapide de la probabilité de survie, ce qui peut signifier que l'événement d'intérêt a une forte influence sur la survie. À l'inverse, une courbe qui descend lentement suggère une probabilité de survie relativement élevée au fil du temps.
Dans la figure 4.2, on observe que la courbe de survie descend rapidement, indiquant une diminution rapide de la probabilité de survie. Cela confirme les résultats trouvés sur la distribution de survie : plus le temps augmente, plus la probabilité de survie diminue.

4.3 Modèle multivarié

Nous avons différentes méthodes pour sélectionner les 100 meilleures variables parmi les 24925 génes en relation avec la survie, notamment la corrélation Pearson, l'information mutuelle et la corrélation de Spearman. La sélection des caractéristiques joue un rôle essentiel dans l'amélioration des performances prédictives des modèles, et cette étude explore trois approches pour identifier les variables les plus pertinentes et ainsi affiner la qualité de la prédiction : la corrélation de Pearson, la corrélation de Spearson et l'information mutuelle.

4.4 Méthode basée sur la corrélation et approches en apprentissage automatique

Pour la méthode de sélection de caractéristiques de Pearson, nous avons calculé la corrélation de Pearson en valeur absolue pour l'ensemble des 24925 gènes par rapport à la survie. Nous avons sélectionné les 100 meilleures caractéristiques qui sont les plus corrélées avec la survie. Ensuite, nous avons calculé les statistiques descriptives de la corrélation des variables sélectionnées. Les résultats obtenus sont représentés dans le tableau 4.5.

Corrélation	
Minimum	9.21×10^{-6}
Premier quartile (Q1)	0.02
Médiane	0.046
Troisième quartile (Q3)	0.085
Maximum	0.33

TABLE 4.5 – Statistiques sur la Durée de Survie (days_to_death) pour le Cancer du Sein

Les statistiques révèlent une gamme captivante de valeurs de corrélation, reflétant la diversité des liens potentiels entre les gènes et les résultats de survie. Parmi les valeurs présentées, on observe un éventail allant d'une corrélation minimale de 9.21×10^{-6} à une corrélation maximale de 0.33. Cette dispersion souligne la variabilité des interactions possibles entre les gènes et la survie, offrant ainsi un aperçu des relations subtiles et complexes au sein de l'ensemble de données. Les quartiles de corrélation fournissent également des repères essentiels pour saisir la distribution des valeurs. Le premier quartile, à 0.02151, indique que 25% des gènes présentent une corrélation supérieure à ce seuil, tandis que la médiane, à 0.05, illustre la médiane des valeurs de corrélation, signalant ainsi le point central de cette distribution. Le troisième quartile, à 0.085, met en évidence la proportion de gènes ayant des corrélations encore plus marquées avec la survie. Ces chiffres suggèrent que les gènes peuvent avoir des degrés

variables de corrélation avec la survie, allant des liens plus faibles aux associations plus prononcées. Il est crucial de souligner que ces mesures de corrélation offrent un aperçu statistique des relations potentielles et ne tiennent pas compte des interactions biologiques complexes qui peuvent sous-tendre ces observations.
Nous avons appliqué de manière analogue la méthode de sélection de caractéristiques de Spearman et celle de l'information mutuelle.

En somme, ces résultats apportent un éclairage précieux sur la complexité des liens entre les gènes et la survie, incitant à des investigations plus approfondies pour comprendre les implications biologiques et cliniques de ces découvertes.

Nous avons évalué sept modèles de régression couramment utilisés dans la prédiction médicale en utilisant un jeu de données comprenant des informations sur des patients atteints de cancer du sein et le temps qui s'est écoulé avant leur décès.Les modèles étudiés sont tous détaillés dans le chapitre 3, qui sont les suivants : forêt aléatoire, régression Ridge, régression Lasso, régression linéaire, régression élastique net, SVR cas non linéaire et SVR cas linéaire. Pour chaque modèle, nous avons enregistré le nombre de variables utilisées et le coefficient de détermination R^2 pour évaluer les performances de prédiction.

4.5 Étude comparative

4.5.1 Méthode de sélection de caractéristique de Pearson

Après avoir utilisé la méthode de sélection de Pearson pour sélectionner les 100 variables les plus importantes dans la survie, nous avons mis en place notre premier modèle de régression linéaire en utilisant une boucle pour évaluer les modèles

avec différentes combinaisons de caractéristiques. À chaque itération de la boucle, un modèle de régression linéaire est entraîné sur l'ensemble d'entraînement avec validation croisée. Les prédictions sont ensuite faites sur l'ensemble de validation et le métrique d'évaluation telle que l'erreur absolue moyenne (MAE) est calculée. Nous avons également calculé la corrélation et l'indice de concordance entre les valeurs réelles et prédites. Enfin, nous avons tracé un graphique montrant la performance R^2 en fonction du nombre de variables, et nous avons utilisé cette performance dans cette étude pour évaluer les différents modèles. Nous avons répété le même processus pour tous les modèles dans les différentes méthodes de sélection de caractéristiques. Le graphique trouvé pour le modèle de régression linéaire est représenté dans la figure 4.3.

La figure 4.3 représente la performance R^2 en fonction du nombre de variables. Le R^2 trouvé est égal à 0.1984, ce qui est très faible, montrant l'incapacité du modèle de régression linéaire à fournir une meilleure explication de la variable cible days_to_death. Le nombre de variables associées est égal à 38 variables.

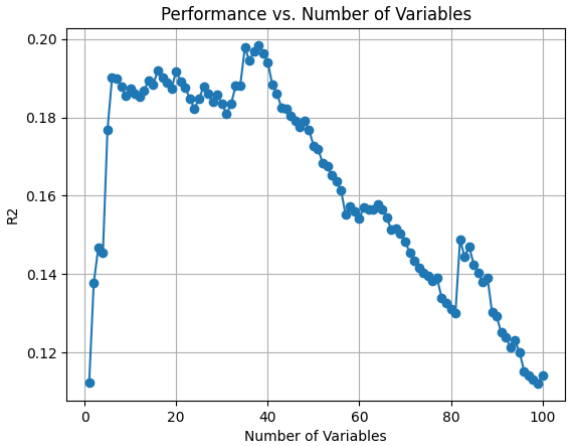

FIGURE 4.3 – Performance du modèle de régression linéaire pour la méthode de sélection de caractéristiques de Pearson

La figure 4.4 représente la corrélation entre les valeurs réelles et les valeurs prédites en fonction du nombre de variables dans le modèle de régression linéaire. La figure 4.4

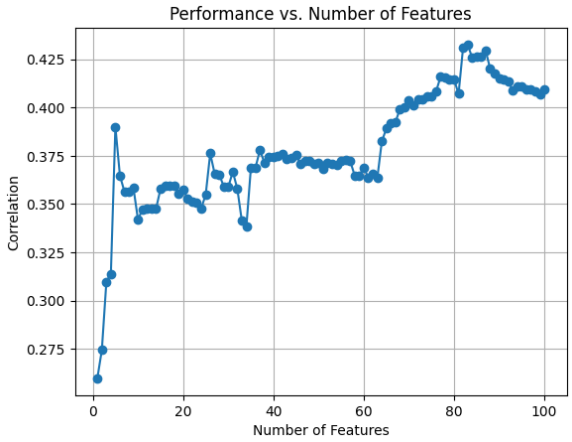

FIGURE 4.4 – Corrélation entre les valeurs réelles et les valeurs prédites en fonction du nombre de variables

montre une corrélation maximale de 0,43248, ce qui indique une corrélation positive modérée entre les valeurs réelles et prédites. Le nombre maximal de variables associées est égal à 82 variables. Cela suggère que les valeurs prédites augmentent généralement avec les valeurs réelles et vice versa, mais cette relation n'est pas extrêmement forte.

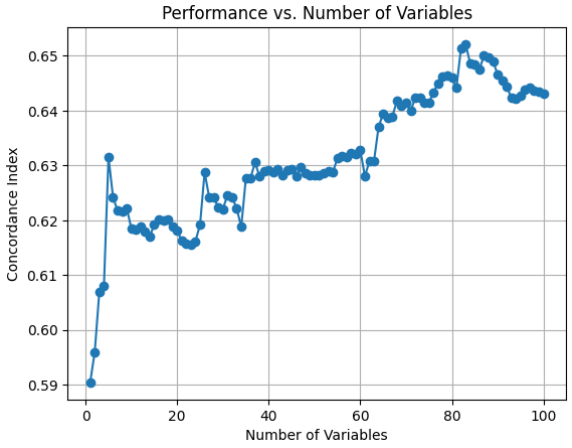

FIGURE 4.5 – indice de concordance entre les valeurs réelles et les valeurs prédites en fonction du nombre de variables

La figure 4.5 représente l'indice de concordance entre les valeurs réelles et les valeurs prédites en fonction du nombre de variables dans le modèle de régression linéaire. La figure 4.5 montre un indice de concordance maximale de 0,6521, ce qui indique un accord substantiel entre les classements des valeurs réelles et prédites dans le modèle de régression linéaire. Le nombre maximal de variables associées est de 82. En d'autres

termes, les positions relatives des observations dans les deux ensembles de données sont similaires. Cette valeur élevée de l'indice de concordance suggère que le modèle est capable de reproduire avec précision les ordres des valeurs réelles dans ses prédictions.

Pour la suite, nous avons décidé de représenter les résultats obtenus sur les graphiques sous forme de tableau pour tous les modèles afin d'avoir une meilleure visibilité.

Modèles	Nombre de variables	R^2	MAE
Forêt aléatoire	94	0,88924	1226,96
Régression Ridge	36	0,21511	1241,05
Régression Lasso	36	0,20838	1239,49
Régression Linéaire	38	0,1984	1255,31
Régression Net Élastique	96	0,0968	134,06
SVR non linéaire	3	0,0221	1421,39
SVR linéaire	93	-0,01248	1425,50

TABLE 4.6 – Tableau comparatif des performances des modèles de régression pour la méthode de sélection de caractéristiques de Pearson

Les résultats de notre étude comparative montrent une variabilité significative dans les performances des différents modèles. Le modèle de Forêt aléatoire se démarque en affichant la meilleure performance parmi tous les modèles évalués. Son coefficient de détermination R^2 élevé de 0,88924 suggère qu'il est capable d'expliquer une grande partie de la variance dans les données. De plus, sa MAE (Erreur Absolue Moyenne) relativement faible de 1226,96 indique que les prédictions du modèle sont proches des valeurs réelles. Ces résultats indiquent que le modèle de Forêt aléatoire offre des prédictions précises et robustes. En revanche, les modèles de régression régularisée, tels que la Régression Ridge et la Régression Lasso, présentent des performances similaires en termes de R^2 et de MAE, bien que légèrement inférieures à celles du Forêt aléatoire. Malgré cela, ils montrent un R^2 relativement faible qui suggère qu'ils ne parviennent pas à capturer la structure sous-jacente des données. Quant aux modèles

de Régression Nette Élastique, SVM Radial et SVM Linéaire, ils affichent des performances plus faibles, caractérisées par des R^2 proches de zéro ou négatifs et des MAE plus élevées. Cela suggère qu'ils ont du mal à prédire avec précision les valeurs de survie, ce qui peut être attribué à leur incapacité à capturer la complexité de la relation entre les caractéristiques et la variable cible.

Les résultats de notre étude comparative montrent une variabilité significative dans les performances des différents modèles. Le modèle de Forêt aléatoire se démarque avec un R^2 impressionnant de 0,87747, indiquant une bonne capacité à prédire le nombre de jours avant le décès. En revanche, la Régression Linéaire, la Régression Nette Élastique, et les SVR cas linéaire et non linéaires présentent des R^2 relativement faibles, suggérant une moindre capacité à expliquer la variance des données.

Modèles	Nombres de variables	Corrélation de Pearson
Forêt Aléatoire	63	0,47107
Régression Ridge	82	0,43745
Régression Lasso	86	0,44409
Régression Linéaire	82	0,43249
Régression élastique Net	10	0,36034
SVR non linéaire	4	0,4017
SVR linéaire	99	0,3315

TABLE 4.7 – Corrélations de Pearson maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles

Les résultats obtenus reflétent des variations intéressantes dans les performances des modèles en fonction de la corrélation de Pearson. Voici ce que nous avons observé pour chaque modèle : Forêt Aléatoire (Corrélation : 0,47107) : La corrélation positive suggère une cohérence entre les valeurs prédites et les valeurs réelles. Cela peut indiquer que le modèle Random Forest capture efficacement les tendances et les variations des données. Régression Ridge (Corrélation : 0,43745) : Bien que légèrement inférieure à celle du Random Forest, cette corrélation dénote toujours une relation significative entre les prédictions et les valeurs réelles. Le modèle Régression Ridge parvient à saisir certaines des tendances des données. Régression Lasso (Corrélation :

0,44409) : Avec une corrélation similaire à celle du Random Forest, la régression Lasso montre également une corrélation positive. Cela indique une correspondance entre les prédictions du modèle et les valeurs réelles. Régression Linéaire (Corrélation : 0,43249) : Bien que légèrement inférieure, la corrélation positive de la Régression Linéaire montre que ce modèle parvient à capter des tendances générales dans les données. SVM Radial (Corrélation : 0,4017) : Une corrélation inférieure suggère que ce modèle peut nécessiter des ajustements pour mieux saisir les subtilités des données et fournir des prédictions plus précises.

Modèles	Nombres de variables	Indice de concordance
Forêt Aléatoire	93	0,6524
Régression Ridge	82	0,6514
Régression Lasso	87	0,6520
Régression Linéaire	82	0,6521
Régression élastique Net	4	0,6212
SVR non linéaire	4	0,6305
SVR linéaire	99	0,6059

TABLE 4.8 – Indice de concordance maximale entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles

Dans le domaine de la prédiction de la survie du cancer du sein, l'accord entre les valeurs prédites par les modèles et les valeurs réelles revêt une importance capitale pour la prise de décisions cliniques éclairées. Le modèle de Forêt aléatoire se distingue en affichant un indice de concordance élevé de 0,6524, ce qui suggère un accord solide entre les prédictions et les résultats observés, permettant ainsi de fournir des estimations précises de la survie des patients. Les modèles de Régression Ridge, Lasso et Linéaire présentent des indices de concordance similaires, tous autour de 0,65, indiquant également un bon accord entre les prédictions et les valeurs réelles dans le contexte de la survie du cancer du sein. Ces modèles offrent ainsi une estimation fiable de la survie des patients, facilitant ainsi la prise de décisions médicales. En revanche, les indices de concordance des modèles de Régression Élastique Net, SVR non linéaire et SVR Linéaire sont légèrement inférieurs, ce qui suggère un accord légèrement moins robuste entre les prédictions et les valeurs réelles pour ces modèles

dans le cadre spécifique de la survie du cancer du sein. Bien que ces modèles puissent fournir des prédictions utiles, leur capacité à estimer précisément la survie des patients peut être légèrement limitée par rapport aux autres modèles.

4.5.2 Méthode de sélection de caractéristique de Spearman

Le tableau 4.9 compare les performances des différents modèles de régression en utilisant la méthode de sélection de caractéristiques de Spearman.

Modèles	Nombres de variables	R^2	MAE
Forêt Aléatoire	67	0,88883	1223,80
Régression Ridge	45	0,21826	1244,52
Régression Lasso	43	0,21335	1243,55
Régression Linéaire	20	0,20262	1262,7972
Régression élastique Net	98	0,12468	1388,2999
SVR non linéaire	2	-0,01036	1421,3405
SVR linéaire	98	-0,01103	1426,9453

TABLE 4.9 – Tableau comparatif des performances des modèles de régression pour la méthode de sélection de caractéristiques de Spearman

Les résultats de notre étude comparative révèlent une variabilité significative dans les performances des différents modèles évalués. Parmi eux, le modèle de Forêt aléatoire se distingue en affichant la meilleure performance. Son coefficient de détermination R^2 élevé de 0,88883 suggère qu'il est capable d'expliquer une grande partie de la variance dans les données, tandis que sa faible MAE (Erreur Absolue Moyenne) de 1223,80 indique des prédictions proches des valeurs réelles. Ces résultats mettent en avant la précision et la robustesse des prédictions du modèle de Forêt aléatoire. En revanche, les modèles de régression régularisée, tels que la Régression Ridge et la Régression Lasso, présentent des performances similaires, bien que légèrement inférieures en termes de R^2 et de MAE par rapport au Forêt aléatoire. Malgré cela, leur R^2 relativement faible suggère une difficulté à capturer la structure sous-jacente

des données. Quant aux modèles de Régression Élastique Net, SVR non linéaire et SVR Linéaire, ils affichent des performances plus faibles, caractérisées par des R^2 proches de zéro ou négatifs et des MAE plus élevées. Cette faible précision peut être attribuée à leur difficulté à modéliser la relation complexe entre les caractéristiques et la variable cible.

Modèles	Nombres de variables	Corrélation de Spearman
Forêt Aléatoire	86	0,4894
Régression Ridge	88	0,4342
Régression Lasso	88	0,4433
Régression Linéaire	88	0,4265
Régression élastique Net	95	0,3654
SVR non linéaire	7	0,3954
SVR linéaire	99	0,3377

TABLE 4.10 – Corrélations de Spearman maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles

Le tableau 4.10 présente les corrélations de Spearman maximales entre les valeurs réelles et prédites pour différents modèles de régression, ainsi que le nombre de variables associées à ces corrélations maximales. Dans ce tableau, nous observons que le modèle de Forêt aléatoire affiche une corrélation maximale de 0,4894 avec 86 variables associées. Cette valeur suggère une relation monotone relativement forte entre les valeurs réelles et prédites pour ce modèle. Les modèles de Régression Ridge, Lasso et Linéaire présentent également des corrélations élevées, toutes autour de 0,43 à 0,44, avec un nombre similaire de variables associées. Cela indique une relation monotone positive, bien que légèrement moins forte que celle observée avec le modèle de Forêt aléatoire. En revanche, les modèles de Régression Élastique Net, SVR non linéaire et SVR Linéaire affichent des corrélations plus faibles, comprises entre 0,33 et 0,40, avec un nombre variable de caractéristiques associées. Cette observation suggère une relation monotone moins forte entre les valeurs réelles et prédites pour ces modèles, ce qui peut indiquer une performance inférieure en termes de capacité à reproduire la relation entre les caractéristiques et la variable cible.

Le tableau 4.11 présente les indices de concordance maximale entre les valeurs réelles et prédites pour différents modèles de régression, ainsi que le nombre de variables associées à ces indices maximaux. Le modèle de Forêt aléatoire affiche le plus haut indice de concordance, avec une valeur de 0,6565, et il est associé à 86 variables. Cela suggère un bon accord entre les prédictions du modèle et les valeurs réelles, ce qui indique une capacité robuste à estimer avec précision les résultats. Les modèles de Régression Ridge, Lasso et Linéaire présentent des indices de concordance similaires, tous autour de 0,65, avec un nombre similaire de variables associées. Cela indique également un bon accord entre les prédictions et les valeurs réelles pour ces modèles, bien que légèrement inférieur à celui de Forêt aléatoire. En revanche, les modèles de Régression Élastique Net, SVR non linéaire et SVR Linéaire ont des indices de concordance légèrement plus faibles, avec des valeurs comprises entre 0,62 et 0,63. Cela suggère un accord légèrement moins robuste entre les prédictions et les valeurs réelles pour ces modèles, mais reste tout de même assez satisfaisant dans l'ensemble.

Modèles	Nombres de variables	Indice de concordance
Forêt Aléatoire	86	0,6565
Régression Ridge	97	0,6489
Régression Lasso	97	0,6518
Régression Linéaire	46	0,6412
Régression Élastique Net	87	0,62
SVM radial	4	0,6304
SVM linéaire	99	0,6089

TABLE 4.11 – Indice de concordance maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles

4.5.3 L'information mutuelle

Les conclusions de notre analyse comparative mettent en évidence une diversité notable dans les performances des divers modèles évalués.

Statistique	**Valeur**
Minimum	0.0
Premier quartile	0.0
Médiane	0.007
Troisième quartile	0.02
Maximum	0.12081836810798352

TABLE 4.12 – Tableau des pourcentages des scores d'information mutuelle

Les quartiles de l'information mutuelle fournissent des points de référence significatifs pour comprendre la distribution des valeurs. Avec un premier quartile et une médiane à 0.0, cela indique que 25 % des gènes peuvent avoir une relation nulle avec la survie. La médiane, à 0.007, offre un aperçu de la valeur typique d'association. Le troisième quartile, à 0.02, suggère que certains gènes peuvent présenter une association plus forte avec la survie. Ces chiffres démontrent que les gènes peuvent avoir une gamme de relations avec la survie, allant de connexions peu significatives à des associations potentiellement importantes.

Modèles	Nombres de variables	R^2	MAE
Forêt Aléatoire	34	0,88898	1231,15
Régression Ridge	23	0,21696	1263,20
Régression Lasso	24	0,21220	1271,40
Régression Linéaire	24	0,20729	1287,95
Régression Élastique Net	98	0,0589	1438,17
SVR non linéaire	1	0,01362	1433,33
SVR linéaire	94	0,00945	1428,10

TABLE 4.13 – Tableau comparatif des performances des modèles de régression pour la méthode de sélection de caractéristiques de mutuelle information

Le tableau 4.13 présente les performances des différents modèles évalués en termes de coefficient de détermination R^2 et d'Erreur Absolue Moyenne (MAE), ainsi que le nombre de variables associées à ces performances maximales. Le modèle de Forêt aléatoire se distingue avec un R^2 élevé de 0,88898, indiquant qu'il est capable d'expliquer une grande partie de la variance dans les données. De plus, sa MAE relativement basse de 1231,15 suggère que les prédictions du modèle sont proches des valeurs

réelles, ce qui témoigne de sa capacité à fournir des estimations précises. Les modèles de Régression Ridge, Lasso et Linéaire présentent des R^2 et des MAE légèrement inférieurs, mais ils montrent tout de même une capacité à capturer une partie de la variabilité des données. Le modèle de Régression élastique Net affiche des performances encore moins bonnes, avec un R^2 de 0,0589 et une MAE de 1438,17, ce qui suggère une capacité limitée à expliquer la variance des données et à fournir des prédictions précises. Enfin, les modèles SVR non linéaire et SVR linéaire montrent les performances les plus faibles, avec des R^2 proches de zéro et des MAE relativement élevées. Cela suggère qu'ils ont du mal à capturer la structure des données et à fournir des prédictions précises.

Modéles	Nombres de variables	Corrélation
Forêt aléatoire	64	0,4795
Régression Ridge	55	0,3881
Régression Lasso	55	0,3758
Régression Linéaire	76	0,3656
Régression élastique Net	86	0,3094
SVR non linéaire	50	0,3500
SVR linéaire	78	0,3167

TABLE 4.14 – Corrélations maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles

Le tableau 4.14 présente les corrélations maximales entre les valeurs réelles et prédites pour différents modèles de régression, ainsi que le nombre de variables associées à ces corrélations maximales. Le modèle de Forêt aléatoire affiche la corrélation maximale la plus élevée, avec une valeur de 0,4795, associée à 64 variables. Cela suggère une relation positive modérée entre les valeurs réelles et prédites pour ce modèle, indiquant une capacité relativement bonne à reproduire les observations réelles. Les modèles de Régression Ridge, Lasso et Linéaire présentent des corrélations maximales légèrement inférieures, toutes autour de 0,38 à 0,39, avec un nombre similaire de variables associées. Cela indique également une relation positive modérée entre les valeurs réelles et prédites pour ces modèles, bien que légèrement moins forte que celle observée avec le modèle de Forêt aléatoire. En revanche, les modèles de Régression

Élastique Net, SVR non linéaire et SVr Linéaire ont des corrélations maximales plus faibles, avec des valeurs comprises entre 0,31 et 0,35. Cela suggère une relation positive plus faible entre les valeurs réelles et prédites pour ces modèles, indiquant une performance globalement moins satisfaisante en termes de capacité à reproduire la relation entre les caractéristiques et la variable cible.

Modèles	Nombres de variables	Indice de concordance
Forêt aléatoire	64	0,6556
Régression Ridge	85	0,6334
Régression Lasso	84	0,6304
Régression Linéaire	76	0,6245
Régression élastique Net	86	0,6041
SVR non linéaire	86	0,6126
SVR linéaire	78	0,5967

TABLE 4.15 – Indice de concordance maximales entre les valeurs réelles et prédites avec les nombres de caractéristiques maximales associées pour les différents modèles

Le tableau 4.15 présente les indices de concordance maximaux entre les valeurs réelles et prédites pour différents modèles de régression, ainsi que le nombre de variables associées à ces indices maximaux. Le modèle de Forêt aléatoire affiche le plus haut indice de concordance, avec une valeur de 0,6556, et il est associé à 64 variables. Cela suggère un bon accord entre les prédictions du modèle et les valeurs réelles, ce qui indique une capacité robuste à estimer avec précision les résultats. Les modèles de Régression Ridge, Lasso et Linéaire présentent des indices de concordance similaires, tous autour de 0,62 à 0,63, avec un nombre similaire de variables associées. Cela indique également un bon accord entre les prédictions et les valeurs réelles pour ces modèles, bien que légèrement inférieur à celui du Forêt aléatoire. En revanche, les modèles de Régression Élastique Net, SVR non linéaire et SVR Linéaire ont des indices de concordance légèrement plus faibles, avec des valeurs comprises entre 0,60 et 0,61. Cela suggère un accord légèrement moins robuste entre les prédictions et les valeurs réelles pour ces modèles, mais reste tout de même assez satisfaisant dans l'ensemble.

4.6 Synthèse des résultats obtenus

Les tableaux 4.15 et 4.16 représentent la synthèse des résultats pour les différents modèles selon le type de sélection de caractéristiques.

	Corrélation Pearson			Corrélation Spearman			Mutuelle information		
Modèles	NV	R^2	MAE	NV	R^2	MAE	NV	R^2	MAE
Forêt aléatoire	94	0,88924	1226,96	67	0,88883	1223,80	34	0,88898	1234,15
Régression Ridge	36	0,21511	1241,05	45	0,21826	1244,25	23	0,21269	1263,20
Régression Lasso	36	0,20838	1239,49	43	0,21335	1243,55	24	0,21220	1271,40
Régression linéaire	38	0,1984	1255,31	20	0,20262	1262,79	24	0,20729	1287,95
Régression Élastique Net	96	0,0968	1384,06	98	0,12468	1388,29	98	0,0586	1438,17
SVR non linéaire	3	0,0221	1421,39	2	-0,01036	1421,34	1	0,013	1433,33
SVR Linéaire	93	-0,01248	1425,50	98	-0,01103	1426,94	94	0,0094	1428,10

TABLE 4.16 – Comparaison des performances des différents modèles de régression dans la prédiction de la survie des patients atteints de cancer du sein selon les méthodes de sélection de caractéristiques

	Corrélation Pearson		Corrélation Spearman		Mutuelle information	
Modèles	NV	IC	NV	IC	NV	IC
Forêt aléatoire	93	0,6524	86	0,6565	64	0,6556
Régression Ridge	82	0,6514	97	0,6489	85	0,6334
Régression Lasso	87	0,6520	97	0,6518	84	0,6304
Régression linéaire	82	0,6521	46	0,6412	76	0,6245
Régression Élastique Net	4	0,6212	87	0,62	86	0,6041
SVR non linéaire	4	0,6305	7	0,6304	86	0,6126
SVR Linéaire	99	0,6059	95	0,6089	78	0,5967

TABLE 4.17 – Indices de concordance entre les valeurs réelles et prédites pour les différents modèles de régression dans la prédiction de la survie des patients atteints de cancer du sein selon les méthodes de sélection de caractéristiques

Les tableaux 4,16 et 4,17 présentent une analyse comparative des performances de divers modèles de régression dans la prédiction de la survie des patients atteints de cancer du sein, en utilisant des caractéristiques géniques comme variables. Ces tableaux fournissent plusieurs mesures de performance, notamment les coefficients de détermination R^2, les erreurs absolues moyennes (MAE), les corrélations de Pearson ou de Spearman, ainsi que les indices de concordance entre les valeurs réelles et prédites. Globalement, nous constatons une variabilité significative dans les performances des

modèles évalués. Le modèle de Forêt aléatoire se distingue généralement en affichant les meilleures performances, avec des R^2 élevés, des MAE relativement faibles et des corrélations significatives entre les valeurs prédites et réelles. En particulier, le meilleur R^2 est obtenu avec la méthode de sélection de caractéristiques de Pearson, atteignant une valeur remarquable de 0,88924, ce qui souligne la capacité précise de ce modèle à expliquer la variance des données.

Cependant, certains modèles de régression linéaire simple ou de régression régularisée présentent des performances moins impressionnantes.

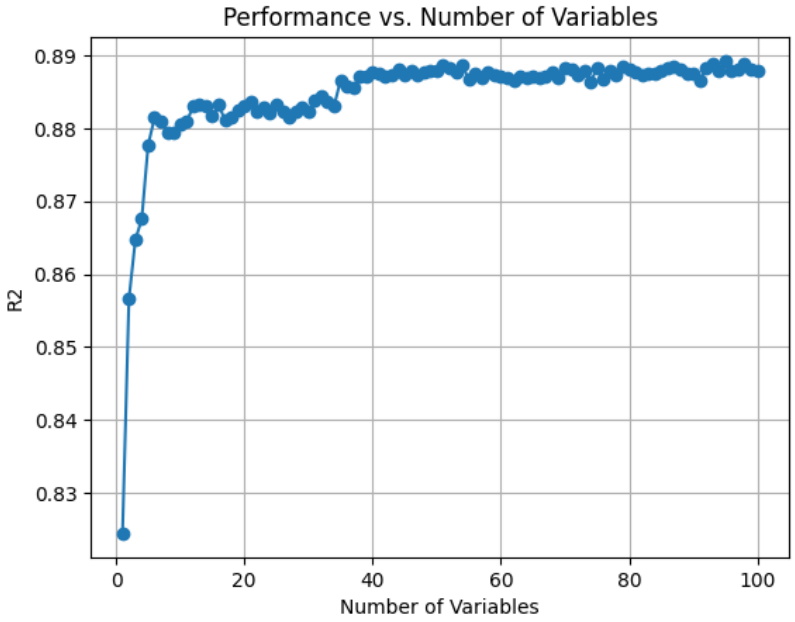

FIGURE 4.6 – Performance du modèle de random Forest pour la méthode de sélection de caractéristiques de Pearson

Après avoir identifié le modèle comme le meilleur modèle pour la méthode de sélection de caractéristiques de Pearson, dont les résultats sont représentés dans la figure 4.6, nous avons récupéré le nombre maximal de caractéristiques associées à la valeur R^2, qui est égal à 94 variables. Nous avons ensuite utilisé ces 94 variables pour créer un nouveau modèle de Forêt aléatoire dans la base de validation, afin de valider les résultats obtenus. Enfin, nous avons tracé la courbe entre les valeurs réelles et prédites, donnant la figure 4.7 et le R^2 associé, qui est égal à 0,89 qui vient de confirmer les résultats obtenus dans la base de découverte.

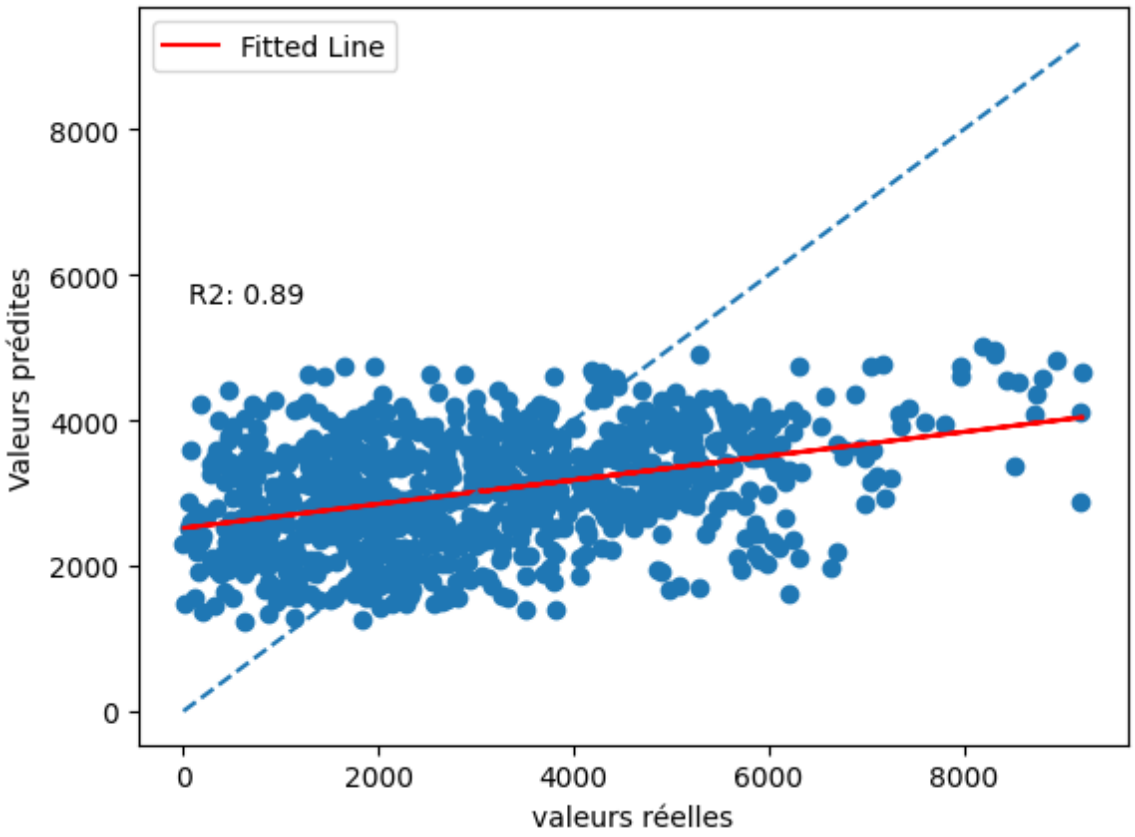

FIGURE 4.7 – Courbe entre valeurs réelles et prédites

Ces informations peuvent être extrêmement utiles pour les médecins et les professionnels de la santé lors de la prise de décisions cliniques. En utilisant ces modèles de prédiction de survie basés sur les données géniques, les médecins peuvent évaluer de manière plus approfondie les risques et les résultats potentiels pour les patients atteints de cancer du sein. Par exemple, un modèle avec un R^2 élevé et une faible MAE peut indiquer une capacité à fournir des prédictions précises sur la survie des patients, ce qui pourrait guider les médecins dans l'élaboration de plans de traitement personnalisés et la prise de décisions thérapeutiques plus éclairées. De plus, les informations sur les variables géniques associées aux meilleures performances des modèles peuvent offrir des insights précieux sur les facteurs influençant la survie des patients, ce qui pourrait informer le développement de nouvelles stratégies de traitement et de gestion des soins. Dans l'ensemble, l'utilisation de modèles de prédiction de la survie basés sur des données géniques peut contribuer à une prise de décision clinique plus objective, informée et personnalisée, ce qui finalement améliore les résultats pour les patients atteints de cancer du sein.

Chapitre 5

Conclusion et perspectives

Dans ce mémoire, plusieurs méthodes de sélection de caractéristiques et des modèles d'apprentissage automatique ont été étudiés dans le but de prédire la survie du cancer du sein, afin de permettre aux médecins de prendre des décisions éclairées. Nous avons appliqué trois méthodes de sélection de caractéristiques, à savoir la corrélation de Pearson, de Spearman et l'information mutuelle, sur une base de données génétiques contenant les identifiants des patients et leurs caractéristiques génétiques. Ensuite, nous avons combiné les caractéristiques sélectionnées de chaque méthode avec la base de données clinique contenant la variable de survie days_to_death. Pour déterminer le modèle offrant la meilleure explication de la survie du cancer du sein, nous avons comparé sept modèles d'apprentissage automatique, à savoir la forêt aléatoire, la régression linéaire, la régression Ridge, Lasso et élastique Net, ainsi que la SVR linéaire et non linéaire pour chaque méthode de sélection de caractéristiques.

L'analyse des résultats révèle que sur l'ensemble des méthodes de sélection de caractéristiques, le modèle de forêt aléatoire offre systématiquement les meilleurs résultats, avec des valeurs élevées de R^2, d'indices de concordance et des erreurs absolues moyennes (MAE) faibles. Ces résultats revêtent une importance cruciale pour

les médecins dans leur prise de décision clinique. Ils peuvent les aider à concevoir des stratégies de traitement personnalisées et à identifier les patients à haut risque, leur permettant ainsi d'adapter les interventions thérapeutiques de manière plus précise et efficace. De plus, ces résultats peuvent contribuer à une meilleure compréhension des facteurs génétiques sous-jacents au cancer du sein et à l'identification de nouvelles cibles thérapeutiques potentielles.

Dans la mesure où le cancer du sein est impacté par plusieurs facteurs de risque, nous envisageons d'ajouter d'autres facteurs tels que le stress, la ménopause et l'environnement dans notre jeu de données pour généraliser nos résultats, ainsi que d'appliquer des réseaux de neurones artificiels pour améliorer les performances de notre étude.

Bibliographie

[1] X. Liu, *Survival analysis : models and applications*. John Wiley & Sons, 2012.

[2] P. D. Allison, *Survival analysis using SAS : a practical guide*. Sas Institute, 2010.

[3] J. Zhang, L. Chen, A. Vanasse, J. Courteau et S. Wang, « Survival prediction by an integrated learning criterion on intermittently varying healthcare data », *in Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, 2016.

[4] C. Oberije, D. De Ruysscher, R. Houben, M. van de Heuvel, W. Uyterlinde, J. O. Deasy, J. Belderbos, A.-M. C. Dingemans, A. Rimner, S. Din *et al.*, « A validated prediction model for overall survival from stage iii non-small cell lung cancer : toward survival prediction for individual patients », *International Journal of Radiation Oncology* Biology* Physics*, vol. 92, no. 4, p. 935–944, 2015.

[5] B. S. Chhikara et K. Parang, « Global cancer statistics 2022 : the trends projection analysis », *Chemical Biology Letters*, vol. 10, no. 1, p. 451–451, 2023.

[6] S. Deo, J. Sharma et S. Kumar, « Globocan 2020 report on global cancer burden : challenges and opportunities for surgical oncologists », *Annals of surgical oncology*, vol. 29, no. 11, p. 6497–6500, 2022.

[7] F. Bray, J. Ferlay, I. Soerjomataram, R. L. Siegel, L. A. Torre et A. Jemal, « Global cancer statistics 2018 : Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries », *CA : a cancer journal for*

clinicians, vol. 68, no. 6, p. 394–424, 2018.

[8] K. E. Lukong, « Understanding breast cancer–the long and winding road », *BBA clinical*, vol. 7, p. 64–77, 2017.

[9] L. Wilkinson et T. Gathani, « Understanding breast cancer as a global health concern », *The British Journal of Radiology*, vol. 95, no. 1130, p. 20211033, 2022.

[10] M. E. Barnard, C. E. Boeke et R. M. Tamimi, « Established breast cancer risk factors and risk of intrinsic tumor subtypes », *Biochimica et Biophysica Acta (BBA)-Reviews on Cancer*, vol. 1856, no. 1, p. 73–85, 2015.

[11] H. Chouaib, « Sélection de caractéristiques : méthodes et applications », *Paris Descartes University : Paris, France*, 2011.

[12] G. Zeng *et al.*, « A unified definition of mutual information with applications in machine learning », *Mathematical Problems in Engineering*, vol. 2015, 2015.

[13] A. Shadvar, « Dimension reduction by mutual information feature extraction », *arXiv preprint arXiv :1207.3394*, 2012.

[14] L. Song, P. Langfelder et S. Horvath, « Comparison of co-expression measures : mutual information, correlation, and model based indices », *BMC bioinformatics*, vol. 13, no. 1, p. 1–21, 2012.

[15] K. Torkkola et W. M. Campbell, « Mutual information in learning feature transformations », *in ICML*, p. 1015–1022, Citeseer, 2000.

[16] J. Cheng, J. Sun, K. Yao, M. Xu et Y. Cao, « A variable selection method based on mutual information and variance inflation factor », *Spectrochimica Acta Part A : Molecular and Biomolecular Spectroscopy*, vol. 268, p. 120652, 2022.

[17] L. Zhou, K. K. Lai et J. Yen, « Empirical models based on features ranking techniques for corporate financial distress prediction », *Computers & mathematics with applications*, vol. 64, no. 8, p. 2484–2496, 2012.

[18] F. Rossi, A. Lendasse, D. François, V. Wertz et M. Verleysen, « Mutual information for the selection of relevant variables in spectrometric nonlinear modelling », *Chemometrics and intelligent laboratory systems*, vol. 80, no. 2, p. 215–226, 2006.

[19] C. Jutten et R. Gribonval, « L'analyse en composantes indépendantes : un outil puissant pour le traitement de l'information », *in Proc. XIXe colloque GRETSI (traitement du signal et des images), 8-11 septembre 2003*, vol. 1, p. 11–16, GRETSI, Groupe d'Etudes du Traitement du Signal et des Images, 2003.

[20] J. Hauke et T. Kossowski, « Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data », *Quaestiones geographicae*, vol. 30, no. 2, p. 87–93, 2011.

[21] C. Croux et C. Dehon, « Influence functions of the spearman and kendall correlation measures », *Statistical methods & applications*, vol. 19, p. 497–515, 2010.

[22] P. Dutilleul, J. D. Stockwell, D. Frigon et P. Legendre, « The mantel test versus pearson's correlation analysis : Assessment of the differences for biological and environmental studies », *Journal of agricultural, biological, and environmental statistics*, p. 131–150, 2000.

[23] N. S. Chok, *Pearson's versus Spearman's and Kendall's correlation coefficients for continuous data*. Thèse doctorat, University of Pittsburgh, 2010.

[24] M.-T. Puth, M. Neuhäuser et G. D. Ruxton, « Effective use of spearman's and kendall's correlation coefficients for association between two measured traits », *Animal Behaviour*, vol. 102, p. 77–84, 2015.

[25] Z.-T. Liu, M. Wu, W.-H. Cao, J.-W. Mao, J.-P. Xu et G.-Z. Tan, « Speech emotion recognition based on feature selection and extreme learning machine decision tree », *Neurocomputing*, vol. 273, p. 271–280, 2018.

[26] V. Nasteski, « An overview of the supervised machine learning methods », *Horizons. b*, vol. 4, p. 51–62, 2017.

[27] S. Vieira, W. H. L. Pinaya et A. Mechelli, « Introduction to machine learning », *in Machine learning*, p. 1–20, Elsevier, 2020.

[28] S. Ray, « A quick review of machine learning algorithms », *in 2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon)*, p. 35–39, IEEE, 2019.

[29] B. Remeseiro et V. Bolon-Canedo, « A review of feature selection methods in medical applications », *Computers in biology and medicine*, vol. 112, p. 103375, 2019.

[30] O. Maimon et L. Rokach, « Introduction to supervised methods », *Data mining and knowledge discovery handbook*, p. 149–164, 2005.

[31] M. Ippolito, J. Ferguson et F. Jenson, « Improving facies prediction by combining supervised and unsupervised learning methods », *Journal of Petroleum Science and Engineering*, vol. 200, p. 108300, 2021.

[32] A. Mechelli, A. Lin, S. Wood, P. McGorry, P. Amminger, S. Tognin, P. McGuire, J. Young, B. Nelson et A. Yung, « Using clinical information to make individualized prognostic predictions in people at ultra high risk for psychosis », *Schizophrenia research*, vol. 184, p. 32–38, 2017.

[33] Z. Ghahramani, « Unsupervised learning », *in Summer school on machine learning*, p. 72–112, Springer, 2003.

[34] S. Badillo, B. Bánfai, F. Birzele, I. I. Davydov, L. Hutchinson, T. Kam-Thong, J. Siebourg-Polster, B. Steiert et J. D. Zhang, « An introduction to machine learning », *Clinical Pharmacology and Therapeutics*, vol. 107, p. 871 – 885, 2020.

[35] E. Oja, « Unsupervised learning in neural computation », *Theoretical computer science*, vol. 287, no. 1, p. 187–207, 2002.

[36] A. Ławrynowicz et V. Tresp, « Introducing machine learning », *Perspectives on Ontology Learning*, vol. 12, p. 19, 2014.

[37] O. Kramer, *Dimensionality reduction with unsupervised nearest neighbors*, vol. 51. Springer, 2013.

[38] R. Y. Choi, A. S. Coyner, J. Kalpathy-Cramer, M. F. Chiang et J. P. Campbell, « Introduction to machine learning, neural networks, and deep learning », *Translational vision science & technology*, vol. 9, no. 2, p. 14–14, 2020.

[39] Y. Reddy, P. Viswanath et B. E. Reddy, « Semi-supervised learning : A brief review », *Int. J. Eng. Technol*, vol. 7, no. 1.8, p. 81, 2018.

[40] R. Sabatier, C. Reynes et M. Vivien, « Grain 7 : Régression linéaire », *Chemoocs, Session*, vol. 1, 2016.

[41] A. Guyader, « Régression linéaire », *Université Rennes*, vol. 2, p. 60–61, 2011.

[42] T. P. Ryan, *Modern regression methods*, vol. 655. John Wiley & Sons, 2008.

[43] D. Birkes et Y. Dodge, *Alternative methods of regression*. John Wiley & Sons, 2011.

[44] A. B. Owen, « A robust hybrid of lasso and ridge regression », *Contemporary Mathematics*, vol. 443, no. 7, p. 59–72, 2007.

[45] N. Boyko et O. Moroz, « Comparative analysis of regression regularization methods for life expectancy prediction. », *in MoMLeT+ DS*, p. 310–326, 2021.

[46] S. Mensah, J. Keung, M. F. Bosu et K. E. Bennin, « Duplex output software effort estimation model with self-guided interpretation », *Information and Software Technology*, vol. 94, p. 1–13, 2018.

[47] V. Jakkula, « Tutorial on support vector machine (svm) », *School of EECS, Washington State University*, vol. 37, no. 2.5, p. 3, 2006.

[48] M. Hasan et F. Boris, « Svm : Machines à vecteurs de support ou séparateurs à vastes marges », *Rapport technique, Versailles St Quentin, France. Cité*, vol. 64, 2006.

[49] F. Lauer et G. Bloch, « Méthodes svm pour l'identification », *in Journées Identification et Modélisation Expérimentale (JIME'2006)*, p. CDROM, 2006.

[50] R. Trinchero, M. Larbi, H. M. Torun, F. G. Canavero et M. Swaminathan, « Machine learning and uncertainty quantification for surrogate models of integrated devices with a large number of parameters », *IEEE Access*, vol. 7, p. 4056–4066, 2018.

[51] X. Yao, A. Panaye, J.-P. Doucet, R. Zhang, H. Chen, M. Liu, Z. Hu et B. T. Fan, « Comparative study of qsar/qspr correlations using support vector machines, radial basis function neural networks, and multiple linear regression », *Journal of chemical information and computer sciences*, vol. 44, no. 4, p. 1257–1266, 2004.

[52] V. Rodriguez-Galiano, M. Sanchez-Castillo, M. Chica-Olmo et M. Chica-Rivas, « Machine learning predictive models for mineral prospectivity : An evaluation of neural networks, random forest, regression trees and support vector machines », *Ore Geology Reviews*, vol. 71, p. 804–818, 2015.

[53] M. O. Faruqe et M. A. M. Hasan, « Face recognition using pca and svm », *in 2009 3rd international conference on anti-counterfeiting, security, and identification in communication*, p. 97–101, IEEE, 2009.

[54] V. Cherkassky et Y. Ma, « Practical selection of svm parameters and noise estimation for svm regression », *Neural networks*, vol. 17, no. 1, p. 113–126, 2004.

[55] L. Breiman, « Random forests », *Machine learning*, vol. 45, p. 5–32, 2001.

[56] M. Denil, D. Matheson et N. De Freitas, « Narrowing the gap : Random forests in theory and in practice », *in International conference on machine learning*, p. 665–673, PMLR, 2014.

[57] G. Biau et E. Scornet, « A random forest guided tour », *Test*, vol. 25, p. 197–227, 2016.

[58] V. Svetnik, A. Liaw, C. Tong et T. Wang, « Application of breiman's random forest to modeling structure-activity relationships of pharmaceutical molecules », *in Multiple Classifier Systems : 5th International Workshop, MCS 2004, Cagliari, Italy, June 9-11, 2004. Proceedings 5*, p. 334–343, Springer, 2004.

[59] L. Rokach, « Decision forest : Twenty years of research », *Information Fusion*, vol. 27, p. 111–125, 2016.

[60] S. Caron, « Une introduction aux arbres de décision », *Stéphane Caron*, vol. 31, 2011.

[61] D. Huang, « An optimized method for battery swapping demand prediction based on random forest regression », *in 2021 IEEE 5th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 5, p. 1739–1743, IEEE, 2021.

[62] L. Xue, Y. Liu, Y. Xiong, Y. Liu, X. Cui et G. Lei, « A data-driven shale gas production forecasting method based on the multi-objective random forest

regression », *Journal of Petroleum Science and Engineering*, vol. 196, p. 107801, 2021.

[63] Y. Zhao, X. Zhao, L. Yan, Z. Liu, Q. Liu, C. Liu, H. Zhou et K. Huang, « Reconstruction of the statistical characteristics of electric fields in enclosures with an aperture based on random forest regression », *IEEE Transactions on Electromagnetic Compatibility*, vol. 62, no. 4, p. 1151–1159, 2019.

[64] D. Borup, B. J. Christensen, N. S. Mühlbach et M. S. Nielsen, « Targeting predictors in random forest regression », *International Journal of Forecasting*, vol. 39, no. 2, p. 841–868, 2023.

[65] M. Atlam, H. Torkey, N. El-Fishawy et H. Salem, « Coronavirus disease 2019 (covid-19) : Survival analysis using deep learning and cox regression model », *Pattern Analysis and Applications*, vol. 24, p. 993–1005, 2021.

[66] N. Ata et M. T. Sözer, « Cox regression models with nonproportional hazards applied to lung cancer survival data », *Hacettepe Journal of Mathematics and Statistics*, vol. 36, no. 2, p. 157–167, 2007.

Annexe A

Importation et Transformation des données

```
    # importation du module pandas
import pandas as pd

    # Importation et transformation de la base de découverte clinique
df_discovery = pd.read_csv('/Users/cheikhsall/Documents/DOSSIER_
SUJET_DE_RECHERCHE/PROJET_MEMOIRE/data/nou/Discovery-Clinical.csv')
df_discovery.head(2)
# Renommer les noms des colonne de la base de découverte clinique
df_discovery = df_discovery.rename(columns='Unnamed : 0' : 'index_ID', 'x' : 'days_to_death')

    # Liste des variables à supprimer
cols_to_drop = ['index_ID']

    # Supprimer les variables de la liste
df_discovery = df_discovery.drop(cols_to_drop, axis=1)
```

```
df_discovery.head()

    # Importation et transformation de la base de validation clinique



    # Charger le fichier CSV df_validation = pd.read_csv('/Users/cheikhsall/Documents/DOSSIER_
PROJET_MEMOIRE/data/nou/Validation-Clinical.csv')

    df_validation = df_validation.rename(columns='Unnamed : 0' : 'index_ID', 'x' :
'days_to_death')

    # Liste des variables à supprimer
cols_to_drop = ['index_ID']

    # Supprimer les variables de la liste
df_validation = df_validation.drop(cols_to_drop, axis=1)

    df_validation.head()

    # Importation des données génétique
# Importation du module pyreadr
import pyreadr

    Discovery_Genomics = pyreadr.read_r('/Users/cheikhsall/Documents/DOSSIER_SUJET_DE_
RECHERCHE/PROJET_MEMOIRE/data/nou/Discovery-Genomics.RData')
Discovery_Genomics = Discovery_Genomics['edata.discovery']
Validation_Genomics = pyreadr.read_r('/Users/cheikhsall/Documents/DOSSIER_SUJET_DE_
RECHERCHE/PROJET_MEMOIRE/data/nou/Validation-Genomics.RData')
Validation_Genomics = Validation_Genomics['edata.validation']
```

```
    # Transposition des deux DataFrames en utilisant l'attribut "T"
Discovery_Genomics = Discovery_Genomics.T
Validation_Genomics = Validation_Genomics.T



    Discovery_Genomics.reset_index(drop=True, inplace=True)
Validation_Genomics.reset_index(drop=True, inplace=True)
# concaténation des deux dataframes en ajoutant les lignes l'une après l'autre
Discovery_Genomics_concat = pd.concat([Discovery_Genomics, df_discovery], axis=1)
Validation_Genomics_concat= pd.concat([Validation_Genomics, df_validation], axis=1)
```

Annexe B

Analyse exploratoire des données

```
    # importation des module import numpy as np
import pandas as pd
import seaborn as sns
import random
import plotly.graph_objects as go
import plotly.express as px
import scipy.stats as stt
import matplotlib.pyplot as plt



    # Calcul des statatistiques descriptives de la variable de survie



    Discovery_Genomics_concat['days_to_death'].describe()
# affichage des résultats print("Minimum :", percentiles[0])
print("25e percentile :", percentiles[1])
print("50e percentile (médiane) :", percentiles[2])
```

```
print("75e percentile :", percentiles[3])
print("Maximum :", percentiles[4])



    # Construction de l'histogramme de distribution de la variable de survie
# Supprimer les valeurs manquantes ou infinies
data = Discovery_Genomics_concat['days_to_death'].dropna().replace([np.inf, -np.inf],
np.nan)

    # Convertir les valeurs restantes en entiers
data = data.astype(int)

    # Tracé de l'histogramme avec une densité lissée
sns.histplot(data, bins=range(0, max(data)+500, 500), kde=True, color='blue', ed-
gecolor='black')

    # Ajouter les étiquettes d'effectif pour chaque barre
for count, bin_edge in zip(np.histogram(data, bins=range(0, max(data)+500, 500))[0],
range(0, max(data)+500, 500)[ :-1]) : plt.text(bin_edge + (500 / 2), count, str(int(count)),
ha='center', va='bottom', fontsize=8)

    # Mise en forme du graphique
plt.xlabel('Days_to_death') plt.ylabel('Effectif ') plt.title('Histogramme de la Durée
de Survie ')

    # Affichage du graphique
plt.show()

    # Analyse de survie
import pandas as pd
```

```
from lifelines import CoxPHFitter
from lifelines.plotting import plot_lifetimes
import matplotlib.pyplot as plt


    df = Discovery_Genomics_concat[sorted_test_results_features]


    # Remplacement des valeurs NaN par la médiane de chaque colonne
df = df.fillna(df.median())

    # Préparation des données pour l'analyse de survie
X = df.drop("days_to_death", axis=1)
y = df["days_to_death"]]


    # Initialisation de CoxPHFitter
cph = CoxPHFitter()


    # Ajustement du modèle de risques proportionnels de Cox
cph.fit(pd.concat([X, y], axis=1), duration_col='days_to_death')

    kmf = KaplanMeierFitter()
kmf.fit(durations=y["days_to_death"], event_observed=y["days_to_death"] > 0)

    # Tracé de la courbe de survie plt.figure(figsize=(10, 6))
ax = plt.subplot(111)
kmf.plot(ax=ax)
```

```
    plt.title("Courbe de survie ")
plt.xlabel("Temps")
plt.ylabel("Probabilité de survie")
plt.show()
```

Annexe C

Sélection de caractéristique de Pearson et modèle d'apprentissage automatique

# Corrélation de pearson # Sélectionner la variable quantitative à étudier
target_variable = 'days_to_death'

# Calculer la corrélation avec toutes les autres variables quantitatives
corr_matrix = Discovery_Genomics_concat.select_dtypes(include=['float64', 'int64']).corr()
# Sélectionner la colonne correspondante à la variable étudiée et trier par ordre décroissant
corr_values = corr_matrix[target_variable].abs().sort_values(ascending=False)

# Récupération des 100 variables les plus corrélées avec la variable quantitative
corr_values_first100=corr_values[:101] corr_values_first100

Modèle régression linéaire

```
    import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
import numpy as np
import matplotlib.pyplot as plt


    # Données de validation validation_data = Validation_Genomics_concat.dropna()

    # Données de découverte discovery_data = Discovery_Genomics_concat.dropna()

    # Les caractéristiques les plus importantes
top_features = column_names

    # Définition du nombre maximum de caractéristiques (100 dans cet exemple)
max_features = 100

    # Créer des tableaux pour stocker les résultats
num_features = np.arange(1, max_features + 1)
correlation_values = np.zeros(max_features)
correlation_val_values = np.zeros(max_features)
num_variables = []
performance = []
# Stocker les valeurs MAE
mae_values = []
```

# Définir le nombre de folds pour la validation croisée
k = 10

# Fixer la reproductibilité
np.random.seed(123)

Diviser les données en ensembles d'entraînement et de test
train_data, test_data = train_test_split(discovery_data, test_size=0.2)

Effectuer la validation croisée en k-fold
for i in num_features : # Sélectionner les i caractéristiques les plus importantes
selected_features = top_features[:i]

# Créer un sous-ensemble des données avec les caractéristiques sélectionnées
sub_train_data = train_data[['days_to_death'] + selected_features]

# Créer le modèle de régression linéaire
lm_model = LinearRegression()

# Effectuer une validation croisée en 10-fold sur l'ensemble d'entraînement
kfold = KFold(n_splits=10, random_state=42, shuffle=True) scores = cross_val_score(lm_model, sub_train_data[selected_features], sub_train_data['days_to_death'], cv=kfold) # utilise R^2 comme score

lm_model.fit(sub_train_data[selected_features], sub_train_data['days_to_death'])

```
    # Faire des prédictions sur les données de test
predictions = lm_model.predict(test_data[selected_features])

    # Calculer la concordance index entre les valeurs prédites et réelles sur l'ensemble
de test
concordance = concordance_index(predictions, test_data['days_to_death']) # Calculer
la corrélation entre les valeurs prédites et réelles sur l'ensemble de test
correlation = np.corrcoef(predictions, test_data['days_to_death'])[0, 1] # Stocker la
valeur de corrélation dans le tableau correspondant
correlation_values[i-1] = correlation

    # Prédictions sur les données de validation #


    # Faire des prédictions sur les données de validation
predictions_val = lm_model.predict(validation_data[selected_features])

    # Calculer la corrélation entre les valeurs prédites et réelles sur l'ensemble de va-
lidation
correlation_val = np.corrcoef(predictions_val, validation_data['days_to_death'])[0, 1]
# Calculer la concordance index entre les valeurs prédites et réelles sur l'ensemble de
validation


    concordance_val = concordance_index(predictions_val, validation_data['days_to_death'])


    # Stocker la valeur de corrélation dans le tableau correspondant
correlation_val_values[i-1] = correlation_val
```

```
    # Calculer le score de performance moyen
avg_score = np.mean(scores)



    # Ajouter le nombre de variables et la performance correspondante aux listes
num_variables.append(i)
performance.append(avg_score)
mae_values.append(mean_absolute_error(test_data['days_to_death'], predictions))



    # Tracer le graphique
plt.plot(num_variables, performance, marker='o')
plt.xlabel('Nombre de Variables')
plt.ylabel('R2')
plt.title('Performance vs. Nombre de Variables')
plt.grid(True)
plt.show()
print(f"MAE Moyenne : np.mean(mae_values)")



    # Affichage du performance
import numpy as np
# Trouver la performance maximale
max(performance)



    # Trouver l'index de la valeur maximale
max_index = np.argmax(performance) +1

    # Affichage l'index et performance maximale
```

```
print(max_index)
print(max(performance))


    import matplotlib.pyplot as plt
import pandas as pd


    # Créer un dataframe avec les num_features et les valeurs de corrélation
df = pd.DataFrame('num_features' : num_features, 'correlation_values' : correlation_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['correlation_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Correlation')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Créer un dataframe avec les num_features et les valeurs de l'indice de concor-
dance
df = pd.DataFrame('num_features' : num_features, 'concordance_values' : concordance_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['concordance_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Concordance index')
```

```
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Modèle régression Ridge


    import numpy as np
from sklearn.model_selection import train_test_split,
KFold, cross_val_score
from sklearn.metrics import r2_score, mean_absolute_error
from sklearn.linear_model import Ridge
import matplotlib.pyplot as plt


    # Données de validation
validation_data = Validation_Genomics_concat.dropna()


    # Données de découverte
discovery_data = Discovery_Genomics_concat.dropna()


    # Caractéristiques importantes top_features = column_names


    # Définition du nombre maximal de caractéristiques (100 dans cet exemple)
max_features = 100
```

```
    # Création des tableaux pour stocker les résultats
num_features = np.arange(1, max_features+1)
correlation_values = np.zeros(max_features)
correlation_val_values = np.zeros(max_features)



    num_variables = []
performance = []
mae_values = [] # Stocker les valeurs MAE



    # Nombre de plis pour la validation croisée
k = 10



    # Graine pour la reproductibilité
np.random.seed(123)



    # Division des données en ensembles d'entraînement et de test
train_data, test_data = train_test_split(discovery_data, test_size=0.2, random_state=123)



    # Validation croisée en k plis for i in num_features :
# Sélection des i meilleures caractéristiques selected_features = top_features[ :i]



    if len(selected_features) ¡= 1 :

    sub_train_data = train_data[['days_to_death'] + selected_features] x_train = sub_train_data[selecte
```

```
y_train = sub_train_data['days_to_death'].to_numpy()

    # Régression Ridge
ridge_model = Ridge(alpha=1.0) # ajustement de la valeur de alpha pour la régularisation
ridge_model.fit(x_train, y_train)



    # Validation croisée
kfold = KFold(n_splits=10, random_state=42, shuffle=True)
scores = cross_val_score(ridge_model,
sub_train_data[selected_features],
sub_train_data['days_to_death'], cv=kfold)



    # Prédictions sur les données de test
x_test = test_data[selected_features].to_numpy()
predictions = ridge_model.predict(x_test)



    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
test
correlation = np.corrcoef(predictions, test_data['days_to_death'].to_numpy())[0, 1]
correlation_values[i-1] = correlation



    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de test
concordance = concordance_index(predictions, test_data['days_to_death'].to_numpy())
concordance_values[i-1] = concordance
```

```
    # Prédictions sur les données de validation
x_val = validation_data[selected_features].to_numpy()
predictions_val = ridge_model.predict(x_val)


    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
validation
correlation_val = np.corrcoef(predictions_val, validation_data['days_to_death'].to_numpy())[0,
1] correlation_val_values[i-1] = correlation_val

    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de validation
concordance_val = concordance_index(predictions_val, validation_data['days_to_death'].to_numpy())
concordance_val_values[i-1] = concordance_val


    # Calcul de la performance moyenne
avg_score = np.mean(scores1)


    # Ajout du nombre de variables et de la performance correspondante aux listes
num_variables.append(i)
performance.append(avg_score1)
mae_values.append(mean_absolute_error(test_data['days_to_death'], predictions))


    # Tracer le graphique
plt.plot(num_variables, performance, marker='o')
plt.xlabel('Nombre de Variables')
plt.ylabel('R2')
```

```
plt.title('Performance vs. Nombre de Variables')
plt.grid(True)


    # Affichage de la valeur moyennes de MAE
print(f"MAE moyen : np.mean(mae_values)")


    plt.show()


    # Affichage du performance
import numpy as np
# Trouver la performance maximale
max(performance)


    # Trouver l'index de la valeur maximale
max_index = np.argmax(performance) +1

    # Affichage l'index et performance maximale
print(max_index)
print(max(performance))


    import matplotlib.pyplot as plt
import pandas as pd


    # Créer un dataframe avec les num_features et les valeurs de corrélation
```

```
df = pd.DataFrame('num_features' : num_features, 'correlation_values' : correlation_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['correlation_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Correlation')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Créer un dataframe avec les num_features et les valeurs de l'indice de concor-
dance
df = pd.DataFrame('num_features' : num_features, 'concordance_values' : concordance_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['concordance_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Concordance index')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Modèle régression Lasso

    import numpy as np
from sklearn.model_selection import train_test_split,
```

```
KFold, cross_val_score
from sklearn.metrics import r2_score, mean_absolute_error
from sklearn.linear_model import Lasso
import matplotlib.pyplot as plt


    # Données de validation
validation_data = Validation_Genomics_concat.dropna()


    # Données de découverte
discovery_data = Discovery_Genomics_concat.dropna()


    # Caractéristiques importantes top_features = column_names


    # Définition du nombre maximal de caractéristiques
max_features = 100


    # Création des tableaux pour stocker les résultats
num_features = np.arange(1, max_features+1)
correlation_values = np.zeros(max_features)
correlation_val_values = np.zeros(max_features)


    num_variables = []
performance = []
mae_values = [] # Stocker les valeurs MAE
```

# Nombre de plis pour la validation croisée
k = 10

# Graine pour la reproductibilité
np.random.seed(123)

Division des données en ensembles d'entraînement et de test
train_data, test_data = train_test_split(discovery_data, test_size=0.2, random_state=123)

Validation croisée en k plis for i in num_features :
Sélection des i meilleures caractéristiques selected_features = top_features[:i]

if len(selected_features) ¡= 1 :

sub_train_data = train_data[['days_to_death'] + selected_features] x_train = sub_train_data[selecte
y_train = sub_train_data['days_to_death'].to_numpy()

# Régression Lasso
lasso_model = Lasso(alpha=1.0) # ajustement de la valeur de alpha pour la régularisation
lasso_model.fit(x_train, y_train)

# Validation croisée
kfold = KFold(n_splits=10, random_state=42, shuffle=True)

```
scores = cross_val_score(lasso_model,
sub_train_data[selected_features],
sub_train_data['days_to_death'], cv=kfold)


    # Prédictions sur les données de test
x_test = test_data[selected_features].to_numpy()
predictions = lasso_model.predict(x_test)


    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
test
correlation = np.corrcoef(predictions, test_data['days_to_death'].to_numpy())[0, 1]
correlation_values[i-1] = correlation


    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de test
concordance = concordance_index(predictions, test_data['days_to_death'].to_numpy())
concordance_values[i-1] = concordance


    # Prédictions sur les données de validation
x_val = validation_data[selected_features].to_numpy()
predictions_val = lasso_model.predict(x_val)


    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
validation
correlation_val = np.corrcoef(predictions_val, validation_data['days_to_death'].to_numpy())[0,
```

```
1] correlation_val_values[i-1] = correlation_val

    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de validation
concordance_val = concordance_index(predictions_val, validation_data['days_to_death'].to_numpy())
concordance_val_values[i-1] = concordance_val



    # Calcul de la performance moyenne
avg_score = np.mean(scores1)



    # Ajout du nombre de variables et de la performance correspondante aux listes
num_variables.append(i)
performance.append(avg_score1)
mae_values.append(mean_absolute_error(test_data['days_to_death'], predictions))



    # Tracer le graphique
plt.plot(num_variables, performance, marker='o')
plt.xlabel('Nombre de Variables')
plt.ylabel('R2')
plt.title('Performance vs. Nombre de Variables')
plt.grid(True)



    # Affichage de la valeur moyennes de MAE
print(f"MAE moyen : np.mean(mae_values)")
```

```
    plt.show()



    # Affichage du performance
import numpy as np
# Trouver la performance maximale
max(performance)



    # Trouver l'index de la valeur maximale
max_index = np.argmax(performance) +1

    # Affichage l'index et performance maximale
print(max_index)
print(max(performance))



    import matplotlib.pyplot as plt
import pandas as pd



    # Créer un dataframe avec les num_features et les valeurs de corrélation
df = pd.DataFrame('num_features' : num_features, 'correlation_values' : correlation_values)



    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['correlation_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Correlation')
plt.title('Performance vs. Nombre de variable')
```

```
plt.grid(True)
plt.show()


    # Créer un dataframe avec les num_features et les valeurs de l'indice de concor-
dance
df = pd.DataFrame('num_features' : num_features, 'concordance_values' : concordance_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['concordance_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Concordance index')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Modèle régresion Elastique Net


    import numpy as np
from sklearn.model_selection import train_test_split,
KFold, cross_val_score
from sklearn.metrics import r2_score,
mean_absolute_error
from sklearn.linear_model import ElasticNet
import matplotlib.pyplot as plt
```

```
    # Données de validation
validation_data = Validation_Genomics_concat.dropna()



    # Données de découverte
discovery_data = Discovery_Genomics_concat.dropna()



    # Caractéristiques importantes top_features = column_names



    # Définition du nombre maximal de caractéristiques
max_features = 100



    # Création des tableaux pour stocker les résultats
num_features = np.arange(1, max_features+1)
correlation_values = np.zeros(max_features)
correlation_val_values = np.zeros(max_features)



    num_variables = []
performance = []
mae_values = [] # Stocker les valeurs MAE



    # Nombre de plis pour la validation croisée
k = 10
```

```
    # Graine pour la reproductibilité
np.random.seed(123)



    # Division des données en ensembles d'entraînement et de test
train_data, test_data = train_test_split(discovery_data, test_size=0.2, random_state=123)



    # Validation croisée en k plis for i in num_features :
# Sélection des i meilleures caractéristiques selected_features = top_features[ :i]



    if len(selected_features) ¡= 1 :

    sub_train_data = train_data[['days_to_death'] + selected_features] x_train = sub_train_data[selecte
y_train = sub_train_data['days_to_death'].to_numpy()

    # Régression Élastique net
elastic_net_model = ElasticNet(alpha=1.0, l1_ratio=0.5) # ajustement de la valeur
de alpha et de l1_radio pour la régularisation
elastic_net_model.fit(x_train, y_train)



    # Validation croisée
kfold = KFold(n_splits=10, random_state=42, shuffle=True)
scores = cross_val_score( elastic_net_model,
sub_train_data[selected_features],
sub_train_data['days_to_death'], cv=kfold)
```

```
    # Prédictions sur les données de test
x_test = test_data[selected_features].to_numpy()
predictions = elastic_net_model.predict(x_test)


    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
test
correlation = np.corrcoef(predictions, test_data['days_to_death'].to_numpy())[0, 1]
correlation_values[i-1] = correlation


    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de test
concordance = concordance_index(predictions, test_data['days_to_death'].to_numpy())
concordance_values[i-1] = concordance


    # Prédictions sur les données de validation
x_val = validation_data[selected_features].to_numpy()
predictions_val = elastic_net_model.predict(x_val)


    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
validation
correlation_val = np.corrcoef(predictions_val, validation_data['days_to_death'].to_numpy())[0,
1] correlation_val_values[i-1] = correlation_val

    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de validation
concordance_val = concordance_index(predictions_val, validation_data['days_to_death'].to_numpy())
```

```
concordance_val_values[i-1] = concordance_val


    # Calcul de la performance moyenne
avg_score = np.mean(scores)


    # Ajout du nombre de variables et de la performance correspondante aux listes
num_variables.append(i)
performance.append(avg_score)
mae_values.append(mean_absolute_error(test_data['days_to_death'], predictions))


    # Tracer le graphique
plt.plot(num_variables, performance, marker='o')
plt.xlabel('Nombre de Variables')
plt.ylabel('R2')
plt.title('Performance vs. Nombre de Variables')
plt.grid(True)


    # Affichage de la valeur moyennes de MAE
print(f"MAE moyen : np.mean(mae_values)")


    plt.show()


    # Affichage du performance
import numpy as np
```

```
# Trouver la performance maximale
max(performance)



    # Trouver l'index de la valeur maximale
max_index = np.argmax(performance) +1

    # Affichage l'index et performance maximale
print(max_index)
print(max(performance))



    import matplotlib.pyplot as plt
import pandas as pd



    # Créer un dataframe avec les num_features et les valeurs de corrélation
df = pd.DataFrame('num_features' : num_features, 'correlation_values' : correlation_values)



    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['correlation_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Correlation')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()



    # Créer un dataframe avec les num_features et les valeurs de l'indice de concor-
```

```
dance
df = pd.DataFrame('num_features' : num_features, 'concordance_values' : concordance_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['concordance_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Concordance index')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Modèle SVR cas linéaire


    import numpy as np
from sklearn.model_selection import train_test_split,
KFold, cross_val_score
from sklearn.metrics import r2_score,
mean_absolute_error
from sklearn.svm import LinearSVR
import matplotlib.pyplot as plt


    # Données de validation
validation_data = Validation_Genomics_concat.dropna()


    # Données de découverte
```

```
discovery_data = Discovery_Genomics_concat.dropna()


    # Caractéristiques importantes top_features = column_names


    # Définition du nombre maximal de caractéristiques
max_features = 100


    # Création des tableaux pour stocker les résultats
num_features = np.arange(1, max_features+1)
correlation_values = np.zeros(max_features)
correlation_val_values = np.zeros(max_features)


    num_variables = []
performance = []
mae_values = [] # Stocker les valeurs MAE


    # Nombre de plis pour la validation croisée
k = 10


    # Graine pour la reproductibilité
np.random.seed(123)


    # Division des données en ensembles d'entraînement et de test
```

train_data, test_data = train_test_split(discovery_data, test_size=0.2, random_state=123)

Validation croisée en k plis for i in num_features :
Sélection des i meilleures caractéristiques selected_features = top_features[:i]

if len(selected_features) ¡= 1 :

sub_train_data = train_data[['days_to_death'] + selected_features] x_train = sub_train_data[selecte
y_train = sub_train_data['days_to_death'].to_numpy()

# SVR cas linéaire
svm_model = LinearSVR()
svm_model.fit(x_train, y_train)

# Validation croisée
kfold = KFold(n_splits=10, random_state=42, shuffle=True)
scores = cross_val_score(svm_model,
sub_train_data[selected_features],
sub_train_data['days_to_death'], cv=kfold)

# Prédictions sur les données de test
x_test = test_data[selected_features].to_numpy()
predictions = svm_model.predict(x_test)

Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de

```
test
correlation = np.corrcoef(predictions, test_data['days_to_death'].to_numpy())[0, 1]
correlation_values[i-1] = correlation


    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de test
concordance = concordance_index(predictions, test_data['days_to_death'].to_numpy())
concordance_values[i-1] = concordance


    # Prédictions sur les données de validation
x_val = validation_data[selected_features].to_numpy()
predictions_val = svm_model.predict(x_val)


    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
validation
correlation_val = np.corrcoef(predictions_val, validation_data['days_to_death'].to_numpy())[0,
1] correlation_val_values[i-1] = correlation_val

    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de validation
concordance_val = concordance_index(predictions_val, validation_data['days_to_death'].to_numpy())
concordance_val_values[i-1] = concordance_val


    # Calcul de la performance moyenne
avg_score = np.mean(scores)
```

```
    # Ajout du nombre de variables et de la performance correspondante aux listes
num_variables.append(i)
performance.append(avg_score)
mae_values.append(mean_absolute_error(test_data['days_to_death'], predictions))



    # Tracer le graphique
plt.plot(num_variables, performance, marker='o')
plt.xlabel('Nombre de Variables')
plt.ylabel('R2')
plt.title('Performance vs. Nombre de Variables')
plt.grid(True)



    # Affichage de la valeur moyennes de MAE
print(f"MAE moyen : np.mean(mae_values)")



    plt.show()



    # Affichage du performance
import numpy as np
# Trouver la performance maximale
max(performance)



    # Trouver l'index de la valeur maximale
max_index = np.argmax(performance) +1
```

```
    # Affichage l'index et performance maximale
print(max_index)
print(max(performance))



    import matplotlib.pyplot as plt
import pandas as pd



    # Créer un dataframe avec les num_features et les valeurs de corrélation
df = pd.DataFrame('num_features' : num_features, 'correlation_values' : correlation_values)



    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['correlation_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Correlation')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()



    # Créer un dataframe avec les num_features et les valeurs de l'indice de concor-
dance
df = pd.DataFrame('num_features' : num_features, 'concordance_values' : concordance_values)



    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['concordance_values'], marker='o')
plt.xlabel('Number of Features')
```

```
plt.ylabel('Concordance index')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Modèle SVR cas non linéaire


    import numpy as np
from sklearn.model_selection import train_test_split,
KFold, cross_val_score
from sklearn.metrics import r2_score,
mean_absolute_error
from sklearn.svm import LinearSVR
import matplotlib.pyplot as plt


    # Données de validation
validation_data = Validation_Genomics_concat.dropna()


    # Données de découverte
discovery_data = Discovery_Genomics_concat.dropna()


    # Caractéristiques importantes top_features = column_names


    # Définition du nombre maximal de caractéristiques
```

```
max_features = 100


    # Création des tableaux pour stocker les résultats
num_features = np.arange(1, max_features+1)
correlation_values = np.zeros(max_features)
correlation_val_values = np.zeros(max_features)


    num_variables = []
performance = []
mae_values = [] # Stocker les valeurs MAE


    # Nombre de plis pour la validation croisée
k = 10


    # Graine pour la reproductibilité
np.random.seed(123)


    # Division des données en ensembles d'entraînement et de test
train_data, test_data = train_test_split(discovery_data, test_size=0.2, random_state=123)


    # Validation croisée en k plis for i in num_features :
# Sélection des i meilleures caractéristiques selected_features = top_features[ :i]
```

```
    if len(selected_features) ¡= 1 :

    sub_train_data = train_data[['days_to_death'] + selected_features] x_train = sub_train_data[selecte
y_train = sub_train_data['days_to_death'].to_numpy()

    # SVR cas non linéaire
svm_model = SVR(kernel='rbf')
svm_model.fit(x_train, y_train)



    # Validation croisée
kfold = KFold(n_splits=10, random_state=42, shuffle=True)
scores = cross_val_score(svm_model,
sub_train_data[selected_features],
sub_train_data['days_to_death'], cv=kfold)



    # Prédictions sur les données de test
x_test = test_data[selected_features].to_numpy()
predictions = svm_model.predict(x_test)



    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
test
correlation = np.corrcoef(predictions, test_data['days_to_death'].to_numpy())[0, 1]
correlation_values[i-1] = correlation



    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de test
```

concordance = concordance_index(predictions, test_data['days_to_death'].to_numpy())
concordance_values[i-1] = concordance

# Prédictions sur les données de validation
x_val = validation_data[selected_features].to_numpy()
predictions_val = svm_model.predict(x_val)

# Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de validation
correlation_val = np.corrcoef(predictions_val, validation_data['days_to_death'].to_numpy())[0, 1] correlation_val_values[i-1] = correlation_val

# Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble de validation
concordance_val = concordance_index(predictions_val, validation_data['days_to_death'].to_numpy())
concordance_val_values[i-1] = concordance_val

# Calcul de la performance moyenne
avg_score = np.mean(scores)

# Ajout du nombre de variables et de la performance correspondante aux listes
num_variables.append(i)
performance.append(avg_score)
mae_values.append(mean_absolute_error(test_data['days_to_death'], predictions))

```
    # Tracer le graphique
plt.plot(num_variables, performance, marker='o')
plt.xlabel('Nombre de Variables')
plt.ylabel('R2')
plt.title('Performance vs. Nombre de Variables')
plt.grid(True)


    # Affichage de la valeur moyennes de MAE
print(f"MAE moyen : np.mean(mae_values)")


    plt.show()


    # Affichage du performance
import numpy as np
# Trouver la performance maximale
max(performance)


    # Trouver l'index de la valeur maximale
max_index = np.argmax(performance) +1

    # Affichage l'index et performance maximale
print(max_index)
print(max(performance))


    import matplotlib.pyplot as plt
```

```
import pandas as pd


    # Créer un dataframe avec les num_features et les valeurs de corrélation
df = pd.DataFrame('num_features' : num_features, 'correlation_values' : correlation_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['correlation_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Correlation')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Créer un dataframe avec les num_features et les valeurs de l'indice de concor-
dance
df = pd.DataFrame('num_features' : num_features, 'concordance_values' : concordance_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['concordance_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Concordance index')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()
```

```
    # Modéle Forêt aléatoire


    from sklearn.model_selection import train_test_split,
KFold, cross_val_score
from sklearn.metrics import r2_score,
mean_absolute_error
from sklearn.ensemble import RandomForestRegressor
import matplotlib.pyplot as plt


    # Données de validation
validation_data = Validation_Genomics_concat.dropna()


    # Données de découverte
discovery_data = Discovery_Genomics_concat.dropna()


    # Caractéristiques importantes top_features = column_names


    # Définition du nombre maximal de caractéristiques
max_features = 100


    # Création des tableaux pour stocker les résultats
num_features = np.arange(1, max_features+1)
correlation_values = np.zeros(max_features)
correlation_val_values = np.zeros(max_features)
```

num_variables = []
performance = []
mae_values = [] # Stocker les valeurs MAE

# Nombre de plis pour la validation croisée
k = 10

# Graine pour la reproductibilité
np.random.seed(123)

Division des données en ensembles d'entraînement et de test
train_data, test_data = train_test_split(discovery_data, test_size=0.2, random_state=123)

Validation croisée en k plis for i in num_features :
Sélection des i meilleures caractéristiques selected_features = top_features[:i]

if len(selected_features) ¡= 1 :

sub_train_data = train_data[['days_to_death'] + selected_features] x_train = sub_train_data[selecte
y_train = sub_train_data['days_to_death'].to_numpy()

# Forêt aléatoire
rf_model = RandomForestRegressor()

```
rf_model.fit(x_train, y_train)


    # Validation croisée
kfold = KFold(n_splits=10, random_state=42, shuffle=True)
scores = cross_val_score( rf_model,
sub_train_data[selected_features],
sub_train_data['days_to_death'], cv=kfold)


    # Prédictions sur les données de test
x_test = test_data[selected_features].to_numpy()
predictions = rf_model.predict(x_test)


    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
test
correlation = np.corrcoef(predictions, test_data['days_to_death'].to_numpy())[0, 1]
correlation_values[i-1] = correlation


    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de test
concordance = concordance_index(predictions, test_data['days_to_death'].to_numpy())
concordance_values[i-1] = concordance


    # Prédictions sur les données de validation
x_val = validation_data[selected_features].to_numpy()
predictions_val = rf_model.predict(x_val)
```

```
    # Calcul de la corrélation entre les valeurs prédites et réelles sur l'ensemble de
validation
correlation_val = np.corrcoef(predictions_val, validation_data['days_to_death'].to_numpy())[0,
1] correlation_val_values[i-1] = correlation_val

    # Calcul de la concordance index entre les valeurs prédites et réelles sur l'ensemble
de validation
concordance_val = concordance_index(predictions_val, validation_data['days_to_death'].to_numpy())
concordance_val_values[i-1] = concordance_val


    # Calcul de la performance moyenne
avg_score = np.mean(scores)


    # Ajout du nombre de variables et de la performance correspondante aux listes
num_variables.append(i)
performance.append(avg_score)
mae_values.append(mean_absolute_error(test_data['days_to_death'], predictions))


    # Tracer le graphique
plt.plot(num_variables, performance, marker='o')
plt.xlabel('Nombre de Variables')
plt.ylabel('R2')
plt.title('Performance vs. Nombre de Variables')
plt.grid(True)
```

```
    # Affichage de la valeur moyennes de MAE
print(f"MAE moyen : np.mean(mae_values)")


    plt.show()


    # Affichage du performance
import numpy as np
# Trouver la performance maximale
max(performance)


    # Trouver l'index de la valeur maximale
max_index = np.argmax(performance) +1

    # Affichage l'index et performance maximale
print(max_index)
print(max(performance))


    import matplotlib.pyplot as plt
import pandas as pd


    # Créer un dataframe avec les num_features et les valeurs de corrélation
df = pd.DataFrame('num_features' : num_features, 'correlation_values' : correlation_values)


    # Tracer la performance en fonction du nombre de caractéristiques
```

```
plt.plot(df['num_features'], df['correlation_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Correlation')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Créer un dataframe avec les num_features et les valeurs de l'indice de concor-
dance
df = pd.DataFrame('num_features' : num_features, 'concordance_values' : concordance_values)


    # Tracer la performance en fonction du nombre de caractéristiques
plt.plot(df['num_features'], df['concordance_values'], marker='o')
plt.xlabel('Number of Features')
plt.ylabel('Concordance index')
plt.title('Performance vs. Nombre de variable')
plt.grid(True)
plt.show()


    # Autre modèle de Forêt aléatoire après évaluation des modèles


    # Sélectionner les i meilleures caractéristiques
selected_features = top_features[ :max_index]

    # Créer un sous-ensemble des données avec les caractéristiques sélectionnées
sub_train_data = train_data[['days_to_death'] + selected_features]
```

```
x_train = sub_train_data[selected_features].to_numpy()
y_train = sub_train_data['days_to_death'].to_numpy()


    # Nombre de plis pour la validation croisée
k = 10


    # Créer l'objet de contrôle pour la validation croisée k-fold
ctrl = KFold(n_splits=k)


    # Entraîner le modèle de forêt aléatoire avec validation croisée
rf_model = RandomForestRegressor()


    # Effectuer la validation croisée
cv_results = cross_validate(rf_model, x_train, y_train, cv=ctrl, scoring=('neg_mean_absolute_error',
'r2'), return_train_score=True)


    # Extraire l'erreur absolue moyenne (MAE) des résultats de validation croisée
mae = -np.mean(cv_results['test_neg_mean_absolute_error'])

    r2_train = np.mean(cv_results['train_r2'])


    # Ajuster le modèle sur l'ensemble des données d'entraînement
rf_model.fit(x_train, y_train)
```

```
    # Faire des prédictions sur les données de test
x_test = test_data[selected_features].to_numpy()
predictions = rf_model.predict(x_test)


    Stocker les valeurs R et MAE dans les vecteurs correspondants
rsquared_values = r2_train
performance = rsquared_values
find_out = np.sqrt(-mae)


    # Faire des prédictions sur les données de validation
x_val = validation_data[selected_features].to_numpy()
predictions_val = rf_model.predict(x_val)


    import matplotlib.pyplot as plt


    # Valeurs réelles de y_val
y_val = validation_data['days_to_death']


    # Créer un dataframe avec les prédictions et les valeurs réelles
prediction_data = pd.DataFrame('Prédictions' :
predictions_val, 'Valeurs_Réelles' : y_val)


    # Tracer le nuage de points
plt.scatter(prediction_data['Valeurs_Réelles'],
```

```
prediction_data['Prédictions'])
plt.xlabel('Valeurs réelles')
plt.ylabel('Valeurs prédites')


    # Ajouter une ligne ajustée fit_line = np.polyfit(prediction_data['Valeurs_Réelles'],
prediction_data['Prédictions'], 1)
fit_function = np.poly1d(fit_line)
plt.plot(prediction_data['Valeurs_Réelles'],
fit_function(prediction_data['Valeurs_Réelles']),color='red', label='Ligne ajustée')


    Afficher la valeur de corrélation
plt.text(0.05, 0.6, f'R2 : r2_train :.2f', transform=plt.gca().transAxes)
plt.plot([min(y_val), max(y_val)], [min(y_val),
max(y_val)], linestyle='dashed')
plt.legend()
plt.show()
```

Annexe D

Sélection de caractéristiques de Spearman, de l'information mutuelle et les modèles d'apprentissage automatique.

```
    # Corrélation de Spearman
# Sélectionne des variables quantitative à étudier target_variable = 'days_to_death'

    # Calculer la corrélation de Spearman avec toutes les autres variables quantitatives
corr_matrix = Discovery_Genomics_concat.select_dtypes(include=['float64', 'int64']).corr(method='s
# Sélectionner la colonne correspondante à la variable étudiée et trier par ordre
décroissant.
corr_values = corr_matrix[target_variable].abs().sort_values(ascending=False)

    corr_values_first100=corr_values[ :101]
```

```
corr_values_first100


    # L'information mutuelle


    from sklearn.feature_selection import mutual_info_regression


    def select_top_features(X, y, top_n=100) :
# Calculer les scores d'information mutuelle
mi_scores = mutual_info_regression(X, y)


    # Créer un DataFrame avec les scores et les noms des caractéristiques
mi_scores_df = pd.DataFrame('Feature' : X.columns, 'MI_Score' : mi_scores)

    # Trier le DataFrame par score d'information mutuelle
top_features_df = mi_scores_df.sort_values(by='MI_Score', ascending=False).head(top_n)


    return top_features_df


    Discovery_Genomics_concat = Discovery_Genomics_concat.dropna()


    # Extraire les caractéristiques (X) et la variable cible (y)
X = Discovery_Genomics_concat.drop('days_to_death', axis=1).dropna() y = Disco-
very_Genomics_concat['days_to_death'].dropna()
```

Spécifier le nombre de caractéristiques supérieures que vous souhaitez (par exemple, 100) top_n_features = 100

# Sélectionner les N meilleures caractéristiques avec les scores d'information mutuelle les plus élevés
selected_features = select_top_features(X, y, top_n=top_n_features)

# Afficher les caractéristiques sélectionnées
print(selected_features)

De manière analogue pour les modèles d'apprentissage automatique dans la méthode de sélection de caractéristiques de Pearson.