
RÉPUBLIQUE DU BÉNIN
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

**UNIVERSITÉ D'ABOMEY-CALAVI**

**INSTITUT DE FORMATION ET DE RECHERCHE EN INFORMATIQUE**

BP 526 Cotonou Tel : +229 21 14 19 88
http://www.ifri-uac.net Courriel : contact@ifri.uac.bj


# MÉMOIRE

pour l'obtention du

**Diplôme de Licence en Informatique**

**Option :** Génie Logiciel

**Présenté par** :
Iffanice Bridiné **Houndayi**

# Système de prédiction des performances académiques des étudiants : cas de IFRI-UAC

**Sous la supervision de** :
Dr Ing. Vinasétan Ratheil **Houndji**
Ing. Pierre Jérôme **Zohou**

Année Académique : **2017 - 2018**

# Dédicace

A

Tous ceux qui œuvrent activement pour le développement des sciences informatiques en Afrique et particulièrement au Bénin.

# Remerciements

Nous tenons particulièrement à remercier :

- Notre famille, pour tout le soutien et l'encouragement dont elle a fait preuve à notre égard durant ces trois années de formation à l'IFRI ;
- Monsieur Ratheil HOUNDJI, notre maître de mémoire pour sa disponibilité, son accompagnement et son soutien dans la réalisation de ce mémoire ;
- Monsieur Jérôme ZOHOU, pour sa précieuse aide en ce qui concerne l'acquisition des données utilisées dans cette étude ;
- Monsieur Anani AMEDUITE, pour ses conseils qui nous ont guidé dans le choix de l'IFRI après notre baccalauréat ;
- L'administration de l'IFRI ;
- Tous nos enseignants, camarades et amis de l'IFRI pour avoir créé un environnement propice à un partage de connaissances tout au long de notre formation.

Que tous ceux qui de près ou de loin ont contribué à la réalisation de notre projet de fin d'étude trouvent à travers ce travail le signe de notre profonde reconnaissance.

*Une cause très petite, qui nous échappe, détermine un effet considérable que nous ne pouvons pas ne pas voir, et alors nous disons que cet effet est dû au hasard. Si nous connaissions exactement les lois de la nature et la situation de l'univers à l'instant initial, nous pourrions prédire exactement la situation de ce même univers à un instant ultérieur. Mais, lors même que les lois naturelles n'auraient plus de secret pour nous, nous ne pourrons connaître la situation initiale qu'approximativement.*

**—Henri Poincaré,** *Science et méthode*

# Table des matières

# Liste des figures



# Liste des tableaux

# Liste des Algorithmes

# Sigles et Abréviations

**AJAX :** Asynchronous JavaScript and XML 21
**API :** Application Programming Interface 33

**CSV :** Comma Separated Values 15, 20

**DOM :** Document Object Model 21

**EC :** Elément Constitutif 22

**IFRI :** Institut de Formation et de Recherche en Informatique 2, 11, 21, 22, 33

**LMD :** Licence Master Doctorat 1

**MPC :** Moyenne Pondérée Cumulative 9

**RMSE :** Root Mean Square Error 25, 33

**SGBDR :** Système de Gestion de Bases de Données Relationnelles 20

**TIC :** Technologies de l'Information et de la Communication 1

**UE :** Unité d'Enseignement vi, vii, 2, 12, 14–17, 22, 24, 25, 30, 32, 33
**UML :** Unified Modeling Language 12

# Glossaire

**data leakage :**

Création d'informations supplémentaires inattendues dans les données d'apprentissage, aboutissant généralement à des résultats de modèle trop optimistes et peu réalistes. 24

**dataset :**

Jeu de données représenté par un ensemble de valeurs où chaque valeur est associée à une variable et à une observation 5, 22

**faux négatif :**

Prédiction incorrecte d'un modèle détectant une UE comme validée alors qu'elle n'est pas validée par l'étudiant 24, 25

**faux positif :**

Prédiction incorrecte d'un modèle détectant une UE comme non validée alors qu'elle est validée par l'étudiant 24, 32

**hypothèse nulle :**

Hypothèse soumise au test et considérée comme vraie tout au long du test 24

**matrice de confusion :**

Tableau servant à évaluer la qualité d'une classification et qui est obtenu en comparant les données issues de la prédiction avec les données de référence 25

**modèle prédictif :**

Modèle mathématique permettant d'effectuer une estimation des résultats des étudiants dans une UE vi, vii, 2, 22, 23, 25, 32, 33

**précision :**

Fraction de prédictions positives correctes 24, 32

**sensibilité :**

Fraction de toutes les instances positives identifiées correctement par un modèle comme positives 24, 32

**vrai positif :**

Prédiction correcte d'un modèle détectant une UE comme non validée par un étudiant 24

# Résumé

L'impact de l'éducation sur le progrès économique et social à moyen et long terme d'un pays n'est plus à démontrer. A l'université, l'insuffisance de support adéquat d'accompagnement et d'orientation personnalisée des étudiants augmente le taux d'échec de ces derniers. Ce travail vise à contribuer à la réduction du taux d'échec. Nous avons réalisé une application dont l'objectif est de prédire les performances académiques des étudiants dans le système LMD, en particulier ceux de l'Institut de Formation et de Recherche en Informatique de l'Université d'Abomey-Calavi (IFRI- UAC) pour mieux les guider. L'approche utilisée a permis d'établir pour chacune des UEs d'un semestre donné des estimations des résultats des étudiants. Pour y parvenir, des techniques d'apprentissage automatique ont été utilisées. En exploitant des données recueillies sur deux années à l'IFRI, plusieurs algorithmes offerts par la bibliothèque Scikit-learn du langage Python ont été testés afin d'effectuer les prédictions. Les résultats obtenus sur les données de test révèlent que par rapport à cinq des neuf UEs pour lesquelles l'étude a été réalisée on obtient une mesure $F_2$ d'au moins 75% pour la classification et une RMSE inférieure ou égale à 2,93 dans chacune des UEs en ce qui concerne la régression. La solution fournit donc relativement de bons résultats et il est prévu plusieurs améliorations afin d'en affiner la précision et d'en étendre le champ d'action.

**Mots clés** : apprentissage automatique, classification et régression, prédiction de performances académiques, UE.

# Abstract

The impact of education on the achievement of medium and long term socioeconomic progress in a country is significant. At the university, the lack of adequate support for coaching and personalized guidance of students increases the failure rate of the latter. This work aims to contribute to the reduction of the failure rate. We realized an application in order to predict the academic performances of students in the LMD system, especially those of the Computer Training and Research Institute of the University of Abomey-Calavi (IFRI-UAC) to better guide them. The approach used allows to establish, for each of the teaching units of a given semester, some estimates of the students results. To achieve this, machine learning techniques were used. Using data collected over two years at IFRI, several algorithms offered by the Python Scikit-learn library were tested to make predictions. The results obtained on the test data reveal that, compared to five of the nine teaching units for which the study was conducted, we obtain an $F_2$ score of at least 75% for the classification and an RMSE of less than or equal to 2.93 in all the teaching units with respect to regression. The solution therefore provides relatively good results and several improvements are planned in order to refine its precision and extend its scope.

**Keywords**: machine learning, classification and regression, students performances prediction, teaching unit.

# Introduction

L'utilisation grandissante des TIC (Technologies de l'Information et de la Communication) dans les différents domaines socio-économiques a contribué à la génération d'une grande quantité de données. L'analyse de ces données par des humains peut s'avérer être une tâche difficile. Ainsi, plusieurs disciplines dont l'apprentissage automatique s'y attellent afin d'extraire un savoir ou de faire ressortir des structures intéressantes à partir de ces données dans le but de résoudre des problèmes ou d'améliorer des solutions existantes. Quand l'on s'intéresse au domaine éducatif, de part toutes les interactions effectuées et données produites par les acteurs de ce domaine, une grande quantité d'informations contenant des tendances et des modèles cachés est également générée. La présente étude s'articulera autour de l'application de techniques en apprentissage automatique pour améliorer la prise de décision et donc rehausser le taux de succès dans ce domaine.

## Problématique

Pour garantir une bonne formation des étudiants, il est important que ces derniers reçoivent un support adéquat pour s'améliorer et réussir aisément leur cursus. Malheureusement, plusieurs conditions rendent difficile cet état de choses ; et notamment dans les classes à grands effectifs il est plus difficile d'effectuer un suivi des étudiants. Ce qui diminue leurs chances de réussite.
Notre étude s'intègre donc à une initiative de réduction du taux d'échec des étudiants à travers la prédiction des performances académiques de ceux-ci, ce qui permettrait d'anticiper sur leurs résultats afin d'orienter la prise de décision par rapport aux mesures qui s'imposent pour éviter au maximum leur échec.

## Contexte

L'éducation constitue un facteur important dans la réalisation du progrès économique et social à moyen et long terme d'un pays. Les étudiants en tant que futurs cadres se doivent donc d'avoir de bonnes performances académiques puisqu'elles représentent le principal moyen d'évaluer la qualité de leur travail et par extension de leur compétence. Le présent projet vise à prédire les performances académiques des étudiants dans le système LMD (Licence Master Doctorat) pour leur permettre d'être mieux guidés et de ce fait de diminuer le taux d'échec à l'université. Le prototype a été conçu

et réalisé pour l'IFRI (Institut de Formation et de Recherche en Informatique), une jeune école de l'Université d'Abomey-Calavi qui accueille depuis quelques années de nombreux étudiants.

## Objectifs

La solution vise à prédire les résultats des étudiants à partir de modèles prédictifs. En particulier, elle permettra :

- d'effectuer la tâche de prédiction sous deux aspects, la classification et la régression ;
- d'effectuer pour un semestre donné les prédictions des étudiants à travers des estimations des résultats dans chaque UE (Unité d'Enseignement) du semestre ;
- de sortir la liste des étudiants pouvant valider et celle des étudiants pouvant ne pas valider une UE d'un semestre ;
- de fournir des visualisations relativement aux prédictions de chaque UE ;
- d'enregistrer des rapports de prédictions pour des fins de consultations futures.

## Organisation du mémoire

Le présent travail est structuré en trois chapitres. Le chapitre 1 présente les généralités liées au concept d'apprentissage automatique ou machine learning et une étude de l'existant par rapport aux travaux traitant du sujet de la prédiction de performances académiques. Le chapitre 2 expose notre solution à travers sa modélisation, son fonctionnement et les choix techniques liés à sa réalisation. Les résultats obtenus par rapport à la réalisation de la solution et les insuffisances qui lui sont liées seront quant à eux présentés dans le chapitre 3.

Chapitre 1

# Revue de littérature

## Introduction

Pour mener à bien la présente étude, il est nécessaire d'effectuer un état de l'art afin de mieux la situer dans le contexte des solutions existantes s'intégrant dans le même sens. Ainsi, ce chapitre fait un résumé de l'existant en matière de prédiction de performances académiques. Il consistera à présenter le concept du machine learning, puis à établir une synthèse des différents travaux portant sur la prédiction de performances académiques.

## 1.1 Généralités sur le Machine Learning

Le Machine Learning (ou apprentissage automatique) désigne l'étude de programmes informatiques (ou algorithmes) qui peuvent apprendre par l'exemple et généraliser à partir d'exemples existants d'une tâche [1]. Tom M. Mitchell fournit un formalisme court et largement utilisé à ce sujet qui stipule ce qui suit : *On dit qu'un programme informatique apprend de l'expérience $E$ par rapport à une classe de tâches $T$ et une mesure de performance $P$, si sa performance à réaliser des tâches dans $T$, mesurée par $P$, s'améliore avec l'expérience $E$* [1]. Le domaine de l'apprentissage automatique vise à répondre à la question : *Comment pouvons-nous construire des systèmes informatiques qui s'améliorent automatiquement avec l'expérience, et quelles sont les lois fondamentales qui régissent tous les processus d'apprentissage ?* [2]. Le but de l'apprentissage automatique est donc d'enseigner aux machines à effectuer des tâches en leur fournissant quelques exemples [3]. C'est ainsi qu'à partir de ces exemples, des algorithmes spécialisés arrivent à capturer des modèles afin d'établir des prédictions sur des données préalablement inconnues et non utilisées pendant la phase d'apprentissage. Le terme **prédiction** fait donc référence à la sortie d'un algorithme après qu'il ait été entraîné sur un ensemble de données historiques et appliqué à de nouvelles données lorsqu'on essaye de prévoir la probabilité d'un résultat particulier [17].

Il existe plusieurs types d'apprentissage automatique qui peuvent être catégorisés selon les méthodes utilisées et les applications souhaitées. Les types les plus utilisés dans les applications quotidiennes et

[1] Kevyn Collins-Thompson (Associate Professor of Information & Computer Science University of Michigan) dans son introduction au cours **Applied Machine Learning in Python** sur Coursera.

considérés comme principaux sont l'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage par renforcement (figure 1.1).

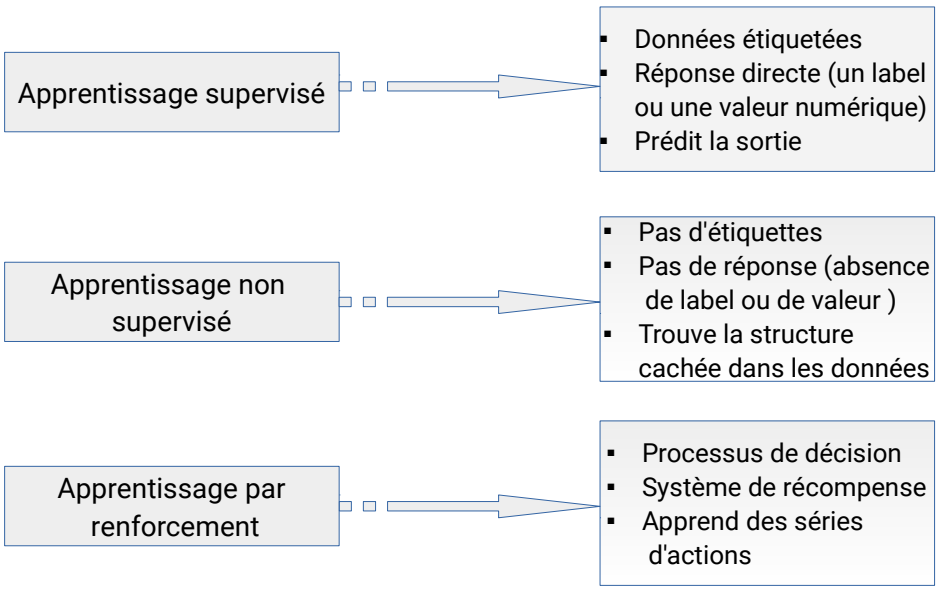


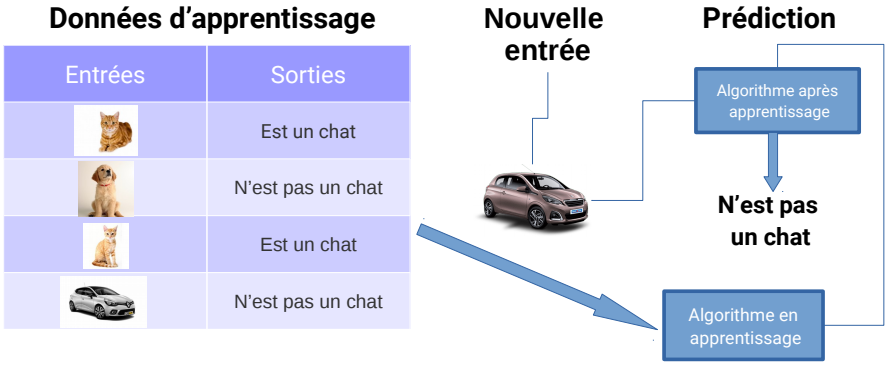

FIGURE 1.1 – Les trois types d'apprentissage automatique [5]

### 1.1.1 Apprentissage supervisé

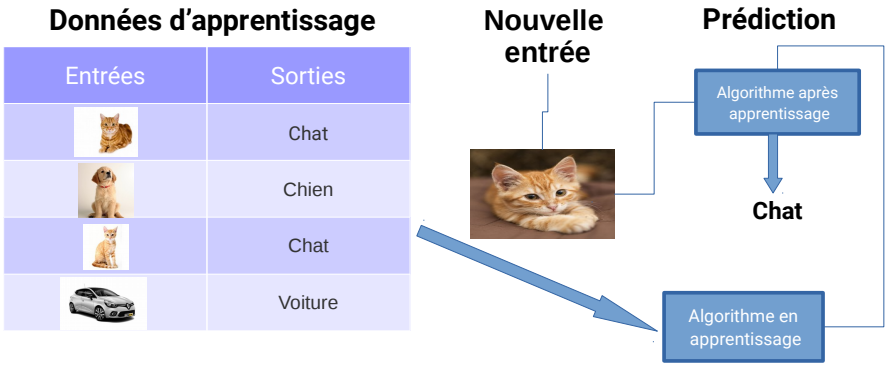
Les algorithmes d'apprentissage automatique fournissant les meilleurs résultats sont ceux qui automatisent les processus décisionnels en généralisant à partir d'exemples connus. Ici, l'utilisateur fournit l'algorithme avec des paires d'entrées et de sorties souhaitées, et l'algorithme trouve un moyen de produire la sortie souhaitée, pour une entrée donnée. En particulier, l'algorithme est capable de créer une sortie pour une entrée qu'il n'a jamais vue auparavant, et ceci sans aucune aide humaine [4]. Il existe deux types majeurs en ce qui concerne les problèmes d'apprentissage supervisé : la classification et la régression. En **classification**, le but est de prédire le label relié à une classe qui est un choix parmi une liste de possibilités prédéfinies (figure 1.2). La classification est souvent séparée en classification **binaire** qui est le cas particulier de la distinction entre exactement deux classes et en classification en **classes multiples** qui est la classification entre plus de deux classes [4]. Considérons l'exemple suivant où il sera question d'identifier l'élément présent sur une image. La classification binaire pourrait répondre à ceci à travers une prédiction qui consisterait à dire si oui ou non il s'agit d'un chat sur l'image (figure 1.2a) tandis que la classification en classes multiples répondra au problème en identifiant la classe effective à laquelle appartient l'élément sur l'image (figure 1.2b). Il est à noter que ceci ne peut-être fait qu'après la phase d'apprentissage (basée sur le jeu de données "entrées-sorties").

Pour les tâches de **régression**, l'objectif est de prédire un nombre continu ou un nombre à virgule flottante en terme informatique (ou un nombre réel en terme mathématique) [4]. Un exemple en régression serait de prédire le revenu annuel d'une personne à partir de ses études, de son âge et de son lieu de résidence [4] ou prédire la moyenne d'un étudiant dans une UE à partir de ses résultats passés, son âge, la série et l'ancienneté de son baccalauréat, etc [2].

[2] Ceci couvre une partie des objectifs de la présente étude.




(a) Classification binaire




(b) Classification en classes multiples

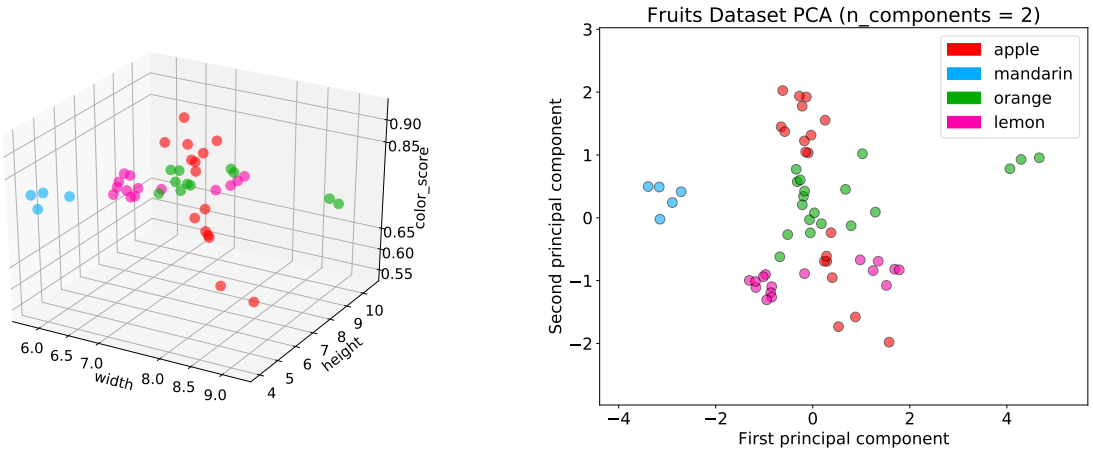
FIGURE 1.2 – Classification

En ce qui concerne les algorithmes les plus utilisés dans l'apprentissage supervisé , il s'agit de : K-nearest Neighbors (k-NN), Naive Bayes, Decision Trees, Random Forest, Linear Regression, Logistic Regression, Support Vector Machines (SVM), Neural Networks. Les tâches d'apprentissage automatique supervisé couvrent par exemple : l'identification d'un code postal à partir de chiffres manuscrits sur une enveloppe, la détermination de la gravité d'une tumeur sur la base d'une image médicale, la détection d'une activité frauduleuse dans les transactions par carte de crédit, la reconnaissance d'images, etc [24] .

### 1.1.2 Apprentissage non supervisé

En apprentissage non supervisé, seules les données d'entrée sont connues et aucune donnée de sortie connue n'est passée à l'algorithme. Ce type d'apprentissage est le plus souvent utilisé pour les transformations sur le dataset et le clustering. En ce qui concerne les **transformations sur le dataset**, il s'agit d'algorithmes qui créent une nouvelle représentation des données qui pourrait être plus facile à comprendre pour les humains ou pour d'autres algorithmes d'apprentissage automatique par rapport à la représentation originale des données. Une application courante des transformations non supervisées est la réduction de dimensionnalité, qui prend une représentation de grande dimension des données et trouve une nouvelle façon de représenter ces données en résumant les caractéristiques

essentielles. Un cas courant pour la réduction de dimensionnalité est la réduction à deux dimensions à des fins de visualisation [4] (figure 1.3) . Une autre application pour les transformations non su-




(a) Représentation en 3D

(b) Représentation en 2D après apprentissage non supervisé

FIGURE 1.3 – Réduction de dimensionnalité

pervisées consiste à trouver les pièces ou composants qui constituent les données. Un exemple de ceci est l'extraction de sujet sur des collections de documents texte. Ici, la tâche consiste à trouver les sujets inconnus dont il est question dans chaque document et à savoir quels sujets apparaissent dans chaque document. Cela peut être utile pour suivre la discussion de thèmes tels que les élections, le contrôle des armes à feu sur les réseaux sociaux [4].

En ce qui concerne le **clustering**, les algorithmes partitionnent les données en groupes distincts d'éléments similaires [4] (figure 1.4). Ici, l'algorithme ne définira pas le libellé réel (comme en apprentis-

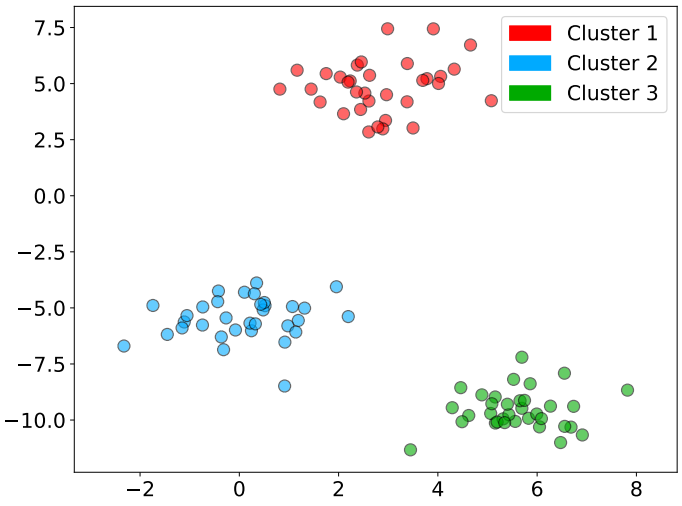



FIGURE 1.4 – Clustering

sage supervisé) ou le contexte pour les groupes formés, mais il aidera à faire ressortir les groupes eux-mêmes [19].

La liste suivante renseigne sur certains algorithmes utilisés dans l'apprentissage automatique non

supervisé : k-means clustering, APriori algorithm, Principal Component Analysis (PCA). L'apprentissage automatique non supervisé est utilisé notamment pour : l'identification des sujets dans un ensemble d'articles de blog, la segmentation des clients d'une entreprise en groupes avec des préférences similaires, la détection de modèles d'accès anormaux sur un site Web, etc [24].

### 1.1.3 Apprentissage par renforcement

L'objectif de l'apprentissage par renforcement est de développer un système (agent) qui améliore ses performances en fonction des interactions avec l'environnement (figure 1.5). Étant donné que les informations sur l'état actuel de l'environnement incluent généralement un signal de récompense (rétroaction), l'apprentissage par renforcement peut être considéré comme un domaine lié à l'apprentissage supervisé. Cependant, dans l'apprentissage par renforcement, cette rétroaction n'est pas la valeur finale prédite, mais une mesure de la façon dont l'action a été évaluée par une fonction de récompense. Grâce à son interaction avec l'environnement, un agent peut alors utiliser l'apprentissage par renforcement pour apprendre une série d'actions qui maximise cette récompense via une approche expérimentale par essais et erreurs [5].

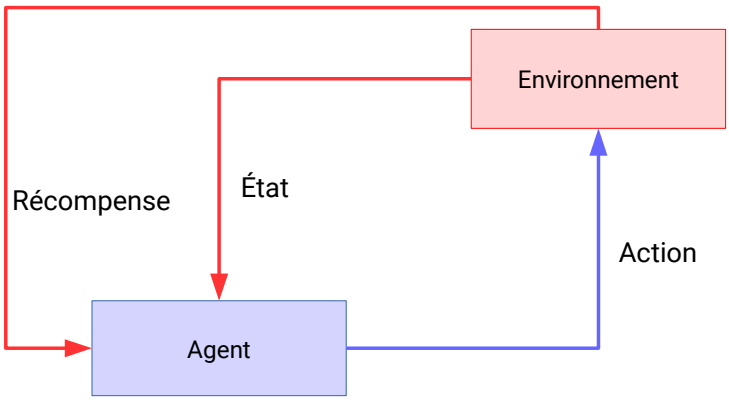



FIGURE 1.5 – Apprentissage par renforcement [5]

Des exemples d'algorithmes utilisés dans l'apprentissage par renforcement :

- Q-learning
- Temporal Difference (TD)

Ce type d'apprentissage est utilisé dans le domaine : des jeux , de la robotique, des contrôleurs industriels (programmateurs d'ascenseurs par exemple) [3], etc [6].

### 1.1.4 Workflow en machine learning

L'apprentissage automatique consiste à la combinaison de seulement trois éléments que sont : **la représentation, l'évaluation** et **l'optimisation** [7, 20].

[3] Un bon programmateur dépensera le minimum de puissance et servira le maximum de personnes. Pour les problèmes du genre les agents d'apprentissage par renforcement peuvent apprendre dans un environnement de simulation [6].

#### 1.1.4.1 La représentation

La première étape pour résoudre un problème avec l'apprentissage automatique consiste à déterminer comment représenter le problème d'apprentissage de manière à ce que l'ordinateur puisse le comprendre. Il est important de rassembler les données ou même de formuler la description liée à celles-ci et également choisir en fonction de leur représentation quel type d'algorithme appliquer à ces données.

#### 1.1.4.2 L'évaluation

Une fonction d'évaluation (également appelée fonction objective) est nécessaire pour distinguer pour une tâche donnée quels sont les algorithmes les mieux adaptés parmi ceux utilisés pour ladite tâche.

#### 1.1.4.3 L'optimisation

Après avoir choisi la manière de représenter les entrées, le type d'algorithme et la méthode d'évaluation, il est important de rechercher le modèle qui donne le meilleur résultat d'évaluation pour ce problème. Le choix de la technique d'optimisation est la clé de l'efficacité de l'algorithme.

Le processus de traitement d'une tâche d'apprentissage automatique est souvent un cycle [20] (figure 1.6). Cela implique un processus itératif, où une première estimation de ce que sont les principales caractéristiques (au niveau des données d'entrée du problème) et l'algorithme qui pourrait être approprié est faite. Le système est par la suite entraîné en utilisant les données d'apprentissage. Une évaluation est effectuée, afin de voir comment fonctionne le modèle et ensuite, sur la base des résultats, voir quels échantillons ont été classifiés correctement ou non. De manière générale, il est possible :

- de passer par ce cycle plusieurs fois pour affiner continuellement les caractéristiques au niveau des données d'entrée et évaluer leur effet sur la précision ;
- d'essayer différents types d'algorithmes, en fonction de la méthode d'évaluation choisie afin de déterminer si le problème est abordé avec une bonne démarche.

Il est très important de suivre cette démarche afin de garantir des résultats optimaux.

## 1.2 Étude de l'existant

L'apprentissage automatique est aujourd'hui fortement utilisé et son utilisation est étendue dans de nombreux domaines notamment celui de l'éducation, où l'obtention d'un fort taux de réussite est un enjeu capital. Ainsi, plusieurs recherches ont été effectuées dans le sens de la prédiction des performances académiques. Il ressort par ailleurs de ces travaux que l'utilisation de l'apprentissage automatique dans le domaine de la prédiction de performances académiques aboutit à de bons résultats.

Une revue sur la prédiction des performances des étudiants en utilisant des techniques de Data Mining a été réalisée au *School of Computer Sciences* à l' *Universiti Sains Malaysia* [8]. Elle s'articule autour des deux questions suivantes : *quels sont les attributs importants utilisés pour prédire les performances des étudiants* et *quelles sont les méthodes utilisées pour la prédiction des performances de ces derniers*. Selon les

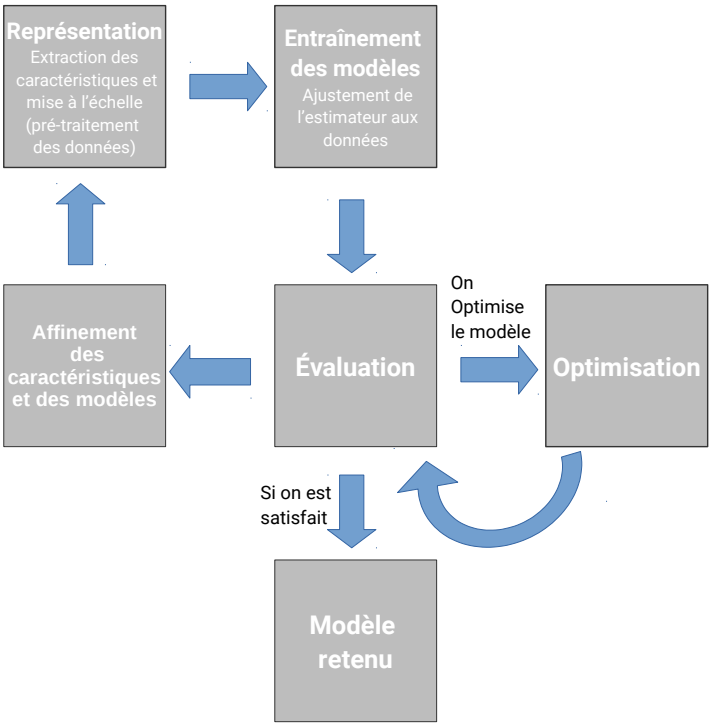



FIGURE 1.6 – Workflow en apprentissage automatique

auteurs, la prédiction des performances des élèves repose sur deux facteurs principaux : les attributs et les méthodes de prédiction. Les attributs fréquemment utilisés par les chercheurs sont : la MPC (Moyenne Pondérée Cumulative), qui est la variable d'entrée la plus importante, l'évaluation interne (travaux de laboratoire, interrogations de classe, présence), la démographie des étudiants (sexe, âge, antécédents familiaux et handicap), les évaluations externes (note obtenue à l'examen final pour une matière particulière), les activités parascolaires, les études secondaires, l'interaction sociale et le facteur psychométrique (rarement utilisé car il se base sur des données qualitatives et il est difficile d'obtenir des données valides). A partir de ces attributs, différentes méthodes d'apprentissage automatique ont été utilisées pour la prédiction des performances des étudiants. Il s'agit des arbres de décision (*Decision Trees*), des réseaux de neurones (*Neural Networks*), des classifieurs bayésiens naïfs (*Naive Bayes*), de la méthode des k plus proches voisins (*K-Nearest Neighbor*) et des machines à vecteurs de support (*Support Vector Machine*). Suite aux différents tests effectués, le modèle ayant obtenu la meilleure précision de prédiction (**98%**) est celui basé sur l'utilisation des réseaux de neurones, s'en suivent respectivement ceux basés sur les arbres de décision (**91%**), les machines à vecteurs de support et la méthode des k plus proches voisins (**83%**), puis enfin les classifieurs bayésiens naïfs (**76%**).

Une étude effectuée en Avril 2017 par **Ali Daud et al.** traite de la prédiction des performances des étudiants (en termes de prédiction d'abandon) en utilisant des techniques avancées d'apprentissage [11]. Le problème de la prédiction des performances des étudiants y est défini comme suit :
Étant donnés pour l'apprentissage $n$ échantillons $(X_1, z_1), (X_2, z_2) \dots (X_n, z_n)$, où $X_i$ est le vecteur des attributs pour l'étudiant $a_i$ et $A$ l'ensemble des $n$ étudiants où $A = \{a_1, a_2, a_3 \dots a_n\}$. $X_i \in R^m$ avec $m$ le nombre total d'attributs et $z_i$ le statut de la performance de l'étudiant (études terminées ou abandonnées) où $z_i \in \{-1, +1\}$. Pour prédire la performance d'un étudiant, la fonction de prédiction

suivante a été proposée :

$$z = F(A/X) \tag{1.1}$$

où

$$F(A/X) = \begin{bmatrix} \geq 0 \text{ si z = +1, terminé} \\ < 0 \text{ si z = -1, abandonné} \end{bmatrix} \tag{1.2}$$

Le but est d'apprendre une fonction prédictive $\widehat{F}\ (.)$ ou de prédire si un étudiant abandonnera ses études :

$$\widehat{z} = \widehat{F}(A/X) \tag{1.3}$$

Ce travail de recherche présente les méthodes de prédiction utilisées, lesquelles utilisent quatre types différents d'attributs à savoir : les dépenses familiales, le revenu familial, les informations personnelles des étudiants et les biens familiaux. Il adapte également le processus de sélection des sous-ensembles d'attributs afin d'identifier les attributs les plus efficaces pour la prédiction. Deux types de modèles de classification (**discriminants** et **génératifs** [13]) sont utilisés pour l'apprentissage de la fonction prédictive souhaitée $\widehat{F}\ (.)$. Les modèles discriminants utilisés sont les machines à vecteur de support (*SVM*), les arbres de décision (arbre de classification et de régression) et le **C4.5** [32] ; ceux génératifs utilisés sont les **réseaux bayésiens** (*Bayes Network*) [31] et les classifieurs bayésiens naïfs (*Naive Bayes*). La performance des expériences est évaluée par la **mesure $\mathbf{F}_1$** (*F1-score*) [30]. Les machines à vecteur de support obtiennent les meilleurs résultats pour les ensembles d'attributs proposés pour l'étude avec une mesure F1 de **86%**.

Une autre étude effectuée à l'*Université de Tampere* en Juin 2017 par **Murat Pojon** aborde la thématique de l'utilisation du machine learning pour prédire les performances des étudiants, dont l'objectif spécifique dans ce cas est la comparaison des méthodes d'apprentissage automatique et des techniques d'**extraction / ingénierie des caractéristiques** [22] (**feature engineering**[4]) en termes d'amélioration des résultats de prédiction [9]. Il est question ici d'un problème de classification où l'issue sera de dire si oui ou non un étudiant réussit. Trois différentes méthodes ont été utilisées pour ce faire. Il s'agit de la régression linéaire (*Linear Regression*), des arbres de décision (*Decision Trees*) et des classifieurs bayésiens naïfs (*Naive Bayes*). Les différentes méthodes utilisées ont été appliquées sur deux jeux de données. La meilleure méthode par rapport au premier jeu de données est celle des classifieurs bayésiens naïfs (avec une **justesse** (*Accuracy*) [30] de **95,8%** et de **97,5%** respectivement sur les données brutes et sur les données après extraction des caractéristiques). Par rapport au deuxième jeu de données, la meilleure méthode avant extraction des caractéristiques est la régression linéaire avec une justesse de **74,5%** tandis que la meilleure méthode après extraction des caractéristiques est celle des arbres de décision avec une justesse de **77,6%** (**77,5%** pour la régression linéaire). Les résultats de cette étude indiquent que l'extraction / ingénierie des caractéristiques apporte plus d'amélioration aux résultats de la prédiction que la sélection des méthodes (choix d'un algorithme à partir de valeurs indicatives telles que la justesse). Ceci va dans le même sens qu'**Andrew Ng** lorsqu'il affirme que *l'apprentissage automatique appliqué est fondamentalement l'ingénierie des caractéristiques* [10]. Mais bien que l'ingénierie des caractéristiques ait été plus efficace que la sélection des méthodes, la combinaison des deux approches donne de meilleurs résultats.

Des travaux similaires ont été effectués par d'autres chercheurs notamment à l'*Université de Minho*

[4] Le feature engineering renvoie à l'ensemble des interventions effectuées sur les données brutes avant qu'elles ne soient prises en compte par un algorithme d'apprentissage automatique[21].

en 2008 par **Paulo Cortez et Alice Silva** [12] où le sujet de prédiction académique a été appliqué aux élèves du secondaire. Les jeux de données contenaient les résultats scolaires antérieurs, les données démographiques, sociales et autres données scolaires, l'objectif étant de prédire le rendement des élèves en mathématiques et en portugais, et si possible, d'identifier les variables clés qui influent sur la réussite ou l'échec scolaire. Le travail consistait donc en trois tâches à savoir la classification binaire (échec/réussite), la classification à 5 niveaux (la note prédite est prise entre A ([16-20]), B ([14-15]), C ([12-13]), D ([10-11]) et F ([0-9]) ) et la régression (sortie numérique entre 0 et 20) ; les arbres de décision, les forêts d'arbres décisionnels ou forêts aléatoires (*Random Forest*), les réseaux de neurones et les machines à vecteurs de support ont été utilisés et comparés en termes de justesse. Les résultats obtenus révèlent que la performance de l'élève est fortement affectée par ses résultats précédents. Une analyse des connaissances fournies par les meilleurs modèles prédictifs a montré que, dans certains cas, il existe d'autres caractéristiques pertinentes, telles que : les autres données scolaires (nombre d'absences, soutien scolaire supplémentaire), la démographie (par exemple l'âge de l'élève, l'emploi et l'éducation des parents) et les variables sociales (par exemple sortir avec des amis, consommer de l'alcool).

Par ailleurs, une implémentation d'un modèle [23] basée sur l'article de Paulo Cortez et Alice Silva [12] a été réalisée. L'objectif était de permettre de prédire si un élève échouerait ou non au cours de mathématiques suivi en se focalisant sur les taux d'échec. Le modèle est basé sur les machines à vecteur de support linéaire. Ce modèle a obtenu les meilleurs résultats par rapport aux autres modèles, tels que les classifieurs bayésiens naïfs, la régression logistique (*Logistic Regression*) et les forêts d'arbres décisionnels.

Une importante remarque à soulever à partir de tous ces travaux est qu'il n'existe pas un algorithme qui soit adapté à n'importe quel type de données c'est-à-dire que la méthode utilisée est fortement conditionnée par la structure et le contenu des données. Et puisque nous ne disposons pas de toutes les données employées dans les précédents travaux, il est important que la présente étude soit faite afin de proposer une solution adaptée et contextualisée aux données disponibles à l'IFRI. Ladite solution sera en mesure de proposer une interface utilisateur à travers une application Web afin de garantir une utilisation plus facile et offrir des visualisations pour une meilleure interactivité.

## Conclusion

Il existe plusieurs types d'apprentissage automatique et différentes méthodes (algorithmes) qui leur sont associées. Dans le cas général de l'analyse prédictive et sur la base des précédents travaux il n'existe cependant pas un algorithme générique capable d'obtenir de bons résultats indépendamment de la structure des données utilisées. Pour le présent projet, il s'agira donc d'appliquer différents algorithmes aux données disponibles et en déduire le mieux adapté afin de créer un modèle capable de bien généraliser sur de nouvelles données puis enfin créer une interface Web pour l'utilisation de celui-ci. Le chapitre suivant fera l'objet d'une présentation de notre solution à travers sa conception et les outils utilisés pour sa réalisation.

Chapitre 2

# Solution et choix techniques

## Introduction

La réalisation d'une application nécessite un regroupement et une hiérarchisation des idées liées à sa conception et à son développement. Le présent chapitre présente la solution proposée ainsi que les différentes phases de sa modélisation. Il présente également un aperçu des différents outils et technologies utilisés pour la réalisation de la solution, que nous avons nommée **Pythia**.

## 2.1 Description de la solution

Pythia vient répondre à la problématique de prédiction de performances académiques, à travers une plateforme Web qui permet de réaliser pour un semestre donné une estimation des performances des étudiants en effectuant des prédictions relatives aux résultats des étudiants dans chaque UE dudit semestre. Ces prédictions sont de deux ordres, parlant ainsi de classification et de régression. En ce qui concerne la classification, le système sera en mesure de prédire si un étudiant valide ou non une UE ; la régression quant à elle permettra d'anticiper sur les résultats des étudiants en terme de moyennes dans chaque UE. Dans le but de garantir une interactivité lors des analyses réalisées, des visualisations sont affichées pour orienter l'utilisateur.

## 2.2 Modélisation

Afin de modéliser les fonctionnalités de **Pythia**, le choix a été porté sur **UML (Unified Modeling Language)**, qui est aujourd'hui un outil de communication incontournable, utilisé sur des centaines de projets de par le monde [14]. UML est un langage de modélisation graphique conçu pour fournir un ensemble de représentations standardisées afin de réaliser la conception architecturale et fonctionnelle d'un logiciel. Il est également indépendant du langage de programmation utilisé pour le développement dudit logiciel. Ainsi, les diagrammes de cas d'utilisation, de classes et de séquence exposeront la modélisation de la présente application.

### 2.2.1 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation regroupe tous les cas d'utilisation possibles d'un logiciel. Il est la représentation complète des fonctionnalités du système modélisé. La figure 2.1 illustre le diagramme de cas d'utilisation de **Pythia**. Il en ressort que pour notre système il existe deux catégories d'acteurs :

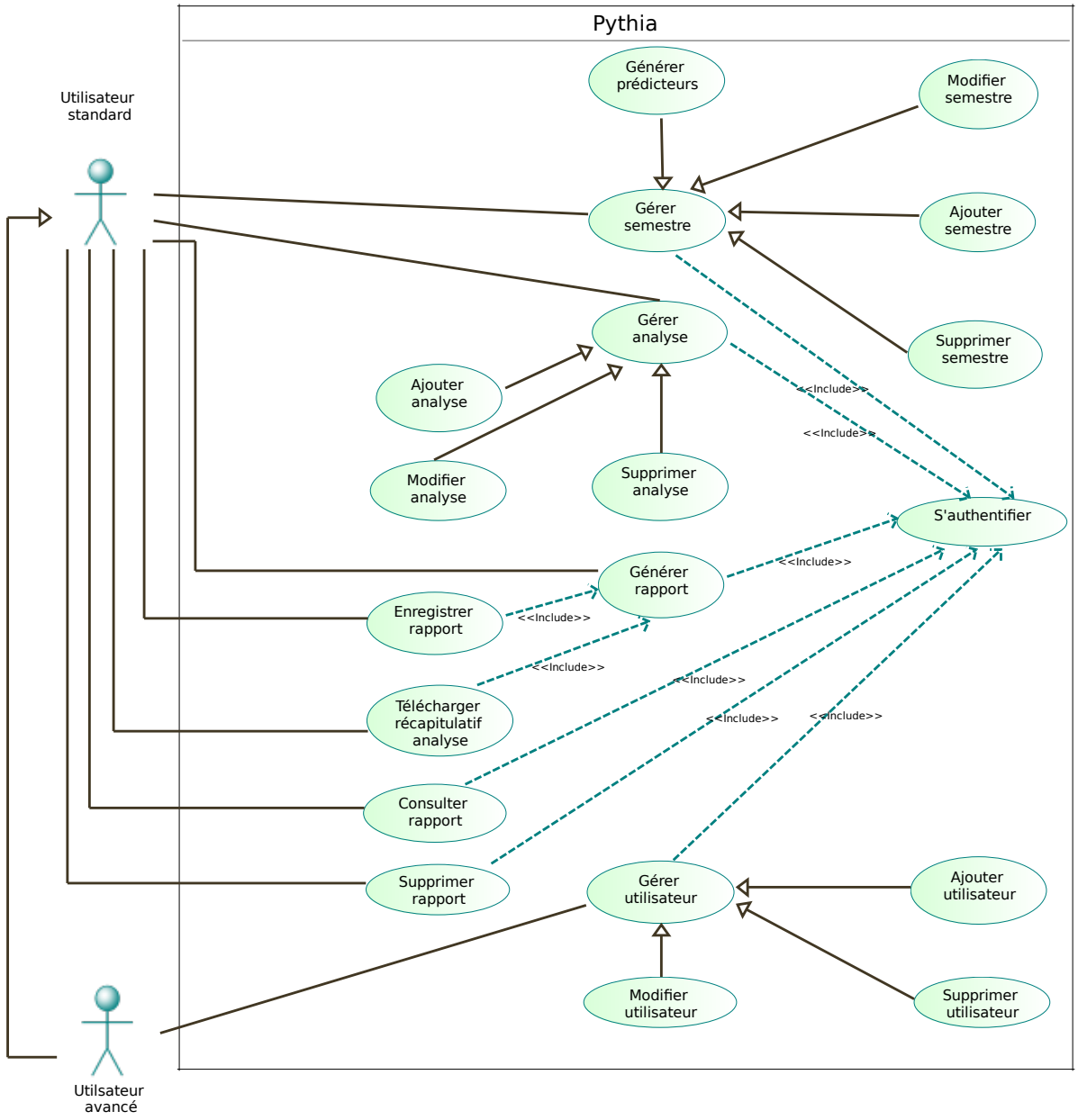



FIGURE 2.1 – Diagramme de cas d'utilisation de Pythia

les utilisateurs standards et les utilisateurs avancés ; chacun d'eux avec leurs champs d'actions respectifs. Pour n'importe quel acteur du système, il est obligatoire de s'authentifier avant toute action. Cette authentification se fait par saisie d'un nom d'utilisateur et d'un mot de passe, à la suite de laquelle il pourra avoir accès aux fonctionnalités offertes par l'application. L'utilisateur avancé est une spécialisation de l'utilisateur standard, ce qui signifie qu'il peut effectuer toutes les tâches effectuées par l'utilisateur standard. A celles-ci s'ajoutent, la gestion des utilisateurs standards à travers l'ajout,

la modification et la suppression de ceux-ci. L'utilisateur standard quant à lui pourra donc, conformément aux objectifs de la présente étude effectuer la prédiction des issues des UEs d'un semestre. Dans ce sens, il peut ajouter, modifier ou supprimer des semestres, générer des prédicteurs pour un semestre, ajouter, modifier ou supprimer des analyses relativement à ces semestres, générer, enregistrer ou supprimer des rapports basés sur une analyse [1] et télécharger un récapitulatif de l'analyse. Pour des consultations futures, en lieu et place de reprendre la génération d'un rapport il pourra tout simplement l'enregistrer sur la plateforme et pourra le consulter lorsqu'il le souhaite.

### 2.2.2 Diagramme de classes

Afin de spécifier les classes intervenant dans le système et quels liens elles entretiennent entre elles, nous utiliserons un diagramme de classes. La figure 2.2 présente le diagramme de classes de Pythia. Il contient les classes **Utilisateur**, **Utilisateur avancé**, **Rapport**, **Semestre** et **Analyse**. Une attention particulière sera portée sur les classes **Semestre** et **Analyse**.

- La classe **Semestre** permet d'enregistrer les données relatives à un semestre. Elle est liée à la classe **Utilisateur**, ce qui entraîne qu'un objet de type Semestre a une propriété **auteur** de type **Utilisateur**. Lorsqu'un semestre est ajouté avec les fichiers d'apprentissage (échantillon de résultats de semestre(s) antérieur(s) et échantillon de résultats du semestre), il est possible pour un utilisateur de générer les prédicteurs de ce semestre, lesquels seront utilisés pour générer le rapport d'une analyse liée au dit semestre.
- La classe **Analyse**, également liée à la classe **Utilisateur**, permet de configurer les informations relatives à l'analyse que désire effectuer l'utilisateur en précisant notamment le type de l'analyse, le fichier de l'analyse et le semestre auquel cette analyse est liée. Il est à préciser que seuls les semestres ayant déjà leurs prédicteurs générés pourront être utilisés comme semestre de base d'une analyse.

### 2.2.3 Diagramme de séquence

Pour décrire la chronologie d'un cas d'utilisation et les objets intervenant dans sa réalisation il est commun d'utiliser un diagramme de séquence. Les diagrammes de séquence permettent donc de donner une première idée de la manière dont les scénarios des cas d'utilisation se déroulent. Il s'agira d'exposer le diagramme de séquence du cas d'utilisation *Générer rapport* (voir figure 2.3).
Lorsque l'utilisateur veut générer le rapport d'une analyse, le système vérifie si le fichier d'analyse de ladite analyse correspond au point de vue structurel au fichier d'échantillon de résultats de semestre(s) antérieur(s) au semestre de base de l'analyse [2]. Si c'est le cas, le fichier d'analyse est traité à partir des prédicteurs du semestre de base de l'analyse et les résultats sont envoyés à l'utilisateur. Dans le cas contraire, ce dernier reçoit un message d'erreur lui demandant de passer à la reconfiguration de l'analyse en changeant le fichier d'analyse.

[1] Il est à noter que la modification et la suppression d'un semestre (d'une analyse ou d'un rapport) n'est possible que par l'utilisateur qui l'a ajouté

[2] Ceci est important puisque la tâche de prédiction se fera en se basant sur les mêmes attributs que ceux qui seront utilisés lors de la phase d'apprentissage.

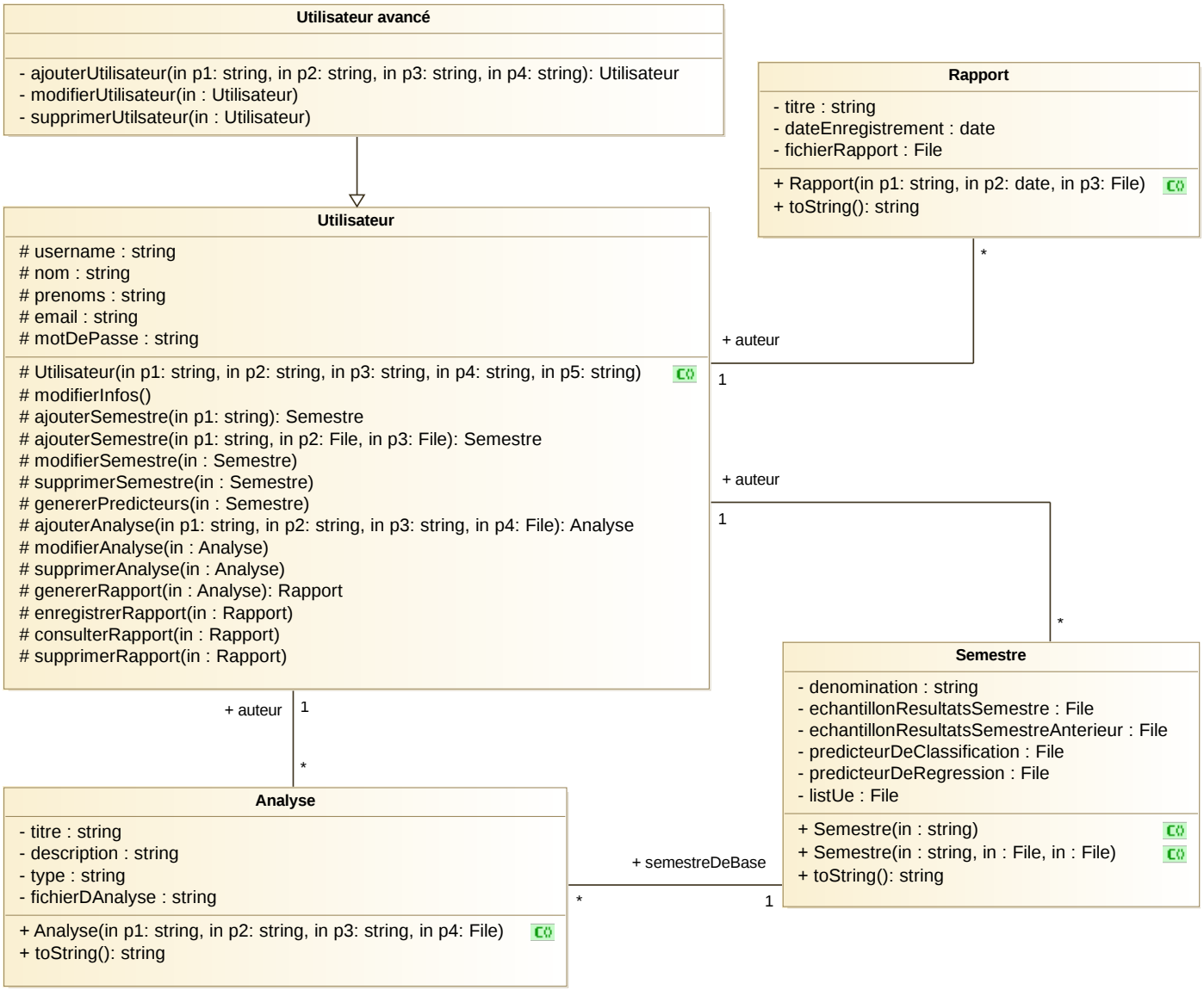



FIGURE 2.2 – Diagramme de classes de Pythia

## 2.3 Fonctionnement du système

### 2.3.1 Principe général de fonctionnement de Pythia

Le but de **Pythia** est de permettre à l'utilisateur d'effectuer des analyses afin d'avoir une estimation de la performance semestrielle (par UE) de certains étudiants (voir figure 2.4 ). Comme spécifié au niveau des cas d'utilisation, l'utilisateur, pour effectuer n'importe qu'elle action doit s'authentifier. Après authentification, l'ajout d'un semestre se fait en précisant deux fichiers pour l'apprentissage, tous deux de type **CSV (Comma Separated Values)**. Il s'agit dans un premier temps d'un fichier contenant des échantillons de résultats d'étudiants pour un semestre antérieur et dans un deuxième

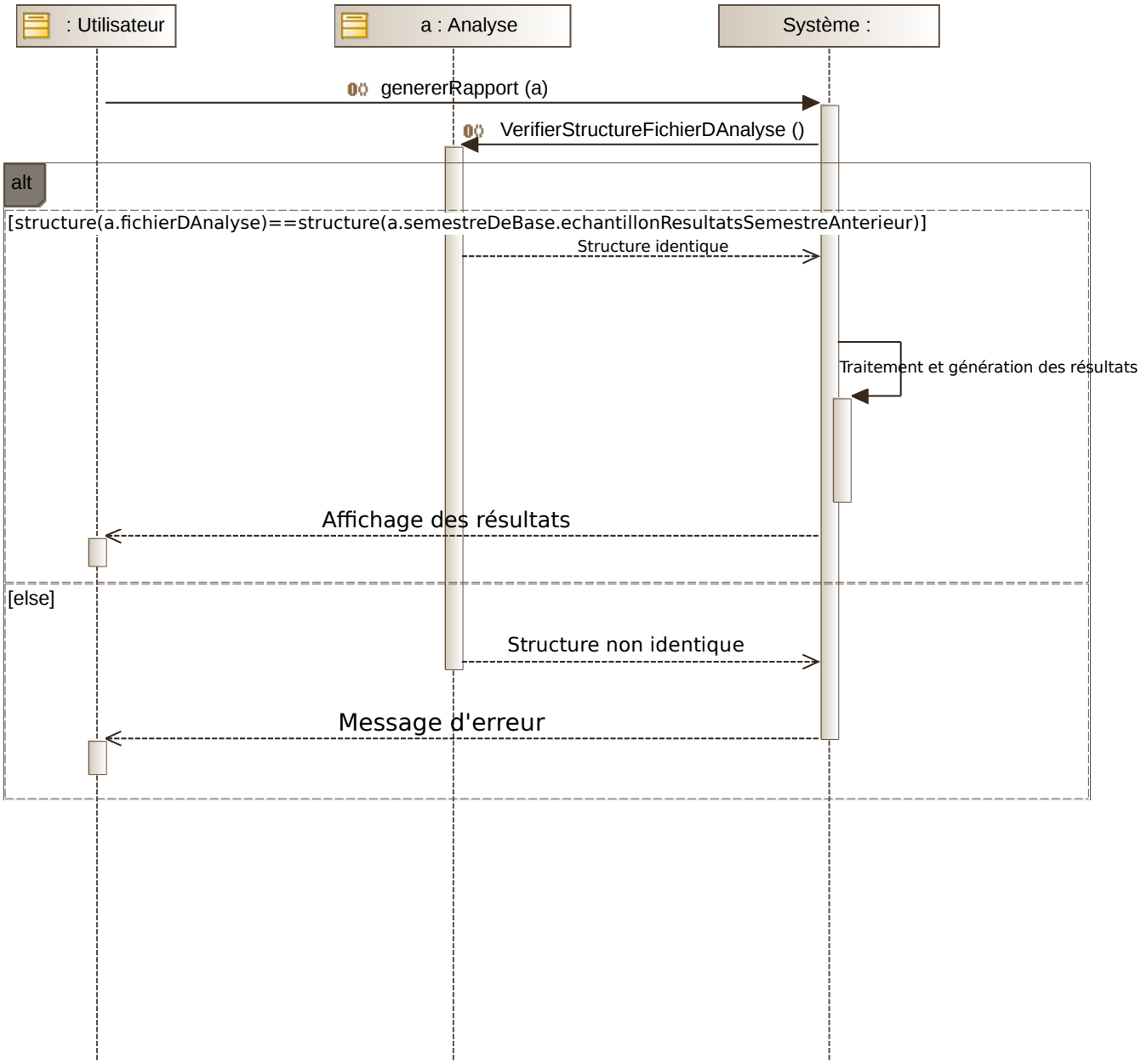



FIGURE 2.3 – Diagramme de séquence du cas d'utilisation *Générer rapport*

temps un fichier contenant les échantillons de résultats d'étudiants pour le semestre courant [3]. Ceci permet de faire un mappage entre les données du premier fichier (les attributs utilisés pour l'apprentissage) et celles du second (les issues) afin d'établir pour chaque instance des variables d'entrée et des variables de sortie (ici connues). L'idéal serait d'avoir pour ces deux fichiers des informations sur les mêmes étudiants puisque pour la phase d'apprentissage il faut des instances dont les variables d'entrées sont associées à une (des) variable(s) de sortie [4]. En ce qui concerne la génération des prédicteurs d'un semestre, la phase d'apprentissage est réalisée et c'est à cette étape que les UEs pour lesquelles les prédictions seront faites sont déduites. Les meilleurs modèles sont donc retenus. Pour

[3] Il s'agit du semestre dont l'ajout sera fait.

[4] Dans le cas contraire les étudiants dont les informations manquent dans un fichier ou l'autre sont tout simplement ignorés pour l'apprentissage.

réduire le temps lié aux rapports d'analyses[5] les modèles sont sérialisés c'est-à-dire qu'ils sont enregistrés sur le disque dur de la machine sous des fichiers d'extension **.pythia** et simplement chargés (désérialisés) pour faire les prédictions. En cas de modification des fichiers d'un semestre il faudra repasser à la phase de génération des prédicteurs. Une fois les rapports générés il est possible de télécharger un récapitulatif des résultats sous forme d'un fichier PDF.

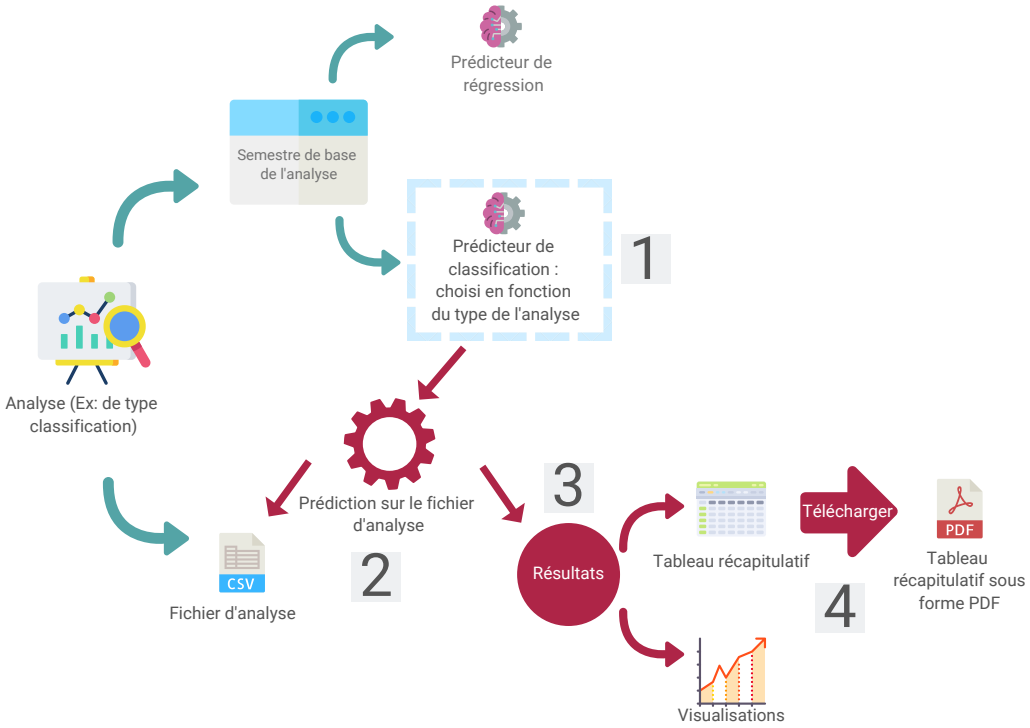



FIGURE 2.4 – Principe de fonctionnement de Pythia

### 2.3.2 Pseudo-algorithme décrivant la phase de prédiction de performances

Pour la phase de prédiction, en fonction de l'analyse voulant être effectuée par l'utilisateur, un fichier d'analyse est spécifié. Et lorsque la structure de ce fichier correspond à celle du fichier d'échantillon de résultats d'un semestre antérieur à celui pour lequel la prédiction est faite, un prétraitement est réalisé afin de dégager clairement les variables d'entrées qui seront utilisées pour la prédiction. Ainsi en fonction du type de l'analyse, les prédicteurs du semestre auquel est lié l'analyse réaliseront pour chacun des étudiants présents dans le fichier d'analyse et pour chaque UE du semestre une prédiction des performances. Les issues prédites pour chaque UE sont **validée/non validée** et une **note entre 0 et 20** respectivement pour une analyse de type classification (binaire) et une analyse de type régression. Ce processus est résumé par l'algorithme 1.

[5] Pour éviter de refaire l'entraînement des modèles.

**Algorithme 1 :** Pseudo-algorithme décrivant la phase de prédiction de performances

**Entrées :** Une instance $a$ de la classe Analyse
**Sorties :** L'ensemble des prédictions des performances de tous les étudiants présents dans le fichier d'analyse de $a$

```
1  début
2      si structure(a. fichierDAnalyse) =
         structure(a.semestreDeBase.echantillonResultatsSemestreAnterieur) alors
3          etudiants ← pretraitement(a.fichierDAnalyse);
4          si type de a = "Classification" alors
5              predicteurs ← a.semestreDeBase.predicteurDeClassification;
6          sinon
7              predicteurs ← a.semestreDeBase.predicteurDeRegression;
8          fin
9          predictions ← [ ];
10         liste_ue ← a.semestreDeBase.listUe;
11         pour i ← 0 à taille(etudiants) − 1 faire
12             prediction_etudiant ← [ ];
13             pour j ← 0 à taille(list_ue) − 1 faire
14                 ŷ ← predicteurs[j].predire(etudiants[i]);
15                 prediction_etudiant[j] ← ŷ;
16             fin
17             predictions[i] ← prediction_etudiant;
18         fin
19         retourner predictions;
20     sinon
21         retourner ("Erreur! Reconfigurer l'analyse");
22     fin
23 fin
```

## 2.4 Choix techniques

### 2.4.1 Méthodologie de développement

La conduite du projet s'est faite en se conformant à la **méthodologie agile**, une approche de développement itératif qui permet d'observer concrètement l'évolution du projet. La notion même de "gestion de projet" est remise en question au profit de "gestion de produit", de façon à raisonner davantage "produit" que "projet" [38]. Le choix a été porté sur **Scrum** qui est une méthode agile où le développement est découpé en plusieurs phases courtes appelées itérations ou **sprint**. L'ensemble des fonctionnalités du sprint (habituellement 2-4 semaines) viennent du "backlog" du projet, qui est un ensemble priorisé d'exigences importantes à achever [39]. Chaque itération inclut des travaux de conception, de spécification fonctionnelle et technique quand c'est nécessaire, de développement et de test [38]. Dans notre cas, le développement a été découpé en itérations de deux semaines. L'ensemble des fonctionnalités prévues pour notre solution nous a permis d'établir des **user stories** qui ont été découpées en tâches, afin d'établir clairement le travail à réaliser. Cette méthode permet de mieux s'adapter aux imprévus pouvant subvenir pendant le projet et facilite la livraison d'un produit de qualité dans un temps planifié.

### 2.4.2 Outils et technologies utilisées

La réalisation de notre solution a nécessité l'utilisation de plusieurs technologies. Elles seront regroupées en deux groupes : le Back-end et le Front-end.

#### 2.4.2.1 Back-end

Le Back-end représente tout ce que l'utilisateur ne voit pas relativement à l'application. Il s'agit des traitements logiques et actions effectués pour pouvoir conserver, fournir, ou modifier les données au sein de l'application. Les technologies que nous avons utilisées à cet effet sont :

- **Python**
  Python est un langage de programmation interprété, orienté objet et de haut niveau avec une sémantique dynamique. Ses structures de données intégrées de haut niveau, combinées à un typage dynamique et à une liaison dynamique, le rendent très attrayant pour le développement rapide d'applications, ainsi que pour son utilisation en tant que langage de script. Python prend en charge les modules et les packages, ce qui encourage la modularité du programme et la réutilisation du code [42]. Il est notamment utilisé dans les domaines de la sécurité informatique, de l'intelligence artificielle, etc. Il nous a donc servi de base pour le développement de Pythia à travers l'écriture de règles et de routines pour assurer le traitement des tâches de prédictions.

- **Scikit-learn**
  Scikit-learn est la bibliothèque Python la plus utilisée pour l'apprentissage automatique [43]. C'est un projet open-source qui est en développement et en amélioration continus et englobe un large éventail d'importants algorithmes d'apprentissage automatique. Elle dispose par ailleurs d'une excellente documentation en ligne et constitue de ce fait un outil dont la prise en main est facile. Notre solution a donc fait usage de Scikit-learn pour l'utilisation d'algorithmes qui ont servi à la création des prédicteurs d'un semestre donné.

- **NumPy**
  NumPy est l'un des packages fondamentaux pour le calcul scientifique en Python. Il contient des fonctionnalités pour les tableaux multidimensionnels, des fonctions mathématiques de haut niveau telles que les opérations d'algèbre linéaire et la transformée de Fourier, ainsi que des générateurs de nombres pseudo-aléatoires. Dans scikit-learn, le tableau NumPy est la structure de données fondamentale. Scikit-learn prend des données sous la forme de tableaux NumPy. Toutes les données qui seront utilisées devront être converties en tableau NumPy et tous les éléments du tableau doivent être du même type [4].

- **Pandas**
  Pandas est une bibliothèque Python pour la gestion et l'analyse des données. Elle est construite autour d'une structure de données appelée DataFrame, qui est modélisée d'après la structure DataFrame du langage R. Un DataFrame est un tableau, similaire à un tableur Excel. Pandas fournit une grande variété de méthodes pour modifier et utiliser ce tableau ; en particulier, il autorise les requêtes et les jointures de tables de type SQL. Contrairement à NumPy, qui exige que toutes les entrées d'un tableau soient du même type, Pandas permet à chaque colonne d'avoir un type distinct (par exemple, entiers, dates, nombres à virgule flottante et chaînes). Elle permet de lire une grande variété de formats de fichiers et de bases de données comme

SQL, les fichiers Excel et les fichiers CSV [4]. Elle a donc été utilisée dans notre application pour la gestion des données chargées depuis les fichiers CSV.

- **Django**
  Django est un framework Web Python de haut niveau qui encourage le développement rapide et une conception propre et pragmatique [44]. Construit par des développeurs expérimentés, il prend en charge la majeure partie des problèmes liés au développement Web tels que la scalabilité, la rapidité, la sécurité, etc. Ainsi il est possible de se concentrer sur l'écriture de l'application sans avoir à réinventer la roue. C'est un outil gratuit et open-source. Ces raisons nous ont poussé à le choisir pour l'écriture du serveur Web de Pythia.

- **PostgreSQL**
  PostgreSQL est un SGBDR (Système de Gestion de Bases de Données Relationnelles) open-source, robuste et puissant, aux fonctionnalités riches et avancées, capable de manipuler en toute fiabilité de gros volumes de données, mêmes dans des situations critiques [45]. Dans le cadre de notre solution, il a été préféré à **MySql** un autre SGBDR simple d'utilisation à cause de sa robustesse.

#### 2.4.2.2 Front-end

Le Front-end, par opposition au Back-end fait référence aux éléments que l'on voit à l'écran, avec lesquels on peut interagir ; il désigne la partie visuelle de l'application. Il s'agit d'éléments comme les images, les fenêtres, les boutons, les menus déroulants, etc. Les technologies que nous avons utilisées à cette fin sont :

- **HTML 5 & CSS 3**
  HTML ou HyperText Markup Language est un langage de description et de balisage des pages Web. Il permet de structurer et de mettre en forme le contenu des pages Web avec des balises de formatage. Les balises permettent d'indiquer la façon dont la page doit être présentée et les liens qu'elle partage avec d'autres pages [49].
  CSS ou Cascading Style Sheets (en français feuilles de style en cascade) définit des règles permettant de modifier l'aspect des pages Web dont le contenu a été décrit en HTML (les polices, les couleurs, les espacements, etc). Le principe des feuilles de style consiste à regrouper dans un même document des caractéristiques de mise en forme associées à des groupes d'éléments [48].
  Ces deux technologies ont donc été utilisées pour décrire et mettre en forme le contenu des pages Web de Pythia.

- **JavaScript**
  JavaScript est un langage de programmation qui permet d'implémenter des mécanismes complexes sur une page web. Il peut intervenir lorsque qu'une page web fait plus que simplement afficher du contenu statique, par exemple afficher du contenu mis à jour à des temps déterminés, des cartes interactives, des animations 2D/3D, des menus vidéo défilants, etc. C'est la troisième couche des technologies standards du web, les deux premières étant HTML et CSS [47].

- **Bootstrap**
  Bootstrap [46] est un framework front-end gratuit pour un développement Web plus rapide et plus facile. Bootstrap comprend des modèles de conception basés sur HTML et CSS pour la typographie, les formulaires, les boutons, les tableaux, la navigation, les carrousels d'images et

bien d'autres, ainsi que des plugins JavaScript en option. Il permet de faire des interfaces qui s'adaptent automatiquement à tous les appareils, des petits téléphones aux grands ordinateurs de bureau.
Pythia utilise donc cette technologie afin de fournir à l'utilisateur de belles interfaces s'adaptant convenablement à n'importe quel type d'écrans.

- **jQuery**
  jQuery est une bibliothèque JavaScript qui est une collection de routines JavaScript, qui prend en compte de nombreuses tâches et les encapsule dans de simples méthodes faciles de manipulation [41]. La bibliothèque jQuery englobe des fonctionnalités telles que la manipulation des éléments du DOM (Document Object Model), la manipulation CSS, les méthodes d'événements, les effets et animations, AJAX (Asynchronous JavaScript and XML), etc. Elle a été utilisée dans notre solution pour gérer l'interactivité des pages et pour les requêtes AJAX qui permettaient de communiquer avec le serveur afin de modifier une page sans avoir à la recharger et sans la bloquer.

- **Chart.js**
  Chart.js est une bibliothèque JavaScript de représentation de données sous forme de graphes qui utilise l'élément **canvas HTML5**, ce qui la rend de ce fait compatible avec la quasi totalité des navigateurs supportant HTML5 [40]. C'est une bibliothèque légère ne nécessitant aucune dépendance, ce qui facilite ainsi son intégration. Les représentations graphiques générées par Chart.js sont de différents types, notamment les courbes, les diagrammes en bâtons, les camemberts, etc et s'adaptent à tout type d'écran. Chart.js a été utilisée par Pythia pour générer les visualisations liées aux rapports d'analyse afin de permettre aux utilisateurs d'avoir une représentation graphique des résultats des prédictions.

## Conclusion

Ce chapitre a présenté notre solution à travers sa modélisation, son fonctionnement et les choix techniques liés à sa réalisation. Le prochain chapitre fournira les résultats obtenus suite à l'application de notre solution aux données de l'IFRI.

Chapitre 3

# Résultats et discussion

## Introduction

Avant la concrétisation du développement de Pythia, plusieurs algorithmes ont été testés afin d'obtenir des modèles capables de répondre au mieux aux besoins de l'étude. La performance de ces algorithmes a donc été évaluée (voir tableaux 3.1 et 3.2) et ceci a conduit, pour chaque UE, au choix de deux algorithmes (respectivement pour la classification et la régression). Ce chapitre présente une description détaillée des résultats obtenus au terme des tests, un aperçu de l'application et enfin une discussion autour des insuffisances de l'étude.

## 3.1 Données et algorithmes utilisés

Nous avons utilisé les données de l'IFRI pour les tests de l'application. A l'IFRI le premier cycle consiste en trois (03) années académiques, chacune de deux semestres. Les enseignements qui y sont dispensés concernent plusieurs UEs subdivisées en ECs (Eléments Constitutifs) ( Ex :Logique mathématique, langage C, etc). La présente étude s'est faite en tenant compte des données disponibles, lesquelles concernaient celles de la première année de Licence SI/GL recueillies sur deux années (2016-2017 et 2017-2018). La tâche de prédiction a été donc effectuée sur le deuxième semestre de Licence 1 (ne disposant pas de données pertinentes pour le faire pour le premier semestre) par rapport auquel neuf (09) des dix (10) UEs ont été prises en compte (l'UE de Discipline étant celle qui a été isolée). Ainsi, pour base de l'apprentissage, des données sur les **notes dans les ECs du premier semestre** et des données à caractère social telles que l'**âge** et le **sexe** ont été utilisées. Enfin, après prétaitement et isolation des données impertinentes, les données ont été rassemblées dans un dataset (avec 258 instances) puis séparées en données d'apprentissage (180 instances) et en données de tests (78 instances)[1].
Avant d'ajuster les modèles certains pré-traitements étaient nécessaires. Ainsi, les attributs de type nominal, dans notre cas le sexe ("Masculin" ou "Féminin") ont été transformés en type binaire (0 ou 1). Des tests effectués en utilisant plusieurs algorithmes (voir figure 3.1) ont mené à la sélection des modèles idéaux pour chaque UE. En ce qui concerne les algorithmes testés, il s'agit :

[1] Il est à noter qu'à aucun moment de l'apprentissage les données de test ne sont intervenues.

- des machines à vecteurs de support (Support Vector Machines : SVM) [28],
- des arbres de décision (Decision Trees) [26],
- des forêts d'arbres décisionnels ou forêts aléatoires (Random Forest) [27],
- de la régression d'arête (Ridge regression, utilisé pour la régression) [15]
- de la régression logistique (Logistic Regression, utilisé pour la classification) [18],
- de l'algorithme AdaBoost [33],
- de l'algorithme Gradient Boosting Machine (GBM) [34],
- de la méthode des k plus proches voisins (KNN) [25], et
- des réseaux de neurones à propagation avant (Feed Forward Neural Network/Multi-layer Perceptron : MLP) [29].

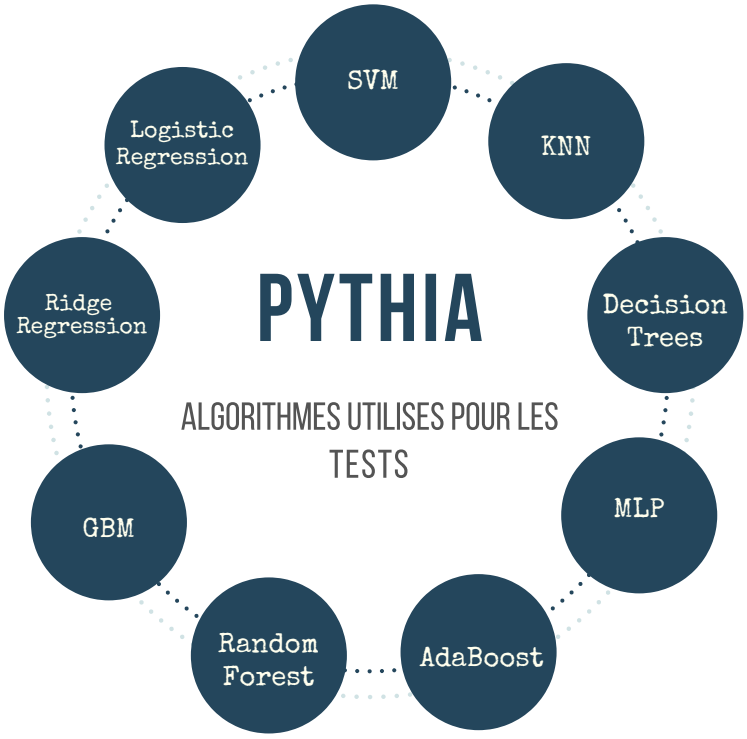



FIGURE 3.1 – Algorithmes utilisés pour les tests

## 3.2 Evaluation des algorithmes

Pour la sélection et l'évaluation des modèles, l'approche **"Entraînement-Validation-Test"** a été utilisée. La méthode de validation croisée **k-fold cross-validation** [35] et la recherche par grille des hyperparamètres de chaque algorithme ont été effectuées sur le jeu de données d'entraînement pour la sélection des modèles et des paramètres. Ceci consiste à construire et évaluer un modèle en fonction des paramètres spécifiés dans une grille en se basant sur une technique d'échantillonage où le jeu de données initial pour l'apprentissage est subdivisé en **k** échantillons ou **"folds"**. Un des k échantillons

sera utilisé pour la validation, les k-1 restants pour l'entraînement ; ce processus étant répété k fois en changeant à chaque fois l'échantillon utilisé pour la validation. A la fin, la moyenne de la performance de chaque modèle sur les différents échantillons de validation est calculée pour déterminer le meilleur modèle pour la phase de validation. Ainsi, pour l'évaluation effective du modèle, dans le but d'éviter des potentiels risques de **data leakage** [16], des données inconnues n'étant pas intervenues dans la phase d'apprentissage et de validation sont utilisées : il s'agit des données de test.
La performance des algorithmes a été évaluée en utilisant des métriques bien définies. En ce qui concerne la classification, l'hypothèse nulle est fixée au fait qu'un étudiant ne valide pas une UE. Dans le souci de détecter au maximum les cas d'étudiants qui pourraient ne pas valider une UE, on préférera dans ce contexte commettre une erreur de deuxième espèce, c'est-à-dire accepter l'hypothèse nulle alors qu'elle fausse. Dans ce sens, comme métrique d'évaluation le choix a été porté sur la **mesure $F_2$** [36] (un cas particulier de la **mesure $F_\beta$** où $\beta = 2$) qui est une moyenne pondérée de la sensibilité [30] et de la précision [30], donnant un plus grand poids à la sensibilité. Ce qui entraîne le fait que les **faux négatifs (FN)** soient considérés comme réduisant plus la performance que les **faux positifs (FP)** [2]. L'objectif ici sera donc de limiter la proportion de faux négatifs tout en augmentant celle de vrais positifs (VP).

Le calcul de la **mesure $F_2$** est donné par la formule :

$$F_2 = 5 \times \frac{précision \times sensibilité}{4 \times précision + sensibilité} = \frac{5 \cdot VP}{5 \cdot VP + 2 \cdot FN + FP}$$

où la précision est définie comme :

$$\frac{VP}{VP + FP}$$

et la sensibilité comme :

$$\frac{VP}{VP + FN}.$$

Les résultats des performances (**mesure $F_2$**) des différents algorithmes de classification après validation croisée et réglage d'hyperparamètres relativement à chaque UE sont consignés dans le tableau 3.1

TABLE 3.1 – Récapitulatif des performances des algorithmes de classification (résulats arrondis à $10^{-2}$ près - meilleurs scores par UE en gras - meilleurs scores par algorithme soulignés)

| Mesures $F_2$ | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| UEs / Algorithmes | UE1 | UE2 | UE3 | UE4 | UE5 | UE6 | UE7 | UE8 | UE9 |
| Support Vector Machines | 0,88 | **0,87** | 0,76 | 0,76 | 0,55 | 0,57 | **0,60** | 0,65 | 0,62 |
| Decision Tree | 0,85 | 0,70 | 0,65 | 0,68 | 0,28 | 0,40 | 0,57 | 0,68 | 0,47 |
| Random Forest | 0,81 | 0,83 | 0,74 | 0,72 | 0,51 | **0,65** | 0,53 | **0,75** | 0,54 |
| Logistic Regression | 0,80 | 0,80 | **0,77** | **0,79** | 0,53 | 0,59 | 0,57 | 0,68 | **0,65** |
| AdaBoost | 0,73 | 0,80 | 0,66 | 0,62 | **0,57** | 0,26 | 0,58 | 0,62 | 0,40 |
| Gradient Boosting | 0,91 | 0,83 | 0,54 | 0,13 | 0,53 | 0,34 | 0,38 | 0,1 | 0,41 |
| KNN | 0,83 | 0,79 | 0,44 | 0,56 | 0,47 | 0,38 | 0,31 | 0,40 | 0,40 |
| Feed forward neural network | **0,92** | 0,83 | 0,57 | 0,51 | 0,29 | 0,52 | 0,38 | 0,66 | 0,47 |

[2] Cela correspond en réalité à notre objectif car l'on souhaite éviter de commettre une erreur de première espèce, c'est-à-dire rejeter l'hypothèse nulle alors qu'elle est vraie.

La figure 3.2 présente les matrices de confusion des prédictions dans chaque UE en utilisant les meilleurs modèles retenus à partir du tableau 3.1. On remarque que les taux de faux négatifs au niveau de chaque UE sont faibles et que globalement les modèles arrivent à détecter une bonne proportion d'étudiants qui ne valident pas une UE. Ce qui satisfait l'objectif fixé au départ.

En ce qui concerne la régression, la métrique utilisée pour l'évaluation est la racine carrée de l'erreur quadratique moyenne, en anglais RMSE (Root Mean Square Error) [37]. C'est la racine carrée de la moyenne des différences au carré entre la prévision et l'observation réelle. Elle permet d'évaluer l'erreur globale d'une prédiction. Son calcul est donné par la formule :

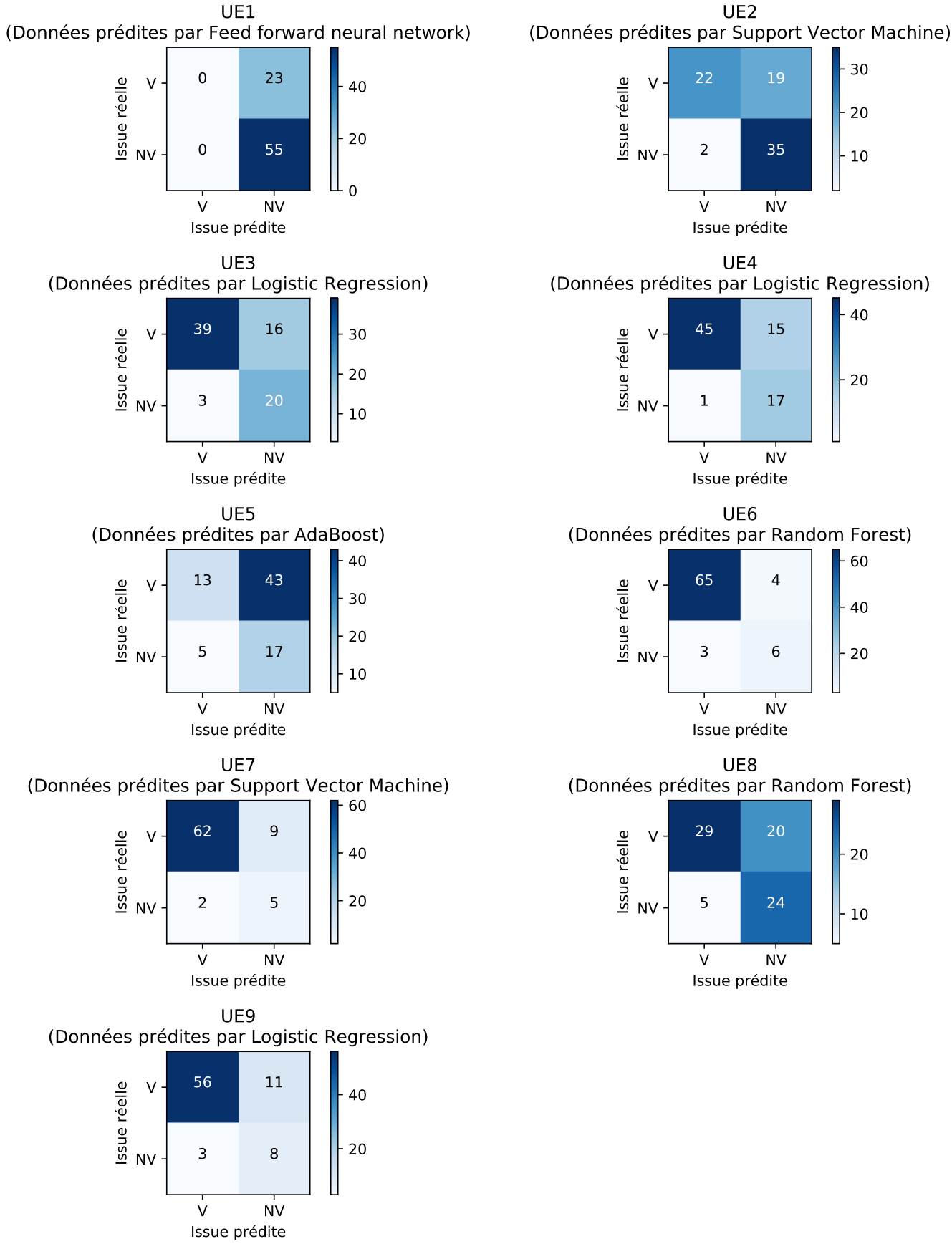
$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

où n représente la taille de l'échantillon sur lequel la prédiction est faite, $y_i$ la valeur réelle pour le i-ème élément dans l'échantillon et $\hat{y}_i$ la valeur prédite pour cet élément.
Les résultats des performances (**RMSE**) des différents algorithmes de régression après validation croisée et réglage d'hyperparamètres relativement à chaque UE sont consignés dans le tableau 3.2

TABLE 3.2 – Récapitulatif des performances des algorithmes de régression (résulats arrondis à $10^{-2}$ près - erreurs minimales par UE en **gras** - erreurs minimales par algorithme soulignées)

| RMSEs | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| UEs / Algorithmes | UE1 | UE2 | UE3 | UE4 | UE5 | UE6 | UE7 | UE8 | UE9 |
| Support Vector Machines | 1,97 | 2,62 | 2,70 | 2,31 | 2,63 | **2,89** | 2,56 | **2,93** | 2,91 |
| Decision Tree | 2,41 | 3,10 | 4,34 | 2,90 | 2,73 | 3,81 | 3,73 | 3,51 | 3,44 |
| Random Forest | **1,92** | **2,60** | 2,51 | 2,23 | 2,58 | 2,91 | **2,50** | 3,04 | 2,85 |
| Ridge Regression | 2,11 | 2,90 | 3,04 | **2,10** | **2,41** | 2,92 | 2,71 | 3,28 | 2,84 |
| AdaBoost | 2,01 | 2,75 | 2,65 | 2,31 | 2,62 | 3,17 | 3,19 | 3,06 | 2,82 |
| Gradient Boosting | 2,02 | 2,69 | **2,49** | 2,37 | 2,74 | 2,93 | 2,65 | 3,10 | **2,81** |
| KNN | 2,30 | 2,76 | 2,52 | 2,46 | 2,85 | 3,01 | 2,62 | 3,51 | 2,94 |
| Feed forward neural network | 2,09 | 3,17 | 3,19 | 2,18 | 2,63 | 3,46 | 3,38 | 3,49 | 2,86 |

La figure 3.3 expose une confrontation des notes réelles et des résultats obtenus après prédiction dans certaines UEs (UE1, UE4 et UE5) à partir des algorithmes retenus. On remarque que les modèles arrivent à capturer les tendances générales mais donnent quelques fois des résultats s'éloignant vraiment des valeurs réelles.




V= Validé / NV = Non validé

FIGURE 3.2 – Matrices de confusion


FIGURE 3.3 – Confrontation des notes réelles et des résultats obtenus après prédiction en régression

## 3.3 Cas pratique

Nous présenterons ici quelques fonctions de l'application Web Pythia. Il s'agit des fonctions : ajout d'un semestre, génération des prédicteurs d'un semestre, ajout d'une analyse et génération de rapport d'une analyse.

### 3.3.1 Ajout d'un semestre

L'utilisateur ajoute un semestre en spécifiant une dénomination et les fichiers d'apprentissage de ce dernier (voir figure 3.4). Après clic sur le bouton **Ajouter**, le semestre est créé (voir figure 3.5)

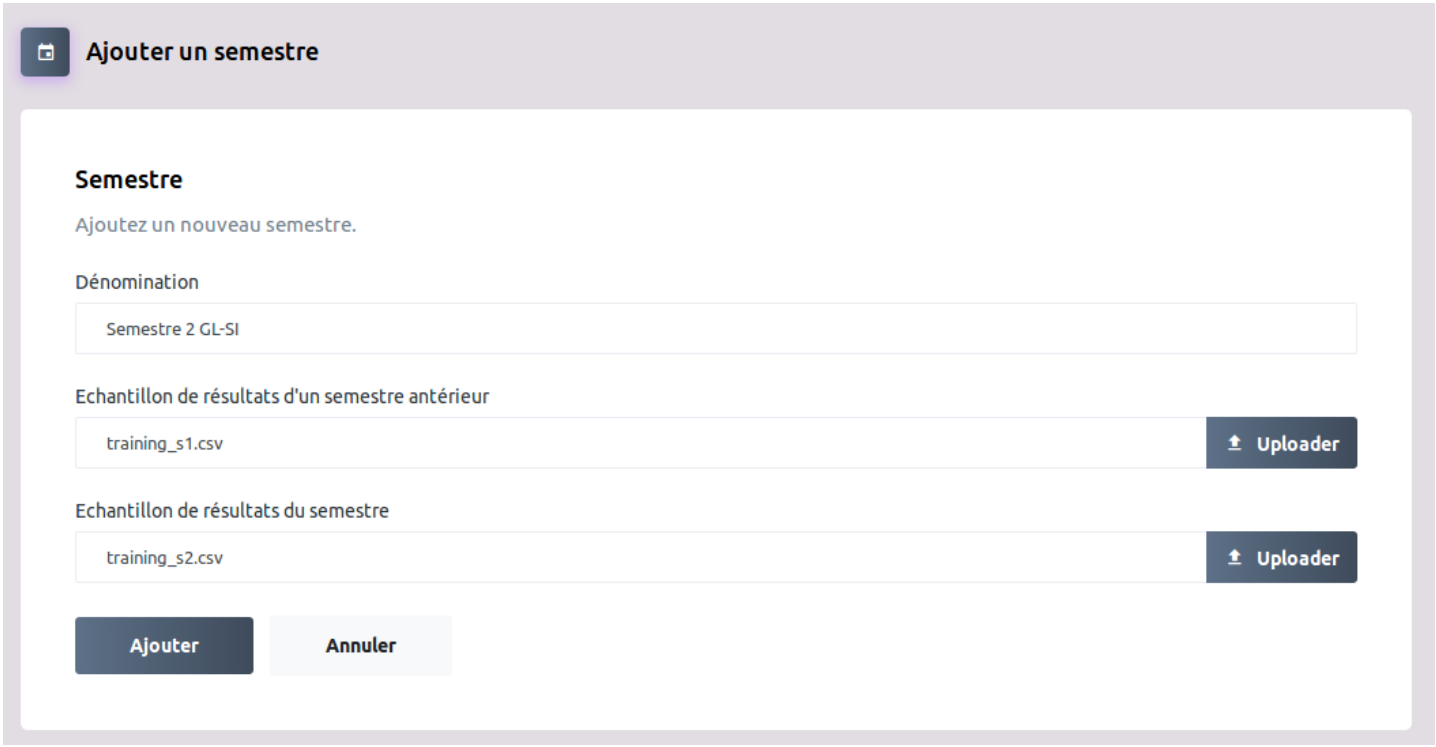



FIGURE 3.4 – Ajout d'un semestre

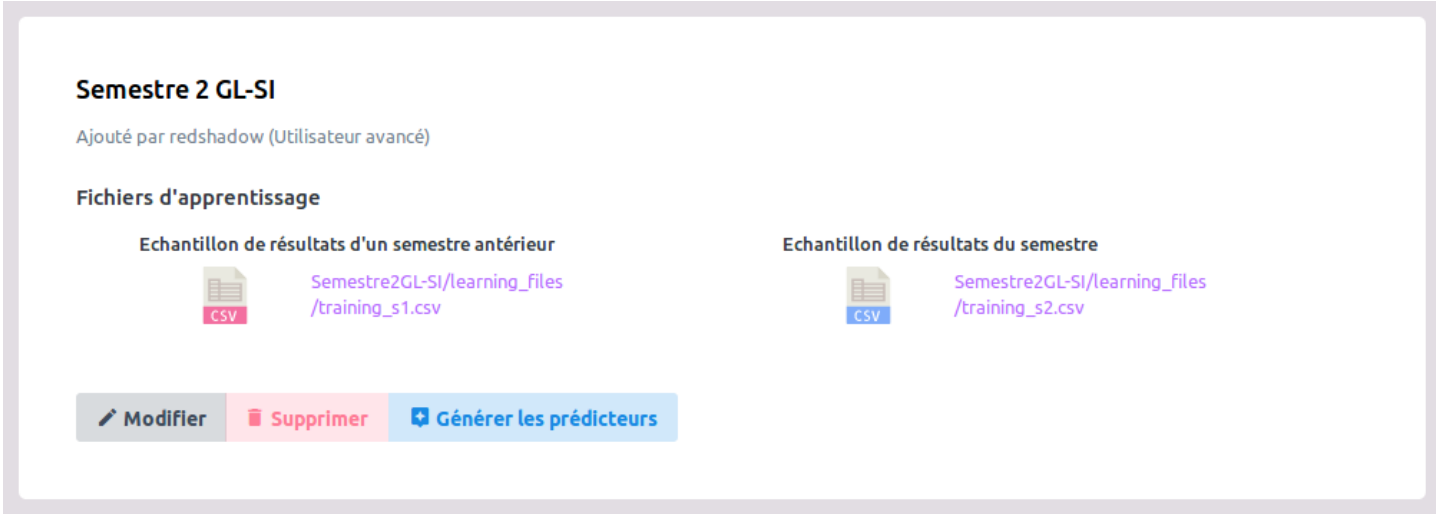



FIGURE 3.5 – Semestre ajouté

### 3.3.2 Génération des prédicteurs d'un semestre

Pour générer les prédicteurs du semestre, l'utilisateur clique sur le bouton **Générer les prédicteurs** (voir figure 3.6).


FIGURE 3.6 – Semestre après génération des prédicteurs

### 3.3.3 Ajout d'une analyse

L'utilisateur crée une analyse en précisant entre autres le type de l'analyse, le semestre de base et le fichier d'analyse (voir figure 3.7). Après clic sur le bouton **Ajouter**, l'analyse est ajoutée (voir figure 3.8). Nous présentons dans la suite une analyse de type "Classification".

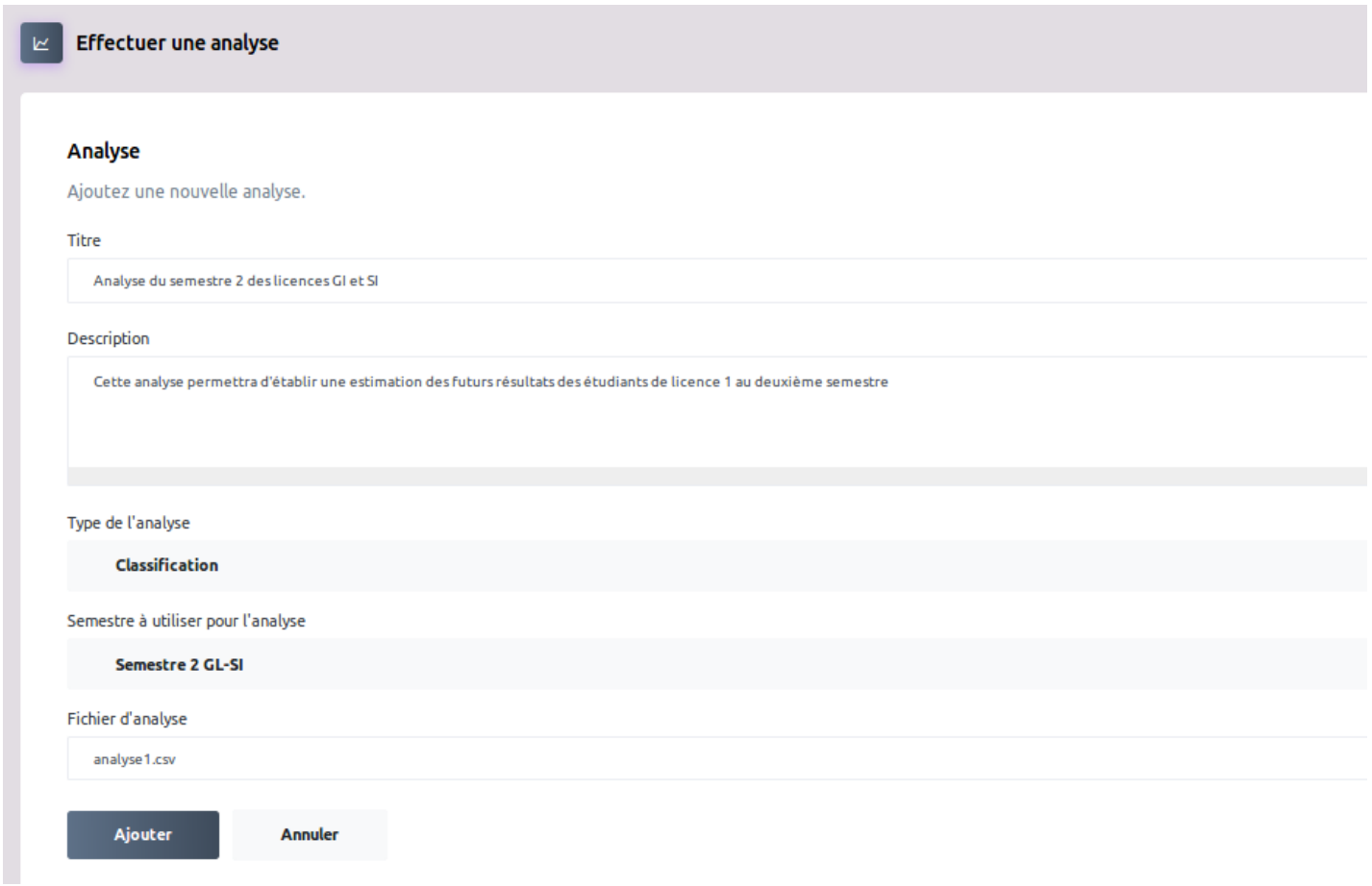


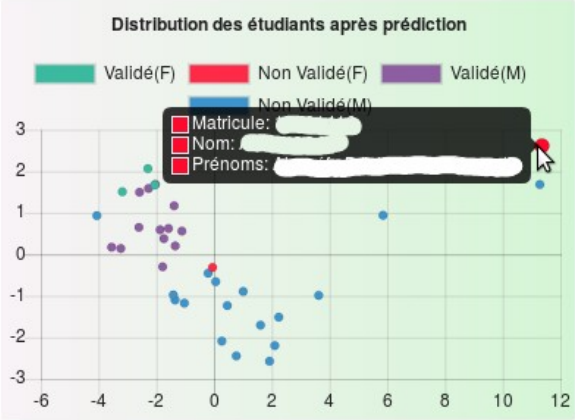

FIGURE 3.7 – Ajout d'une analyse

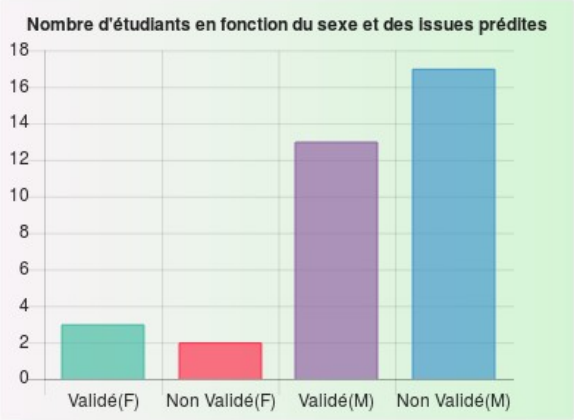

FIGURE 3.8 – Analyse ajoutée

### 3.3.4 Génération du rapport d'une analyse

L'utilisateur a la possibilité de générer le rapport d'une analyse en cliquant sur le bouton **Générer le rapport**. Ainsi il obtient des visualisations par rapport à chaque UE et un tableau récapitulatif de l'analyse. Les figures 3.9 et 3.10 présentent respectivement les visualisations des résultats d'une UE et une partie du tableau récapitulatif d'une analyse [3].

[3] Les informations sensibles telles que le matricule, le nom et le prénoms sont cachées.




**Pourcentages de réussite de l'UE après prédiction:**

- Etudiants de sexe féminin: 60.0% (3 sur 5)
- Etudiants de sexe masculin: 43.33% (13 sur 30)

FIGURE 3.9 – Rapport d'analyse : exemple des résultats d'une UE

| Numéro matricule | Nom | Prénoms | Sexe | Analyse Mathematique | Mathematiques Appliquees | Fondamentaux Des Bases De Donnees Relationnelles | Concepts Avances Des Reseaux Informatiques |
|---|---|---|---|---|---|---|---|
| [illegible] | [illegible] | [illegible] | Masculin | Non validé | Validé | Validé | Validé |
| [illegible] | [illegible] | [illegible] | Masculin | Non validé | Validé | Validé | Validé |
| [illegible] | [illegible] | [illegible] | Feminin | Non validé | Non validé | Non validé | Non validé |
| [illegible] | [illegible] | [illegible] | Masculin | Non validé | Validé | Validé | Validé |
| [illegible] | [illegible] | [illegible] | Masculin | Non validé | Validé | Validé | Validé |
| [illegible] | [illegible] | [illegible] | Masculin | Non validé | Non validé | Validé | Validé |
| [illegible] | [illegible] | [illegible] | Masculin | Non validé | Validé | Validé | Validé |
| [illegible] | [illegible] | [illegible] | Feminin | Non validé | Validé | Validé | Validé |
| [illegible] | [illegible] | [illegible] | Masculin | Non validé | Non validé | Non validé | Non validé |

FIGURE 3.10 – Rapport d'analyse : extrait du tableau récapitulatif

## 3.4 Discussion

Notre application permet d'effectuer des prédictions relativement à un semestre donné et propose à des fins d'analyses la possibilité d'avoir des visualisations par rapport à chaque UE et la possibilité d'obtenir un document PDF pour une synthèse.
Les phases de test et d'évaluation des algorithmes utilisés nous ont permis de faire ressortir les insuffisances liées à notre approche. En effet au niveau de la classification, il ressort qu'après prédiction, on obtient un fort de taux de faux positifs. Ce qui n'est pas étonnant au vu de la métrique choisie pour l'évaluation, qui pondère plus la sensibilité au profit de la précision. Ainsi, plusieurs étudiants qui valideraient normalement une UE , selon les modèles choisis pourraient ne pas la valider. En revanche, notre solution a le mérite de limiter la situation contraire, (c'est-à-dire prédire qu'un étudiant valide une UE alors que ce n'est pas le cas) qui peut s'avérer être plus dangereuse si les résulats des prédictions sont utilisés pour orienter ou cibler des cours de renforcement pour les étudiants qui ne valident pas. Dans ce sens des étudiants qui normalement auront besoin de renforcement n'en bénéficieront pas. Au niveau de la régression, les modèles utilisés arrivent à faire de bonnes prédictions mais il est à noter que pour certains cas ils s'éloignent vraiment des vraies valeurs. Cela peut suggérer une mauvaise représentativité des données (certaines tendances que les modèles n'ont pas su capturer au niveau des données d'apprentissage mais qui se révèlent être présentes au niveau des tests). Par ailleurs, l'ensemble des données utilisées pour l'entraînement des modèles n'est pas suffisant, aussi bien en ce qui concerne la quantité, que l'ensemble des différents attributs utilisés. A cet effet, l'étude de l'existant nous a révélé qu'on atteignait de meilleurs résultats en combinant aux notes antérieures des étudiants, des attributs à caractère social mais compte tenu des données disponibles, nous nous sommes malheureusement limités comme données à caractère social à l'âge et au sexe.
Il est à noter que pour les préliminaires à une prédiction, Pythia requiert un temps relativement long car il faudra générer pour le semestre de la prédiction des prédicteurs, l'équivalent dans notre cas de dix-huit (18) modèles, soit un prédicteur de classification et un préditeur de régression par UE , avec un total de neuf (09) UEs .

## Conclusion

Ce chapitre a fait l'objet d'un exposé des résultats obtenus à travers la présentation des performances des différents algorithmes utilisés pour les tests et une description sommaire de quelques fonctionnalités de Pythia. Les insuffisances liées à la solution ont été soulevées et seront utilisées comme base pour formuler les perspectives.

# Conclusion générale et perspectives

L'utilisation de l'apprentissage automatique est en pleine expansion et est également étendue au milieu de l'éducation pour la prédiction de performances académiques. Nous avons proposé un système capable d'effectuer la prédiction des performances académiques à l'IFRI. Plus précisement, nous avons réalisé une application Web basée sur des techniques d'apprentissage automatique qui permet de réaliser des prédictions sur les résultats des étudiants dans chaque UE d'un semestre donné, lesquels résultats sont accompagnés de visualisations sous forme de graphes. Les algorithmes utilisés pour effectuer les prédictions ont été choisis après une série de tests. Pour l'évaluation de leurs performances les métriques mesure $F_2$ et RMSE ont été respectivement utilisées pour la classification et la régression. Les prédictions sur certaines UEs sont relativement très bonnes (par exemple l'UE1 où les réseaux de neurones à propagation avant ont obtenu une mesure $F_2$ de 0,92 pour la classification et où les forêts d'arbres décisionnels ont commis une RMSE de 1,92 pour la régression). En revanche, par rapport à certaines UEs, on obtient des résultats moins précis (par exemple l'UE5 où la meilleure mesure $F_2$ est de 0,57 et l'UE8 où la RMSE minimale est de 2,93).

Des perspectives intéressantes sont envisagées pour ce travail. Par exemple, dans la vision d'une étude plus poussée et plus précise, il serait intéressant d'obtenir des données supplémentaires afin d'agrandir la base des données utilisées pour l'apprentissage et de mieux explorer les attributs à caractère social (par exemple la distance séparant la maison de l'étudiant à l'école, le fait que l'étudiant dispose d'une connexion internet, le niveau d'éducation des parents, etc.) et autres données académiques (par exemple le nombre d'absences aux cours) car ils renferment plusieurs aspects déterminants qui n'ont pas été pris en compte dans la présente étude. Dans le même sens, il serait important d'effectuer des prétraitements plus avancés sur les données et d'explorer plus en profondeur les algorithmes utilisés afin de faire un meilleur calibrage de leurs paramètres, dans le but d'obtenir de meilleurs résultats. Par ailleurs, il est envisageable que dans ses prochaines versions Pythia puisse déterminer quels sont exactement les principaux facteurs qui déterminent l'issue d'une UE. Un autre axe de recherche en ce qui concerne les travaux futurs à long terme serait l'ouverture du champ d'action de Pythia à d'autres universités/écoles à travers la mise en place d'une API (Application Programming Interface) adaptative. Il suffirait donc simplement de fournir les fichiers de données passées sur des étudiants de n'importe quelle école tout en précisant les UEs pour lesquelles les prédictions seront effectuées, pour que les prédicteurs des semestres concernés soient générés ; une phase de calibrage des paramètres des algorithmes sera donc automatiquement effectuée pour guider le choix de meilleurs modèles en fonction des données qui seront fournies. Enfin, il serait intéressant qu'en plus de l'analyse prédictive, Pythia puisse être un système de recommandations c'est-à-dire qu'elle exploiterait les résultats issus de l'analyse prédictive pour effectuer des suggestions.

# Bibliographie


[1] Tom M. Mitchell, *Machine Learning*, McGraw Hill, ISBN 0-07-042807-7, 1997.

[2] Tom M. Mitchell, *The Discipline of Machine Learning*, CMU-ML-06-108, 2006.

[3] Willi Richert, Luis Pedro Coelho, *Building Machine Learning Systems with Python*, Packt Publishing, 2013.

[4] Andreas C. Müller, Sarah Guido, *Introduction to Machine Learning with Python - A Guide for Data Scientists*, 2016.

[5] Sebastian Raschka, Vahid Mirjalili, *Python Machine Learning*, Second Edition - Fully revised and updated, 2017.

[6] Prateek Joshi, *Artificial intelligence with Python - Build real-world Artificial Intelligence applications with Python to intelligently interact with the world around you* , 2017.

[7] Pedro Domingos, *A Few Useful Things to Know about Machine Learning - Communications of the ACM*, 55(10), 78. doi :10.1145/2347736.2347755, 2012.

[8] Amirah Mohamed Shahiri, Wahidah Husain, Nur'aini Abdul Rashid, *A Review on Predicting Student's Performance using Data Mining Techniques*, School of Computer Sciences Universiti Sains Malaysia, 2015.

[9] Murat Pojon, *Using Machine Learning to Predict Student Performance*, University of Tampere, 2017.

[10] Andrew Ng, *Machine Learning and AI via Brain simulations*, Stanford University, 2013.

[11] Daud, A., Aljohani, N. R., Abbasi, R. A., Lytras, M. D., Abbas, F., & Alowibdi, J. S., *Predicting student performance using advanced learning analytics.* In Proceedings of the 26th International Conference on World Wide Web Companion (pp. 415-421), International World Wide Web Conferences Steering Committee, 2017.

[12] Paulo Cortez, Alice Silva, *Using Data Mining To Predict Secondary School Student Performance*, University of Minho, 2008.

[13] Andrew Y. Ng, Michael I. Jordan, *On Discriminative vs Generative classifiers : A comparison of logistic regression and naive Bayes*, Advances in neural information processing systems (pp 841–848), 2002.

[14] Pascal Roques, *UML 2 par la pratique*, Eyrolles, 2011.

[15] Wessel N. van Wieringen, *Lecture notes on ridge regression*, Version 0.20, 2018.

[16] Kaufman, Shachar et al., *"Leakage in Data Mining : Formulation, Detection, and Avoidance"*, *KDD*, 2011.

# Webographie


[17] DataRobot, *What does Prediction mean in Machine Learning ?*, https://www.datarobot.com/wiki/prediction/, consulté le 1er Août 2018.

[18] Analytics Vidhya, *Essentials of Machine Learning Algorithms (with Python and R Codes)*, https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/, consulté le 02 Août 2018.

[19] Clearbrain Blog,*The Two Types of Machine Learning*, https://blog.clearbrain.com/posts/the-two-types-of-machine-learning, consulté le 06 Août 2018.

[20] Coursera, *Kevyn Collins-Thompson , Key Concepts in Machine Learning*, https://www.coursera.org/learn/python-machine-learning/lecture/hrHXm/key-concepts-in-machine-learning, consulté le 13 Août 2018.

[21] Journal Du Net, *Le feature engineering, futur du data scientist*, https://www.journaldunet.com/solutions/expert/69609/le-feature-engineering--futur-du-data-scientist.shtml, consulté le 22 Août 2018.

[22] Cours d'initiation au machine learning, *Représentation : extraction de caractéristiques*, https://developers.google.com/machine-learning/crash-course/representation/feature-engineering?hl=fr, consulté le 22 Août 2018.

[23] Github, *Student Performance Prediction*, https://github.com/sachanganesh/student-performance-prediction, consulté le 23 Août 2018.

[24] Quora, *What is the list of algorithms present in machine learning ?*, https://www.quora.com/What-is-the-list-of-algorithms-present-in-machine-learning, consulté le 27 Août 2018.

[25] Analytics Vidhya, *Introduction to k-Nearest Neighbors : Simplified (with implementation in Python)*, https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/, consulté le 27 Août 2018.

[26] StatSoft, *Popular Decision Tree : Classification and Regression Trees (C&RT)*, http://www.statsoft.com/Textbook/Classification-and-Regression-Trees, consulté le 27 Août 2018.

[27] Towards Data Science, *The Random Forest Algorithm*, https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd, consulté le 27 Août 2018.

[28] Towards Data Science, *Support Vector Machine–Introduction to Machine Learning Algorithms*, https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47, consulté le 27 Août 2018.

[29] Towards Data Science, *Neural Networks : All YOU Need to Know*, https://towardsdatascience.com/nns-aynk-c34efe37f15a, consulté le 27 Août 2018.

[30] Towards Data Science, *Accuracy, Precision, Recall or F1 ?*, https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9, consulté le 28 Août 2018.

[31] UNSW – Computer Science and Engineering, *Definition of Bayesian Networks*, http://www.cse.unsw.edu.au/~cs9417ml/Bayes/Pages/Bayesian_Networks_Definition.html, consulté le 28 Août 2018.

[32] Quora, *What is the C4.5 algorithm and how does it work ?*, https://www.quora.com/What-is-the-C4-5-algorithm-and-how-does-it-work, réponse de Sumit Saha, consulté le 28 Août 2018.

[33] Towards Data Science, *Adaboost for Dummies : Breaking Down the Math (and its Equations) into Simple Terms*, https://towardsdatascience.com/adaboost-for-dummies-breaking-down-the-math-and-its-equations-into-simple-terms-87f439757dcf, consulté le 07 Novembre 2018.

[34] Towards Data Science, *Boosting algorithm : GBM*, https://towardsdatascience.com/boosting-algorithm-gbm-97737c63daa3, consulté le 07 Novembre 2018.

[35] Machine Learning Mastery, *A Gentle Introduction to k-fold Cross-Validation* , https://machinelearningmastery.com/k-fold-cross-validation/, consulté le 07 Novembre 2018.

[36] Clusteval, *F2-Score*, https://clusteval.sdu.dk/1/clustering_quality_measures/5, consulté le 07 Novembre 2018.

[37] Statistics How To, *RMSE : Root Mean Square Error*, https://www.statisticshowto.datasciencecentral.com/rmse/, consulté le 08 Novembre 2018.

[38] L'Agiliste, *Introduction aux méthodes agiles et Scrum* , https://agiliste.fr/introduction-methodes-agile14, consulté le 14 Novembre 2018.

[39] Supinfo, *Méthodologies de développement de logiciel*, https://www.supinfo.com/articles/single/6765-methodologies-developpement-logiciel, consulté le 14 Novembre 2018.

[40] Chart.js, *Simple yet flexible JavaScript charting for designers & developers*, https://www.chartjs.org/, consulté le 18 Novembre 2018.

[41] jQuery, *jQuery write less, do more*, https://jquery.com/, consulté le 18 Novembre 2018.

[42] Python, *What is Python ? Executive Summary*, https://www.python.org/doc/essays/blurb/, consulté le 18 Novembre 2018.

[43] Scikit-learn, *Scikit-learn Machine Learning in Python*, https://scikit-learn.org/stable/, consulté le 18 Novembre 2018.

[44] Django, The web framework for perfectionists with deadlines, *Why Django ?*, https://www.djangoproject.com/start/overview/, consulté le 18 Novembre 2018.

[45] PostgreSQL, *PostgreSQL : The World's Most Advanced Open Source Relational Database*, https://www.postgresql.org/, consulté le 18 Novembre 2018.

[46] GetBootstrap, *Bootstrap, The most popular HTML, CSS, and JS library in the world.*, https://getbootstrap.com/, consulté le 18 Novembre 2018.

[47] Documentation du Web - MDN, *Qu'est-ce que le JavaScript ?* , https://developer.mozilla.org/fr/docs/Learn/JavaScript/First_steps/What_is_JavaScript, consulté le 18 Novembre 2018.

[48] Comment ça marche, *CSS (Feuilles de style)* , https://www.commentcamarche.com/contents/230-css-feuilles-de-style, consulté le 19 Novembre 2018.

[49] Comment ça marche, *HTML - Langage* , https://www.commentcamarche.com/contents/498-html-langage, consulté le 19 Novembre 2018.

# Annexe

### Libellés des UEs utilisées pour l'étude

- UE1 : Analyse Mathématique
- UE2 : Mathématiques Appliquées
- UE3 : Fondamentaux des bases de données relationnelles
- UE4 : Concepts avancés des réseaux informatiques
- UE5 : Développement personnel
- UE6 : Systèmes avancés
- UE7 : Administration des bases de données relationnelles
- UE8 : Programmation procédurale
- UE9 : Technologie Web

### Page de connexion de Pythia

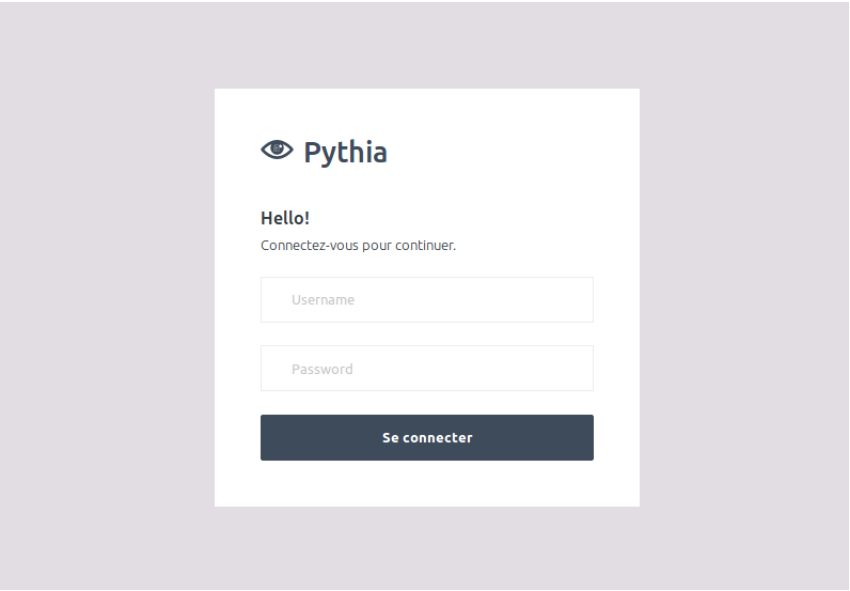



FIGURE 3.11 – Page de connexion de Pythia

## Dashboard après connexion

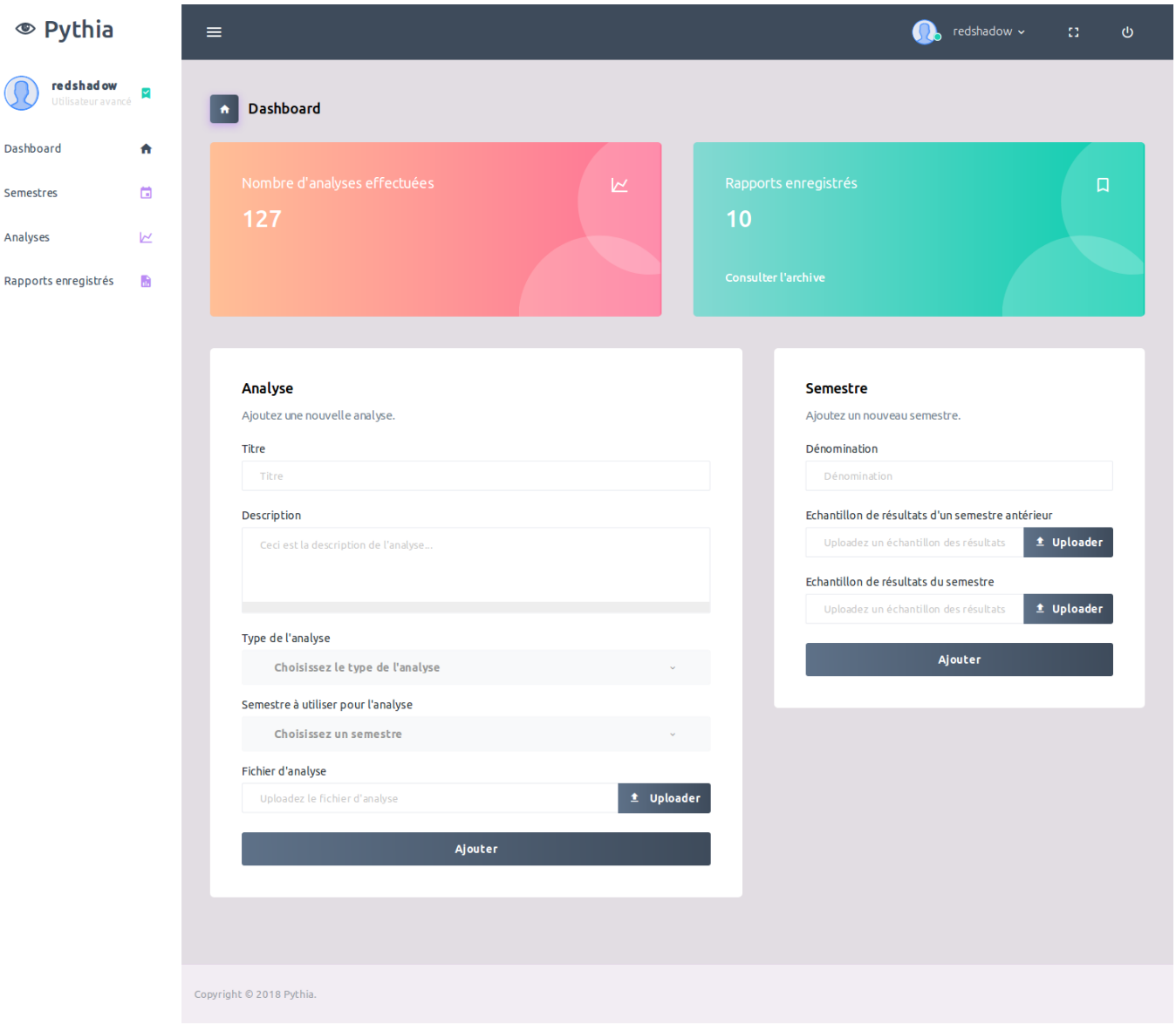
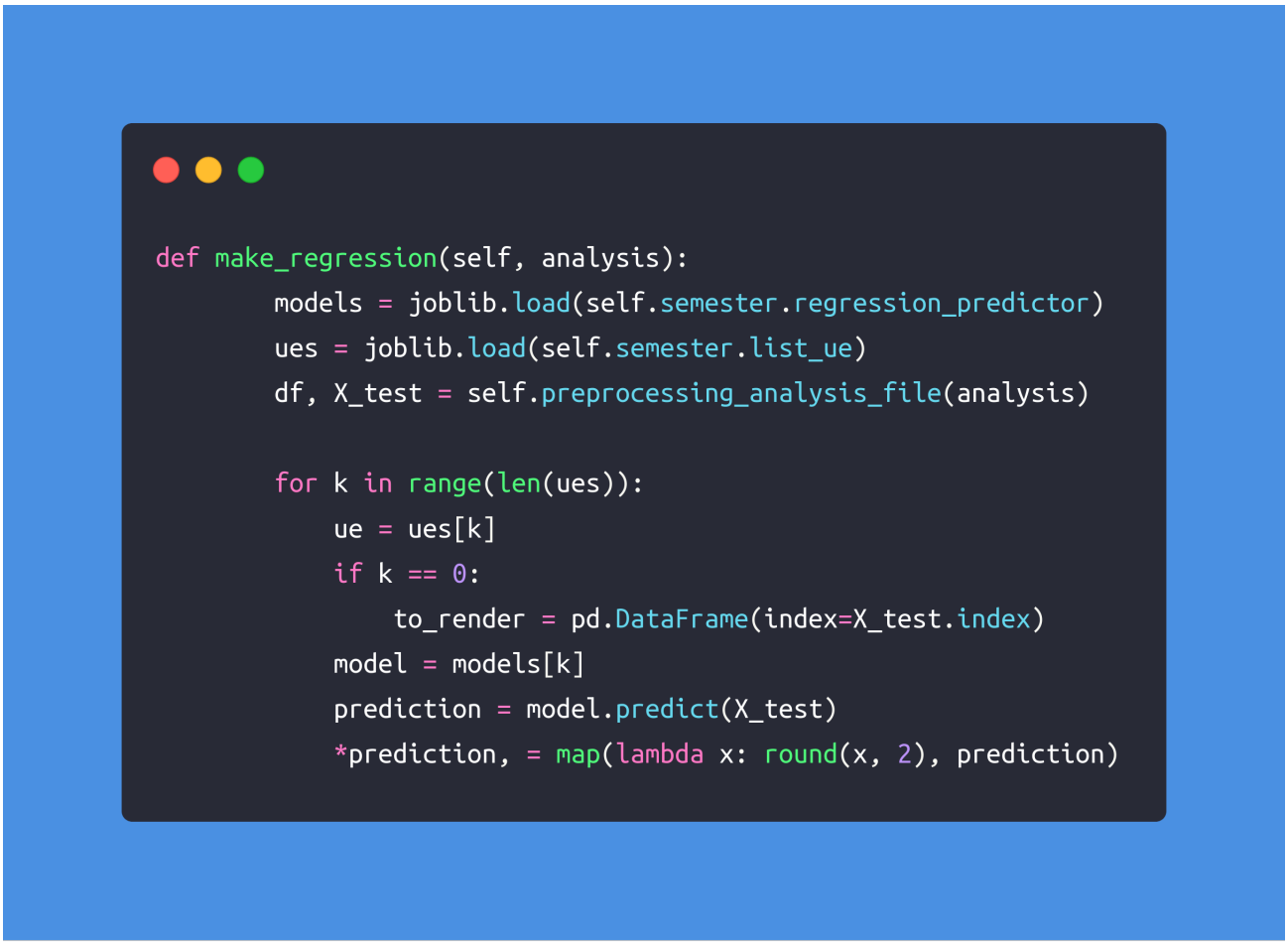



FIGURE 3.12 – Dashboard après connexion

**Extrait de code source pour la tâche de prédiction basée sur une analyse de type régression**



```
def make_regression(self, analysis):
        models = joblib.load(self.semester.regression_predictor)
        ues = joblib.load(self.semester.list_ue)
        df, X_test = self.preprocessing_analysis_file(analysis)

        for k in range(len(ues)):
            ue = ues[k]
            if k == 0:
                to_render = pd.DataFrame(index=X_test.index)
            model = models[k]
            prediction = model.predict(X_test)
            *prediction, = map(lambda x: round(x, 2), prediction)
```

FIGURE 3.13 – Extrait de code source pour la tâche de prédiction (régression)