UNIVERSITE DE KINSHASA

FACULTE DES SCIENCES

DEPARTEMENT DE MATHEMATIQUE ET INFORMATIQUE

APPRENTISSAGE ARTIFICIEL BASE SUR LA COMBINAISON DES CLASSIFIEURS AUTOMATIQUE PAR VOTE MAJORITAIRE POUR LA PREDICTION DE CLIENTS CREDIBLES

BAKETI REBECCA Raissa

Mémoire présenté et defendu en vue de l'obtention du titre de licenciée en Science

Option : Mathématique et informatique

Directeur : KAFUNDA KATALAY Pierre
Professeur

2020-2021

TABLE DE MATIERE

EPIGRAPHE

« Il n'y a pas de rose sans épine c'est-à-dire le bonheur n'est jamais parfait »

JEAN FRANCOIS RABELAIS

DÉDICACE

À mon très cher père

BAKETI MIANGO ALPHONSE

et

à ma précieuse mère

LUBAMBA NGALULA CHANTAL

pour leur amour et volonté de prendre quotidiennement leur responsabilité au sujet de mes études.

REMERCIEMENTS

À Dieu tout puissant, créateur du ciel et de la terre, le maître de l'intelligence et de la sagesse, le réalisateur de l'impossible et du possible pour m'avoir accordé le souffle de vie et la force de concrétiser ce travail.

Aux autorités de la faculté des Sciences et du Département de Mathématiques et Informatique. Ainsi, pour tous les membres, de notre alma mater, qui ont aidé à la réalisation de ce travail ; de près ou de loin je vous dis merci.

Notre profonde reconnaissance se destine au professeur KAFUNDA KATALAY Pierre qui, malgré ses multiples occupations, a bien voulu diriger ce travail.

Nous exprimons notre gratitude à toute l'équipe du laboratoire du professeur KAFUNDA notamment le professeur Félicien MASAKUNA, l'assistant Armel MBENZA et tous les autres pour leur disponibilité dans l'encadrement de ce travail.

.

À mes frères et sœurs de la famille BAKETI, j'ai cité **Alphonsine BAKETI, Sophie BAKETI, Honore BAKETI, Amour BAKETI, Céline BAKETI, Jephté PANGO, Méchack KANGA, Christian NTAMA** et à mes nièces et neveux vous faites la fierté de notre famille.

Nul ne peut ignorer ses origines car si la destination n'est pas connue, l'origine le sera toujours. Je suis particulièrement reconnaissante à mon meilleur père **BAKETI MIANGO Alphonse** et à ma chérie mère **LUBAMBA NGALULA Chantal**

A mes amis et connaissances **BALINGA MANSANGA Esther, KABUNDA TUMBA Médie, MANZA MANZA Esther, N'SENGA MUILU Divine, NZEBULA Deborah, MANANGA Gradi, MUTEBA Gustave, NGONGO Deo, KASAYA Gloria** et à tous ceux dont je n'ai pas pu citer les noms, la liste est longue je vous remercie.

BAKETI REBECCA Raïssa.

LISTE DES FIGURES

LISTE DES ABREVIATIONS

N°	Abréviation	Description
1	BD	Base de données
2	URL	une chaîne de caractères uniforme qui permet d'identifier une ressource du World Wide Web
3	VO	Vision par ordinateur
4	RF	Reconnaissance de formes
5	REM	Reconnaissance de l'écriture manuscrite
6	RV	Reconnaissance vocale
7	TAL	Traitement automatique de la langue
8	BI	Bio-informatique
9	IA	Intélligence Artificielle
10	RNA	Réseau de neurone artificiel
11	SVM	Support Vector Machine

INTRODUCTION GENERALE

Contexte

Le mot crédit provient du latin « creditum » « de credere » qui signifie croire ou avoir confiance. (Charles Petit-Dutaillis, 2012) « faire crédit, c'est faire confiance, mais c'est aussi donner librement la disposition affective et immédiate d'un bien réel ou d'un pouvoir d'achat, contre la promesse que le même bien ou l'équivalent vous sera restitué dans un certain délai, le plus souvent avec la rémunération du service rendu et du danger couru. Danger de perte partielle ou totale que comporte la nature même de ce service »

Depuis quelques années, le domaine de conservation des clients devient une priorité dans le secteur bancaire sous l'effet de la multiplication, de la sophistication et de la prégnance des dispositifs de gestion appliqués à la « relation client ». Plusieurs méthodes ont été développées pour tenter de satisfaire les besoins des systèmes existants dans diverses applications telles que la prédiction de la crédibilité d'un client. Sachant que les banques évoluent dans des environnements ultra concurrentiels, l'entretien d'un client crédible s'avère être un travail de longue haleine.

La combinaison des classifieurs automatiques, une approche de classification les plus utilisées pour la reconnaissance, le classement automatique d'un nouvel individu où plusieurs classifieurs sont confrontés avec le principe de trouver une fonction qui associe un jeu de données de départ à un ensemble de classes d'arrivée. Les performances des classifieurs dépendent de la variété des entrées qui leurs sont présentées durant la phase d'apprentissage ainsi que de la pertinence et la qualité de ces entrées. Il existe une grande variété de caractéristiques pouvant être extraites à partir d'une donnée d'entrée. Comme ces caractéristiques sont très importantes pour différencier les données à analyser, il serait intéressant de profiter de la variété des informations qu'elles apportent et ceci pour éviter les facteurs subjectifs des différents classifieurs qui peuvent présenter, quelques erreurs causant ainsi des faux négatifs. C'est donc un intérêt capital pour le banquier qui doit établir un diagnostic fiable et valide dans le but de détecter les clients crédibles ou non crédibles.

Problématique

Une approche prometteuse est d'aider le banquier dans sa recherche des clients crédible sur son fichier de la clientèle. Étant donné que la banque évolue dans un environnement ultra compétitif, elle doit servir ses clients à temps réels et garantir leur confiance. Nous notons que l'entretien d'un client dans une institution bancaire a un

coût. ***Ecobank***, l'institution bancaire où nous avons mené notre recherche scientifique avoisine aujourd'hui les 29 millions de clients à travers le monde. D'où même si les dirigeants de ladite institution bancaire accordait ne serait-ce qu'un dollars américaine (1 USD) le mois, pour l'entretien d'un client, il faudrait, pour la banque, mobiliser autour de vingt-neuf millions de dollars américaine (29 millions USD) pour l'entretien l'ensemble de sa clientèle. Alors que toute cette masse volumique des clients ne vaut pas la peine d'être entretenue de la même manière d'autant plus qu'il y a dans ce fichier de la clientèle, les clients qui sont crédibles et ceux qui ne le sont pas.

Dans un tout autre aspect, la rentabilité d'un établissement de crédit représente son aptitude à dégager de son exploitation des gains suffisants, après déduction des coûts nécessaires à cette exploitation, pour poursuivre durablement son activité. Le crédit est obligatoirement lié à une notion de profitabilité et de risque. Ces deux éléments restent indissociables dans le cadre de l'activité bancaire. La recherche d'une plus-value toujours plus importante sur les prêts bancaires n'est pas toujours un choix judicieux car cela implique de lourdes précautions. En fonction de la politique de chaque établissement de crédit, un choix se porte entre une préférence de qualité ou de volume pour l'octroi de crédit. Cette décision stratégique engendre des conséquences car elle définit la ligne directrice de la banque et sa politique de prêt. Il devient nécessaire de gérer de façon optimale le couple risque, rentabilité pour que la banque puisse réaliser un maximum de plus-value avec un minimum de pertes 4 .

Au jour d'aujourd'hui, il est possible pour un programme d'intelligence artificiel, d'assister les experts dans le processus de la prise de décision, dans des environnements complexes et évolutifs, à l'occurrence, l'analyse de marché financier ainsi que la prédiction de la crédibilité des clients.

Se référant à ce qui précède, nous avons formulé quelques questions qui conduisent la démarche de notre recherche scientifique, notamment sur la thématique de ce travail. A savoir :

- Est-il important de prédire le comportement d'un nouveau client ?
- Est-il pertinent de connaître le statut d'un client ?
- Comment une banque peut avoir assez confiance en un tiers pour lui prêter des fonds ?
- Est-il possible de reduire le risque de remboursement de crédit ?

Hypothèse

Un système automatique pourrait contribuer à la détection des erreurs d'analyse pour une meilleure fiabilité des résultats. Sur ce, **l'apprentissage artificiel basé sur la combinaison des classifieurs automatiques par vote majoritaire pour la prédiction de client crédible** saveur être une solution adéquate en ce qui concerne l'entretien de client dans une institution bancaire.

En effet, la gestion du risque de crédit est en constante amélioration, compte tenu de la complexité des menaces de l'activité de prêt. Les établissements de crédit ont un intérêt fondamental à maximiser la gestion des risques pour limiter les pertes monétaires et temporelles par l'utilisation d'une intelligence artificielle capable de classifier un client selon que ce dernier a déposé un bon ou mauvais dossier de prêt.

Cette classification automatique qui dérive des combinaisons de classifieurs, conduit à des données de grandes dimensions où la possibilité d'avoir en sortie des résultats stables et fiables est très élevée. Une analyse discriminante linéaire (LDA) est réalisée pour extraire les caractéristiques discriminantes et pertinentes afin de réduire le temps d'exécution, les redondances et le bruit. 8 9.

Contribution

Pour cette raison, nous proposons une approche d'apprentissage artificiel dont le but est la prédiction des clients crédibles. Cet apprentissage est basé sur la combinaison des décisions des classifieurs automatiques entraînés.

Techniques et méthodes

Afin d'atteindre les objectifs assignés, nous avons utilisé les méthodes et techniques suivantes :

Méthodes de recherche

- **la méthode analytique :** elle nous a permis de faire une analyse approfondie de données afin d'en tirer les conclusions qui s'imposent ;
- **la méthode historique :** elle nous a aidé à cerner l'évolution des technologies utilisées dans ce travail;

Techniques

- **La technique documentaire :** elle nous permettra de consulter les ouvrages et autres documents tels que journal officiel, notes de cours, ouvrages afin de disposer des renseignements utiles ;

L'interview libre : nous permettra sous forme d'un jeu de questions réponses à sens unique, d'obtenir des informations auprès de professionnelle de la comptabilité et de la fiscalité.

Plan de travail

Notre travail est organisé en quatre chapitres hormis l'introduction et la conclusion. Il est présenté comme suite :

- Le premier chapitre est consacré aux concepts théoriques et notions de base sur ***l'Apprentissage Artificiel***. Nous présentons ici quelques principales méthodes de classification supervisée et non supervisée : les machines à vecteurs de support (SVM), les réseaux de neurones, les arbres de décision et le K plus proches voisins …
- Dans le deuxième chapitre, nous parlons des techniques, principes et besoins ***des combinaisons des Classifieurs*** automatiques.
- Le troisième chapitre est divisé en deux parties. La première partie présente la banque *Ecobank, son statut, son fonctionnement et sa structuration*. La deuxième partie introduit les notions de crédit bancaire.
- Le quatrième chapitre est consacré à ***l'Implémentation et interprétation*** des résultats. Il explique en détaille les différentes modélisations de fusions réalisées ainsi une discussion des résultats obtenus.
- La conclusion générale résume l'ensemble de notre travail, et présente quelques perspectives.

CHAPITRE I. APPRENTISSAGE ARTIFICIEL

I.0. Introduction

Il est aujourd'hui possible pour un programme d'intelligence artificielle de reconnaître des commandes vocales, d'analyser automatiquement des photos satellites, d'assister des experts pour prendre des décisions dans des environnements complexes et évolutifs (analyse de marchés financiers, diagnostics médicaux...), de fouiller d'immenses bases de données hétérogènes, telles les innombrables pages du Web[1].

L'apprentissage artificiel s'intéresse à l'écriture de programmes d'ordinateur capables de s'améliorer automatiquement au fil du temps, soit sur la base de leur propre expérience, soit à partir de données antérieures fournies par d'autres programmes. Dans le domaine scientifique relativement jeune de l'informatique, l'apprentissage artificiel joue un rôle de plus en plus essentiel.

L'apprentissage artificiel, dans sa première phase, estime un modèle à partir de données, appelées observations, qui sont disponibles et en nombre fini, lors de la phase de conception du système. Cette phase dite « d'apprentissage » ou « d'entraînement » est généralement réalisée préalablement à l'utilisation pratique du modèle.

L'apprentissage artificiel est une science qui consiste à développer des algorithmes d'apprentissage qui apprennent à résoudre une tâche. L'apprentissage artificiel fait référence à la capacité d'un système à acquérir et intégrer de façon autonome des connaissances. Cette notion englobe toute méthode permettant de construire un modèle de la réalité à partir de données, soit en améliorant un modèle partiel ou moins général, soit en créant complètement le modèle.

Le *machine learning* est un domaine qui a pris de l'ampleur au temps actuel. Issu de nombreuses disciplines comme les statistiques, l'optimisation, l'algorithmique ou le traitement du signal[2]

1. « apprentissage automatique » [archive], *Le Grand Dictionnaire terminologique*, Office québécois de la langue française (consulté le 28 janvier 2020).

2. (en) Alan Turing, « *On computable numbers, with an application to the entscheidungsproblem », Proceedings of the London Mathematical Society*, s2-42, 12 novembre 1936, p. 230-265 (DOI 10.1112/plms/s2-42.1.230, (consulté le 23 septembre 2022)

Dans les années à venir, le machine learning nous permettra vraisemblablement d'améliorer la sécurité routière (y compris 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. Le but de ce chapitre est d'établir plus clairement ce qui relève ou non du machine learning, ainsi que des branches de ce domaine dont cet ouvrage traitera.

I.1. Définitions

L'apprentissage artificiel ou machine learning est un champ d'étude de l'intelligence artificielle qui se fonde sur des approches mathématiques pour donner aux ordinateurs la capacité d'apprendre à partir de données, c'est-à-dire d'améliorer leurs performances à résoudre des tâches sans être explicitement programmés pour chacune.

L'apprentissage artificiel fait référence au développement, à l'analyse et à l'implémentation de méthodes qui permettent à une machine (au sens large) d'évoluer grâce à un processus d'apprentissage, et ainsi de remplir des tâches qu'il est difficile ou impossible de remplir par des moyens algorithmiques plus classiques[3].

L'apprentissage artificiel est une tentative de comprendre et reproduire cette faculté d'apprentissage dans des systèmes artificiels. Il s'agit, très schématiquement, de concevoir des algorithmes capables, à partir d'un nombre important d'exemples (les données correspondant à l'expérience passée), d'en assimiler la nature afin de pouvoir appliquer ce qu'ils ont ainsi appris aux cas futurs[4].

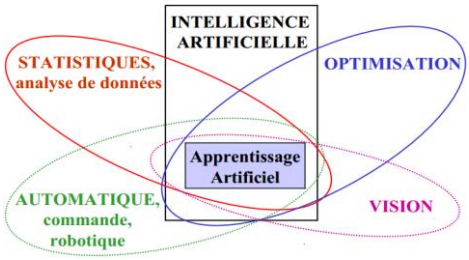


Figure I.1. Nature multidisciplinaire de Machine learning(MINES ParisTech mai 2011)

De toutes ces définitions nous pouvons dire que l'apprentissage artificiel est la généralisation de tout modèle permet de construire un programme susceptible d'évoluer dans le temps pour la résolution des problèmes non classiques.

---

[3] sage-automatique.htmlhttps://www.techno-science.net/glossaire-definition/Apprentis (consulté en février 2022).

[4] (en-US) John Markoff, *« On 'Jeopardy!' Watson Win Is All but Trivial »*, *The New York Times*, 16 février 2011 (ISSN 0362-4331, lire en ligne [archive], consulté le 11 mai 2018).

En bref, comme nous venons de nous baser sur les différentes définitions de l'apprentissage artificiel, la suite nous montre que l'apprentissage artificiel s'applique plusieurs problèmes et dans plusieurs domaines.

I.2. Objectifs

L'apprentissage artificiel a pour objectif extraction et l'exploration automatique de l'information présente dans un jeu de données. L'apprentissage artificiel vise à bien comprendre des phénomènes naturels en vue d'en construire des modèles de simulation. Il s'ensuit aussi que le but de cette discipline est d'extraire de la connaissance de manière automatique à partir de données brutes. Cette connaissance peut alors être exploitée pour prendre des décisions. On parle parfois de stratégie pilotée par les données pour une entreprise[5].

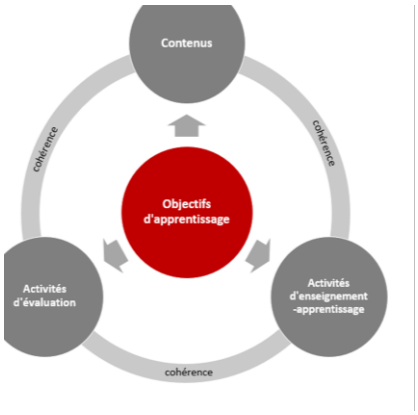


Figure.I.2. cohérence et objectif d'apprentissage

La formulation des objectifs d'apprentissage permet de déterminer les connaissances à acquérir et les compétences à développer par les étudiantes et les étudiants au terme d'une activité d'apprentissage. Cet exercice intervient dès le début du processus de planification d'un cours. Par souci de cohérence, le choix des méthodes d'évaluations, des activités d'enseignement-apprentissage et l'organisation du contenu du cours découlent des objectifs d'apprentissage.

[5] Y. LeCun, B. Boser, J. S. Denker et D. Henderson, « Backpropagation Applied to Handwritten Zip Code Recognition », *Neural Computation*, vol. 1, n° 4, 1er décembre 1989, p. 541–551 (ISSN 0899-7667)

I.3. Référentiel

Avant de nous plonger dans les détails de cette discipline, nous passons tout d'abords à l'explication des termes clés qui seront exploités dans ce travail.

I.3.1. Individu (Instance ou Exemple)

Un individu est le constituant d'un ensemble, représenté par ses caractéristiques étudiées. Dans la suite, on appellera individu chaque $x_i \in X$ avec :

X : Ensemble d'individus.

Pour une étude sur la prédiction des clients crédibles, les individus sont les clients. Un individu est le constituant d'un ensemble, représenté par ses caractéristiques étudiées.

I.3.2. Variable

Une variable est une fonction qui a pour but d'affecter à chaque individu une valeur donnée sur le domaine d'observation, soit le client est crédible ou non crédible.

$$y_h : X \rightarrow O_h$$
$$x_i : \rightarrow y_h(x_i)$$

Avec :

X : Ensemble d'individus ou d'instances

O_h: Domaine d'observation. Dans notre cas, ce domaine n'est constitué que des valeurs uniques classiques.

Ce qui implique que les variables qui seront utilisées dans la suite soient à description quantitative ou qualitative.

Une variable qualitative est une fonction pour laquelle la valeur mesurée sur chaque individu ne représente pas une quantité. Les différentes valeurs que peut prendre cette variable sont appelées les catégories, modalités ou niveaux.

Une variable est quantitative si elle reflète une notion de grandeur, c'est-à-dire si les valeurs qu'elle peut prendre sont des nombres pouvant être classés en utilisant la relation inférieure ou égale à.

Par exemple :

- Long URL : quantitatif,
- Protocole : qualitatif,

- Mode : qualitatif

I.3.3. Classe

Une classe est un sous ensemble de X tels que ses constituants sont jugés comme homogène.

Soit X l'ensemble d'individus, on appellera une classe C_k avec $k \in \{1, \dots, K\}$ et K le nombre total de classes, un élément du recouvrement C de X.
Nous ne considérons que des recouvrements dont les éléments sont deux à deux disjoints, donc une partition. C'est-à-dire :

- $C_k \neq \emptyset \; \forall k$
- $C_k \cap C_l = \emptyset \;\; \forall k \neq l$
- $\cup_{k=1}^{K} C_k = X$

Par exemple, la classe d'observation des clients crédible et nom crédible.

I.3.4. Classifieur ou Classificateur

Un classifieur est un algorithme qui, après avoir été construit, est en mesure de réaliser les tâches d'affectation. C'est-à-dire prédire la classe d'affectation pour un nouvel individu qui se présente. Selon notre cas, nous devrons mettre au point des classifieurs qui seront capables de dire si un client est crédible ou non crédible[6].

I.3.5. Base d'apprentissage

Une base d'apprentissage est un ensemble d'exemples déjà traité tel que pour chaque entrée, sa sortie est connue. C'est-à-dire un ensemble.

$$\beta = \{(x_i, y_i)_{i \in \{1,\dots,N\}} | \; x_i \in X, y_i \in Y\}$$

Où Y est l'ensemble de valeurs de sortie.

---

[6] (en) Guo, S., « *An introduction to Surrogate modeling, Part I: fundamentals* » , sur *Towards Data Science*, p12, 2020

I.4. Les Tâches de l'apprentissage artificiel

L'une des tâches de l'apprentissage artificiel est la classification (prédiction), elle consiste à construire un classifieur capable de classer une variable donnée. Pour notre exemple la valeur de la variable "Statut" en fonction des nouvelles caractéristiques (nouvel individu) présentées. Cette valeur correspond ainsi à la classe d'affectation de ce nouvel individu.

Disposant des données issues d'une expérience comme notre tableau, il faut entraîner le classificateur de manière à ce qu'il soit en mesure de prédire déjà la valeur de la variable "Statut" qui est connue d'avance. Cette phase s'appelle, phase d'apprentissage et les données utilisées durant cette phase constituent la base d'apprentissage. Ainsi, on parle alors d'un apprentissage supervisé.

Il existe aussi d'autres techniques d'apprentissage dit non supervisé dont l'un des buts est de regrouper les individus qui sont similaires dans des classes. On parle dans ce cas de la classification automatique, et tant d'autres.

I.4.1. Modèle

Un modèle est une représentation détaillée et simplifiée de la réalité, dans notre travail le modèle sera un algorithme ou une fonction mathématique qui prédit au mieux les résultats.

I.4.2. Méthode

La méthode est un ensemble de démarches que suit l'esprit pour démontrer ou découvrir une vérité. De ce fait, nous pouvons dire que la méthode est bien établie en avance et sur cette méthode chacun peut développer sa technique dans le but d'atteindre rapidement et facilement le résultat.

I.5. Principes d'apprentissage Artificiel

L'apprentissage artificiel utilise les algorithmes dans une certaine mesure ou à un système piloté par ordinateurs (un robot éventuellement), ou assisté par ordinateur d'adapter ses analyses, et comportements en réponse en se fondant sur l'analyse de données empiriques provenant d'une base de donnée ou de capteur[7].

[7] Op cit, p 34

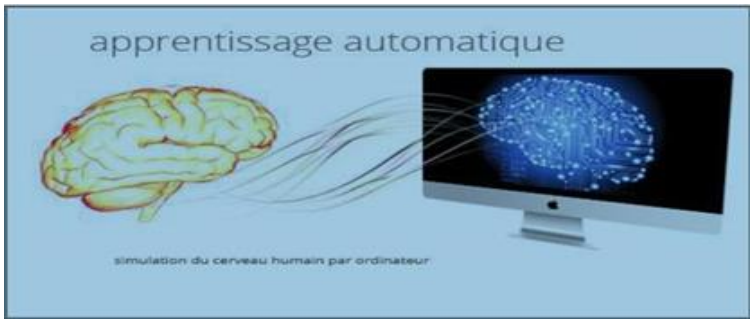


Figure.I.3.système piloté par Ordinateurs

Ces programmes, selon leur degré de perfectionnement intègrent des capacités en probabilités et statistiques, traitement de données et éventuellement d'analyse de données de capteurs, de reconnaissance (reconnaissance vocale, reconnaissance de forme, d'écriture, etc.).

La difficulté réside dans le fait que l'ensemble de tous les comportements possibles compte tenu de toutes les entrées possibles devient rapidement trop complexes à décrire dans les langages de programmation disponibles, de sorte qu'on confie en quelque sorte à des programmes le soin d'apprendre de manière à auto-améliorer le système d'analyse ou de réponse (commande adaptative...), ce qui est une des formes que peut prendre l'intelligence artificielle.

I.6. caractéristiques d’apprentissage artificiel

Parmi les principales caractéristiques et facultés adoptées par les modèles d’apprentissage, nous citons: l’adaptation, la généralisation, l’intelligibilité, reconnaissance, l'amélioration.

I.6.1. Adaptation

Elle peut être vue comme étant la disposition du modèle (algorithme ou système) à corriger son comportement ou à remanier sa réponse (ex., prédiction) par rapport à de nouvelles situations. Pour les tâches de perception, en vision artificielle, on accumule les bonnes et mauvaises expériences, et à partir d'elles, on peut faire évoluer les règles pour mieux effectuer la tâche, c'est le phénomène d'adaptation ou d'amélioration

I.6.2. Généralisation

D'une autre manière, l'apprentissage est typiquement caractérisé par une généralisation rationnelle des règles ; c'est-à-dire si d'une expérience accumulée sur un certain nombre d'exemples, on tire des règles de comportement, il faudrait que celles-ci soient également applicables à des situations encore non rencontrées.

I.6.3. Intelligibilité

C'est améliorer la compréhension des résultats d'apprentissage, afin que le modèle puisse fournir une connaissance claire et compréhensible. Au sens interprétable (en anglais, on parle de comprehensibility ou understandability). Par exemple, quand un expert extrait de la connaissance des bases de données (BDs), il apprend une manière de les résumer ou de les formuler (expliquer, expliciter de manière simple et précise).

D'un point de vue fouille de données, ça revient purement et simplement à contrôler l'intelligibilité (clarté) d'un modèle obtenu. Actuellement, la mesure d'intelligibilité se réduit à vérifier que la connaissance produite est intelligible et que les résultats sont exprimés dans le langage de l'utilisateur et la taille des modèles n'est pas excessive.

I.6.4. Reconnaissance

Avec la reconnaissance de la parole, par exemple, le programme d'apprentissage n'aura pas besoin d'apprendre tous les sons de ladite parole. Il va extraire une règle de classification qui lui permettra de traiter au mieux les sons qu'il aura à décoder.

I.6.5. l'amélioration

Les sciences cognitives définissent l'apprentissage comme étant une capacité améliorant les performances au fur et à mesure de l'exercice d'une activité. C'est le cas d'un joueur du scrabble au fil des parties, où l'assimilation de l'expérience et la puissance du raisonnement se combinent dans sa progression.

I.7. Prétraitement Des Données

Il est souvent important de faire un prétraitement (un nettoyage) des données avant de les utiliser dans l'algorithme d'apprentissage. Les différents problèmes à considérer pour le prétraitement sont les suivants:

- ✔ La sélection d'attributs : consiste à éliminer les attributs les moins pertinents pour l'apprentissage. Le but est de diminuer la dimensionnalité du problème qui est à la fois une source d'imprécision et une difficulté calculatoire. Si on possède une description des données par un ensemble d'attributs, le problème est de chercher un sous-ensemble d'attributs qui préserve au mieux les informations nécessaires à l'algorithme d'apprentissage.

- ✔ Le choix des attributs de description : Généralement le choix des attributs vise à diminuer le nombre du descripteur afin de faciliter l'apprentissage sans détruire la qualité du résultat. On distingue deux grandes approches.

- ✔ L'extraction d'attributs : réduit la dimensionnalité de l'espace d'entrée en appliquant des transformations, linéaires ou non, aux attributs initiaux.

I.8. Espace des données d'apprentissage

Le problème de la classification est l'apprentissage d'une fonction dite de prédiction, de décision etc. au travers des données. Pour espérer obtenir un classifieur adapté à la tâche considérée, quelques points sont à survoler.

Ces points se résument autour de deux problèmes essentiels :

- ✔ Celui de la représentation adéquate des données **;**
- ✔ Et celui de la représentation des hypothèses faites par le programme d'apprentissage.

I.8.1. DONNÉES

Une donnée peut être définie de diverses manières, en Machine Learning, Les données sont appelées échantillons. Les échantillons sont souvent notés sous forme de vecteur :

Où est le nombre de coordonnées aussi appelé nombre d'attributs, dimensions ou caractéristiques, variable, champs.

Une donnée est caractérisée par un certain nombre d'attributs. Un attribut peut être de nature qualitative ou quantitative en raison des valeurs qu'il peut prendre.

- **Il est qualitatif**

Si les valeurs qu'il peut prendre ne sont pas mesurables; sa valeur est d'un type défini par extension (une catégorie sociale, un sexe, une nationalité, un rang social, la couleur de peau, état civil,…).

- **Il est quantitatif**

Si les valeurs qu'il peut prendre sont mesurables ou repérables, discrètes ou continues (la surface d'un espace, l'âge d'une personne, le nombre d'enfants, la taille, le poids,…). Mesurable suppose que l'on peut y appliquer diverses opérations mathématiques. En outre de ces deux natures, un attribut peut également être une structure d'enregistrement, le cas d'une date, composé lui-même de sous-attributs.

I.8.2. Données bruitées

Les données dont certains attributs ont des valeurs inconnues ou non valides sont dites bruitées (elles sont appelées bruits). Ces données ne doivent pas être sujettes d'une élimination à l'aveugle, car dans la plupart des cas, une masse considérable de données que l'on aura à traiter, se trouvera dans ce cas. Pour des fins de traitement, des techniques de préparation des données à l'apprentissage sont présentées ci-dessous.

I.8.3. Sur apprentissage

Le sur-apprentissage (Overfitting en anglais) désigne le fait que le modèle prédictif produit par l'algorithme de Machine Learning s'adapte à tous les aspects et détails qui caractérisent les données. Quand un tel événement se produit, le modèle pourra donner de très bonnes prédictions sur les données mais il prédira mal sur les données auxquelles il n'a pas assez de connaissances lors de la phase d'apprentissage.

I.8.4. Sous apprentissage

Le sous-apprentissage (Underfitting en anglais) sous-entend que le modèle prédictif généré lors de la phase d'apprentissage s'adapte mal à tous les aspects et détails qui caractérisent les données, le coût d'erreur en phase d'apprentissage reste grand. Le modèle prédictif ne peut aussi prédire sur les données qu'il n'a pas encore vu.

I.8.5. Modèle généraliste de base d'apprentissage

Un bon modèle d'apprentissage en Machine Learning est un modèle qui généralise. La généralisation, c'est la capacité d'un modèle à faire des prédictions non seulement sur les données que vous avez utilisées pour le construire, mais surtout sur des nouvelles données.

I.9. Types d'apprentissages

Il y a plusieurs types d'apprentissage on peut le catégoriser selon le mode d'apprentissage qu'ils emploient:

I.9.1. Apprentissage supervisé

L'apprentissage supervisé est une technique artificielle où l'on cherche à produire automatiquement des règles à partir d'une base de données d'apprentissage contenant des « exemples » (en générale des cas déjà traités et validés) où l'on agit sur les données tout en ayant les hypothèses ou les informations supplémentaires fournies sur les données par le superviseur.

Si les classes sont prédéterminées et les exemples connus, le système apprend à classer selon un modèle de classement ; on parle alors d'apprentissage supervisé (ou d'analyse discriminante). Un expert (ou oracle) doit préalablement correctement étiqueter des exemples. L'apprenant peut alors trouver ou approximer la fonction qui permet d'affecter la bonne « étiquette » à ces exemples. Parfois il est préférable d'associer une donnée non pas à une classe unique, mais une probabilité d'appartenance à chacune des classes prédéterminées.

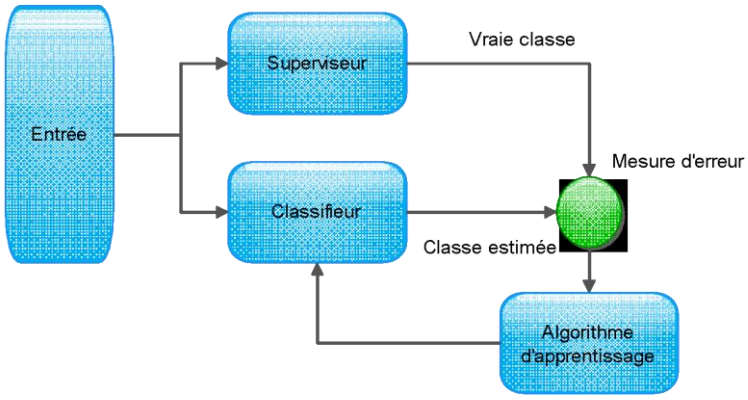


Figure.I.4. Schéma de l'apprentissage supervisé [Prof Kafunda]

L'apprentissage supervisé consiste à créer pour des modèles de prédiction des machines intelligentes capables d'affecter un individu à sa classe.

Ici les données sont étiquetées c'est-à-dire nous agissons sur les données tout en ayant des hypothèses ou des informations supplémentaires fournies par l'expert du domaine, les classes sont prédéfinies nous cherchons à affecter chaque individu dans sa classe, en d'autres termes nous effectuons le classement.

• Quelques méthodes développées

Le processus se passe en deux phases :

- Lors de la première phase (hors ligne, dite d'apprentissage), il s'agit de déterminer un modèle des données étiquetées.
- La seconde phase (en ligne, dite de test) consiste à prédire l'étiquette d'une nouvelle donnée, connaissant le modèle préalablement appris.

Parfois il est préférable d'associer une donnée non pas à une classe unique, mais une probabilité d'appartenance à chacune des classes prédéterminées (on parle alors d'apprentissage supervisé probabiliste).

• Quelques méthodes développées

Les méthodes utilisées sont nombreuses parmi lesquelles nous citons :

- Réseaux de neurones,
- Boosting,
- SVM (Support Vector Machine) ou MVS(ou Machines à Vecteurs Supports),
- Arbres de décision,
- Etc.

• Quelques applications

Plusieurs applications existent parmi lesquelles nous avons retenu :

- Vision par ordinateur (VO)
- Reconnaissance de formes (RF)
- Reconnaissance de l'écriture manuscrite (REM)
- Reconnaissance vocale (RV)
- Traitement automatique de la langue (TAL)
- Bio-informatique (BI)
- Etc.

• Définition mathématique

Le problème devant être résolu est le suivant :

Soit β un ensemble de N individus classés, c'est-à-dire pour chaque individu $x_i \in X$ caractérisé par p variables notées $V_{k\in\{1,\dots,p\}}$, sa classe ou sa valeur de sortie $y_i \in Y$ est connue. On a donc que $\beta = \{(x_i, y_i)_{i\epsilon\{1,\dots,N\}}\}$.

Le problème devant être résolu est le suivant :

En se basant sur l'ensemble β ainsi définit, quelle est la valeur d'un nouvel individu ? Lorsque :

- $Y \subset \Re$ ***alors*** il s'agit d'une régression parce que la valeur de sortie est variante dans l'ensemble de réels
- $Y = \{1, \dots, K\}$ alors il s'agit de la classification parce que la valeur de sortie correspond à une classe d'appartenance, c'est-à-dire que chaque codifie une classe d'appartenance

I.9.2. Apprentissage non-supervisé

Quand le système ou l'opérateur ne disposent que d'exemples, mais non étiquetés, et que le nombre de classe et leur nature n'ont pas été prédéterminés, on parle d'apprentissage non supervisé ou clustering. Aucun expert n'est disponible ni requis. L'algorithme doit découvrir par lui-même la structure plus ou moins cachée des données. Le clustering est un algorithme d'apprentissage non supervisé.

Le système doit avoir dans l'espace de description (la somme des données) pour cibler les données selon leurs attributs disponibles, afin de les classer en groupe homogènes d'exemples. La similarité est généralement calculée selon la fonction de distance entre paires d'exemples. C'est ensuite à l'opérateur d'associer ou déduire du sens pour chaque groupe et pour les patterns d' apparition des groupes dans leur « espace ».

Divers outils mathématiques et logiciels peuvent l'aider. On parle aussi d'analyse des données en régression. Si l'approche est probabiliste (c'est à dire que chaque exemple au lieu d'être classé dans une seule classe est associé aux probabilité d'appartenir à chacune des classes), on parle alors de « soft clustering » (par opposition au « hard clustering »).

Ici les données ne sont pas étiquetées. Nous agissons sur les données sans avoir des connaissances supplémentaires sur les données fournies par l'expert du domaine. Contrairement à l'apprentissage supervisé, l'apprentissage non supervisé est une technique d'apprentissage automatique où l'on agit sur les données sur lesquelles aucune hypothèse n'a été formulée et aucune.

Certes, l'objectif de l'apprentissage non supervisé, vise à concevoir un modèle structurant l'information. La différence ici est que les comportements (ou catégories ou encore les classes) des données d'apprentissage ne sont pas connus, c'est ce que l'on cherche à trouver.

A. Classification automatique

L'idée de base de la classification automatique est de créer des sous-ensembles homogènes à partir d'une population hétérogène de manière à ce que les individus ayant les mêmes caractéristiques se ressemblent plus et ceux des caractéristiques différentes se diffèrent encore plus.

En d'autres termes, la classification automatique permet de maximiser l'inertie intra-classe et minimiser l'inertie inter-classe. En quelque sorte, la classification automatique effectue la segmentation en se basant sur le théorème d'Huygens.

Théorème de Huygens

$$= \mathcal{W}(z, g) + B(z) \quad (15)$$

$inertietotalinertie\ intralasse\ inertieinterclasse$

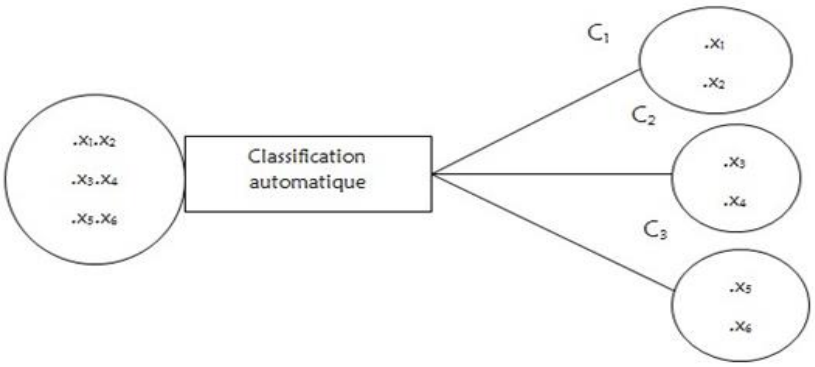


fig I.5. Apprentissage non supervisé

- **Quelques méthodes développées**

Il existe trois grandes méthodes dans l'apprentissage non supervisé :

- Les méthodes de partitionnement
- Les méthodes hiérarchiques
- Les méthodes de mélange gaussien

Méthodes de partitionnement

- K-means
- Nuées dynamiques
- Etc

Méthodes hiérarchiques

- Classification ascendante hiérarchique
- Classification descendante hiérarchique

I.9.3. Apprentissage par renforcement

Il fait référence à une classe de problèmes d'apprentissage automatique dont le but est d'apprendre, à partir d'expériences, ce qu'il convient de faire en différentes situations, de façon à optimiser une récompense quantitative au cours du temps.

Un paradigme classique pour présenter les problèmes d'apprentissage par renforcement consiste à considérer un agent autonome, plongé au sein d'un environnement, et qui doit prendre des décisions en fonction de son état courant. En retour, l'environnement procure à l'agent une récompense, qui peut être positive ou négative. L'agent cherche, au travers d'expériences intégrées, un comportement décisionnel (appelé stratégie ou politique, et qui est une fonction associant à l'état courant l'action à exécuter) optimal, en ce sens qu'il maximise la somme des récompenses au cours du temps.

L'algorithme apprend un comportement étant donné une observation. L'action de l'algorithme sur l'environnement produit une valeur de retour qui guide l'algorithme d'apprentissage.

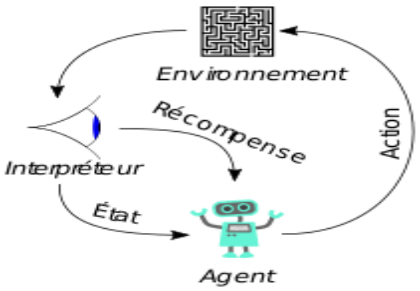


Figure .I.6. Schéma d'un modèle apprentissage par renforcement

I.10. CONCLUSION

Il a été question pour nous de présenter la technique à développer sur les machines Learning, tout en soulevant l'hypothèse selon laquelle qu'il est toujours difficile d'implémenter un classifieur parfait, vu que l'homme lui-même n'a pas encore atteint cette perfection prédictive, mais le but est tantôt d'utiliser un classifieur qui va minimiser les erreurs de prédiction. Les objectifs fixés étant atteints pour ce chapitre, nous allons passer maintenant au chapitre suivant, qui nous permettra d'aborder les différents classifieurs de classement que nous allons utiliser et comment les combiner.

CHAPITRE II. COMBINAISON DES CLASSIFIEURS AUTOMATIQUES

II.0. Introduction

Ces dernières années, un nombre significatif de travaux a porté sur la résolution de problèmes de classification par combinaison de classifieurs. Ainsi, plusieurs classifieurs entrainés à résoudre le même problème de classification peuvent être combinés, dans le but d'améliorer leurs performances individuelles. La notion de combinaison de classifieurs peut également être utilisée pour décomposer un problème complexe en sous-problèmes plus simples à résoudre : chaque sous-problème est résolu au moyen d'un classifieur, et les résultats ainsi obtenus sont ensuite combinés pour déterminer la solution du problème global. Il est donc légitime de s'attendre à ce que plusieurs classifieurs , travaillant sur des représentations identiques ou distinctes de la forme montrent des résultats meilleurs et dépassant ceux de chacun d'entre eux pris séparément , de plus, il a été observé que lorsqu'il s'agit de trouver le meilleur classifieur parmi un ensemble de classifieurs, les erreurs produites par chacun d'entre eux ne chevauchaient pas, c'est-à-dire qu'elles n'étaient pas forcément les mêmes, laissant penser que les informations fournies par chacun d'eux sont dans ce cas complémentaires.

II.1. Définition[8]

La combinaison d'informations consiste à regrouper des informations issues de plusieurs sources afin d'améliorer la prise de décision » précise Arnaud Martin.

Cette définition nous est utile dans la mesure où, nous combinons des résultats venant de la sortie de trois classifieurs comme le cas de notre modeste travail, notamment le réseau de neurones, machine à vecteur support et de l'arbre de décision. Ces informations des classifieurs peuvent être des données provenant de sources différentes ou bien présentant des caractéristiques différentes, extraites des mêmes données initiales.

Pour cette raison, on trouve souvent dans la littérature le terme fusion d'informations au lieu de fusion de données.

---

[8] T. Mitchell, Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression. Draft Version, 2005.

Il y a eu plusieurs définitions du terme fusion d'informations. En effet, le terme est souvent confondu avec le terme fusion de données. Dans un autre sens, il a été suggéré d'utiliser le terme combinaison d'informations dans un sens beaucoup plus large que la fusion d'information. En tout cas, ces termes définissent tout processus qui implique une opération mathématique effectuée sur au moins deux sources d'informations.

La combinaison n'est pas définie comme un terme opposé à la fusion. Le concept combinaison s'égale au concept fusion. Une synthèse de l'ensemble des terminologies et définitions relatives au domaine de fusion en télédétection a été aussi effectuée par Wald.

D'une manière générale, les techniques de combinaison (ou fusion) des informations sont divisées en systèmes répartis et centralisés selon que les conclusions tirées par les différents experts, sont combinées, ou que les différentes sources d'informations sont utilisées ensemble par un mécanisme d'inférence simple. Il s'agit ici de combiner les résultats de classifieurs en proposant un schéma de combinaison des systèmes répartis. D'où, le terme « combinaison de classifieurs ».

II.2. Intérêts de la combinaison

On se demande pourquoi combiner les classifieurs. Voici en quelque sorte les différents intérêts de la combinaison des classifieurs :

- La distribution des caractéristiques sur des classifieurs adaptés ;
- L'exploitation de la complémentarité entre classifieurs ;
- La prise en compte des performances de chacun des classifieurs ;
- La réduction de l'importance des certains choix initiaux.

II.3. Importance de combiner plusieurs classifieurs

Plusieurs chercheurs ont fait les constats suivants :

- Il n'existe pas de « meilleur » classifieur capable de traiter (apprendre) n'importe quelle distribution des données d'apprentissage ;
- Aucun classifieur ne peut discriminer suffisamment correctement un ensemble important de classes ;
- Le « réglage » d'un classifieur est un problème extrêmement difficile (on procède souvent par essai/erreur).

De ces constats et de bien d'autres aussi, a émergé l'idée de faire coopérer le classifieur

II.4. Types d'architecture

D'une manière générale et connue, il existe trois types d'architectures de combinaisons de classifieurs qui sont les suivants :

1. La combinaison séquentielle ou en série ;
2. La combinaison parallèle ;
3. La combinaison hybride (mélange de la combinaison séquentielle et la combinaison parallèle).

II.5. COMBINAISON EN SÉRIE

II.5.1. Définitions[9]

Cette combinaison est une organisation en niveaux successifs de décision permettant de réduire progressivement le nombre de classes possibles. h(t) permet de faire la prédiction ; ainsi le classifieur qui prédit correctement sera pondéré positivement (+P) et l'individu (l'instance) mal classé sera doté de signe négatif (-α). La pondération est appelée ici le poids qu'on va calculer par rapport à la distance qui existe avec le séparateur tracé par le premier classifieur. A chaque niveau, un seul classifieur prend en compte la réponse fournie par le classifieur précédent pour:

II.5.2. Boosting

A. Avantages

Filtrage progressif des décisions (réduction de l'ambiguïté)

B. Inconvénients

La sensibilité de l'ordre dans lequel sont placés les classifieurs, la supposition d'une connaissance à priori du comportement de chacun des classifieurs et la difficulté d'optimisation de l'ensemble (de classifieurs).

[9] A. Y. Ng and M. I. Jordan. On Discriminative vs. Generative Classifiers: A comparison of logistic regression and Naive Bayes. in NIPS 14, 2002.

II.5.3. COMBINAISON EN PARALLÈLE

Les classifieurs opèrent indépendamment les uns des autres puis on fusionne leurs réponses respectives. L'idée de base est de chercher un point d'accord entre les classifieurs pour aboutir à une décision unique. Le problème de la combinaison de classifieur en parallèle est celui de savoir comment élaborer une réponse finale unique à partir des k résultats fournis par k classifieurs.

II.5.4. Combinaison Hybride

L'approche hybride consiste à combiner à la fois des architectures séquentielles et parallèles afin de tirer pleinement avantage de chacun des classifieurs utilisés présente un exemple de combinaison hybride dans laquelle on combine un classifieur en série avec deux classifieurs en parallèle.

C'est une de combinaison des avantages deux architectures précédentes, elle permet :

- la réduction de l'ensemble des classes possibles.
- recherche d'un consensus entre les classifieurs

Cette hybridation présente les avantages ci-dessous :

- Tirer pleinement avantage de chacun des classifieurs utilisés.
- Nombreux schémas de combinaison pour tirer parti au mieux des données.

Par contre, elle souffre de la dépendance des données à traiter qui peuvent être très complexes à optimiser. Une telle combinaison a été appliquée dans la figure III.3 illustre la combinaison hybride.

II.6. CRITÈRES DE COMBINAISON[10]

La réussite de la combinaison des classifieurs dans le but d'avoir de meilleures performances, nécessite des critères suivants :

- Nombre de classifieur à entraîner : ce critère permet d'augmenter la performance collective des classifieurs, ce qui signifie plus le nombre est grand, plus on s'approche de 100% de performance ;
- Compétence : il faudrait au moins que nos classifieurs prédisent à plus de 50% pour atteindre une meilleure performance, au cas contraire, la majorité converge vers 0% ;

[10] (en) Richard O. Duda, Peter E. Hart, David G. Stork, *Pattern Classification*, Wiley-interscience, 2001.

- Minimum de diversité des prédictions : chaque classifieur doit prédire différemment des autres, au cas contraire ils commettent les mêmes erreurs et cela ne sert à rien de combiner les classifieurs.

II.7. VOTE MAJORITAIRE

Le jugement majoritaire est une méthode de vote par valeurs (les électeurs attribuent une mention à chaque candidat et peuvent attribuer la même mention à plusieurs candidats) pour laquelle la détermination du gagnant se fait par la médiane plutôt que par la moyenne.Contrairement aux méthodes utilisant la moyenne, le jugement majoritaire utilise des échelles de mentions verbales plutôt que numériques pour évaluer les candidats. Cette possibilité permet, d'après les inventeurs du jugement majoritaire, d'offrir aux électeurs des mentions dont les acceptions sont plus homogènes parmi les électeurs[11].

Le vote majoritaire consiste à faire voter de manière indépendante plusieurs classifieurs et à retenir la réponse ou la sortie majoritaire, correspondant normalement à la meilleure solution, c'est-à-dire celle qui ne sera pas trop loin de réalité que nous recherchons. Ceci est issu du fait que les classifieurs pris individuellement sont incapables d'assurer une bonne décision pour toutes les décisions.

Un vote de majorité est une méthode de combinaison de classifieurs ou l'on cherche à optimiser la pondération des votants. L'objectif est alors de construire un vote final plus performant et plus robuste que les votants individuels. Cependant, choisir des poids pertinents est une tâche parfois complexe.

La technique de VM est basée sur le critère de majorité (maximum, minimum, produit, moyenne, etc.). Elle est donc appliquée pour obtenir un résultat ou une décision optimale. Illustrons par un exemple, comment la décision peut être prise à partir de la TVM (technique de vote majoritaire) qui est la technique utilisée dans ce travail.

II.8. CONSTRUCTION D'UN SYSTÈME MULTI-CLASSIFIEURS

Un système multi-classifieur (Multiple Classifier System : MCS en anglais) est constitué d'un ensemble de différents classifieurs et d'une fonction de décision pour

[11] (en) Tom M. Mitchell, *Machine Learning*, 1997

combiner leurs sorties. La description d'un Système Multi-Classifieur suit les deux phases suivantes :

- Générer un ensemble de classifieurs complémentaire qui peuvent être combinés pour arriver à une solution optimale ;
- Définir la fonction de combinaison pour donner une décision finale

La difficulté de choix des classifieurs a poussé les chercheurs à développer des méthodes pour aider les concepteurs à effectuer ce choix. Parmi ces méthodes, celle de « test et sélection ».

L'idée principale de cette méthode est de produire un ensemble initial large de classifieurs candidats, puis sélectionner un sous-ensemble qui est jugé le plus valable pour aboutir à des performances optimales.

Pour le faire, ce procédé suit deux cycles :

- Le choix de la fonction de décision joue un rôle très important dans la conception d'un système multi-classifieur. La fonction de décision peut être conçue comme étant une fonction de combinaison, par conséquent la sortie du MCS reflète la décision de tout l'ensemble en utilisant par exemple le vote majoritaire, la somme pondérée, etc… ou bien comme étant une fonction de sélection dynamique d'un classifieur, dans ce cas il faut avoir au moins un classifieur dans l'ensemble qui pourra classer correctement un individu à l'entrée

- Le mécanisme de sélection sera plus efficace lorsque les classifieurs sont « spécialisés», d'où la facilité de calcul des conditions de sélection affectant chaque instance d'entrée au classifieur le plus convenable. Par contre lorsque les classifieurs manifestent des comportements différents.

II.9. Méthode de combinaison

Le regroupement des résultats se fait généralement par vote, c'est-à-dire par une combinaison linéaire des décisions suivie d'une décision ferme. Comme nous le verrons, pour un problème à K classes, on peut soit combiner des classificateurs partiels, qui ne savent par exemple que distinguer une classe d'une autre, soit combiner des classificateurs complets[12].
Le problème de différents classificateurs appris sur les mêmes données et son extension à l'apprentissage à deux étages (staked généralisation) ; Une extension du

[12] (en) Maloof, M. A., *Machine Learning and Data Mining for Computer Security*, Springer, 2006.

cas précédent, le codage correcteur de classes, qui permet en particulier de combiner des classificateurs à deux classes K=2 en un classificateur à K classes K2 :

- L'amplification, ou boosting, qui réutilise le même classificateur plusieurs fois en pondérant différemment les données d'apprentissage avant de combiner les résultats ;
- Le bagging, qui en est une variante ;
- L'apprentissage en cascade (cascading) qui utilise différents classificateurs à la suite les uns des autres. Les méthodes du vote à majorité simple et celle de la somme pondérée sont le plus souvent utilisées.

II.9.1. Combinaison par majorité simple

Après avoir obtenu les prédictions des classes de chaque classifieur, la combinaison par vote majoritaire consiste à choisir la classe la plus proposée par les classifieurs. Un résultat de rejet est généré si toutes les classes ont le même nombre de votes. La condition ici est que chaque classifieur doit donner un résultat différent des deux autres[13].

II.9.2. Combinaison par somme pondérée

Pour prendre en considération l'importance que peut avoir un classifieur par rapport aux autres, nous utilisons une pondération afin de représenter cette notion d'importance. Alors, la combinaison consiste à faire la somme du produit des classifieurs avec les pondérations qui leurs sont associées.

II.9.3. Combinaison homogène et hétérogène

La majorité des combinaisons de classifieur appliquent un seul algorithme dans l'apprentissage de base, ce qui se traduit par une homogénéité chez tous les classifieurs. Les classifieurs homogènes font référence aux classifieurs du même type, avec des qualités similaires. D'autres méthodes appliquent des classifieurs hétérogènes, donnant lieu à des ensembles hétérogènes. Les classifieurs hétérogènes sont de différents types. Plusieurs méthodes qui implémentent les techniques de combinaison de classifieur séquentielle, parallèle et hybride ont été proposées (le bagging, et boosting et le staking) dont le bagging et le boosting sont les plus connues.

[13] (en) Dash, T., « A review of some techniques for inclusion of domain-knowledge into deep neural networks », *Scientific Reports*, 2022.

Dans le cadre de notre travail, nous parlerons seulement du Boosting car c'est l'objet de notre travail**.**

II.10. DÉFINITION

Un classifieur est une fonction mathématique ou un algorithme qui réalise les tâches d'affectation[14].

Il se charge de ranger ou classer les individus homogènes dans une même classe de manière à prédire correctement leur classe d'appartenance. Ce rangement ne se fait pas au hasard bien entendu mais sur base des certaines caractéristiques ou données établies au préalable après le prétraitement de ces caractéristiques ou données.

Le classifieur a pour fonction d'assigner une étiquette de classe à l'objet qui lui est présenté par l'extracteur de caractéristiques. Ainsi le rôle d'un classifieur est de donner une probabilité d'appartenance d'un objet à une classe, donc le classifieur doit pouvoir gérer ces imperfections, et la façon d'entraîner des caractéristiques pour y faire face est un domaine de recherche en soi.

II.11. PROBLÈMES

L'analyse prédictive utilise un outil appelé « classifieur », pour résoudre une vaste gamme de problèmes. Un de ces problèmes est celui de déterminer, à partir de ses caractéristiques, l'état d'une situation cachée ce qui signifie qu'on réfère à un problème de classification.

Le problème est de savoir comment prédire un résultat parmi de nombreuses données et être sûr du résultat. Les classifieurs permettent de faire la prédiction d'un résultat ou des résultats malgré le nombre de caractéristiques présentées.

C'est ainsi que dans ce chapitre nous parlerons plus largement des classifieurs automatiques et aussi ses différents types. Il se basera sur les différents types de classifieurs, dont nous allons expliciter le fonctionnement de chacun de types et différents principes.

Nous allons soulever l'hypothèse selon laquelle il est toujours difficile de créer un classifieur parfait, dans le sens où l'homme lui-même n'a pas encore atteint cette

[14] (en) Montanez-Barrera, J. A., « Correlated-informed neural networks: A new machine learning framework to predict pressure drop in micro-channels », *International Journal of Heat and Mass Transfer*, 2022.

perfection prédictive, mais le but est de créer un classifieur qui va minimiser les erreurs de prédiction.

1. Dans le domaine bancaire l'un des problèmes qui nécessite la combinaison des classifieurs est le problème de l'octroi crédit, cela signifie que comment attribuer une somme d'argent à un client en tenant compte des caractéristiques établies par la banque ;
2. La banque cherche à s'assurer sur un client, s'il est crédible ou pas. Sur ce, on demande l'avis des plusieurs experts ensuite juger en faveur de la majorité ;
3. L'éligibilité des clients en fonction des critères adoptés par la banque.

II.12. FONCTIONNEMENT D'UN CLASSIFIEUR[15]

Dans ce contexte, le rôle d'un classifieur est de classer dans des groupes ou classes les échantillons qui ont des propriétés similaires (identiques), mesurées sur des mêmes observations. Donc au sens plus raisonnable et bref, le classifieur se base sur une population des plusieurs échantillons ; et forme dans cette population des sous-groupes ou classes de manière à ce que chaque individu ait connaissance de sa classe d'appartenance. Et cela se fait grâce aux différentes méthodes ou algorithmes de prédiction des classifieurs entre autres : SVM, K plus proches voisins, Réseaux de neurones, Arbres de décision, Etc

II.13. TYPES DE SORTIE D'UN CLASSIFIEUR

Un classifieur peut donner une valeur en sortie de différents types qui sont les suivants :

1. Sortie de type classe (classifieur de type I)

C'est le type le plus général mais qui apporte le moins d'informations. Le classifieur prend en entrée un nouvel individu, après avoir étudié les caractéristiques de cet individu, le classifieur propose une seule classe parmi les classes dans laquelle sera affecté cet individu.

$$e_j(x) = C_i(i \in \{1, \dots, m\}) \qquad (15)$$

2. Sortie de type ensemble (classifieur de type II)

Le classifieur prend en entrée un nouvel individu et après avoir étudié son comportement, il propose un ensemble de classes candidates sans préciser ses préférences.

$$e_j(x) = \{C_i / \, i \leq m\} \qquad (16)$$

[15] (en) Bikmukhametov, T., « Combining machine learning and process engineering physics towards enhanced accuracy and explainability of data-driven models », *Computers & Chemical Engineering*, 2020.

3. Sortie de type rang (classifieur de type III)

Le classifieur prend en entrée un nouvel individu et après avoir étudié son comportement, il propose un ensemble de classe candidate en précisant l'ordre de préférence. Cela se traduit par l'attribution d'un rang pour chaque classe ;

si la probabilité que l'individu soit affecté dans la classe $_1$ est grande, alors cette classe aura pour rang r_1.

4 Sortie de type mesure (classifieur de type IV)

Cette sortie est la plus riche en informations puisque le classifieur dans ce cas associe à chaque classe une mesure de confiance qui peut être, par exemple, une probabilité

$e_j(x) = M_1{}^j, M_2{}^j, \dots, M_m{}^j]$où $M_i{}^j$est la mesure attribuée à la classe () par le classifieur(j).

II.14. MESURES DE PERFORMANCES D'UN CLASSIFIEUR[16]

Pour une entrée donnée, un classifieur peut générer les réponses suivantes :

- Un rejet **:** pour indiquer que le classifieur n'a pas pu identifier cette entrée ;
- Une reconnaissance **:** dans ce cas, il identifie bien l'entrée, et il lui attribue sa classe appropriée ;
- Une substitution **:** le classifieur attribue une autre classe à l'entrée.

La performance d'un classifieur peut être mesurée en calculant les trois taux suivant:

$$taux\ de\ rejet = \frac{Nombre\ de\ clients\ réjetées}{Nombre\ total\ de\ clients}$$

$$taux\ de\ reconnaissance = \frac{Nombre\ de\ clients\ reconnues}{Nombre\ total\ de\ clients}$$

$$taux\ de\ substitution = \frac{Nombre\ de\ clients\ males\ reconnus}{Nombre\ total\ de\ clients}$$

[16] http://www.college-de-france.fr/site/stanislas-dehaene/_course.htm [Consulté en avril 2022].

II.15. COMPLEXITÉ D'UN CLASSIFIEUR

Le but de la théorie de la complexité est de pouvoir dire si un problème est (ou non) soluble efficacement par un ordinateur. Dans la plupart des cas, c'est le temps de calcul excessif qui limite l'utilisation de certains algorithmes ce qui implique que la mesure de complexité le plus utile, c'est la complexité temporelle

II.15.1. Complexité temporelle

La complexité temporelle d'un algorithme A est la fonction Temps où Temps(x) =

est le nombre d'instructions exécutées pendant le calcul A(x).

On aimerait définir la complexité temporelle d'un problèmeΠ comme étant la complexité temporelle de l'algorithme A le plus efficace pour résoudre, mais il est possible de démontrer qu'on peut toujours rendre un algorithme A plus efficace. La complexité temporelle doit être asymptotique. Pour toute fonction f, la complexité temporelle d'un langage L est en O(f) s'il existe un algorithme A qui décide L et des constantes n0; c telles que[17] :

$$\forall x|x| > n_o \Rightarrow Temps_A(x) \leq cf(|x|)$$

La classe de complexité temporelle TEMPS[f] est la classe de tous les langages de complexité temporelle en O(f).

$$\mathbf{TEMPS[n] \subseteq TEMPS[n \log n \subseteq TEMPS[n^2] \subseteq TEMPS[2^n]}$$

II.16. Complexité polynomiale

II.16.1. Définition

Pour tout polynôme p(n), La classe P est identique pour les différents modèles de calcul (machine de Turing, machine RAM, etc.) La définition de P est aussi indépendante des détails du système de codage (tels que la structure de données qu'on utilise.

Pour stocker un graphe). Est Ce que P = traitable ("tractable")? Ce n'est pas le cas si le degré du polynôme est trop élevé ou si les constantes sont trop grandes.

---

[17] https://www.college-de-france.fr/site/yann-lecun/Recherches-sur-l-intelligence-artificielle.htm [Consulté en juillet 2022].

II.16.2.Problème NP

Le nom NP vient de "Nondeterministic Polynomial-time" [Karp 1972], en français ‘ Non-déterministe polynomial’. La complexité polynomiale sur une machine non-déterministe (une machine qui essaie toutes les possibilités en même temps) ≡ un vérificateur de complexité polynomiale. Pour les problèmes fonctionnels, il existe des classes équivalentes à P et NP : FP et FNP.

Un problème fonction avec relation binaire associée R (I; sol) appartient à FP s’il existe un algorithme A de complexité polynomiale tel que renvoie sol telle que R (I; sol) est vrai (ou ‘ non’ si un tel sol n’existe pas).

Un problème fonction avec relation binaire associée R (I; sol) appartient à FNP si, étant données I et sol, on peut vérifier R (I; sol) en temps polynomial.

$$FP \neq FNP \Leftrightarrow P \neq NP$$

II.17.QUELQUES CLASSIFIEURS[18]

II.17.1. ARBRES DE DÉCISION

Un arbre est un graphe connexe sans cycle. On l’appelle arborescence (ou arbre enraciné) s’il possède un nœud spécial appelé racine. Un arbre de décision est un arbre au sens informatique du terme. L’arbre de décision est un outil de classification et prédiction, sa popularité repose en grande partie sur sa simplicité. Il est donc composé de :

- Les feuilles qui représentent les valeurs (les décisions possibles) des variables cibles (les étiquettes de classe) ;
- Les arcs qui correspondent à des combinaisons de variables d’entrée qui mènent à ces valeurs ;
- Les nœuds internes qui décrivent un test sur des variables d’apprentissage, ils sont appelés nœuds de test. Pour parcourir un arbre de décision et trouver une solution il faut partir de la racine. Chaque réponse possible est prise en compte et permet de se diriger vers un des fils du nœud. On descend de l’arbre en passant par les nœuds de test jusqu'à tomber sur une feuille pour classer l’individu testé. C’est ainsi qu’on dit qu’un arbre de décision est une structure qui permet de déduire un résultat à partir de décisions successives.

[18] Pierre Kafunda K., *note de cours info centre*, L2 math info Unikin, p. 17, 2022

Les arbres de décision sont des règles de classification qui basent leur décision sur une suite de tests associés aux attributs, les tests étant organisés de manière arborescente.

L'objectif est de rendre l'arbre aussi petit que possible (facilitant la recherche) tout en faisant un compromis entre les taux d'erreurs sur l'ensemble d'apprentissage et sur l'ensemble de tests de pouvoir généraliser. Un exemple expliquant le fonctionnement d'un arbre de décision.

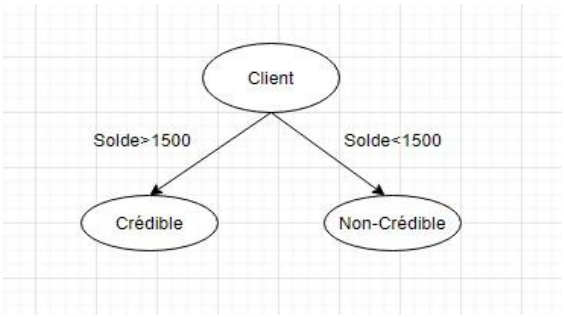


Figure.II.1. Arbre de décision

Cet arbre explique la crédibilité d'un client dans une banque. Le client est crédible dans cet exemple si son solde bancaire est supérieur à 1500$, dans le reste des cas le client n'est pas crédible.

Principe

Tout algorithme d'arbre de décision se fait en deux phases suivantes :

1) La phase d'expansion (où l'on construit l'arbre) qui permet la construction de l'arbre en séparant le jeu de données en différents sous ensemble en fonction de la valeur d'entrée. C'est un processus répétitif et qui s'arrête après remplir tous les partitionnements. On l'appelle alors l'induction descendante d'arbres de décision.

Dans toutes les méthodes, on trouve les trois opérateurs suivants :

- Décider si un nœud est terminal, c'est-à-dire décider si un nœud doit être étiqueté comme une feuille ou porter un test.
- Si un nœud n'est pas terminal, sélectionnez un test à lui associer.
- Si un nœud est terminal, lui affecter une classe. On peut définir alors un modèle général d'algorithme, sans spécifier comment seront définis les trois opérateurs décrits plus haut :

Algorithme

Début
Initialiser l'arbre courant à l'arbre vide ; la racine est le nœud courant Répéter
Décider si le nœud courant est terminal
Si le nœud est terminal alors
Lui affecter une classe
Sinon
Sélectionner un test et créer autant de nouveaux nœuds fils qu'il y a de réponses possibles au test

FinSi
Passer au nœud suivant non exploré s'il en existe
Jusqu'à obtenir un arbre de décision Fin.

2) La phase d'élagage qui consiste à supprimer les parties de l'arbre qui ne semblent pas performantes pour prédire la classe de nouveaux cas, remplacés par un nœud terminal (associé à la classe majoritaire).

Généralement cette suppression est de type "bottom-up" (du bas vers le haut: des extrémités vers la racine), base sur une estimation du taux d'erreur de classification: un arbre est élagué a un certain nœud si le taux d'erreur estimé à ce nœud (en y allouant la classe majoritaire) est inférieur au taux d'erreur obtenu en considérant les sous arbres terminaux.

II.17.2. L'algorithme CART

Du nom anglais « classification and regression Trees » autrement dit « Arbres de classement et de régression » en français. Il fut proposé pour la première fois par Leo Breiman en 1984 et ne génère qu'un arbre binaire c'est-à-dire pas plus de deux fils. L'algorithme choisit la variable et le seuil qui maximisent la décroissance de l'impureté du nœud par rapport à la cible. Cette impureté est mesurée par un indice dit l'indice de Gini. Cette mesure est la vraisemblance qu'un élément du nœud soit incorrectement étiqueté par un tirage aléatoire qui respecte la loi statistique de la cible estimée dans le nœud. C'est donc l'indice de Gini dans le critère de partitionnement. L'indice de Gini, due Corrado Gini est une métrique qui mesure avec laquelle fréquence un élément aléatoire de l'ensemble serait mal classé soit son étiquette était choisie aléatoirement selon la distribution des étiquettes dans le sous-ensemble.

L'indice de diversité de Gini peut être calculé en sommant la probabilité pour chaque élément d'être choisi, multiplié par la probabilité qu'il soit mal classé. Il atteint sa valeur minimale (0) lorsque tous les éléments de l'ensemble sont dans une même classe de la variable cible. Si on suppose que la classe prend une valeur dans l'ensemble 1,2,…, n, et si f_i est la fraction des éléments de l'ensemble avec l'étiquette i dans l'ensemble, on aura :

$$G = \sum_{i=1}^{n} f_i (1 - f_i) = 1 - \sum_{i=1}^{n} {f_i}^2 \qquad (19)$$

Les principales idées de **CART** sont les suivantes :

- L'hétérogénéité d'un nœud se mesure par une fonction non négative qui doit être :
- Nulle si et seulement si le nœud est homogène cela signifie que tous les individus appartiennent à la même valeur de Y.
- Maximale lorsque les valeurs de Y sont équiprobables ou très dispersées.
- Plus l'indice de Gini est proche de 0 plus le nœud est pur ;
- La variable de partitionnement retenue est celle qui maximise le gain de pureté qui est défini par :

$$Gain = G(S) - [G(Fils_1) + G(Fils_2)] avec\ Gain >= 0. \quad (20)$$

- Pour déterminer la taille de l'arbre, on utilise la procédure de post-élagage.
- Arbre complètement développé sur un premier échantillon.

Arbre réduit de manière à optimiser le taux de mauvais classement sur un deuxième échantillon. L'algorithme CART commence par le jeu de données initial X comme nœud racine. La première étape de CART consiste à découper au moins cette racine en deux nœuds fils. A chaque itération de l'algorithme, on recherche alors, suivant le même procédé, la meilleure façon de les découper en deux nouveaux nœuds, et ainsi de suite. Les arbres sont développés jusqu'à atteindre une condition d'arrêt.

Une règle d'arrêt classique consiste à ne pas découper des nœuds qui contiennent moins d'un certain nombre d'observations. Les nœuds terminaux qui ne sont plus découpés sont appelés les feuilles de l'arbre. Il est à noter, que l'on ne découpe pas

un nœud pur cela signifie qu'un nœud ne contenant que des observations dont les sorties sont les mêmes. Cet algorithme a des avantages comme :

- Il est non paramétrique ;
- Il n'utilise pas de sélection de variables nécessaires ;
- Il est invariable aux transformations monotones des attributs ;
- Il fait une bonne gestion des données aberrantes.

Nota : l'algorithme est de type rang

II.17.3. RÉSEAUX DE NEURONES

Les réseaux de neurones, fabriqués de structures cellulaires artificielles, constituent une approche qui permet d'aborder sous des nouveaux angles les problèmes de perception, de mémoire, d'apprentissage et de raisonnement. Ils s'avèrent aussi des alternatives très importantes pour contourner certaines des limitations des ordinateurs classiques. Grâce à leur traitement combiné de l'information et à leurs mécanismes inspirés des cellules nerveuses (neurones humains), ils infèrent des propriétés émergentes permettant de solutionner des problèmes qui étaient qualifiés de complexes.

2. Présentation

Le terme de réseau de neurones a un lien étroit avec la biologie bien qu'il est plus métaphorique que scientifique. Les méthodes mathématiques évoquées ici de manière générale ont été développées pour la modélisation des systèmes nerveux vivants. Leurs applications se situent dans les domaines qui n'ont généralement aucun rapport avec la neurobiologie. Le système nerveux humain est constitué des neurones, comme unités de traitements de l'information. Le cerveau biologique a plus de 10 milliards de neurones, une diversité de neurones que des fonctions ; voir, parler, entendre ou se déplacer. Les fonctions plus abstraites comme penser, imaginer, inventer, choisir ou décider nécessitent plusieurs actions simultanées. C'est l'ensemble du cerveau et ces neurones qui peuvent être sollicités pour donner à chacun de nous, la capacité de l'intelligence.

La jonction entre deux neurones est appelée la synapse ; ce sont les points de contact entre deux neurones, des microscopiques espaces de communication grâce auxquels chaque neurone va pouvoir communiquer avec 10000 autres qui forment au total plus des milliers des liaisons neuronales dans le cerveau. Le neurone est une cellule composée d'un corps et d'un noyau. Le corps cellulaire se ramifie pour former ce que l'on nomme les dendrites. Celles-ci sont parfois si nombreuses que l'on parle alors de chevelure dendritique ou d'arborisation dendritique. C'est par les dendrites que l'information est acheminée de l'extérieur vers le soma (corps du neurone).

L’information traitée par le neurone chemine ensuite le long de l’axone (unique) pour être transmise aux autres neurones.

La transmission entre deux neurones n’est pas directe. En fait, il existe un espace intercellulaire de quelques dizaines d’Angstroms (10^{-9} m) entre l’axone du neurone afférent et les dendrites du neurone efférent.

3.1. Neurone biologique et neurone formel

3.2. Neurone biologique

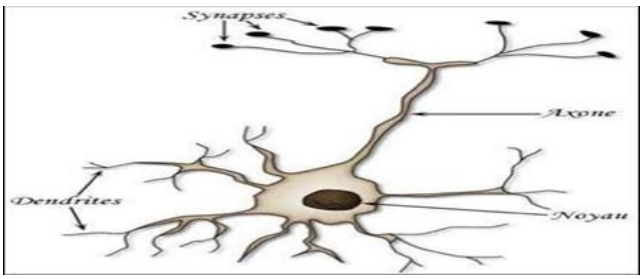


Figure II.2. Neurone biologique

3.3. Neurone formel

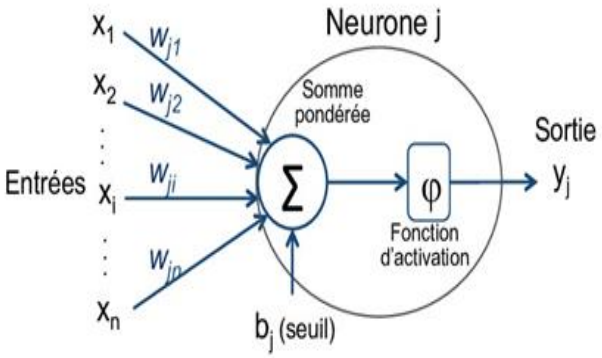


Figure.II.3. Neurone formel ou artificiel

u_i représente la somme pondérée des entrées de neurone :

$$u_i = \sum_j w_{ij} + b_j \qquad (21)$$

Avec :

- x_j**;** l’entrée j connectée au neurone i représenté ;

- b_j;le seuil du neurone ;
- w_{ij};le poids de connexion reliant l'entrée j au neurone i (poids synaptiques). $Y_j = \varphi(u_i)$la sortie du neurone et φ la fonction d'activation

On appelle réseau de neurone artificiel (RNA), est un ensemble des processeurs (neuronaux) interconnectés fonctionnant en parallèle, chaque processeur calcule le potentiel (la somme pondérée) sur base des données qu'il reçoit, applique la fonction d'activation sur le potentiel, le résultat est transmis aux processeurs en aval.

3.4. Architecture d'un réseau de neurone

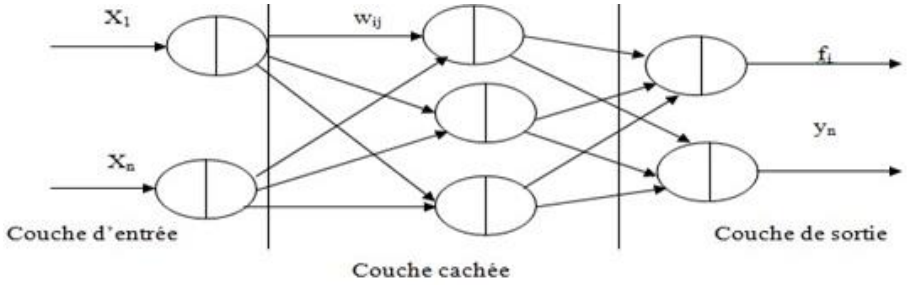


Figure.II.4. Architecture d'un réseau de neurone

L'architecture du réseau de neurones se résume de la manière suivante : La couche d'entrée contient des neurones qui activent le réseau, et la seule tâche de cette couche est de transmettre des informations. La couche cachée reçoit des informations de la couche d'entrée et les transmet aux autres couches (couches cachées ou couche de sortie). La couche de sortie ne peut recevoir que des informations de la couche cachée et renvoie la sortie du réseau neuronal. Un réseau contient exactement une couche d'entrée et une couche de sortie; et le nombre de couches cachées dépend de la configuration. Mais il est recommandé d'utiliser une couche cachée pour des raisons de simplicité. L'objectif du réseau des neurones est de trouver l'ensemble des poids w qui correspondra au mieux la sortie exacte. Lorsque nous traitons des données non linéairement séparables, on utilise MultiLayer Perceptron (MLP).

L'architecture MLP contient au moins une couche cachée. Chaque neurone, à l'exception de ceux de la couche d'entrée, exécute la somme des informations reçues et active la somme, c'est-à-dire qu'elle calcule la somme d'une valeur réelle r(ex. $r \in [0,1]$). Le modèle est défini de telle sorte que chaque neurone a la même fonction d'activation g(.) et a son propre biais b. Un biais permet de déplacer la sortie de 0 et 1 ou vice versa.

5. Approches des réseaux de neurones

5.1. Approche constructive

Elle est utilisée lorsque les individus sont linéairement séparables. Ici, on apprend le nombre d'unités et les poids en même temps en commençant avec une seule unité en général.

5.2. Approche par rétro propagation de gradient

Cette approche est utilisée lorsque les individus sont non linéairement séparables. La rétro propagation de gradient détermine les poids qui minimisent un coût.

6. Algorithmes des réseaux de neurones

6.1. Algorithme de Hebb

L'algorithme de Hebb est basé sur la loi de Hebb de 1949. Cette loi dit que « deux neurones qui sont activés au même instant, augmentent la force de connexion ». Le changement de poids dépend de la co-activation des neurones pré et post-synaptiques.

Algorithme

Début

L'ensemble des n individus d'apprentissage est X. Chaque exemple xi possède p variables dont les valeurs sont notées x_{ij} **Pour chaque donnée, xi 0 est une variable virtuelle toujours égale à 1. La classe de l'individu xi et yi.**

d_i **est la classe (sortie) désirée de l'individu** x_i**.**

- **Initialisations :**

Initialiser les poids w_i **et le seuil Ɵ à des valeurs petites aléatoirement.**

- **Répéter**
- **Présenter une entrée de l'ensemble d'apprentissage**
- **Calculer la sortie** y_i **de l'individu** x_i **en connaissant ses entrées telle que**

$$a = \sum_{j=0}^{p} (w_j x_{ij}) - \theta \quad (22)$$

$y_i = sign(a)$

Si $y_i \neq d_i$ **cela signifie que la classe prédite est différente de la classe désirée alors il y a modification des poids** w_k **Pour tous**

les poids $w_k \in \{1, \dots, p\}$ **faire** $w_k = w_k + \alpha(d_i x_{ik})$

α est une constante positive qui spécifie le pas de modification des poids (0< α < 1).

Fin pour

Finsi

Jusqu'à ce que tous les individus de l'ensemble d'apprentissage ne soient pas traités correctement (c'est-à-dire, qu'il y a modification des poids.

Fin

II.19. Conclusion

Nous avons traité au chapitre les différents classifieurs ou classificateurs que nous utilisons dans ce travail. Dans ce présent chapitre nous touchons les différentes approches ou méthodes de la combinaison des classifieurs, tel que les méthodes séquentielle et parallèle. Certes, tout en s'accentuant sur la deuxième méthode parallèle, où nous avons développé la technique de vote majoritaire pour la mise en place d'un système de combinaison parallèle. En revanche, nous voici arrivés à la fin de ce deuxième chapitre qui nous a donné une idée sur notre travail. Enfin nous passons à notre troisième chapitre qui nous donne l'opportinuté de descendre sur terrain

CHAPITRE 3 ANALYSE PRÉALABLE

III.1. Présentation de l'entreprise (*Eco Bank*) [19]

Ecobank Transnational Incorporated (ETI), société anonyme, est créée en 1985 comme holding bancaire à l'initiative de la Fédération des chambres de commerce d'Afrique de l'Ouest avec le soutien de la Communauté économique des États de l'Afrique de l'Ouest (CEDEAO). Au début des années 1980, l'industrie bancaire de l'Afrique de l'Ouest est dominée par les banques d'État et les banques étrangères. Il n'existe pratiquement aucune banque du secteur privé africain dans la région.

La Fédération ouest-africaine des chambres de commerce a favorisé et accompagné le lancement d'un projet pour la création d'un établissement bancaire régional privé en Afrique de l'Ouest. Eco promotions S.A. est créée en 1984. Ses actionnaires fondateurs ont réuni le capital de départ destiné aux études de faisabilité et aux activités promotionnelles ayant conduit à la création d'ETI.

En octobre 1985, ETI est créée avec un capital autorisé de 100 millions $ US. Le capital initial versé de 32 millions $ EU est mobilisé auprès de plus de 1500 personnes et institutions des pays de l'Afrique de l'Ouest. Le principal actionnaire est le Fonds de coopération, de compensation et de développement (Fonds de la CEDEAO), bras financier de la CEDEAO.

ETI signe un accord de siège avec le gouvernement du Togo en 1985, qui lui confère le statut d'organisation internationale, jouissant des droits et privilèges nécessaires lui permettant d'exercer en tant qu'institution régionale, avec un statut d'établissement financier non-résident. ETI débute ses activités avec sa première filiale au Togo en mars 1988

III.1.2. Pôles d'activités

Ecobank est une banque universelle axée sur l'Afrique subsaharienne, offrant des services de banque de détail, banque de grande clientèle et banque d'investissement, ainsi que des services bancaires transactionnels aux états, aux établissements

[19] Le service clientèle ***Eco BANK***

financiers, aux multinationales, aux entreprises locales, aux petites et moyennes entreprises (PME) et aux particuliers.

Ecobank propose ses services à travers trois pôles axés sur la clientèle : Services bancaires aux particuliers, Banque commerciale et Banque d'investissement. Une plate-forme informatique intégrée exploitée par eProcess, la filiale technologique du groupe située à Accra, prend en charge les trois pôles d'activités.

III.1.3. Banque de grande clientèle et d'investissement

La banque de grande clientèle et d'investissement s'aligne constamment sur les opportunités de marché. Elle propose des solutions financières aux multinationales, aux entreprises régionales, aux entreprises publiques, aux établissements financiers et aux organisations internationales. La banque de grande clientèle et d'investissement fait appel à des technologies de pointe pour mieux servir ses clients. Elle offre en particulier les services suivants : banque transactionnelle, taux, change et matières premières (FICC), banque d'investissement, valeurs mobilières, gestion de patrimoine et d'actifs, cartes, crédits et Liquidités. Le service recherche et trésorerie de la banque appuie la banque de grande clientèle et d'investissement.

III.1.4. Banque Commerciale

La Banque Commerciale offre aux entreprises locales et aux PME les services suivants : Banque transactionnelle, Taux, Change et Matières premières, Crédits et Liquidités.

III.1.5. Implantation

Ecobank est aujourd'hui la première banque panafricaine, présente dans 33 pays du continent. Elle est implantée dans plus de pays africains qu'aucune autre banque. Ecobank est, à ce jour, opérationnelle dans des pays d'Afrique de l'Ouest, de l'Est, Centrale et Australe, à savoir : Afrique du Sud, Bénin, Burkina Faso, Burundi, Cap Vert, Cameroun, Congo Brazzaville, Côte d'Ivoire, Éthiopie, Gabon, Gambie, Ghana, Guinée, Guinée Bissau, Guinée équatoriale, Kenya, Liberia, Malawi, Mali, Mozambique, Niger, Nigeria, Ouganda, République centrafricaine, République démocratique du Congo, Rwanda, Sao Tomé & Principe, Sénégal, Sierra Leone, Sud Soudan, Tanzanie, Tchad, Togo, Zambie et Zimbabwe. La Banque possède également une filiale à Paris et des bureaux de représentation à Dubaï, Johannesburg, Londres, Luanda et Pékin.

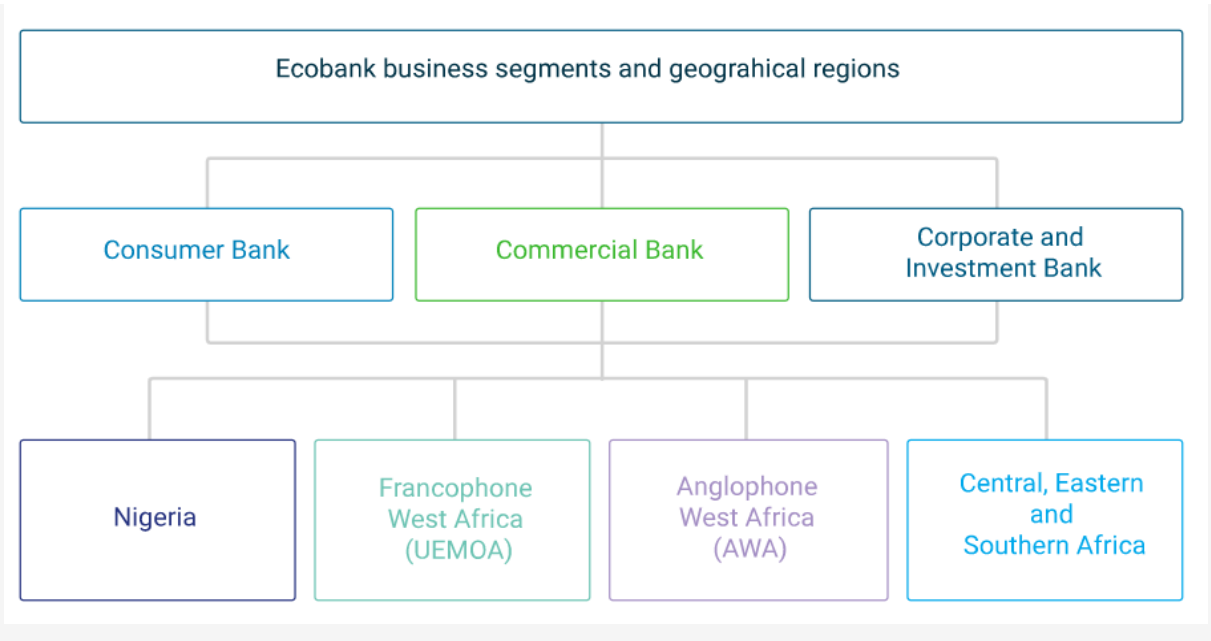


Fig:III.1. Implantation de la banque[http://ecobank.org}

III.1.6. Organigramme

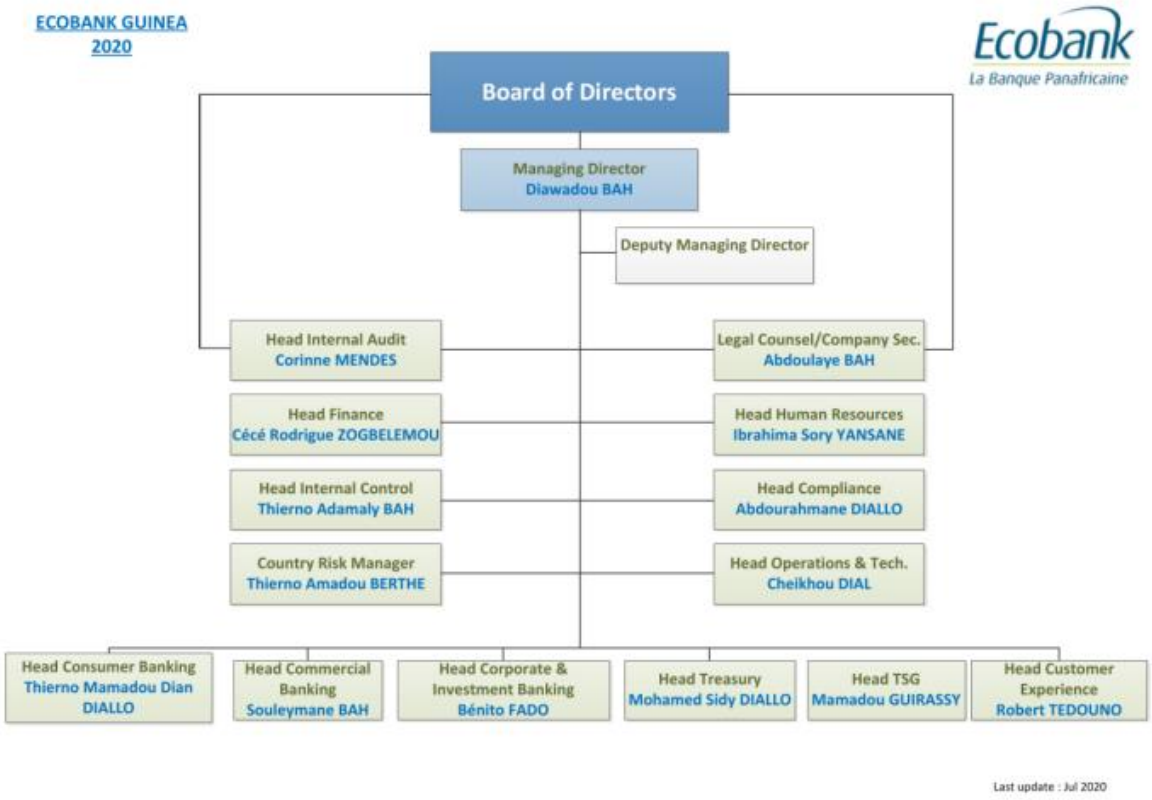


Fig:III.2. Organigramme de la banque [http://ecobank.org}

III.2. Crédit bancaire

III.2.1. Définition[20]

Le mot « Crédit » à la même étymologie que le mot « Croire » (en latin, « crédo » = je crois, j'ai confiance). C'est donc une activité qui repose la confiance, celle que le prêteur accorde à l'emprunteur, de qui, il attend le remboursement du prêt.

En finance, le crédit englobe les diverses activités de prêt d'argent, que se croit sous la forme de contrats de prêts bancaire ou de délais de paiement d'un fournisseur à un client.

Le crédit est généralement porteur d'un intérêt que doit payer le débiteur. (le bénéficiaire du crédit, appelé aussi emprunteur) au créancier (celui qui accorde le crédit, appelé aussi prêteur).

Dans le domaine bancaire, un crédit bancaire est une mise (ou une promesse) à disposition de fonds à une date ou une période donnée contre obligation de remboursement moyennant une rémunération. Un crédit se conclut par l'intermédiaire d'un contrat entre un emprunteur et un préteur. Les banques sont les principaux fournisseurs de crédit, tant aux particuliers qu'aux entreprises.

III.2.2.Typologie des crédits bancaires[21]

Plusieurs types de crédit bancaire peuvent se distinguer selon le critère retenu par l'analyste en l'occurrence : l'objet, la durée et les caractéristiques.

a- Selon l'objet du crédit :

Les crédits pour les particuliers : (personnes physiques)

- § Crédit-bail (leasing, location-vente)
- § Crédit à la consommation (affecté, personnel, revolving)
- § Crédit immobilier (Epargne logement)

Les crédits pour les entreprises et les professionnels :

- § Crédit d'exploitation (escompte, faculté de caisse, affacturage, Credoc)
- § Crédit d'investissement (prêt d'équipement, crédit-bail).

---

[20] Louis-Ferdinand Céline, *Mort à crédit*, Paris, Gallimard, coll. « Folio », p.622, 1985.

[21] Jérôme Lasserre Capdevielle et Michel Storck, *Le crédit aspects juridiques et économiques*, Paris, Dalloz,p 210, ,2012.

b- Selon la durée du crédit :

- § Crédit à très court terme (jusqu'à 3 mois)
- § Crédit à court terme (jusqu'à 2 ans)
- § Crédit à moyen terme (jusqu'à 7ans)
- § Crédit à long terme (jusqu'à 20 ans)
- § Crédit à très long terme (au-delà de 20 ans, voire perpétuel.

c- Selon la forme du crédit (les caractéristiques)

- § Monnaie : En Monnaie nationale, en devises étrangers ;
- § Mode d'amortissement : constant, à annuité constante ou remboursable infinie,
- § Type de taux : A taux fixe, à taux variable ou indexé, à taux variable capé,
- § Mécanisme : permanent ou revolving, sur ligne de crédit ;
- § Contrat : sur compte débiteur, sur contrat de prêt, emprunt obligation, en pool,
- § Garantie : En blanc, garanti.

3- Cycle de vie d'un crédit bancaire

Le cycle de vie d'un crédit est différent selon la nature du crédit. Toutefois, on peut distinguer des phases communes à tous les types.

a- Etude et Mise en place[22]

Cette phase d'instruction de dossiers comporte une série d'activités :

- § Formulation Demande du client
- § Analyse des dossiers
- § Prise de décision
- § Prise des garanties
- § Déblocage du crédit en cas de crédit par décaissement

b- Remboursement du crédit[23]

[22] Gérard Biardeaud et Philippe Florès, *Crédit à la consommation : protection du consommateur*, Paris, Delmas Express, p200, 2012.

[23] www.legifrance.gouv.fr (consulté le 23 septembre 2020).

Le second processus s'articule autour des activités suivantes :

- § Appel d'échéances
- § Remboursement d'échéances normales
- § Remboursement anticipé des échéances
- § Clôture et délivrance des mains levées

c- Recouvrement

Le recours à la troisième phase s'opère généralement selon la logique suivante :

- § Constatation des impayés
- § Renégociation des conditions
- § Classement des crédits en créance douteuse
- § Contentieux

CHAPITRE IV. IMPLEMENTATION ET INTERPRETATION DE DONNEES

IV.1. Introduction

Le dataset original contient 1000 entrées avec 20 attributs catégoriels/symboliques préparés par le professeur Hofmann. Dans ce travail, chaque entrée représente une personne qui prend un crédit auprès de notre banque, ainsi chaque personne est classifiée comme ayant un bon ou mauvais dossier prêt selon la description de nos données.

Il est presque impossible de comprendre l'ensemble de données d'origine en raison de son système compliqué de catégories et de symboles. Ainsi, nous avons écrit un petit script Python pour le convertir en un fichier CSV lisible. Plusieurs colonnes sont simplement ignorées, car à notre avis, soit, elles ne sont pas importantes, soit leurs descriptions sont obscures. Les attributs sélectionnés sont:

- **Âge** (numérique)
- **Sexe** (texte : masculin, féminin)
- **Emploi** (numérique : 0 - non qualifié et non résident, 1 - non qualifié et résident, 2 - qualifié, 3 - hautement qualifié)
- **Logement** (texte : propre, loué ou gratuit)
- **Comptes d'épargne** (texte - peu, modéré, assez riche, riche)
- **Compte courant** (numérique, en DM - Deutsch Mark)
- **Montant du crédit** (numérique, en DM)
- **Durée** (numérique, en mois)
- **Objectif** (texte : voiture, mobilier/équipement, radio/TV, appareils électroménagers, réparations, éducation, affaires, vacances/autres)
- **Risque** (Valiable cible - Bon ou Mauvais dossier de prêt)

IV.2. Implémentation en python

- Importation de bibliothèques
- Importation du dataset

```
# le bibliothèques utilisées
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

#Importiion des données
df_credit = pd.read_csv("german_credit_dataok.csv",index_col=0)
```

IV.3. Visualisation des données :

- Recherche des types de données
- Nombres des valeurs nuls
- Des valeurs uniques
- Les premières lignes de notre dataset

```
#Searching for Missings,type of data and also known the shape of dat
a
print(df_credit.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1000 entries, 0 to 999


Data columns (total 10 columns):
Age                 1000 non-null int64
Sex                 1000 non-null object
Job                 1000 non-null int64
Housing             1000 non-null object
Saving accounts     817 non-null object
Checking account    606 non-null object
Credit amount       1000 non-null int64
Duration            1000 non-null int64
Purpose             1000 non-null object
Risk                1000 non-null object
dtypes: int64(4), object(6)
memory usage: 85.9+ KB
None
```

```
#Looking unique values
print(df_credit.nunique())
#Looking the data
print(df_credit.head())
```

```
Age                  53
Sex                   2
Job                   4
Housing               3
Saving accounts       4
Checking account      3
Credit amount       921
Duration             33
Purpose               8
Risk                  2
dtype: int64


   Age     Sex  Job Housing Saving accounts Checking account  Credit
amount  \
0   67    male    2     own             NaN           little
```

```
1169
1   22  female    2     own          little         moderate
5951
2   49    male    1     own          little              NaN
2096
3   45    male    2    free          little           little
7882
4   53    male    2    free          little           little
4870

   Duration              Purpose  Risk
0         6             radio/TV  good
1        48             radio/TV   bad
2        12            education  good
3        42  furniture/equipment  good
4        24                  car   bad
```

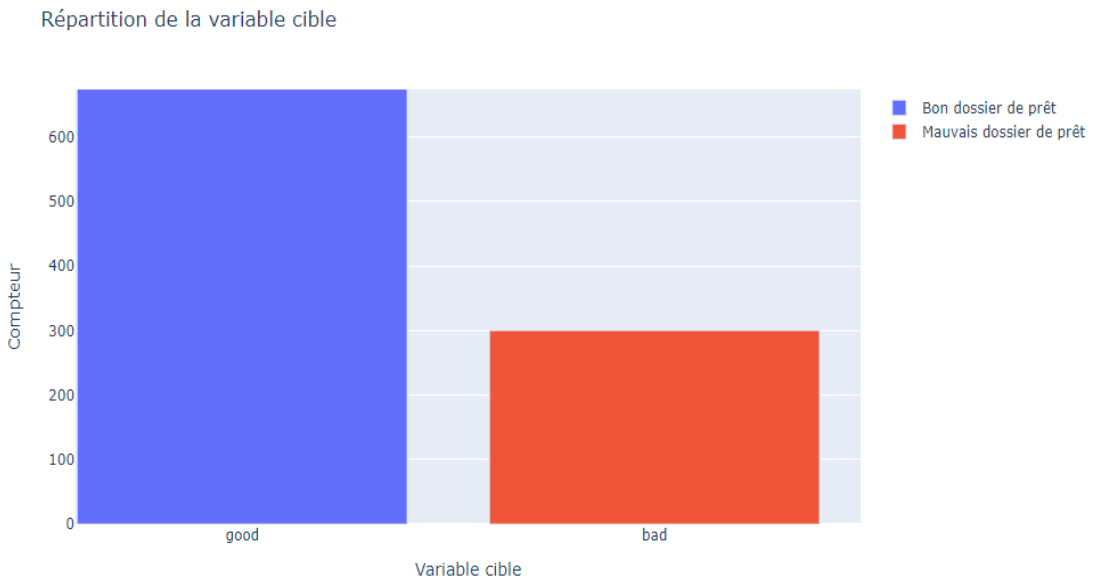
IV.4. Explorations

- En commençant par la distribution de la colonne Age.
- Quelques graphismes
- Traversée de colonnes

Examen de la variable cible et de sa distribution

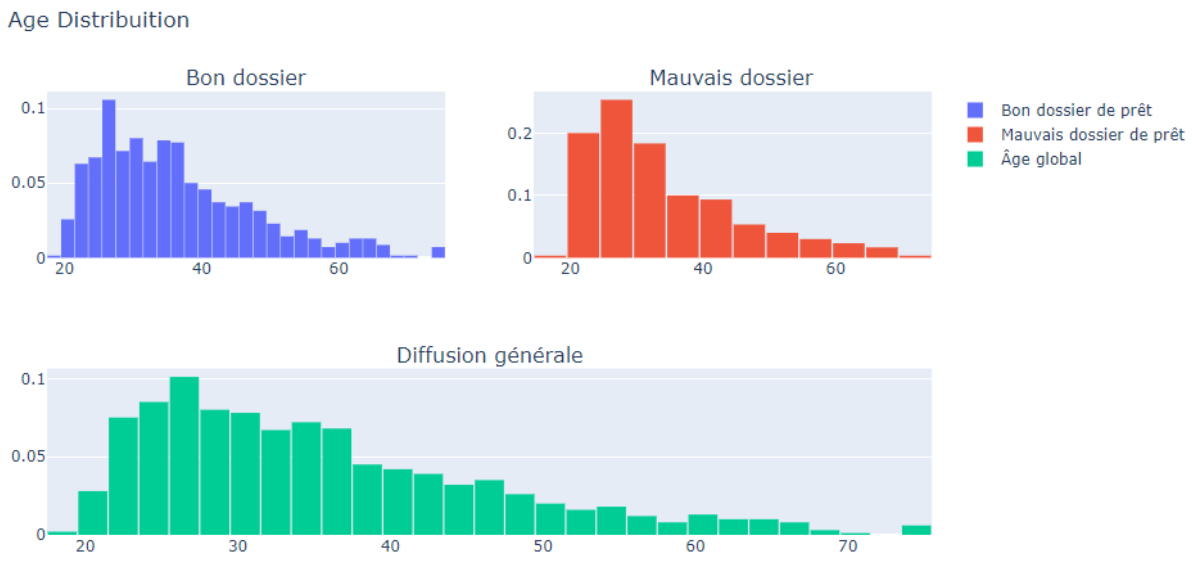
```
#Cette bibliothèque fonctionne avec plotly
import plotly.offline as py
#ce code, nous permet de travailler sur la version offline de plotly
py.init_notebook_mode(connected=True)
# C'est comme "plt" de matplot
import plotly.graph_objs as go
# Il est utile d'obtenir des outils plotly
import plotly.tools as tls
# la bibliothèque sera utilisée pour ignorer certains avertissements
import warnings
# Pour faire le compteur de certaines fonctionnalités
from collections import Counter

trace0 = go.Bar(
            x = df_credit[df_credit["Risk"]== 'good']["Risk"].value_
counts().index.values,
            y = df_credit[df_credit["Risk"]== 'good']["Risk"].value_
counts().values,
            name='Bon dossier de prêt')

trace1 = go.Bar(
            x = df_credit[df_credit["Risk"]== 'bad']["Risk"].value_c
ounts().index.values,
```

```
            y = df_credit[df_credit["Risk"]== 'bad']["Risk"].value_c
ounts().values,
            name='Mauvais dossier de prêt’)

data = [trace0, trace1]

layout = go.Layout()

layout = go.Layout(
    yaxis=dict(
        title='Compteur'
    ),
    xaxis=dict(
        title='Variable cible'
    ),
    title='Répartition de la variable cible'
)

fig = go.Figure(data=data, layout=layout)

py.iplot(fig, filename='grouped-bar')
```

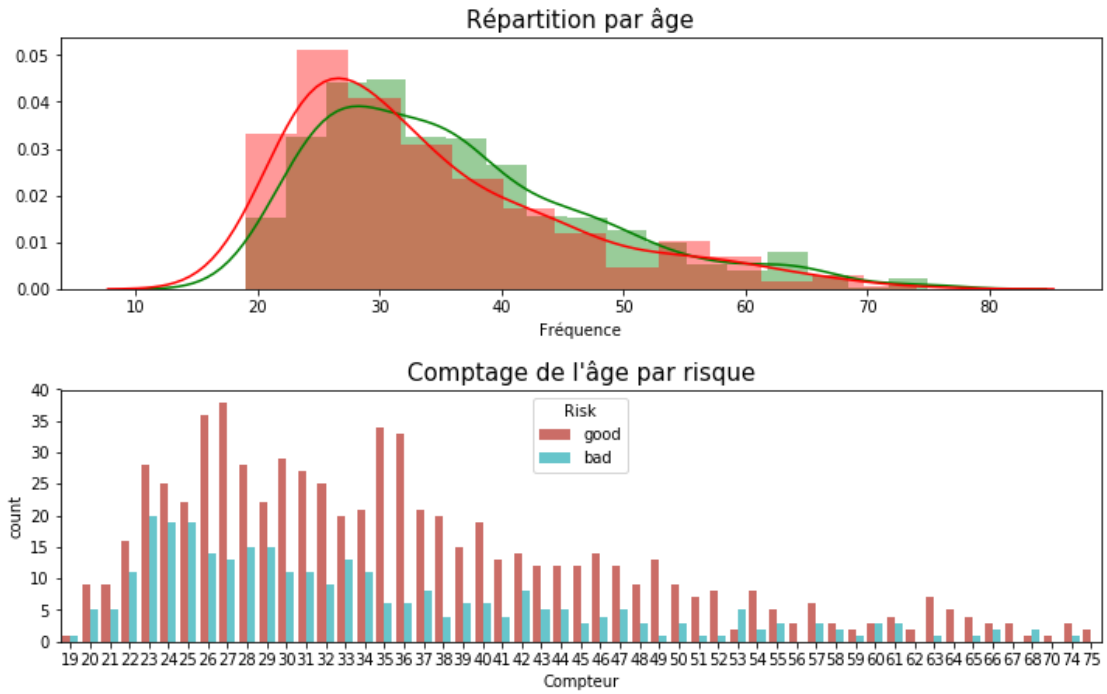



```
df_good = df_credit.loc[df_credit["Risk"] == 'good']['Age'].values.t
olist()
df_bad = df_credit.loc[df_credit["Risk"] == 'bad']['Age'].values.tol
ist()
df_age = df_credit['Age'].values.tolist()

#First plot
trace0 = go.Histogram(
    x=df_good,
```

```
    histnorm='probability',
    name="Bon dossier de prêt"
)
#Second plot
trace1 = go.Histogram(
    x=df_bad,
    histnorm='probability',
    name="Mauvais dossier de prêt"
)
#Third plot
trace2 = go.Histogram(
    x=df_age,
    histnorm='probability',
    name="Âge global"
)

#Creating the grid
fig = tls.make_subplots(rows=2, cols=2, specs=[[{}, {}], [{'colspan'
: 2}, None]],
                        subplot_titles=('Bon dossier','Mauvais dos
sier', 'Diffusion générale'))

#setting the figs
fig.append_trace(trace0, 1, 1)
fig.append_trace(trace1, 1, 2)
fig.append_trace(trace2, 2, 1)

fig['layout'].update(showlegend=True, title='Age Distribuition', bar
gap=0.05)
py.iplot(fig, filename='custom-sized-subplot-with-subplot-titles')
```



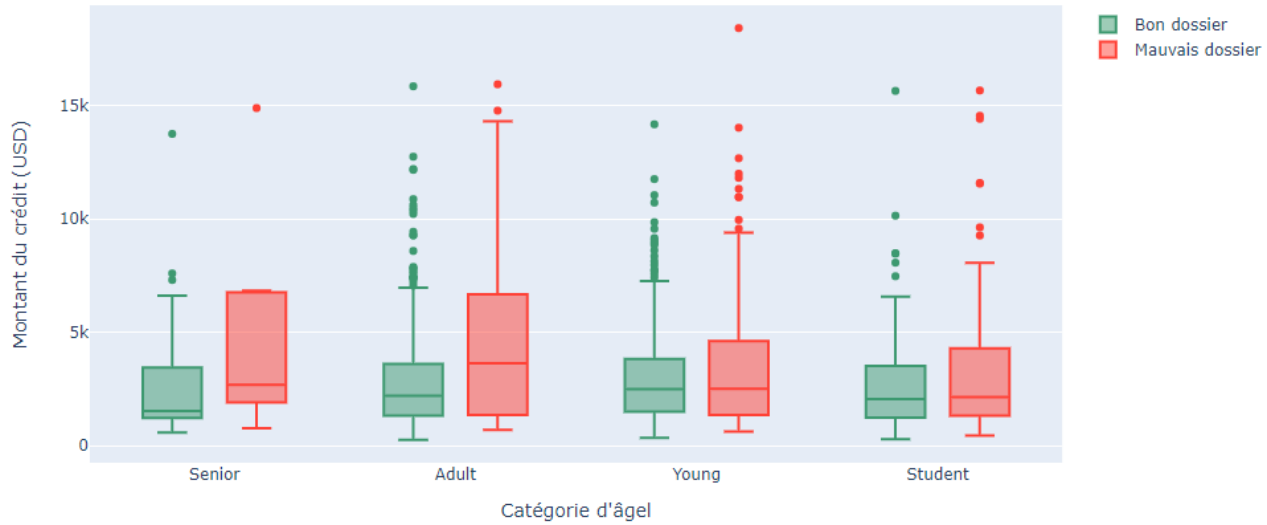
```
df_good = df_credit[df_credit["Risk"] == 'good']
df_bad = df_credit[df_credit["Risk"] == 'bad']

fig, ax = plt.subplots(nrows=2, figsize=(12,8))
plt.subplots_adjust(hspace = 0.4, top = 0.8)

g1 = sns.distplot(df_good["Age"], ax=ax[0],
             color="g")
g1 = sns.distplot(df_bad["Age"], ax=ax[0],
             color='r')
g1.set_title("Répartition par âge", fontsize=15)
g1.set_xlabel("Age")
g1.set_xlabel("Fréquence")

g2 = sns.countplot(x="Age",data=df_credit,
              palette="hls", ax=ax[1],
              hue = "Risk")
g2.set_title("Comptage de l'âge par risque", fontsize=15)
g2.set_xlabel("Age")
g2.set_xlabel("Compteur")
plt.show()
```



Création d'une variable catégorielle à gérer avec la variable Age

```
#Let's look the Credit Amount column
interval = (18, 25, 35, 60, 120)
cats = ['Student', 'Young', 'Adult', 'Senior']
df_credit["Age_cat"] = pd.cut(df_credit.Age, interval, labels=cats)
df_good = df_credit[df_credit["Risk"] == 'good']
df_bad = df_credit[df_credit["Risk"] == 'bad']
```

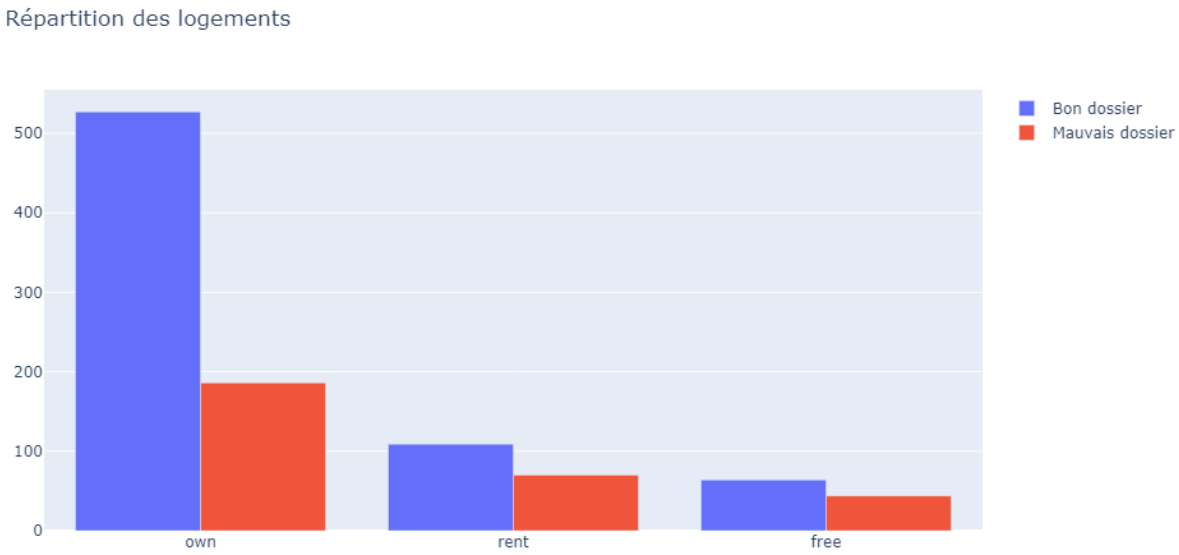
```
trace0 = go.Box(
    y=df_good["Credit amount"],
    x=df_good["Age_cat"],
    name='Bon dossier',
    marker=dict(
        color='#3D9970'))
trace1 = go.Box(
    y=df_bad['Credit amount'],
    x=df_bad['Age_cat'],
    name='Mauvais dossier',
    marker=dict(
        color='#FF4136' ))
data = [trace0, trace1]
layout = go.Layout(
    yaxis=dict(
        title='Montant du crédit (USD)',
        zeroline=False),
    xaxis=dict(
        title="Catégorie d'âgel"),
    boxmode='group')
fig = go.Figure(data=data, layout=layout)
trace0 = go.Bar(
    x = df_credit[df_credit["Risk"]== 'good']["Housing"].value_count
s().index.values,
    y = df_credit[df_credit["Risk"]== 'good']["Housing"].value_count
s().values,
    name='Bon dossier')
#Second plot
trace1 = go.Bar(
    x = df_credit[df_credit["Risk"]== 'bad']["Housing"].value_counts
().index.values,
    y = df_credit[df_credit["Risk"]== 'bad']["Housing"].value_counts
().values,
    name="Mauvais dossier"
)

data = [trace0, trace1]

layout = go.Layout(
    title='Répartition des logements'
)
```



Nous pouvons voir que les propriétaires ont plus des bons dossiers de prêt

```
fig = {
    "data": [
        {
            "type": 'violin',
            "x": df_good['Housing'],
            "y": df_good['Credit amount'],
            "legendgroup": 'Good Credit',
            "scalegroup": 'No',
            "name": 'Bon dossier',
            "side": 'negative',
            "box": {
                "visible": True
            },
            "meanline": {
                "visible": True
            },
            "line": {
                "color": 'blue'
            }
        },
        {
            "type": 'violin',
            "x": df_bad['Housing'],
            "y": df_bad['Credit amount'],
            "legendgroup": 'Bad Credit',
            "scalegroup": 'No',
            "name": 'Mauvais dossier',
            "side": 'positive',
            "box": {
                "visible": True
            },
            "meanline": {
```

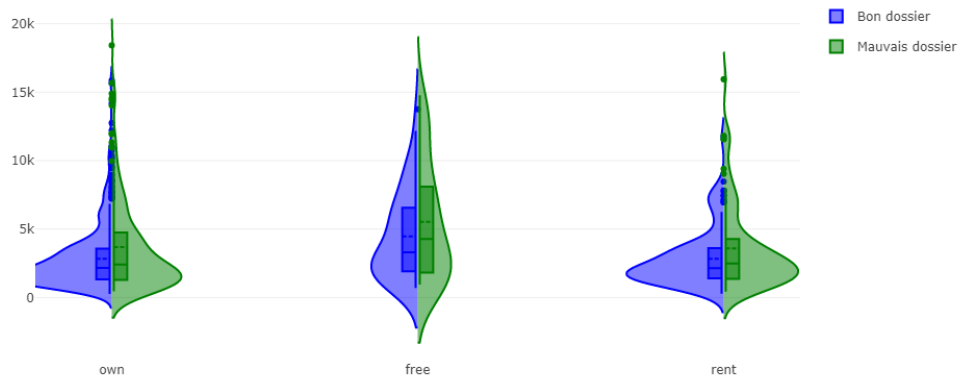
```
                "visible": True
            },
            "line": {
                "color": 'green'
            }
        }
    ],
    "layout" : {
        "yaxis": {
            "zeroline": False,
        },
        "violingap": 0,
        "violinmode": "overlay"
    }
}
```



Mouvements intéressants! Les valeurs les plus élevées proviennent de la catégorie "libre" et nous avons une répartition différente par Race

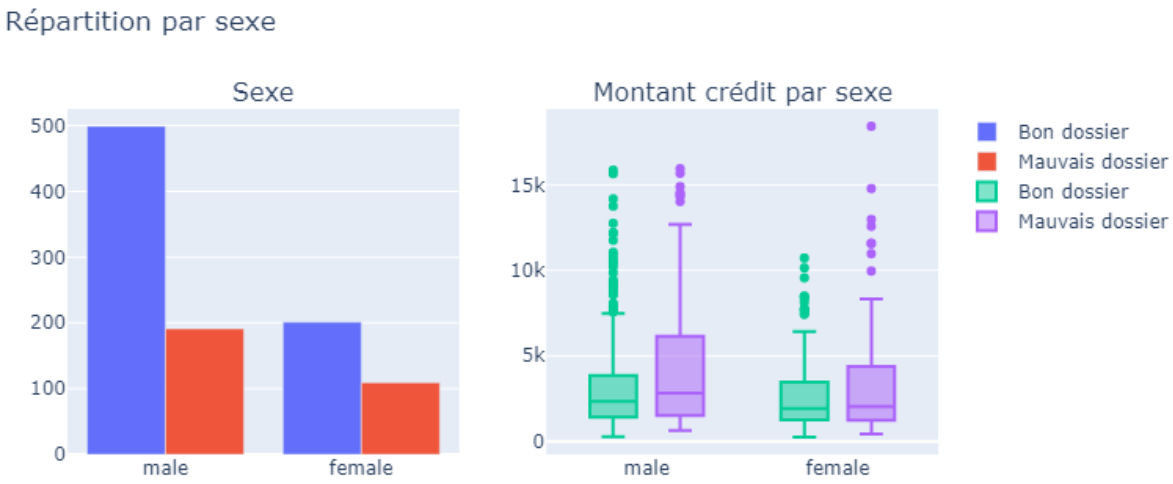
```
                        #First plot
                        trace0 = go.Bar(
    x = df_credit[df_credit["Risk"]== 'good']["Sex"].value_counts().
                            index.values,
    y = df_credit[df_credit["Risk"]== 'good']["Sex"].value_counts().
                               values,
                            name='Bon dossier'
                                )

                        #First plot 2
                        trace1 = go.Bar(
    x = df_credit[df_credit["Risk"]== 'bad']["Sex"].value_counts().i
```

```
                ndex.values,
    y = df_credit[df_credit["Risk"]== 'bad']["Sex"].value_counts().v
                alues,
               name="Mauvais dossier"
                )

                #Second plot
               trace2 = go.Box(
        x = df_credit[df_credit["Risk"]== 'good']["Sex"],
    y = df_credit[df_credit["Risk"]== 'good']["Credit amount"],
               name=trace0.name
                )

                #Second plot 2
               trace3 = go.Box(
         x = df_credit[df_credit["Risk"]== 'bad']["Sex"],
     y = df_credit[df_credit["Risk"]== 'bad']["Credit amount"],
               name=trace1.name
                )

          data = [trace0, trace1, trace2,trace3]


         fig = tls.make_subplots(rows=1, cols=2,
                   subplot_titles=('Sexe', 'Montant crédit par
                        sexe'))

             fig.append_trace(trace0, 1, 1)
             fig.append_trace(trace1, 1, 1)
             fig.append_trace(trace2, 1, 2)
             fig.append_trace(trace3, 1, 2)
```

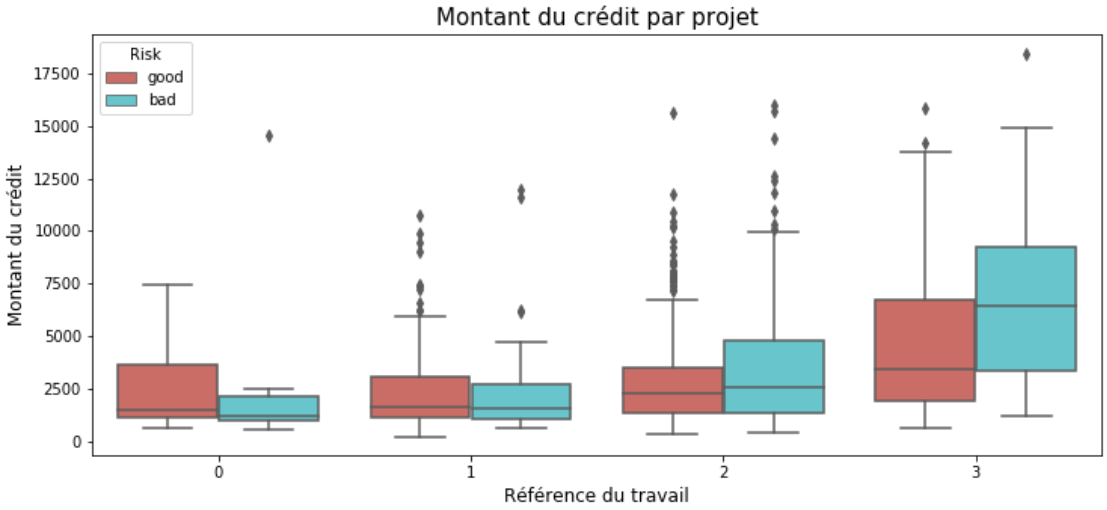


Nous créons des catégories d'âge et observation la répartition du montant du crédit par dossier de prêt

```
#First plot
trace0 = go.Bar(
    x = df_credit[df_credit["Risk"]== 'good']["Job"].value_counts().
index.values,
    y = df_credit[df_credit["Risk"]== 'good']["Job"].value_counts().
values,
    name='Bonne répartition du crédit'
)
```



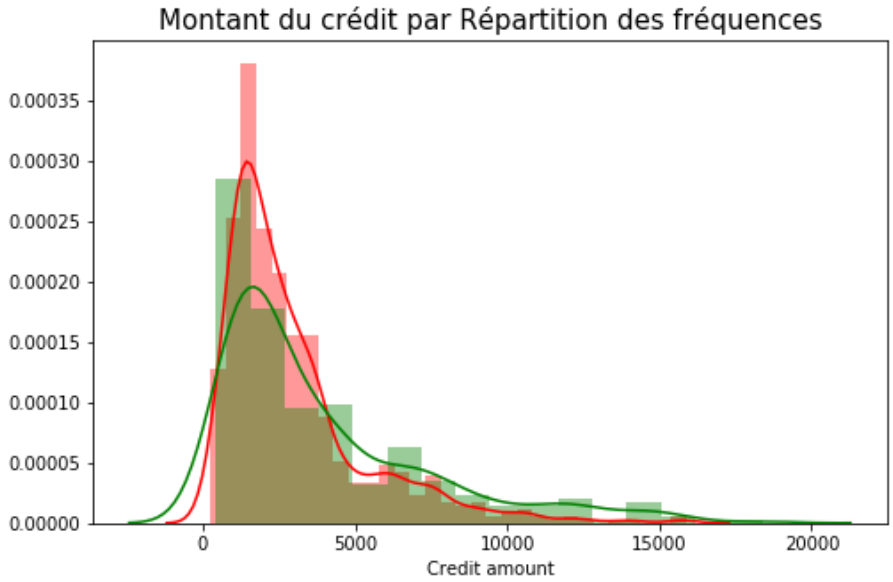

```
g1 = sns.boxplot(x="Job", y="Credit amount", data=df_credit,
            palette="hls", ax=ax[0], hue="Risk")
g1.set_title("Montant du crédit par projet", fontsize=15)
g1.set_xlabel("Référence du travail", fontsize=12)
g1.set_ylabel("Montant du crédit", fontsize=12)

g2 = sns.violinplot(x="Job", y="Age", data=df_credit, ax=ax[1],
                hue="Risk", split=True, palette="hls")
g2.set_title("Référence du type d'emploi x âge", fontsize=15)
g2.set_xlabel("Référence du travail", fontsize=12)
g2.set_ylabel("Age", fontsize=12)

plt.subplots_adjust(hspace = 0.4,top = 0.9)

plt.show()
```



En regardant la distribution du montant du crédit

```
import plotly.figure_factory as ff

import numpy as np

# Add histogram data
x1 = np.log(df_good['Credit amount'])
x2 = np.log(df_bad["Credit amount"])

# Group data together
hist_data = [x1, x2]

group_labels = ['Bon dossier', 'Mauvais dossier']

# Create distplot with custom bin_size
fig = ff.create_distplot(hist_data, group_labels, bin_size=.2)
```

```
plt.figure(figsize = (8,5))

g= sns.distplot(df_good['Credit amount'], color='r')
g = sns.distplot(df_bad["Credit amount"], color='g')
g.set_title("Montant du crédit par Répartition des fréquences", font
size=15)
plt.show()
```



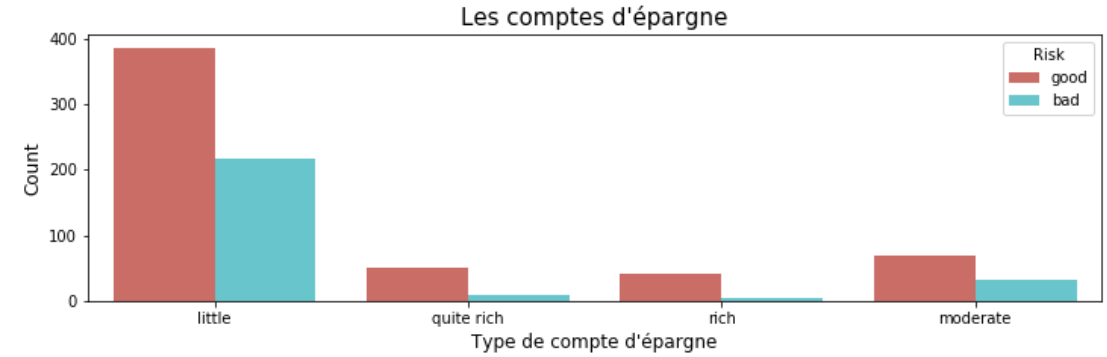
Répartition des comptes d'épargne par risque de remboursement

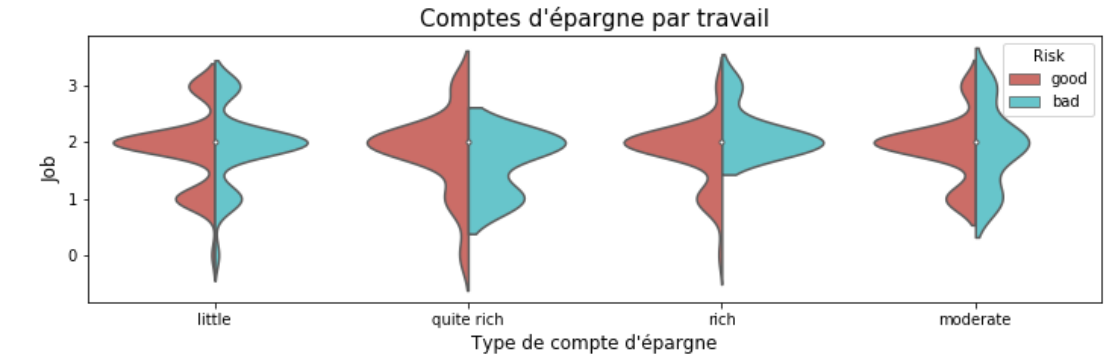
```
from plotly import tools
import numpy as np
import plotly.graph_objs as go

count_good = go.Bar(
    x = df_good["Saving accounts"].value_counts().index.values,
    y = df_good["Saving accounts"].value_counts().values,
    name='Bon dossier'
)
count_bad = go.Bar(
    x = df_bad["Saving accounts"].value_counts().index.values,
    y = df_bad["Saving accounts"].value_counts().values,
    name='Mauvais dossier'
)


box_1 = go.Box(
    x=df_good["Saving accounts"],
    y=df_good["Credit amount"],
    name='Bon dossier'
)
box_2 = go.Box(
```

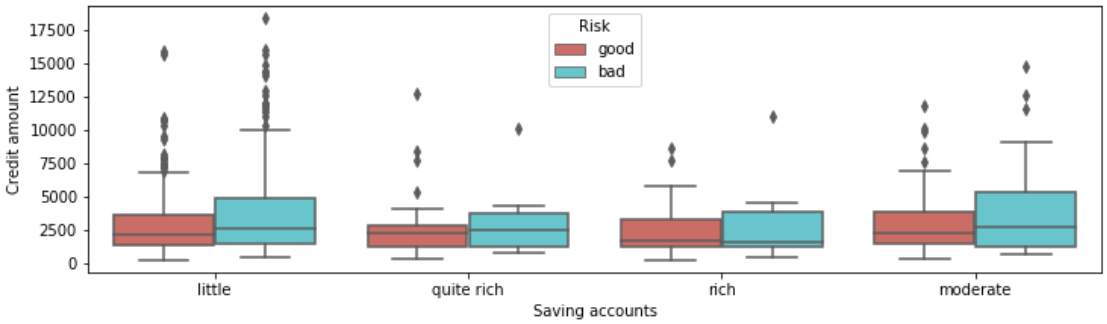
```
    x=df_bad["Saving accounts"],
    y=df_bad["Credit amount"],
    name='Mauvais dossier'
)

scat_1 = go.Box(
    x=df_good["Saving accounts"],
    y=df_good["Age"],
    name='Bon dossier'
)
scat_2 = go.Box(
    x=df_bad["Saving accounts"],
    y=df_bad["Age"],
    name='Mauvais dossier'
)

data = [scat_1, scat_2, box_1, box_2, count_good, count_bad]

fig = tools.make_subplots(rows=2, cols=2, specs=[[{}, {}], [{'colspa
n': 2}, None]],
                          subplot_titles=("Comptes d'épargne","Monta
nt du crédit par compte d'épargne",
                                          "Âge par comptes épargne")
)

fig.append_trace(count_good, 1, 1)
fig.append_trace(count_bad, 1, 1)

fig.append_trace(box_2, 1, 2)
fig.append_trace(box_1, 1, 2)

fig.append_trace(scat_1, 2, 1)
fig.append_trace(scat_2, 2, 1)



fig['layout'].update(height=700, width=800, title='Saving Accounts E
xploration', boxmode='group')

py.iplot(fig, filename='combined-savings')

print("Description des comptes d'épargne de distribution par risque
de remboursement:  ")
print(pd.crosstab(df_credit["Saving accounts"],df_credit.Risk))

fig, ax = plt.subplots(3,1, figsize=(12,12))
g = sns.countplot(x="Saving accounts", data=df_credit, palette="hls"
,
              ax=ax[0],hue="Risk")
g.set_title("Les comptes d'épargne", fontsize=15)
```

```
g.set_xlabel("Type de compte d'épargne", fontsize=12)
g.set_ylabel("Count", fontsize=12)

g1 = sns.violinplot(x="Saving accounts", y="Job", data=df_credit, pa
lette="hls",
               hue = "Risk", ax=ax[1],split=True)
g1.set_title("Comptes d'épargne par travail", fontsize=15)
g1.set_xlabel("Type de compte d'épargne", fontsize=12)
g1.set_ylabel("Job", fontsize=12)

g = sns.boxplot(x="Saving accounts", y="Credit amount", data=df_cred
it, ax=ax[2],
            hue = "Risk",palette="hls")
g2.set_title("Comptes d'épargne par montant de crédit", fontsize=15)
g2.set_xlabel("Type de compte d'épargne", fontsize=12)
g2.set_ylabel("Montant du crédit (USD)", fontsize=12)

plt.subplots_adjust(hspace = 0.4,top = 0.9)

plt.show()
```

```
Description des comptes d'épargne de distribution par risque de remb
oursement:
Risk             bad  good
Saving accounts
little           217   386
moderate          34    69
quite rich        11    52
rich               6    42
```



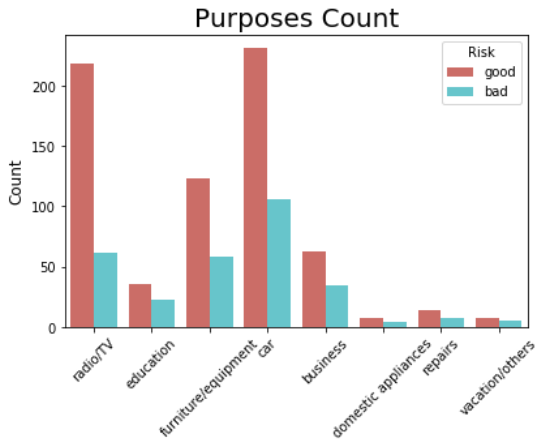


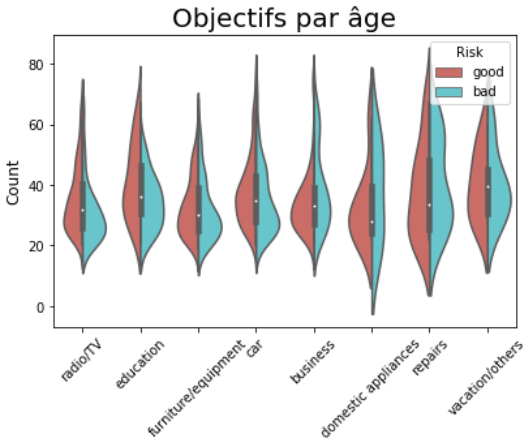


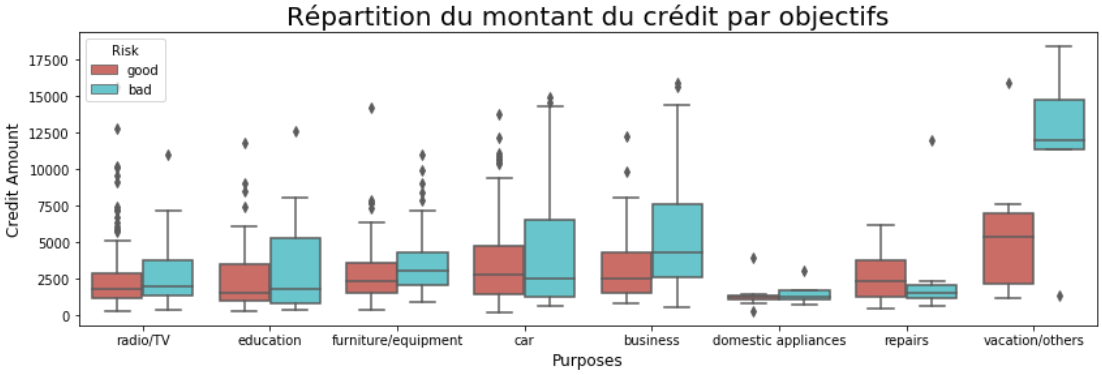

```
print("Values describe: ")
print(pd.crosstab(df_credit.Purpose, df_credit.Risk))

plt.figure(figsize = (14,12))

plt.subplot(221)
g = sns.countplot(x="Purpose", data=df_credit,
                  palette="hls", hue = "Risk")
g.set_xticklabels(g.get_xticklabels(),rotation=45)
g.set_xlabel("", fontsize=12)
g.set_ylabel("Count", fontsize=12)
g.set_title("Purposes Count", fontsize=20)

plt.subplot(222)
g1 = sns.violinplot(x="Purpose", y="Age", data=df_credit,
                    palette="hls", hue = "Risk",split=True)
g1.set_xticklabels(g1.get_xticklabels(),rotation=45)
g1.set_xlabel("", fontsize=12)
g1.set_ylabel("Count", fontsize=12)
```

```
g1.set_title("Objectifs par âge", fontsize=20)

plt.subplot(212)
g2 = sns.boxplot(x="Purpose", y="Credit amount", data=df_credit,
                 palette="hls", hue = "Risk")
g2.set_xlabel("Purposes", fontsize=12)
g2.set_ylabel("Credit Amount", fontsize=12)
g2.set_title("Répartition du montant du crédit par objectifs", fonts
ize=20)

plt.subplots_adjust(hspace = 0.6, top = 0.8)

plt.show()
```

```
Values describe:
Risk                   bad  good
Purpose
business                34    63
car                    106   231
domestic appliances      4     8
education               23    36
furniture/equipment     58   123
radio/TV                62   218
repairs                  8    14
vacation/others          5     7
```

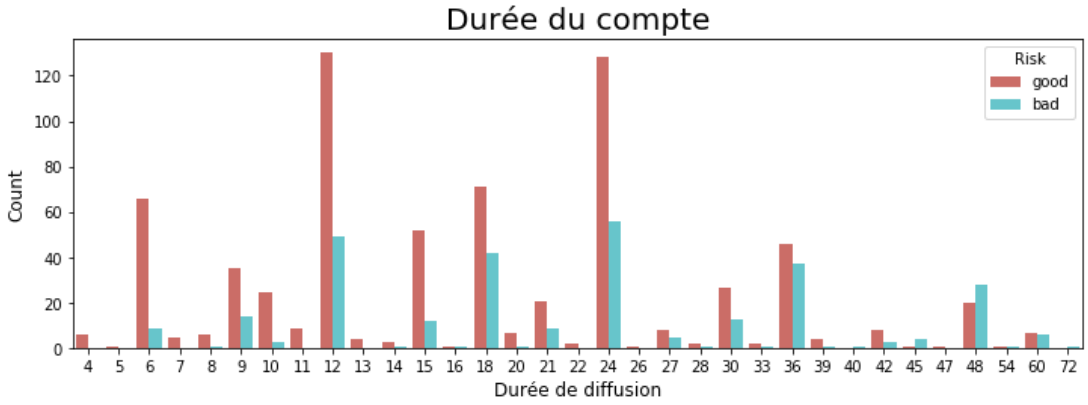


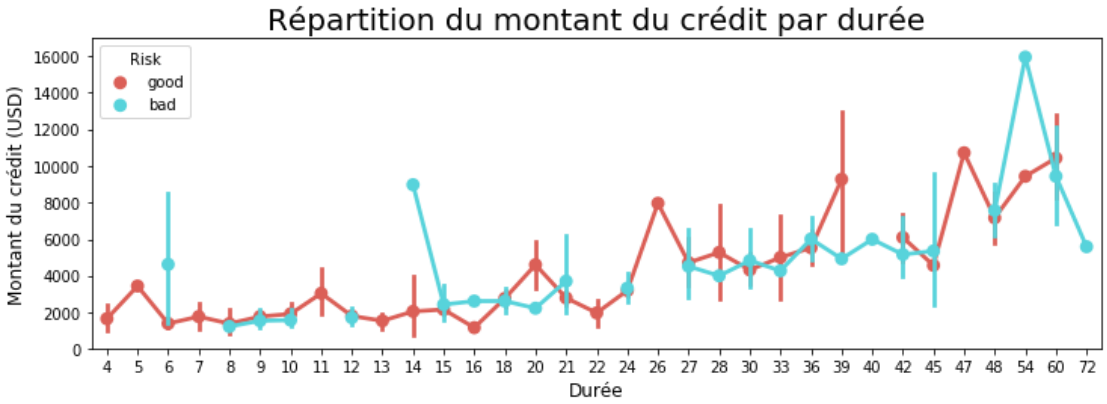


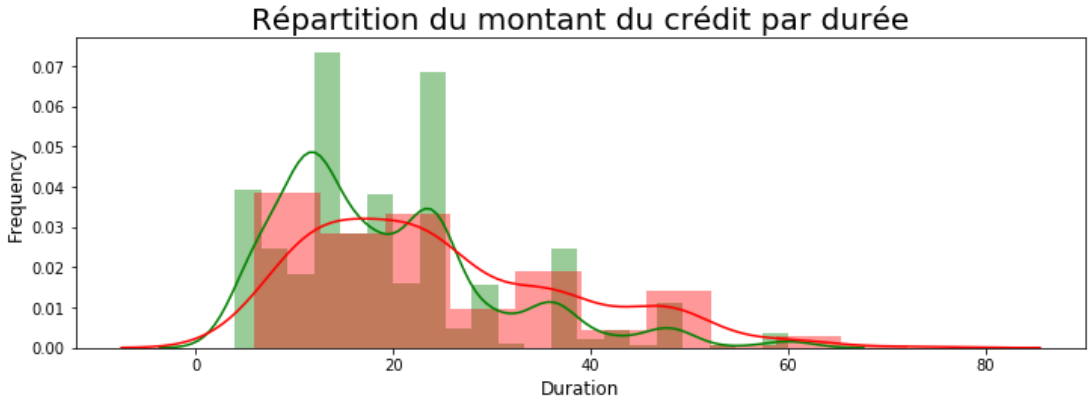


Durée de la distribution et densité des prêts

```
plt.figure(figsize = (12,14))

g= plt.subplot(311)
g = sns.countplot(x="Duration", data=df_credit,
              palette="hls",  hue = "Risk")
g.set_xlabel("Durée de diffusion", fontsize=12)
g.set_ylabel("Count", fontsize=12)
g.set_title("Durée du compte", fontsize=20)

g1 = plt.subplot(312)
g1 = sns.pointplot(x="Duration", y ="Credit amount",data=df_credit,
                   hue="Risk", palette="hls")
g1.set_xlabel("Durée", fontsize=12)
g1.set_ylabel("Montant du crédit (USD)", fontsize=12)
g1.set_title("Répartition du montant du crédit par durée", fontsize=
20)

g2 = plt.subplot(313)
g2 = sns.distplot(df_good["Duration"], color='g')
g2 = sns.distplot(df_bad["Duration"], color='r')
g2.set_xlabel("Duration", fontsize=12)
g2.set_ylabel("Frequency", fontsize=12)
g2.set_title("Répartition du montant du crédit par durée", fontsize=
20)

plt.subplots_adjust(wspace = 0.4, hspace = 0.4,top = 0.9)

plt.show()
```







Intéressant, nous pouvons dire que les plus grandes durées ont les montants les plus élevés. cette durée se situe entre [12 ~ 18 ~ 24] mois

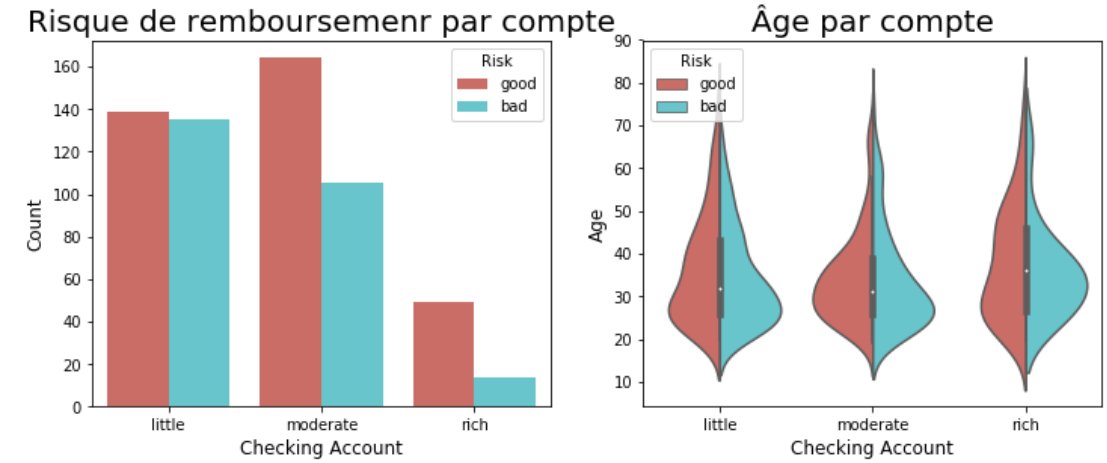
```
#First plot
trace0 = go.Bar(
    x = df_credit[df_credit["Risk"]== 'good']["Checking account"].va
lue_counts().index.values,
    y = df_credit[df_credit["Risk"]== 'good']["Checking account"].va
lue_counts().values,
    name='Bon dossier de prêt'

)

#Second plot
```

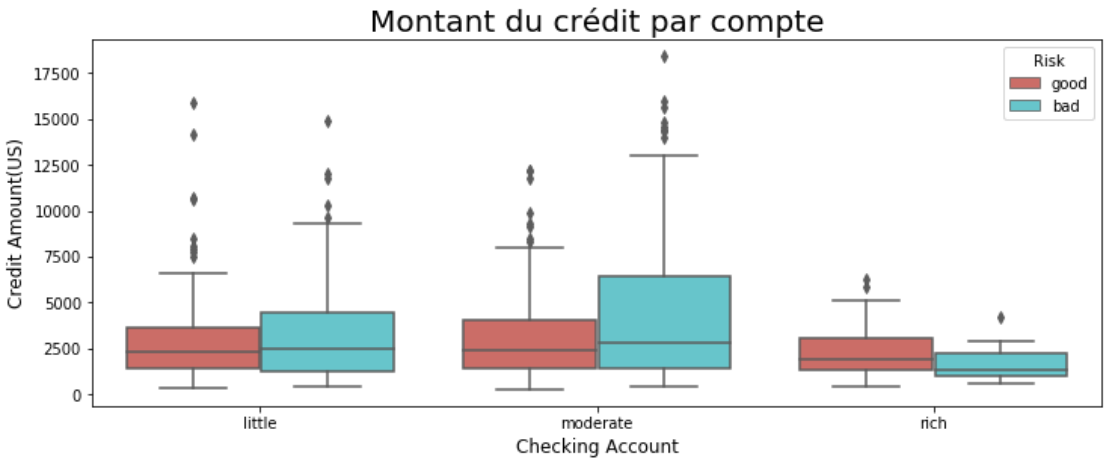
```
trace1 = go.Bar(
    x = df_credit[df_credit["Risk"]== 'bad']["Checking account"].val
ue_counts().index.values,
    y = df_credit[df_credit["Risk"]== 'bad']["Checking account"].val
ue_counts().values,
    name="Mauvais dossier de prêt"
)

data = [trace0, trace1]

layout = go.Layout(
    title='Vérification par compte',
    xaxis=dict(title='Vérification par nom des comptes'),
    yaxis=dict(title='Count'),
    barmode='group'
)


fig = go.Figure(data=data, layout=layout)

py.iplot(fig, filename = 'Age-ba', validate = False)
```

```
df_good = df_credit[df_credit["Risk"] == 'good']
df_bad = df_credit[df_credit["Risk"] == 'bad']

trace0 = go.Box(
    y=df_good["Credit amount"],
    x=df_good["Checking account"],
    name='Bon dossier',
    marker=dict(
        color='#3D9970'
    )
)

trace1 = go.Box(
    y=df_bad['Credit amount'],
    x=df_bad['Checking account'],
    name='Muvais dossier',
    marker=dict(
        color='#FF4136'
    )
)

data = [trace0, trace1]

layout = go.Layout(
    yaxis=dict(
        title='Contrôle par distribution'
    ),
    boxmode='group'
```

```
)
fig = go.Figure(data=data, layout=layout)

py.iplot(fig, filename='box-age-cat')

print("Valeurs totales de la variable manquante :")
print(df_credit.groupby("Checking account")["Checking account"].coun
t())

plt.figure(figsize = (12,10))

g = plt.subplot(221)
g = sns.countplot(x="Checking account", data=df_credit,
                  palette="hls", hue="Risk")
g.set_xlabel("Checking Account", fontsize=12)
g.set_ylabel("Count", fontsize=12)
g.set_title("Risque de remboursemenr par compte", fontsize=20)

g1 = plt.subplot(222)
g1 = sns.violinplot(x="Checking account", y="Age", data=df_credit, p
alette="hls", hue = "Risk",split=True)
g1.set_xlabel("Checking Account", fontsize=12)
g1.set_ylabel("Age", fontsize=12)
g1.set_title("Âge par compte", fontsize=20)

g2 = plt.subplot(212)
g2 = sns.boxplot(x="Checking account",y="Credit amount", data=df_cre
dit,hue='Risk',palette="hls")
g2.set_xlabel("Checking Account", fontsize=12)
g2.set_ylabel("Credit Amount(US)", fontsize=12)
g2.set_title("Montant du crédit par compte", fontsize=20)

plt.subplots_adjust(wspace = 0.2, hspace = 0.3, top = 0.9)

plt.show()
plt.show()
```

```
Valeurs totales de la variable manquante :
Checking account
little      274
moderate    269
rich         63
Name: Checking account, dtype: int64
```





Session crosstab et autres pour explorer nos données par une autre métrique un peu plus en profondeur

```
print(pd.crosstab(df_credit.Sex, df_credit.Job))
```

```
Job       0    1    2    3
Sex
female   12   64  197   37
male     10  136  433  111
```

```
plt.figure(figsize = (10,6))

g = sns.violinplot(x="Housing",y="Job",data=df_credit,
                   hue="Risk", palette="hls",split=True)
g.set_xlabel("Housing", fontsize=12)
g.set_ylabel("Job", fontsize=12)
g.set_title("Housing x Job - Dist", fontsize=20)

plt.show()
```


```
print(pd.crosstab(df_credit["Checking account"],df_credit.Sex))
```

```
Sex               female  male
Checking account
little                88   186
moderate              86   183
rich                  20    43
```

```
date_int = ["Purpose", 'Sex']
cm = sns.light_palette("green", as_cmap=True)
pd.crosstab(df_credit[date_int[0]], df_credit[date_int[1]]).style.ba
ckground_gradient(cmap = cm)
```

```
<pandas.io.formats.style.Styler at 0x20f7b968198>
```

```
date_int = ["Purpose", 'Sex']
cm = sns.light_palette("green", as_cmap=True)
pd.crosstab(df_credit[date_int[0]], df_credit[date_int[1]]).style.ba
ckground_gradient(cmap = cm)
```

```
<pandas.io.formats.style.Styler at 0x20f7c8d6fd0>
```

Recherche du total des valeurs dans chaque caractéristique

```
print("Purpose : ",df_credit.Purpose.unique())
print("Sex : ",df_credit.Sex.unique())
print("Housing : ",df_credit.Housing.unique())
print("Saving accounts : ",df_credit['Saving accounts'].unique())
print("Risk : ",df_credit['Risk'].unique())
```

```
print("Checking account : ",df_credit['Checking account'].unique())
print("Aget_cat : ",df_credit['Age_cat'].unique())
```

```
Purpose :  ['radio/TV' 'education' 'furniture/equipment' 'car' 'busi
ness'
 'domestic appliances' 'repairs' 'vacation/others']
Sex :  ['male' 'female']
Housing :  ['own' 'free' 'rent']
Saving accounts :  [nan 'little' 'quite rich' 'rich' 'moderate']
Risk :  ['good' 'bad']
Checking account :  ['little' 'moderate' nan 'rich']
Aget_cat :  [Senior, Student, Adult, Young]
Categories (4, object): [Student < Young < Adult < Senior]
```

Faisons de l'ingénierie de fonctionnalités sur ces valeurs et créons des variables factices des valeurs

```
def one_hot_encoder(df, nan_as_category = False):
    original_columns = list(df.columns)
    categorical_columns = [col for col in df.columns if df[col].dtyp
e == 'object']
    df = pd.get_dummies(df, columns= categorical_columns, dummy_na=
nan_as_category, drop_first=True)
    new_columns = [c for c in df.columns if c not in original_column
s]
    return df, new_columns
```

Transformer les données en variables factices

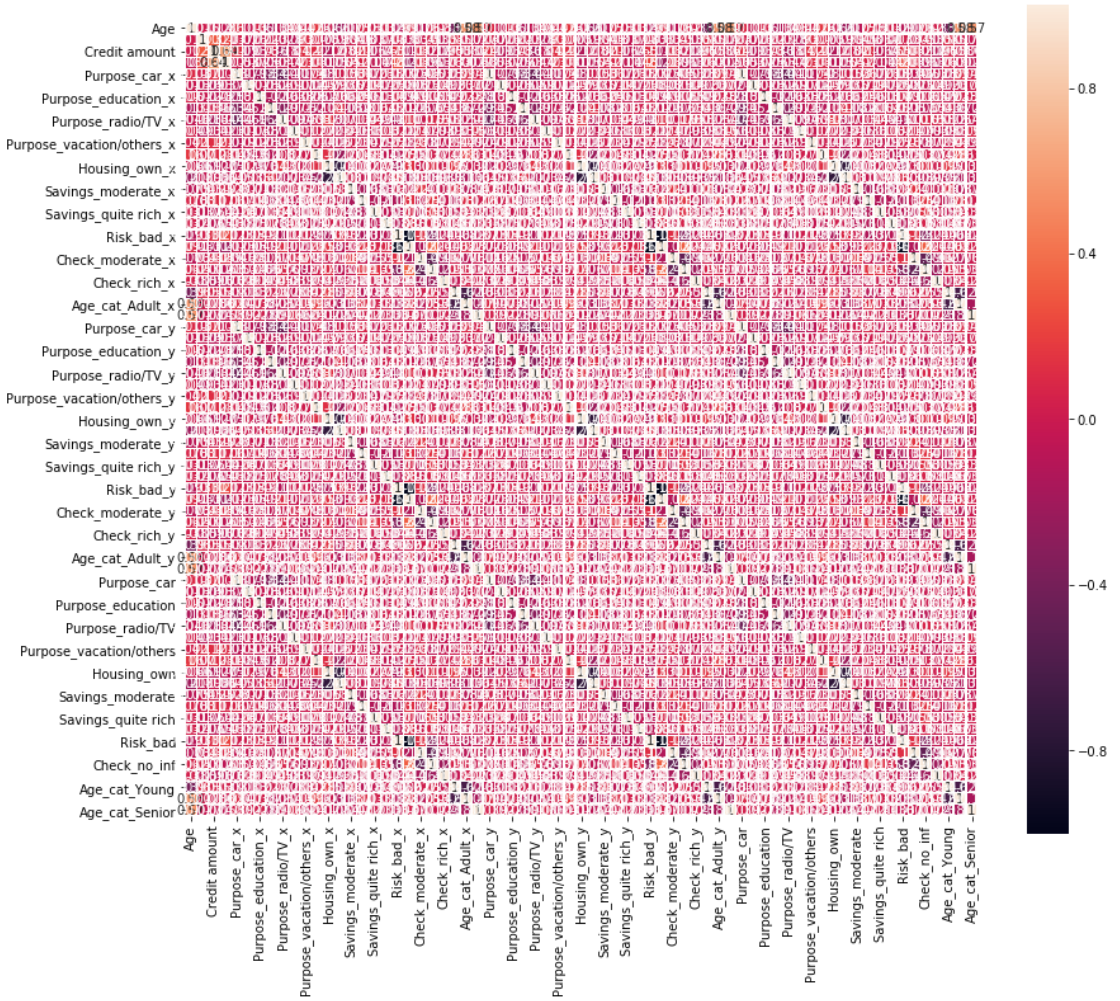
```
df_credit['Saving accounts'] = df_credit['Saving accounts'].fillna('
no_inf')
df_credit['Checking account'] = df_credit['Checking account'].fillna
('no_inf')

#Purpose to Dummies Variable
df_credit = df_credit.merge(pd.get_dummies(df_credit.Purpose, drop_f
irst=True, prefix='Purpose'), left_index=True, right_index=True)
#Sex feature in dummies
df_credit = df_credit.merge(pd.get_dummies(df_credit.Sex, drop_first
=True, prefix='Sex'), left_index=True, right_index=True)
# Housing get dummies
df_credit = df_credit.merge(pd.get_dummies(df_credit.Housing, drop_f
irst=True, prefix='Housing'), left_index=True, right_index=True)
# Housing get Saving Accounts
df_credit = df_credit.merge(pd.get_dummies(df_credit["Saving account
s"], drop_first=True, prefix='Savings'), left_index=True, right_inde
x=True)
# Housing get Risk
df_credit = df_credit.merge(pd.get_dummies(df_credit.Risk, prefix='R
```

```
isk'), left_index=True, right_index=True)
# Housing get Checking Account
df_credit = df_credit.merge(pd.get_dummies(df_credit["Checking accou
nt"], drop_first=True, prefix='Check'), left_index=True, right_index
=True)
# Housing get Age categorical
df_credit = df_credit.merge(pd.get_dummies(df_credit["Age_cat"], dro
p_first=True, prefix='Age_cat'), left_index=True, right_index=True)

#Excluding the missing columns
del df_credit["Saving accounts"]
del df_credit["Checking account"]
del df_credit["Purpose"]
del df_credit["Sex"]
del df_credit["Housing"]
del df_credit["Age_cat"]
del df_credit["Risk"]
del df_credit['Risk_good']
```

IV.5. Correlation:

- Recherche de la corrélation entre les données Regardons la corrélation des données

```
plt.figure(figsize=(14,12))
sns.heatmap(df_credit.astype(float).corr(),linewidths=0.1,vmax=1.0,
            square=True,  linecolor='white', annot=True)
plt.show()
```



IV.6. Prétraitement :

- Importation de bibliothèques ML
- Réglage des variables X et y à la prédiction
- Fractionnement des données

```
from sklearn.model_selection import train_test_split, KFold, cross_v
al_score # to split the data
from sklearn.metrics import accuracy_score, confusion_matrix, classi
fication_report, fbeta_score #To evaluate our model

from sklearn.model_selection import GridSearchCV

# Algorithmns models to be compared
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
```

```
from sklearn.svm import SVC
from xgboost import XGBClassifier

df_credit['Credit amount'] = np.log(df_credit['Credit amount'])

### Creating the X and y variables
X = df_credit.drop('Risk_bad', 1).values
y = df_credit["Risk_bad"].values

# Spliting X and y into train and test version
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
= 0.25, random_state=42)

df_credit.drop('Risk_bad', 1)
```

	Age	Job	Credit amount	Duration	Purpose_car_x	\
0	67	2	1.954998	6	0	
1	22	2	2.162324	48	0	
2	49	1	2.034416	12	0	
3	45	2	2.194146	42	0	
4	53	2	2.138989	24	1	
5	35	1	2.209490	36	0	
6	53	2	2.073146	24	0	
7	35	3	2.179989	36	1	
8	61	1	2.082667	12	0	
9	28	3	2.147443	30	1	
10	25	2	1.969385	12	1	
11	24	2	2.124442	48	0	
12	22	2	1.995641	12	0	
13	60	1	1.958579	24	1	

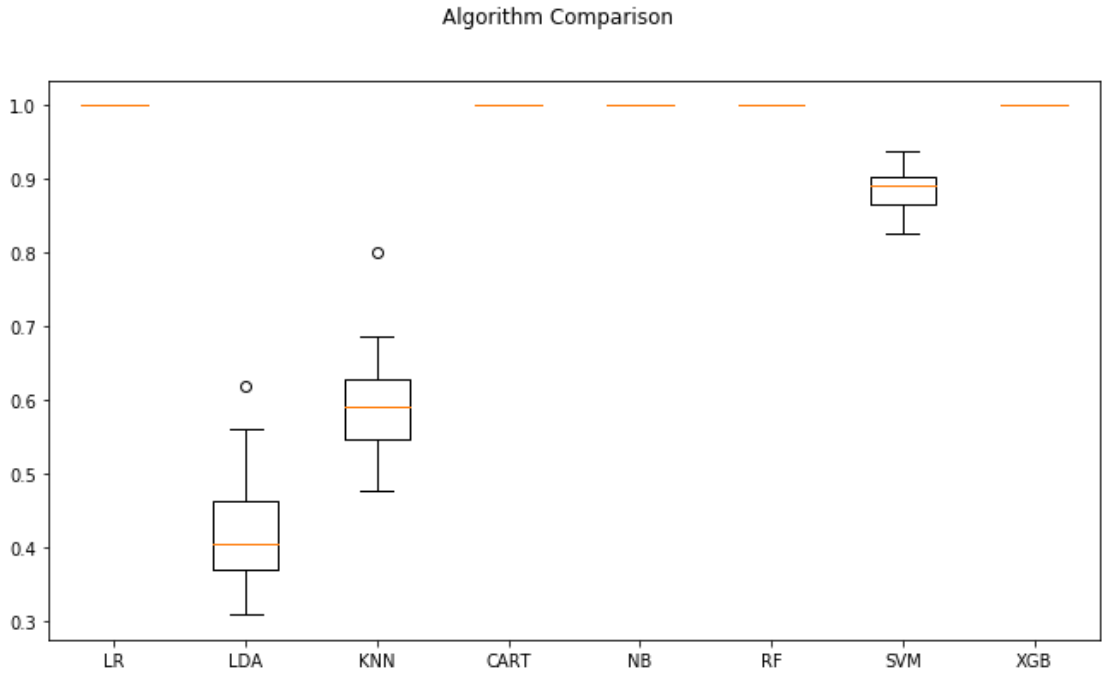
…

[1000 rows x 68 columns]

```
# implementation des différents modles à combiner
seed = 7

# preparation des modèles
models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('RF', RandomForestClassifier()))
models.append(('SVM', SVC(gamma='auto')))
models.append(('XGB', XGBClassifier()))

# evaluation de chaque modèle à tour de rôle
```

```
results = []
names = []
scoring = 'recall'

for name, model in models:
        kfold = KFold(n_splits=10, random_state=seed)
        cv_results = cross_val_score(model, X_train, y_train, cv=kfo
ld, scoring=scoring)
        results.append(cv_results)
        names.append(name)
        msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.s
td())
        print(msg)

# boxplot algorithm comparison
fig = plt.figure(figsize=(11,6))
fig.suptitle('Algorithm Comparison')
ax = fig.add_subplot(111)
plt.boxplot(results)
ax.set_xticklabels(names)
plt.show()
```

```
LR: 1.000000 (0.000000)
LDA: 0.428815 (0.093162)
KNN: 0.604042 (0.085142)
CART: 1.000000 (0.000000)
NB: 1.000000 (0.000000)
RF: 1.000000 (0.000000)

SVM: 0.887861 (0.031029)
XGB: 1.000000 (0.000000)
```



Nous pouvons constater que, presque tous les modèles affichent une faible valeur de retour

Nous pouvons observer que nos meilleurs résultats ont été obtenus avec le modèle CART, NB et XGBoost. Nous allons implémenter quelques modèles et essayer de faire un réglage simple sur eux

IV.7. Modèle 1 :

- Utilisation de la forêt aléatoire pour prédire le pointage de crédit
- Certains paramètres de validation

```
rf = RandomForestClassifier(max_depth=None, max_features=10, n_estim
ators=15, random_state=2)

#trainning with the best params
rf.fit(X_train, y_train)

RandomForestClassifier(bootstrap=True, class_weight=None, criterion=
'gini',
            max_depth=None, max_features=10, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=15, n_jobs=No
ne,
            oob_score=False, random_state=2, verbose=0, warm_start=F
alse)
```

```
#Tester le modèle
#Prédire avec notre modèle
y_pred = rf.predict(X_test)

# Vérification les résultats obtenus
print(accuracy_score(y_test,y_pred))
print("\n")
print(confusion_matrix(y_test, y_pred))
print("\n")
print(fbeta_score(y_test, y_pred, beta=2))
```

```
1.0   = 100%  pour l'entrenement
```

Matrice de confision

```
[[178   0]
[  0  72]]


1.0
```

Des résultats très intéressants! nous pouvons constater que tous les individus ont été bien classifiés. Aucun cas des faux réjet. la précision est donc à 100%. 178 individus ont été classifies comme ayant des bons dossiers de prêt et 72 ayant des mauvais dossier. Ainsi aucun individu a été mal classifié.

VI.7.Model 2 : le modèle gaussien

```
from sklearn.utils import resample
from sklearn.metrics import roc_curve

# Criando o classificador logreg
GNB = GaussianNB()

# Fitting with train data
model = GNB.fit(X_train, y_train)

# Printing the Training Score
print("Training score data: ")
print(model.score(X_train, y_train))
```

```
Training score data:
1.0
```

Le score de la base d'entrenement est de 100%

```
y_pred = model.predict(X_test)

print(accuracy_score(y_test,y_pred))
print("\n")
```

```
print(confusion_matrix(y_test, y_pred))
print("\n")
print(classification_report(y_test, y_pred))
```

```
1.0 = 100%
```

Matrice de confision

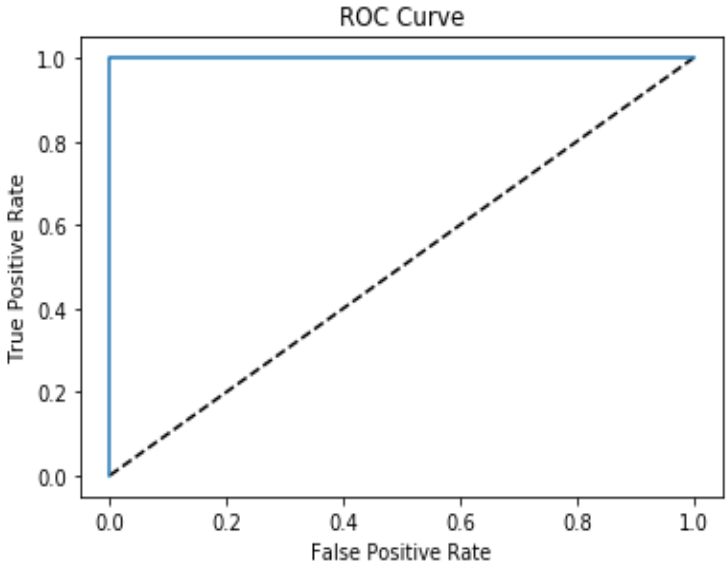
```
[[178   0]
[  0  72]]
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       178
           1       1.00      1.00      1.00        72

   micro avg       1.00      1.00      1.00       250
   macro avg       1.00      1.00      1.00       250
weighted avg       1.00      1.00      1.00       250
```

La prédiction est encore meilleur avec le modèle gaussien, nous avons obtenu comme dans le cas précdent 178 bons dossier de prêt et 72 mauvais dossiers de prêt. L'exactitude mathématique est de 100%.

Vérifions la courbe ROC

```
y_pred_prob = model.predict_proba(X_test)[:,1]

fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
# Tracer la courbe ROC
plt.plot([0, 1], [0, 1], 'k--')
plt.plot(fpr, tpr)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('le modèle ROC Curve')
plt.show()

from sklearn.metrics import roc_auc_score
y_pred_prob = model.predict_proba(X_test)[:,1]
print(f"ROC_AUC score: {roc_auc_score(y_test, y_pred_prob)}")
```



ROC_AUC score: 100%

```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from sklearn.pipeline import FeatureUnion
from sklearn.linear_model import LogisticRegression
from sklearn.decomposition import PCA
from sklearn.feature_selection import SelectKBest

features = []
features.append(('pca', PCA(n_components=2)))
features.append(('select_best', SelectKBest(k=6)))
feature_union = FeatureUnion(features)
# create pipeline
estimators = []
estimators.append(('feature_union', feature_union))
estimators.append(('logistic', GaussianNB()))
model = Pipeline(estimators)
# evaluate pipeline
seed = 7
kfold = KFold(n_splits=10, random_state=seed)
results = cross_val_score(model, X_train, y_train, cv=kfold)
print(results.mean())
```

```
1.0
```

```
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print(accuracy_score(y_test,y_pred))
print("\n")
print(confusion_matrix(y_test, y_pred))
```

```
print("\n")
print(fbeta_score(y_test, y_pred, beta=2))
```

```
0.72 = 72%
```

Matrice de confision

```
[[149  29]
 [ 41  31]]
```

Nous constatons par contre que les individus ont été bien classifié avec 41 cas des fau x réjet pour les bons dossiers de prêt et 29 cas pour les mauvais dossiers. la précision est donc à 72%.

```
0.44540229885057464
```

IV.8.Implémentation d'un modèle pipeline

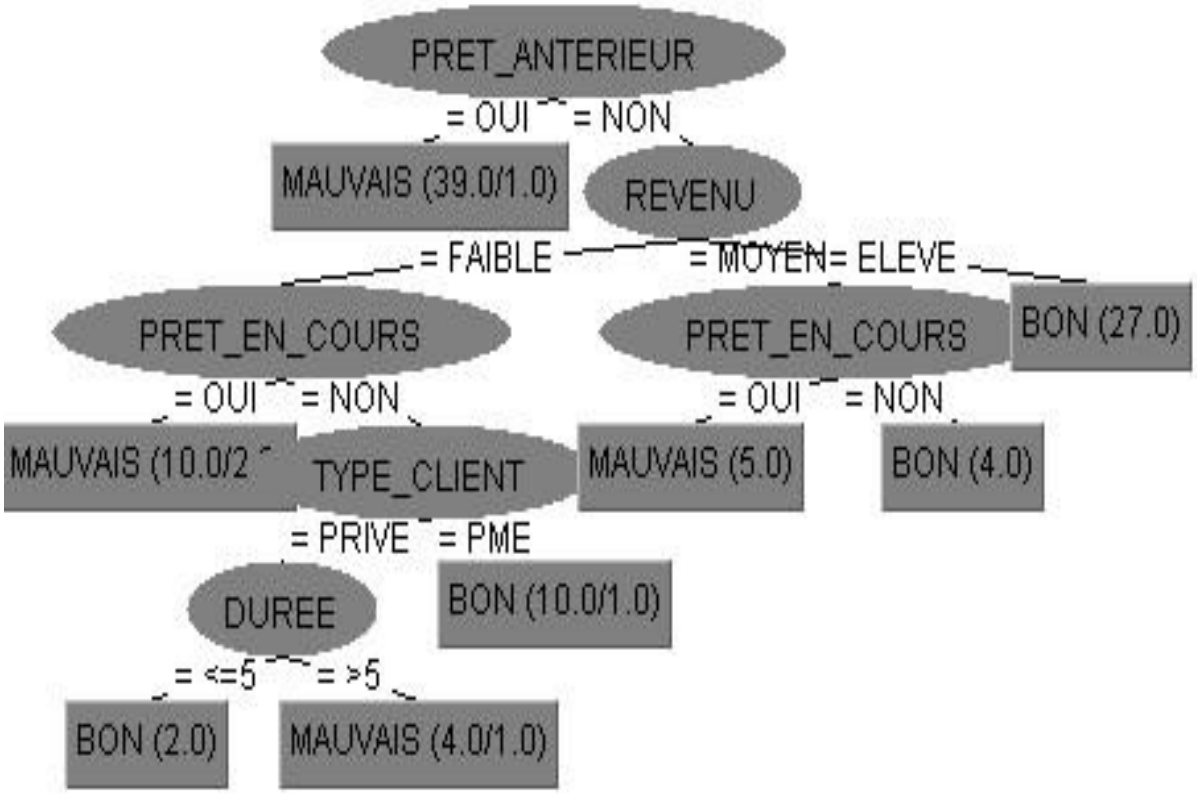
```
#Définition des hyper paramètres


param_test1 = {
 'max_depth':[3,5,6,10],
 'min_child_weight':[3,5,10],
 'gamma':[0.0, 0.1, 0.2, 0.3, 0.4],
# 'reg_alpha':[1e-5, 1e-2, 0.1, 1, 10],
 'subsample':[i/100.0 for i in range(75,90,5)],
 'colsample_bytree':[i/100.0 for i in range(75,90,5)]
}
```

Création du classifieur CART : arbres de decision

```
model_xg = CARTClassifier(random_state=2)

grid_search = GridSearchCV(model_xg, param_grid=param_test1, cv=5, s
coring='recall')
grid_search.fit(X_train, y_train)
```

```
GridSearchCV(cv=5, error_score='raise-deprecating',
       estimator=XGBClassifier(base_score=None, booster=None, callba
cks=None,
       colsample_bylevel=None, colsample_bynode=None,
       colsample_bytree=None, early_stopping_rounds=None,
       enable_categorical=False, eval_metric=None, gamma=None, gpu_i
d=None,
       grow_policy=None, importance_type=...,
       tree_method=None, use_label_encoder=False, validate_parameter
s=None,
```

```
       verbosity=None),
       fit_params=None, iid='warn', n_jobs=None,
       param_grid={'max_depth': [3, 5, 6, 10], 'min_child_weight': [
3, 5, 10], 'gamma': [0.0, 0.1, 0.2, 0.3, 0.4], 'subsample': [0.75, 0
.8, 0.85], 'colsample_bytree': [0.75, 0.8, 0.85]},
       pre_dispatch='2*n_jobs', refit=True, return_train_score='warn
',
       scoring='recall', verbose=0)

grid_search.best_params_

{'colsample_bytree': 0.75,
 'gamma': 0.0,
 'max_depth': 3,
 'min_child_weight': 3,
 'subsample': 0.75}
```



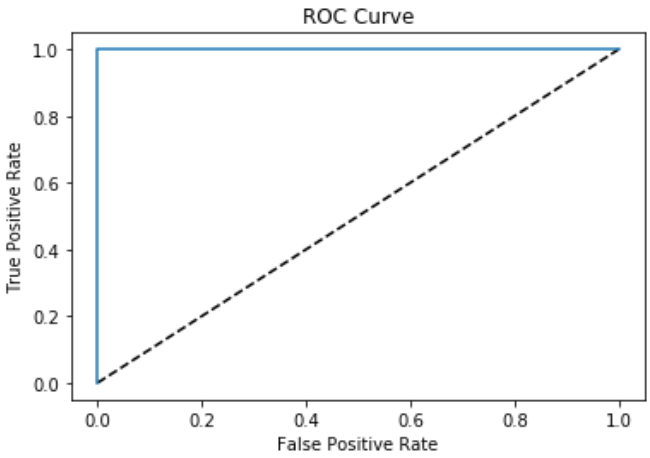
La figure ci – dessus nous présente l'arbre de décision dans sa globalité. Après exploration des données d'apprentissage, nous remarquons que nous avons la facilité de la décision qui du moins sera optimale, réfléchie et rationnelle. Nous pouvons à présent faire le classement d'un nouvel individu à partir du dossier que celui-ci va présenter.

Notre classifieur

```
y_pred = grid_search.predict(X_test)

# Verificaar os resultados obtidos
print(accuracy_score(y_test,y_pred))
print("\n")
print(confusion_matrix(y_test, y_pred))
from sklearn.metrics import roc_auc_score
y_pred_prob = grid_search.predict_proba(X_test)[:,1]


print(f"ROC_AUC score: {roc_auc_score(y_test, y_pred_prob)}")


# Generate ROC curve values: fpr, tpr, thresholds
fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)

# Plot ROC curve
plt.plot([0, 1], [0, 1], 'k--')
plt.plot(fpr, tpr)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.show()


1.0 = 100%
```

Matrice de confision

```
[[178   0]
 [  0  72]]


ROC_AUC score: 1.0
```



Tester l'anti-classification

```
# Tester l'anti-classification sur les attributs protégés : age
# L'âge correspond à age_cat_young (idx 21) / age_cat_adult (idx 22)
/ age_cat_senior (idx 23)
# et le genre correspond à idx 11

# Tout d'abord, obtenez la prédiction du modèle avec age_cat_young d
e tous les participants = 1.

X_test_with_age = X_test
for participant in X_test_with_age:
    participant[21] = 1
    participant[22] = 0
    participant[23] = 0
model_pred_with_young = grid_search.predict(X_test_with_age)


# Deuxièmement, nous obtenons la prédiction du modèle avec age_cat_a
dult de tous les participants = 1.

for participant in X_test_with_age:
    participant[21] = 0
    participant[22] = 1
    participant[23] = 0
model_pred_with_adult = grid_search.predict(X_test_with_age)

# Enfin, obtenez la prédiction du modèle avec tous les participants
age_cat_senior = 1.
for participant in X_test_with_age:
    participant[21] = 0
    participant[22] = 0
    participant[23] = 1
model_pred_with_senior = grid_search.predict(X_test_with_age)

def check_anti_classcification_age(model_pred_with_young, model_pred
_with_adult, model_pred_with_senior):
    inconsistenty_count = 0
    num_participants = len(model_pred_with_young)
    for pred_young, pred_adult, pred_senior in zip(model_pred_with_y
oung, model_pred_with_adult, model_pred_with_senior):
        if pred_young == pred_adult == pred_senior:
            pass
        else:
            inconsistenty_count+=1
    inconsistency_percentage = (inconsistenty_count / num_participan
ts) * 100
    print(f"Pourcentage d'incohérence : {inconsistency_percentage}")
```

```
check_anti_classcification_age(model_pred_with_young, model_pred_wit
h_adult, model_pred_with_senior)
```

```
Pourcentage d'incohérence : 0.0
```

Nous pouvons voir que aucune incohérence n'est possible

```
# Test de l'anti-classification sur l'attribut "sexe". Il correspond
à index=11.
# Tout d'abord, nous avons obtenu la prédiction du modèle avec sex_m
ale de tous les participants = 1.

X_test_with_gender = X_test
for participant in X_test_with_gender:
    participant[11] = 1
model_pred_with_male = grid_search.predict(X_test_with_gender)

# Deuxièmement, nous avons obtenu la prédiction du modèle avec sex_m
ale de tous les participants = 0.

for participant in X_test_with_gender:
    participant[11] = 0
model_pred_with_female = grid_search.predict(X_test_with_gender)

def check_anti_classcification_gender(model_pred_with_male, model_pr
ed_with_female):
    inconsistenty_count = 0
    num_participants = len(model_pred_with_male)
    for pred_male, pred_female in zip(model_pred_with_male, model_pr
ed_with_female):
        if pred_male == pred_female:
            pass
        else:
            inconsistenty_count+=1
    inconsistency_percentage = (inconsistenty_count / num_participan
ts) * 100
    print(f"Pourcentage d'incohérence: {inconsistency_percentage}")

check_anti_classcification_gender(model_pred_with_male, model_pred_w
ith_female)
```

```
Pourcentage d'incohérence: 0.0
```

Test de l'équité du groupe

```
X = df_credit.drop('Risk_bad', 1).values
y = df_credit["Risk_bad"].values

# Diviser X et y en version train et test
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
= 0.25, random_state=42)

# Tout d'abord, prenons l'équité de groupe sur les attributs protégé
s = âge.
X_test_idx_young = []
X_test_idx_adult = []
X_test_idx_senior = []

# Notez que certaines des valeurs ont une entrée (0,0,0) pour (Age_c
at_Young, Age_cat_Adult, Age_cat_Senior)
for idx, elem in enumerate(X_test):
    if elem[21] == 1:
        X_test_idx_young.append(idx)
    elif elem[22] == 1:
        X_test_idx_adult.append(idx)
    elif elem[23] == 1:
        X_test_idx_senior.append(idx)
    else:
        print(f"Aucun d'eux n'en est un ! index: {idx}")
```

```
Aucun d'eux n'en est un ! index: 0
Aucun d'eux n'en est un ! index: 5
Aucun d'eux n'en est un ! index: 7
Aucun d'eux n'en est un ! index: 13
Aucun d'eux n'en est un ! index: 14
Aucun d'eux n'en est un ! index: 15
Aucun d'eux n'en est un ! index: 18
Aucun d'eux n'en est un …
```

```
# Notez que certains

def check_group_fairness_age(X_test_idx_young, X_test_idx_adult, X_t
est_idx_senior, model_pred):
    positive_count_young = 0
    positive_count_adult = 0
    positive_count_senior = 0
    for idx, elem in enumerate(model_pred):
        if idx in X_test_idx_young:
            if elem == 1:
                positive_count_young += 1
        elif idx in X_test_idx_adult:
            if elem == 1:
                positive_count_adult += 1
        elif idx in X_test_idx_senior:
            if elem == 1:
                positive_count_senior += 1

    positive_rate_young = positive_count_young / len(X_test_idx_youn
g)
```

```
    positive_rate_adult = positive_count_adult / len(X_test_idx_adul
t)
    positive_rate_senior = positive_count_senior / len(X_test_idx_se
nior)

    print(f"Taux des bons dossier des jeunes : {positive_rate_young}
")
    print(f"Taux des bons dossier des adultes : {positive_rate_adult
}")
    print(f"Taux des bons dossier des personne âgée : {positive_rate
_senior}")

model_pred = grid_search.predict(X_test)
check_group_fairness_age(X_test_idx_young, X_test_idx_adult, X_test_
idx_senior, model_pred)
```

```
-------------------------------------------------------------

Taux des bons dossier des jeunes : 0.09278350515463918
Taux des bons dossier des adultes : 0.2777777777777778
Taux des bons dossier des personne âgée : 0.3695652173913043

-------------------------------------------------------------
```

```
# Maintenant, nous prenons l'équité de groupe sur les attributs prot
égés = sexe.

X_test_idx_male = []
X_test_idx_female = []

# Note that some of the values have (0,0,0) entry for (Age_cat_Young
, Age_cat_Adult, Age_cat_Senior)
for idx, elem in enumerate(X_test):
    if elem[11] == 1:
        X_test_idx_male.append(idx)
    else:
        X_test_idx_female.append(idx)

# Notez que certains

def check_group_fairness_gender(X_test_idx_male, X_test_idx_female,
model_pred):
    positive_count_male = 0
    positive_count_female = 0
    for idx, elem in enumerate(model_pred):
        if idx in X_test_idx_male:
            if elem == 1:
                positive_count_male += 1
        elif idx in X_test_idx_female:
            if elem == 1:
                positive_count_female += 1
```

```
        else:
            pass

    positive_rate_male = positive_count_male / len(X_test_idx_male)
    positive_rate_female = positive_count_female / len(X_test_idx_fe
male)

    print(f"Taux des bons dossier des hommes : {positive_rate_male *
100} %")
    print(f"Taux des bons dossier des femmes : {positive_rate_female
* 100} %")

model_pred = grid_search.predict(X_test)
check_group_fairness_gender(X_test_idx_male, X_test_idx_female, mode
l_pred)
```

```
----------------------------------------------------------------

Taux des bons dossier des hommes : 28.24858757062147 %
Taux des bons dossier des femmes : 30.136986301369863 %

----------------------------------------------------------------
```

Test de séparation

```
# Dans un premier temps, nous allons tester la séparation sur les at
tributs protégés = age
# P [Y' = 1 | Y = 0, A = une ] = P [Y' = 1 | Y = 0, A = b] (parité F
PR)
# P [Y' = 0 | Y = 1, A = une ] = P [Y' = 0 | Y = 1, A = b] (parité F
NR)

def check_seperation_age(X_test_idx_young, X_test_idx_adult, X_test_
idx_senior, y_test, model_pred):
    # Vérifier la parité
    label_false_young_count = 0
    label_false_adult_count = 0
    label_false_senior_count = 0

    fp_count_young = 0
    fp_count_adult = 0
    fp_count_senior = 0

    for idx, (y, y_hat) in enumerate(zip(y_test, model_pred)):
        if idx in X_test_idx_young:
            if y == 0:
                label_false_young_count += 1
                if y_hat == 1:
                    fp_count_young +=1
        elif idx in X_test_idx_adult:
```

```
            if y == 0:
                label_false_adult_count += 1
                if y_hat == 1:
                    fp_count_adult +=1
        elif idx in X_test_idx_senior:
            if y == 0:
                label_false_senior_count += 1
                if y_hat == 1:
                    fp_count_senior +=1
        else:
            pass
    if label_false_young_count!=0 and label_false_adult_count!=0 and
label_false_senior_count!=0:
        print(f"Taux d'incohérence sur les jeunes: {fp_count_young /
label_false_young_count},Taux d'incohérence sur les adultes : {fp_co
unt_adult / label_false_adult_count}, Taux d'incohérence sur les per
sonnes âgés : {fp_count_senior / label_false_senior_count}")

    # Check FNR Parity
    label_true_young_count = 0
    label_true_adult_count = 0
    label_true_senior_count = 0

    fn_count_young = 0
    fn_count_adult = 0
    fn_count_senior = 0

    for idx, (y, y_hat) in enumerate(zip(y_test, model_pred)):
        if idx in X_test_idx_young:
            if y == 1:
                label_true_young_count += 1
                if y_hat == 0:
                    fn_count_young +=1
        elif idx in X_test_idx_adult:
            if y == 1:
                label_true_adult_count += 1
                if y_hat == 0:
                    fn_count_adult +=1
        elif idx in X_test_idx_senior:
            if y == 1:
                label_true_senior_count += 1
                if y_hat == 0:
                    fn_count_senior +=1
        else:
            pass
    print(label_true_young_count)
    print(label_true_adult_count)
    print(label_true_senior_count)
```

```
    print(fn_count_young)
    print(fn_count_adult)
    print(fn_count_senior)

    if label_true_young_count!=0 and label_true_adult_count!=0 and l
abel_true_senior_count!=0:
        print(f"Taux d'incohérence sur les jeunes: {fn_count_young /
label_true_young_count},Taux d'incohérence sur les adultes: {fn_coun
t_adult / label_true_adult_count}, Taux d'incohérence sur les person
nes âgés : {fn_count_senior / label_true_senior_count}")

model_pred = grid_search.predict(X_test)
check_seperation_age(X_test_idx_young, X_test_idx_adult, X_test_idx_
senior, y_test, model_pred)

-------------------------------------------------

Taux d'incohérence sur les jeunes: 0.0,
Taux d'incohérence sur les adultes: 0.0,
Taux d'incohérence sur les personnes âgés: 0.0
9
5
17
0
0
0
Taux d'incohérence sur les jeunes: 0.0,
Taux d'incohérence sur les adultes: 0.0,
Taux d'incohérence sur les personnes âgés: 0.0
```

De ce fait, nous pouvons dire que l'incohérence n'est pas du tout possible dans les clas sements qu'éffectuent nos classifieurs.

```
# Dans un premier temps, nous allons tester la séparation sur les
attributs protégés = age
# P [Y' = 1 | Y = 0, A = une ] = P [Y' = 1 | Y = 0, A = b] (parité
FPR)
# P [Y' = 0 | Y = 1, A = une ] = P [Y' = 0 | Y = 1, A = b] (parité
FNR)

def check_seperation_gender(X_test_idx_male, X_test_idx_female,
y_test, model_pred):
   # Vérifier la parité
    label_false_male_count = 0
    label_false_female_count = 0

    fp_count_male = 0
    fp_count_female = 0

    for idx, (y, y_hat) in enumerate(zip(y_test, model_pred)):
        if idx in X_test_idx_male:
            if y == 0:
                label_false_male_count += 1
                if y_hat == 1:
                    fp_count_male +=1
        elif idx in X_test_idx_female:
            if y == 0:
                label_false_female_count += 1
                if y_hat == 1:
                    fp_count_female +=1
        else:
            pass
print(f"Taux d'incohérence sur les hommes : {fp_count_male /
label_false_male_count},Taux d'incohérence sur les femmes:
{fp_count_female / label_false_female_count}")
```

Équité améliorée - anti-classification

```
# Comme attribut protégé, nous choisissons "gender".
# suppression de l'attribut protegé pour anti-classification

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
= 0.25, random_state=42)
# Éliminer les attributs protégés
# index= 11 pour le sexe

X_train_eliminated = np.delete(X_train, [11], axis=1)
X_test_eliminated = np.delete(X_test, [11], axis=1)

param_test1 = {
 'max_depth':[3,5,6,10],
 'min_child_weight':[3,5,10],
```

```
 'gamma':[0.0, 0.1, 0.2, 0.3, 0.4],
# 'reg_alpha':[1e-5, 1e-2, 0.1, 1, 10],
 'sous-échantillon':[i/100.0 for i in range(75,90,5)],
 'colsample_bytree':[i/100.0 for i in range(75,90,5)]
}

#Création du classifieur
model_xg = XGBClassifier(random_state=2)

grid_search = GridSearchCV(model_xg, param_grid=param_test1, cv=5, s
coring='recall')
grid_search.fit(X_train_eliminated, y_train)
```

```
1.0
[[178   0]
 [  0  72]]
ROC_AUC score: 1.0
```

```
# Avant modification.- équité de groupe
check_group_fairness_gender(X_test_idx_male, X_test_idx_female, y_pr
ed)
```

```
Taux des bons dossier des hommes : 28.24858757062147 %
Taux des bons dossier des femmes : 30.136986301369863 %
```

```
def check_group_fairness_gender_with_return(X_test_idx_male, X_test_
idx_female, model_pred):
    positive_count_male = 0
    positive_count_female = 0
    for idx, elem in enumerate(model_pred):
        if idx in X_test_idx_male:
            if elem == 1:
                positive_count_male += 1
        elif idx in X_test_idx_female:
            if elem == 1:
                positive_count_female += 1
        else:
            pass

    positive_rate_male = positive_count_male / len(X_test_idx_male)
    positive_rate_female = positive_count_female / len(X_test_idx_fe
male)
    return positive_rate_male, positive_rate_female

y_pred = y_pred_prob > 0.5
y_pred = y_pred.astype(float)
y_pred
```

```
array([1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 1., 1., 1
., 0.,
       0., 1., 0., 0., 1., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1
```

```
., 0.,
       0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 0., 1., 1., 1., 0
., 1.,
       0., 0., 1., 0., 0., 0., 1., 0., 0., 0., 1., 1., 0., 0., 0., 0
., 0.,
       0., 1., 0., 0., 0., 0., 0., 1., 1., 0., 1., 1., 0., 0., 0., 0
., 0.,
       1., 1., 1., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 1., 1., 0
., 1.,
       1., 0., 0., 0., 0., 0., 0., 1., 1., 0., 0., 0., 0., 0., 1., 1
., 0.,
       0., 0., 0., 0., 0., 1., 1., 1., 0., 0., 0., 0., 1., 1., 0., 0
., 0.,
       1., 0., 0., 0., 1., 0., 1., 0., 0., 1., 0., 0., 1., 1., 0., 0
., 0.,
       1., 0., 0., 0., 0., 0., 1., 1., 0., 1., 0., 0., 1., 0., 0., 0
., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 1
., 1.,
       0., 0., 0., 1., 0., 1., 1., 1., 0., 1., 0., 0., 0., 1., 1., 0
., 1.,
       0., 1., 0., 1., 1., 0., 0., 0., 1., 1., 0., 1., 0., 1., 0., 0
., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0
., 1.,
       1., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
```

```python
# ajustement des seuils pour l'équité et la séparation des groupes
y_pred_prob = grid_search.predict_proba(X_test)[:,1]

def find_threshold_group_fairness(y_pred_prob):
    for i in range(100):
        threshold = i * 0.01
        y_pred = (y_pred_prob > threshold).astype(float)
        pr_male, pr_female = check_group_fairness_gender_with_return
(X_test_idx_male, X_test_idx_female, y_pred)
        if abs(pr_male - pr_female) < 0.01:
            print(f"On trouve seuil : {threshold}")
            print(f"Pour les hommes: {pr_male}")
            print(f"Pour les femmes: {pr_female}")

find_threshold_group_fairness(y_pred_prob)
```

```
On trouve seuil : 0.0
Pour les hommes: 1.0
Pour les femmes: 1.0
On trouve seuil : 0.99
Pour les hommes: 0.0
Pour les femmes: 0.0
```

```
# Après modification de la précision
# On choisit un seuil = 0.67 !
y_pred = (y_pred_prob > 0.67).astype(float)
print(accuracy_score(y_test,y_pred))
print("\n")
# Après modification : équité de groupe
check_group_fairness_gender(X_test_idx_male, X_test_idx_female, y_pr
ed)
1.0


Taux des bons dossier des hommes: 28.24858757062147 %
Taux des bons dossier des femmes: 30.136986301369863 %
```

Équité améliorée - Séparation

```
# Entraînons d'abord le modèle, avec le dataset

param_test1 = {
 'max_depth':[3,5,6,10],
 'min_child_weight':[3,5,10],
 'gamma':[0.0, 0.1, 0.2, 0.3, 0.4],
# 'reg_alpha':[1e-5, 1e-2, 0.1, 1, 10],
 'subsample':[i/100.0 for i in range(75,90,5)],
 'colsample_bytree':[i/100.0 for i in range(75,90,5)]
}

#Création du classifieur
model_xg = XGBClassifier(random_state=2)

grid_search = GridSearchCV(model_xg, param_grid=param_test1, cv=5, s
coring='recall')
grid_search.fit(X_train, y_train)
# Avant modification - précision du modèle :
y_pred = grid_search.predict(X_test)

# Vérification des résultats obtenus
print(accuracy_score(y_test,y_pred))
print("\n")
print(confusion_matrix(y_test, y_pred))
from sklearn.metrics import roc_auc_score
y_pred_prob = grid_search.predict_proba(X_test)[:,1]


print(f"ROC_AUC score: {roc_auc_score(y_test, y_pred_prob)}")

1.0   100%

      Matrice de confision
      [[178   0]
```

```
       [  0  72]]
      ROC_AUC score: 1.0
```

```
def check_seperation_gender_with_return(X_test_idx_male, X_test_idx_
female, y_test, model_pred):
    # Check FPR Parity
    label_false_male_count = 0
    label_false_female_count = 0

    fp_count_male = 0
    fp_count_female = 0

    for idx, (y, y_hat) in enumerate(zip(y_test, model_pred)):
        if idx in X_test_idx_male:
            if y == 0:
                label_false_male_count += 1
                if y_hat == 1:
                    fp_count_male +=1
        elif idx in X_test_idx_female:
            if y == 0:
                label_false_female_count += 1
                if y_hat == 1:
                    fp_count_female +=1
        else:
            pass

    # Check FNR Parity
    label_true_male_count = 0
    label_true_female_count = 0

    fn_count_male = 0
    fn_count_female = 0

    for idx, (y, y_hat) in enumerate(zip(y_test, model_pred)):
        if idx in X_test_idx_male:
            if y == 1:
                label_true_male_count += 1
                if y_hat == 0:
                    fn_count_male +=1
        elif idx in X_test_idx_female:
            if y == 1:
                label_true_female_count += 1
                if y_hat == 0:
                    fn_count_female +=1
        else:
            pass

    fp_rate_male = fp_count_male / label_false_male_count
    fp_rate_female = fp_count_female / label_false_female_count
```

```
    fn_rate_male = fn_count_male / label_true_male_count
    fn_rate_female = fn_count_female / label_true_female_count
   return fp_rate_male, fp_rate_female, fn_rate_male, fn_rate_female
y_pred_prob = grid_search.predict_proba(X_test)[:,1]

def find_threshold_seperation(y_pred_prob):
    for i in range(100):
        threshold = i * 0.01
        y_pred = (y_pred_prob > threshold).astype(float)
        fp_rate_male, fp_rate_female, fn_rate_male, fn_rate_female =
check_seperation_gender_with_return(X_test_idx_male, X_test_idx_fema
le, y_test, y_pred)
        if abs(fp_rate_male - fp_rate_female) < 0.01 and abs(fn_rate
_male- fn_rate_female) < 0.01:
            print(f"Nous trouvons le seuil: {threshold}")
            print(f"Taux de precision pour les hommes: {fp_rate_male
}")
            print(f"Taux de precision pour les femmes: {fp_rate_fema
le}")



find_threshold_seperation(y_pred_prob)
```

```
Nous trouvons le seuil: 0.0
Taux de precision pour les hommes: 1.0
Taux de precision pour les femmes: 1.0
Nous trouvons le seuil: 0.01
Taux de precision pour les hommes: 0.0
Taux de precision pour les femmes: 0.0
Nous trouvons le seuil: 0.02
Taux de precision pour les hommes: 0.0
Taux de precision pour les femmes: 0.0
```

CONCLUSION GENERALE

Le problème traité dans ce travail concerne l'amélioration de la classification des dosiers de prêts dans les conditions d'insuffisance d'informations à priori, déterministes et fiables sur l'état et la nature de la formation de dossiers de prêts à l'instant de prise de vue.

Notre expérience montre que la combinaison de classifieurs améliore nettement les performances du système de classification automatique par rapport à chacun des classifieurs prisde manière isolé, ainsi que le choix de classifieurs est important pour aboutir à une meilleure performance avec un minimum de classifieurs sélectionnés.

L'approche du vote majoritaire des classifieurs pour une meilleure classification des de dossiers de prêts des clients a été proposée. La théorie de l'évidence est basée sur le principe de combinaison parallèle des systèmes répartis. Elle est générale et applicable à tout type de classifieurs.

Perspectives :

Après un long momemnt passé dans la recherche pour la problématique de precdiction de la crédibilité des cliens principalement sur la question de prêt bancaire, nous proposons comme perspectives :

– Ajouter d'autres espaces de données pour augmenter la quantité de résultats.
– intéresser les banques à faire fois aux systèmes intélligents pour rationnaliser leurs décisions en ce qui concerne le prêt bancaire.
– Appliquer des post traitements sur les résultats obtenus pour améliorer la qualité de la décision.
– Introduire les notions de voisinage.
– Explorer d'autres modèles de classification.

BIBLIOGRAPHIE

Ouvrages généraux

- A. Y. Ng and M. I. Jordan. On Discriminative vs. Generative Classifiers: A comparison of logistic regression and Naive Bayes. in NIPS 14, 2002.
- Bikmukhametov, T., « Combining machine learning and process engineering physics towards enhanced accuracy and explainability of data-driven models », *Computers & Chemical Engineering*, 2020.
- Dash, T., « A review of some techniques for inclusion of domain-knowledge into deep neural networks », *Scientific Reports*, 2022.
- John Markoff, *« On 'Jeopardy!' Watson Win Is All but Trivial »*, *The New York Times*, 16 février 2011
- T. Mitchell, Generative and Discriminative Classifiers: Naive Bayes and Logistic Regression. Draft Version, 2005.
- Gérard Biardeaud et Philippe Florès, *Crédit à la consommation : protection du consommateur*, Paris, Delmas Express, p200, 2012.
- Guo, S., *« An introduction to Surrogate modeling, Part I: fundamentals »*, sur *Towards Data Science*, p12, 2020.
- Louis-Ferdinand Céline, *Mort à crédit*, Paris, Gallimard, coll. « Folio », p.622, 1985.
- Jérôme Lasserre Capdevielle et Michel Storck, *Le crédit aspects juridiques et économiques*, Paris, Dalloz,p 210, ,2012.
- Maloof, M. A., *Machine Learning and Data Mining for Computer Security*, Springer, 2006.
- Montanez-Barrera, J. A., « Correlated-informed neural networks: A new machine learning framework to predict pressure drop in micro-channels », *International Journal of Heat and Mass Transfer*, 2022.
- Richard O. Duda, Peter E. Hart, David G. Stork, *Pattern Classification*, Wiley-interscience, 2001.
- Tom M. Mitchell, *Machine Learning*, Wiley-interscience,1997.

Articles scientifiques

- Alan Turing, « *On computable numbers, with an application to the entscheidungsproblem* », *Proceedings of the London Mathematical Society*, s2-42, 12 novembre 1936, p. 230-265 (DOI 10.1112/plms/s2-42.1.230).
- Y. LeCun, B. Boser, J. S. Denker et D. Henderson, « Backpropagation Applied to Handwritten Zip Code Recognition », *Neural Computation*, vol. 1, nº 4, 1er décembre 1989, p. 541–551.

Note de cours et autres

- Pierre Kafunda K., *note de cours info centre*, L2 math info Unikin, p. 17, 2022.
- Le service clientèle ***Eco BANK.***

Sites web

- www.legifrance.gouv.fr (consulté le 23 septembre 2020).
- http://www.college-de-france.fr/site/stanislas-dehaene/_course.htm [Consulté en avril 2022].
- https://www.college-de-france.fr/site/yann-lecun/Recherches-sur-l-intelligence-artificielle.htm [Consulté en juillet 2022].
- sage-automatique.htmlhttps://www.techno-science.net/glossaire-definition/Apprentis (consulté en février 2022).
- « apprentissage automatique » [archive], Le Grand Dictionnaire terminologique, Office québécois de la langue française (consulté le 28 janvier 2020)