Université Mohammed Premier, Oujda
Faculté multidisciplinaire, Nador
Département d'informatique
◇ Maroc ◇

Mémoire de Projet

Module : datamining

Data Science and Intelligent Systems .

Par

ghizlane chtouki

CREDIT RISCK

Encadré par :
Pr. el ansari anass

année universitaire : 2022/2023

Remerciement

Je tiens à exprimer ma gratitude à toutes les personnes qui m'ont aidé dans la réalisation de ce rapport. Tout d'abord, je tiens à remercier mon professeur el anassari pour ses conseils et son soutien tout au long du processus. Sa patience et son expertise ont été inestimables.

Enfin, je tiens à remercier ma famille et mes amis pour leur soutien et leurs encouragements constants. Leur confiance en moi a été une source d'inspiration et de motivation tout au long de ce projet.

Merci à tous ceux qui ont contribué à la réalisation de ce rapport.

Résumé

Afin de valider notre module du data mining, les étudiants doivent traiter un projet qui résoudre un problème de classification ou de régression. Et comme la plupart des étudiants curieux par ce domaine, j'ai eu l'opportunité de réaliser un projet pour transformer ma capacité et l'enrichir en collaboration avec notre professeur Anas EL ANSARI. Ma mission a pour finalité de réaliser un projet qui traite un problème de classification de credit risck en utilisant le modèle CRISP-DM. Ce rapport décrit les différentes phases et étapes de la réalisation de ce projet en cinq principaux chapitres : la première chapitre intitulé « introduction » qui contient une introduction générale .

Le deuxieme chapitre intitulé « data mining » est le point de départ. Il consiste dans un premier lieu, à donner une vision générale sur le concept de l'intelligence artificielle ainsi qu'une explication du data mining et son objectif.

Le deuxième troixieme intitulé « Compréhension du problème et du dataset », on va exploiter ce chapitre pour comprendre la problématique et aussi pour comprendre notre dataset, ainsi qu'une présentation de l'ensembles des outils utilisés tout au long du développement de ce projet.

Dans la quatrieme chapitre qui intitulé "Analyse exploratoire et Préparation des données", on va analyser notre dataset en déterminant les relations entre les différents features ainsi que la relation les features et le target, et on va préparer cette dataset pour quelle être utilisé dans des algorithmes.

Puis le cinquième nommé « Modeling et evaluation », on va entamer la partie réalisation et implémentation dans lesquels on présente l'ensemble des algorithmes utilisés pour développer les modèles ainsi que l'évaluation de ces modèles, et l'amélioration des modèles sélectionnées.

le sixième et le dernier chapitre intitulé « déploiement » dans lequel on va s'assure que le système est prêt pour être exploit par les clients, tel qu'on va définir dans ce chapitre les étapes pour mettre en production notre modèle.

en fin la derniere chapitre qui contient une conclusion generale .

Table des matières

Table des figures

Chapitre 1

Introduction

Le présent rapport vise à étudier et à analyser le domaine crucial de la classification du risque de crédit, une discipline essentielle dans le secteur financier. La prise de décision concernant l'octroi de crédits est une tâche complexe pour les institutions financières, car elle implique une évaluation minutieuse de la solvabilité et de la capacité de remboursement des emprunteurs potentiels. Une gestion efficace du risque de crédit est essentielle pour prévenir les défauts de paiement, minimiser les pertes financières et assurer la stabilité du système financier dans son ensemble.

Dans ce rapport, nous explorerons diverses techniques de classification qui nous permettent de prédire le risque de défaut de paiement d'un emprunteur. Pour cela, nous utiliserons des algorithmes d'apprentissage automatique tels que le k-plus proches voisins (k-NN), l'arbre de décision, le Random Forest, la régression logistique et Naive Bayes, afin de créer des modèles prédictifs robustes. Ces modèles seront ensuite évalués en utilisant des métriques telles que l'exactitude, la précision, le rappel et le F1-score pour mesurer leurs performances et leur capacité à généraliser aux données inconnues.

La classification du risque de crédit est un domaine en constante évolution, et notre rapport abordera également les défis et les enjeux associés à l'utilisation de ces modèles prédictifs dans un contexte réel. Nous discuterons des considérations éthiques liées à l'utilisation de données sensibles et de l'importance d'une interprétabilité adéquate des modèles pour garantir la confiance des utilisateurs et des régulateurs.

En conclusion, ce rapport sur la classification du risque de crédit fournira un aperçu approfondi de l'utilisation de l'apprentissage automatique pour évaluer le risque de crédit, en mettant l'accent sur l'importance de l'exactitude et de la fiabilité des modèles. Nous espérons que cette étude contribuera à une meilleure compréhension des techniques de classification du risque de crédit et à leur application efficace dans le secteur financier pour améliorer la gestion du risque et favoriser la stabilité économique.

Chapitre 2

Data Mining

2.1 Introduction

Le data mining n'est pas né lors de l'ère numérique. Ce concept existe depuis plus d'un siècle mais il est devenu réellement connu dans les années 1980. Depuis, un long chemin a été parcouru. Les entreprises utilisent désormais le data mining et le machine learning pour accomplir de nombreuses tâches, de l'amélioration du processus de vente à l'interprétation des données financières pour l'investissement.

Découvrez dans ce guide tout ce qu'il faut savoir sur le data mining, de sa définition à son utilité concrète dans l'entreprise en passant par sa mise en œuvre opérationnelle

2.2 Définition de datamining

L'exploration de données notes , connue aussi sous l'expression de fouille de données, forage de données, prospection de données, data mining, ou encore extraction de connaissances à partir de données, a pour objet l'extraction d'un savoir ou d'une connaissance à partir de grandes quantités de données, par des méthodes automatiques ou semi-automatiques.

Elle se propose d'utiliser un ensemble d'algorithmes issus de disciplines scientifiques diverses telles que les statistiques, l'intelligence artificielle ou l'informatique, pour construire des modèles à partir des données, c'est-à-dire trouver des structures intéressantes ou des motifs selon des critères fixés au préalable, et d'en extraire un maximum de connaissances.

L'utilisation industrielle ou opérationnelle de ce savoir dans le monde professionnel permet de résoudre des problèmes très divers, allant de la gestion de la relation client à la maintenance préventive, en passant par la détection de fraudes ou encore l'optimisation de sites web. C'est aussi le mode de travail du journalisme de données.

L'exploration de données fait suite, dans l'escalade de l'exploitation des données de l'entreprise, à l'informatique décisionnelle. Celle-ci permet de constater un fait, tel que le chiffre d'affaires, et de l'expliquer comme le chiffre d'affaires décliné par produits, tandis que l'exploration de données permet de classer les faits et de les prévoir dans une certaine mesure-notes ou encore de les éclairer en révélant par exemple les variables ou paramètres qui pourraient faire comprendre pourquoi le chiffre d'affaires de tel point de vente est supérieur à celui de tel autre.

2.3 Les avantages de datamining

Les entreprises voient arriver des données dans de multiples formats à une vitesse et dans des volumes sans précédent. Être une entreprise data-driven (pilotée par la donnée) n'est plus une option.

Le succès de toute structure dépend désormais de sa rapidité à exploiter les insights issus du Big Data et à les intégrer dans le processus décisionnel et métier afin d'identifier et conduire des actions pertinentes au sein de l'organisation.

Le data mining aide les entreprises à optimiser leur avenir. Il leur permet de comprendre le passé et le présent et de faire des prédictions précises sur ce qui est susceptible d'arriver.

Le data mining peut être utilisé pour répondre à de nombreux objectifs business et commerciaux comme :

2.4 CRISP-DM

CRISP-DM ou CR oss Industry S tandard P rocess for Data Mining est un modèle de processus à six phases qui décrit naturellement le cycle de vie de la science des données . C'est comme un ensemble de garde-corps pour vous aider à planifier, organiser et mettre en œuvre votre projet de science des données (ou d'apprentissage automatique).

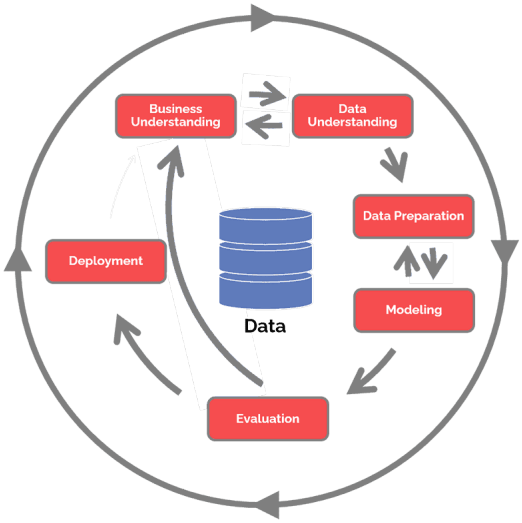

FIGURE 2.1 – CRISP-DM

2.4.1 Business Understanding

"Business Understanding" (Compréhension du domaine) est la première phase du processus. Cette étape est cruciale car elle vise à comprendre le problème métier que le projet de data mining vise à résoudre et à définir clairement les objectifs du projet.

Voici les principales activités réalisées lors de la phase "Business Understanding" :

Définir les objectifs commerciaux : Cette étape consiste à discuter avec les parties prenantes et les parties intéressées pour comprendre les objectifs commerciaux du projet de data mining. Ces objectifs peuvent inclure des questions telles que l'augmentation des ventes, la réduction des coûts, l'amélioration de la satisfaction client, etc.

Identifier les problèmes à résoudre : Il s'agit de comprendre les problèmes métier spécifiques que le projet de data mining doit résoudre. Cela peut inclure l'identification des tendances, des schémas, des risques ou des opportunités qui peuvent être exploités pour améliorer les performances de l'entreprise.

Définir les critères de succès : Cette étape consiste à déterminer les critères qui indiqueront si le projet de data mining est réussi ou non. Ces critères peuvent être des mesures de performance spécifiques, des seuils de qualité ou d'autres indicateurs clés de succès.

Collecte des ressources : Il s'agit de rassembler les ressources nécessaires pour mener à bien le projet de data mining, y compris les données pertinentes, les outils logiciels, les compétences techniques, etc.

Contraintes et limitations : Identifier les contraintes et les limitations potentielles qui pourraient affecter le projet de data mining, telles que les limites de budget, les délais, les ressources disponibles, etc.

Planifier la suite : À la fin de cette phase, une compréhension claire du domaine est établie, et les objectifs commerciaux, les critères de succès et les contraintes sont définis. Le projet peut alors passer à la phase suivante de CRISP-DM, c'est-à-dire "Data Understanding" (Compréhension des données), où l'exploration des données commence.

FIGURE 2.2 – Business Understanding

2.4.2 Data Understanding

"Data Understanding" (Compréhension des données) est la deuxième phase du processus. Cette étape vise à explorer et à comprendre les données disponibles pour le projet de data mining.

Voici les principales activités réalisées lors de la phase "Data Understanding" :

Collecte des données : Cette étape consiste à rassembler toutes les sources de données pertinentes pour le projet de data mining. Cela peut inclure des bases de données, des fichiers CSV, des API, des fichiers texte, etc.

Exploration des données : Une fois que les données ont été collectées, il est important de les explorer pour en comprendre la structure, la qualité, les caractéristiques et les relations. Cela peut impliquer des tâches telles que l'inspection des données, la détection des valeurs manquantes ou aberrantes, l'analyse des distributions des variables, etc.

Vérification de la qualité des données : Cette étape consiste à évaluer la qualité des données, en identifiant les erreurs, les doublons, les valeurs manquantes ou incohérentes. Il est également important de comprendre les limites et les biais des données disponibles.

Exploration des relations entre les variables : Il est essentiel d'analyser les relations entre les variables pour identifier les corrélations, les tendances ou les motifs intéressants. Cela peut inclure des techniques d'analyse statistique, de visualisation ou d'autres méthodes exploratoires.

Vérification de la pertinence des données : À cette étape, il convient d'évaluer la pertinence des données par rapport aux objectifs du projet de data mining. Certaines variables peuvent être inutiles ou redondantes, tandis que d'autres peuvent nécessiter une transformation ou une ingénierie pour être utilisables.

Documentation des résultats : À la fin de cette phase, il est important de documenter les résultats de la compréhension des données, y compris les découvertes clés, les problèmes identifiés, les transformations de données nécessaires, etc. Ces informations seront utiles pour les phases ultérieures du processus CRISP-DM.

FIGURE 2.3 – data préparation

2.4.3 Préparation des données

"Data Preparation" (Préparation des données) est la troisième phase du processus. Cette étape vise à préparer les données de manière à ce qu'elles soient appropriées pour l'analyse et le modèle de data mining.

Voici les principales activités réalisées lors de la phase "Data Preparation" :

Nettoyage des données : Cette étape consiste à traiter les problèmes de qualité des données identifiés lors de la phase "Data Understanding". Cela peut inclure le traitement des valeurs manquantes, la gestion des valeurs aberrantes ou incorrectes, la résolution des doublons, etc.

Sélection des variables : Il peut être nécessaire de sélectionner les variables les plus pertinentes pour le modèle de data mining. Cela peut impliquer l'analyse de la corrélation entre les variables, l'utilisation de techniques de sélection de variables telles que la régression logistique, les arbres de décision, etc.

Transformation des variables : Il peut être nécessaire de transformer les variables pour les rendre plus adaptées à l'analyse. Cela peut inclure la normalisation des variables numériques, la discrétisation des variables continues, la création de variables dérivées ou l'encodage des variables catégorielles.

Construction de nouvelles variables : Il peut être nécessaire de créer de nouvelles variables à partir des variables existantes pour capturer des informations supplémentaires ou pour améliorer la performance du modèle. Cela peut inclure l'ingénierie des caractéristiques, la création d'indicateurs ou la combinaison de variables existantes.

Partitionnement des données : Il est courant de diviser les données en ensembles d'entraînement, de validation et de test. L'ensemble d'entraînement est utilisé pour ajuster le modèle, l'ensemble de validation est utilisé pour ajuster les paramètres du modèle et l'ensemble de test est utilisé pour évaluer la performance du modèle final.

Documentation des étapes de préparation des données : Il est important de documenter toutes les étapes de préparation des données, y compris les transformations appliquées, les variables sélectionnées, les nouvelles

variables créées, etc. Cette documentation permet de reproduire les résultats et de faciliter la compréhension du processus de data mining.

FIGURE 2.4 – preparation des donnes

2.4.4 Modeling

"Modeling" (Modélisation) est la quatrième phase du processus. Cette phase consiste à construire des modèles prédictifs ou des modèles de classification à partir des données préparées lors des étapes précédentes.

Voici les principales activités réalisées lors de la phase "Modeling" :

Construction des modèles : Cette étape implique la sélection des techniques de modélisation appropriées en fonction du problème métier et des données préparées. Les modèles peuvent inclure des algorithmes tels que les arbres de décision, les réseaux neuronaux, les modèles linéaires, les machines à vecteurs de support, etc. Les modèles sont construits en utilisant les données d'entraînement.

Réglage des modèles : Une fois les modèles sélectionnés, il est souvent nécessaire de régler les paramètres et les hyperparamètres des modèles pour optimiser leurs performances. Cela peut être réalisé en utilisant des techniques telles que la recherche par grille, la validation croisée, etc. L'objectif est de trouver les meilleures combinaisons de paramètres pour chaque modèle.

Comparaison des modèles : Si plusieurs modèles ont été construits, il est important de les comparer pour sélectionner le modèle le plus performant. La comparaison peut se faire en fonction de critères prédéfinis tels que l'exactitude, la précision, le rappel, le F1-score, etc. L'objectif est de choisir le modèle qui répond le mieux aux objectifs du projet.

Documentation des résultats : À la fin de la phase "Modeling", il est essentiel de documenter les modèles construits, les techniques utilisées, les paramètres réglés et les performances obtenues. Cette documentation facilite la communication des résultats aux parties prenantes et permet de retracer les étapes de modélisation.

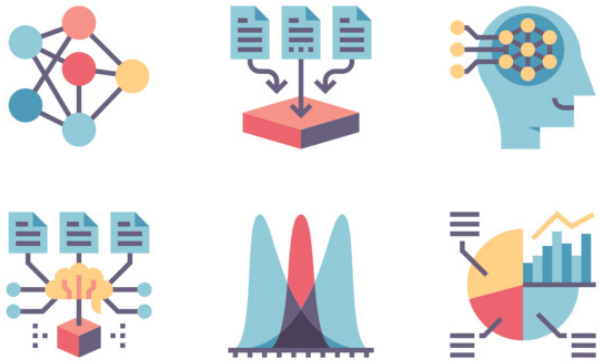

FIGURE 2.5 – Évaluation et interprétation des résultats

2.4.5 Evaluation

L'évaluation permet de mesurer les performances des modèles construits et d'évaluer leur capacité à généraliser les données non vues. Voici une explication de la partie d'évaluation dans la phase "Modeling" :

Métriques d'évaluation : Lors de l'évaluation des modèles, différentes métriques peuvent être utilisées en fonction du type de problème et des objectifs du projet. Ces métriques incluent des mesures telles que l'exactitude (accuracy), la précision (precision), le rappel (recall), le F1-score, l'aire sous la courbe ROC (AUC-ROC), etc. Ces métriques permettent de quantifier la performance du modèle sur les données d'évaluation.

Jeux de données d'évaluation : Les données d'évaluation sont utilisées pour tester les performances des modèles construits. Ces données doivent être distinctes des données d'entraînement et des données de test utilisées lors des autres phases du projet. L'utilisation de jeux de données indépendants permet d'évaluer la capacité des modèles à généraliser et à faire des prédictions précises sur de nouvelles données.

Validation croisée : La validation croisée est une technique couramment utilisée lors de l'évaluation des modèles. Elle consiste à diviser les données d'entraînement en plusieurs sous-ensembles, puis à effectuer plusieurs cycles d'entraînement et d'évaluation en utilisant différentes combinaisons de ces sous-ensembles. Cela permet d'obtenir une estimation plus robuste des performances des modèles.

Comparaison des modèles : L'évaluation permet également de comparer les performances des différents modèles construits. En comparant les métriques d'évaluation, il est possible de déterminer quel modèle est le plus performant pour résoudre le problème métier spécifique. La comparaison des modèles aide à sélectionner le modèle final qui sera utilisé dans la phase suivante du processus.

Documentation des résultats : À la fin de la phase d'évaluation, il est important de documenter les résultats obtenus, y compris les métriques d'évaluation, les performances des modèles, ainsi que les décisions prises sur la base des résultats. Cette documentation facilite la communication avec les parties prenantes et permet de retracer les étapes d'évaluation pour référence future.

L'évaluation des modèles est essentielle pour s'assurer de leur qualité et de leur aptitude à résoudre le problème métier. Elle fournit des informations précieuses sur la performance des modèles et guide les décisions ultérieures dans le processus de data mining.

2.4.6 Déploiement

La dernière étape du processus CRISP-DM est le déploiement des résultats obtenus lors de l'évaluation et de l'interprétation des données. Le déploiement consiste à mettre en œuvre les modèles développés

dans un environnement opérationnel pour une utilisation pratique et à grande échelle. Cela nécessite une planification minutieuse, une préparation des ressources adéquate et une implémentation soigneuse des modèles.

Lors de la planification du déploiement, il est essentiel de collaborer avec les parties prenantes pour définir les exigences opérationnelles spécifiques, les contraintes techniques et les étapes de mise en œuvre. Cette étape permet de s'assurer que le déploiement se déroule de manière fluide et efficace.

Ensuite, il est nécessaire de préparer les ressources requises pour le déploiement, telles que l'infrastructure informatique, les logiciels et les bases de données. Cela garantit que les modèles et les recommandations peuvent être exécutés et utilisés efficacement dans l'environnement opérationnel.

Une fois les ressources prêtes, il est temps d'implémenter les modèles dans l'environnement de production. Cela peut nécessiter la création d'applications logicielles, de services en ligne ou l'intégration des modèles dans des systèmes existants. L'objectif est de permettre l'utilisation des modèles pour la prise de décision en temps réel et d'assurer leur fonctionnement fluide.

Après le déploiement, il est important de suivre et d'évaluer en continu les performances des modèles dans l'environnement opérationnel. Cela peut inclure la collecte de données en temps réel, l'évaluation régulière des résultats obtenus et l'ajustement des modèles si nécessaire. Le suivi en production permet de garantir que les modèles fonctionnent correctement et fournissent des résultats fiables pour soutenir les décisions opérationnelles.

Enfin, pour une utilisation efficace des modèles déployés, il est essentiel de fournir une formation et un support aux utilisateurs. Cela peut inclure des sessions de formation sur l'utilisation des modèles, l'interprétation des résultats et l'accompagnement pour résoudre d'éventuels problèmes techniques. Une formation adéquate et un support continu garantissent que les utilisateurs peuvent tirer le meilleur parti des modèles déployés.

FIGURE 2.6 – Déploiement

2.5 Conclusion

Dans ce chapitre consacré à Data Mining et CRISP-DM (Cross-Industry Standard Process for Data Mining), nous avons exploré les principaux concepts, méthodes et approches associés à l'extraction de connaissances à partir de grandes quantités de données.

Chapitre 3

Compréhension du problème et du dataset

3.1 Intrduction

Dans le chapitre suivant, nous examinerons la problématique actuelle et présenterons une vue d'ensemble détaillée du dataset pour une meilleure compréhension.

3.2 Problèmatique

Les banques prennent un risque financier lorsqu'elles accordent des prêts à des clients, car il existe toujours une possibilité que le client ne rembourse pas l'argent dans le délai convenu, ce risque est connu sous le nom de "crédit risk". Pour évaluer cette probabilité de remboursement, les banques doivent analyser de nombreux paramètres tels que le revenu, les biens, et les dépenses actuelles du client. Cependant, cette analyse manuelle est très chrono-phage et nécessite d'importantes ressources financières.

Grâce aux techniques de machine learning, il est possible d'automatiser cette tâche et de prédire avec une précision accrue les clients susceptibles de présenter des risques de défaut de paiement. En utilisant des modèles d'apprentissage automatique, les banques peuvent évaluer plus efficacement la solvabilité des demandeurs de crédit, ce qui permet de réduire les délais de traitement et d'économiser des ressources. Cette approche permet également de prendre des décisions plus éclairées en matière d'octroi de crédits, ce qui contribue à minimiser les risques pour les institutions financières tout en améliorant la satisfaction des clients.

3.3 Objectif

l objectif principale de cettev project c eest réaliser une application web basée sur un système de prise de décision pour accepter ou refuser une demande de crédit d'un demandeur en fonction de ses données telles que le sexe, le salaire, le salaire du conjoint, le niveau d'études, la situation familiale, etc. Ce type de système est couramment utilisé dans le domaine des services financiers pour évaluer le risque de crédit d'un individu.

3.4 Description du datasets

Dans notre étude sur le risque de crédit, nous avons utilisé un ensemble de données comprenant 665 instances, où chaque instance représente une demande de crédit. Nous avons identifié 11 colonnes ou caractéristiques pour chaque demande, qui nous permettent de mieux comprendre et évaluer le risque associé à chaque emprunteur.

Notre objectif principal était de prédire le risque de défaut de paiement pour chaque demande, ce qui nous a amenés à définir deux classes de sortie. Lorsque la cible est 1, cela signifie qu'il y a un risque de défaut de paiement associé à cette demande (risque de crédit élevé). D'autre part, lorsque la cible est 0, cela indique qu'il n'y a pas de risque de défaut de paiement (risque de crédit faible).

Voici Les détails des colonnes du datasets sont les suivants :

LoanID : Identifiant unique du prêt
Gender : Genre du demandeur (masculin ou féminin)
Married : Indique si le demandeur est marié ou non
Dépendents : Nombre de personnes à charge du demandeur
Éducation : Niveau d'éducation du demandeur (diplômé ou non diplômé)
Self Employed : Indique si le demandeur est travailleur indépendant ou non
ApplicantIncome : Revenu mensuel du demandeur principal
CoapplicantIncome : Revenu mensuel du co-demandeur
Credit_History : Historique de crédit du demandeur (1 pour crédit existant, 0 pour crédit inexistant)
Property_Area : Zone de localisation de la propriété (urbaine, semi-urbaine ou rurale)
Loan_Status : Statut du prêt (il y a un risk de donne le credit a le demandeur ou non)

3.5 Les outils et les bibliothèques utilisées

Après la compréhension de notre problème on a décidé de travailler avec les outils suivants :

3.5.1 Python

Python est un langage de programmation polyvalent et convivial, conçu pour faciliter le développement d'applications de manière rapide et efficace. Son objectif principal est de permettre aux développeurs de créer des logiciels en utilisant une syntaxe claire et concise, favorisant ainsi la lisibilité du code. Python est un langage interprété, ce qui signifie qu'il n'a pas besoin d'être compilé avant d'être exécuté, ce qui accélère le processus de développement. Grâce à sa bibliothèque standard riche et à sa communauté active, Python offre une multitude de modules et d'outils prêts à l'emploi, permettant aux développeurs de créer rapidement des applications pour divers domaines tels que le développement web, l'analyse de données, l'intelligence artificielle et bien plus encore. En raison de sa simplicité et de sa flexibilité, Python est devenu l'un des langages de programmation les plus populaires et est largement utilisé par les programmeurs, les ingénieurs et les scientifiques du monde entier.

FIGURE 3.1 – Python

3.5.2 Colab

Colab (abréviation de Google Colaboratory) est une plateforme d'apprentissage machine basée sur le cloud, développée par Google. Son objectif principal est de fournir un environnement de développement gratuit et collaboratif qui permet aux utilisateurs d'écrire, d'exécuter et de partager du code Python. Colab est spécialement conçu pour faciliter le développement et l'expérimentation avec des projets d'apprentissage machine et d'intelligence artificielle.

Le rôle principal de Colab est de fournir un accès à une instance gratuite de la machine virtuelle, qui offre la possibilité d'exécuter des blocs de code en utilisant Jupyter Notebooks. Cette approche interactive permet aux utilisateurs d'écrire et d'exécuter du code par étapes, ce qui facilite la visualisation et le débogage des résultats. Colab fournit également un accès aux puissants GPU (Graphical Processing Units) et aux TPU (Tensor Processing Units) de Google, ce qui accélère le processus de formation des modèles d'apprentissage machine.

FIGURE 3.2 – Python

3.5.3 Streamlit

Streamlit est une bibliothèque open-source en Python conçue pour faciliter la création rapide d'applications web interactives pour l'analyse de données et la visualisation. Son objectif principal est de permettre

aux développeurs et aux data scientists de transformer facilement leur code Python en applications web conviviales sans avoir à maîtriser des connaissances approfondies en développement web.

Le rôle principal de Streamlit est de simplifier le processus de déploiement d'analyses de données en transformant le code Python en applications web interactives en quelques lignes seulement. Grâce à sa syntaxe simple et intuitive, les utilisateurs peuvent créer des tableaux de bord interactifs, des graphiques, des cartes et d'autres éléments interactifs sans nécessiter de connaissances préalables en HTML, CSS ou JavaScript.

FIGURE 3.3 – Streamlit

3.5.4 Sckit-learn

scikit-learn (également appelé sklearn) est une bibliothèque open-source en Python dédiée à l'apprentissage automatique (machine learning). Son objectif principal est de fournir une interface simple et cohérente pour divers algorithmes d'apprentissage supervisé et non supervisé, ainsi que des outils pour l'évaluation des modèles, la sélection des caractéristiques, la préparation des données et la mise en œuvre de pipelines d'apprentissage.

Le rôle principal de scikit-learn est de faciliter le processus de développement et d'expérimentation en apprentissage automatique en fournissant des implémentations optimisées et bien documentées d'algorithmes populaires. Il permet aux utilisateurs de mettre en œuvre rapidement et efficacement des modèles d'apprentissage sur leurs données sans avoir à écrire tous les détails de l'algorithme de zéro.

En plus de fournir des outils pour l'apprentissage, scikit-learn offre également des fonctionnalités pour la validation croisée, la recherche d'hyperparamètres, la réduction de dimensionnalité, et bien plus encore.

FIGURE 3.4 – Sckit-learn

3.5.5 Seabron

Seaborn est une bibliothèque de visualisation de données en Python basée sur Matplotlib. Son objectif principal est de simplifier la création de graphiques attractifs et informatifs en utilisant une syntaxe simple et intuitive. Seaborn est spécialement conçu pour fonctionner avec des DataFrames de Pandas, ce qui facilite la visualisation des données directement à partir de structures de données courantes en science des données.

Le rôle principal de Seaborn est d'améliorer la visualisation des données en fournissant des fonctions hautement personnalisables pour créer des graphiques tels que les graphiques à barres, les diagrammes à nuages de points, les diagrammes de dispersion, les diagrammes en boîte, les diagrammes en violon, etc. En mettant l'accent sur l'esthétique, Seaborn offre des palettes de couleurs attrayantes et des thèmes prédéfinis pour améliorer la présentation visuelle des graphiques

FIGURE 3.5 – Seabron

3.5.6 Numpy

NumPy est une bibliothèque fondamentale en Python pour le calcul scientifique et le traitement de tableaux multidimensionnels. Son objectif principal est de fournir une prise en charge efficace des opérations mathématiques sur les tableaux, ce qui en fait un outil puissant pour la manipulation et l'analyse de données numériques.

Le rôle principal de NumPy est de permettre aux utilisateurs de créer, manipuler et effectuer des calculs rapides et efficaces sur des tableaux multidimensionnels. En utilisant des structures de données appelées "tableaux NumPy", les utilisateurs peuvent effectuer des opérations vectorisées (opérations sur des tableaux entiers plutôt que des éléments individuels) et des calculs matriciels de manière optimisée, ce qui améliore considérablement les performances et l'efficacité du code.

FIGURE 3.6 – Numpy

3.5.7 Matblotlib

Matplotlib est une bibliothèque de visualisation de données en Python largement utilisée dans le domaine de la science des données, de la recherche scientifique et de l'analyse de données. Son objectif principal est de permettre aux utilisateurs de créer des graphiques et des visualisations de manière simple et efficace.

Le rôle principal de Matplotlib est de fournir des outils pour créer une grande variété de graphiques, tels que les graphiques linéaires, les graphiques à barres, les graphiques en nuage de points, les diagrammes en boîte, les graphiques en secteurs et bien d'autres. Il permet également une personnalisation détaillée des graphiques, ce qui permet aux utilisateurs de contrôler chaque aspect de la visualisation, tels que les couleurs, les étiquettes, les titres, les axes, etc.

FIGURE 3.7 – Matblotlib

3.5.8 Pandas

Pandas est une bibliothèque open-source en Python conçue pour faciliter la manipulation et l'analyse de données tabulaires et séries temporelles. Son objectif principal est de fournir des structures de données puissantes et flexibles, appelées DataFrames et Series, pour faciliter le traitement, le nettoyage et l'analyse de données.

Le rôle principal de Pandas est de permettre aux utilisateurs d'effectuer des opérations de manière intuitive sur des ensembles de données, en leur offrant des fonctionnalités avancées telles que l'indexation, le filtrage, l'agrégation, le regroupement et la fusion de données. Pandas permet également d'importer et d'exporter des données à partir de divers formats de fichiers, tels que CSV, Excel, SQL, etc., ce qui simplifie l'accès aux données externes.

FIGURE 3.8 – Pandas

3.5.9 Pickle

Pickle est un module en Python qui permet de sérialiser (convertir en un format binaire) et de désérialiser (reconstituer depuis le format binaire) des objets Python. Son objectif principal est de permettre la sauvegarde et le chargement d'objets complexes, tels que des listes, des dictionnaires, des classes et des modèles d'apprentissage automatique, en les convertissant en un format binaire. Cela facilite le stockage et le partage de données ou de modèles sans perdre leur structure et leurs attributs.

FIGURE 3.9 – Pickle

3.6 Conclusion

Dans ce chapitre j'ai expliqué la problématique et le dataset, et j'ai aussi présenter l'ensemble des outils nécessaires pour réaliser ma tâche.

Chapitre 4

analyse d'exploration et pré-traitement des donnes

4.1 Introduction

Dans ce chapitre je vais expliquer la phase exploratoire et la phase préparation des données tel qu'on va visualiser notre dataset et déterminer la relation entre l'ensemble des variables, ainsi qu'on va nettoyer et préparer cette data pour qu'elle puisse être utilisé par un algorithme et donne par la suite des meilleures prédictions.

4.2 Analyse d'exploration des donnes AED

Dans cette partie, nous allons effectuer une analyse exploratoire pour obtenir une connaissance générale de notre jeu de données. Cette étape est cruciale pour comprendre les caractéristiques, les propriétés et les tendances présentes dans les données avant de passer à la modélisation et à la prédiction.

4.2.1 Anlyse de forme

Tous d abord on va commencer par les importations des bibliothèques utilisent

Entrée [495]:
```
# Importer les packages
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns



from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
import pandas as pd
import matplotlib.pyplot as plt
from imblearn.over_sampling import SMOTE
from sklearn.metrics import matthews_corrcoef
import pandas as pd
from sklearn.metrics import confusion_matrix

import pickle

import warnings
warnings.filterwarnings('ignore')
```

FIGURE 4.1 – importer les bibliothèques

Utilise la fonction read_csv() de Pandas pour lire un fichier CSV et stocker son contenu dans un dataframe appelé df_credit.

```
Entrée [496]:
df_credit=pd.read_csv('ghiz1.csv')

Entrée [497]:
df_credit
```

Out[497]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	1.0	Urban	Y
...	...	...	...	...	...	...	...	...	...	...	...
609	LP002978	Female	No	0	Graduate	No	2900	0.0	1.0	Rural	Y
610	LP002979	Male	Yes	3+	Graduate	No	4106	0.0	1.0	Rural	Y
611	LP002983	Male	Yes	1	Graduate	No	8072	240.0	1.0	Urban	Y
612	LP002984	Male	Yes	2	Graduate	No	7583	0.0	1.0	Urban	Y
613	LP002990	Female	No	0	Graduate	Yes	4583	0.0	0.0	Semiurban	N

614 rows × 11 columns

FIGURE 4.2 – l affichage de notre data

A l'aide du méthode info() de Pandas on va afficher les informations sur un dataframe, y compris le nombre total de lignes, le nombre de colonnes, le nom et le type de données de chaque colonne, ainsi que la quantité de mémoire utilisée par le dataframe.

```
Entrée [499]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            614 non-null    object
 1   Gender             601 non-null    object
 2   Married            611 non-null    object
 3   Dependents         599 non-null    object
 4   Education          614 non-null    object
 5   Self_Employed      582 non-null    object
 6   ApplicantIncome    614 non-null    int64
 7   CoapplicantIncome  614 non-null    float64
 8   Credit_History     564 non-null    float64
 9   Property_Area      614 non-null    object
 10  Loan_Status        614 non-null    object
dtypes: float64(2), int64(1), object(8)
memory usage: 52.9+ KB
```

FIGURE 4.3 – afficher Résumé des données

A l'aide du méthode describe() de Pandas on va générer un résumé statistique des colonnes numériques d'un dataframe. Elle calcule des statistiques descriptives telles que la valeur moyenne, l'écart type, les quartiles et les valeurs minimales et maximales

```
Entrée [500]: df.describe()
```

Out[500]:

	ApplicantIncome	CoapplicantIncome	Credit_History
count	614.000000	614.000000	564.000000
mean	5403.459283	1621.245798	0.842199
std	6109.041673	2926.248369	0.364878
min	150.000000	0.000000	0.000000
25%	2877.500000	0.000000	1.000000
50%	3812.500000	1188.500000	1.000000
75%	5795.000000	2297.250000	1.000000
max	81000.000000	41667.000000	1.000000

FIGURE 4.4 – Statistiques descriptives.

Pour obtenir les dimensions (le nombre de lignes et de colonnes) d'un dataframe.lignes et le nombre de colonnes.

```
Entrée [501]: df.shape
Out[501]: (614, 11)
```

FIGURE 4.5 – les nombre de lignes et colonnes

La méthode isnull() de Pandas est utilisée pour vérifier si chaque valeur dans un dataframe est nulle ou non. Lorsqu'elle est combinée avec la méthode sum(), elle permet de compter le nombre de valeurs nulles dans chaque colonne du dataframe. On remarque ici qu'on a des valeurs nulles dans nombreuses de colonnes

```
Entrée [502]: # verifier si on a des valeurs manquantes
              df.isnull().sum()
Out[502]: Loan_ID              0
          Gender              13
          Married              3
          Dependents          15
          Education            0
          Self_Employed       32
          ApplicantIncome      0
          CoapplicantIncome    0
          Credit_History      50
          Property_Area        0
          Loan_Status          0
          dtype: int64
```

FIGURE 4.6 – le nombre de valeur manquantes dans chaque colonne

-J'ai utilise value_counts().plot.bar(title='title') pour a l fois afficher tous les categories ou les distincts dans chaque colonne et aussi afficher le nombre de chaque instance dans chaque colonnes

On remarque que notre datasets est pas équilibrée il faut l equilibrer

```
Entrée [510]: df['Loan_Status'].value_counts().plot.bar(title='Loan_Status ')
  Out[510]: <AxesSubplot:title={'center':'Loan_Status '}>
```

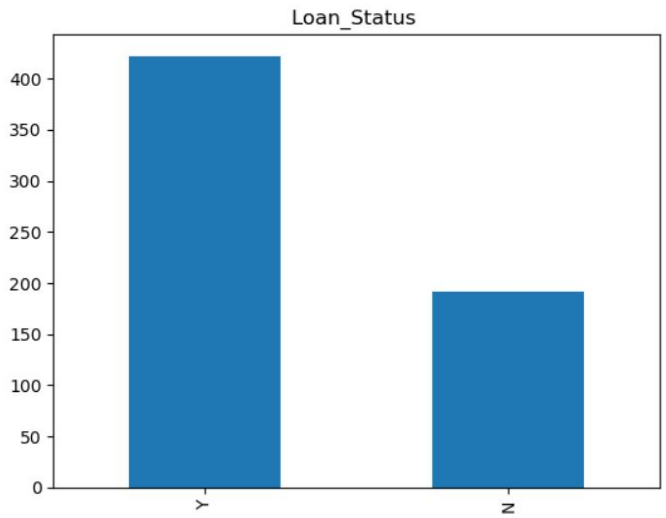

FIGURE 4.7 – Diagramme à barres de comptage.

4.2.2 Analyse de fond

On passe a l'analyse de fond se réfère à une étape clé où l'on explore et examine les caractéristiques et les propriétés des données brutes afin de mieux comprendre leur structure, leur distribution et leurs relations. Cette analyse vise à identifier les motifs, les tendances, les aberrations et les informations potentiellement intéressantes dans les données.

on va commence avec la matrice de correlation qui decrit la realations entre les attributs

```
Entrée [511]:
#correlation des variables numérique
matrix=df.corr()
f,ax=plt.subplots(figsize=(10,12))
sns.heatmap(matrix,vmax=.8,square=True,cmap='BuPu',annot=True)
```

FIGURE 4.8 – matrice de corrélation

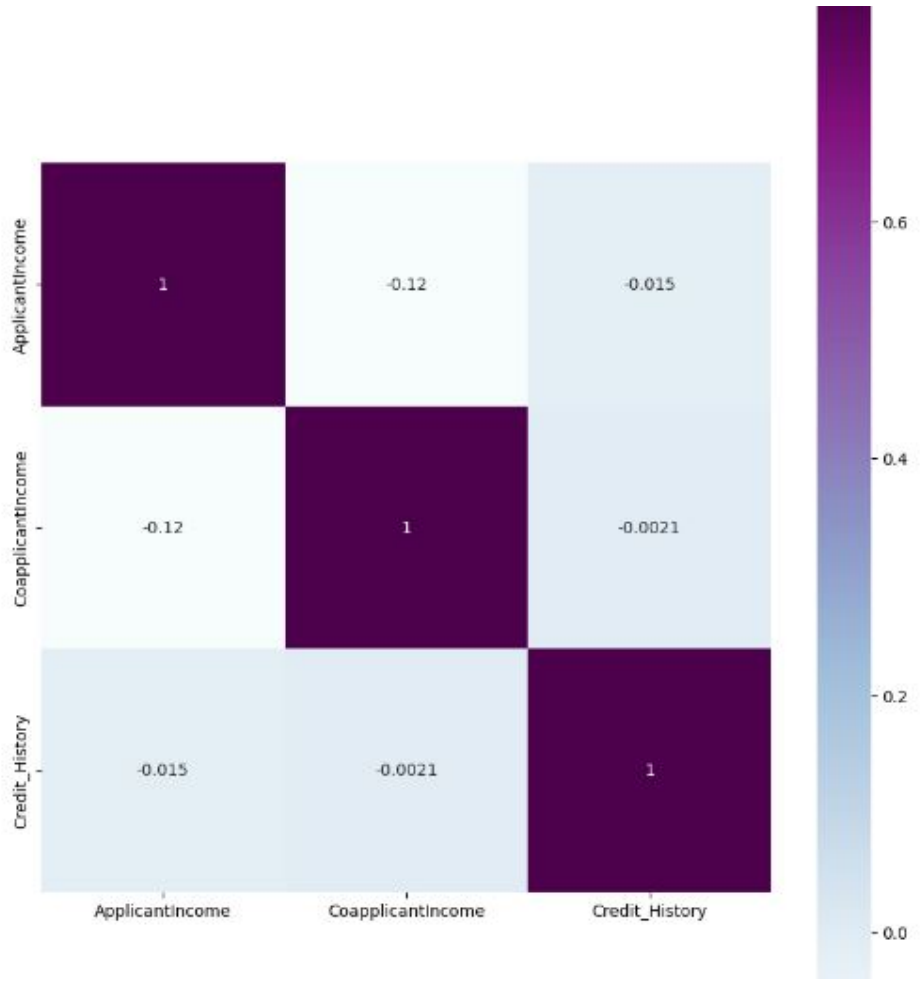

FIGURE 4.9 – matrice de corrélation

On va passe a l'affichage des outliers pour les colonnes numériques.On remarque ici qu'on a des outliers dans les deux colonnes Coapplicantome et Applicantome.

Entrée [512]:
```
# applicantincome
plt.figure(1)

plt.subplot(121)
sns.distplot(df['ApplicantIncome'])

plt.subplot(122)
df['ApplicantIncome'].plot.box(figsize=(16,5))

plt.suptitle('')
plt.show()
```

FIGURE 4.10 – l'affichage des outliers

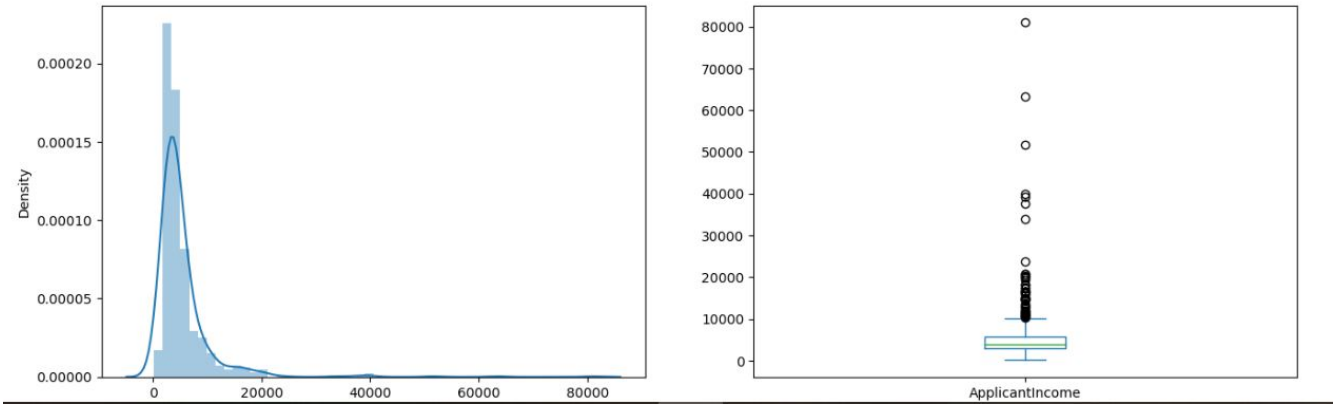

FIGURE 4.11 – l'affichage des outliers

```
Entrée [513]: # applicantincome
              plt.figure(1)

              plt.subplot(121)
              sns.distplot(df['CoapplicantIncome'])

              plt.subplot(122)
              df['CoapplicantIncome'].plot.box(figsize=(16,5))

              plt.suptitle('')
              plt.show()
```

FIGURE 4.12 – l'affichage des outliers

```
Entrée [513]: # applicantincome
              plt.figure(1)

              plt.subplot(121)
              sns.distplot(df['CoapplicantIncome'])

              plt.subplot(122)
              df['CoapplicantIncome'].plot.box(figsize=(16,5))

              plt.suptitle('')
              plt.show()
```

FIGURE 4.13 – l'affichage des outliers

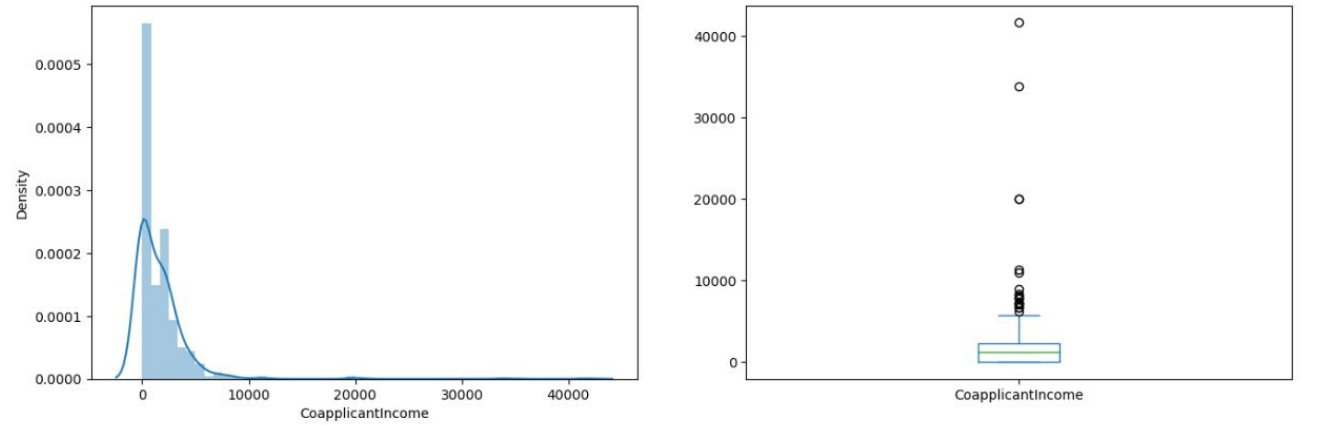

FIGURE 4.14 – l'affichage des outliers

4.3 Pré-traitement

Dans cette partie on va concentrer sur la correction des donnes on va suivre l'étape suivante :

4.3.1 Remplacer des valeurs manquantes

Dans cette partie j'ai essayé de remplacé les valeurs manquantes avec la valeur la plus fréquente dans chaque colonne.

Entrée [514]:
```
df['Gender'].fillna(df['Gender'].mode()[0],inplace=True)
df['Married'].fillna(df['Married'].mode()[0],inplace=True)
df['Dependents'].fillna(df['Dependents'].mode()[0],inplace=True)
df['Self_Employed'].fillna(df['Self_Employed'].mode()[0],inplace=True)
df['Credit_History'].fillna(df['Credit_History'].mode()[0],inplace=True)
```

FIGURE 4.15 – Remplacer des valeurs manquantes

4.3.2 Corriger les outeliers

On va passer maintenant à la correction des outliers tous d abord j'ai calculé l'intervalle des valeurs qui n'ont pas outliers et j'ai remplacé les valeurs qui ont supérieur a cet intervalle avec la valeur de borne supérieure et les valeurs qui ont inférieur à cet intervalle avec la borne inférieure

Entrée [516]:
```
# Calcul de Q1, Q3 et IQR
Q1 = df['CoapplicantIncome'].quantile(0.25)
Q3 = df['CoapplicantIncome'].quantile(0.75)
IQR = Q3 - Q1
IQR
# Calcul des bornes inférieure et supérieure de l'intervalle IQR
borne_inf = Q1 - 1.5 * IQR
borne_sup = Q3 + 1.5 * IQR
# I
df.loc[df['CoapplicantIncome'] < borne_inf, 'CoapplicantIncome'] =borne_inf
df.loc[df['CoapplicantIncome'] > borne_sup, 'CoapplicantIncome'] = borne_sup
df
```

FIGURE 4.16 – Corriger les outeliers

Entrée [518]:
```
# Calcul de Q1, Q3 et IQR
Q1 = df['ApplicantIncome'].quantile(0.25)
Q3 = df['ApplicantIncome'].quantile(0.75)
IQR = Q3 - Q1

# Calcul des bornes inférieure et supérieure de l'intervalle IQR
borne_inf = Q1 - 1.5 * IQR
borne_sup = Q3 + 1.5 * IQR

# Winsorization
df.loc[df['ApplicantIncome'] < borne_inf, 'ApplicantIncome'] = borne_inf
df.loc[df['ApplicantIncome'] > borne_sup, 'ApplicantIncome'] = borne_sup

df
```

FIGURE 4.17 – Corriger les outeliers

4.3.3 Encodage des donnes nominal

l'encodage des valeurs nominal a l aide encoderLabel les variables catégorielles en variables numériques. Elle permet de transformer les catégories en valeurs numériques ordinales, ce qui facilite l'utilisation de ces variables dans les algorithmes d'apprentissage automatique.

```
Entrée [520]: from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
df['Gender'] = le.fit_transform(df['Gender'])
df['Married'] = le.fit_transform(df['Married'])
df['Education'] = le.fit_transform(df['Education'])
df['Self_Employed'] = le.fit_transform(df['Self_Employed'])
df['Property_Area'] = le.fit_transform(df['Property_Area'])
df['Loan_Status'] = le.fit_transform(df['Loan_Status'])
df['Credit_History'] = le.fit_transform(df['Credit_History'])
df['Dependents'] = le.fit_transform(df['Dependents'])
df
```

FIGURE 4.18 – Encodage des donnes nominal

4.3.4 Equilibre de data

J'ai équilibré datasets avec oversimpling , j ai pas utiliser undersimpling parce-que j ai pas une data très volumineux.

```
Entrée [526]: X = df.drop('Loan_Status', axis=1)
              X = df.drop('Loan_ID', axis=1)
              y = df['Loan_Status']
              smote = SMOTE()

Entrée [527]: X_oversampled, y_oversampled = smote.fit_resample(X, y)
              oversampled_df = pd.concat([pd.DataFrame(X_oversampled)], axis=1)
              oversampled_df.shape

  Out[527]: (844, 10)
```

FIGURE 4.19 – l'équilibre de data

4.3.5 Normalisation

La normalisation des données joue un rôle essentiel dans l'analyse de données et la modélisation statistique. Son objectif principal est de mettre les valeurs des différentes variables sur une échelle commune.

```
Entrée [116]: from sklearn import preprocessing

              min_max_scaler = preprocessing.MinMaxScaler()
              X=min_max_scaler.fit_transform(X)
              X

  Out[116]: array([[1.        , 0.        , 0.        , ..., 0.        , 1.        ,
                    1.        ],
                   [1.        , 1.        , 0.33333333, ..., 0.26257482, 1.        ,
                    0.        ],
                   [1.        , 1.        , 0.        , ..., 0.        , 1.        ,
                    1.        ],
                   ...,
                   [1.        , 1.        , 0.        , ..., 0.38441355, 0.        ,
                    0.        ],
                   [1.        , 1.        , 0.        , ..., 0.37012694, 0.        ,
                    0.        ],
                   [1.        , 1.        , 0.66666667, ..., 0.27121337, 0.        ,
                    0.        ]])
```

FIGURE 4.20 – Normalisation

4.3.6 Division de data

```
Entrée [533]: # spécifier la partie test et la partie train
              from sklearn.model_selection import train_test_split

              x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=6)
```

FIGURE 4.21 – Division de data

4.4 Conclusion

Dans ce chapitre j'ai expliqué et détaillé les deux principales phases dans du modèle CRISPDM, c'est la phase exploratoire des données et la phase de préparation des données.

Chapitre 5

Modiling et Evaluation

5.1 Introduction

Ce chapitre se concentre sur la phase de modélisation du projet et présente une sélection d'algorithmes utilisés pour cette tâche, notamment le KNN, l'arbre de décision, Random Forest, la régression logistique et Naive Bayes. L'objectif de ce chapitre est d'améliorer certains de ces modèles en utilisant la recherche de grille (grid search).

5.2 Les algorithmes de machines learning

Dans cette partie j'applique plusieurs algorithmes avec l'évaluation bien sur pour se voir qu'il algorithme a l'accuracy et Mcc le plus élevé.

5.2.1 Knn(k plus proches voisins)

L'algorithme des k plus proches voisins, également connu sous le nom de KNN ou k-NN, est un discriminant d'apprentissage supervisé non paramétrique, qui utilise la proximité pour effectuer des classifications ou des prédictions sur le regroupement d'un point de données individuel. Bien qu'il puisse être utilisé pour des problèmes de régression ou de classification, il est généralement utilisé comme algorithme de classification, en partant de l'hypothèse que des points similaires peuvent être trouvés les uns à côté des autres.

5.2.2 Regression Logistique

La régression logistique fait partie des modèles mathématiques très souvent utilisés dans le domaine du Machine Learning et dans celui de l'intelligence artificielle (IA). En effet, la régression logistique est un modèle linéaire généralisé qui utilise une fonction logistique comme une fonction de lien

5.2.3 Random Forest

L'algorithme de forêt aléatoire est une extension de la méthode de bagging car il utilise à la fois le bagging et le caractère aléatoire des caractéristiques pour créer une forêt non corrélée d'arbres de décision. Le caractère aléatoire des caractéristiques, également connu sous le nom de mise en sac de caractéristiques ou « la méthode du sous-espace aléatoire » génère un sous-ensemble aléatoire de caractéristiques, ce qui garantit une faible corrélation entre les arbres de décision. Il s'agit d'une différence essentielle entre les arbres de décision et les forêts aléatoires. Alors que les arbres de décision prennent en compte toutes les divisions de fonctionnalités possibles, les forêts aléatoires ne sélectionnent qu'un sous-ensemble de ces fonctionnalités.

5.2.4 Décision Tree

L'algorithme de décision tree permet de diviser récursivement l'ensemble de données en sous-ensembles plus petits en fonction des caractéristiques des données, jusqu'à ce que chaque sous-ensemble contienne des exemples d'une seule classe ou atteigne un critère d'arrêt. Ensuite, il construit un arbre de décision en utilisant ces divisions pour prendre des décisions basées sur les caractéristiques des données.

5.2.5 Naives Bayes

Naive Bayes est un algorithme d'apprentissage supervisé utilisé principalement pour la classification. Son principe repose sur le théorème de Bayes et l'assomption de naïveté, qui considère que toutes les caractéristiques (variables) sont indépendantes les unes des autres.

L'algorithme de Naive Bayes calcule la probabilité qu'une instance appartienne à une classe donnée en utilisant la probabilité a priori des classes et les probabilités conditionnelles des caractéristiques étant donné chaque classe. Lorsqu'il est confronté à une nouvelle instance, il calcule ces probabilités et attribue la classe avec la probabilité la plus élevée à l'instance.

5.3 GridSearch

GridSearch, également connu sous le nom de recherche de grille, est une technique d'optimisation des hyperparamètres utilisée dans l'apprentissage automatique. Son objectif principal est de rechercher de manière systématique les combinaisons optimales d'hyperparamètres pour un modèle donné.

L'hyperparamétrage joue un rôle crucial dans la performance des modèles d'apprentissage automatique, et GridSearch permet d'automatiser le processus de recherche des valeurs optimales. Il fonctionne en définissant une grille de valeurs possibles pour chaque hyperparamètre, puis en évaluant toutes les combinaisons possibles à l'aide d'une validation croisée.

En parcourant cette grille, GridSearch évalue la performance du modèle pour chaque combinaison d'hyperparamètres et identifie celle qui offre les meilleures performances sur un ensemble de validation. Cela permet de trouver les hyperparamètres optimaux qui maximisent la précision du modèle et minimisent le risque de surajustement (overfitting) aux données d'entraînement.

Grâce à GridSearch, les chercheurs et les développeurs peuvent économiser du temps et de l'effort en automatisant le processus fastidieux de réglage manuel des hyperparamètres. En utilisant cette technique, il est possible de trouver plus facilement les configurations optimales pour les modèles, ce qui conduit à des modèles mieux performants et plus généralisables pour des tâches de prédiction et de classification dans l'apprentissage automatique.

5.4 Application des algorithmes

Dans cette partie, j' appliquee la modélisation et l'évaluation en utilisant la recherche de grille (Grid Search) pour plusieurs algorithmes afin de choisir celui qui présente les meilleures performances. La recherche de grille est une technique d'optimisation des hyperparamètres qui permet de parcourir différentes combinaisons d'hyperparamètres pour chaque algorithme afin de trouver les valeurs qui maximisent les performances du modèle.

5.4.1 knn(k plus proches voisins)

KNeighborsClassifier

Entrée [42]:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import matthews_corrcoef
import matplotlib.pyplot as plt
param_grid = {'n_neighbors': [3, 5, 7, 9, 11]}
model = KNeighborsClassifier()

grid_search = GridSearchCV(model, param_grid, scoring='accuracy', cv=5)
grid_search.fit(x_train, y_train)

results = grid_search.cv_results_
params = results['params']
mean_scores = results['mean_test_score']

best_model = grid_search.best_estimator_
y_pred = best_model.predict(x_test)

mcc = matthews_corrcoef(y_test, y_pred)

print("Meilleurs paramètres :", grid_search.best_params_)
print("Meilleur score (exactitude) :", grid_search.best_score_)
print("Coefficient de corrélation de Matthews (MCC) :", mcc)
```

```
Meilleurs paramètres : {'n_neighbors': 11}
Meilleur score (exactitude) : 0.7813559322033898
Coefficient de corrélation de Matthews (MCC) : 0.5515259789104578
```

FIGURE 5.1 – knn

5.4.2 DecisionTreeClassifier

DecisionTreeClassifier

Entrée [43]:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, matthews_corrcoef
import matplotlib.pyplot as plt


param_grid = {'max_depth': [3, 5, 7, 9, 11], 'min_samples_leaf': [1, 2, 3, 4, 5]}
clf = DecisionTreeClassifier()

grid_search = GridSearchCV(clf, param_grid, scoring='accuracy', cv=5)
grid_search.fit(x_train, y_train)

results = grid_search.cv_results_
params = results['params']
mean_scores = results['mean_test_score']
x_values = list(range(len(params)))


best_clf = grid_search.best_estimator_
y_pred = best_clf.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
mcc = matthews_corrcoef(y_test, y_pred)

print("Exactitude sur l'ensemble de test :", accuracy)
print("Coefficient de corrélation de Matthews (MCC) sur l'ensemble de test :", mcc)
```

```
Exactitude sur l'ensemble de test : 0.7244094488188977
Coefficient de corrélation de Matthews (MCC) sur l'ensemble de test : 0.4532338308457711
```

FIGURE 5.2 – DecisionTreeClassifier

5.4.3 RandomForest

RandomForestClassifier

Entrée [44]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, matthews_corrcoef
import matplotlib.pyplot as plt

param_grid = {'n_estimators': [50, 100, 150, 200], 'max_depth': [3, 5, 7, 9], 'min_samples_leaf': [1, 2, 3, 4]}
clf = RandomForestClassifier()

grid_search = GridSearchCV(clf, param_grid, scoring='accuracy', cv=5)
grid_search.fit(x_train, y_train)

results = grid_search.cv_results_
params = results['params']
mean_scores = results['mean_test_score']
x_values = list(range(len(params)))
best_clf = grid_search.best_estimator_
y_pred = best_clf.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
mcc = matthews_corrcoef(y_test, y_pred)

print("Meilleurs paramètres :", grid_search.best_params_)
print("accuracy :", accuracy)
print("Coefficient de corrélation de Matthews (MCC) sur l'ensemble de test :", mcc)
```

```
Meilleurs paramètres : {'max_depth': 7, 'min_samples_leaf': 4, 'n_estimators': 100}
accuracy : 0.8031496062992126
Coefficient de corrélation de Matthews (MCC) sur l'ensemble de test : 0.6064425446353342
```

FIGURE 5.3 – RandomForest

5.4.4 Régression Logistique

LogisticRegression

Entrée [47]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score

param_grid = {'C': [0.1, 1, 10], 'penalty': ['l1', 'l2']}
clf = LogisticRegression()
grid_search = GridSearchCV(clf, param_grid, scoring='accuracy', cv=5)
grid_search.fit(x_train, y_train)
best_params = grid_search.best_params_
clf = LogisticRegression(C=best_params['C'], penalty=best_params['penalty'])
clf.fit(x_train, y_train)
y_pred = clf.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy : {:.2f}".format(accuracy*100))

mcc = matthews_corrcoef(y_test, y_pred)
print("MCC : {:.2f}".format(mcc*100))
recall = recall_score(y_test, y_pred)
print("Rappel : {:.2f}".format(recall))
precision = precision_score(y_test, y_pred)
print("Précision : {:.2f}".format(precision))
f1 = f1_score(y_test, y_pred)
print("Score F1 : {:.2f}".format(f1))
```

```
Accuracy : 78.74
MCC : 57.71
Rappel : 0.87
Précision : 0.76
Score F1 : 0.81
```

FIGURE 5.4 – Régression Logistique

5.4.5 Naives Bayes

naive_bayes

Entrée [51]:
```
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, matthews_corrcoef
import matplotlib.pyplot as plt


param_grid = {}  # Pas de paramètres spécifiques pour Naive Bayes
clf = GaussianNB()

grid_search = GridSearchCV(clf, param_grid, scoring='accuracy', cv=5)
grid_search.fit(x_train, y_train)

results = grid_search.cv_results_
params = results['params']
mean_scores = results['mean_test_score']

best_clf = grid_search.best_estimator_
y_pred = best_clf.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
mcc = matthews_corrcoef(y_test, y_pred)



print("Exactitude sur l'ensemble de test :", accuracy)
print("Coefficient de corrélation de Matthews (MCC) :", mcc)
```

```
Exactitude sur l'ensemble de test : 0.7913385826771654
Coefficient de corrélation de Matthews (MCC) : 0.611191877417172
```

FIGURE 5.5 – Naives Bayes

5.5 resultats

l'algorithme utilisé	l'accuracy	MCC
knn (k plus proches voisins)	0.78	0.55
DecisionTreeClassifier	0.72	0.45
RandomForest	0.80	0.60
Régression Logistique	0.78	0.57
Naives Bayes	0.80	0.61

Après l'utilisation de ces algorithmes knn, décision tree, random forest, naïves bayées et régression logistiques .et la comparaison entre l'accuracy et Mcc de chaque modèle j'ai choisi l'algorithme de naïves bayées parce que c'est l'algorithme qui contient l'accuracy et Mcc le plus élève.

5.6 Matrice de confusion

Entrée [387]:
```
confusion_mat = confusion_matrix(y_test, pred)
df_confusion = pd.DataFrame(confusion_mat, index=['Vrai Negatif (TN)', 'Vrai Positif (TP)'],
                            columns=['Prédit Négatif (FN)', 'Prédit Positif (FP)'])
print("Matrice de Confusion :")
print(df_confusion)
```

```
Matrice de Confusion :
                   Prédit Négatif (FN)  Prédit Positif (FP)
Vrai Negatif (TN)                   56                   27
Vrai Positif (TP)                    7                   79
```

FIGURE 5.6 – Matrice de confusion

La matrice de confusion est représentée sous la forme d'un tableau à deux dimensions, où les lignes représentent les classes réelles (vraies étiquettes) et les colonnes représentent les classes prédites par le modèle.

Chapitre 6

Déploiement

6.1 Introduction

Dans ce chapitre, on va expliquer la phase de déploiement, les étapes utilisées pour mettre en production notre modèle sélectionné, et par la suite on va exécuter notre application pour tester sa performance.

6.2 Main.py

6.2.1 La creation de l'interface

j'ai essaye de creer interface avec utilisant la bibliothèque Streamlit pour prédire le risque de crédit d'un client en fonction de différentes caractéristiques. Voici une explication de chaque partie du code :

Import des bibliothèques :j'ai importé les bibliothèques nécessaires pour mon application, notamment Streamlit pour la création de l'interface utilisateur, Pandas pour la manipulation des données, NumPy pour les calculs numériques et Pickle pour charger le modèle de prédiction.

Création de l'interface utilisateur :j'ai utilisé la fonction st.markdown pour afficher un titre de niveau 1 (h1) de couleur rouge dans votre application.

Sélection des caractéristiques : j'ai utilisé différentes fonctions de Streamlit pour permettre à l'utilisateur de sélectionner les caractéristiques du client nécessaires pour la prédiction du risque de crédit, telles que selectbox pour le genre, le statut matrimonial, le niveau d'éducation, etc. j ai également utilisé slider pour le salaire du client et du conjoint, permettant à l'utilisateur de régler les valeurs à l'aide de curseurs.

j'ai créé un dictionnaire data pour stocker les valeurs sélectionnées par l'utilisateur concernant les caractéristiques du client. Ensuite, j ai effectué des modifications sur certaines de ces valeurs pour les convertir en valeurs numériques binaires, afin de les préparer pour la prédiction du risque de crédit.

```python
import streamlit as st
import pandas as pd
import numpy as np
import pickle
import streamlit as st

st.markdown('<h1 style="color: red;">Web Application for Credit Risk</h1>', unsafe_allow_h
Credit_History = st.selectbox('Credit_History', (1, 0))
Gender=st.selectbox('Sexe',('Male','Female'))
Married=st.selectbox('Marié',('Yes','No'))
Dependents=st.selectbox('Enfants',('0','1','2','3'))
Education=st.selectbox('Education',('Graduate','Not Graduate'))
Self_Employed=st.selectbox('Salarié ou Entrepreneur',('Yes','No'))
ApplicantIncome=st.slider('Salaire du client',0,40000,200)
CoapplicantIncome=st.slider('Salaire du conjoint',0,40000,2000)
Property_Area=st.selectbox('Property_Area',('Urban','Rural'))
```

FIGURE 6.1 – Imporatation des bibliotheques et selection des caracteristiques

```python
data={
    'Gender':Gender,
    'Married':Married,
    'Dependents':Dependents,
    'Education':Education,
    'Self_Employed':Self_Employed,
    'ApplicantIncome':ApplicantIncome,
    'CoapplicantIncome':CoapplicantIncome,
    'Credit_History':Credit_History,
    'Property_Area':Property_Area
    }
```

FIGURE 6.2 – La creation d un dectionnaire

```
if data['Gender'] == 'Male':
        data['Gender'] = 1
else:
        data['Gender'] =0

if data['Married'] == 'No':
        data['Married'] = 0
else:
        data['Married'] = 1

if data['Education'] == 'Graduate':
        data['Education'] = 0
else:
        data['Education'] = 1

if data['Self_Employed'] == 'No':
        data['Self_Employed'] = 0
else:
        data['Self_Employed'] = 1

if data['Property_Area'] == 'Urban':
        data['Property_Area'] = 0
else:
        data['Property_Area'] = 1

```

FIGURE 6.3 – convertir en valeurs numériques binaires

6.2.2 Pre-traitement pour les donnes entrer par l utilsateuer

j ai chargé les données d'entraînement à partir du fichier 'X.pkl' à l'aide de la fonction pickle.load. Ces données d'entraînement doivent être utilisées pour le prétraitement et la mise à l'échelle avant de les utiliser pour effectuer des prédictions.

j'ai appliqué une mise à l'échelle MinMax aux données d'entraînement à l'aide de la classe preprocessing MinMaxScaler(). Ensuite, j'ai transformé les données d'entrée de l'utilisateur, input_df, en utilisant la même transformation MinMax pour les mettre à l'échelle dans la même plage que les données d'entraînement.

j'ai chargé le modèle de prévision à partir du fichier 'prevision_data.pkl' à l'aide de la fonction pickle.load.

j ai utilisé le modèle chargé pour effectuer des prédictions sur les données d'entrée mises à l'échelle, input_df. La variable prevision contient les prédictions du modèle pour les données d'entrée fournies par l'utilisateur.

```
X=pickle.load(open('X.pkl','rb'))
from sklearn import preprocessing

min_max_scaler = preprocessing.MinMaxScaler()
X=min_max_scaler.fit_transform(X)
input_df1= min_max_scaler.transform(input_df)


load_model=pickle.load(open('prevision_data.pkl','rb'))
prevision=load_model.predict(input_df)
```

FIGURE 6.4 – Pre-traitement pour les donnes entrer par l utilsateuer

6.2.3 La condition de prevision

```
if st.button('Checked Me'):
    if prevision==0:
            st.success("we can take the money")
            image="image/img1.png"
            st.image(image)
    else:
            st.error("we can't take you any money ")
            image="image/img2.png"
            st.image(image)
```

FIGURE 6.5 – La condition de prevision

6.3 Les interface

J'ai créé un bouton à l'aide de st.button. Lorsque l'utilisateur clique sur ce bouton, le code suivant sera exécuté pour afficher le résultat de la prédiction.

Affichage des résultats Si la valeur de prevision est égale à 0, cela signifie que le modèle prédit que on a pas un risque de crédit , et il affiche donc un message de réussite "we can take the money" en utilisant st.success. De plus, il affichee une image représentant que le crédit peut être accordé.

Sinon, si la valeur de prevision est différente de 0, cela signifie que le modèle prédit que on a un risque de crédit , il affichee donc un message d'erreur "we can't take you any money" en utilisant st.error. il affiche également une image représentant que le crédit ne peut pas être accordé.

6.3.1 Cas 1 : on a pas de risk

FIGURE 6.6 – Interface de déploiement

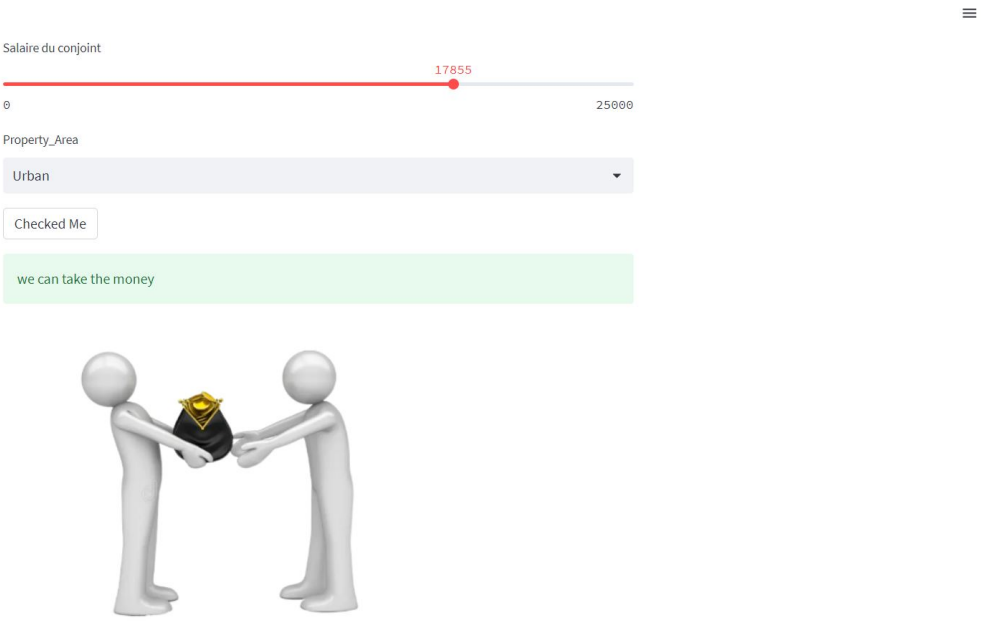

FIGURE 6.7 – Interface de déploiement

6.3.2 cas 2 : On a de risk

FIGURE 6.8 – Interface de déploiement

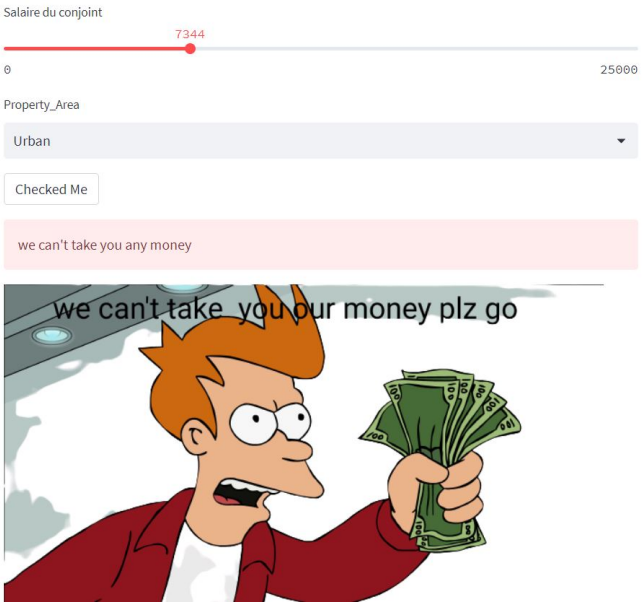

FIGURE 6.9 – Interface de déploiement

Chapitre 7

Conclusion

En conclusion, notre étude sur la classification du risque de crédit a mis en évidence l'efficacité des techniques d'apprentissage automatique dans la prédiction du risque de défaut de paiement des emprunteurs. Nous avons exploré différents algorithmes de classification tels que le k-plus proches voisins (k-NN), l'arbre de décision, le Random Forest, la régression logistique et Naive Bayes, et nous les avons évalués en utilisant des métriques appropriées.

Nos résultats ont démontré que les modèles de classification basés sur l'apprentissage automatique peuvent offrir des performances prometteuses dans la prédiction du risque de crédit. Les algorithmes tels que le Random Forest et Naive Bayes ont présenté des taux d'exactitude élevés et une capacité de généralisation satisfaisante aux données inconnues. Ces résultats suggèrent que l'utilisation de ces modèles pourrait être bénéfique pour les institutions financières dans leur processus de prise de décision concernant l'octroi de crédits.

Cependant, il est important de noter que la classification du risque de crédit est un domaine sensible et complexe, où la précision des prédictions est essentielle. Nous avons également examiné les enjeux liés à l'utilisation de données sensibles et l'importance d'une interprétabilité adéquate des modèles pour assurer la transparence et la confiance.

En termes d'apports, notre étude fournit une compréhension approfondie des techniques de classification du risque de crédit et de leur application pratique.

En fin notre étude ouvre des perspectives passionnantes pour l'application de l'apprentissage automatique dans le domaine de la classification du risque de crédit, avec le potentiel d'améliorer la gestion du risque et de contribuer à la stabilité du système financier.

Réference

https://www.bing.com/search?q=What%20is%20machine%20learning%20%28ML%29%3F

https://www.ibm.com/topics/machine-learning/

https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML/

https://www.ibm.com/topics/knn/

https://www.analyticsvidhya.com/blog/2021/04/simple-understanding-and-implementation-of-knn-algori

https://www.vtupulse.com/machine-learning/k-nearest-neighbors-algorithm-solved-example//

https://www.educba.com/k-means-clustering-algorithm/

https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm

https://scikit-learn.org/stable/index.html

https://reason.town/machine-learning-keras/

https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explaine
html#:~:text=TensorFlow%20is%20a%20Python-friendly%20open%20source%20library%20for,learning%
20and%20developing%20neural%20networks%20faster%20and%20easier.

https://machinelearningmastery.com/function-optimization-with-scipy/