الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي

UNIVERSITE BADJI MOKHTAR - ANNABA
BADJI MOKHTAR – ANNABA UNIVERSITY

جامعة باجي مختار – عنابـــــــــة

Faculté : Sciences de l'ingéniorat.

Département : Informatique.

Domaine : Mathématique et informatique.

Filière : Informatique.

Spécialité : Systèmes d'informations et décision.

Mémoire

Présenté en vue de l'obtention du Diplôme de Master

Thème :

Système de recommandation pour la gestion du bien-être dans une communauté estudiantine.

Présenté par : *FEZARI Wael.*

Encadrant : ***Pr.*** *KHADIR Mohamed Tarek.*

Jury de Soutenance :

FARAH Nadir	Professeur	Badji Mokhtar Annaba	Président
KHADIR Mohamed Tarek	Professeur	Badji Mokhtar Annaba	Encadrant
SERIDI Hassina	Professeur	Badji Mokhtar Annaba	Examinateur

Année Universitaire : 2020/2021

Résumé

La vie estudiantine est très mouvementée, entre la validation du semestre, leurs vie sociale et différents autres facteurs qui peuvent influencer leur santé mentale. Pour les aider nous avons eu l'idée de créer un système intelligent qui donne des recommandations au étudiants pour minimiser l'impact des études sur leur santé mentale tout en maximisant leur notes et niveau académique.

Abstract

Student's life is very hectic, between semester validation, their social life and various other factors that can influence their mental health. To help them we had the idea of developing an intelligent system that gives recommendations to students to minimise the impact of studying on their mental health while maximising their grades and academic performance.

Remerciements

Au terme de la rédaction de ce mémoire, c'est un devoir agréable d'exprimer en quelques lignes la reconnaissance que je dois à tous ceux qui ont contribué de loin ou de près à l'élaboration de ce travail, qu'ils trouvent ici mes vifs respects et mes profondes gratitudes. Je remercie **ALLAH** le tout puissant de m'avoir donné la santé, la force, et la volonté d'entamer et de finir ce mémoire. A mon Maître et encadrant de ce mémoire de fin d'études.
Monsieur Khadir Mohamed Tarek Professeur en Informatique.
Je ne serait jamais trouver assez de mots pour témoigner ma reconnaissance, non seulement pour l'intérêt que vous portez à ce travail, mais aussi, la spontanéité avec laquelle vous avez accepté de le diriger. Ce travail, est le fruit du suivi sans relâche dont vous avez fait preuve à mon égard. Mon séjour au sein de vos cours m'as permis d'apprécier en vous vos imminentes qualités humaines et scientifiques. Votre rigueur dans la démarche scientifique, votre amour pour le travail bien fait et votre ponctualité font de vous un maître exemplaire. Veuillez accepter cher professeur, le témoignage de mon profond respect et de ma sincère gratitude. Mes vifs remerciements pour les membres du jury à commencer par „„„ qui nous fait l'honneur de présider ce jury. Je remercie aussi très vivement „„„ de me faire l'honneur de juger ce modeste travail.

Dédicaces

Je dédie ce mémoire de fin d'études,
A MA TRÈS CHÈRE MÈRE : Tu n'as cessé de me soutenir et de m'encourager durant toutes les années de mes études, tu as toujours été présente à mes côtés pour me consoler quand il fallait. Puisse le tout puissant te donner santé, bonheur et longue vie.
A MON TRÈS CHER PÈRE : Tu as su m'inculquer le sens de la responsabilité, de l'optimisme et de la confiance en soi face aux difficultés de la vie. Je ferai toujours de mon mieux pour rester ta fierté. Que Dieu le tout puissant te préserve, t'accorde santé, bonheur, quiétude de l'esprit et te protège de tout mal.
A Mes chers frères Rami et Missou : vous deux rayonnez mon existence de bonheur, que Dieu vous procure une longue vie pleine de santé et de bonheur.
A mes merveilleux Amis qui sont pour moi une deuxième famille sur laquelle je peux toujours compter. Sans vous tous je ne serais sûrement pas arriver là ou j'en suis aujourd'hui.
Enfin, à toutes les personnes qui m'ont aidée de près ou de loin à la réalisation de ce travail.

Table des matières

Table des figures

Chapitre 1

Introduction Générale

En raison du développement rapide du niveau des études supérieurs partout dans le monde, l'attention s'est portée sur les étudiants universitaires. Sous la pression d'une vie social et d'un travail académique très lourds, de plus en plus d'étudiants ont été confrontées à des problèmes de santé mentale. Ce type de maladie qui se caractérise principalement par une perturbation émotionnelle est appelé trouble émotionnel. Les maladies émotionnelles n'affectent pas seulement la vie quotidienne du patient, mais aussi ses performances au travail ou dans l'apprentissage, ses relations sociales ou familiales, etc. Si elle n'est pas détectée à temps ou si elle ne fait pas l'objet de mesures de traitement efficaces, ce type de maladie entraînera également des maladies physiques et psychologiques plus graves, menaçant la sécurité de la vie du patient. Le principal défi dans l'étude des troubles émotionnels est de maîtriser les changements d'émotion du patient. À l'heure actuelle, la recherche sur la détection des émotions de l'utilisateur comprend principalement trois catégories, à savoir la reconnaissance des émotions basée sur les signaux audio-visuels, les signaux physiologiques et les données multimodales. Ces études se concentrent sur le niveau de détection des émotions, qui est conçu pour améliorer la précision de la reconnaissance des émotions, mais la profondeur de l'analyse des émotions n'est pas suffisante, par exemple, un résultat de reconnaissance des émotions triste, nerveux et autre passif est susceptible d'être un précurseur de la dépression ou d'autres maladies émotionnelles. Et si on pouvait prédire les problèmes émotionnelles qui peuvent tomber sur un étudiant ? mieux encore si on pouvait l'aider a les surmonter tout en gardant un niveau académique très correct ? Ceci est notre but dans ce mémoire. Grâce a une étude faite dans le campus de Dartmouth lors d'un trimestre, L'idée est de créer un système de recommandation qui aidera les étudiants a rester heureux sans affecter leur coté académique.

Chapitre 2

Vie estudiantine et systèmes de recommandations

L'augmentation massive des données structurées et non structurées disponibles sur internet a introduit le concept de Big Data qui est difficile à traiter avec les techniques traditionnelles de traitement des données [1]. Cette abondance de données non catégorisées sur internet rend la recherche d'informations utiles difficile et crée le problème de surcharge d'informations [2]. Afin de traiter ces problèmes, deux technologies Internet majeures, la recherche d'informations et les recommandations, L'exploration de données dans l'enseignement a également été prise en compte récemment. Depuis que les universités souhaitent améliorer la qualité de leur enseignement, l'utilisation de l'exploration de données dans l'enseignement supérieur pour aider les universités, les instructeurs et les étudiants à améliorer leurs performances est devenue de plus en plus attrayante pour les gestionnaires et les chercheurs universitaires. Dans ce chapitre, nous présentons un aperçu général des systèmes de recommandation, puis nous discutons de la valeur des systèmes de recommandation dans l'industrie. Nous décrivons ensuite la motivation de notre proposition de recherche, puis les buts et objectifs de la recherche proposée, enfin nous présenterons un petit état de l'art sur notre projet.

2.1 La vie éstudiantine, entre scolarité et santé mentale

Des scientifiques du monde entier étudient l'interdépendance entre les paramètres physiologiques des étudiants et leur stress mental (stress arithmétique). Ils ont tenté de déterminer dans quelle mesure le cerveau humain pouvait travailler avant d'approcher la limite de la "baisse de productivité" - c'est ainsi que les scientifiques désignaient la fatigue. À leur grande surprise, ils ont appris que le cerveau pouvait fonctionner de manière fluide et efficace après une journée de travail de huit, voire douze heures. Les psychologues affirment que la fatigue dépend beaucoup des états émotionnels et psychologiques, la majeure partie

de la fatigue est attribuable à des causes psychologiques. Les émotions peuvent déclencher, interrompre ou perturber le traitement de l'information et entraîner un traitement sélectif de l'information, ou elles peuvent organiser la mémorisation. Les émotions ont un effet sur l'apprentissage et la réussite, médiatisé par l'attention, l'autorégulation et la motivation[3]. L'ennui, le dépit, le mécontentement, et l'anxiété sont des facteurs émotionnels, qui peuvent émacier les gens et réduire leur productivité. L'anxiété, la tension et les déceptions émotionnelles sont les trois causes les plus courantes de la fatigue. Elles rendent les gens fatigués. Les personnes ayant un travail sédentaire sont surtout fatiguées en raison de facteurs psychologiques et émotionnels[3]. L'évolution des paramètres physiologiques dans le temps peut aider à déterminer le niveau d'efficacité de l'apprentissage d'un éltudiant.

2.1.1 La santé mentale

Définir le concept de Santé Mentale est un travail complexe dans le sens où il n'existe pas de limites tangibles entre le sain et le malade, de surcroit et au fil du temps, le concept de Santé Mentale a évolué, on rencontre des variations selon les époques, et la capacité des services sanitaire du moment. La qualité de vie permet d'agrandir la perception du bien-être et de l'aborder depuis une perspective plus opérationnelle et, a priori, plus mesurable. Selon l'Organisation Mondiale de Santé 1998 (OMS), "La Santé Mentale est un état sujet a des fluctuations provenant des facteurs biologiques et sociaux avec lesquels l'individu se trouve en condition de suivre une synthèse satisfaisante de ses tendances instinctives potentiellement antagoniques avec les autres, et fomenter et soutenir des relations harmonieuses avec les autres, et participer constructivement aux changements qui peuvent surgir dans son entourage physique et social". La santé mentale s'est convertie en l'un des champs de la bataille des administrations chargées de veiller à la santé dans les pays développés. Presque 165 millions d'Européens subissent un type de dérangement psychiatrique ou neurologique, comme une dépression, une insomnie, une anxiété ou une maladie d'Alzheimer. Cependant, seulement un tiers d'eux reçoivent le traitement adéquat, une donnée qui tire sur les statistiques d'absentéisme de travail et leste l'économie. (Llopis, J. 2005) [4].

2.1.2 Relation entre le coté académique et la santé mentale

Il existe une relation directement proportionnelle entre les différentes aptitudes mentales - verbales, de raisonnement et de calcul et de la performance académique des étudiants, car elle indique qu'à un plus grand développement de ces aptitudes, plus grand c'est la probabilité que l'étudiant obtient une bonne moyenne académique. (Palacio, 2007) Il a été reporté de façon réitérée que les facteurs de tension auxquels se voient soumis les universitaires, influent sur

leur rendement académique et sur leur qualité de vie. Pour (Gómez et Patiño 2002), la tension émotionnelle a comme conséquences : diminution de l'attention, difficultés dans le processus d'abstraction, dans le choix de données, dans la Résolution de problèmes et dans la prise de décision et incapacité à se souvenir d'instructions ou d'entraînements. Connaître le degré de Santé Mentale Positive des étudiants, et la participation que cela a dans l'adaptation à la vie universitaire est d'une importance primordiale puisque c'est à travers de cela que nous pouvons soutenir les programmes de rétention et de bien-être universitaire, en dirigeant sa gestion non pas tant à la partie pathologique mais plutôt dans le sens de la promotion et de prévention. Dans une étude menée auprès des étudiants universitaires dans la ville de Barranquilla, lorsqu'on les interroge sur les faibles rendements, les résultats indiquent que les élèves ne reçoivent pas une orientation professionnelle adéquate avant d'entrer dans l'enseignement supérieur qui leur permettrait d'analyser leurs capacités et leurs compétences en ce qui concerne une carrière professionnelle. (Contreras, 2007)[4].

2.2 Système de recommandation

Les systèmes de recommandation sont des applications logicielles qui sont utilisées pour recommander un produit ou un service dans le but d'optimiser certains objectifs orientés utilisateur à la lumière de l'incertitude inhérente concernant les utilisateurs et le contenu [5]. Traditionnellement, les systèmes de recommandation ont été utiles pour assister les utilisateurs dans leurs recherches dans de grands espaces d'information tels que des collections de produits (films, livres, CD de musique), des documents (articles d'actualité, textes médicaux, articles de Wikipedia), ou des utilisateurs pour la mise en relation (services de rencontres, joueurs/équipes de jeux en ligne, marchés de consommateurs à consommateurs) [6]. Un système de recommandation peut être défini comme une approche de prise de décision pour les utilisateurs dans des environnements d'information complexes [7]. Il s'agit d'une application logicielle avancée qui aide les utilisateurs à rechercher dans des dossiers de connaissances en fonction de leurs intérêts et de leurs préférences exprimés sous forme d'évaluations. Les recommandations utilisent ces évaluations pour prédire ce que l'utilisateur aimera dans le futur.

Les évaluations peuvent être collectées explicitement sous la forme d'un retour d'information basé sur l'expérience d'un élément de l'utilisateur ou implicitement en observant le comportement de recherche de l'utilisateur. Par exemple, dans le cas d'un retour d'information explicite sur Amazon, un utilisateur évalue explicitement un article après l'avoir acheté, sur la base de son expérience. En comparaison, le feedback implicite sont les cliques ou le comportement de recherche de l'utilisateur qui est observé pour associer les préférences de l'utilisateur à son profil. La recherche proposée se concentre davantage sur le retour

d'information explicite que sur le retour d'information implicite pour la raison suivante : l'utilisateur peut fournir un retour d'information explicite après avoir vécu une expérience, par exemple, un utilisateur peut évaluer un film après l'avoir regardé. Le feedback explicite fournit des données plus fiables car il n'implique pas d'extraire les préférences des actions des utilisateurs comme le fait le feedback implicite. Le retour explicite apporte également de la transparence au processus de recommandation, ce qui se traduit par une meilleure qualité de recommandation [8]. Les systèmes de recommandation s'appuient sur les théories, les algorithmes et les technologies de différents domaines tels que la recherche d'information (RI), l'intelligence artificielle (IA), l'apprentissage automatique (ML), l'interaction homme-machine (IHM) et le marketing du commerce électronique. Ils restent un sujet de recherche émergent et actif qui a attiré davantage d'attention au cours de la dernière décennie. Un système de recommandation tente d'estimer/prédire une fonction d'évaluation R sur la base d'un ensemble initial d'évaluations. La fonction de notation R peut être spécifiée à l'aide de l'équation 2.1 :

$$R = Utilisateur * element \rightarrow Notation \tag{2.1}$$

Le problème de base du système de recommandation est d'estimer ou de prédire une fonction d'utilité qui peut aider à prédire comment l'utilisateur aimera un article. Dans la fonction d'utilité, U est représenté comme un ensemble d'utilisateurs U := *utilisateurs*, I est représenté comme un ensemble d'articles recommandables I := *article recommandable* où F := *fonction d'utilité*, alors 2.2 :

$$F = U * I \rightarrow R \tag{2.2}$$

Où R : = *article recommandé* et pour chaque utilisateur u, nous voulons choisir les éléments i qui maximisent f comme le montre l'équation 2.3 :

$$u \in U \qquad i_u = argmax_f F(u, i) \tag{2.3}$$

Le paradigme traditionnel du système de recommandation comporte trois aspects principaux : utilisateur, élément et le classement. La notation, dans ce cas, représente le retour qu'un utilisateur donne a un article spécifique.

2.2.1 Techniques pour un système de recommandation :

Traditionnellement, les systèmes de recommandation visent à réduire la surcharge d'informations et agissent comme des filtres d'informations. Le système

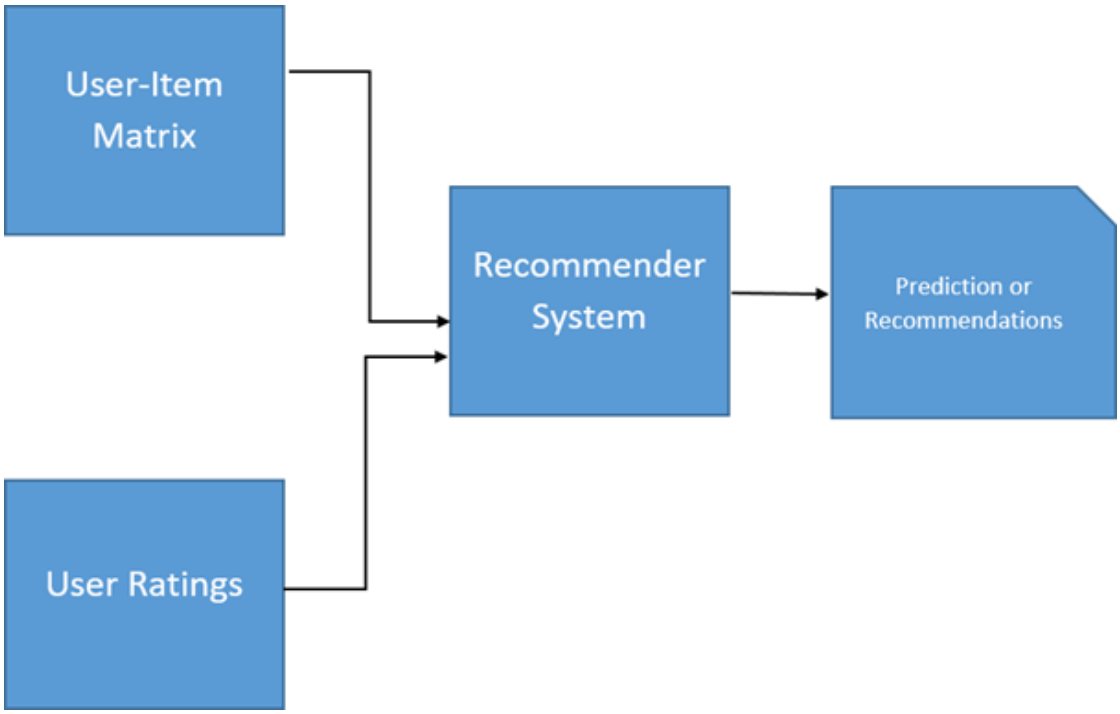


FIGURE 2.1 – Une approche traditionnelle a la recommandation

de recommandation le plus connu, et en fait l'un des premiers systèmes de recommandation commerciaux, est le système " Les clients qui ont acheté cet article ont également acheté " d'Amazon. L'objectif du système de recommandation est de rendre consommables de vastes catalogues de produits en apprenant les préférences de l'utilisateur et en les appliquant à des articles auparavant inconnus de l'utilisateur, ce qui permet de recommander ce qui a une forte probabilité d'être intéressant pour l'utilisateur cible. Pour la recommandation d'articles, les données d'apprentissage sont généralement des informations unaires sur les articles consultés, cliqués, achetés, etc. par les utilisateurs respectifs. La prédiction de notation utilise principalement les informations de notation elles-mêmes comme données d'apprentissage. [9]

Filtrage collaboratif

Au début des systèmes de recommandation, le contenu était considéré comme une donnée d'entraînement très précieuse et les ensembles de données de recherche contenaient beaucoup d'informations sur les attributs pour l'entraînement des algorithmes. Mais depuis la fin des années 90, l'approche dite du filtrage collaboratif prévaut. Le filtrage collaboratif repose sur l'hypothèse que des utilisateurs similaires aiment des choses similaires et, étant donné qu'il ne tient pas compte du contenu, il se concentre uniquement sur les notes attribuées par

le passé. Dans ce travail, nous utilisons la factorisation matricielle [10], qui est connue pour être l'une des méthodes les plus efficaces pour la prédiction des notes, surpassant les autres méthodes de point [11].

Avantages du filtrage collaboratif Les méthodes de filtrage collaboratif ne nécessitent pas d'informations sur le contenu d'un élément, la méthode reste optimale pour les recommandations [12]. Le filtrage collaboratif exploite seulement les évaluations des éléments données par les utilisateurs et n'implique aucune information sur les utilisateurs ou les éléments. Les systèmes de filtrage collaboratif peuvent générer des recommandations plus personnalisées en tenant compte de l'expérience des personnes et peuvent générer des recommandations basées sur l'expérience. Un autre avantage principal d'une approche basée sur le filtrage collaboratif est que les systèmes de recommandation peuvent recommander des articles en observant les comportements d'achat similaires des utilisateurs.

Lacunes du filtrage collaboratif Le plus grand problèmes des systèmes de recommandation et général et du filtrage collaboratif en particulier est le manque de données, il est impossible de créer un système de recommandation sans un nombre assez important de données que ce soit sur les utilisateurs ou bien sur le contenu, et sachant qu'il est impossible de réaliser une augmentation de données pour des raisons de fiabilitées on peut voir très rapidement que la source de données est très importante pour les recommandations. Cependant, les méthodes de filtrage collaboratif ne peuvent pas générer de recommandations si les utilisateurs n'ont pas fourni d'évaluation pour certains éléments ou si les éléments ne sont pas évalués par les utilisateurs. Les méthodes de filtrage collaboratif déterminent la faible précision des recommandations lorsque peu d'évaluations d'utilisateurs sont disponibles, ce qui signifie qu'un nouvel utilisateur doit fournir suffisamment d'évaluations pour les éléments avant que le système puisse recommander des éléments aux utilisateurs, ce problème est connu sous le nom de problème de démarrage à froid dans les méthodes traditionnelles basées sur les FC. Au départ, la plupart des recherches se sont concentrées sur le développement de méthodes plus efficaces pour les systèmes de recommandation, qui pourraient être utiles pour surmonter les limites des systèmes de recommandation traditionnels. En particulier, de nombreux travaux dans ce domaine ont été réalisés pour réduire la sparsité et le problème du démarrage à froid des systèmes de recommandation basés sur le filtrage collaboratif.

Filtrage Basées sur le contenu

Dans les recommandations basées sur le contenu, les informations proviennent directement du contenu des articles plutôt que des opinions de l'utilisateur. Afin de modéliser les données de préférence d'un utilisateur à partir d'exemples, un

algorithme d'apprentissage automatique est utilisé sur la base de la description du contenu, où le contenu des articles est constitué d'attributs ou de caractéristiques explicites, par exemple le genre d'un film, le film et l'année de sortie. Dans les recommandations basées sur le contenu, les éléments basés sur le profil sont recommandés à un utilisateur particulier en analysant le contenu des éléments que l'utilisateur a aimés dans le passé [13]. Par exemple, si un utilisateur a aimé un livre scientifique dans le passé, on lui recommandera d'autres livres du même type. La tâche difficile dans la recommandation basée sur le contenu est d'extraire les caractéristiques qui peuvent aider à recommander les articles aux utilisateurs. Une technique de recommandation basée sur le contenu est plus adaptée aux produits textuels, par exemple les livres et les pages Web, où les articles peuvent être décrits par leurs caractéristiques et où les utilisateurs peuvent être décrits par les mots-clés des articles qu'un utilisateur a achetés. La technique courante de recommandation basée sur le contenu consiste à établir le profil de l'utilisateur à partir de la liste des mots-clés qui peuvent décrire les articles [14].

Filtrage hybride

Les systèmes de recommandation hybrides sont plus avancés et combinent deux ou plusieurs techniques de recommandation afin d'obtenir de meilleures performances dans les systèmes de recommandation. Dans un système de recommandation hybride, la plupart du temps, un algorithme CF basé sur la mémoire est combiné avec un algorithme de CF basés sur le modèle pour surmonter les limites et les inconvénients de l'approche typique des méthodes de filtrage collaboratif.[15] La combinaison de différentes techniques de recommandation dans un système de recommandation hybride dépend de la nature de l'application finale. Cependant, certaines combinaisons de techniques de recommandation peuvent ne pas être en mesure d'utiliser les modèles ou les caractéristiques générés par l'autre technique.

2.3 Problème

Un nombre suffisant d'études ont été réalisées dans le monde entier et de nombreux systèmes ont été développés pour appliquer des technologies physiologiques afin d'établir différents états d'être humain. L'innovation globale du système de recommandation développé réside principalement dans le fait qu'il détermine automatiquement le niveau de productivité de l'apprentissage, qu'il compile de nombreuses recommandations alternatives applicables à un étudiant spécifique (comment augmenter la productivité de l'apprentissage), qu'il effectue une analyse à critères multiples de ces recommandations et qu'il sélectionne les dix recommandations les plus rationnelles pour cet étudiant. Aucun

autre système au monde ne remplit ces fonctions à ce jour. Le modèle de système de recommandation facilite une analyse intégrée des recommandations alternatives et la sélection de la plus rationnelle d'entre elles, même si l'efficacité des recommandations alternatives est souvent évaluée en tenant compte d'aspects éthiques, psychologiques, économiques, esthétiques, techniques, technologiques, de confort, sociaux et autres. Sur la base des modèles ci-dessus, le système de recommandation crée automatiquement des variantes alternatives de recommandations, génère leur analyse multicritères, détermine leur degré d'utilité et sélectionne les variantes les plus efficaces. La prochaine étape du développement du système de recommandation prévoit l'intégration de ce système avec d'autres systèmes intelligents d'analyse de la voix et d'IRIS, que les auteurs ont également développés. Une telle intégration de systèmes intelligents et physiologiques permettrait d'évaluer encore mieux les états émotionnels et la productivité d'apprentissage des étudiants et de leur soumettre des recommandations spécifiques.

2.4 Travaux éxistant

Un grand nombre de chercheurs, tel que Braak (2004)[16], ont obtenu des résultats assez encourageants prouvant que la fiabilité de l'auto-évaluation est suffisante. En raison des expériences de ces auteurs, il a été décidé d'obtenir des informations sur la productivité de l'éducation des étudiants, leur intérêt pour l'éducation et leurs états émotionnels, entre autres, au moyen de l'auto-évaluation. Les auteurs ont basé le système de recommandation qu'ils ont développé sur la présomption que, en attribuant à un étudiant des questions d'auto-évaluation et en recherchant ensuite l'interdépendance avec les paramètres physiologiques de cet étudiant à ce moment-là, il est possible de déterminer cette interdépendance de manière assez précise. Avant de commencer le travail, l'étudiant doit évaluer subjectivement sa propre productivité d'apprentissage, son intérêt pour l'apprentissage et ses états émotionnels en utilisant le questionnaire d'auto-évaluation électronique. L'étudiant procède à l'auto-analyse en attribuant des valeurs à chacun de ces paramètres sur une échelle de dix points. L'étudiant remplit le questionnaire et clique sur le bouton intitulé "Commencer à travailler". La souris de l'ordinateur physiologique et le doigt physiologique commencent à sélectionner les données physiologiques de l'élève ; celles-ci s'accumulent dans la base de données. L'un des principaux objectifs du système de recommandation est de déterminer les interdépendances entre onze données soumises sur l'état d'être d'un élève et douze paramètres physiologiques de cet élève au moment de l'auto-évaluation.[17]

L'un des algorithmes de pointe pour apprendre et prédire les comportements des utilisateurs (dans notre cas, les étudiants) dans un système de recommandation est appelé Deep Auto Encoders (DEA) .Ces algorithmes sont construits

sur des réseaux neuronaux artificiels (ANN) qui utilisent les entrées comme sorties pour former le réseau par apprentissage semi-supervisé [18]. De plus, ces algorithmes sont feedforward et utilisent l'algorithme de rétropropagation pour mettre à jour les poids des synapses avec le taux fourni par la descente de gradient stochastique. De plus, ces algorithmes fonctionnent bien avec les matrices éparses générées par les interactions élève/objet, qui ne nécessitent pas un ensemble de données étiquetées et de nombreux hyperparamètres, étant capables de traiter des données de bruit et permettent de nombreuses couches cachées, ce qui augmente l'apprentissage du réseau pour éviter le sur- et le sous-adaptation.

Les systèmes de recommandation appliqués au contexte commercial visent à augmenter les ventes en reconnaissant les préférences des utilisateurs [19] Cependant, un SEA doit inclure certaines caractéristiques pour être considéré comme efficace. Par exemple, le système devrait considérer le cas où un élève A marqué un objet qui fait que cet objet devient disponible pour un autre élève B, sans que cet élève soit conscient de cette interaction. Ainsi, le cycle des étudiants (sélectionner et noter) et du système (regrouper et filtrer) fournit les cinq composants, ou niveaux, d'un système de recommandation e-learning adaptatif. Au premier niveau, un système de recommandation fonctionne avec un filtrage collaboratif, c'est-à-dire qu'il dépend d'une interaction volontaire et responsable entre les étudiants et les objets [20] Par exemple, une mauvaise évaluation d'un livre peut générer une mauvaise recommandation, mais d'un autre côté, une évaluation faite par un groupe d'utilisateurs spécialisés (professeurs, tuteurs, réviseurs et/ou chercheurs) peut augmenter la valeur des recommandations. Dans un second temps, le système de recommandation présente une intelligence collaborative. Par exemple, un livre recommandé à l'étudiant A n'est pas lié au comportement de l'étudiant B, mais le comportement d'une certaine collectivité est responsable des recommandations .Troisièmement, un système de recommandation est sous le contrôle de l'utilisateur. Par exemple, un livre recommandé, qui fait partie de la bibliographie obligatoire, ne peut pas être rejeté par les étudiants (bien que ce livre puisse être évalué par les étudiants). Par conséquent, la couverture des recommandations est limitée. Quatrièmement, les systèmes de recommandation offrent des conseils. Par exemple, une liste d'articles recommandés, qui aide les étudiants dans leurs recherches. Cette orientation représente un problème de performance fondamental dans les sciences de l'apprentissage ; un excès d'autonomie peut être coûteux et générer des résultats de faible performance. L'idée clé est d'atteindre l'équilibre. Enfin, les systèmes de recommandation offrent une adaptabilité. Des étudiants différents ne profitent pas de la même manière des mêmes instructions ; une adaptabilité est donc nécessaire.[21]

Comme l'indiquent Manouselis et al. [22], de nombreux systèmes de recommandation ont été déployés dans l'apprentissage automatique. Concrètement,

Garc'ıa et al. [23] utilisent la fouille de règles d'association pour découvrir des informations intéressantes à travers les données de performance des étudiants sous forme de règles IF-THEN, puis génèrent des recommandations basées sur ces règles données sur les performances des étudiants sous la forme de règles SI-ALORS, puis génèrent des recommandations sur la base de ces règles ; Bobadilla et al. [24] ont proposé une équation pour le filtrage collaboratif qui incorpore les résultats des tests des apprenants dans la fonction de prédiction des éléments, Ge et al. [25] ont combiné le filtrage basé sur le contenu et le filtrage collaboratif pour personnaliser les recommandations pour un module de sélection de cours. Soonthornphisaj et al. [26] ont appliqué le filtrage collaboratif pour prédire les documents les plus appropriés pour les apprenants pour prédire les performances des étudiants, Romero et al. [27] ont comparé différentes méthodes et techniques d'exploration de données pour classer les étudiants en fonction de leurs résultats sur Moodle et des notes finales obtenues dans leurs cours respectifs. Bekele et Menzel [28] ont utilisé des réseaux bayésiens pour prédire les résultats des étudiants. Cen et al. [29] ont proposé une méthode pour améliorer un modèle cognitif, qui est un ensemble de règles/compétences codées dans des tuteurs intelligents pour modéliser la manière dont les étudiants résolvent les problèmes, en utilisant la régression logistique, arbres de décision et réseaux bayésiens) pour prédire les performances académiques ; tandis que Thai-Nghe et al [30] la prédiction des performances des étudiants en traitant le problème du déséquilibre des classes (c'est-à-dire que le rapport entre les étudiants qui réussissent et ceux qui échouent est généralement biaisé). Contrairement à la littérature, au lieu d'utiliser les méthodes traditionnelles de classification ou de régression, nous proposons d'utiliser des techniques de pointe dans les systèmes de recommandation (par exemple, la factorisation matricielle) pour prédire les performances des étudiants.

2.5 Conclusion

Dans ce chapitre, nous avons défini le problème de base de la recommandation et expliqué le fonctionnement des systèmes de recommandation traditionnels. Nous avons passé en revue les différentes techniques utilisées pour les systèmes de recommandation.

Chapitre 3

Les Réseaux de neurones profonds pour les systèmes de recommandations

La recherche dans le domaine de l'intelligence artificielle s'est considérablement développée en plus d'un demi-siècle, en grande partie grâce aux progrès récents de l'apprentissage automatique, et en particulier aux progrès réalisés grâce au deep learning. Ce domaine de recherche a de plus en plus attiré l'attention de nombreuses disciplines et chercheurs d'horizons très divers et est devenu un sujet largement interdisciplinaire. Avec le développement technologique, l'intelligence artificielle s'est répandue et s'est développée rapidement, et a également favorisé le développement d'autres disciplines. Ce chapitre fournit une description des techniques avancées d'apprentissage automatique, en s'intéressant particulièrement à l'apprentissage profond, les réseaux de neurones

3.1 Généralitées

Dans cette section nous allons discuter de quelque concepts de base essentiels pour bien comprendre le contenu du reste du mémoire.

3.1.1 Concepts d'apprentissage automatique

Dans le domaine d'informatiques, le terme "intelligence artificielle" est largement utilisé et il est profondément lié au concept d'apprentissage automatique. Sans aucun doute, le concept d'apprentissage automatique est lié à la capacité des machines à apprendre comme les humains. Étrangement, ce concept implique que la machine peut apprendre des choses comme un esprit humain. Ainsi, s'il émet des recommandations sur l'expérience d'un utilisateur, il pourrait apprendre la fois suivante les préférences de ces utilisateurs. Elle est basée sur le concept d'auto-apprentissage des machines qui peuvent apprendre et s'adapter à un autre type d'informations qu'elles reçoivent. Il est nécessaire

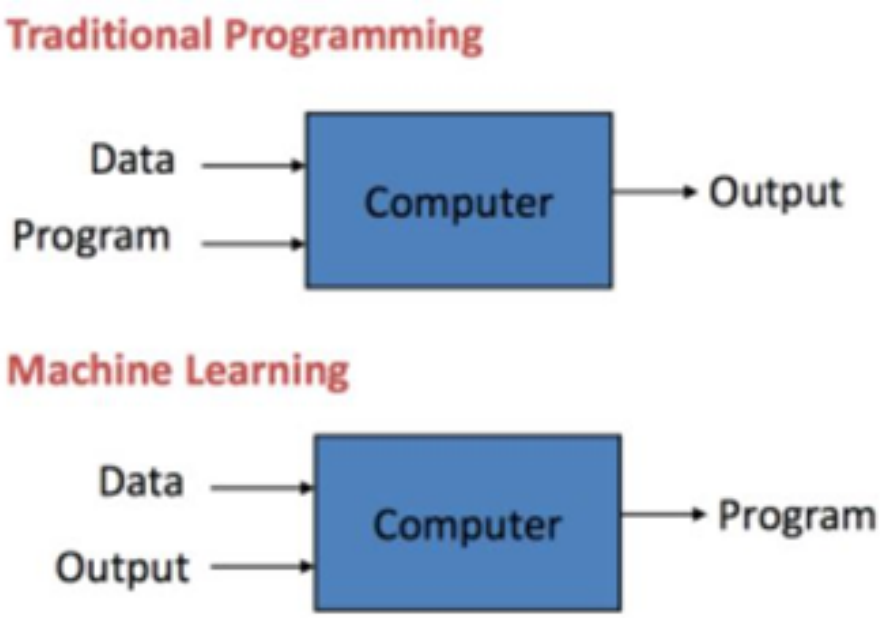


FIGURE 3.1 – La différence entre la programmation classique et l'apprentissage automatique

de comprendre le fonctionnement des machines qui sont capables d'apprendre de l'expérience. Cependant, il existe des différences entre l'apprentissage automatique et l'intelligence artificielle. Voyons comment ces deux choses se distinguent l'une de l'autre sur la base de divers facteurs. L'intelligence artificielle peut être comprise comme le concept où le concept d'intelligence est grandement encouragé. Cependant, dans l'apprentissage automatique, les gens peuvent être intéressés par l'apprentissage d'une nouvelle compétence. Par conséquent, on peut voir que ces deux concepts sont utiles à leurs propres égards. Il existe une légère différence entre les objectifs de chaque concept, l'un concernant l'apprentissage automatique et l'autre l'intelligence artificielle. La caractéristique de l'intelligence artificielle est davantage basée sur la prise de décision et l'apprentissage automatique ne peut pas décider des choses par lui-même. Il ne serait pas faux de dire que le domaine de l'intelligence artificielle joue un rôle important dans la sagesse tandis que l'apprentissage automatique concerne principalement la connaissance. Dans le domaine de l'intelligence artificielle, on peut constater qu'elle travaille sur le modèle de l'esprit humain. Cependant, le système d'apprentissage automatique fonctionne sur la base d'algorithmes qui accomplissent l'art de l'auto-apprentissage [31]. L'apprentissage automatique est un domaine d'étude dans lequel les ordinateurs sont chargés d'apprendre divers concepts. Il s'agit sans aucun doute de l'une des caractéristiques les plus passionnantes des technologies informatiques, dans lesquelles elles ne sont pas

programmées au préalable. On peut comprendre cette chose à propos de l'apprentissage automatique que ce système a tellement évolué qu'il est assez différent de la forme traditionnelle d'apprentissage automatique. Auparavant, ce système n'intégrait pas incorporait pas l'idée d'effectuer des calculs complexes, ce qui est maintenant le cas. Fondamentalement, il fonctionne sur le concept de l'auto-apprentissage afin que la machine puisse apprendre à accepter de nouveaux changements. Le raisonnement est le même que celui de l'esprit humain qui peut traiter et apprendre plus rapidement les choses qu'il rencontre. C'est pourquoi il est devenu très populaire avec le temps, car les gens ont appris à tirer profit de l'utilisation de ce système. Les tâches d'apprentissage automatique sont généralement divisées en trois grandes catégories différentes [32], selon la nature du problème rencontré 3.2 :

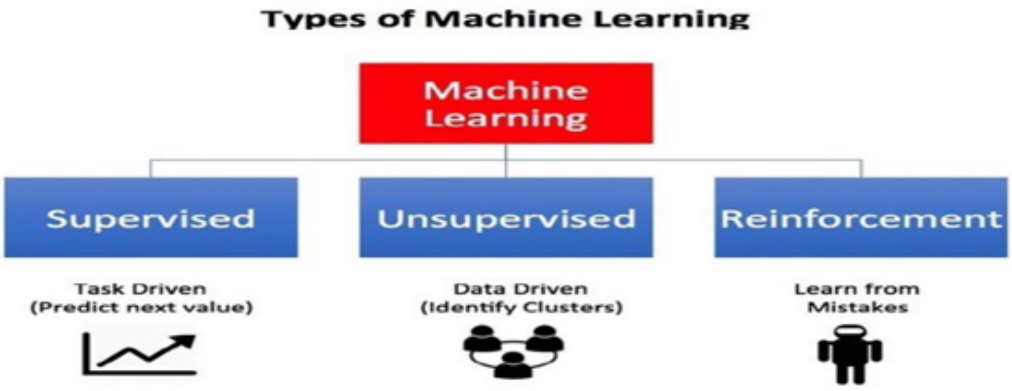


FIGURE 3.2 – Les différents type d'apprentissage automatique

L'apprentissage supervisé

La forme la plus simple de l'apprentissage automatique consiste à utiliser le paradigme supervisé de l'apprentissage automatique. Elle est basée sur l'hypothèse de l'utilisation de cartes flash qui aident les gens à apprendre des choses plus rapidement et mieux. Cependant, il existe une certaine forme d'algorithmes qui fonctionnent sur un principe similaire aux cartes flash. Elle peut également être qualifiée d'orientée vers la tâche, car elle se concentre sur un seul aspect d'une tâche. Cependant, la forme non supervisée de l'apprentissage automatique est complètement opposée à cette forme supervisée de l'apprentissage automatique. Plusieurs techniques sont utilisées dans le processus d'apprentissage automatique supervisé. Ces techniques comprennent des techniques de classification et de régression importantes pour la prédiction de des réponses d'un certain type 3.3.

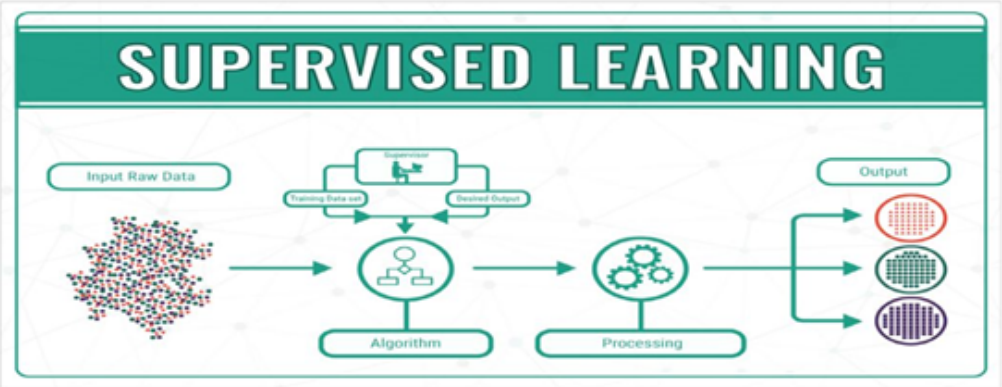


FIGURE 3.3 – Forme supervisée de l'apprentissage automatique

L'apprentissage non supervisé

Dans l'apprentissage automatique non supervisé, les étiquettes ne sont pas utilisées ou identifiées comme c'est le cas dans l'apprentissage automatique supervisé. l'apprentissage automatique supervisé. Dans cette forme d'apprentissage machine, il y a beaucoup de données disponibles et aucune direction n'est donnée à la machine. Au lieu de cela, la machine apprend automatiquement à organiser les données qui sont incorporées en elle . L'utilisation de données non étiquetées a tellement augmenté ici que les données deviennent difficiles à gérer. Cependant, cela aide les systèmes de données à devenir organisés même si aucune supervision n'est donnée à cet égard. Il y a quelques caractéristiques dans cette forme d'apprentissage, notamment les systèmes de recommandation et les journaux d'utilisateurs de groupe. Le clustering est un processus largement utilisé dans cette forme d'apprentissage qui emploie diverses applications comme la reconnaissance d'objets et l'analyse d'images. Il s'agit d'une technique qui peut détecter les détails les plus infimes et les retrouver dans divers ensembles d'images. En gardant à l'esprit ces caractéristiques de la forme non supervisée de l'apprentissage automatique 3.4, on peut trouver celle qui est la plus appropriée à utiliser. On cite les algorithmes de regroupement tels que le regroupement hiérarchique, k-means et k-NN, ainsi que les algorithmes d'association tel que l'algorithme apriori.

L'apprentissage par renforcement

Un type d'algorithme d'apprentissage automatique qui permet au modèle de décider de la meilleure action suivante en fonction de son état actuel, en apprenant des comportements qui maximiseront les récompenses. Les algorithmes de renforcement apprennent des actions optimales par essais et erreurs. Ils sont généralement utilisés en robotique, où un robot peut apprendre à éviter les

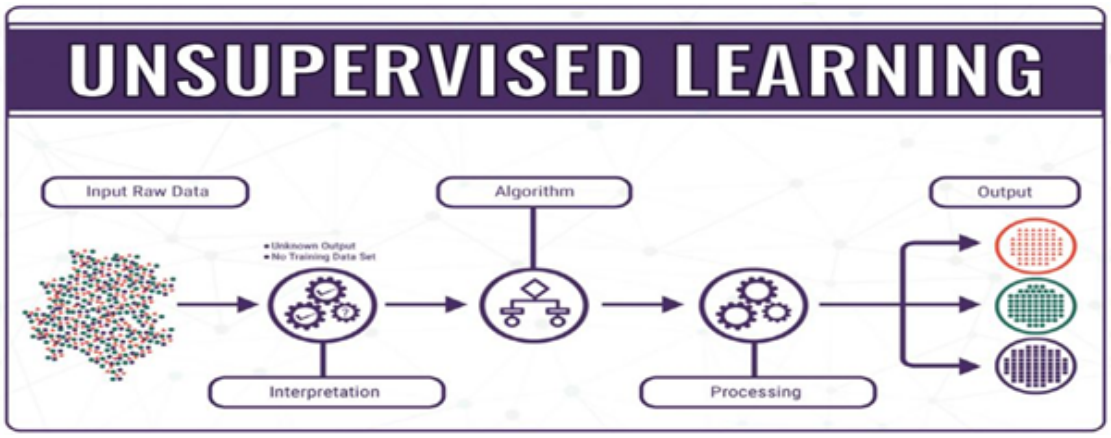


FIGURE 3.4 – Forme non-supervisée de l'apprentissage automatique

collisions avec des objets en recevant un retour d'informations négatif après avoir heurté des obstacles, ainsi que dans les jeux vidéo, où les essais et erreurs révèlent des mouvements spécifiques qui récompensent le joueur, la machine peut alors utiliser ces récompenses pour comprendre l'état optimal du jeu et choisir l'action suivante.3.5

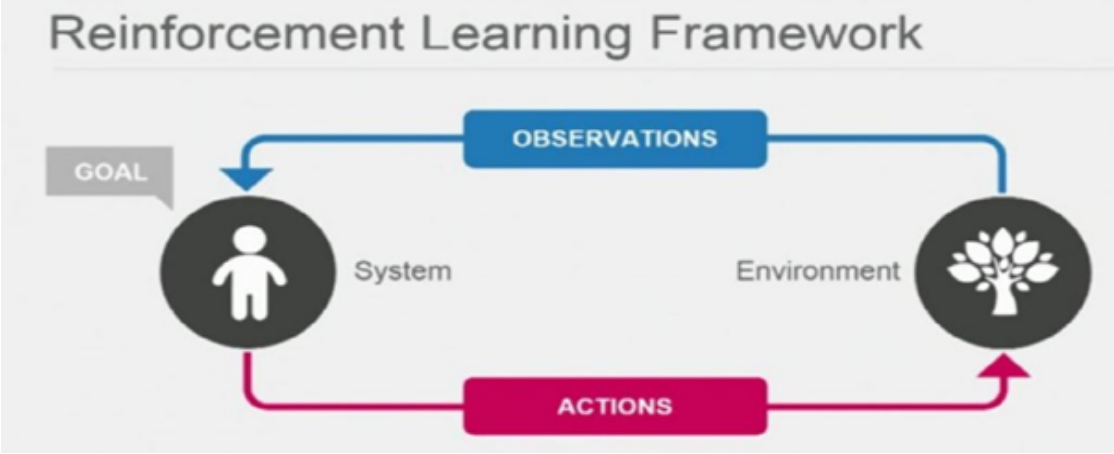


FIGURE 3.5 – L'apprentissage par renforcement

3.1.2 L'apprentissage profond

L'apprentissage profond ou le Deep learning (en anglais) est un sous-ensemble de l'apprentissage automatique 3.2 basé sur le réseau neuronal artificiel, qui s'inspire de la structure et du fonctionnement du cerveau humain. L'essence

typique de la définition de l l'apprentissage profond est qu'il apprend des représentations profondes, c'est-à-dire l'apprentissage de niveaux multiples de représentations et d'abstractions à partir de données. Aujourd'hui, l'apprentissage profond a été appliqué dans de nombreux domaines pour résoudre des tâches très complexes. Il a également été démontré que l'apprentissage profond surpasse de nombreuses techniques d'apprentissage automatique de pointe en raison de sa capacité à apprendre des caractéristiques non linéaires et complexes à partir de données massives. Le réseau neuronal multi-couches est la forme la plus courante d'architecture d'apprentissage profond. Il se compose de trois couches : une couche d'entrée, des couches cachées et une couche de sortie. Les couches d'entrée reçoivent les données d'entrée dans le réseau. Ensuite, les couches cachées, situées entre la couche d'entrée et la couche de sortie, composées d'une ou plusieurs couches, effectuent des calculs sur les entrées et génèrent la sortie vers la couche de sortie. Chaque couche est constituée d'un ou plusieurs neurones. Ces neurones sont connectés d'une couche à l'autre. La connexion entre chaque couche de nœuds contient des poids, qui stockent les modèles d'information appris.

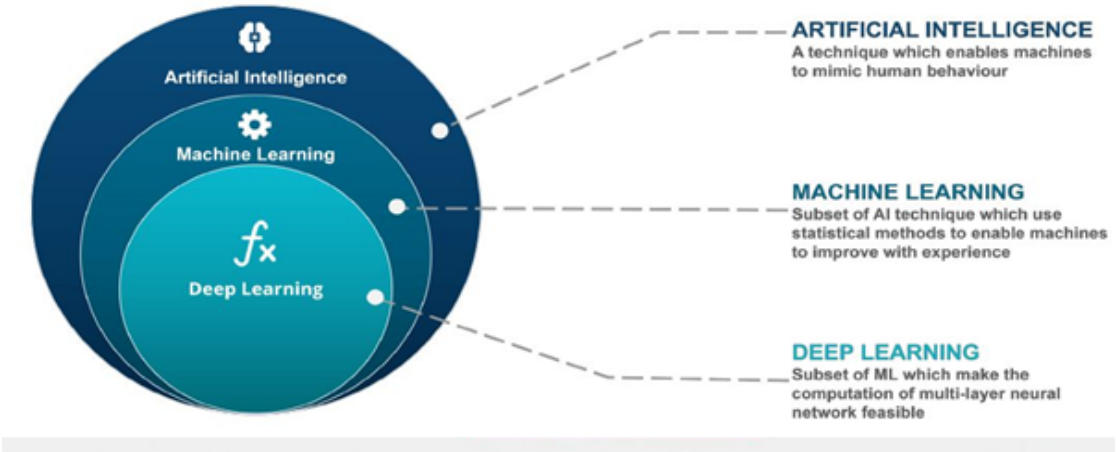


FIGURE 3.6 – L'apprentissage profond (Deep Learning)

Pour obtenir des résultats impressionnants avec des modèles d'appren tissage profond, on peut considérer que l'utilisation de données étiquetées et la puissance de calcul sont importantes. Données étiquetées ainsi que la puissance de calcul. Non seulement, ils sont utilisés dans le concept derrière les voitures sans conducteur, mais ils sont également importants en médecine et dans l'industrie.[32]

Différences entre Deep learning et machine learning Il existe une grande flexibilité dans le concept d'apprentissage profond dans lequel le monde est représenté par différents concepts sous forme hiérarchique. Il est important de

mentionner que l'apprentissage profond est une forme d'apprentissage automatique dans laquelle les gens voient la représentation du monde sous forme d'images différentes. Il faut comprendre que pour former des représentations par l'apprentissage automatique, il faut savoir que les données structurées sont importantes à cet égard. Il existe différents types de réseaux neuronaux artificiels qui jouent un rôle important dans l'établissement du dans l'établissement du concept d'apprentissage profond. Il y a donc un grand besoin d'inclure des données structurées dans le paradigme de l'apprentissage automatique, alors que l'apprentissage profond nécessite des réseaux neuronaux artificiels[32]. Le plus important des réseaux artificiels qu'il est essentiel d'inclure dans le système de deep learning est le réseau de neurones convolutifs. Un point qui doit être inclus ici pour comprendre le fonctionnement de ces systèmes est qu'ils ont besoin de données étiquetées pour percevoir les choses. Cependant, les connexions d'apprentissage profond ne nécessitent pas l'intervention humaine indispensable au fonctionnement des réseaux neuronaux. Il faut se concentrer sur la qualité des données car le résultat dépend entièrement de la qualité. Par conséquent, l'accent ne doit pas être mis sur la quantité, mais sur la qualité et la performance de ces données qui sont utilisées dans ces systèmes.

3.2 Les réseaux de neurones pronfonds

Les réseaux de neurones Un réseau neuronal artificiel (RNA) est un système non linéaire inspiré des réseaux neuronaux biologiques, c'est-à-dire du cerveau. Un réseau neuronal artificiel permet d'apprendre une correspondance d'un espace d'entrée donné à un espace de sortie souhaité. La vie d'un réseau neuronal artificiel typique est caractérisée par deux phases : la phase d'apprentissage, qui peut être supervisée, non supervisée ou hybride, et la phase de prédiction. La structure d'un RNA est généralement composée d'une couche d'entrée, d'une couche de sortie et d'une ou plusieurs couches cachées. Chaque couche d'un réseau neuronal artificiel est composée de plusieurs neurones. Chaque neurone est connecté à tous les autres neurones de la couche adjacente, c'est-à-dire qu'il prend en entrée les sorties de la couche précédente et les combine linéairement pour générer le stimulus qui alimente la fonction d'activation. Les fonctions d'activation typiques sont la sigmoïde, la tangente hyperbolique, l'unité linéaire rectifiée (ReLU).[33]

La sortie de chaque neurone est donc un nombre qui sera l'entrée des neurones de la couche suivante. L'apprentissage profond consiste à utiliser un réseau neuronal artificiel avec plusieurs couches entre l'entrée et la sortie. Une architecture profonde est appliquée dans des domaines tels que la vision par ordinateur, la reconnaissance vocale, la bio-informatique, la reconnaissance audio, etc. L'apprentissage profond est une classe d'algorithmes d'apprentissage automatique qui utilise une cascade de plusieurs couches d'unités de traitement non

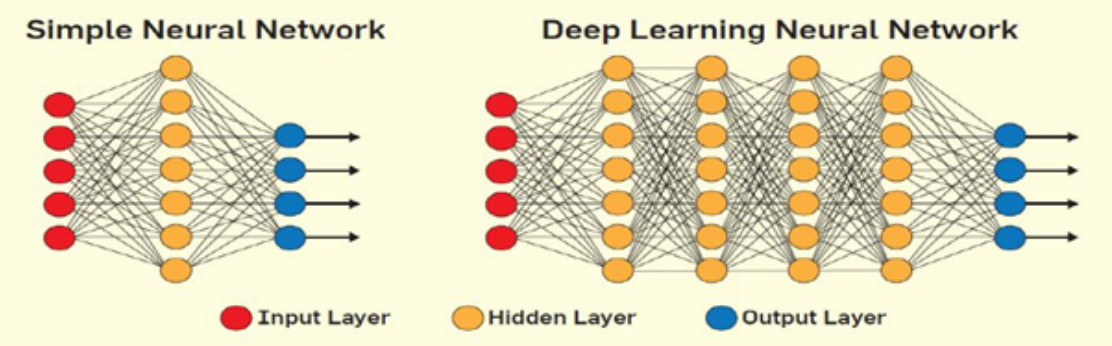


FIGURE 3.7 – l'architecture d'un Réseau de neurones

linéaires pour l'analyse des données. D'unités de traitement non linéaires pour l'extraction et la transformation de caractéristiques afin de résoudre une tâche particulière.3.8

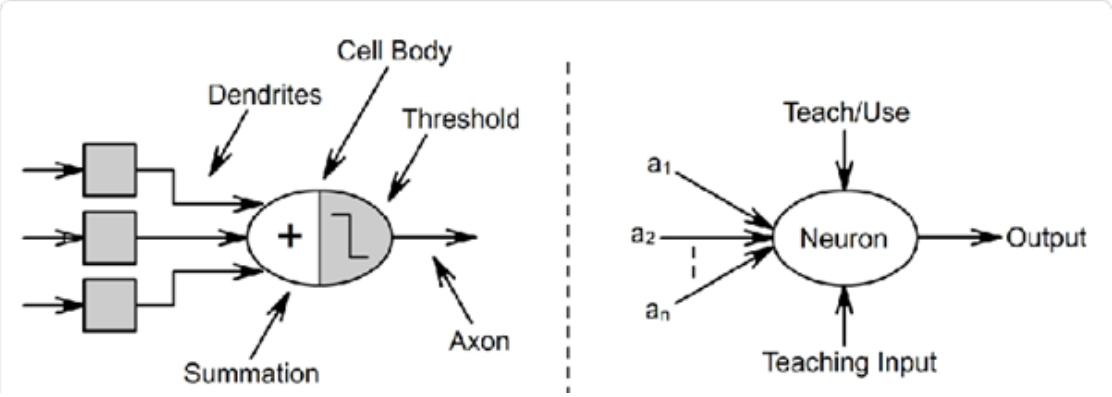


FIGURE 3.8 – L'architecture d'un neurone

Quand on parle d'apprentissage, il s'agit d'amener l'ordinateur à trouver une configuration valide de poids et de biais pour qu'il résolve réellement le problème à portée de main. Pour savoir quels poids de connexion doivent être modifiés et de combien pour effectuer correctement la tâche de classification souhaitée, nous devons utiliser un algorithme qui modifie efficacement les différents poids de connexion afin de minimiser les erreurs à la sortie. Les algorithmes d'optimisation utilisés pour l'apprentissage des modèles profonds diffèrent des algorithmes d'optimisation traditionnels à plusieurs égards. L'apprentissage automatique agit généralement de manière indirecte. Dans la plupart des scénarios d'apprentissage automatique, nous nous intéressons à une certaine mesure de performance P, qui est définie par rapport à l'ensemble de test et qui peut également être irréalisable. Nous n'optim-isons donc P qu'indirectement. Nous réduisons une autre fonction de coût L(θ) en espérant que cela

améliorera P. Cela contraste avec l'optimisation pure, où la minimisation de L(θ) est un objectif en soi. Typiquement, la fonction de coût peut être écrite sous forme d'une moyenne sur l'ensemble d'apprentissage est maintenant effectuée, visant à minimiser la fonction de coût. L'algorithme de rétro-propagation est itéré jusqu'à convergence. La vitesse de convergence dépend de :

- La complexité du problème de classification considéré (relation *E/S*)
- La complexité de la fonction d'approximation (c'est-à-dire le nombre de poids et de biais nombre de couches cachées).
- La valeur du taux d'apprentissage, un paramètre qui peut être fixé ou modifié de manière adaptative et qui indique à quel rythme les poids sont mis à jour.[34]

Trois stratégies différentes de mise à jour des poids (apprentissage) peuvent être adoptées :

- Apprentissage par modèle : pour chaque observation, on estime les erreurs et on met à jour les poids (un exemple à la fois est utilisé).
- Apprentissage par lot : les algorithmes d'optimisation qui utilisent l'ensemble de l'apprentissage sont appelés méthodes batch ou déterministes car ils traitent tous les échantillons d'apprentissage.
- Formation stochastique (mini-batch) : dans la descente de gradient stochastique (DGS), chaque échantillon d'apprentissage est traité en fonction de son importance.

Chaque mise à jour de paramètre est calculée par rapport à quelques échantillons d'entraînement (le choix des échantillons d'entraînement est important). Echantillons d'apprentissage (le choix des échantillons d'apprentissage introduit un caractère aléatoire), en soustrayant le gradient de la perte par rapport à l'échantillon d'apprentissage, mis à l'échelle par le taux d'apprentissage. L'optimisation stochastique est préférable car :

- L'algorithme converge beaucoup plus rapidement en termes de calcul total s'il est autorisé à calculer rapidement des estimations approximatives du gradient plutôt que de le calculer lentement le gradient exact.
- Il peut y avoir de la redondance dans l'ensemble d'apprentissage et nous pouvons trouver un grand nombre d'exemples qui contribuent tous de manière très similaire au gradient.

Cependant, DGS peut parfois être lent alors que ses variantes sont utilisées pour atteindre des performances de pointe. L'un des algorithmes de rétropropagation les plus populaires est Adam, ce nom dérive de l'expression "Adaptive Moments". Il s'agit d'un algorithme d'optimisation du taux d'appr entissage

adaptatif connu comme étant assez robuste au choix des hyperparamètres. [35] Un modèle d'apprentissage profond est généralement entraîné avec de nombreuses époques. Une époque est le moment où un ensemble de données entier, divisé en plusieurs lots plus petits, est passé une fois en avant et en arrière à travers le réseau neuronal. Le nombre d'itérations est défini comme le nombre de lots nécessaires pour compléter une époque. Un plus grand nombre d'époques est nécessaire car les méthodes d'optimisation sont toutes des processus itératifs et la mise à jour des paramètres en un seul passage n'est pas suffisante, ce qui pourrait entraîner un sous-ajustement. Cependant, il est important de ne pas régler excessivement le réseau sur les échantillons d'entraînement afin de laisser la place à une classification correcte des nouveaux modèles.

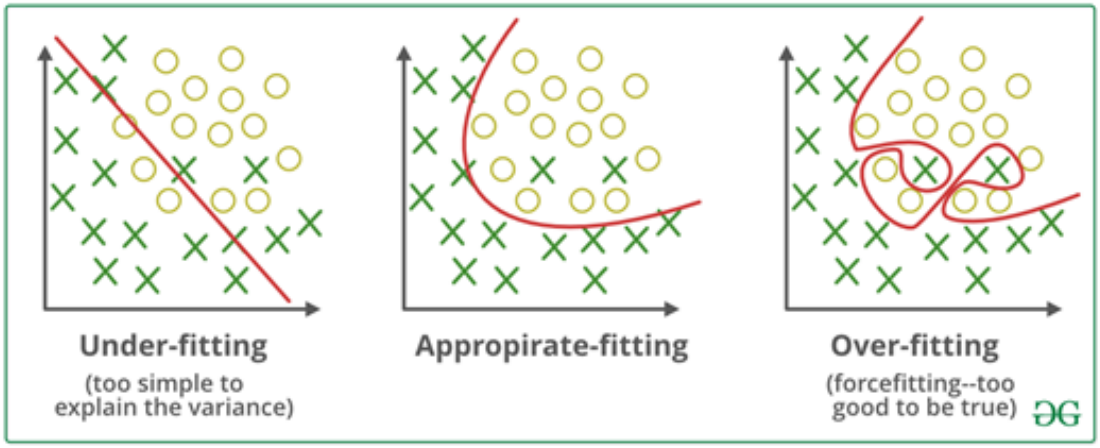


FIGURE 3.9 – Les courbes de l'apprentissage (sur-apprentissage/sous-apprentissage/bon apprentissage)

3.2.1 Fonction d'activation

La fonction d'activation est très importante pour les réseaux profonds. Elles sont des équations mathématiques qui déterminent la sortie d'un réseau neuronal. La fonction est attachée à chaque neurone du réseau et détermine s'il doit être activé ou non, en fonction de si l'entrée de chaque neurone est pertinente (selon une règle ou un seuil) pour la prédiction du modèle. Les fonctions d'activation permettent également de normaliser la sortie de chaque neurone dans une plage comprise entre 1 et 0 ou entre -1 et 1.

3.2.2 Fonction de perte

Dans l'apprentissage automatique, une fonction de perte est utilisée afin d'évaluer la qualité de modélisation d'un algorithme spécifique. Si la sortie est loin

de la réalité, les coûts sont importants et faibles si les prédictions sont proches des valeurs réelles. L'objectif principal de la formation d'un réseau neuronal est de minimiser la fonction de perte (Loss function) du réseau et de s'assurer que le réseau se généralise efficacement avec une entrée inconnue. Le choix de la fonction de coût dépendra de la question de savoir s'il s'agit d'un problème de classification ou de régression et de la sélection de l'unité de sortie.

3.2.3 Initialisation des paramètres

Les techniques d'optimisation de l'apprentissage profond sont itératives par nature, ce qui oblige l'utilisateur à définir le point de départ initial des méthodes [36]. Le choix de l'initialisation aura une influence sur la vitesse de convergence de l'apprentissage, s'il peut converger. Des recherches empiriques ont démontré qu'une technique d'initialisation bien choisie augmente considérablement le rythme de convergence [37], tandis que l'optimisation basée sur le gradient à partir d'une initialisation aléatoire peut rester bloquée près de mauvaises solutions [38].

3.3 Les Réseaux de neurones profonds et les systèmes de recommandations

Avant de plonger dans les détails des avancées récentes, il est utile de comprendre les raisons de l'application des techniques d'apprentissage profond aux systèmes de recommandation. Il est évident que de nombreux systèmes de recommandation profonde ont été proposés en l'espace de quelques années le domaine est en effet en pleine effervescence d'innovation. À ce stade, il serait facile de s'interroger sur la nécessité d'un si grand nombre d'architectures différentes, voire sur l'utilité des réseaux neuronaux dans le domaine concerné. Dans le même ordre d'idées, il serait judicieux de fournir une justification claire des raisons pour lesquelles chaque architecture proposée et le scénario pour lequel elle serait la plus bénéfique. Dans l'ensemble, cette question est très pertinente pour la problématique des tâches, des domaines et des scénarios de recommandation. L'une des propriétés les plus attrayantes des architectures neuronales est qu'elles sont :

- Différentiables de bout en bout
- fournissent des biais inductifs adaptés au type de données d'entrée.

Ainsi, s'il existe une structure inhérente que le modèle peut exploiter, les réseaux de neurones profonds devraient être utiles. Par exemple, les CNN (Convolutional Neural Networks) et les RNN (Recurrent Neural Networks) ont depuis longtemps exploité la structure intrinsèque de la vision (et/ou du le langage

humain). De même, la structure séquentielle des journaux de sessions ou de clics convient parfaitement aux biais inductifs fournis par les modèles récurrents/convolutionnels [33, 34, 35]. De plus, les réseaux de neurones profonds sont également composites dans le sens où de multiples blocs de construction neuronaux peuvent être composés en une seule (gigantesque) fonction différentiable et entraînés de bout en bout. Le principal avantage de cette approche réside dans la recommandation basée sur le contenu, ce qui est inévitable lors de la modélisation des utilisateurs et des articles sur le web, où les données multimodales sont monnaie courante. Par exemple, lorsqu'on traite des données textuelles (critiques [39], tweets [40]), des données d'images (messages sociaux, images de produits), les CNN/RNN deviennent des blocs de construction neuronaux indispensables. Dans ce cas, l'alternative traditionnelle (conception de caractéristiques spécifiques à la modalité, etc.) devient nettement moins intéressante et, par conséquent, le système de recommandation ne peut pas tirer parti de l'apprentissage conjoint (de bout en bout) de la représentation. Dans un certain sens, les développements dans le domaine des systèmes de recommandation sont également étroitement liés aux progrès de la recherche dans des modalités connexes (comme la vision ou les communautés linguistiques). Par exemple, pour traiter les critiques, il faudrait effectuer un prétraitement coûteux (par exemple, extraction de phrases clés, modélisation de thèmes, etc.), alors que les nouvelles approches basées sur l'apprentissage profond sont capables d'ingérer toutes les informations textuelles de bout en bout [39]. Dans l'ensemble, les capacités de l'apprentissage profond dans cet aspect peuvent être considérées comme un changement de paradigme et la capacité de représenter des images, du texte et des interactions dans un cadre commun unifié [41] n'est pas possible sans ces avancées récentes. En ce qui concerne l'interaction seule (c'est-à-dire le remplissage de la matrice ou le problème de classement collaboratif), l'idée clé est que les réseaux neuronaux profonds sont justifiés lorsque la complexité est énorme ou lorsqu'il y a un grand nombre d'instances d'apprentissage. Dans [42], les auteurs ont utilisé un PMC (Perceptron Multi Couches) pour approximer la fonction d'interaction et ont montré des gains de performance raisonnables par rapport aux méthodes traditionnelles telles que MF. Bien que ces modèles neuronaux soient plus performants, nous notons également que les modèles d'apprentissage automatique standard tels que BPR, MF et CML sont connus pour être raisonnablement performants lorsqu'ils sont entraînés avec une descente de gradient basée sur le Momentum sur des données d'interaction uniquement [43]. Cependant, nous pouvons également considérer ces modèles comme des architectures neuronales, puisqu'ils tirent parti des récentes avancées en matière d'apprentissage profond, telles que Adam, Dropout ou Batch de normalisation [42, 44]. Il est également facile de voir que les algorithmes de recommandation traditionnels (factorisation matricielle, machines à factoriser, etc.) peuvent également être exprimés sous forme

d'architectures neuronales/différentiables [42, 45] et entraînés de manière efficace avec un cadre tel que Tensorfow ou Pytorch, ce qui permet un entraînement efficace sur GPU et une différenciation automatique gratuite. Par conséquent, dans le climat actuel de la recherche d'aujourd'hui (et même dans l'industrie), il n'y a aucune raison de ne pas utiliser des outils basés sur l'apprentissage profond pour le développement d'un système de recommandation. Pour récapituler, nous résumons les points forts des modèles de recommandation basés sur l'apprentissage profond

- **Transformation non linéaire** Contrairement aux modèles linéaires, les réseaux neuronaux profonds sont capables de modéliser la non-linéarité des données avec des fonctions d'activations non linéaires telles que ReLu, sigmoïde, tanh, etc. Cette propriété permet de capturer les modèles d'interaction complexes et compliqués entre les éléments de l'utilisateur. Les méthodes conventionnelles telles que la factorisation matricielle, la machine à factoriser, le modèle linéaire clairsemé sont essentiellement des modèles linéaires. Par exemple, la factorisation matricielle modélise l'interaction utilisateur-article en combinant linéairement les facteurs latents de l'utilisateur et de l'article [42] ; la machine à factoriser est un membre de la famille linéaire multivariée [45] ; de toute évidence, le SLIM est un modèle de régression linéaire avec des contraintes d'éparpillement. l'hypothèse linéaire, qui constitue la base de nombreux systèmes de recommandation traditionnels, est trop simplifiée et limite considérablement l'expressivité de leur modélisation. Il est bien établi que les réseaux neuronaux sont capables d'approximer n'importe quelle fonction continue avec une précision arbitraire en variant les choix et les combinaisons d'activation [46, 47]. cette propriété permet de traiter des modèles d'interaction complexes et de refléter précisément les préférences de l'utilisateur.
- **Apprentissage des représentations** Les réseaux neuronaux profonds sont efficaces pour apprendre les facteurs explicatifs sous-jacents et les représentations utiles à partir des données d'entrée. En général, les applications du monde réel contiennent une grande quantité d'informations descriptives sur les articles et les utilisateurs est disponible dans les applications du monde réel. L'utilisation de ces informations permet d'améliorer notre compréhension des articles et des utilisateurs, ce qui se traduit par une meilleure recommandation. En tant que les avantages de l'utilisation des réseaux de neurones profonds pour aider à l'apprentissage de la représentation sont de deux ordres :

1. Elle réduit les efforts de conception manuelle des caractéristiques. Les réseaux de neurones profonds permettent d'apprendre automatiquement des caractéristiques à partir de données brutes, selon une approche supervisée ou non.
2. Ils permettent aux modèles de recommandation d'inclure des informations de contenu hétérogène, telles que du texte, des images, du son et même de la vidéo.

Les réseaux d'apprentissage profond ont fait des percées dans le traitement des données multimédias et ont montré leur potentiel dans l'apprentissage de représentations à partir de diverses sources.

- **Modélisation séquentielle** Les réseaux de neurones profonds ont montré des résultats prometteurs sur un certain nombre de tâches de modélisation séquentielle telles que la traduction automatique, la compréhension du langage naturel, la reconnaissance vocale, les chatbots, et bien d'autres. Les RNN et CNN jouent un rôle essentiel dans ces tâches. Les RNN y parviennent grâce à des états de mémoire internes, tandis que les CNN y parviennent grâce à des filtres qui glissent dans le temps. Les deux sont largement applicables et flexibles dans l'extraction de la structure séquentielle des données. La modélisation des signaux séquentiels est un sujet important pour l'exploration de la dynamique temporelle du comportement des utilisateurs et de l'évolution des articles. Par exemple, la prédiction du prochain article/panier et la recommandation basée sur la session sont des applications typiques. En tant que tels, les réseaux de neurones profonds sont parfaitement adaptés à cette tâche d'extraction de modèles séquentiels.
- **Flexibilité** Les techniques d'apprentissage profond possèdent une grande flexibilité, surtout avec l'avènement de nombreux cadres d'apprentissage profond populaires, tels que Tensorfow, Keras, Caffe, MXnet, DeepLearning4j, PyTorch, theano, etc. La plupart de ces outils sont développés de manière modulaire et bénéficient d'une communauté active et d'un soutien professionnel. la bonne modularisation rend le développement et l'ingénierie beaucoup plus efficaces. Par exemple, il est facile de combiner différentes structures neuronales pour formuler de puissants modèles hybrides, ou de remplacer un module par d'autres. Ainsi, nous pourrions facilement construire des modèles de recommandation hybrides et composites pour capturer simultanément différentes caractéristiques et différents facteurs.

3.4 Conclusion

Dans ce chapitre nous avons présenter les concepts généraux de l'intelligence artificiel, nous nous somme concentrer sur l'apprentissage profond et les réseaux de neurones tout en mentionnant leurs impacte sur les systèmes de recommandations.

Chapitre 4

Conception du système

4.1 Dataset Student Life

StudentLife est la première étude qui utilise des données de détection passive et automatique provenant des téléphones d'une classe de 48 étudiants de l'université Dartmouth pendant un trimestre de 10 semaines pour évaluer leur santé mentale (la dépression, la solitude, le stress), leurs résultats scolaires (les notes de toutes leurs classes, la moyenne générale du trimestre et la moyenne générale cumulée) et les tendances comportementales (le stress, le sommeil, les visites à la salle de sport, etc.). Une grande partie du stress et des contraintes de la vie étudiante reste cachée. En réalité, les professeurs, les doyens des étudiants et les cliniciens savent peu de choses sur leurs étudiants en dehors de la classe. Les étudiants peuvent connaître leur propre situation et leurs propres habitudes, mais en savent peu sur leurs camarades de classe. Pour faire la lumière sur la vie estudiantine. Pourquoi certains étudiants réussissent-ils mieux que d'autres ? Dans des conditions similaires, pourquoi certains individus excellent-ils alors que d'autres échouent ? Pourquoi les étudiants s'épuisent-ils, abandonnent-ils des cours, et même quittent-ils l'université ? Quel est l'impact du stress, de l'humeur, de la charge de travail, de la sociabilité, du sommeil et de la santé mentale sur les résultats scolaires (c'est-à-dire la moyenne générale) ? L'étude a utilisé une application mobile installée sur les smartphones portés par 48 étudiants pendant un trimestre de 10 semaines pour trouver des réponses à certaines de ces questions pressantes. Nous utilisons des méthodes informatiques et des algorithmes d'apprentissage automatique sur le téléphone pour évaluer les données des capteurs et faire des déductions de plus haut niveau (c'est-à-dire sommeil, sociabilité, activité, etc.) L'application StudentLife qui fonctionne sur les téléphones des étudiants a mesuré automatiquement les comportements humains suivants, 24 heures sur 24 et 7 jours sur 7, sans aucune interaction avec l'utilisateur :

- L'heure du coucher, l'heure du réveil et la durée du sommeil.
- Le nombre de conversations et la durée de chaque conversation par jour.

- L'activité physique (marcher, s'asseoir, courir, rester debout).
- L'endroit où ils se trouvaient et la durée de leur séjour (c'est-à-dire le dortoir, la classe, la fête, le gymnase).
- Le nombre de personnes entourant un étudiant au cours de la journée.
- La mobilité extérieure et intérieure (dans les bâtiments du campus).
- Le niveau de stress tout au long de la journée, de la semaine et du semestre.
- L'affect positif (comment ils se sentent bien dans leur peau).
- Les habitudes alimentaires (où et quand ils mangent).
- L'utilisation des applications.[48]

La dataset StudentLife contient des informations récoltées par l'application mentionnée ci-dessus, ces données sont répertoriées en 4 catégories (Sensing, EMA, Education, Survey) 4.1

```
dataset
  |-sensing
  |-EMA
  |-education
  |-survey
```

FIGURE 4.1 – La Racine de la Dataset StudentLife

Sensing Contient les informations récoltées par les capteurs du téléphone portable des étudiants (Wifi, GPS, etc.). Notons que les données sont organisées par fichier, c'est-à-dire chaque étudiant a son propre fichier sous chaque sous dossier. [48]

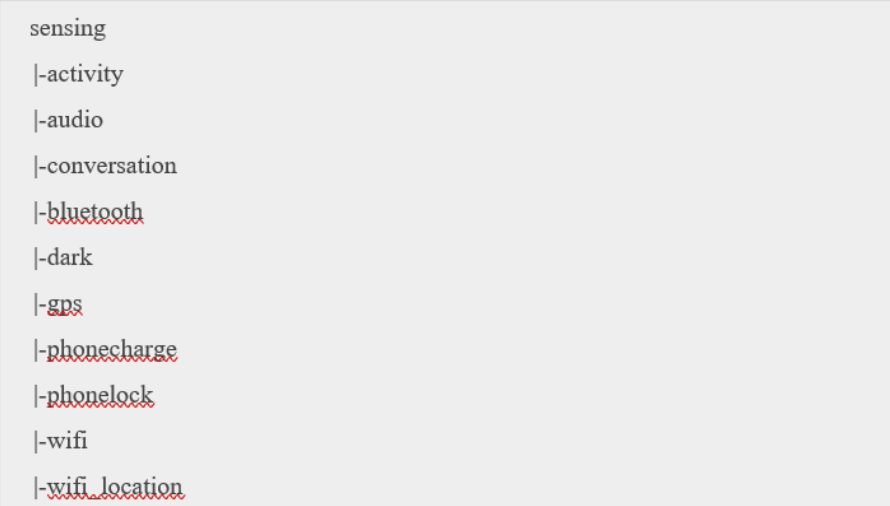


FIGURE 4.2 – Contenu du dossier Sensing

EMA Les réponses des participants sont stockées dans EMA/responses. Le nom des sous-répertoires sous EMA/responses correspond au nom de la question EMA. Par exemple, EMA/responses/Stress contient toutes les réponses des participants à l'EMA Stress. Les définitions des questions EMA se trouvent dans EMA/EMA_definition.json. [48]

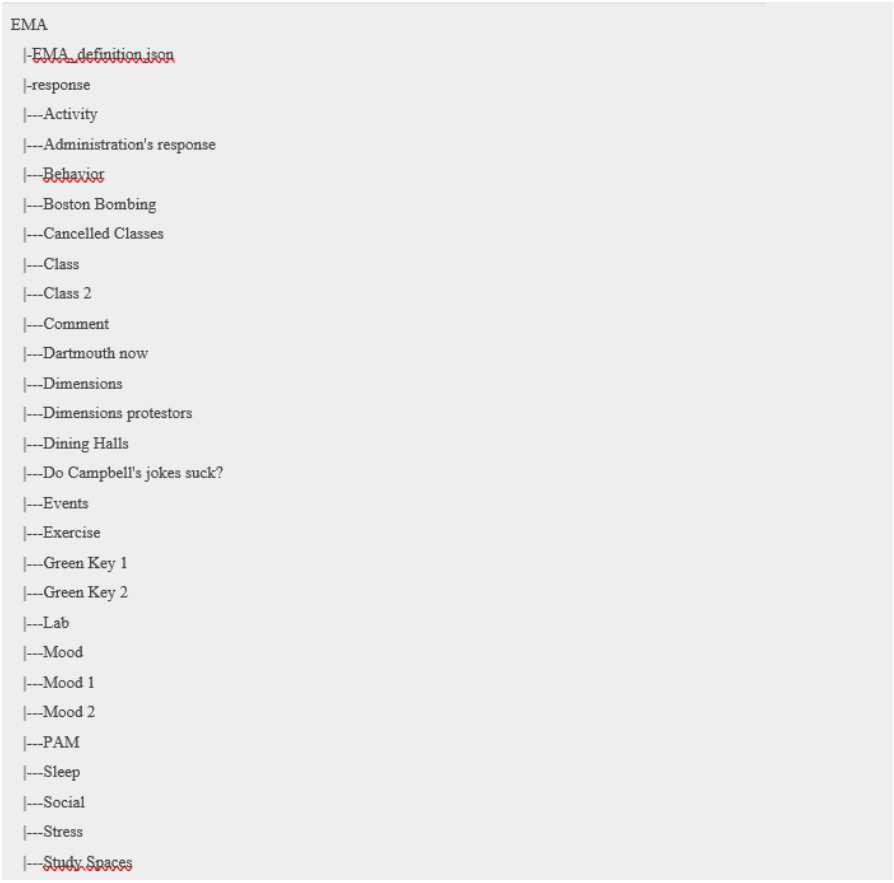


FIGURE 4.3 – Contenu du dossier EMA

Survey Contient les évaluations psychologiques des étudiants, Le répertoire est organisé par noms d'évaluation. Par exemple, vous pouvez trouver les réponses des participants à l'échelle de dépression PHQ-9 dans survey/PHQ-9.csv.[49, 50]

FIGURE 4.4 – Contenu du dossier Survey

Education Comme le nom l'indique, ce dossier contient les notes des étudiants, grades.csv est le fichier le plus important ici car contenant la note du trimestre des étudiants.[48]

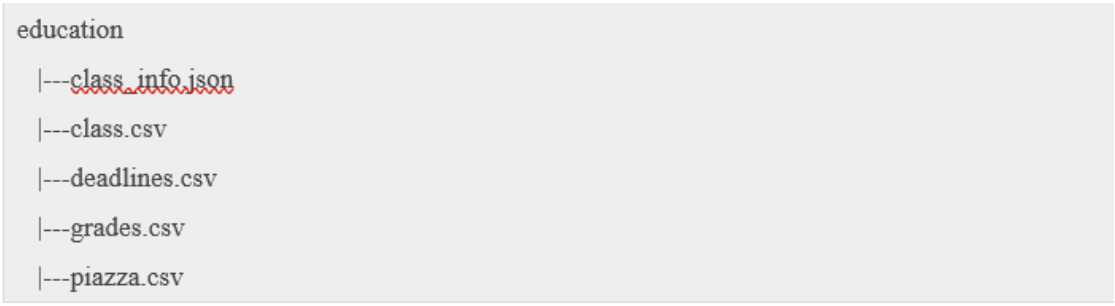


FIGURE 4.5 – Contenu du dossier Education

4.2 Environnement de travail

Sur cette partie nous allons discuter de tout les outils nécessaires pour pouvoir développer notre projet.

4.2.1 Python

Le langage utilisé dans ce travail est python 3.7. Python est un langage puissant, à la fois facile à apprendre et riche en possibilités. Plusieurs bibliothèques peuvent ainsi être installées pour, par exemple, développer des interfaces graphiques en Python. [51] Python est un langage de programmation interprété, c'est-à-dire que les instructions que vous lui envoyez sont « transcrites » en langage machine au fur et à mesure de leur lecture. D'autres langages sont appelés « langages compilés », car, avant de pouvoir les exécuter, un logiciel spécialisé se charge de transformer le code du programme en langage machine. Les avantages d'un langage interprété sont la simplicité et la portabilité. En contrepartie, un langage compilé se révélera bien plus rapide qu'un langage interprété, bien que cette différence tende à se faire de moins en moins sentir au fil des améliorations. De plus, il faudra installer Python sur le système d'exploitation que vous utilisez pour que votre ordinateur puisse assimiler votre code.

4.2.2 Google COLAB

A cause du besoin de machines très puissante pour faire ce travail nous avons utilisé est Google Colab qui est un service cloud gratuit et prend désormais en charge les GPU gratuits! vous pouvez, Améliorez vos compétences de codage dans le langage de programmation Python. Développez des applications d'apprentissage profond à l'aide de bibliothèques populaires telles que Keras, TensorFlow, PyTorch et OpenCV. La caractéristique la plus importante qui distingue Colab des autres services cloud gratuits est : Colab fournit des GPU même pour sa version gratuite, aussi il offre jusqu'au 32 GB de Ram.

4.3 Conception du système

Le système de recommandation proposée est très différent des systèmes de recommandation classique bases sur les principes de profiles similaires (Collaborative Filtering et Content Based Filtering) en raison de la dataset pauvre en échantillons (que 48 étudiants) et le nombre énorme de blanc, en plus de ça, les informations de cette dateset sont implicite ce qui rend la tâche de plus en plus difficile. Pour cela nous proposons un système de recommandation basé sur 3 étapes :

4.3.1 Analyse des données

Le but de cette partie est d'extraire les caractéristiques des étudiants qui seront considéré bon, c'est-à-dire qui ont de bonnes notes et une bonne sante mental a la fin du semestre. Pour cela nous allons faire une analyse de données sur le différent fichier tel que (sleep.json, dinning.txt, etc.) pour voir une approximation de ce que font les bons étudiants. Nous prenderont donc que les étudiants éxcellents académiquement et nous ferons un visualisation des autres données en ésperant pouvoir trouver des corrélations. Notons que les fichiers contenant beaucoup de blancs seront ignorés.

4.3.2 Score

Nous allons calculer un score pour chaque étudiant, ce score sera basé sur deux critères : les notes de l'étudiant (gpaAll surtout) et la sante mental selon les réponses de l'étudiant en question, le Score sera notre moyen pour évaluer chaque étudiant.

4.3.3 Prédiction du score

Mettre en place un modèle de prédiction qui prend en entrée les caractéristiques extraites dans l'analyse des données et prédit le score de l'étudiant, cela se fera avec un classifieur RNA profond.

4.3.4 Recommandation

Apres avoir prédit le score de l'étudiant, on fait des changements sur ses caractéristiques selon les bonnes caractéristiques extraites lors de l'analyse des données, Apres chaque changement on prédit le nouveau score, si ce dernier est supérieur au score initial alors on fait la recommandation sur la caractéristique changée, sinon on recommande à l'étudiant de continuer sur sa lancée.

4.4 Apprentissage et validation du classifieur

Pour tout ce qui concerne la prédiction du score de l'étudiant, qui est sans doute l'étape la plus importante concernant notre projet, nous allons utiliser un Réseau de neurones artificiels RNA.3

4.4.1 Architecture du Réseau de neurones

Notre modèle comprendra 3 couches : une d'entrée, une couche cachée et la couche de résultat. Pour ce qui est de la couche d'entrée chaque neurone sera

le résultat final d'une analyse de données sur un certain aspect de la vie estudiantine, par exemple son sommeil, ces données seront prétraitées pour en sortir une seule information pertinente. La taille de cette couche dépendra de ce prétraitement. Pour ce qui est de la couche cachée c'est une boite noire, on peut contrôler la taille c'est-à-dire le nombre de neurones et la fonction d'activation : Les fonctions d'activation sont spécialement utilisées dans les réseaux de neurones artificiels pour transformer un signal d'entrée en un signal de sortie qui, à son tour, sert d'entrée à la couche suivante de la pile. Dans un réseau neuronal artificiel, on calcul la somme de produits des entrées et de leurs poids correspondants et enfin, nous lui appliquons une fonction d'activation pour obtenir la sortie de cette couche particulière et la fournir comme entrée à la couche suivante.[52] Pour La couche cachée nous utiliserons une fonction d'activation ReLu, quant à la couche finale nous utiliseront la fonction Sigmoid.

4.4.2 Entrainement du Réseau de neurones

Nous allons maintenant discuter de l'entrainement du RNA avec tout ses paramètres.

Optimiser

Le choix de l'algorithme d'optimisation pour votre modèle d'apprentissage en profondeur est le facteur qui peut nous faire soit gagner soit perdre beaucoup de temps lors de l'apprentissage. L'algorithme d'optimisation choisie dans notre cas est 'ADAM' qui est une extension de la descente de gradient stochastique qui a récemment vu une adoption plus large pour les applications d'apprentissage en profondeur dans la vision par ordinateur et le traitement du langage naturel.

Loss Function

La fonction de pertes ou Loss Function est utilisé pour estimer la perte du modèle afin que les poids puissent être mis à jour pour réduire la perte lors de la prochaine évaluation.[53] Pour notre modèle nous utiliserons Cross-Entropy Loss. 4.1

$$CE = -\sum_{i}^{C} t_i \log(s_i) \tag{4.1}$$

Comme nous avons utilisé la fonction Sigmoïdale pour la couche finale, nous allons alors utiliser la fonction de pertes Binary Cross-Entropy qui est une variation de la fonction Cross-Entropy[53]. 4.2

$$CE = -\sum_{i=1}^{C=2} t_i \log(s_i) = -t_1 \log(s_1) - (1 - t_1) \log(1 - s_1) \tag{4.2}$$

Epoques

En termes de réseaux de neurones artificiels, une époque se réfère à un cycle à travers l'ensemble complet de données d'entraînement. Habituellement, la formation d'un réseau de neurones prend plus de quelques époques. En d'autres termes, si nous alimentons ce dernier avec les données d'entraînement pour plus d'une époque dans des schémas différents, nous espérons une meilleure généralisation lorsqu'il est donné une nouvelle entrée « invisible » (données de test). Une époque est souvent confondue avec une itération. Les itérations sont le nombre de lots ou d'étapes à travers des paquets partitionnés des données d'apprentissage, nécessaires pour terminer une époque. Nombre d'époques choisis dans notre cas est 100.

4.5 Conclusion

Dans ce chapitre nous avons discuté des outils nécessaires a notre travail, nous avons bien expliqué les données de notre dataset en montrant ses lacunes Et présenter l'environnement de travail. Puis nous avons élaboré une conception assez simple d'un système qui pourrait faire des recommandations adéquates utilisant cette dataset. Dans le chapitre suivant nous allons rendre vie a cette conception avec l'implémentation du système.

Chapitre 5

Implémentation

Dans ce chapitre nous allons discuter l'implémentation de notre projet, avec des captures d'écrans du code sources, des figures illustratives, des courbes et toutes les explications nécessaires. D'abord nous allons parler des données de la dataset, tout ce qui est prétraitement et chargement, ensuite on parlera du score des étudiants et du système de prédiction de ce dernier, Enfin on discutera de tout ce qui concerne les recommandations.

5.1 Analyse des données

Dans cette partie, nous ferons l'analyse de données nécessaires au système de prédiction, tout d'abord nous allons scruter la dataset car, en raison du nombre de blanc dans certaines parties, les informations seront erronées et cela aura un impacte négatif sur la prédiction et la recommandation, aussi il y a certaines parties qui ne nous intéressent pas vraiment car impossible de faire une recommandation sure. Voila la liste des informations à prendre en compte :

- Activity
- Behaviour
- Class
- Dinning
- Events
- Exercise
- Mood, Mood1, Mood2
- PhoneLocked
- Sleeping
- Stress

- Social

Après que cela soit fait nous ferons une analyse de données sur chaque partie pour extraire une ou plusieurs informations sur les étudiants et leur bien-être académique et mental, cela ce fera manuellement car nous allons visualiser chaque partie individuellement, et comme il n'y a pas vraiment beaucoup de dimensions (2 sur la plupart) on peut alors faire une analyse par observation qui sera beaucoup plus facile et contrôlée. Pour toute la partie d'analyse des données on a des snippets de code assez similaire que nous présenterons maintenant :

5.1.1 Le téléchargement et décompression de la dataset

Téléchargement Dans cette étape nous allons télécharger la dataset depuis le lien fourni et l'enregistrer sur le disque dur de notre machine virtuelle sur Google Colab. Cela est facilement achevable grave a une seule ligne de code Python, On a juste besoin du lien du fichier5.1.

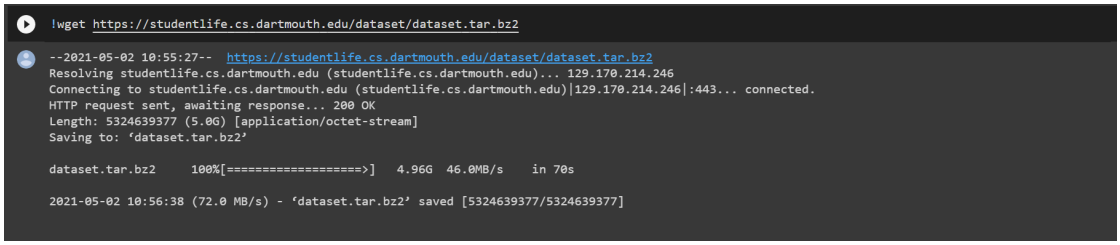


FIGURE 5.1 – Code pour le téléchargement de la Dataset sur le serveur Colab

Décompression Nous pouvons constater depuis l'éxtension du fichier téléchargé que ce dernier et compresser sur deux reprise (la première compression au format .tar et la deuxième au format .bz2), nous devons alors le décompresser pour pouvoir accéder au données, cela est aussi facilement achevable grâce à Python5.2.

FIGURE 5.2 – Code pour la décompression de la Dataset

5.1.2 Importation des bibliothèques

Pour cette partie du code nous aurons besoin que de 3 bibliothèques :

- Pandas
- Numpy
- Mtplotlib.pyplot

FIGURE 5.3 – Importation Pandas, Numpy et Mtplotlib.pyplot

5.1.3 Retourner la liste des étudiants qui sont excellent académiquement

Pour cela, nous nous tournerons vers le ficher grades, plus exactement la colonne gpa_all, Le GPA est une norme de moyenne de 0 à 4 pour noter le semestre de l'étudiant, un GPA inferieur a 2.0 est considérer comme étudiant ajourné, un bon GPA est généralement entre 3.0 et 4.0

FIGURE 5.4 – Code pour retourner la liste d'étudiants excellent académiquement

Pour notre cas, on prendra en compte que les étudiants ayant un GPA supérieur à 3.5, c'est-à-dire que les étudiants excellent (un hard treshold en quelque sortes).

5.1.4 Obtenir l'état mentale des étudiants excellents

Nous allons aussi faire l'analyse en tenons compte de la santé mentale des étudiants, pour cela nous allons nous concentrer sur le fichier BIG5 qui conti- ent une assertions de comment se sentent les étudiants grâce a une liste de questions aux qu'elles ils doivent répondre.

Maintenant qu'on a discuter de ces bouts de code qui revienne sur chaque analyse on va parler de l'analyse et visualisation des données, pour cela nous allons donner quelque exemples des 13 avec qui on va travailler car les codes sont trop répétitifs, lors de la recommandation nous allons parler des résultats de cette analyse des données.

```
big  = pd.read_csv('/content/dataset/survey/BigFive.csv')
big_suc_std = big[big['uid'].isin(list(succ_student.uid))]

big_suc_std = big_suc_std [['uid',list(big_suc_std.columns)[5] , list(big_suc_std.columns)[15]]]
big_suc_std = big_suc_std.drop_duplicates()
big_suc_std.set_index("uid", inplace = True)
cls = list(big_suc_std.columns)
big_suc_std =  big_suc_std[((big_suc_std[cls[0]] == 'Disagree Strongly') |  (big_suc_std[cls[0]] =='Disagree a little'))
& ((big_suc_std[cls[1]] == 'Disagree Strongly') |  (big_suc_std[cls[1]] =='Disagree a little'))]
```

FIGURE 5.5 – Code pour obtenir l'état mentale des étudiants excellents

5.1.5 Activity

Sur cet ensemble de fichiers nous étudierons les activités de chaque étudiant qu'il soit seul ou en groupe, Depuis le fichier EMA_Responses.json on peut alors comprendre les informations de chaque Activity_uXX.json


```
"name": "Activity",
"questions": [
    {
        "options": "[1]0-10%, [2]11-25%, [3]26-50%, [4]51-75%, [5]76-100%, ",
        "question_id": "working",
        "question_text": "alone working"
    },
    {
        "options": "[1]0-10%, [2]11-25%, [3]26-50%, [4]51-75%, [5]76-100%, ",
        "question_id": "relaxing",
        "question_text": "alone relaxing"
    },
    {
        "options": "[1]0-10%, [2]11-25%, [3]26-50%, [4]51-75%, [5]76-100%, ",
        "question_id": "other_working",
        "question_text": "with other people working"
    },
    {
        "options": "[1]0-10%, [2]11-25%, [3]26-50%, [4]51-75%, [5]76-100%, ",
        "question_id": "other_relaxing",
        "question_text": "with other people relaxing"
    },
    {
        "options": "",
        "question_id": "location",
        "question_text": "location"
    }
]
```


FIGURE 5.6 – Codage des réponses sur Activity_uXX.json

On peut maintenant bien interpréter les informations contenues sur les fichiers c'est-à-dire que chaque corp json contient une activité marquer par question_id, les activités sont : working, relaxing, working with other et relaxing with others, les réponses à ces questions sont des pourcentage qui représentent combien les étudiants font cette activité, il y a aussi un location mais cela ne nous importe peu, mais d'abord on commence par lire ces derniers, cela ce fait grâce a une boucle sur les identifiants des étudiants excellent.

```
for user in list(succ_student.uid):
  activity = pd.read_json('/content/dataset/EMA/response/Activity/Activity_'+user+'.json')
```

FIGURE 5.7 – Données Activity.json des étudiants excellents

Après cela, on commence notre visualisation des données pour mieux voir les caractéristiques recherchées grâce a l'aide de graphe et de plot 2D.

```
    if 'working' in list(activity.columns):
      activity = activity[activity['working'].notna()]
      if activity.shape[0]>0:

        x = int(i/3)
        y = i % 3
        ax[x][y].hist(activity['working'])
        ax[x][y].set_title(user)
    i += 1

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

FIGURE 5.8 – Code pour dessiner les plots de l'activité *working* des étudiants excellents

Ce bout de code va nous donner des plots sur le pourcentage de travaille des étudiants tout seuls (on a compris cela grâce au fichier EMA_responses.json), Le résultat et comme suit 5.9.

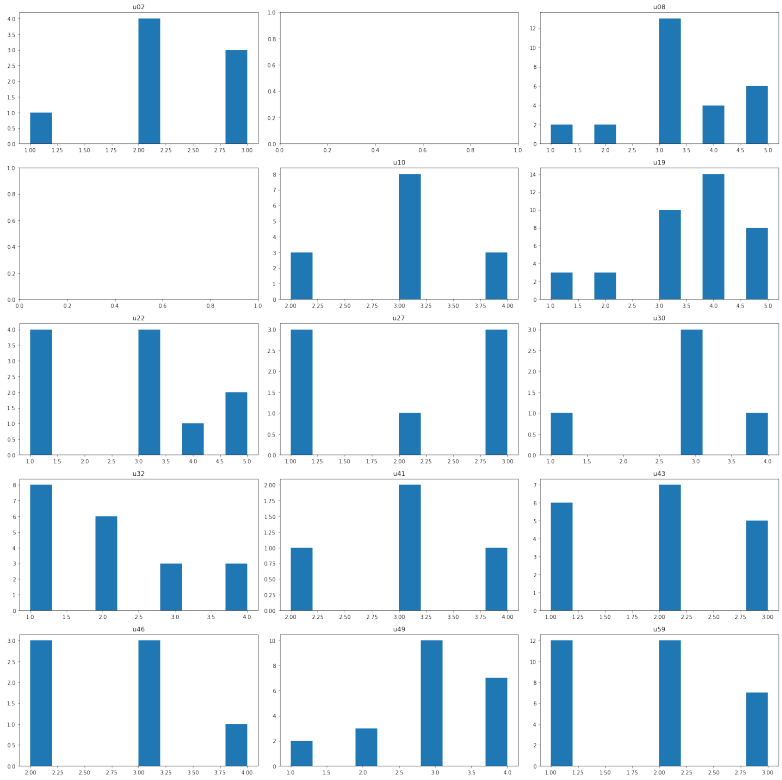


FIGURE 5.9 – Les plots de l'activité *working* des étudiants excellents

On peut voir que dans 25% à 50% du temps,la moitié des bons étudiants qui sont en bonne santé mentale travaille tout seul ce qui veut dire qu'on peut mettre cette caractéristique comme *bonne* et l'utiliser après pour donner la recommendation. On refait le même travaille pour les 3 autres choix (Relaxing, Working in Group, Relxaing in gourp), les résultats seront comme suit 5.10.

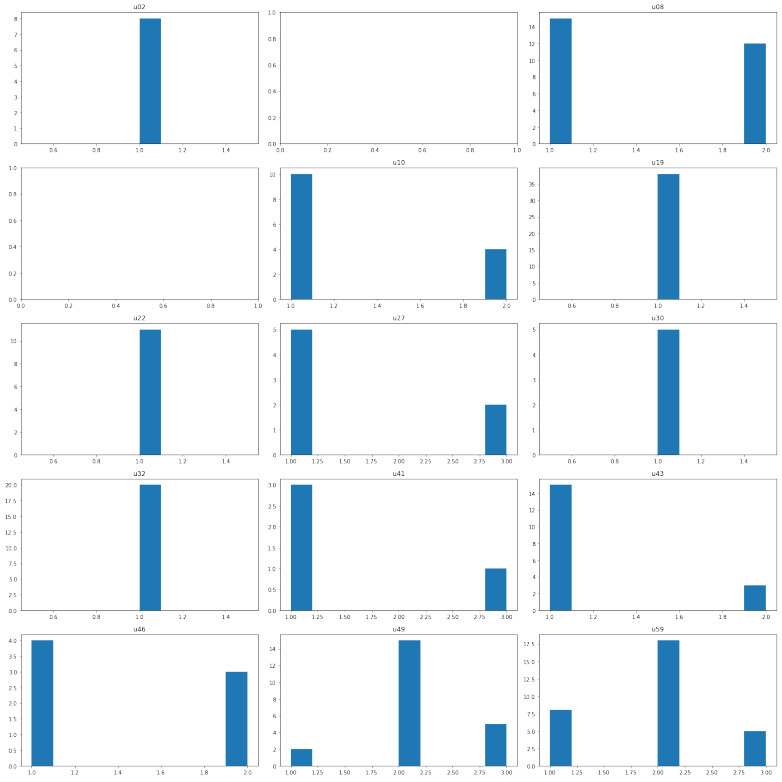


FIGURE 5.10 – Les plots de l'activité *Relaxing* des étudiants excellents

- **Relaxing** : On peut constater que la plupart des étudiants ne se relaxent pas tout seuls, ces résultat sont assez anonyme et la visualisation des données nous a beaucoup aider a prendre cette décision assez rapidement.

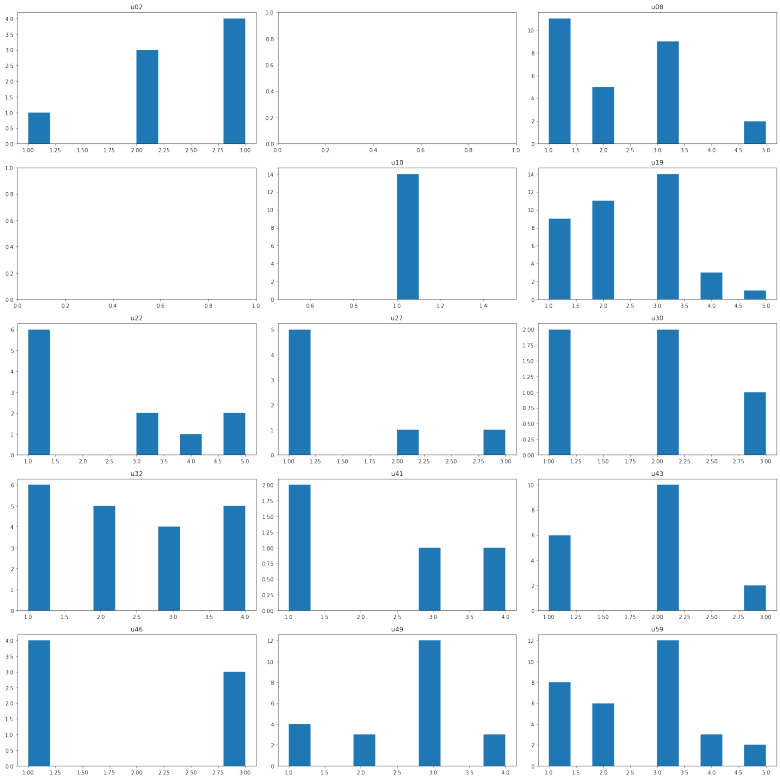


FIGURE 5.11 – Les plots de l'activité *Working in grou* des étudiants excellents

- **Working In Group** :On utilisera ces informations pour bientôt, maintenant on refait la même procédure avec le reste des fichiers, c'est pour ça que pour le reste on présentera que les informations nécessaires pour comprendre le fichier plus quelques graphes pour les résultats.

5.1.6 Behaviour

Dans cet ensemble nous allons nous intéresser au comportement de l'étudiant, afin mieux comprendre les données nous allons nous tourner encore vers EMA _responses.json, ou on peut voir sous Behaviour ce qui suit 5.12.


```
"name": "Behavior",
"questions": [
    {
        "options": "(Not at all) 1 2 3 4 5 (Extremely)",
        "question_id": "enthusiastic",
        "question_text": "In the past 15 minutes, I was extraverted, enthusiastic.
        (Scale: 1=Not at all; 2=A little; 3=somewhat; 4=very; 5=extremely)"
    },

    {
        "options": "(Not at all) 1 2 3 4 5 (Extremely)",
        "question_id": "critical",
        "question_text": "......, I was critical, quarrelsome."
    },
    {
        "options": "(Not at all) 1 2 3 4 5 (Extremely)",
        "question_id": "dependable",
        "question_text": "......, I was dependable, self-disciplined."
    },
    {
        "options": "(Not at all) 1 2 3 4 5 (Extremely)",
        "question_id": "anxious",
        "question_text": "......, I was anxious, easily upset."
    },
```


FIGURE 5.12 – Codage des réponses sur Behaviour_uXX.json

Comme on peut constater, le fichier se compose de question marquer par un ID et les réponses qui sont sur une échelle de 1 à 5. Pour le code source, c'est les mêmes étapes que pour le premier ensemble de fichier, les résultats sont comme suit 5.13.

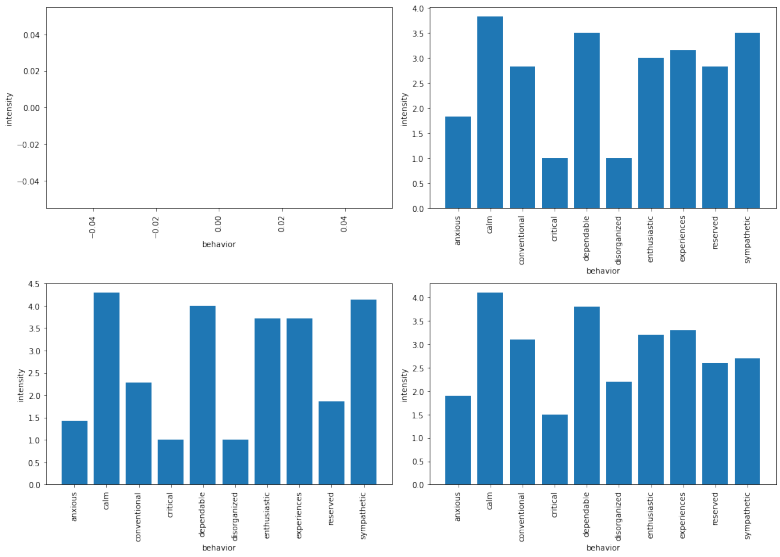


FIGURE 5.13 – Plots Behaviour

On peut voir depuis la figure 5.13 que les étudiants ont tendances a être calmes, dépendants et sympathiques.

5.1.7 Restes des résultats

Comme on peut voir l'analyse des données est assez répétitive, c'est pour cela qu'on va juste présenter le reste des résultats ci dessous :

- Durée d'exercices physiques plus que la moyenne.
- Presque zéro exercice manqué.
- Smartphone Moyennement utilisé surtout la journée.
- Très peu de classe raté.
- Durée de sommeil moyenne de 8 heures.
- Activité sociale de 4/5 personnes par jour.
- Régime alimentaire et repas toujours dans le temps.

5.2 Système de recommandation

Dans cette partie nous allons parler des pré-traitement des données, du score, de la prédiction pour enfin arriver a la recommandation.

5.2.1 Pré-traitement des données

Dans cette étape nous allons préparer les données pour pouvoir les passées vers le modèle de prédiction, pour cela nous utiliserons des moyennes et des pourcentages pour représenter les données, bien que ce ne soit pas la méthode la plus représentatives.

Les données d'entrées

Nous allons parler ici des données capturées chez les étudiants, ces données la vont êtres pré-traiter et utiliser dans le système de prédiction (le réseau de neurones profond). Les traitements que l'ont va effectuer sont assez simple, on va les présenter ci dessous.

A. Moyenne Nous représenterons cette partie par deux valeurs : la durée des exercices s'il y'en a et la cause de ne pas faire d'exercice s'il y'en a pas, nous allons enregistrer les données dans un dataframe data. Voici le code en question 5.14 Comme nous pouvons voir dans cette exemple nous avons utiliser

```
exr_dur = []
for user in list(data['uid']):

  exerc = pd.read_json('/content/dataset/EMA/response/Exercise/Exercise_'+user+'.json')
  if exerc.shape[0]>0:
    did_exercise = exerc[exerc['have']==1][['exercise','resp_time']].sort_values(by='resp_time')
    if did_exercise.shape[0]>0:
      exr_dur.append(did_exercise.exercise.mean())
    else:
      exr_dur.append(0)
  else:
      exr_dur.append(0)
data['exercise_duration'] = exr_dur
```

FIGURE 5.14 – Code pré-traitement des données durée d'exercices

la moyenne du temps d'exercices comme données, la moyenne est définit par l'équation 5.1.

$$mean = \frac{1}{n}\sum_{i=1}^{n} a_i = \frac{a_1 + a_2 + \cdots + a_n}{n} \tag{5.1}$$

B. Somme Dans ce cas-là nous avons utilisé la somme des raisons d'exercices, ou s'il n'y a pas d'exercices pour des raisons de calendrier c'est-à-dire pas intentionnelle est marquer par +1 et si les raisons sont juste de la fainéantise est marquer par -1, on voit si la somme et supérieur à 0. ensuite nous n'avons plus qu'a calculer la somme de ces valeurs.

```
cause_schedule = []
for user in list(data.uid):

  exerc = pd.read_json('/content/dataset/EMA/response/Exercise/Exercise_'+user+'.json')
  if exerc.shape[0]>0:
    did_exercise = exerc[exerc['have']==2]
    did_exercise = did_exercise[did_exercise['schedule'].notna()]
    did_exercise = did_exercise[did_exercise['schedule']==1]
    if did_exercise.shape[0]>0:
      cause_schedule.append(did_exercise[did_exercise['schedule']==1]['schedule'].sum())
    else:
      cause_schedule.append(0)
  else:
    cause_schedule.append(0)
data['scaped_exercise'] = cause_schedule
data.head()
```

FIGURE 5.15 – Code pré-traitement des données d'exercices

C. Pourcentage Dans cette partie nous allons montrer la dernière méthode de préparation de données qui est le pourcentage, pour cela l'exemple sera les fichiers Dinning.txt qui représentent les repas alimentaire de l'étudiant, depuis l'analyse de données déjà faites nous avons conclus que les étudiants ayant les meilleurs note en plus d'être dans un bon état mental mange le plus 2 repas qui sont le dîner et le déjeuner, c'est pour cela qu'on va calculer le pourcentage de dîner + déjeuner de chaque étudiant par rapport au reste des repas.

```
prcnt_lunch_sup = []
for user in list(grades.uid):
  try :
    dt = pd.read_csv('/content/dataset/dinning/'+user+'.txt',header=None,delimiter=',')
    dt.columns = ['time','food','meal']
    dt = dt.dropna()
    prcnt = 0.0
    if dt.shape[0]>0:
      dt1 = dt[(dt['meal']=='Supper') | (dt['meal']=='Lunch')]
      prcnt = dt1.shape[0] / dt.shape[0]
    prcnt_lunch_sup.append(prcnt)

  except :
    prcnt_lunch_sup.append(0.0)

data['prc_sup_lunch'] = prcnt_lunch_sup
data.head()
```

FIGURE 5.16 – Code pré-traitement des données des repas

Les données de sortie

Pour ce qui est des données de sortie, on prendra en compte le score (à voir dans la section suivante) qui sera normalisé en sortie binaire soit 1 soit 0, pour cela en prendra un thershold qu'on comparera avec le score, si ce dernier est inférieur au threshold alors l'information sera 0 sinon ça sera 1.

```
data['score'] = score.apply(lambda x : 0 if x < 9 else 1)
data.head()
```

FIGURE 5.17 – Transformation des scores en donnée binaire

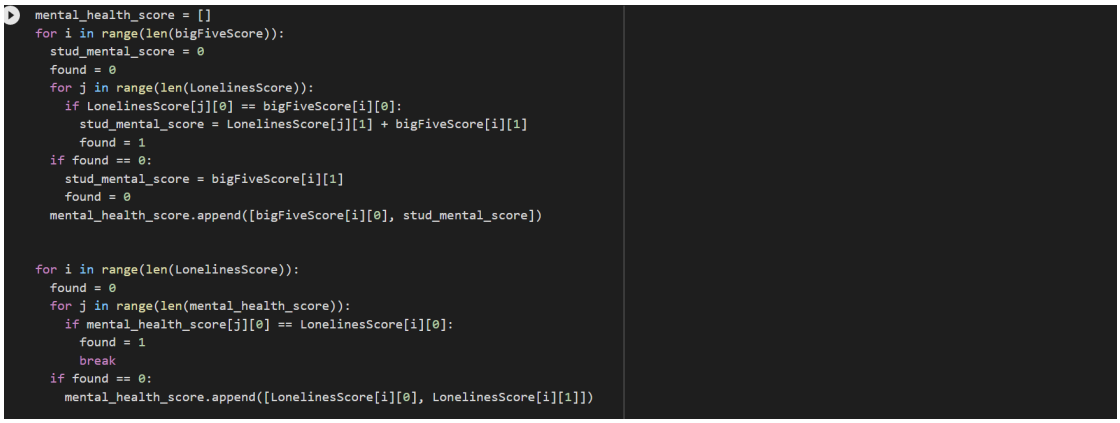
5.2.2 Score

Le score est l'un des éléments les plus important de notre système, c'est l'information sur qui retombe la décision de données une recommandation ou non. Pour notre cas nous allons faire un score assez simple, déjà on commence par calculer ce qu'on va appeler le score de santé mental, ce dernier exprime la santé mental de chaque étudiant selon leur réponses au questionnaire poser par les psychologues, notons que les réponses a ces questionnaire sont toujours 5 choix allant du plus mauvais au plus bon, alors on donne a ces réponses un barème ou la plus mauvaise question donne un -2, la moins mauvaise un -1, la réponse neutre un 0, la réponse moins bonne +1 et la réponse la plus bonne +2, on calcule alors la somme qui est notre score de santé mental. Notons que certains étudiants n'ont pas répondus aux questions alors on leur donne un score mental de 0. Ensuite il nous reste qu'a calculer la somme de chaque étudiant. On refait

```
if bigFive.iloc[i,5] == 'Agree strongly':
  mental_health -= 2
elif bigFive.iloc[i,5] == 'Agree a little':
  mental_health -= 1
elif bigFive.iloc[i,5] == 'Disagree a little':
  mental_health +=1
elif bigFive.iloc[i,5] == 'Disagree strongly':
  mental_health +=2
```

FIGURE 5.18 – Codification des réponses pour calculer le score

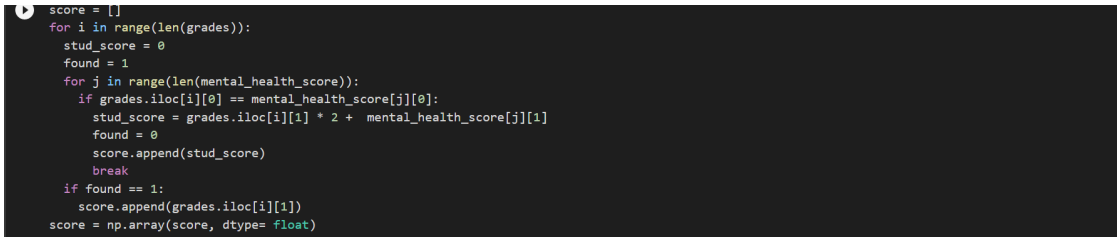
la même chose pour le reste des questions et les autres fichier qui concernent la santé mentale FlourishingScale, LonelinessScale etc.. Ensuit on a qu'a calculer le score global de l'étudiant, pour cela on multiplie le GPA par deux pour lui donner un poids plus important et on lui rajoute le score de santé mentale.



```
mental_health_score = []
for i in range(len(bigFiveScore)):
  stud_mental_score = 0
  found = 0
  for j in range(len(LonelinesScore)):
    if LonelinesScore[j][0] == bigFiveScore[i][0]:
      stud_mental_score = LonelinesScore[j][1] + bigFiveScore[i][1]
      found = 1
  if found == 0:
    stud_mental_score = bigFiveScore[i][1]
    found = 0
  mental_health_score.append([bigFiveScore[i][0], stud_mental_score])


for i in range(len(LonelinesScore)):
  found = 0
  for j in range(len(mental_health_score)):
    if mental_health_score[j][0] == LonelinesScore[i][0]:
      found = 1
      break
  if found == 0:
    mental_health_score.append([LonelinesScore[i][0], LonelinesScore[i][1]])
```

FIGURE 5.19 – Calcule du score de santé mentale pour trois réponses du fichier BigFive

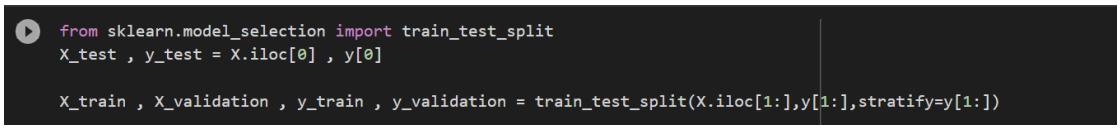


```
score = []
for i in range(len(grades)):
  stud_score = 0
  found = 1
  for j in range(len(mental_health_score)):
    if grades.iloc[i][0] == mental_health_score[j][0]:
      stud_score = grades.iloc[i][1] * 2 +  mental_health_score[j][1]
      found = 0
      score.append(stud_score)
      break
  if found == 1:
    score.append(grades.iloc[i][1])
score = np.array(score, dtype= float)
```

FIGURE 5.20 – Calcul du score de chaque étudiant

5.2.3 Prédiction du score

Pour la prédiction du score, comme nous en avons déjà parler lors de la conception, nous allons utiliser un réseau de neurones profonds, car les types de données sont assez simple. Nous allons utiliser la bibliothèque Keras de TensorFlow. Mais d'abord on doit diviser les données en deux, pour l'entraînement et pour le test, cela se fera grâce à la bibliothèque SKLearn. Pour la phase de test on ne prendra qu'un seul étudiant car la dataset est déjà assez petite pour faire de la prédiction, en plus on ne peut pas faire de l'augmentation car les données sont assez fragiles (se sont principalement des questions poser aux étudiants, en générant des données fictives on peut perdre la crédibilité du système). En-



```
from sklearn.model_selection import train_test_split
X_test , y_test = X.iloc[0] , y[0]

X_train , X_validation , y_train , y_validation = train_test_split(X.iloc[1:],y[1:],stratify=y[1:])
```

FIGURE 5.21 – Code de division des données en entraînement/validation

suite, on construit le modèle qui est un Réseau de neurones Profond. Grace à la bibliothèque Keras le tout se fait en 4 lignes, la première pour préciser qu'on va mettre des couches en séquence, ensuite on donne la couche d'entrée qui est constituée de 16 neurones, puis la couche cachée constituée de 8 neurones avec comme fonction d'activation ReLu, et enfin la couche finale de sortie qui est constituer d'un seul neurone avec comme fonction d'activation sigmoïdale. Apres avoir construit le modèle, on commence l'apprentissage, comme discu-

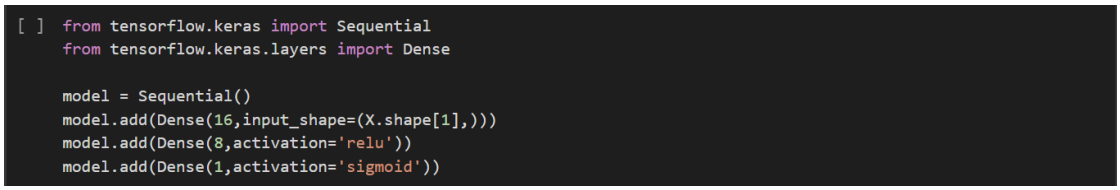


FIGURE 5.22 – Architecture du réseau de neurones

ter au paravent on utilisera l'Optimizer Adam, avec comme fonction de perte binary cross-entropy et 80 epoques. Les résultats du RNP sont assez bon avec

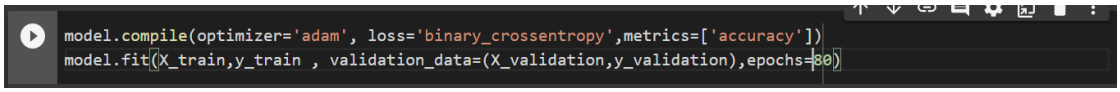


FIGURE 5.23 – Lancement de l'entraînement du réseau de neurones

une précision de 95%. C'est-à-dire que la prédiction est très bonne et peut être utilisée pour faire des recommandations.

5.2.4 Recommandations

Maintenant on arrive enfin au bout de notre projet, l'étape de recommandation est l'étape la plus simple, on commence par construire un dictionnaire de recommandation, ce dernier contient les phrases qui vont être afficher à l'étudiant, chacune de ces phrases a une relation avec une partie des informations recueillis chez les étudiants.

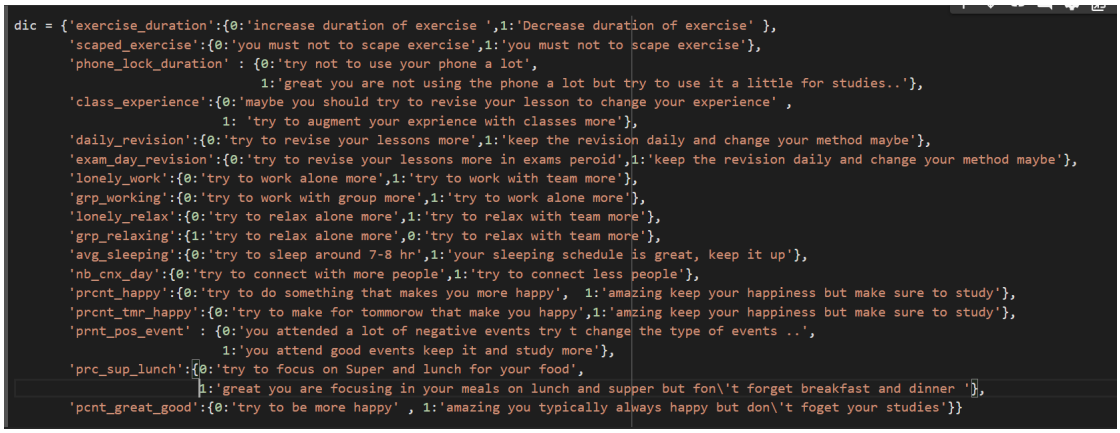


FIGURE 5.24 – Dictionnaire des recommandations

Ensuit on charge le fichier Analyse.csv, qui contient les résultats de l'analyse des données faites au paravent, ces résultats ont aussi été pré-traiter pour pouvoir être entrer dans notre modèle de prédiction. Ensuite on prédit le score de l'étu-

FIGURE 5.25 – Lecture du fichier d'analyse

diant avec ces données personnelles, on enregistre ce score et on commence par changer ces données par celle du fichier analysis.csv une par une, après chaque changement X de données on prédit le score de l'étudiant a nouveau, si ce dernier est supérieur au score initial de l'étudiant, alors on donne une recommandation adéquate au changement de donnée X qu'on a fait et cette recommandation depuis le dictionnaire qu'on a créer. 5.24

FIGURE 5.26 – Code pour la recommandation

Le résultat de l'exécution du code 5.26 sont des recommandations selon le dictionnaire créer 5.24.

```
keep the revision daily and change your method maybe
try to work with team more
try to sleep less 7-8 hr
amzing keep your happiness but make sure to study
great you are focusing in your meals on lunch and supper but fon't forget breakfast and dinner
amazing you typically always happy but don't foget your studies
```

FIGURE 5.27 – Résultat de la recommandation

5.2.5 Évaluation des recommandations

On calcule maintenant le taux de bonnes recommandations pour cette étudiant, pour cela on prend le nombre de recommandations qui donne un meilleure score et on divise par le nombre total de recommandations, n'oublions pas que ce taux est extrêmement lier a la précision du RNP. On obtient alors le résultat suivant5.28

```
prd = model.predict(x_test)
good_rec = 0
columns = list(analysis.columns)
sum = len(columns)
for i in range(len(columns)):
  atrb = columns[i]
  cp = np.copy(x_test)
  cp[0][i] = analysis[atrb][0]
  new_prd = model.predict(cp)
  if new_prd > prd:
    good_rec +=1

good_rec_pcnt = good_Rec / sum

print("Le taux de bonne recommandations pour cette etudiant est de: ",good_rec_pcnt)
```

```
Le taux de bonne recommandations pour cette etudiant est de:  0.9411764705882353
```

FIGURE 5.28 – Taux de bonnes recommandations

Chapitre 6

Conclusion et perspectives

6.1 Conclusion Générale

Dans ce mémoire nous avons taclé un sujet extrêmement intéressant surtout me concernant moi personnellement étant étudiant, l'idée d'un système de recommandation pour améliorer la vie estudiantine pourrait changer la façons dont les étudiants voient chaque semestre en leur donnant des conseils sur leur choix de chaque jours. On pourrait voir plus de productivité et moins de jeunes qui abandonnent leurs études pour des raisons de santé mentale qui se détériore... Et le fait qu'a la base, le début de StudentLife est une application mobile facilite énormément l'accès au services que l'on peut proposer dans ce mémoire, sachant que tout les jeunes de nos jours ont un smartphone. Aussi nous pouvons conclure que les systèmes de recommandations font partie des idées les plus excitantes pour le future, surtout que les plus grandes firmes tel que Facebook et Google basent de plus en plus de ressources sur ces modèles qui deviennent très performant grâce au nouvelle technologies proposées par les chercheurs.

6.2 Ce que nous réserve l'avenir ?

Malgré le grand travaille effectuer dans ce mémoire, ce projet ne reste qu'un commencement, ou même une version Beta. Mais cela n'est pas de notre faute, car la base de données recueilli par l'étude StudentLife [48] reste très pauvre en information, alors que faire ? l'idée est que cette version "Beta" de notre système de recommandation puisse attirer plus d'étudiants a utiliser l'application, cela nous donnera accès a plus de données ce qui ouvrira la porte a un vrai système de recommandation plus performant et surtout plus polyvalent que le système proposée dans ce mémoire. Avec plus de données on pourrait développer un filtrage hybride abstrait (qui était l'idée original des que j'ai vu le projet).

Bibliographie

[1] Tantan LIU, Fan WANG et Gagan AGRAWAL. « Stratified sampling for data mining on the deep web ». In : *2010 IEEE International Conference on Data Mining*. IEEE, 2010.

[2] Guandong XU. « Web mining techniques for recommendation and personalization ». Thèse de doct. Citeseer, 2008.

[3] G ADOMAVICIUS et A TUZHILIN. « Toward the next generation of recommender systems : a survey of the state-of-the-art and possible extensions ». In : *IEEE Trans. Knowl. Data Eng.* 17.6 (2005), p. 734-749.

[4] Gina Liceth Navarro BAENE. « Santé mentale positive chez des étudiants universitaires de la faculté de Médecine : étude comparative compte tenu des contextes socioculturels colombien et français ». Thèse de doct. Université Paris-Est, 2015.

[5] Tariq MAHMOOD et Francesco RICCI. « Improving recommender systems with adaptive conversational strategies ». In : *Proceedings of the 20th ACM conference on Hypertext and hypermedia - HT '09*. New York, New York, USA : ACM Press, 2009.

[6] G ADOMAVICIUS et A TUZHILIN. « Extending recommender systems : A multidi- mensional approach ». In : Seattle Washington August, 2001, p. 4-6.

[7] A M RASHID et al. « Getting to know you : learning new user preferences in recommender systems ». In : ACM, 2002, p. 127-134.

[8] Jürgen BUDER et Christina SCHWIND. « Learning with personalized recommender systems : A psychological view ». en. In : *Comput. Human Behav.* 28.1 (2012), p. 207-216.

[9] Nguyen THAI-NGHE et al. « Recommender system for predicting student performance ». en. In : *Procedia Comput. Sci.* 1.2 (2010), p. 2811-2819.

[10] Yehuda KOREN, Robert BELL et Chris VOLINSKY. « Matrix factorization techniques for recommender systems ». In : *Computer (Long Beach Calif.)* 42.8 (2009), p. 30-37.

[11] Robert M BELL et Yehuda KOREN. « Scalable collaborative filtering with jointly derived neighborhood interpolation weights ». In : *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. IEEE, 2007.

[12] Xiaoyuan SU et Taghi M KHOSHGOFTAAR. « A survey of collaborative filtering techniques ». en. In : *Adv. Artif. Intell.* 2009 (2009), p. 1-19.

[13] T BOGERS et A VAN DEN BOSCH. *Collaborative and content-based filtering for item recommendation on social bookmarking websites*. 2009.

[14] Toine BOGERS et Marijn KOOLEN. « Report on RecSys 2015 workshop on new trends in content-based recommender systems ». en. In : *SIGIR forum* 49.2 (2016), p. 141-146.
[15] S BOSTANDJIEV, J O'DONOVAN et T HÖLLERER. « Tasteweights : a visual interactive hybrid recommender system ». In : ACM, 2012, p. 35-42.
[16] J P V BRAAK. « Domains and determinants of university students' self-perceived computer competence ». In : *Computers & Education* 43.3 (2004), p. 299-312.
[17] Arturas KAKLAUSKAS et al. « Recommender system to research students' study efficiency ». en. In : *Procedia Soc. Behav. Sci.* 51 (2012), p. 980-984.
[18] Julio BARBIERI et al. « Autoencoders and recommender systems : COFILS approach ». In : *Expert Syst. Appl.* 89 (2017), p. 81-90.
[19] S SEDHAIN et al. « Autorec : Autoencoders meet collaborative filtering ». In : 2015-05, p. 111-112.
[20] C KRAUSS. « Smart learning : Time-dependent context-aware learning object recommendations ». In : 2016-03.
[21] Everton GOMEDE, Rodolfo Miranda de BARROS et Leonardo de Souza MENDES. « Deep auto encoders to adaptive E-learning recommender system ». en. In : *Computers and Education : Artificial Intelligence* 2.100009 (2021), p. 100009.
[22] Nikos MANOUSELIS et al. « Recommender systems in technology enhanced learning ». In : *Recommender systems handbook*. Springer, 2011, p. 387-415.
[23] Enrique GARCÍA et al. « An architecture for making recommendations to courseware authors using association rule mining and collaborative filtering ». In : *User Modeling and User-Adapted Interaction* 19.1-2 (2009), p. 99-132.
[24] JESUS BOBADILLA, Francisco SERRADILLA, Antonio HERNANDO et al. « Collaborative filtering adapted to recommender systems of e-learning ». In : *Knowledge-Based Systems* 22.4 (2009), p. 261-265.
[25] Ge LIANG, Kong WEINING et Luo JUNZHOU. *Courseware recommendation in e-learning system*. Springer, 2006.
[26] Nuanwan SOONTHORNPHISAJ, Ekkawut ROJSATTARAT et Sukanya YIM-NGAM. « Smart e-learning using recommender system ». In : *International conference on intelligent computing*. Springer. 2006, p. 518-523.
[27] Cristóbal ROMERO et al. « Data mining algorithms to classify students ». In : *Educational data mining 2008*. 2008.
[28] Rahel BEKELE et Wolfgang MENZEL. « A bayesian approach to predict performance of a student (bapps) : A case with ethiopian students ». In : *algorithms* 22.23 (2005), p. 24.
[29] Hao CEN, Kenneth KOEDINGER et Brian JUNKER. « Learning factors analysis–a general method for cognitive model evaluation and improvement ».

In : *International Conference on Intelligent Tutoring Systems*. Springer. 2006, p. 164-175.

[30] Nguyen THAI-NGHE, Andre BUSCHE et Lars SCHMIDT-THIEME. « Improving academic performance prediction by dealing with class imbalance ». In : *2009 Ninth international conference on intelligent systems design and applications*. IEEE. 2009, p. 878-883.

[31] A BURKOV. *The hundred-page machine learning book*. T. 1. Canada : Andriy Burkov, 2019.

[32] Nur Farhana HORDRI et al. « Deep learning and its applications : a review ». In : 2016.

[33] Balázs HIDASI et al. « Session-based recommendations with recurrent neural networks ». In : (2015). eprint : 1511.06939.

[34] J TANG et K WANG. « February). Personalized top-n sequential recommendation via convolutional sequence embedding ». In : 2018, p. 565-573.

[35] C Y WU et al. « February). Recurrent recommender networks ». In : 2017, p. 495-503.

[36] Ian GOODFELLOW, Yoshua BENGIO et Aaron COURVILLE. *Deep Learning*. London, England : MIT Press, 2016.

[37] X GLOROT et Y BENGIO. « Understanding the difficulty of training deep feedforward neural networks ». In : 2010-03, p. 249-256.

[38] Y BENGIO et al. « Greedy layer-wise training of deep networks ». In : 2007, p. 153-160.

[39] Lei ZHENG, Vahid NOROOZI et Philip S YU. « Joint Deep Modeling of Users and Items Using Reviews for Recommendation ». In : 2017.

[40] Y GONG et Q ZHANG. « Hashtag recommendation using attention-based convolutional neural network ». In : 2016-07, p. 2782-2788.

[41] Y ZHANG et al. « Joint representation learning for top-n recommendation with heterogeneous information sources ». In : 2017-11, p. 1449-1458.

[42] X HE et al. « Neural collaborative filtering ». In : 2017-04, p. 173-182.

[43] Y TAY, L ANH TUAN et S C HUI. « Latent relational metric learning via memory-based attention for collaborative ranking ». In : 2018-04, p. 729-739.

[44] Shuai ZHANG et al. « NeuRec : On nonlinear transformation for personalized ranking ». In : (2018). eprint : 1805.03002.

[45] Xiangnan HE et Tat-Seng CHUA. « Neural factorization machines for sparse predictive analytics ». In : *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '17*. New York, New York, USA : ACM Press, 2017.

[46] Kurt HORNIK. « Approximation capabilities of multilayer feedforward networks ». en. In : *Neural Netw.* 4.2 (1991), p. 251-257.

[47] Kurt HORNIK, Maxwell STINCHCOMBE et Halbert WHITE. « Multilayer feedforward networks are universal approximators ». en. In : *Neural Netw.* 2.5 (1989), p. 359-366.

[48] *StudentLife Study*. `https://studentlife.cs.dartmouth.edu/`. Accessed : 2021-7-5.

[49] D J FRANCE et al. « Acoustical properties of speech as indicators of depression and suicidal risk ». en. In : *IEEE Trans. Biomed. Eng.* 47.7 (2000), p. 829-837.

[50] K KROENKE, R L SPITZER et J B WILLIAMS. « The PHQ-9 : validity of a brief depression severity measure ». en. In : *J. Gen. Intern. Med.* 16.9 (2001), p. 606-613.

[51] Brendan CHOI. « Learn Python Basics on Windows ». In : *Introduction to Python Network Automation*. Berkeley, CA : Apress, 2021, p. 23-112.

[52] Siddharth SHARMA, Simone SHARMA et Anidhya ATHAIYA. « ACTIVATION FUNCTIONS IN NEURAL NETWORKS ». In : *International Journal of Engineering Applied Sciences and Technology* 04.12 (2020), p. 310-316.

[53] R Á GÓMEZ BRUBALLA. *Exploiting the interplay between visual and textual data for scene interpretation*. Universitat Autònoma de Barcelona, 2020.