N°d'ordre: .....

République Algerienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la Recherche scientifique
Université Mouloud Mammeri de Tizi Ouzou
Faculté des Sciences
Département de Mathématiques
Laboratoire LMPA

Mémoire de Master

Filière : Mathématiques
Spécialité : Mathémathiques appliquées à la gestion des entreprises

Par

Belkacem Razika

Titouche Narimane

Les algorithmes de machine learning pour la construction de modèles décisionnels ou prédictifs

Soutenue le Octobre 2022 devant le jury :

BennaniChrifa	MCB	UMMTO	Président du jury
Mousouni Samia	MAA	UMMTO	Examinateur
Abdouche Safia	MAA	UMMTO	Encadreur

Année Universitaire : 2021/2022

Remerciement

Avant tout, nous remercions Dieu de nous avoir aidé à réaliser ce travail. S'il faut beaucoup de mautivation, de rigueur et d'enthousiasme pour mener a bien ce mémoire, alors ce travail de recherche a eu besoin de la contribution de plusieurs personnes que nous tennont à remercier. Notre encadrant, Mme **Abdouche Safia** pour tout ses précieux conseils, pour son écoute active, nous la remercions vivement pour la qualité de son encadrement, pour sa patience, sa disponibilité durant l'élaboration de ce mémoire.

J'adresse mes remerciements les plus sincères aux membres du jury qui ont bien voulu examiné ce modèste travail .

Nous présentons également notre gratitude à tous les professeurs, chef de département, assistants de l'université de **Mouloud Maammeri** en général, et singulièrement ceux de la faculté de Science Dépaetement mathémathique pour leur dévouement.

Ainsi, nous remercions pour leur soutien tant moral, spirituel et matériel, nos parents sans laisser de côté nos frères et soeurs, amis et compagnons qui nous ont aidé, conseillé et encouragé ; trouvent ici l'expression de notre profonde reconnaissance.

Nous vous souhaitons une agréable lécture

Dédicaces

.... Nous dédions ce travail:

$\mathcal{A}$

Nos Parents
Pour tous leurs sacrifices, leurs soutiens,
Leurs encouragements et leurs amours qui ont été la raison de notre réussite.
Que dieu leur présente une bonne santé et une longue vie.

$\mathcal{A}$

Nos soeurs et Nos frères
Pour leur disponibilité à entendre nos frustrations et les sources de notre stress
Avec nos souhaits de bonheur et de réussite dans leur vie.

Qu'ils trouvent dans ce travail l'expression de nos sentiments les plus affectueux.

Sommaire

Table des figures

Liste des acronymes :

Ce mémoire contient un certain nombre d'acronymes, d'usage courant, employés le long de ce travail. Ces abréviations sont, explicitement, définies ci-dessous. Afin de faciliter la tâche du lecteur, le repérage de ces acronymes est établi par ordre alphabétique.

STA	Statistique et application
ML	Machine learning
KNN	K Nearest Neighbour
SVM	Support vecteur machine
RNA	Réseaux de Neurones Artificielles

Introduction général

A l'ère où la digitalisation est dans tout les domaines, de vastes ensembles de données sont accessibles à tout moment et en temps réel, ce contexte permet notamment à l'intelligence artificielle et au Machine Learning d'adopter une approche holistique du traitement de données, la technologie étant désormais assez poussée pour accéder à des quantités colossales d'informations et en assurer l'analyse. De ce fait, nombreuses sont les entreprises à se joindre à Google et Amazon afin d'implémenter des solutions machine learning dans leurs sociétés.

L'idée est d'automatiser certaines tâches quotidiennes qui impliquent beaucoup de temps et d'argent en injectant de l'intelligence aux processus. Et pour profiter pleinement du machine learning dans l'entreprise, il faut que la mise en place de ce système se fasse correctement. Cela permettra de résoudre plusieurs problèmes. Mais au préalable il est nécessaire de bien comprendre toutes les notions qui y sont liées pour assurer l'exécution efficace des application ML. Il est également important de bien connaître les types d'algorithmes disponibles et les différentes sortes de problèmes qu'ils sont capables de résoudre. c'est aussi une nécessité d'être bien situé sur les différentes sources des données internes et externes qui peuvent être volumineuse et complexes (big data).

Les entreprises qui opèrent à partir d'une plate-forme numérique, par exemple les détaillants en ligne et les réseaux sociaux, sont confrontées à de grands défis dans la capture, le stockage, l'analyse et la protection d'énorme volumes de données générés par leurs activités. La vitesse et la précision avec lesquelles ces grands ensembles de données peuvent être exploités pour apporter des informations qui soient pertinentes et utiles peuvent avoir un effet significatif sur la performance de l'entreprise (Keim et al., 2008). Il faut donc des systèmes qui permettent une gestion efficace de ces ensembles de données « big data ». Ainsi l'utilisation d'algorithmes de ML pour l'analyse de "big data" est une étape logique pour les entreprises qui cherchent à maximiser la valeur potentielle de leurs données.

CHAPITRE 1

GÉNÉRALITÉS SUR LE MACHINE LEARNING

1.1 Introdction

Le machine learning est une branche de l'intelligence artificielle qui repose sur l'idée que les systèmes informatiques peuvent apprendre des données, identifier des tendances et prendre des décisions avec un minimum d'intervention humaine.[1]
Le machine Learning (ML) ou l'apprentissage automatique est un domaine captivant à la croisée de nombreuses disciplines comme la statistique, les probabilités, l'optimisation, l'algorithmique et le traitement du signal. C'est un champ d'études en mutation constante qui s'est imposé dans notre société. Déjà utilisé depuis des décennies dans la reconnaissance automatique de caractères ou les filtres anti-spam, il sert maintenant à protéger contre la fraude bancaire, recommander des livres, films ou autres produits adaptés à nos goûts, identifier les visages dans le viseur de notre appareil photo, ou traduire automatiquement des textes d'une langue vers une autre.
Dans les années à venir, le machine Learning nous permettra vraisemblablement d'améliorer la sécurité routière notamment grâce aux véhicules autonomes, la réponse d'urgence aux catastrophes naturelles, le développement de nouveaux médicaments, ou l'efficacité énergétique de nos bâtiments et industries.[6]
On trouve la première définition du ML dès 1959, due à Arthur Samuel, l'un des pionniers de l'intelligence artificielle, qui définit le machine Learning comme le champ d'étude visant à donner la capacité à une machine d'apprendre sans être explicitement programmée [2]

En 1997, Tom Mitchell, de l'université de Carnegie Mellon, propose une définition plus précise : " On dit qu'un programme informatique apprend de l'expérience E en ce qui

concerne une tache T, et une mesure de performance P, si sa performance sur T, mesurée par P, s'améliore avec l'expérience E"[4]. Là où un programme traditionnel exécute des instructions, un algorithme de machine learning améliore ses performances au fur et à mesure de son apprentissage, mais aussi au fil de l'évolution du contexte et de réentrainements successifs. Plus on le "nourrit" de données, plus il devient précis.
Le Machine Learning peut également être définit comme un processus de résolution de problèmes pratiques par la collecte d'un ensemble de données et la construction algorithmique d'un modèle statistique basé sur cet ensemble de données dans un but prédictif et décisionnel [3].

En effet, construire des modèles prédictifs est le but ultime du Machine Learning, ils permet d'extraire des insights exploitables à partir de larges ensembles de données, afin d'offrir la possibilité de prendre de meilleures décisions stratégiques, en accord avec des objectifs prédéfinis [5], le Machine Learning constitue sans aucun doute le niveau supérieur de l'analyse de données.

1.2 Utilisation des algorithmes de ML :

- Filtrage de Courier électronique
- La vision artificielle
- Recommandation de produit sur les réseaux
- Publicité ciblée sur internet
- Reconnaissance d'image
- Génération de données
- Médecine (l'assistance au diagnostic et aux procédés chirurgicaux.)
- La conduite automatique
- Marketing
- Transactions boursières
- Détection d'anomalies, de fraudes, d'outliers

1.3 Type d'apprentissage automatique :

Il existe plusieurs façons d'apprendre automatiquement à partir des données dépendamment des problèmes à résoudre et des données disponibles, la figure 1 donne un sommaire des types d'apprentissage automatique les plus connus :

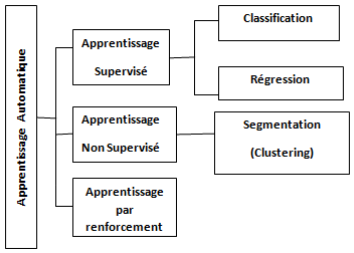


FIGURE 1 – Les grandes classes de l'apprentissage automatique

1.4 Apprentissage supervisé (supervised learning) :

L'algorithme est entrainé en utilisant une base de données d'apprentissage (échantillon d'apprentissage ou jeu de données) contenant des exemples de données qui contiennent à la fois les entrées x_i (variables explicatives ou features) et les sorties (variable à prédire, target ou variable cible) pour ensuite inférer la connaissance extraite sur des entrées avec des sorties inconnues. Chaque exemple appelé aussi instance, est un couple d'entrée- sortie $(x_1, y_1), (x_2, y_2), (x_3, y_3) \dots (x_n, y_n)$

- $\mathcal{X}$ est l'ensemble des attributs (les variables explicatives)
- $\mathcal{Y}$ est l'ensemble des valeurs de sortie (la variable cible ou dépendante) peut être continue ou discret.

Lorsque l'algorithme détermine correctement la sortie y pour des entrées X qui ne faisaient pas partie de l'échantillon d'apprentissage, alors la fonction ou le modèle est dit optimal.
En apprentissage supervisé, on distingue entre deux types de taches :

1.4.1 La régression :

On parle de problème de régression lorsque la variable cible à prédire est continue.
Exemples de problèmes de régression :

- Prédiction du montant des ventes d'une entreprise compte tenu du contexte économique.
- Prédiction du prix de vente d'une maison en fonction de plusieurs critères.
- Prédiction de la consommation électrique dans une ville étant donné des conditions météorologiques.

1.4.2 La classification (ou catégorisation) :

On parle de problème de classification (ou catégorisation) lorsque la variable cible à prédire est discrète, chaque valeur correspond à une classe ou une catégorie .

Exemple de problème de catégorisation :

— Reconnaître un code postal à partir de chiffres manuscrits.

— Identifier si une transaction financière est frauduleuse.

— Prédire si un client d'une banque est élligible à un prêt bancaire ou non

— Identifier en quelle langue un texte est écrit.

— Identifier les objets présents sur une photographie.

— Identifier si un e-mail est spam ou non.

Il existe plusieurs algorithmes d'apprentissage supervisé voici les plus importants :

— Régression linéaire simple, polynomiale et multivariée.

— Régression logistique

— K plus proches voisins (K-Nearest Nieghbour).

— Arbres de décision (décision trees).

— Forêts aléatoires (random forest).

— Machines à vecteurs supports ou séparateurs à vaste marge (Support vectors machines : SVM)

— Les réseaux de neurones (Neural Network)

1.5 Apprentissage non supervisé :

Pour ce type d'apprentissage l'ensemble de données d'apprentissage ne contient pas de variable cible (comme on l'a vu en apprentissage supervisé). Il y a seulement un ensemble de données collectées en entrée. L'algorithme doit découvrir par lui-même la structure en fonction des données. On utilise cette technique pour partitionner les données en groupes d'éléments homogènes. La distance est souvent la plus utilisée comme mesure de similarité entre les groupes.[7] Voici quelques-uns des algorithmes d'apprentissage non supervisé les plus importants :

— Clustering :

— K-Means ;

— Analyse des clusters hiérarchique (HCA) ;

— Maximisation des attentes.

— Visualisation et réduction de la dimensionnalité

— Analyse en composante principales (ACP).

— kernel PCA.

— L'encastrement linéaire local (LLE).

— T-distribué Stochastique Neighbors Embedding (t-SNE).

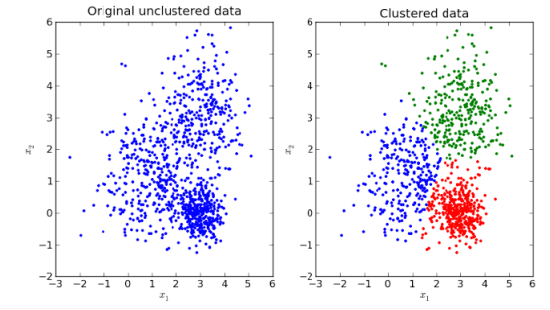


FIGURE 2 – Exemple d'apprentissage non supervisé (clustering)

1.6 Apprentissage semi supervisé :

Il s'agit d'un mixe entre l'apprentissage supervisé et non supervisé en utilisant des données étiquetées et non-étiquetées pour le même ensemble de données. L'avantage d'utiliser cette approche réside dans le fait que l'étiquetage de données peut être coûteux et prend souvent beaucoup de temps. En plus, il pourra entraîner un biais humain dans les données étiquetées. Dans ce cas, l'apprentissage semi-supervisé, qui ne nécessite que quelques étiquettes, est très pratique. Et le fait d'inclure un grand nombre de données non étiquetées au cours du processus d'entraînement a tendance à améliorer la performance du modèle final tout en réduisant le temps et les coûts consacrés à sa construction.[8]

1.7 Apprentissage par renforcement :

Dans ce cas, le programme informatique (l'agent) crée ses propres expériences en interagissant avec un environnement dynamique dans lequel il doit atteindre un objectif. L'agent a la liberté d'entreprendre des actions, selon lesquelles il reçoit un retour d'information analogue à des pénalité ou des recompenses. L'apprentissage par renforcement consiste à apprendre les actions à prendre à partir d'experiences, de façon à maximiser des récompenses au cours du temps

Les application de l'apprentissage par renforcement sont diverse et variées notamment :

les voitures autonomes, les drones , la robotique, les jeux video et tout type de jeux.
Le scénario général de l'apprentissage par renforcement est illustré par la figure 6.

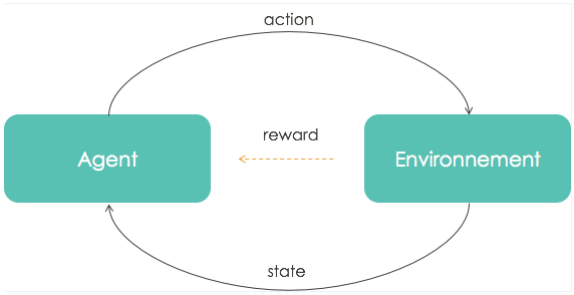


FIGURE 3 – Apprentissage par renforcement

1.8 Le machine learning dans l'entreprise :

Le machine learning en entreprise consiste à automatiser des process que les humain avaient l'habitude d'exécuter. Il contribue à améliorer le fonctionnement de l'entreprise et à lui faire gagner en compétitivité, le machine learning a la capacité d'apporter des solutions aux entreprises afin de développer des produits et des services performants et obtenir de nouvelles sources de revenue.[9] De plus il accélère la cadence de travail, diminue le risque d'erreurs et améliore la précision, aidant ainsi les employés comme les clients .[10] Par exemple d'importantes enseignes du e-commerce, comme Amazon, Netflix et Walmart, s'appuient sur les moteurs de recommandation pour personnaliser l'expérience d'achat et augmenter le panier moyen d'un client.

1.8.1 Le machine learning en marketing :

Le Machine learning a connu une utilisation massive dans le marketing notamment au niveau de la prédiction du comportement des consommateurs, l'analyse de la structure du marché, le développement des systèmes de recommandation, etc.[11]

Principaux avantages de l'apprentissage automatique en marketing :

— Améliore la qualité de l'analyse des données.
— Permet d'analyser plus de données en moins de temps.
— S'adapte aux changements et aux nouvelles données.
— Permet d'automatiser les processus marketing et d'éviter le travail de routine.
— Fait tout ce qui précède rapidement

1.8.2 Le machine learning en Finance :

De nos jours, de nombreuses sociétés de technologie financière et de services financiers de premier plan intègrent l'apprentissage automatique dans leurs opérations, ce qui se traduit par un processus mieux rationalisé, des risques réduits et des portefeuilles mieux optimisés. Les banques et les sociétés de services financiers ont recours à l'analyse pour différencier les interactions frauduleuses des transactions commerciales légitimes. En appliquant des outils d'analyse et d'apprentissage automatique, ils peuvent définir une activité normale en fonction de l'historique d'un client et le distinguer d'un comportement inhabituel indiquant une fraude.[12] L'automatisation des processus est l'une des applications les plus courantes de l'apprentissage automatique en finance. La technologie permet de remplacer le travail manuel, d'automatiser les tâches répétitives et d'augmenter la productivité.[13]

1.8.3 Le machine learning en commerce

Le Machine Learning a de nombreuses applications dans le secteur du commerce électronique qui vont bien au-delà de l'analytique :

— Analyse de paniers,
— Personnalisation de recommandations produites,
— Analyse de sentiment sur les réseaux sociaux,
— Evaluation de la satisfaction client,
— Ventes additionnelles et ventes croisées.

CHAPITRE 2

ÉLÉMENTS DE LA THÉORIE DE L'APPRENTISSAGE STATISTIQUE

2.1 Introduction

La théorie de l'apprentissage statistique est une discipline inventée par Vladimir Vapnik, c'est une évolution de l'inférence statistique traditionnelle vers l'analyse de données complexes, c'est une formulation mathématique des processus d'apprentissage utilisant différentes branches : les statistiques, l'optimisation et l'analyse fonctionnelle.[14]

La théorie de L'apprentissage statistique est un cadre théorique qui permet une analyse rigoureuse et formelle d'un phénomène avec une certaines garanties sur la performance des algorithmes utilisés. Ce cadre permet de prendre en compte toute amélioration éventuelle de l'information et des outils de modélisation.[15]

2.2 Données et modéle statistique

Nous disposons d'un ensemble de données D_n composé de n couples :

$$D_n = \{(x_i, y_i) \quad , \quad i = \widehat{1, n} \quad , n \in \mathbb{N}^*\}$$

que nous supposons être des réalisations indépendantes de variable aléatoire (X, Y) de la loi conjointe P inconnue.

Les entrées $x_i \in \mathcal{X}$, ($\mathcal{X}$ un sous ensemble de $\mathbb{R}^p$).

Chaque x_i=(x_{i1} ,x_{i2}, ... ,x_{ip}) est une réalisation de p variables explicatives :

X_j , $j = 1, ..., p$.

A chaque entrée x_i correspond une sorties $y_i \in \mathcal{Y}$, ($\mathcal{Y}$ un sous ensemble de $\mathbb{R}$).[16]

$$X_1 \quad X_2 \quad \dots \quad X_p$$

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \dots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Objectif et modèle statistique :

Notre objectif est de prédire les valeurs de y associées à chaque valeur possible de $x \in X$ qui ne sont pas présentent dans l'échantillon d'apprentissage.

- Si $\mathcal{Y}$ est un ensemble continue, on parle de régression.

- Si $\mathcal{Y}$ est un ensemble discret, on parle de classification(exp. discrimination binaire $Y = \{-1, 1\}$)

D_n est appelé échantillon d'apprentissage ou jeu de données.[17]

Y est relié aux X_j par la relation :

$$Y = f(X) + \epsilon$$

Et notre objectif revient à trouver la meilleure approximation de f qu'on appellera régle de prévision

2.3 Mesure de qualité

2.3.1 Fonction de côut :

On se donne une fonction de côut (aussi applée fonction de perte) :

$$l : \mathcal{Y} * \mathcal{Y} \to R$$

mesurable, telle que $l(y, y^{'})$ est d'autant plus petit que y et $y^{'}$ sont similaires, on suppose que :

$$\forall\, y\ ,\ y^{'} \in \mathcal{Y} \qquad , \quad l(y, y^{'}) \geq 0 \qquad et \qquad l(y, y) = 0$$

Si f est une régle de prévision, x une enteée, y la sortie qui lui est réellement associée, alors $l(y; f(x))$ mesure une perte encourue lorsque l'on associe à x la sortie $f(x)$.[18]

En pratique, le choix de la fonction de perte l dépend de la nature du problème (classement, régression) et de la famille des prédicteurs $\mathcal{F}$ auquel appartient la règle de prévision f.

Exemple :

$l(y, y^{'}) = |y - y^{'}|^{q}$ en régression (perte absolue si q=1, perte quadratique si q=2)

$l(y, y^{'}) = 1_{y \neq y'}$ en classification binaire.[19]

On va s'intéresser au comportement moyen de cette fonction de perte, il s'agit de la notion de ***risque.***

2.3.2 Le risque absolu ou erreur de généralisation :

Définition :Étant donnée une fonction de perte l , le risque espéré ou l'erreure de généralisation d'une régle de prévision f est défini par :

$$R_p(f) = E_p[l(f(X), Y)]$$

avec E_p :l'espérance par rapport à la distribution inconnue P,

On appelle généralisation la capacité d'un modéle à faire des prédictions correctes sur de nouvelles données, qui n'ont pas été utilisées pour le construire (l'apprentissage).[20]

L'objectif du problème de prévision est donc de trouver un modèle (ou une règle de prévision) $f^* \in \mathcal{F}$ qui minimise le risque espéré :

$$f^* \in \arg min_{f \in \mathcal{F}} R_P(f)$$

c.à.d

$$R_p(f^*) = R^* = \inf_{f \in \mathcal{F}} R_p(f)$$

$R^* = \inf_{f \in \mathcal{F}} R_p(f)$ est appelé risque de bayes et f^* est appelé prédicteur de bayes ou oracle.[21]

Comme la loi P est inconnue, il est donc necéssaire de construire des régles de prévision qui ne dépondent pas de P mais de D_n.

On peut reformuler le problème de prévision comme un probléme d'estimation : trouver un estimateur de f^* à partir de D_n.[22]

Une estimation $\widehat{f}$ de f^* sur un échantillon d'apprentissage D_n est obtenue par un algorithme d'apprentissage.[23]

2.3.3 Le risque empirique :

Le risque empirique est défini par :

$$\widehat{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n} l(y_i, f(x_i))$$

on l'appele aussi erreur d'apprentissage

La minimisation du risque empirique, qui est une extension de la procédure d'estimation d'un modèle, par exemple par les moindres carrés, a été développée par Vapnik (1999).

La minimisation du risque empirique est généralement un problème mal posé au sens de Hadamard, c'est-à-dire qu'il n'admet pas une solution unique. Il se peut par exemple qu'un nombre infini de solutions minimise le risque empirique à zéro.

De plus, la règle de prédiction construite par minimisation du risque empirique n'est pas statistiquement consistante. Rappelons qu'un estimateur θ_n (dépendant de n observations) d'un paramètre θ est consistant s'il converge en probabilité vers θ quand n tend vers l'infini :

$$\forall \epsilon > 0, \quad \lim_{n\to\infty} P(|\theta_n - \theta| \geq \epsilon) = 0$$

La loi des grands nombres nous garantit que le risque empirique converge vers le risque espéré :

$$\forall f \in \mathcal{F}\ , \quad \widehat{R}_n(f) \xrightarrow[n\to\infty]{} R_p(f)$$

Cela ne suffit cependant pas à garantir que le minimum du risque empirique $\min_{f\in\mathcal{F}} \widehat{R}_n(f)$ converge vers le minimum du risque espéré. En effet, si $\mathcal{F}$ est l'espace des fonctions mesurables, $\min_{f\in\mathcal{F}} \widehat{R}_n(f)$ vaut généralement 0, ce qui n'est pas le cas de $R_p(f)$. Il n'y a donc aucune garantie que la fonction f_n qui minimise $\widehat{R}_n(f)$ soit un bon estimateur du minimiseur f^* de $R_p(f)$.

Donc le "meilleur" modèle en un sens prédictif n'est pas nécessairement celui qui minimise le rique empirique c.à.d celui qui ajuste le mieux les données (faible biais), ni même le "vrai" modèle si la variance des estimations est importante, mais le modèle optimal est un modèle parcimonieux qui réalise un meilleur compromis biais/ variance.

Décomposition approximation/estimation(ou biais/variance) :

En statistique et en apprentissage automatique, le dilemme (ou compromis) biais-variance est le problème de minimiser simultanément deux sources d'erreurs qui empêchent les algorithmes d'apprentissage supervisé de généraliser au-delà de leur échantillon d'apprentissage :
Le biais : est l'erreur provenant d'hypothèses erronées dans l'algorithme d'apprentissage. Un biais élevé peut être lié à un algorithme qui manque de detecter les relations pertinentes entre les données en entrée et les sorties prévues (sous-apprentissage).
La variance : est l'erreur due à la sensibilité aux petites fluctuations de l'échantillon d'apprentissage, une variance élevée peut entraîner un surapprentissage, c'est-à-dire modélisé le bruit aléatoire des données d'apprentissage plutôt que les sorties prévues. [24]

2.3.4 Le sur-apprentissage :

On dit d'un modèle qui, plutôt que de capturer la nature sous jacente des données, modélise aussi le bruit et ne sera pas en mesure de généraliser qu'il sur-apprend. En anglais, on parle d'overfitting. Un modèle qui sur-apprend est généralment un modèle trop complexe, qui "colle" trop aux données et capture donc aussi leur bruit.
À l'inverse, il est aussi possible de construire un modèle trop simple, dont les performances ne soient bonnes ni sur les données utilisées pour le construire, ni en généralisation, on parle dans ce cas de sous-apprentissage (undefiting).

Le principal enjeu est de construire une estimation sans biais de ce risque en notant que le risque empirique comme on vient de le voir n'est pas un bon estimateur du risque espére, il est biaisé par optimisme car toute erreur ou risque estimé sur d'autres données et qui n'ont pas servi à estimer le modèle ou apprendre l'algorithme, conduit, en moyenne, à des valeurs plus élevées. Sélectionner un modèle en minimisant le risque empirique conduit au **sur-apprentissage** ou **sur-ajustement** voir figure 4.

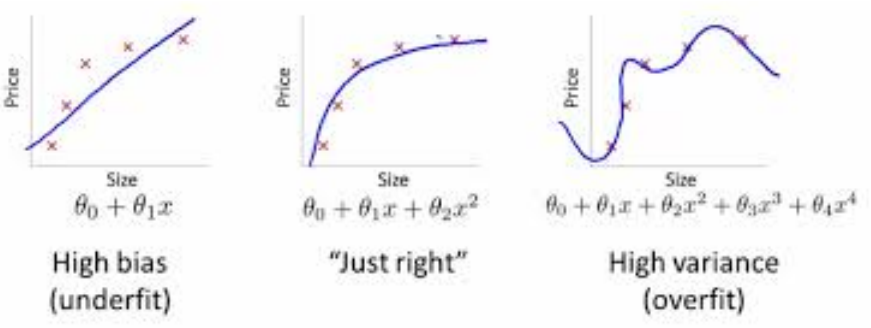


FIGURE 4 – Le sur-apprentissage et le sous-apprentissage

2.3.5 Selection de modèle

Le choix d'un modèle adéquat est crucial et relève des méthodes de sélection de modèles. Trois stratégies pour construire des estimations sans biais du risque espéré (erreur de prévision ou encore capacité de géneralisation) :[23]

1. Une pénalisation de l'erreur d'ajustement ou risque empirique
2. Un partage de l'échantillon : apprentissage, test(validation) afin de distinguer l'estimation du modèle et celle du risque
3. Validation croisée, bootstrap

Le choix dépend de plusieurs facteurs : l'objectif recherché, la taille de l'échantillon initial, la complexité du modéle envisagé, la variance de l'erreur, la complexité des algorithmes.

Estimation du risque empirique

1. **Estimation par pénalisation** : La sélection de modèle par pénalisation consiste en l'ajout d'une pénalisation à la fonction objectif :

$$\widehat{f} = [\arg\min_{f \in F} \widehat{R}_n(\widehat{f}_F, D_n) + pen(\widehat{f})]$$

La pénalité permet de pénaliser les modèle de "grande" taille, afin d'éviter le surajustement. Généralement, plus un modéle est complexe, plus il est flixible et peut s'ajuster aux données observées et donc plus le biais est réduit, en revanche, la partie variance augmente avec cette complexité.

L'enjeu est donc de rechercher un meilleur compromis entre biais et variance : accepter de biaiser l'estimation pour réduire plus favorablement la variance.[25]

2. **Estimation par échantillons indépendants** :

   La façon la plus simple d'estimer sans biais l'erreur de prévision ou un risque consiste à utiliser un échantillon indépendant n'ayant pas participé à l'estimation du modèle. Ceci nécessite donc partager aléatoirement l'echantillon en trois parties respectivement appelées apprentissage, validation et test :

   $D_n = D_{n_1}^{Appr} \bigcup D_{n_2}^{Valid} \bigcup D_{n_3}^{Test}$, avec $n_1 + n_2 + n_3 = n$

   $\widehat{R}_n(\widehat{f}_F(D_{n_1}^{Appr}), D_{n_1}^{Appr})$ est minimisé pour déterminer l'estimateur ou l'algorithme de prévision $\widehat{R}_n(\widehat{f}_F(D_{n_1}^{Appr}))$: pour un modèle fixé , par exemple un modéle de régression polynomiale de degré fixé.

   $\widehat{R}_n(\widehat{f}_F(D_{n_1}^{Appr}), D_{n_2}^{Valid})$ sert à la comparaison des modèle au sein d'une même famille afin de sélectionner celui qui minimise cette erreur , par exemple une famille de modèles polynomiaux de degrés variés. [26]

   $\widehat{R}_n(\widehat{f}_F(D_{n_3}^{Test}), D_{n_3}^{Test})$ est utilisée pour comparer entre eux les meilleurs modèle de chacune des méthode considérée, par exemple le meilleur modèle polynomial au meilleur réseau de neurones.

   Cette solution n'est faisable que si la taille l'échantillon initiale est importante sinon la qualité d'ajustement est dégradée car n_1 serait trop faible et la variance de l'estimation de l'erreur peut être importante ($n_2, n_3 petits$). [27]

3. **Estimation par validation croisée** : Il existe plusieurs versions de validation croisée, elles différent essentiellement par le choix de procédure permettant de séparer itérativement l'échantillon initial en parties apprentissage et validation. L'estimation du risque ou de l'erreur de prévision est itérée puis toutes les estimations moyennées avant d'en calculer la moyenne pour réduire la variance et améliorer la précision lorsque la taille de l'échantillon initial est trop réduite pour en extraire des échantillons de validation et test indépendants de taille suffisante.

La V-fold cross validation ou validation croisée en V segments consists à partager aléatoirement l'échantillon en V segments puis, itérativement à faire jouer à chacun de ces segments le rôle d'échantillon de validation tandis que les $V-1$ autre constituent l'échantillon d'apprentissage servant à estimer le modèle. [28]

Soit $T : \{1, ..., n\} \leftarrow \{1, ..., V\}$ La fonction d'indexation qui, pour chaque observation, donne l'attribution uniformément aléatoire de sa classe.

L'estimation par validation croisée de prévision est :

$$\widehat{R}_{CV} = \frac{1}{n} \sum_{i=1}^{n} l(y_i; \widehat{f}^{-T(i)}(x_i))$$

Où $\widehat{f}^{-T(i)}$ désigne l'estimation de f sans prendre en compte la k-iéme partie de l'échantillon.[29] Le choix couramment utilisé de V est entre 5 et 15, (V=10 par défaut dans les logiciels).

Minimiser l'erreur estimée par validation croisé est une approche largement utilisée pour optimiser le choix d'un modèle au sein d'une famille paramétrée (de paramétres Θ), $\widehat{f}$ est défini par :

$$\widehat{\Theta} = argmin_{\Theta} \widehat{R}_{CV}(\Theta)$$

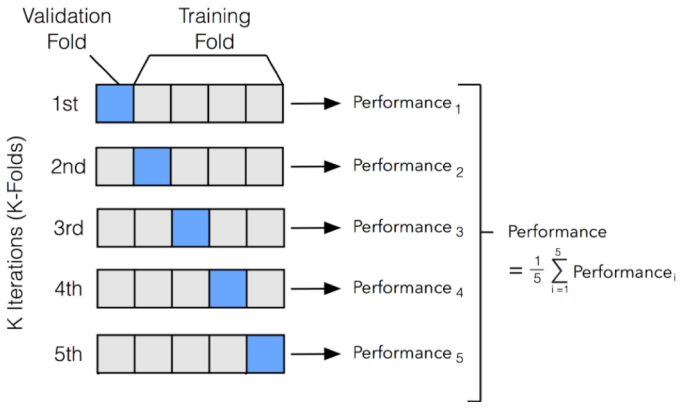


FIGURE 5 – validation croisée en v segments

2.4 Critéres de performance :

Il existe de nombreuses façon d'évaluer la performance prédictive d'un modèle d'apprentissage supervisé.

En classification on utilise la matrice de confusion qui résume le nombre d'erreurs de classification et ça permet d'évaluer la qualité d'un modèle prédictif.

Dans le cas d'une régression, le nombre d'erreurs n'est pas un critére approprié pour évaluer la performance d'un modéle prédictif, on utilise en gńéral ces quatre critères :[30]

1. Erreur quadratique moyenne(MSE) :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2$$

2. Racine de l'erreur quadratique moyenne (RMSE) :

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2}$$

3. Racine du log de l'erreur quadratique moyenne(RMSLE) :

   Dans le cas où les valeurs cible couvrent plusieurs ordre de grandeur, on préfére parfois passer au log :

$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(log(y_i + 1) - log(f(x_i) + 1))^2}$$

4. Coefficient de détermination R^2 :

$$R^2 = 1 - RSE = 1 - \frac{\sum_{i=1}^{n}(y_i - f(x_i))^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}$$

Conclusion : Dans ce second chapitre, nous nous sommes intéressés au domaine de l'apprentissage statistique. En effet, nous avons vue les données et modèle statistique, et puis les mesures de qualité (Fonction de coût, le risque espéré, le risque empirique et le sur apprentissage), pour finir, nous avons présenté les critères de performance.

CHAPITRE 3

LES ALGORITHMES DE L'APPRENTISSAGE SUPERVISÉ

3.1 Introduction :

On parle d'apprentissage supervisé lorsque les données sont étiquetées, nous disposons d'un échantillon d'apprentissage D_n avec :

$$D_n = \{(x_i, y_i) \quad , \quad i = \widehat{1, n} \quad , n \in \mathbb{N}^*\}$$

L'apprentissage vise alors à trouver un modèle liant la valeur de la variable à expliquer, aux variables explicatives. On peut ensuite utiliser ce modèle à des fins préditives. Nous présentons ci-dessous quelques algorithmes de l'apprentissage supervisé dans le cas de la régression.

3.2 La Régression multivarié :

Le modèle de régression multivarié est une généralisation du modèle de régression simple, dans ce cas on a plusieurs variables indépendantes. la variable à expliquer Y est relié aux X_j par la relation :

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p + \epsilon_i, \quad i = 1, 2, \ldots, n$$

$$Y = f(X) + \epsilon$$

où :

— Y est la variable à expliquer (variable cible) ;

— $X_1, \ldots, X_p$ sont les variables explicatives

— ϵ est le terme d'erreur aléatoire du modèle ;

— $\beta_0, \beta_1, \ldots, \beta_p$ sont les paramètres à estimer.

La fonction de coût est de la forme :

$$J(\beta) = \frac{1}{2n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 \quad = \frac{1}{2n}\sum_{i=1}^{n}(\sum_{j=1}^{p}\beta_j x_{ij} + \beta_0 - y_i)^2$$

Notre problème revient donc à trouver le meilleur vecteur $\beta = (\beta_0, \beta_1, \ldots, \beta_p)^t$ tel que $f(X)$ soit "proche" de Y pour les données d'apprentissage, ça revient à trouver le minimum de la fonction $J(\beta)$:

$$\widehat{\beta} = arg \min_{\beta \in \mathbb{R}^{p+1}} \frac{1}{2n}(\sum_{i=1}^{n}(\sum_{j=1}^{p}\beta_j x_{ij} + \beta_0 - y_i)^2)$$

Et pour cela on utilise une méthode analytique bien connue qui est les moindre carrés ordinaires (MCO) (somme des carrées des résidus ou Residual Sum of Squares) ou une méthode numérique comme la méthode de la déscente du gradient

La descente du gradient

L'algorithme de la descente du gradient ou Gradient Descent est un algorithme d'optimisation qui permet de trouver le minimum de n'importe quelle fonction convexe en convergeant progressivement vers celui-ci.

lorsqu'un système est en phase d'apprentissage, il commet des erreurs, le taux d'erreurs diminue au fur et à mesure de l'apprentissage, mais il se peut que à un moment donné l'erreur augemente pour a nouveau rediminuer et atteindre un niveau plus bas que le précédent, c'est le niveau optimal de l'apprentissage.

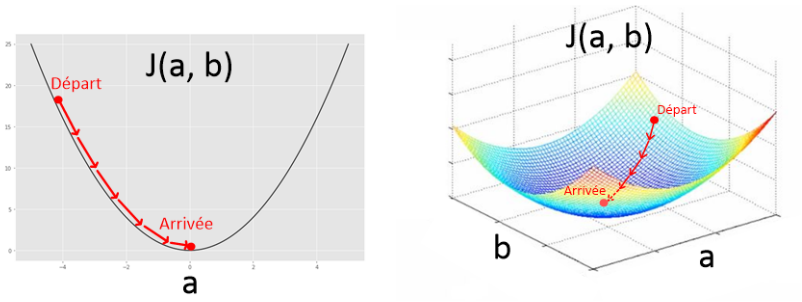


FIGURE 6 – La déscente du gradient

Exemple : illustrons cet alogorithme dans le cas d'une seule variable explicative :

$$y = \beta_0 + \beta_1 x + \epsilon$$

la méthode de La fonction côut dans ce cas est :

$$J(\beta_0, \beta_1) = \frac{1}{2N} \sum_{i=1}^{N} (f(x_i) - y_i)^2 = \frac{1}{2N} \sum_{i=1}^{N} (\beta_0 + \beta_1 x_1 - y_1)^2$$

L'algorithme se décrit comme suit :

— Début de l'algorithme : Initialiser aléatoirement les valeurs de :β_0 et β_1

— répéter jusqu'à convergence au minimum global de la fonction de coût

pour $j \in \mathbb{N} \wedge \forall j \in \{0, 1\}$

$$\beta_j := \beta_j - \alpha \frac{\partial J(\beta_0, \beta_1)}{\partial \beta_j}$$

— retourner et β_1

— Fin algorithme

la vitesse de convergence est déterminer par le facteur α appelé "Learning rate" plus α est grand, plus le modificateur des paramètres entre deux itération est grand (donc la probabilité de "rater" le minimum ou de diverger est grande). A l'inverse, plus α est petit, plus nous avons de probabilité de converger mais le processus est long.

3.3 Les K plus proches voisins(k Nearest Neighbors – KNN)

Il est basé sur une approche de similarité des caractéristiques. Pour prédire la sortie d'une nouvelle donnée d'entrée il va chercher ses K voisins les plus proches (en utilisant la distance euclidienne, ou autres) et choisira la classe ou l'étiquettes des voisins majoritaires. C'est un algorithme d'apprentissage non paramétrique et paresseux (lazy learning) : Les KNN est utilisé pour les problèmes de classification et de régression.[31]

L'algorithme KNN :

1. Charger les données
2. Initialiser k au nombre de plus proches voisins choisi
3. Pour une observation x dont on veut calculer la sortie y faire :
   (a) Calculer la distance entre cette observation et toutes les observations du jeu de données.

(b) Conserver les k observations du jeu de données qui sont les plus "proches" de l'observation dont on veut prédire la sortie .

(c) Prendre les valeurs des observations retenues : Si on effectue une régression, l'algorithme calcule la moyenne (ou la médiane) des valeurs des observations retenues, Si on effectue une classification, l'algorithme assigne le label de la classe majoritaire .

4. Retourner la valeur calculée dans l'étape 3 comme étant la valeur qui a été prédite par l'algorithme pour l'observation en entrée.

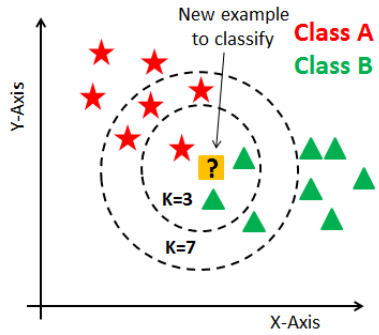


FIGURE 7 – Fonctionnement de l'algorithme des KNN

Exemple de distances :

Il existe plusieurs fonctions de calcul de distance, notamment, la distance euclidienne, la distance de Manhattan, la distance de Minkowski, celle de Jaccard et la distance de Hamming. . . etc. On choisit la distance en fonction des types de données qu'on manipule.

3.4 Les arbres de décision :

Les arbres de décision sont utilisés pour les problèmes de classification et la régression.

C'est un outil d'aide à la décision ou d'exploration de données qui permet de représenter un ensemble de données sous la forme graphique d'un arbre.

L'idée centrale est : diviser récursivement et le plus efficacement possible l'échantillon d'apprentissage au moyen de tests définis à l'aide des attributs(les variables explicatives) jusqu'à obtenir des sous-échantillons contenant des exemples appartenant (presque) tous à une même classe (même sortie en régression) .

Ces méthodes dites de partitionnement récursif ou de segmentation datent des années 60. Elles ont été formalisées dans un cadre générique de sélection de modèle par Breiman et

col. (1984) sous l'acronyme de CART : Classification and Regression Tree. Parallèlement Quinlan (1993) a proposé l'algorithme C4.5 dans la communauté informatique.[32]

3.4.1 Structure d'un arbre de décision

Un arbre de décision (ad) est donc une structure hiérarchique composée de :

- Nœud racine (Root node) : Il s'agit du nœud situé au sommet d'un arbre décisionnel représentant l'ensemble des données.
- Nœuds internes : Il s'agit des points de l'arbre où l'espace des attribues est divisé.
- Nœuds terminaux (ou feuilles ou Terminal nodes) : Il s'agit de nœuds qui ne sont plus divisés et qui constituent une fin en soi
- Branches : Ce sont des lignes reliant différents nœuds (terminaux ou internes).

Tels que :

- les nœuds représentent des sous-espaces (régions) de $\mathcal{X}$.
- La racine contient tout $\mathcal{X}$ tandis que les feuilles des régions unitaires.
- Entre la racine et les feuilles, les nœuds intermédiaires représentent des régions emboîtées :$\mathcal{X} = \mathcal{X}^1 \oplus \ldots \oplus \mathcal{X}^m \oplus \ldots \oplus \mathcal{X}^{p'}$ avec $p' \leq p$ et chaque X^m peut être à nouveau décomposé en sous-régions.
- A chaque nœud m est associé une région $\mathcal{X}^m \subset \mathcal{X}$ et une fonction de décision notée f_m qui à un élément $x \in \mathcal{X}^m$ associe un sous-espace $\mathcal{X}^{m'} \subset \mathcal{X}^m$.
- A chaque feuille de l'arbre est associée un élément de $\mathcal{Y}$:
  - Pour un problème de discrimination (classification) il s'agit donc d'une classe ($\mathcal{Y}$ discret).
  - Pour un problème de régression il s'agit donc d'un réel ($\mathcal{Y}$ continu).
- Chaque feuille correspond à une région de $\mathcal{X}$ et tout les x appartenant à une même feuille ont le même élément de sortie y de $\mathcal{Y}$ associé à la feuille.
- la fonction de discrimination f_m du nœud m est une fonction simple. Mais, l'ensemble des fonctions f_m de chaque nœud de l'arbre tout entier aboutit à une fonction de décision complexe.
- Les arbres décisionnels sont considérées comme des méthodes non paramétriques
- Aucune hypothèse sur la distribution de probabilités des classes.
- La structure de l'arbre n'est pas donnée à l'avance : on ajoute nœuds, arcs (branche) et feuilles, en fonction des données à l'étude.

— Les méthodes de cette famille se distinguent selon :
  — Le type de fonction f_m choisi pour discriminer un ensemble de points.
  — Le type de critère permettant d'évaluer la qualité d'une fonction de discrimination.

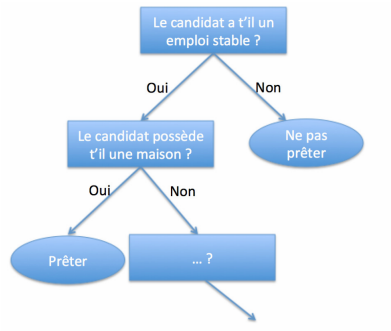


FIGURE 8 – Exemple d'arbre de décision pour accorder ou non un prêt bancaire

Construction d'un arbre de régression :

On considère $\mathcal{Y} = \mathbb{R}$, on parle alors d'arbre de régression. Par contre $\mathcal{X}$ peut être hétérogène c.à.d mélange de variables continues et discrètes(catégorielles). Nous traiterons essentiellement des méthodes univariées c.à.d à chaque noeud m on utilise une seule variable explicative X_j pour définir f_m Si X_j est discrète avec q_j catégories $X_{j_1}, \ldots, X_{q_j}$, alors :

$\forall x \in X\,; f_m(x) \in \{X_{j_1}, \ldots, X_{jq_j}\}$

Il s'agit dans ce cas d'une séparation ou division en q_j régions.
Généralement on opte pour des une division en deux régions(arbre binaire) : $\forall x \in X\,; f_m(x) \in \{X_{j_k}, \overline{X}_{j_k}\}$

Si X_j est continue alors :

$\forall x \in X; f_m(x) \in \{X_j^1, X_j^p\}$

avec $X_j^1 = \{X/X_j < \delta_j\}$ et $X_j^p = \{X/X_j \leq \delta_j\}$ où $\delta \in \mathbb{R}$ est une valeur permettant de faire la meilleure séparation.

3.4.2 Construction des règles :

La première division (on parle aussi de split) est obtenue en choisissant la variable explicative qui permet la meilleure séparation des individus qui est donnée par l'attribut qui renvoie la plus basse "impureté" des noeuds qu'il génère et pour mesurer la pureté d'un nœud m on utilise le critère suivant :

$$err(m) = \frac{1}{N^m} \sum_{y_i \in m} (y_i - g^m)^2$$

où N^mest le cardinal du noeud m
et g^m une tendance central de y relative au nœud m, on utilise la moyenne (ou la médiane si les données sont très bruitées) :

$$g^m = \frac{\sum_{y_i \in \mathcal{X}^m} y_i}{N^m}$$

Dans ce cas $err(m)$ est une variance locale du nœud m, on dit aussi impureté. Le critère qui arrête la progression de l'ad est $err(m) \leq \theta$, θ est donc un seuil en-dessous duquel on estime que la variance de la région relative au nœud m est suffisamment basse.

Cette division donne des sous-populations correspondant au premier nœud de l'arbre. Le processus de split est ensuite répété plusieurs fois pour chaque sous population (noeuds précédemment calculés) jusqu'à ce que le processus de séparation s'arrête.

3.4.3 Critère d'arrêt et d'élagage :

- Règles évidentes :
  - Tout les exemples du nœud sont de même classe
  - Tout exemples du nœud ont mêmes valeurs de variables
  - L'hétérogénéité des nœuds ne diminue plus.
  - Le nombre d'exemples dans le nœud < seuil minimal
- Contrôle des performances de généralisation (sur base de validation indépendante)
- Elagage a posteriori : supprimer des branches peu représentatives et qui causent de la nuisances à la généralisation, pour chaque sous-arbre enlevé, on le remplace par une feuille étiquetée avec une tendance centrale (en général la moyenne), on continue tant que cela fait diminuer l'erreur de généralisation qu'on estimer en utilisant un échantillon de test indépendant de l'ensemble d'apprentissage ou une validation croisée.

3.4.4 les avantages des arbres de décision :

Ils sont simples à comprendre et à interpréter. On peut visualiser les arbres. Aussi, on peut expliquer les résultats obtenus facilement. Ils peuvent travailler sur des données avec peu de préparation. Par exemple, ils n'ont pas besoin de la normalisation des données. Ils acceptent les données numériques et nominales. Les autres algorithmes d'apprentissage sont spécialisés dans un seul type de données. Ils donnent de bonne performance même si leurs hypothèses sont un peu violées par le modèle réel à partir duquel les données ont été générées.

3.5 Les réseaux de neurones artificiels :

les réseaux de neurones sont des algorithmes complexes qui tente de mimer la manière dont fonctionne le cerveau humain.
L'histoire des réseaux de neurones artificiels remonte aux années 1950 et aux efforts de psychologues comme Frank Rosenblatt pour comprendre le cerveau humain. Initialement, ils ont était conçus dans le but de modéliser mathématiquement le traitement de l'information par les réseaux de neurones biologiques. De nos jours, c'est leur efficacité a modéliser des relations complexes et non linéaire qui fait leur succès et non plus leur réalisme biologique.
Dans le cadre de traitement de données, ils sont utile lorsque ces systèmes sont difficiles à modéliser à l'aide des méthodes statistique classiques, et dans les cas où il existe une relation non linéaire entre des variables explicatives et une variable expliquée.

Le premier réseau de neurones artificiels est le perceptron(Rosenblatt, 1957). Loin d'être profond, il comporte une seule couche et a une capacité de modélisation limitée.

En 2006, Geoffrey Hinton et Al. ont publié un article montrant comment former un réseau neuronal profond capable de reconnaître des chiffres manuscrits avec une précision de pointe (98%)

Ils ont baptisé cette technique " Deep Learning " (apprentissage profond).
La formation d'un réseau neuronal profond était largement considérée comme impossible à l'époque et la plupart des chercheurs avaient abandonné l'idée depuis les année 1990. Cet article a ravivé l'intéret de la communauté scientifique et, trés vite ,de nombreux nouveaux articles ont démontré que l'apprentissage profond était non seulement possible, mais qu'il permettait d'obtenir des résultats époustouflants qu'aucune autre technique

d'apprentissage automatique ne pouvait espérer égaler (grace à une puissance de calcul énorme et à de grandes quantités de donnée).

Le perceptron unicouche

Le perceptron ou neurone artificiel est une fonction mathémathique $f : \mathbb{R}^p \to \mathbb{R}$, définie par :

- $\Phi : \mathbb{R} \to \mathbb{R}$
- $w \in \mathbb{R}^{p+1}, w = (w_0, w_1, \ldots, w_p)$
- $\forall x \in \mathbb{R}^p, f(x) = \phi(w_0 + \Sigma_{j=1}^{p} w_j x_j)$

qu'on peut présenter graphiquement par la figure 9

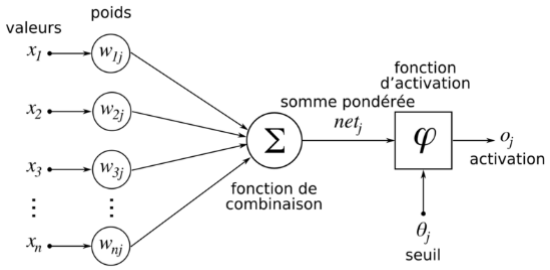


FIGURE 9 – Perceptron unicouche (à couche unique)

il est formée d'une couche d'unités (ou neurones) qui permettent de lire les données : chaque unité correspond à une des variables d'entrée, on peut rajouter une unité de biais qui transmet toujours 1 quelles que soit les données. Ces unités sont pondérées par des poids de connexion. Pour p variables $x_1, x_2, \ldots, x_p$ la sortie reçoit donc $w_0 + \Sigma_{j=1}^{p} w_j x_j$. L'unité de sortie applique alors une fonction d'activation ϕ à cette sortie.

Différents types de fonctions d'activation sont utilisables, en fonction du problème à traiter, du filtrage que l'on veut effectuer sur les données (voir figure 10)

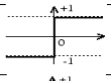
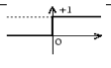
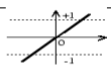
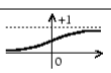
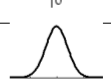
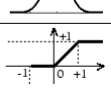
Fonction d'activation	*Modèle mathématique*	*Graphe*
Fonction signe	$\varphi(x) = \begin{cases} 1 \text{ si } x \geq 0 \\ -1 \text{ si } x \geq 0 \end{cases}$	
Fonction seuil	$\varphi(x) = \begin{cases} 1 \text{ si } x \geq s \\ 0 \text{ si } x \geq s \end{cases}$	
Fonction linéaire	$\varphi(x) = x$	
Fonction sigmoïde	$\varphi(x) = \frac{1}{1+e^{-x}}$	
Fonction gaussienne	$\varphi(x) = \frac{1}{\sqrt{2\pi\varphi}} e^{\frac{(x-c)^2}{2\varphi^2}}$	
Fonction saturation	$\varphi(x) = \begin{cases} 1 & si\ x \geq +1 \\ x \text{ si } -1 \leq x \leq +1 \\ -1 & si\ x \leq -1 \end{cases}$	

FIGURE 10 – Tableau de fonctions d'activation

Le perceptron multi-couches :

On appelle perceptron multi-couche, ou multi-layer perceptron (MLP) en anglais, un réseau de neurones construit en insérant des couches intermédiaires entre la couche d'entrée et celle de sortie d'un perceptron. On parlera parfois de couches cachées par référence à l'anglais hidden layers. Chaque neurone d'une couche intermédiaire ou de la couche de sortie reçoit en entrée le produit scalaire entre les sorties des neurones de la couche précédente et les poids de connection, lui meme appliquera une fonction d'activation. On utilise la mème fonction d'activation sur les neurone d'une meme couche. Il n'y a pas de retour d'une couche vers une couche qui la précède ; on parle ainsi d'un réseau de neurones a propagation avant, ou feed-forward en anglais. En utilisant des fonctions d'activation non linéaires, telles que la fonction logistique ou la fonction tangente hyperbolique, on crée ansi un modèle paramétrique hautement non linéaire.[32]

Théorème(Approximation universelle) : Soit $\phi : \mathbb{R} \to \mathbb{R}$ une fonction non constante, bornée, continue et croissante et K un sous-ensemble compact de $\mathbb{R}^p$. Etant donné $\epsilon > 0$ et une fonction f continue sur K, il existe $m \in \mathbb{N}$, m scalaires $\{d_i\}_{i=\widehat{1,m}}$, m scalaires $\{b_i\}_{i=\widehat{1,m}}$ et m vecteurs $\{w_i\}_{i=\widehat{1,m}}$ de $\mathbb{R}^p$ tels que :

$$\forall x \in K, |f(x) - \Sigma_{i=1}^{m} d_i \phi(< w_i, x > + b_i)| < \epsilon$$

avec $< w_i, x >= \Sigma_{j=1}^{p} w_j x_j$

En d'autres termes, toute fonction continue sur un sous-ensemble compact de $\mathbb{R}^p$ peut être approchée avec un degré de precision arbitraire par un perceptron multi-couche à une couche intermédiaire contenant un nombre fini de neurones.

Cependant, ce théorème, dù à George Cybeuko (1989), et affiné par Kurt Hornik (1991), ne nous donne ni le nombre de neurones qui doivent composer cette couche intermédiaire, ni les poids de connexion à utiliser. Les réseaux de neurones à une seule couche cachée sont généralement peu efficaces, et o aura souvent de meilleurs résultats en pratique avec plus de couches.

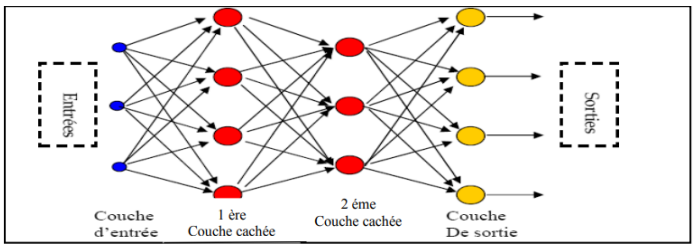


FIGURE 11 – perceptron multicouche

CHAPITRE 4

IMPLÉMENTATION ET COMPARAISON DES ALGORITHME DE MACHINE LEARNING

4.1 Les étapes pour développer un modèle de Machine learning supervisé :

1. Sélectionner le domaine, identifier le problème (classification, régression)
2. Comprendre et identifier les données : s'il s'agit d'une analyse prédictive, répondre aux questions : qu'elle est la variable Y (output) à prédire, quelle sont les variables explicatives X_i (input)
3. Collecter et préparer les données : une fois qu'on a identifié et localisé les données on passe à la collecte et le nettoyage :
   - Recueillir les données.
   - Normaliser le format des données .
   - Remplacer les données incorrectes.
   - Améliorer et augmenter les données.
   - Supprimer les informations superflues et les doublons et les non pertinentes pour l'entraînement.
   - Normaliser et standardiser les données pour qu'elle rentres dans les intervalles attendus (Features scaling).
4. Séparer les données en trois ensembles, un pour l'entraînement l'autre pour la validation et le dernier pour le test.
5. Déterminer les modèles de M.L (les algorithmes) en fonction du problème

6. Choisir le bon algorithme et l'entraîner.
7. Utiliser plusieurs algorithmes (apprentissage ensembliste).
8. Évaluer le modèle en utilisant un échantillon de validation ou une validation croisée.
9. Expérimenter le modèle et ajuster ses hyperparamètres .
10. Exploiter le modèle à des fins prédctives et décisionnelles

4.2 Les logiciels les plus utilisés pour le machine learning

Il existe de nombreux outils permettant d'effectuer de l'apprentissage statistique dans la communauté des datascientists les logiciels Open Source sont particulièrement prisés. Ils profitent en effet d'une actualisation permanente des algorithmes et permettent ainsi de bénéficier des dernières recherches à moindre coût. Parmi ces logiciels, deux se distinguent par leurs nombres d'utilisateurs : Python et R.

Le logiciel R

 R est un langage "Script" , largement utilisé par la communauté des statisticiens. R permet de produire des visualisations agréables. Logiciel d'origine statistique, il incorpore de nombreux algorithmes de Machine Learning. R bénéficie d'une grande communauté de contributeurs de packages qui permet au logiciel de s'adapter à la structure des nouvelles données et des nouvelles infrastructures informatiques.

Le logiciel Python

 Python est un langage "Orienté Objet", largement utilisé par la communauté des informaticiens et des mathématiciens. Le package scikit-learn de machine learning propose de nombreux algorithmes d'apprentissage statistique. Ce langage modulaire est adapté au développement d'outils. Il facilite également les suivis et les échanges grâce à l'interface Ipython Notebook, qui permet de partager et documenter facilement des travaux sans aucun logiciel (publication de pages web). Cependant, le logiciel n'égale pas encore la richesse des packages de R en statistique même si l'écart s'est considérablement réduit.

Dans ce mémoire nous avons opté pour le logiciel R pour son avantage dans la visualisation de données avec son package ggplot2 et la disponibilités de la documentation

pour ses différents packages open-source à profusion et sa large utilisation dans le milieu académique.

4.3 Présentation du jeu de données

On utilise un jeu de données (échantillon) appelé CPS1985(May 1985 Current Population Survey), disponible dans le package (AER) de R, il comporte des données recueillies aux états unis d'Amérique lors d'un recensement effectué en 1985 .

L'échantillon contient 534 observations sur 11 variables dont 4 de type quantitative (numérique) et 7 de type qualitative (catégorielle) .
La variable cible :
Y = salaire : Salaire (en dollars de l'heure).
Les variables explicatives :

- X_1 = éducation : Nombre d'années d'études.
- X_2 = expérience : Nombre d'années d'expérience professionnelle potentielle.
- X_3 = âge : l'âge en années.
- X_4 = appartenance ethnique : qualitative avec les catégories ; "blanc", "hispanique" et "autre".
- X_5 = région : qualitative ; L'individu vit-il dans le "Sud" ou "non" ?
- X_6 = le genre : qualitative ; "homme" ou "femme"
- X_7 = occupation : qualitative avec 6 catégories ; "ouvrier" (ouvrier à la chaîne), "technique" (ouvrier technique ou professionnel), "services" (ouvrier de service), "office" (employé de bureau), "vendeur", "management" (gestion et administration).
- X_8 = secteur : qualitative avec les 3 catégories ; "manufacture"(manufacturier ou minier), "construction", "autre".
- X_9 = syndicat : qualitative ; l'individu est-il sur un emploi "syndiqué" ou "non".
- X_{10} = marié : qualitative ; l'individu est-il "marié" ou "non" .

4.3.1 Exploration numérique des données

Il est toujours bon d'avoir une idée générale sur le comportement des variables du jeu de données, comme on peut le voir sur la figure 12 ci-dessous avec la fonction str(CPS1985)

```
> str(CPS1985)
'data.frame':   534 obs. of  11 variables:
 $ wage      : num  5.1 4.95 6.67 4 7.5 ...
 $ education : num  8 9 12 12 12 13 10 12 16 12 ...
 $ experience: num  21 42 1 4 17 9 27 9 11 9 ...
 $ age       : num  35 57 19 22 35 28 43 27 33 27 ...
 $ ethnicity : Factor w/ 3 levels "cauc","hispanic",..: 2 1 1 1 1 1 1 1 1 1 ...
 $ region    : Factor w/ 2 levels "south","other": 2 2 2 2 2 2 1 2 2 2 ...
 $ gender    : Factor w/ 2 levels "male","female": 2 2 1 1 1 1 1 1 1 1 ...
 $ occupation: Factor w/ 6 levels "worker","technical",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ sector    : Factor w/ 3 levels "manufacturing",..: 1 1 1 3 3 3 3 3 1 3 ...
 $ union     : Factor w/ 2 levels "no","yes": 1 1 1 1 1 2 1 1 1 1 ...
 $ married   : Factor w/ 2 levels "no","yes": 2 2 1 1 2 1 1 1 2 1 ...
```

FIGURE 12 – strucure des données

Les informations simples sur les variables sont obtenues par des représentations graphiques et/ou des mesures numériques.
Après avoir déterminé le type de chaque variable (qualitative, quantitative, discrète continue...), on peut calculer :

- Pour les variables quantitatives :
  - la moyenne et/ou médiane pour évaluer la position
  - la variance, l'étendue, l'écart inter-quartiles etc... pour évaluer la dispersion
  - le coefficient d'asymétrie pour évaluer la forme de la distribution
- Pour les variables qualitatives :
  - les modalités et leurs fréquences ou effectifs
  - le mode (la ou les modalités les plus représentées)

Certaine de ces mesures numériques sont resumés dans la figure 13 ci-dessous.

```
> summary(CPS1985)
      wage            education       experience         age          ethnicity     region       gender
 Min.   : 1.000   Min.   : 2.00   Min.   : 0.00   Min.   :18.00   cauc    :440   south:156   male  :289
 1st Qu.: 5.250   1st Qu.:12.00   1st Qu.: 8.00   1st Qu.:28.00   hispanic: 27   other:378   female:245
 Median : 7.780   Median :12.00   Median :15.00   Median :35.00   other   : 67
 Mean   : 9.024   Mean   :13.02   Mean   :17.82   Mean   :36.83
 3rd Qu.:11.250   3rd Qu.:15.00   3rd Qu.:26.00   3rd Qu.:44.00
 Max.   :44.500   Max.   :18.00   Max.   :55.00   Max.   :64.00
     occupation           sector    union     married
 worker    :156   manufacturing: 99   no :438   no :184
 technical :105   construction : 24   yes: 96   yes:350
 services  : 83   other        :411
 office    : 97
 sales     : 38
 management: 55
```

FIGURE 13 – resumé de quelques mesures numériques des variables

4.3.2 Exploration graphique des données

Il est également souvent très utile de représenter graphiquement les données. Ce qui permet de se faire une idée de la distribution ou encore de repérer facilement des don-

nées aberrantes. L'analyse bivariée permet d'étudier les interactions entre les différentes variables. Selon les cas, on a réalisé :

— Les histogrammes pour les variables quantitatives (Figure 14)

— les boîtes à moustaches pour les variables qualitatives (voir figure 15)

— la matrice des coefficients de corrélation et sa représentation graphique pour les variables quantitatives (voir table 1 et figure 16)

— le nuage de points pour toutes les variables quantitatives et quantitatives (voir Figure 17)

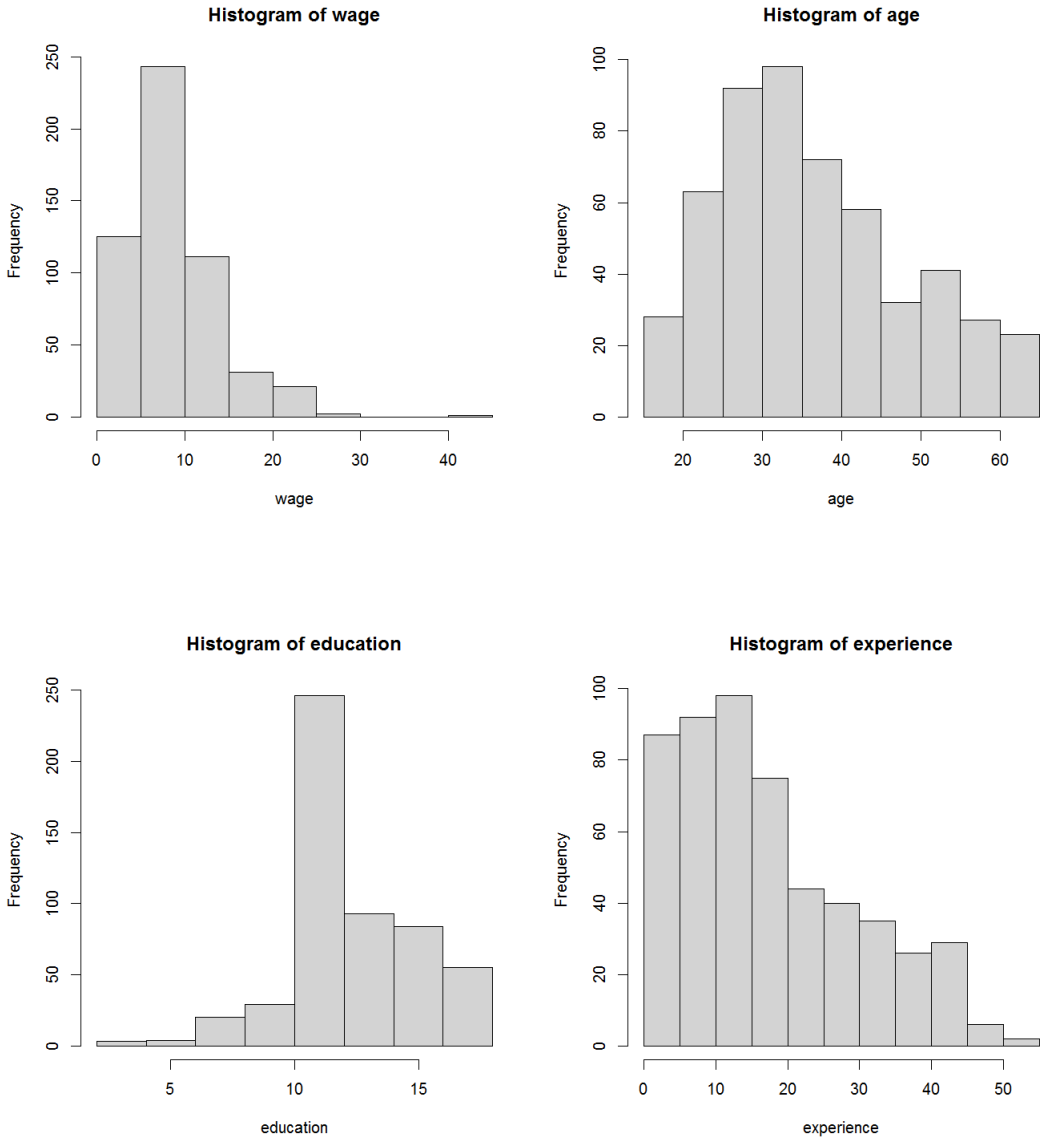


FIGURE 14 – Histogrammmes des variables quantitatives

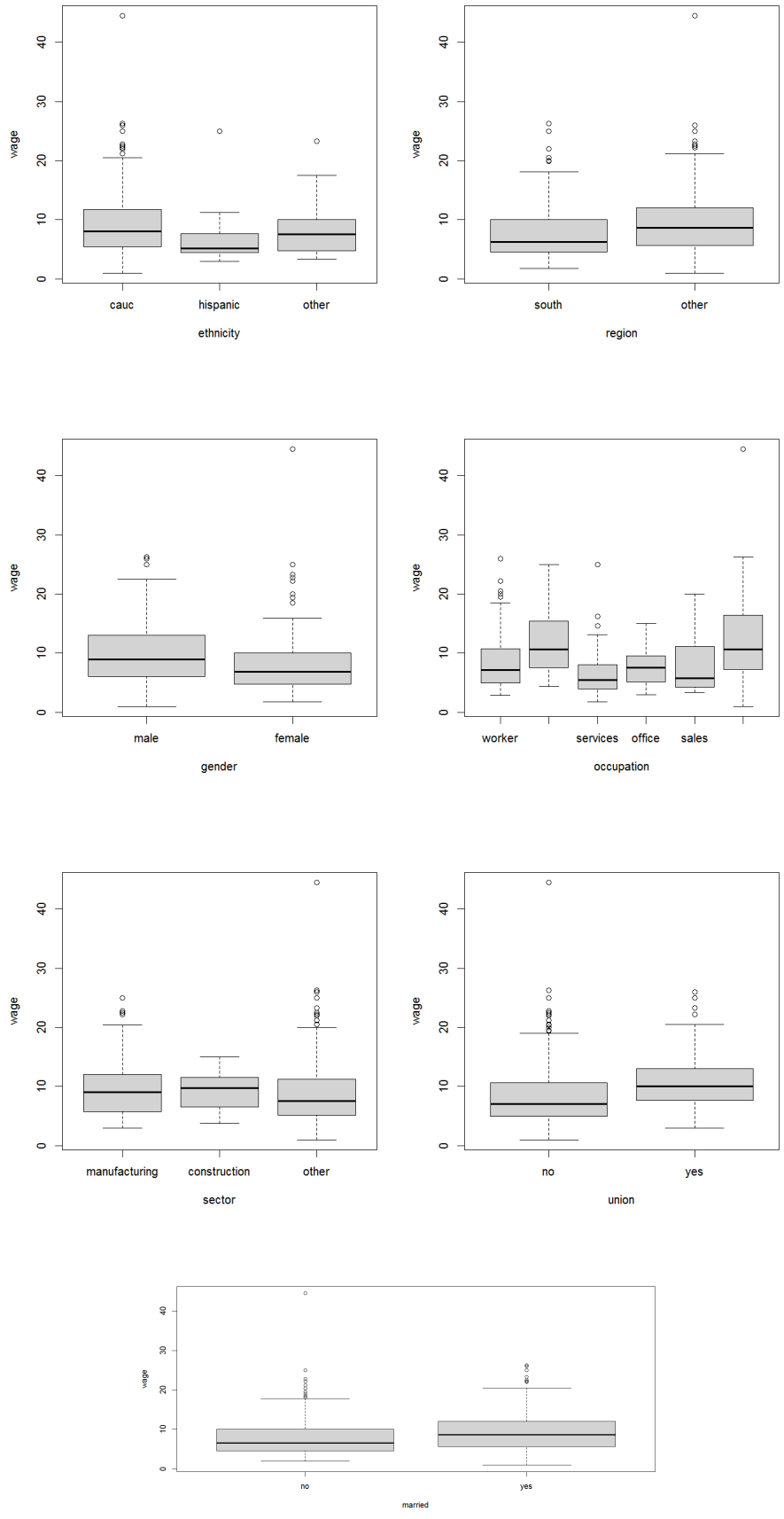


FIGURE 15 – Le boites à moustache des variables qualitatives selon le salaire

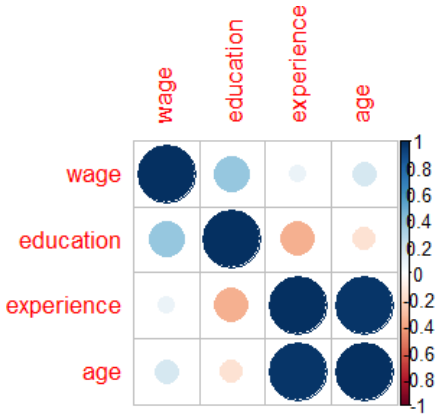


FIGURE 16 – :Diagramme des corrélations

	salaire	éducation	expérience	âge
salaire	1	0.380	0.090	0.180
éducation	0.380	1	−0.350	−0.150
expérience	0.090	−0.350	1	0.980
âge	0.180	−0.150	0.980	1

TABLE 1 – matrice de corrélation.

Évidemment, la dépendance (ou corrélation) est souhaitable entre une variable explicative et la variable à expliquer mais ne l'est pas entre les variables explicatives. Si des variables explicatives ont des comportements très proches (par exemple une corrélation proche de 1 ou −1), il peut être utile de supprimer une des variables pour réduire la dimension du modèle. De manière générale, on peut supprimer une variable explicative si son information est contenue dans plusieurs autres variables.

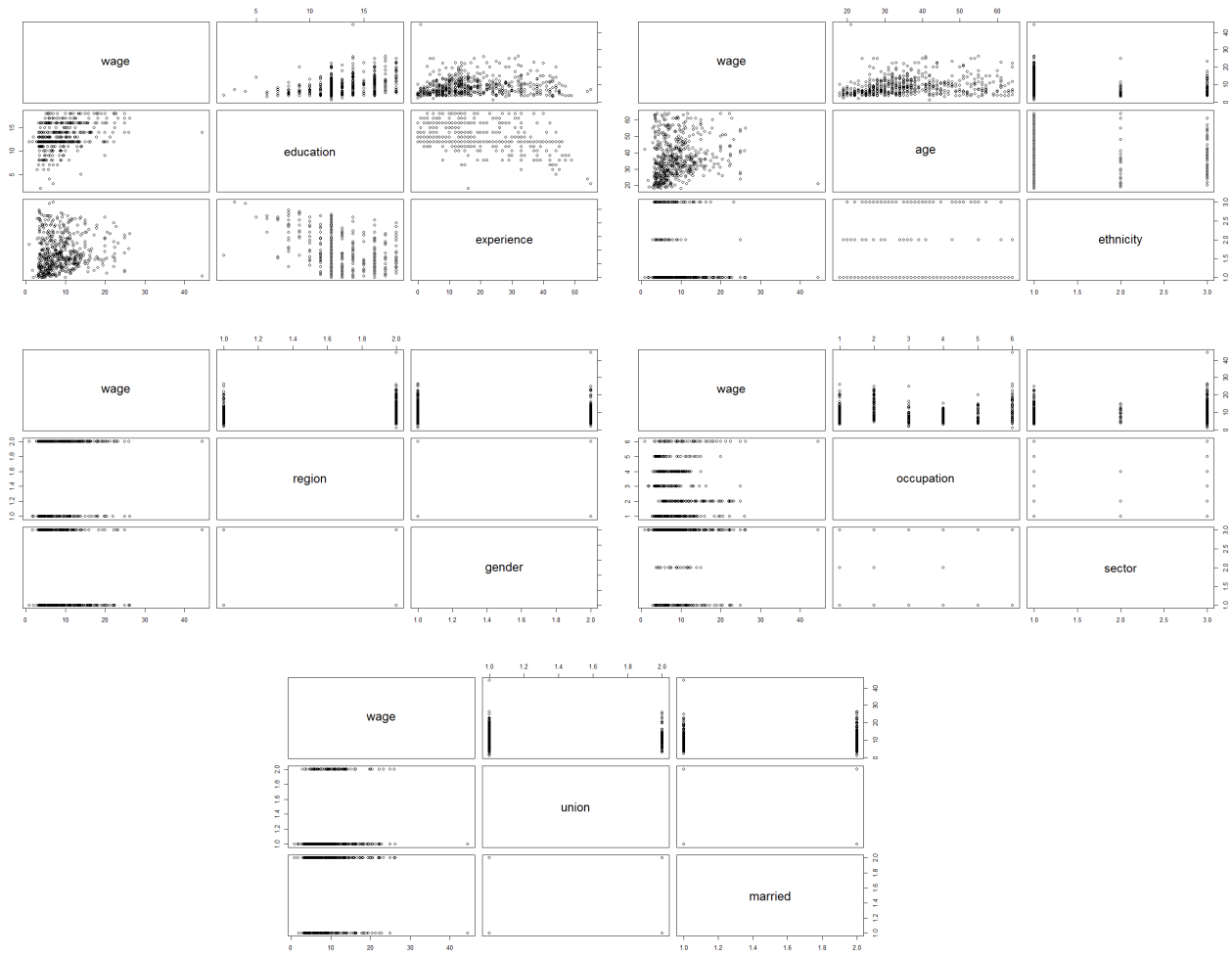


FIGURE 17 – Nuage de points

4.4 CONSTRUCTION DES MODÈLES

Avant de construire les modèles, nous commençons par diviser les données en 3 ensembles : "traîn" , "validation" et "test". Le modèle sera construit sur l'ensemble "train" et sa précision sera vérifiée sur l'ensemble " validation". Ensuite nous utiliseront l'ensemble "test" pour choisir le meilleur modèle parmi les modèles issu des différentes méthodes d'apprentissage.

Nous avons entraîné trois modèles de machine learning : la régression multiple, les arbres de décision et les KNN(les K plus proches voisins) Pour comparer les modèles nous utilisons principalement le RMSE et le coefficient de détermination R^2, rappelons que plus ce dernier indicateur est proche de 1, plus l'on peut considérer que notre modèle est bon.

4.4.1 La régression multiple

Nous avons essayé plusieurs combinaisons, et nous avons retenu ces 4 modèles :

Le modèle 1 : Dans le premier modèle qu'on appelé " reg 1" on fait intervenir toutes les variables explicatives d'une manière linéaire.

```
reg1 <- lm(wage ~ ., data=donnees)
```

Les résultats du modèle "reg 1" sont résumés ci-dessus :

```
Call:
lm(formula = wage ~ ., data = donnees)

Residuals:
    Min      1Q  Median      3Q     Max
-9.3788 -2.2556 -0.5418  1.8542 16.8371

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          -0.1764     6.0573  -0.029 0.976786
education             0.8529     0.9812   0.869 0.385424
experience            0.2461     0.9737   0.253 0.800648
age                  -0.1506     0.9724  -0.155 0.877026
ethnicityhispanic    -1.7974     0.9994  -1.798 0.073111 .
ethnicityother       -0.9712     0.6397  -1.518 0.130035
regionother           0.4476     0.4835   0.926 0.355334
genderfemale         -2.0508     0.4957  -4.137 4.55e-05 ***
occupationtechnical   1.8946     0.8449   2.242 0.025660 *
occupationservices   -0.9939     0.8029  -1.238 0.216697
occupationoffice     -0.1916     0.8172  -0.234 0.814765
occupationsales      -0.5076     0.9666  -0.525 0.599828
occupationmanagement  0.9756     0.9385   1.040 0.299385
sectorconstruction   -1.6771     1.3570  -1.236 0.217455
sectorother          -1.0503     0.6492  -1.618 0.106740
unionyes              2.4229     0.6294   3.850 0.000144 ***
marriedyes            0.3667     0.4707   0.779 0.436542
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.829 on 303 degrees of freedom
Multiple R-squared:  0.3674,    Adjusted R-squared:  0.334
F-statistic:    11 on 16 and 303 DF,  p-value: < 2.2e-16
```

Le modèle 2 : Dans le deuxième modèle que nous appelons " reg 2" nous avons écarté la variable $X_3 =$ âge car nous avons constaté dans l'analyse bivariée une forte corrélation entre cette variable et la variable $X_2 =$ experience

```
reg2 <- lm(wage~ .-age, data = donnees)
```

Les résultats du modèle "reg 2" sont résumés ci-dessus :

```
Call:
lm(formula = wage ~ . - age, data = donnees)

Residuals:
    Min      1Q  Median      3Q     Max
-9.3756 -2.2498 -0.5422  1.8586 16.8387

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          -1.07616    1.71115  -0.629 0.529877
education             0.70209    0.12152   5.778 1.87e-08 ***
experience            0.09532    0.02037   4.680 4.33e-06 ***
ethnicityhispanic    -1.80041    0.99764  -1.805 0.072114 .
ethnicityother       -0.97115    0.63870  -1.521 0.129420
regionother           0.44969    0.48256   0.932 0.352138
genderfemale         -2.04521    0.49357  -4.144 4.43e-05 ***
occupationtechnical   1.90185    0.84225   2.258 0.024650 *
occupationservices   -0.99730    0.80127  -1.245 0.214224
occupationoffice     -0.19571    0.81551  -0.240 0.810505
occupationsales      -0.50901    0.96498  -0.527 0.598246
occupationmanagement  0.97430    0.93699   1.040 0.299248
sectorconstruction   -1.67222    1.35442  -1.235 0.217919
sectorother          -1.04725    0.64785  -1.616 0.107026
unionyes              2.42224    0.62833   3.855 0.000141 ***
marriedyes            0.36226    0.46903   0.772 0.440505
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.823 on 304 degrees of freedom
Multiple R-squared:  0.3673,    Adjusted R-squared:  0.3361
F-statistic: 11.77 on 15 and 304 DF,  p-value: < 2.2e-16
```

Le modèle 3 : dans ce troisième modèle qu'on a appelé " reg 3" en plus de toutes les variables nous avons aussi fait intervenir la variable expérience élevée au carré X_2^2, en supposons qu'il existe une relation quadratique entre le salaire et l'expérience :

```
reg3 <- lm(wage~ .+I(experience^2), data = donnees)
```

Les résultats du modèle "reg 3" sont résumés ci-dessus :

```
Residuals:
    Min      1Q  Median      3Q     Max
-9.8652 -2.2872 -0.6077  1.7774 16.7804

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          1.001084   6.037428   0.166 0.868415
education            1.067827   0.978954   1.091 0.276237
experience           0.637547   0.982023   0.649 0.516691
age                 -0.403864   0.972007  -0.415 0.678075
ethnicityhispanic   -1.516398   1.000101  -1.516 0.130503
ethnicityother      -1.093814   0.637562  -1.716 0.087257 .
regionother          0.441825   0.480193   0.920 0.358256
genderfemale        -2.108047   0.492871  -4.277 2.54e-05 ***
occupationtechnical  1.988016   0.840058   2.367 0.018587 *
occupationservices  -0.735716   0.805266  -0.914 0.361640
occupationoffice    -0.053390   0.813846  -0.066 0.947738
occupationsales     -0.329973   0.963031  -0.343 0.732107
occupationmanagement 1.058829   0.932756   1.135 0.257208
sectorconstruction  -1.848423   1.349666  -1.370 0.171848
sectorother         -1.138569   0.645859  -1.763 0.078933 .
unionyes             2.411784   0.625022   3.859 0.000139 ***
marriedyes           0.201729   0.472930   0.427 0.670010
I(experience^2)     -0.003216   0.001406  -2.287 0.022871 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.802 on 302 degrees of freedom
Multiple R-squared:  0.3782,    Adjusted R-squared:  0.3432
F-statistic:  10.8 on 17 and 302 DF,  p-value: < 2.2e-16
```

Modèle 4 : dans ce quatrième modèle qu'on a appelé "reg 4", en plus de toutes les variables et l'expérience élevée au carré, nous avons fait intervenir une interaction entre les 3 variables : éducation, expérience et âge.

```
reg4 <- lm(wage~ (education + experience+age )^2+I(experience^2)+ ethnicity+region +gender
           + occupation + sector+union+married, data = donnees)
```

Les résultats du modèle "reg 4" sont résumé ci-dessus :

```
Call:
lm(formula = wage ~ (education + experience + age)^2 + I(experience^2) +
    ethnicity + region + gender + occupation + sector + union +
    married, data = donnees)

Residuals:
   Min     1Q Median     3Q    Max
-9.853 -2.334 -0.659  1.846 16.808

Coefficients: (1 not defined because of singularities)
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          5.9903047 13.0522371   0.459 0.646603
education            0.8245559  1.1705699   0.704 0.481727
experience           0.6962673  1.1212475   0.621 0.535087
age                 -0.5991059  1.1558626  -0.518 0.604618
I(experience^2)     -0.0025041  0.0015892  -1.576 0.116147
ethnicityhispanic   -1.4865547  1.0052621  -1.479 0.140250
ethnicityother      -1.0832457  0.6411410  -1.690 0.092151 .
regionother          0.3992052  0.4830234   0.826 0.409193
genderfemale        -2.1288344  0.4944494  -4.305 2.26e-05 ***
occupationtechnical  2.0953492  0.8540479   2.453 0.014719 *
occupationservices  -0.7037601  0.8076223  -0.871 0.384234
occupationoffice    -0.0343008  0.8182635  -0.042 0.966591
occupationsales     -0.3333225  0.9744234  -0.342 0.732537
occupationmanagement 1.1242166  0.9374272   1.199 0.231374
sectorconstruction  -1.8945476  1.3535543  -1.400 0.162641
sectorother         -1.1058564  0.6487860  -1.705 0.089323 .
unionyes             2.4347628  0.6267788   3.885 0.000126 ***
marriedyes           0.2331826  0.4748727   0.491 0.623755
education:experience 0.0004009  0.0295886   0.014 0.989198
education:age        0.0081279  0.0348203   0.233 0.815591
experience:age              NA         NA      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.809 on 300 degrees of freedom
Multiple R-squared:  0.3801,    Adjusted R-squared:  0.3409
F-statistic: 9.683 on 19 and 300 DF,  p-value: < 2.2e-16
```

Les tables ci-dessus résume les différentes erreurs et le coefficient de détermination R^2 dans l'échantillon d'apprentissage "train" (Table 2) et dans l'échantillon de test (Table 3)

	reg1	reg2	reg3	reg4
MSE	13.88	13.88	13.64	13.60
MAE	2.73	2.73	2.70	2.71
RMSE	3.73	3.73	3.69	3.69
R2	0.37	0.37	0.38	0.38

TABLE 2 – : Dans l'ensemble train

	reg1	reg2	reg3	reg4
MSE	18.04	18.05	16.99	17.23
MAE	3.19	3.19	3.11	3.12
RMSE	4.25	4.25	4.12	4.15
R2	0.34	0.34	0.37	0.36

TABLE 3 – : Dans l'ensemble de validation

On constate que les Modèles reg1 et reg2 sont quasiment identique en vue des erreurs et du R^2 que ce soit dans l'échantillon d'apprentissage ou celui de validation cela veut dire

que le fait d'avoir écarté la variable âge dans la régression n'a eu aucun effet. On relève la même chose pour les modèle reg3 et reg4 dans l'échantillon d'apprentissage par contre dans l'échantillon de test le troisième modèle est un peu plus performant donc on prend le modèle reg3 pour la méthode de régression multiple.

Remarque : quand on a une légère différence de performance entre deux modèles on opte toujours pour le modèle le plus simple même quand la performance est en faveur du modèle le plus complexe.

4.4.2 Les arbres de décision

Dans la construction des arbres de décision, les données sont divisées en sous-ensembles où à chaque sous-ensemble correspond une valeur de la variable cible y, en général, on prend la moyenne des valeurs de y appartenant à ce sous ensemble. La meilleure division est celle qui donne les sous-ensembles les plus homogènes c'est-à-dire des sous-ensembles purs. Les hyper-paramètres à régler pour construire l'arbre sont :

- *maxdepth* : la profondeur de l'arbre, (quantifiée par les niveaux/étages de l'arbre à partir de la racine initialisée à 0).
- *minsplit* : le nombre minimum d'individus présents au niveau d'un nœud pour envisager une coupure (en dessous de ce nombre, le nœud devient alors une feuille)
- *minbucket* : le nombre minimum d'individus présents dans une feuille ,
- *cp* : un paramètre de pénalisation de la complexité de l'arbre, en effet, plus le cp est petit, plus grand est l'arbre de régression (beaucoup de noeuds), plus il est grand, plus la complexité est pénalisée.

Le modèle 1 : le premier modèle (mtree1), est un modèle basique, on utilise la fonction *rpart* avec les paramètres par défaut ; la commande *rpart* pose : $maxdepth = 30$, $minsplit = 20$, $minbucket = minsplit/3$ et $cp = 0.01$.

```
mtree1 <- rpart(formula = wage ~ ., data = donnees,
           method  = "anova",cp=0.01)
```

Pour chaque valeur de cp seuil, on a le nombre de split (division) de l'arbre correspondant (nsplit), l'erreur (de la validation croisée) "xerror" et son écart-type "xstd" voir les résultats du modèle 1 ci-dessus :

```
> printcp(mtree1)

Regression tree:
rpart(formula = wage ~ ., data = donnees, method = "anova", cp = 0.01)

Variables actually used in tree construction:
[1] age        education  ethnicity  experience gender     married    occupation
[8] union

Root node error: 7021.7/320 = 21.943

n= 320

         CP nsplit rel error  xerror     xstd
1  0.150139      0   1.00000 1.01241 0.109175
2  0.059047      1   0.84986 0.89153 0.092479
3  0.041587      2   0.79081 0.89460 0.098462
4  0.038550      3   0.74923 0.88066 0.095609
5  0.033663      4   0.71068 0.86546 0.095927
6  0.020702      5   0.67701 0.82836 0.094000
7  0.016383      6   0.65631 0.84927 0.093173
8  0.016371      7   0.63993 0.84390 0.092976
9  0.011895      8   0.62356 0.83406 0.090038
10 0.011334      9   0.61166 0.83832 0.090518
11 0.011061     11   0.58900 0.83981 0.089894
12 0.010000     12   0.57794 0.83087 0.088904
```

Dans la figure 18 ci-dessous, les erreurs sont représentées en fonction du paramètre de complexité cp (en bas) ou du nombre de nœuds terminaux(en haut). Des petits traits verticaux, correspondant à plus ou moins un écart-type de l'erreur, sont ajoutés sur chaque point. La ligne en pointillés horizontale correspond à l'erreur minimale plus son écart-type. Elle indique que toutes les erreurs en dessous de cette ligne peuvent être considérées comme étant du même ordre que l'erreur optimale. Pour des erreurs très proche on choisira toujours l'arbre le plus simple.

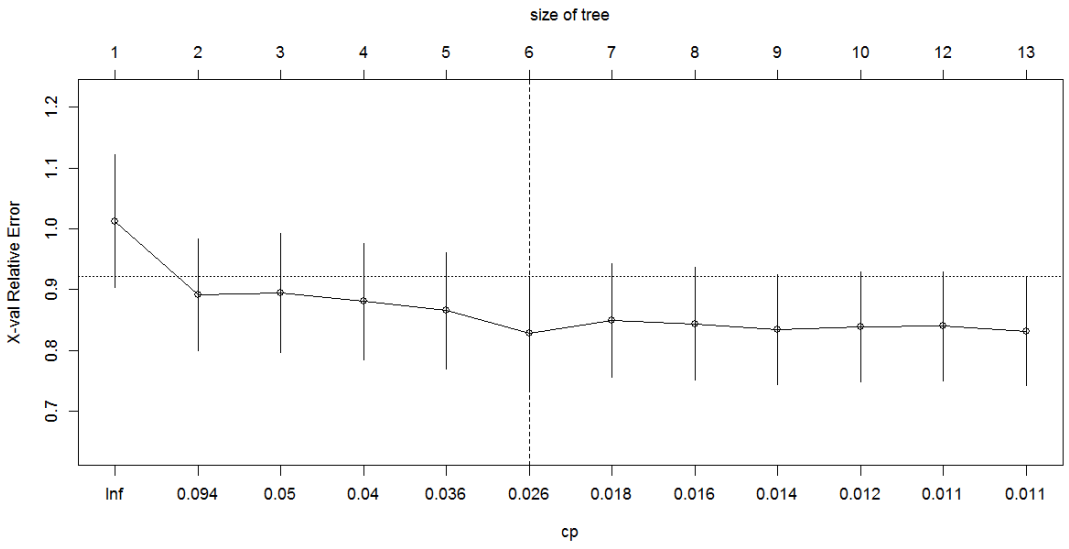


FIGURE 18 – Erreur de prédiction en fonction de cp et de la taille de l'arbre mtree1

Ci-dessous deux représentations graphiques de l'arbre mtree1 (figure 19)

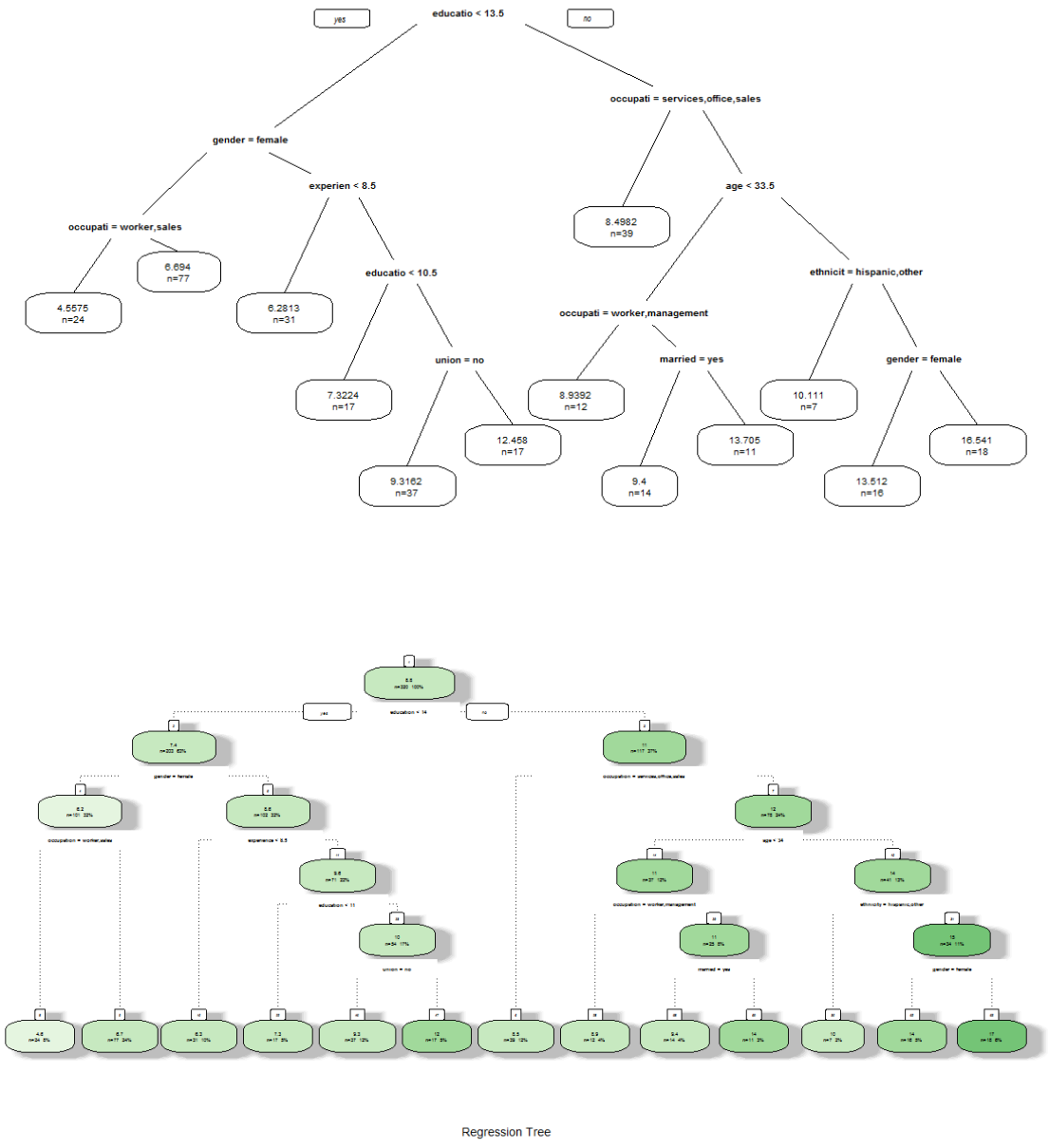


FIGURE 19 – Deux représentations de l'arbre mtree 1

Élagage de l'arbre

Pour élaguer l'arbre, on prend en général la première valeur de cp (i.e. la plus grande) qui est à moins d'un écart-type du minimum de xerror (le trait horizontal). Une fois que la valeur de cp est choisie, on peut récupérer l'arbre correspondant avec la fonction *prunne*. Pour le modèle 1, on va élaguer l'arbre au meilleur cp qui est $cp = 0.026$ et à la taille qui correspond qui est $maxdepth = 6$ (voir la figure 18), on obtient l'arbre optimal extrait du modèle 1 (figure 20)

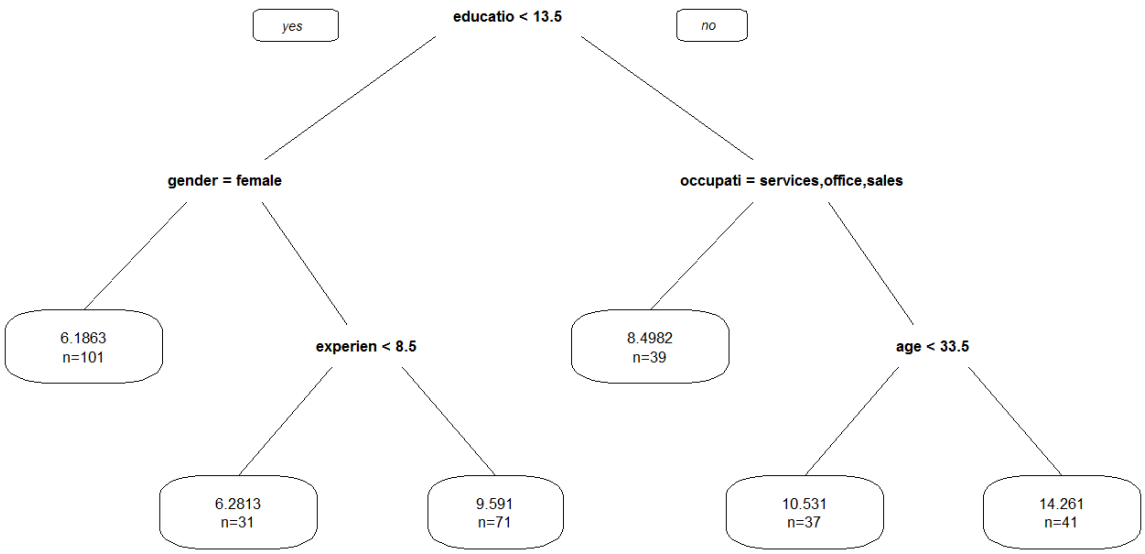


FIGURE 20 – arbre optimale du modèle 1

Le modèle 2 Le deuxième modèle (mtree2) consiste en la construction d'un arbre d'une profondeur maximale, ensuite comme on l'a fait dans le modèle précédent, L'extraction de l'arbre optimal se réalise avec la fonction prune, En récupérant tout d'abord le paramètre de complexité cp optimal de façon automatique, pour la représentation graphique voir la figure 21 et la figure 22 nous donne la représentaton de l'erreur en fonction de cp et la taille de l'arbre.

```
mtree2 <- rpart(wage~.,data=donnees, control=rpart.control(minsplit=5,cp=0))
```

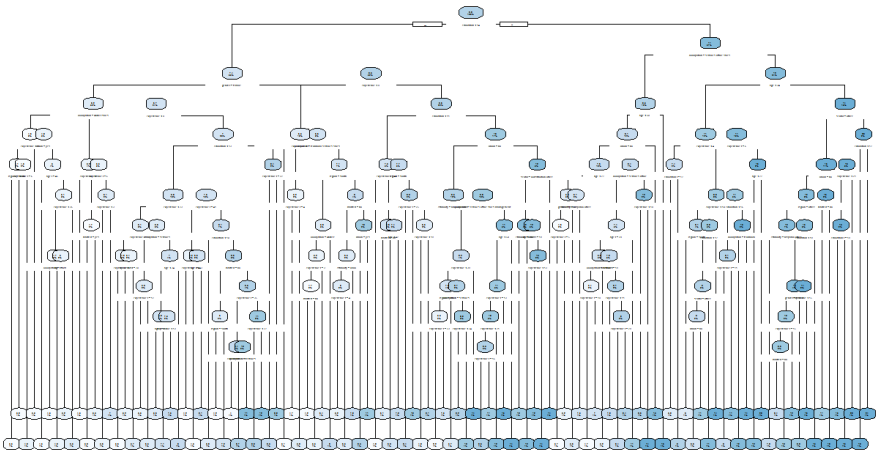


FIGURE 21 – Représentation graphique de l'arbre mtree2

```
Regression tree:
rpart(formula = wage ~ ., data = donnees, control = rpart.control(minsplit =
    cp = 0))

Variables actually used in tree construction:
 [1] age        education  ethnicity  experience gender     married    occup
 [8] region     sector     union

Root node error: 7021.7/320 = 21.943

n= 320

           CP nsplit rel error  xerror     xstd
1  1.5014e-01      0   1.00000 1.00391 0.108641
2  5.9047e-02      1   0.84986 0.89765 0.093082
3  4.1587e-02      2   0.79081 0.88131 0.092202
4  3.8550e-02      3   0.74923 0.86052 0.092914
5  3.3663e-02      4   0.71068 0.85640 0.093503
6  2.7875e-02      5   0.67701 0.83263 0.092319
7  1.9509e-02      6   0.64914 0.84138 0.092260
8  1.6383e-02      8   0.61012 0.87847 0.094460
9  1.6371e-02      9   0.59374 0.86529 0.093532
10 1.6205e-02     10   0.57737 0.85881 0.093424
11 1.5310e-02     11   0.56116 0.86695 0.094369
12 1.4062e-02     12   0.54585 0.88170 0.095886
13 1.2827e-02     13   0.53179 0.88045 0.096251
14 1.1895e-02     14   0.51896 0.86545 0.095638
15 9.9928e-03     15   0.50707 0.86897 0.093996
16 9.7795e-03     18   0.47688 0.87225 0.094680
17 9.6582e-03     21   0.44754 0.88996 0.094640
18 8.1221e-03     23   0.42823 0.90878 0.097331
19 6.7865e-03     24   0.42011 0.90917 0.094473
20 6.0027e-03     25   0.41332 0.93557 0.095550
21 5.6930e-03     28   0.39531 0.96405 0.098330
22 5.3410e-03     29   0.38962 0.99113 0.100294
23 5.2764e-03     30   0.38428 0.99335 0.099092
24 4.9113e-03     31   0.37900 0.99599 0.099106
25 4.7649e-03     32   0.37409 1.00421 0.099660
26 4.7272e-03     36   0.35503 1.01289 0.100556
27 4.4225e-03     37   0.35030 1.01681 0.101010
28 4.3583e-03     43   0.32149 1.01511 0.101047
29 3.7837e-03     44   0.31714 1.01127 0.100728
30 3.6418e-03     47   0.30579 1.02075 0.103074
31 3.5735e-03     48   0.30214 1.02612 0.103057
32 3.5355e-03     49   0.29857 1.02074 0.102903
33 3.1830e-03     50   0.29503 1.03642 0.103163
34 3.0847e-03     51   0.29185 1.04405 0.103490
35 2.7393e-03     53   0.28568 1.04532 0.103818
36 2.7279e-03     54   0.28294 1.03799 0.103913
36 2.7279e-03     54   0.28294 1.03799 0.103913
37 2.1919e-03     56   0.27749 1.03842 0.102848
38 2.1693e-03     58   0.27310 1.05339 0.107056
39 2.0961e-03     59   0.27093 1.05115 0.107109
40 2.0933e-03     61   0.26674 1.05238 0.107083
41 2.0046e-03     62   0.26465 1.05245 0.107402
42 1.7404e-03     63   0.26264 1.05626 0.107292
43 1.6761e-03     64   0.26090 1.04681 0.107305
44 1.5777e-03     65   0.25923 1.05051 0.107601
45 1.5505e-03     66   0.25765 1.06137 0.109186
46 1.5381e-03     67   0.25610 1.06137 0.109186
47 1.5324e-03     68   0.25456 1.05938 0.109052
48 1.5070e-03     69   0.25303 1.06242 0.109066
49 1.1085e-03     70   0.25152 1.06455 0.109030
50 1.0824e-03     71   0.25041 1.06999 0.110222
51 1.0420e-03     72   0.24933 1.07512 0.110490
52 9.6404e-04     73   0.24829 1.07729 0.110555
53 9.1254e-04     74   0.24732 1.07686 0.110502
54 7.7842e-04     75   0.24641 1.07652 0.110242
55 7.6649e-04     76   0.24563 1.07690 0.110050
56 6.8549e-04     77   0.24487 1.07576 0.109738
57 6.2663e-04     78   0.24418 1.08060 0.109660
58 6.1911e-04     79   0.24356 1.08055 0.109662
59 5.9201e-04     80   0.24294 1.08007 0.109682
60 5.8965e-04     82   0.24175 1.08008 0.109681
61 5.8106e-04     84   0.24057 1.07951 0.109696
62 5.4387e-04     85   0.23999 1.08061 0.109753
63 5.1853e-04     86   0.23945 1.07998 0.109753
64 5.1540e-04     87   0.23893 1.07965 0.109762
65 4.9680e-04     88   0.23841 1.07965 0.109762
66 4.4197e-04     89   0.23792 1.07911 0.109775
67 4.1264e-04     93   0.23602 1.07799 0.109744
68 3.8966e-04     95   0.23519 1.07808 0.109742
69 3.8694e-04     96   0.23480 1.07676 0.109705
70 3.3946e-04     97   0.23441 1.07662 0.109709
71 3.0610e-04     98   0.23407 1.07925 0.110069
72 2.8146e-04     99   0.23377 1.07985 0.110056
73 2.6525e-04    100   0.23349 1.07988 0.110055
74 2.5576e-04    101   0.23322 1.07959 0.110050
75 2.4662e-04    102   0.23297 1.08057 0.110036
76 2.3328e-04    103   0.23272 1.08057 0.110036
77 2.1951e-04    104   0.23249 1.08115 0.110025
78 1.7204e-04    105   0.23227 1.08056 0.110038
79 1.7147e-04    106   0.23209 1.08083 0.110034
80 1.4822e-04    107   0.23192 1.08128 0.110022
81 1.4622e-04    108   0.23178 1.07991 0.110041
82 1.1987e-04    109   0.23163 1.07991 0.110041
83 4.8611e-05    110   0.23151 1.08093 0.110062
84 3.1348e-05    111   0.23146 1.08090 0.110157
```

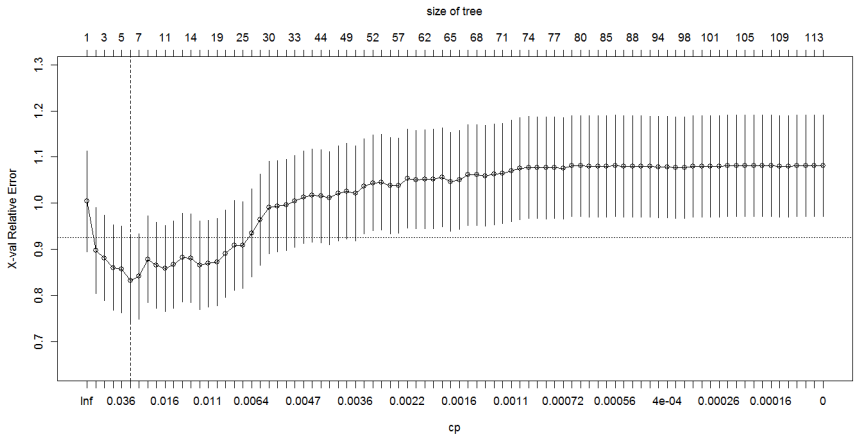


FIGURE 22 – L'erreur en fonction de cp et taille de l'arbre mtree2

Elagage de l'arbre :

Après élagage de l'arbre avec la fonction *prunne* au meilleur cp qui est $cp = 0.027$ et à la taille qui correspond qui est $maxdepth = 6$, on obtient l'arbre optimal du modèle 2 représentée dans la figue 23 ci-dessus :

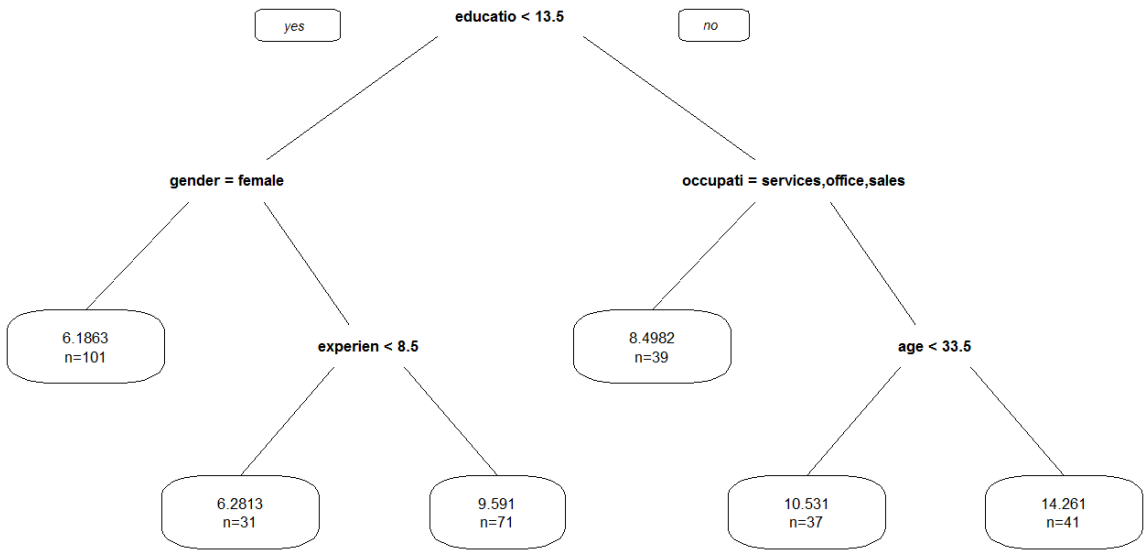


FIGURE 23 – L'arbre optimal du modèle 2

Le modèle 3 : Pour le troisième modèle (mtree3) on a opté pour une maximisation automatique des hyper-paramètres, on a construit une fonction "expand.grid" qui va simplement fixer pour chaque hyper-paramètres, un ensemble de valeurs qu'il peut prendre. Ensuite, pour chaque combinaison d'hyper-paramètres, on va entraîner un arbre maximal, et conserver les résultats de performances en mémoire. Il suffira ensuite de prendre les hyper-paramètres pour lesquels les performances sont les meilleurs.

```
hyper_grid <- expand.grid(
  minsplit = seq(3, 30, 1),
  maxdepth = seq(3, 25, 1)
```

La fonction "expand.grid " fait varier "$minsplit$" dans l'ensemble $\{3, \ldots, 30\}$ et "$maxdepth$" dans l'ensemble $\{3, \ldots, 25\}$ avec un pas égal à 1, le logiciel aura un total de 644 modèles à entraîner.

L'arbre optimal obtenu est représenté par la figure 25 , l'erreur de prédiction en fonction de cp et la taille de l'arbre ci-dessous est représentée graphiquement par la figure 24 :

```
Regression tree:
rpart(formula = wage ~ ., data = donnees, method = "anova", control = list(minsplit = 3
0,
    maxdepth = 15, cp = 0))

Variables actually used in tree construction:
[1] age        education  ethnicity  experience gender     occupation union

Root node error: 7021.7/320 = 21.943

n= 320

          CP nsplit rel error  xerror     xstd
1  0.1501389      0   1.00000 1.00526 0.108659
2  0.0590472      1   0.84986 0.92134 0.094759
3  0.0415867      2   0.79081 0.89720 0.098313
4  0.0385497      3   0.74923 0.88774 0.095913
5  0.0336628      4   0.71068 0.87880 0.096445
6  0.0207019      5   0.67701 0.83303 0.095291
7  0.0163831      6   0.65631 0.81654 0.089426
8  0.0163707      7   0.63993 0.79977 0.088952
9  0.0118947      8   0.62356 0.77242 0.087084
10 0.0110610      9   0.61166 0.78677 0.088562
11 0.0081522     10   0.60060 0.79852 0.089123
12 0.0073922     11   0.59245 0.78397 0.086651
13 0.0064053     12   0.58506 0.79422 0.087089
14 0.0053410     13   0.57865 0.78804 0.086805
15 0.0041868     14   0.57331 0.79007 0.086722
16 0.0036338     15   0.56913 0.79388 0.086835
17 0.0026146     16   0.56549 0.79287 0.086854
18 0.0000000     17   0.56288 0.78835 0.086654
> plotcp(optimal_tree)
```

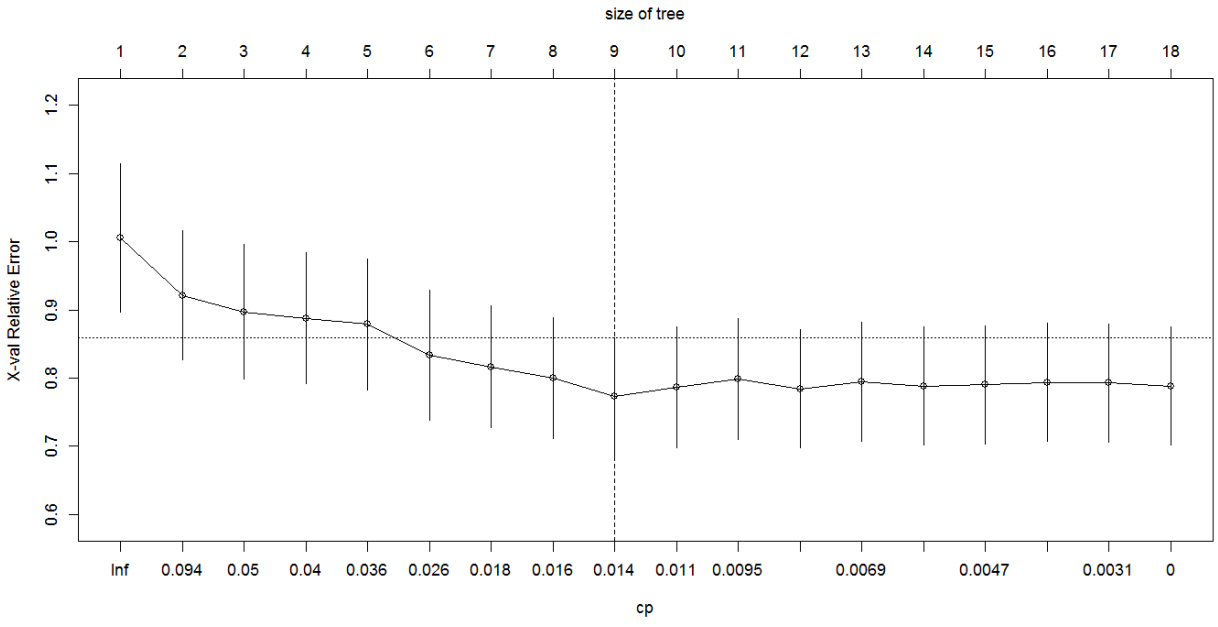


FIGURE 24 – l'erreur de prédiction en fonction de cp et la taille de l'arbre mtree3

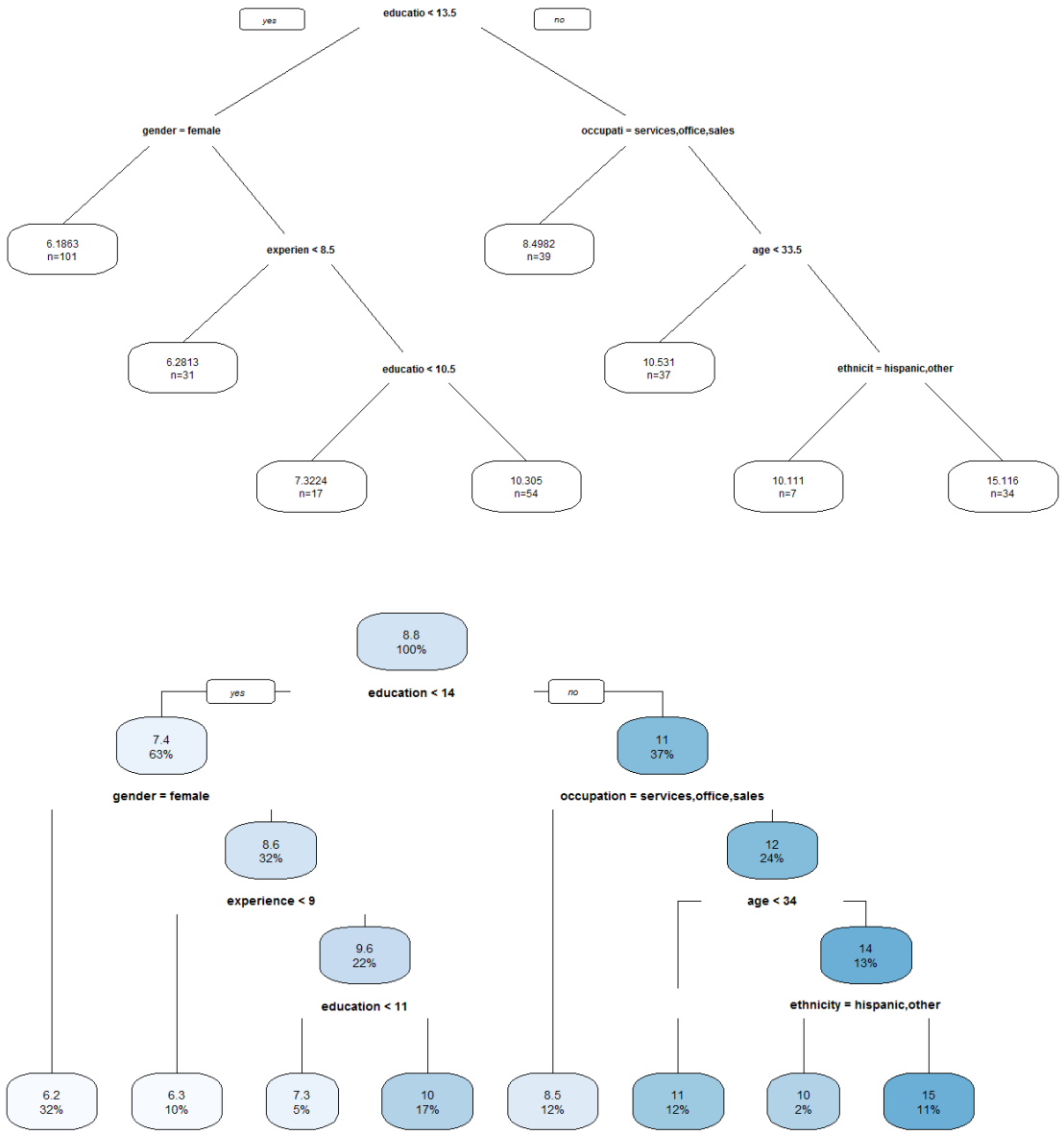


FIGURE 25 – Deux représentations de l'arbre mtree 3

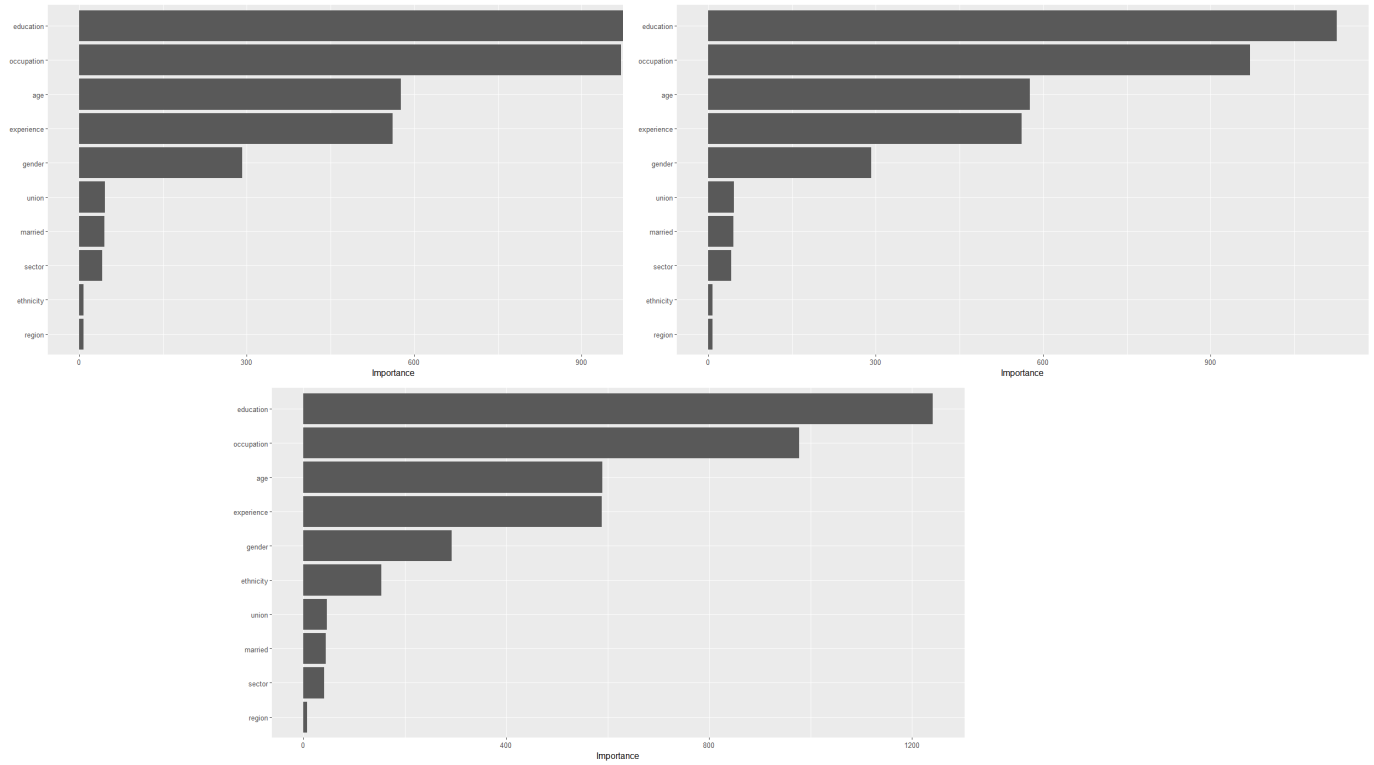


FIGURE 26 – Importance des variables pour les trois arbres

La figure 26 ci-dessus est la représentation graphique de l'importance des variables explicatives pour les trois arbres :
Ci-dessous la représentation graphique de R^2 et de l'erreur de prédiction en fonction du nombre de division pour les trois arbres optimaux (figures 27 et 28)

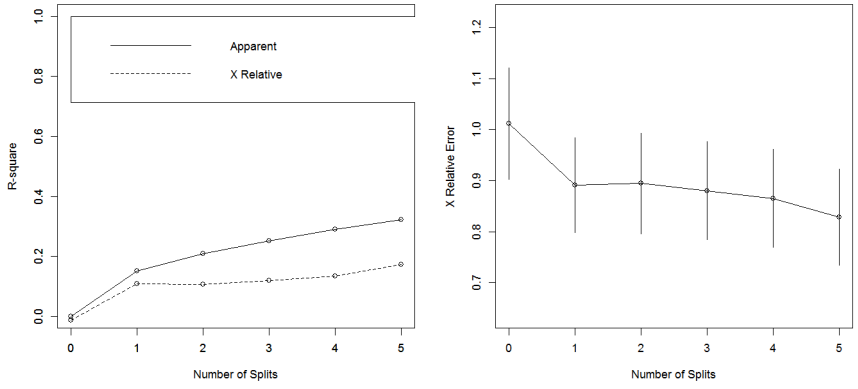


FIGURE 27 – Erreur de prédiction et R^2 en fonction de nombre de division de l'arbre optimal du modèle 1

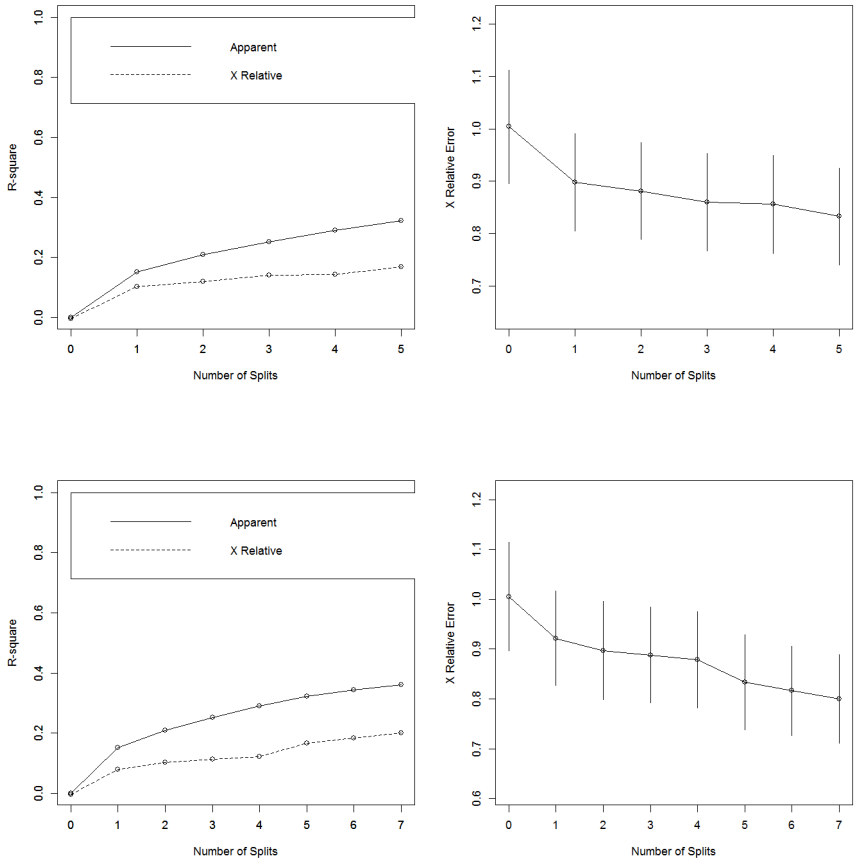


FIGURE 28 – Erreur de prédiction et $R2$ en fonction du nombre de division de l'arbre optimal des modèles 2 et 3

Pour choisir l'arbre optimal parmi ces trois arbres, on va faire des prédictions sur l'échantillon de "validation" et on calcule les erreurs : MSE, MAE, RSME et le coefficient de détermination R^2, La table 4 ci-dessous résume les résultats obtenus :

	Tree1	Tree2	Tree3
MSE	22.01	13.88	13.60
MAE	3.47	2.73	2.71
RMSE	4.69	3.73	3.69
R2	0.24	0.37	0.38

TABLE 4 – Les différentes erreurs et R^2 des trois arbres

On remarque que les deux modèles Tree2 et Tree3 sont très proche en vus des différentes erreurs et le coefficients de détermination R^2, mais sont largement meilleurs que le modèle Tree1. Comme on l'a déja mentionné ci-dessus, on opte pour l'arbre le plus simple c.à.d le modèle Tree2, car la différence de performance n'est pas significative.

4.4.3 Les KNN :

Comme on l'a vus dans le chapitre 3, l'algorithme KNN n'a pas besoin de construire un modèle au préalable c.à.d pour les KNN il n'existe pas de phase d'apprentissage proprement dite. On effectue directement des prédictions sur les données de l'échantillon d'entraînement, et le but est de trouver le meilleur k.

Modèle 1 : dans le premier modèle nous utilisons la fonction "*knnreg*" avec ses paramètres par défaut (k=5)

```
model1 = knnreg(wage ~ ., data =dfTraining)
```

Le modèle 2 : Nous cherchons le k optimal en utilisant la fonction *expand.grid* qui fait varier k de 1 à 100 avec des pas égal à 2, et une validation croisée, ensuite on fait des prédiction avec le k optimal sur l'échantillon de test, la figure 29 ci-dessous représente l'erreur RMSE en fonction de k, on voit que la valeur optimal de k est égal à 9

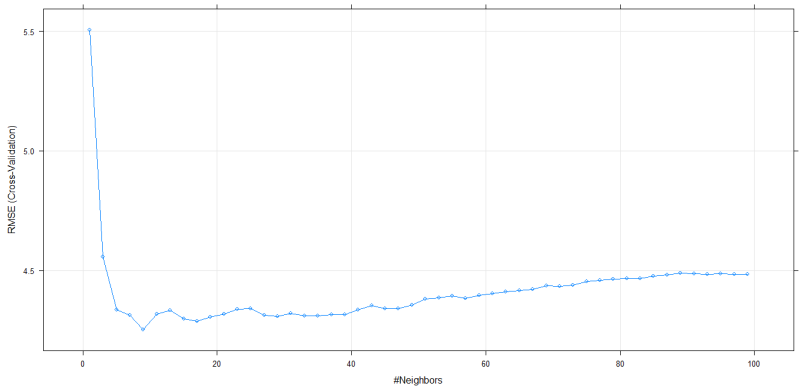


FIGURE 29 – Evolutions de l'erreur RSME en fonction de k

Le modèle 3 : Dans le troisième modèle avant d'effectuer une recherche du k optimal comme dans le deuxième modèle, on commence d'abord par centre et réduire les données numérique, sur la figure 30 ci-dessous on constate que le k optimal est égal à 12

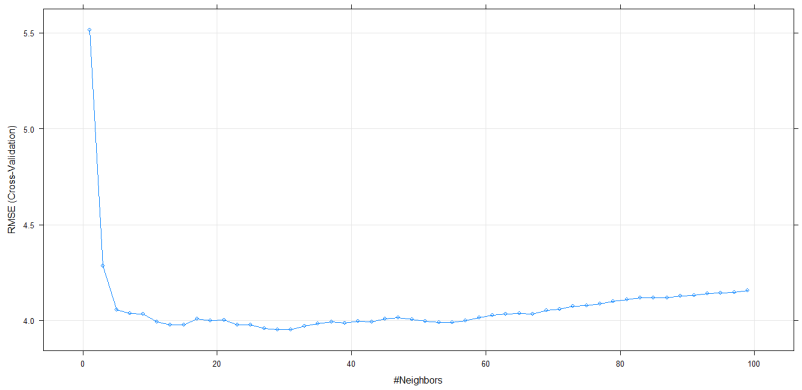


FIGURE 30 – Evolutions de l'erreur RSME en fonction de k avec des donnée centrées et réduites

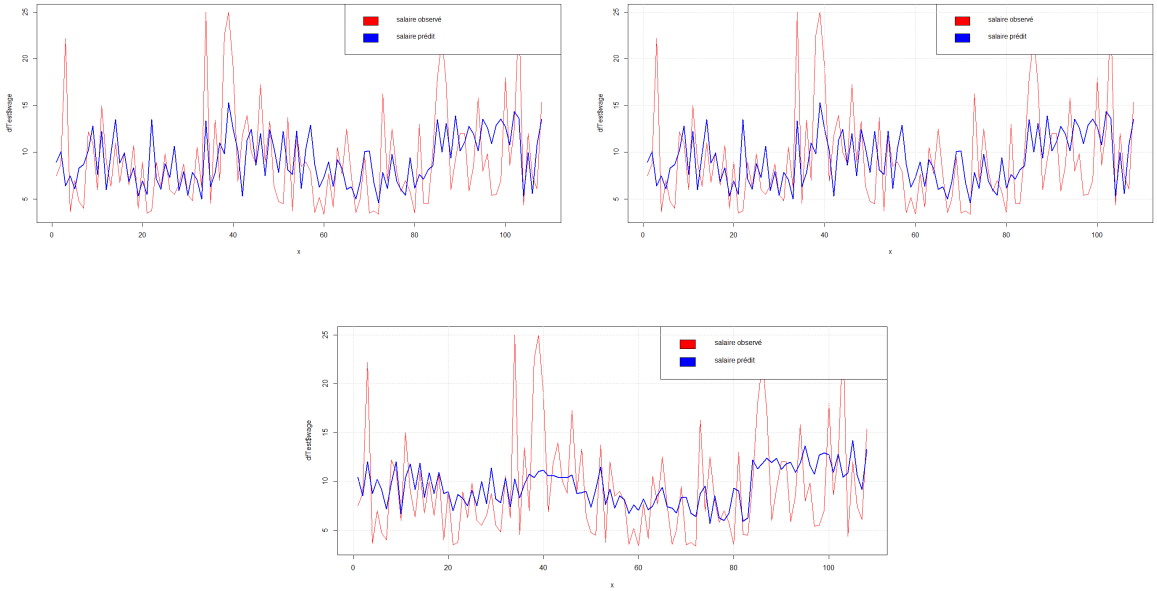


FIGURE 31 – Salaire observé vs. salaire prédit des trois modèles KNN

La figure 31 ci-dessus, représente le salaire observés (en rouge) et le salaire prédit (en bleu) sur l'échantillon "test" des trois modèles KNN.

	knn1	knn2	knn3
MSE	20.86	21.36	20.36
MAE	3.47	3.46	3.39
RMSE	4.57	4.62	4.51
R2	0.22	0.20	0.25

TABLE 5 – Résultats des modèles KNN

La table ci-dessus résume les différente erreurs et R^2, on voit clairement que le le modèle le plus performant est le modèle 3, où on a centré et réduit les données numériques.

4.5 Comparaison des modèles des différentes méthodes :

On passe maintenant à la comparaison des modèles sélectionnés de chaque méthode de ML, à savoir le modèle reg3 de la régression multiple, le modèle Tree2 des arbres de décision et knn3 des KNN, ci-dessus la table récapitulatif des erreurs ($MSE, MAE, RMSE, R^2$) et le coefficient de détermination R^2 pour les prédictions effectuées sur l'ensemble de validation, on remarque la performance des trois modèle a considérablement diminué en particulier le modèle knn ($R^2 = 0.07$), les deux modèles de régression et arbres de décision sont très proche en terme de performance,leur coefficient de détermination est respectivement 0.17 et 0.18 (voir la table 6). Donc au final on peut dire que pour ces données là en particulier l'algorithme des KNN n'est pas du tout performant, et pour la regression multiple et arbres de décision bien qu'ils soient très proches en terme de performance,on opte pour les arbres de décision car en plus d'avoir un coefficient de déermination R^2 légèrement plus élévé, les arbres de décision effectue automatiquement une selection des variables explicatives, en effet pour le modèle "Tree 2" les seules variables qui interviennent dans le modèle sont : éducation, occupation, genre, experience et âge.

	reg3	Tree2	knn3
MSE	32.58	32.26	37.35
MAE	3.81	3.95	3.91
RMSE	5.75	5.67	6.11
R2	0.17	0.18	0.07

TABLE 6 – : Résultats des modèles selectionnés

Conclusion générale

Le machine learning est une discipline qui ne cesse d'évoluer et ce de manière frénétique. Pouvant s'appliquer dans tout les domaines, ses algorithmes sont basés sur les données et des modèles statistiques pour analyser des ensembles de données, puis tirer des conclusions à partir de modèles identifiés et faire des prédictions en fonction de ceux-ci. Notre objectif à travers ce memoire est de fournir une présentation des algorithmes de ML et d'effectuer une étude sur des données réelles afin d'évaluer et comparer quelques uns de ces algorithmes, ainsi le première chapitre contient une présentation globale du machine learning commençant par un bref historique, les différents types d'apprentissage automatiques, puis le machine learning dans l'entreprise.

Le second chapitre présente la théorie de l'apprentissage statistique commençant par la définition des données et le modèle statistique, puis les différentes types de mesures de qualité, et présenter les modèles proposés pour l'estimation de risques empirique et dernièrement une présentation des critères de performance.

Le troisième chapitre comporte une présentation des algorithmes d'apprentissage supervisé : la régression multiples, les K plus proche voisins, les arbres de décision et les réseaux de neurones.

Le quatrièmes chapitre est consacré à notre étude pratique qui consiste en l'implémentation et comparaison de trois types d'algorithmes : la regression multiple, les arbres de décision et les K plus proche voisins, sur un enssemble de données réelles afin de démontrer le fonctionnement de ces modèles et déterminer quel est le modèle le plus performant .

Bibliographie

1. Hastie,., Tibshirani, R., andFriedman, j. (2011).The Éléments of statistical learning
2. Mitchell, T. (1997).machine learning. MCG raw hill.
3. A. Gérons, hands-on machine learning with scikit-Lean, keras, and technique to build intelligent systèmes. O'Relly Média 2019.
4. T. M. Mitchell, "Machine learning, volume 1of1 1997"
5. I.vasilev,D,staters G spacagna, P. Roelants, and v. Zacca, python Deep learning :Explorning Deep learning techniques and neural network architectures with pytorch, keras, and tensor Flow. Packt publishing Ltd, 2019.
6. G. E. Hinton, s. Osindero, and, y-w. Teh, "À fast learning algorithmes for Deep belif nets", Neural computation, vol, 18,no.7,PP.1527-1554,2006.
7. KOZA,J.R.Bennett,F.H.,Andre,D.,& keane, M.A.(1996).Automated design of both thé topology and sizing of analog Electrical circuit using programma ING Dans artificical intelligence in Design
8. L.P.chen,"Mehryar mohri, afshin rostami zadeh, and ameet talwalkar : Foundation of machine learning" 2019.
9. livre, G. Dreyfus, J.M. Martine, M. Samuel ides, M.B.Godon,F.Badran,S.Thiria.
10. https ://www.lemagit.fr
11. https ://www.linkfluence.com
12. https ://www.OWOX.com
13. https ://analyticsinsights.io
14. https ://geekflare.com/fr/choosing-ml-algorithms/
15. livre :P. Barbillons, G. Celeux, A. Grimaud, Y. Le febvre, and E. De Rocquigny. Nom linear méthodes for inverse statistical problèmes. Computation al statistics.S.Data analyses 55(1),132-142,2015.
16. https ://www.votre-it.facile.fr
17. https ://towardsdatascience.com/how-to-choose-the-right-machine-learning-algorithm-for-your-application-1e36c32400b9
18. https ://www.sirris.be/how-choose-right-algorithm-right-task
19. https ://blog.hexstream.com/how-to-choose-the-right-machine-learning-algorithm

20. https ://m.nearbyme.io/search/ ?search(_)term=Choosing

21. https ://moncoachdata.com/blog/algorithme-des-k-plus-proches-voisins

22. https ://cours.etsmtl.ca/gti770/private/notes/PDF/LOG770-ArbresDecision(_)3pp.pdf

23. https ://mrmint.fr/introduction-k-nearest-neighbors distance KNN

24. Bishop, C. (1995). Neural Networks for Pattern Recognition. Oxford : University Press.

25. Carling, A. (1992). Introducing Neural Networks. Wilmslow, UK : Sigma Press.

26. Fausett, L. (1994). Fundamentals of Neural Networks. New York : Prentice Hall.

27. Haykin, S. (1994). Neural Networks : A Comprehensive Foundation. New York : Macmillan Publishing.

28. Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. Biological Cybernetics, 43 :59-69.

29. Patterson, D. (1996). Artificial Neural Networks. Singapore : Prentice Hall.

30. Ripley, B.D. (1996). Pattern Recognition and Neural Networks. Cambridge University Press.

31. Rumelhart, D.E., and J.L. McClelland (1986), Parallel Distributed Processing, Volume 1. The MIT Press. Foundations.

32. Tryon, R. C. (1939). Cluster analysis. New York : McGraw-H