LAGUE Pierre

Année Universitaire 2020-2021

DEPARTEMENT INFORMATIQUE

IUT de Vannes

8 rue Montaigne – BP 561

56017 Vannes CEDEX

RAPPORT DE STAGE DE FIN D'ETUDES

« Développement d'un modèle prédictif de dégradation des routes nationales »

Maître de stage : M. Valentin Villedieu, Application Development Analyst

Entreprise Accenture Technology, 44800 Nantes

Tuteur, IUT de Vannes : M. Minh-Tan Pham

Remerciements

Je tiens tout d'abord et tout particulièrement à remercier Valentin Villedieu, mon maître de stage au sein de l'entreprise Accenture Technology. Grâce à lui j'ai pu effectuer un stage de qualité dans l'une des entreprises informatiques les plus renommées au monde et je lui en suis sincèrement reconnaissant.

Je tiens à le remercier chaleureusement pour le temps consacré à l'encadrement de mon stage. J'ai pu constater la charge de travail à laquelle il était soumis, mais cela ne l'a jamais empêché de répondre à mes questions. Au-delà de ça, il a également fait en sorte de m'enseigner beaucoup sur la conception et la programmation informatique. Ce stage n'a donc pas été uniquement une manière de confronter mes connaissances à une situation concrète, mais il a également été un terrain d'apprentissage.

Je remercie l'équipe pédagogique de l'IUT de Vannes pour l'aide à la préparation de ce stage. Je remercie tout particulièrement Monsieur Pham, mon enseignant tuteur, pour le temps consacré au suivi de mon stage et pour son accessibilité lors de celui-ci. Je remercie également Jean-François Kamp pour sa complète implication et sa disponibilité lors de ma recherche de stage, ainsi que pour l'ensemble des outils de recherche de stage mis à la disposition des étudiants.

Je remercie l'ensemble des collaborateurs de Accenture Technology qui m'ont accueilli lors de mon arrivée dans l'entreprise et en particulier Olivier Demarez et Lucie Prunes pour leur disponibilité et leur accompagnement. La procédure d'accueil de stagiaire à Accenture Technology étant éprouvée, j'ai pu immédiatement trouver ma place dans l'entreprise. La manière dont les stagiaires sont responsabilisés et considérés chez Accenture est unique et je remercie le groupe de m'avoir accueilli.

Sommaire

Remerciements 1
1. Introduction 3
1.1. Présentation de l'entreprise 3
1.1.1. Le groupe Accenture 3
1.1.2. Lieu de travail 4
1.2. Responsable au sein de l'entreprise 6
1.3. Présentation du projet de stage 7
1.3.1. Talents for Innovation 7
1.3.2. Contexte 7
1.3.3. Présentation des technologies utilisées 8
2. Construction d'un modèle de machine learning « from scratch » 12
2.2. Etude de la stratégie de développement 13
2.3. La récolte des données 14
2.4. Enrichissement des données 17
2.4.1. API d'altimétrie IGN 17
2.4.2. Le trafic moyen journalier 18
2.4.3. Les données météorologiques 19
2.5. Machine Learning dans Dataiku 20
2.5.1. Ingénierie des données 20
2.5.2. Sessions d'entraînement des modèles 22
2.5.3. Les algorithmes de machine learning utilisés 24
2.5.4. Analyse de l'importance des facteurs de prédiction 29
2.6. Utilisation du modèle et ouverture 32
3. Analyse des données de vibration des routes 33
3.1. L'application Roads Reader 34
3.1.1. Ecran d'accueil 34
3.1.2. L'échantillonnage et la sauvegarde du fichier csv 35
3.2. Traitement des données issues de l'application 36
3.3. Prochaines étapes pour la fin de stage 37
4. Conclusion 38
5. Table des figures 39
6. Bibliographie 40

1. Introduction

1.1. Présentation de l'entreprise

1.1.1. Le groupe Accenture

Accenture a commencé comme la division de conseil en affaires et en technologie du cabinet comptable Arthur Andersen au début des années 1950. Suite à cela le développement des branches Accenture Social, Business et Consulting à fait naître Andersen Consulting, puis Accenture Consulting.

Figure 1 : Logo d'Accenture

Accenture est une entreprise de conseil et services informatiques, plus communément appelée SSII. Affichant un effectif total de 513 000 collaborateurs dans plus de 40 pays, - Accenture est l'un des leaders mondiaux sur ses domaines de compétence. Le Groupe Accenture a réalisé en 2019 un chiffre d'affaires de 43.2 milliards USD. Elle possède comme clients 96 des 100 plus grandes entreprises mondiales notamment les GAFAM.

Figure 2 : Accenture en bref

Accenture accompagne la transformation organisationnelle, technologique et digitale des plus grandes entreprises et administrations partout dans le monde.

Avec le développement des grandes mutations technologiques et digitales (Blockchain, IA, Big Data, Cloud computing, mobilité), Accenture allie savoir-faire en conseil et technologie pour délivrer la meilleure performance à ses clients.

Concrètement le groupe conçoit et met en œuvre des solutions technologiques qui améliorent la productivité de ses clients – allant jusqu'à gérer certaines de leurs activités.

1.1.2. Lieu de travail

1.1.2.1. Le Nantes Liberty Center

J'ai effectué mon stage dans les locaux de la branche Nantaise de Accenture, au « Nantes Liberty Center » situé dans la périphérie de Nantes.

Figure 3 : Façade du Nantes Liberty Center

Le pôle Nantais de Accenture emploie environ 700 collaborateurs avec une croissance de 25% ainsi que 32 clients multi-industrie, 70 projets en cours et un Global Network de plus de 50 centres de Services. Ce pôle travaille également en collaboration avec des entreprises locales.

1.1.2.2. Le Liquid Studio

Une des particularités de la branche Accenture Technology est son laboratoire de recherche et prototypage : le Liquid Studio.

Figure 4 : Intérieur du Liquid Studio

Le Liquid Studio est une instance du pôle de Nantes de Accenture dirigée par M. Olivier Demarez. Le Liquid Studio intervient dans la mise en place des projets des collaborateurs. Il évalue la faisabilité du projet et développe un prototype répondant à une problématique précise.

Dans les organisations des plateformes technologiques lors du développement d'un projet, le Liquid Studio prend en charge le process du Custom, des DevOps et du Digital. Le Liquid Studio entre aussi en scène pour les SAP.

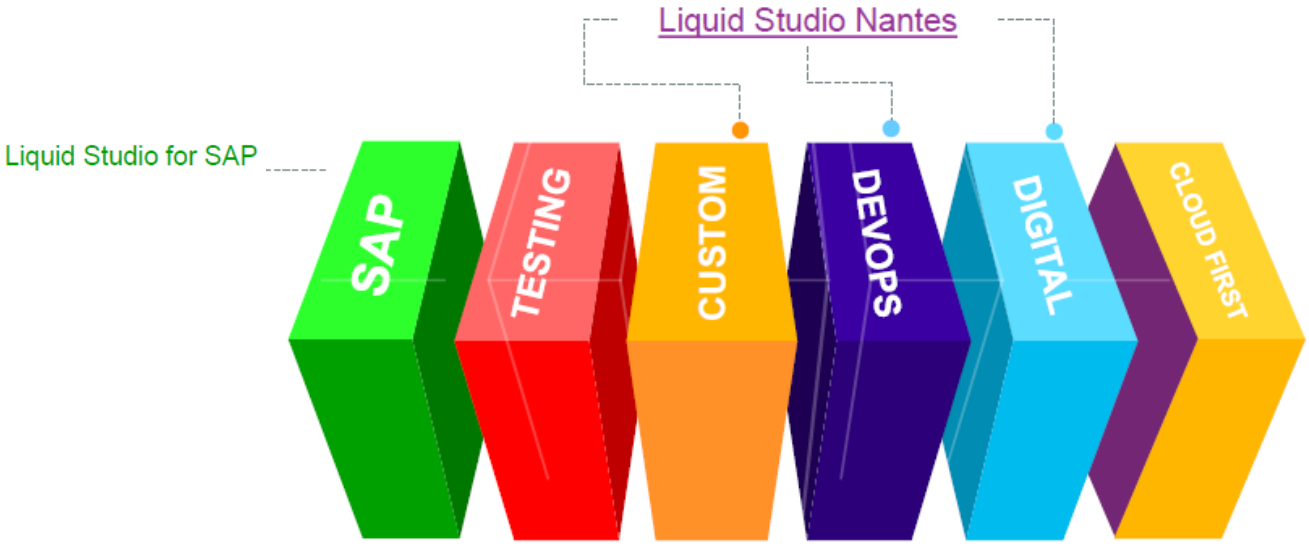

Figure 5 : Organisation en plateformes technologiques

1.2. Responsable au sein de l'entreprise

Valentin Villedieu

L'intitulé du poste de Valentin est « Application Development Analyst ».

Nous allons vous expliquer en quoi consiste le métier de « Application Development Analyst ».

Avant l'ère des CMS et autres progiciels, un programme informatique était généralement développé autour de trois personnes : un chef de projet, un analyste et un développeur. Chacun avait donc un rôle particulier. Dans un souci d'économie, le métier de développeur a fusionné avec celui d'analyste pour former le poste « d'analyste-programmeur ».

L'analyste-programmeur se charge de développer sous la direction du chef de projet, une partie d'un programme voire l'intégralité du logiciel demandé. Pour ce faire, l'analyste-programmeur consulte le cahier des charges composé par le chef de projet. Ainsi, il étudie et décompose les différentes fonctions et usages du programme attendu par le client. Ensuite, il détermine une ou plusieurs solutions techniques avant de développer l'application commandée.

Enfin, après avoir implémenté les fonctionnalités de l'application dans un langage de programmation, il composera lui-même la notice d'utilisation des fonctionnalités relatives à son travail.

1.3. Présentation du projet de stage

1.3.1. Talents for Innovation

Chaque année le groupe Accenture propose un concours appelé « Talents for Innovation ». Ce concours a pour but de pousser les collaborateurs à proposer des idées de projets innovantes en lien avec un thème particulier ou non.

Le projet est d'abord présenté puis élu par vote au niveau du pôle puis au niveau de la branche et enfin au niveau national. Le projet final reçoit le prix d'excellence des « Talents for Innovation » et est lancé en production pour être proposé comme un service aux clients d'Accenture.

Notre projet de stage est un des projets élu au niveau du pôle de Nantes. Nous avons donc la charge du premier développement du projet qui sera utilisé pour la soutenance du projet au niveau de la branche Accenture Technology.

1.3.2. Contexte

La qualité du revêtement routier est essentielle pour améliorer l'expérience de conduite et réduire les accidents de la route.

Les systèmes traditionnels de surveillance de l'état des routes sont limités dans leurs réponses temporelles (vitesse) et spatiales (couverture) nécessaires au maintien de la qualité globale des routes. Plusieurs systèmes alternatifs ont été proposés qui utilisent des capteurs montés sur les véhicules. En particulier, avec l'utilisation omniprésente des smartphones pour la navigation, l'évaluation de l'état des routes par smartphone est apparue comme une nouvelle approche prometteuse.

Notre projet de stage a donc pour but d'analyser différentes techniques d'apprentissage automatique supervisé multi classe pour classifier efficacement l'état de la surface des routes à l'aide des données trouvées en open-source et des données physiques (d'accéléromètre, gyroscope et de GPS). Notre travail se concentre sur la classification de trois étiquettes de classe principales : **"Bon Etat (BE)", "Etat Moyen (EM)", "Mauvais Etat (ME)".**

La problématique associée à ce projet est :

Comment prédire l'état de la surface d'une route à partir de données liées directement ou indirectement à la route à l'aide d'algorithmes de Machine Learning ?

Cette problématique peut se décomposer en d'autres problèmes induits par le fait que les données utilisées sont en open-source :

- La précision des données (qualité de la labellisation)
- La quantité de données accessibles gratuitement
- La pertinence des données acquises (leur apport au modèle)
- Faisabilité des analyses durant la période de stage
- Etc.

Une application disposant de ce modèle et qui bénéficierait d'un apport de données d'entraînement régulier représenterait alors un gain de temps de d'argent pour des instances de l'état comme Le Ministère de le Transition Ecologique. En effet, il suffirait alors d'une une requête pour savoir où les routes se dégradent au lieu d'aller sur place pour faire un échantillonnage.

1.3.3. Présentation des technologies utilisées

Après avoir pris connaissance du principe du projet et des spécificités que M. Alban Deumier (responsable Réalité Virtuelle au LS et concurrent aux Talents For Innovation soutenant ce projet) m'avait communiqué, nous avons dû prendre en main l'outil sur lequel nous passerions une majeure partie de notre temps : DataIku DSS.

1.3.3.1. Présentation de DataIku DSS

Dataiku DSS (Data Science Studio) est une plateforme de développement intégrée, destinée aux professionnels des données. Elle permet de convertir efficacement les données en prédictions. Il s'agit d'un outil tout-en-un permettant de développer un projet de bout en bout, de la préparation au déploiement.

Le logiciel de Dataiku permet d'améliorer une infrastructure existante qu'il s'agisse d'une Data Warehouse SQL ou d'un cluster Spark. C'est un environnement permettant la coexistence entre tous les standards de technologies Big Data et les différents langages.

Figure 6 : Logo de DataIku

Dataiku est née en France à Paris. Florian Douettau, CEO, Clément Stenac, Marc Batty et Thomas Cabrol fondent la startup éponyme. Après un lancement parisien dès 2013, la startup traverse l'atlantique et installe son siège social à New York. En décembre 2018, elle a levé 101 millions de dollars.

En Avril 2017, l'IUT de Vannes a accueilli un Data Analyst travaillant chez DataIku pour une présentation de l'outil dans l'amphithéâtre A.

Les cas d'usage de Dataiku DSS sont nombreux. La plateforme peut servir pour les analyses marketing, la détection de fraude, les graphiques analytiques, la gestion de données, la prévision de demande, les analyses spatiales, l'optimisation de valeur, la maintenance prédictive ou les analyses CRM**. :**

Ses fonctionnalités sont multiples :

- La manipulation de données
- La visualisation de données en workflow
- Machine learning (prediction, clustering, dim. Reduction, DL)
- Forte connectivité (Cloud, SQL, Oracle, Hadoop, …)

Son émergence dans le monde du traitement des données est grâce à ses principaux atouts qui le différencie des autres outils comme le Google Big Query ou Microsoft Azur :

- Collaboration (versioning, listes de tâches, commentaires sur chaque objet, etc.)
- Adaptabilité (tout peut être codé ou modifié avec python, R ou SQL)
- Adaptation à tous les niveaux de compétences (de la simple consultation d'un tableau de bord au développement complexe de l'apprentissage en profondeur / Deep Learning)
- Workflow (vue de bout en bout sur le traitement des données, des connexions au déploiement)

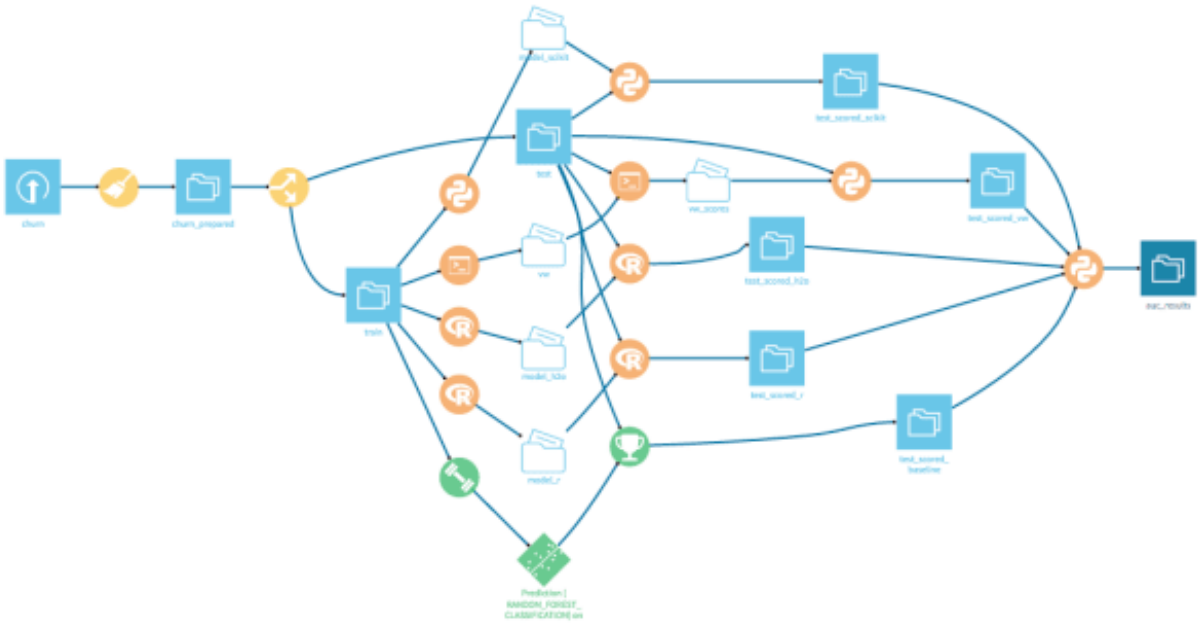

Figure 7 : Exemple de workflow dans DataIku

Cependant DataIku à certaines limitations. En effet, l'ingénierie et nettoyage des données est mal guidée (les fonctionnalités sont disponibles et faciles à manipuler mais il faut savoir quoi faire). De ce fait, il est difficile de commencer un projet nécessitant beaucoup de manipulation de données sans connaissance préliminaires.

La sélection et l'optimisation de modèles sont difficiles pour les non-initiés (les fonctionnalités sont disponibles mais non expliquées). Une des principales fonctionnalités de DataIku c'est le déploiement de modèles d'apprentissage automatique. Cependant, il est important de mener une étude analytique au préalable pour savoir quel modèle utiliser et avec quels paramètres. Beaucoup de ressources sont disponibles dans la documentation mais il faut savoir où regarder.

Enfin, les algorithmes déjà codés sont limités à un type de données basique (vous devez coder vous-même si vous voulez appliquer un réseau de neurones sur des images). Ce qui limite les applications des modèles d'apprentissage automatique à du machine learning ou du deep learning sur des données préparées.

Avec ses avantages et malgré ses limitations, qui peut être intéréssé par DataIku DSS ? En réalité, cet outil est accessible pour beaucoup de profils. Par exemple, les Data scientists reprénsentent une grande partie des clients car les processus qu'ils utilisent très fréquemment sont plus rapides et plus simples à mettre en œuvre grâce à Dataiku. Ils n'ont pas de limitation due au low-code car tout est modifiable à la source des programmes. Enfin, ils peuvent facilement expliquer les résultats à des personnes non initiées.

Les personnes intéressées par les sciences des données sont également des clients. En effet, la création d'algorithmes de machine learning se fait rapidement et sans connaissances en statistiques. C'est aussi un très bon moyen d'apprendre les sciences des données et l'apprentissage automatique.

Enfin, toutes les personnes voulant avoir un aperçu de leurs données. Car il est possible de visualiser facilement tout ce dont on a besoin. Il existe aussi des fonctionnalités de tableau de bord très efficaces.

1.3.3.2. Autres technologies

Tous les scripts additionnels utiles à la préparation des données et à l'enrichissement du modèle ont été réalisés en Python.

Pour manipuler les fichiers csv et un nombre de données important j'ai utilisé les librairies Numpy et Pandas.

NumPy apporte la puissance de calcul de langages comme C et Fortran à Python, un langage beaucoup plus facile à apprendre et à utiliser. Cette puissance s'accompagne de simplicité : une solution en NumPy est souvent claire et élégante.

Pandas est un outil d'analyse et de manipulation de données open source rapide, puissant, flexible et facile à utiliser, construit à partir du langage de programmation Python.

Pour l'analyse des vibrations abordée dans la partie 3, j'ai utilisé la librairie Scipy.

SciPy est une bibliothèque Python gratuite et open-source utilisée pour le calcul scientifique et l'informatique technique. SciPy contient des modules pour l'optimisation, l'algèbre linéaire, l'intégration, l'interpolation, les fonctions spéciales et le traitement du signal et de l'image.

Pendant notre projet, nous avons dû développer du code Python en dehors de l'outil DataIku. Pour cela nous avons utilisé Visual Studio Code qui est un IDE (environnement de développement intégré).

Visual Studio Code est un éditeur de code source gratuit créé par Microsoft pour Windows, Linux et macOS. Ses fonctionnalités incluent la prise en charge du débogage, de la coloration syntaxique, de la complétion de code intelligente et du remaniement de code.

2. Construction d'un modèle de machine learning « from scratch »

L'objectif final de cette construction est d'obtenir un modèle de machine learning auquel on donnerait comme entrée le nom d'une route. Après avoir analysé les données liées à cette route, le modèle devrait être en mesure de la classifier en 3 états :

- Bon état
- Etat moyen
- Mauvais état

La point suivant précise et explique les termes de notions de modèle d'apprentissage automatique et algorithme d'apprentissage automatique.

2.1. Les algorithmes et modèles de machine learning

L'apprentissage automatique implique l'utilisation d'algorithmes et de modèles d'apprentissage automatique. Pour les débutants, c'est très déroutant car souvent "algorithme d'apprentissage automatique" est utilisé de manière interchangeable avec "modèle d'apprentissage automatique". En tant que développeur, notre intuition avec les "algorithmes" tels que les algorithmes de tri et les algorithmes de recherche nous aideront à lever cette confusion. Ce point va permettre de bien différencier algorithme et modèle de machine learning pour que la suite de ce rapport soit plus claire.

Qu'est-ce qu'un algorithme de machine learning ?

Dans le domaine de l'apprentissage automatique, un "algorithme" est une procédure qui est exécutée sur des données pour créer un "modèle" d'apprentissage automatique. Les algorithmes d'apprentissage automatique effectuent une "reconnaissance des formes" par exemple. Les algorithmes "apprennent" à partir de données, ou sont "adaptés" à un ensemble de données.

Il existe de nombreux algorithmes d'apprentissage automatique.

Par exemple, nous disposons d'algorithmes de classification, tels que les k-voisins les plus proches (K closest Neighbours) ou la Random Forest (Forêt d'arbres de décision aléatoire). Il existe des algorithmes de régression, comme la régression linéaire (linear regression), et des algorithmes de regroupement, comme les k-means.

En tant que tels, les algorithmes d'apprentissage automatique ont un certain nombre de propriétés :

- Les algorithmes d'apprentissage automatique peuvent être décrits à l'aide de mathématiques et de pseudocodes.
- L'efficacité des algorithmes d'apprentissage automatique peut être analysée et décrite.
- Les algorithmes d'apprentissage automatique peuvent être mis en œuvre avec n'importe lequel des nombreux langages de programmation modernes.

Qu'est-ce qu'un modèle de machine learning ?

Un "modèle" dans l'apprentissage automatique est le résultat d'un algorithme d'apprentissage automatique exécuté sur des données.

Un modèle représente ce qui a été appris par un algorithme d'apprentissage automatique.

Le modèle est la "chose" qui est sauvegardée après l'exécution d'un algorithme d'apprentissage automatique sur des données d'apprentissage et représente les règles, les chiffres et toute autre structure de données spécifique à l'algorithme nécessaire pour faire des prédictions.

Quelques exemples peuvent rendre cela plus clair :

- L'algorithme de régression linéaire aboutit à un modèle composé d'un vecteur de coefficients avec des valeurs spécifiques.
- L'algorithme de l'arbre de décision (Random Forest) donne un modèle composé d'un arbre d'instructions "si-alors" avec des valeurs spécifiques.
- Les algorithmes de réseau neuronal donnent un modèle composé d'une structure graphique avec des vecteurs ou des matrices de poids ayant des valeurs spécifiques.

Un modèle d'apprentissage automatique est plus difficile à comprendre pour un débutant car il n'existe pas d'analogie claire avec d'autres algorithmes en informatique. La meilleure approche consiste à considérer le modèle d'apprentissage automatique comme un "programme". Le "programme" du modèle d'apprentissage automatique comprend à la fois des données et une procédure d'utilisation des données pour faire une prédiction.

A présent, avant de commencer à développer ce modèle, concentrons-nous sur la stratégie de développement.

2.2. Etude de la stratégie de développement

Le principe d'étude de la surface d'axes routier n'ayant été réalisé qu'en sujets de recherche pour thèse ou masters (principalement aux Etats-Unis d'Amérique), nous pouvions nous lancer dans le développement d'un prototype.

Cette notion de « prototype » signifie que le travail effectué n'a pas pour but d'être utilisé directement, mais pourra servir de base à la définition d'un futur projet.

L'analyse de la stratégie commence par une série de questions :

- Quel modèle : quel type d'algorithme allons-nous utiliser ?
- Quelles données : sur quelles données le modèle va-t-il se baser ?
- Pourquoi : ce modèle aura-t-il une application concrète ?
- Comment l'améliorer : si le modèle ne propose pas de bons résultats, comment l'améliorer ?

Afin de répondre aux questions précédentes, il est important de suivre une méthode de construction précise. Le diagramme suivant illustre bien la stratégie que je vais vous présenter :

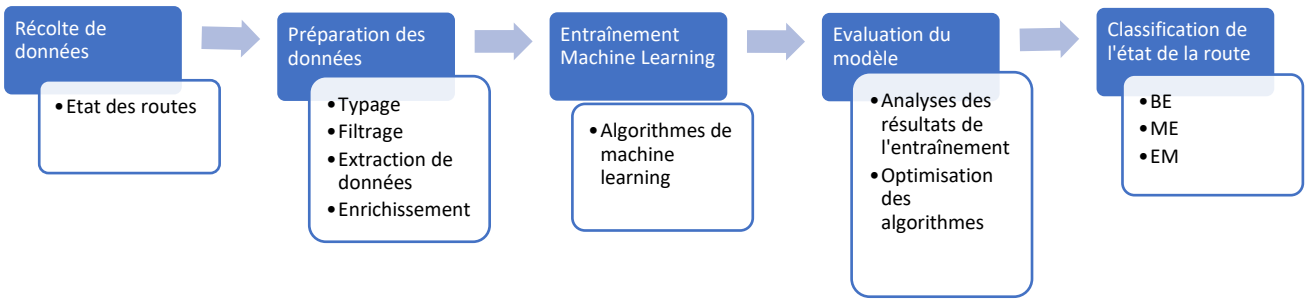

Figure 8 : Diagramme de développement d'un modèle de Machine Learning

2.3. La récolte des données

Nous avons commencé par visiter le site https://www.data.gouv.fr/ et nous avons trouvé 2 datasets :

- Classes d'état des chaussées du réseau routier national à partir de 2019.
- Classes d'état des chaussées du réseau routier national non concédé entre 2015 et 2018.

Chacun des datasets contenait les informations suivantes :

- Nom (code route)
- Longueur de l'échantillonnage
- Coordonnées de début (D) et de fin (F) d'échantillonnage exprimées en coordonnées Lambert93 (X, Y, z=0) ou par un système de repérage routier composé de 4 attributs :
  - pr = Point de repère routier
  - depPr = département où se situe le PR
  - concessionPr = indique si le PR se trouve sur une section concédée (C) ou non (N)
  - abs = abscisse ou distance (en mètres) séparant le point du PR auquel il se rattache
- cote = précise si le PR se trouve sur une chaussée séparée droite (D) ou gauche (G) ou sur une route à chaussée unique (I)
- Son état :
  - BE : bon état
  - EM : état moyen
  - ME : mauvais état

Cependant, à la suite d'une erreur de chargement de fichier de la part du site, le dataset de 2019 était incomplet. Nous avons donc utilisé les données de 2015 à fin 2017.

Après avoir chargé les jeux de données dans DataIku nous avons procédé à la préparation de ses données.

2.3.1. La préparation des données de base

Après une première observation des données, nous avons remarqué qu'il y avait des problèmes de typage (ex : une colonne numérique lue comme une chaîne de charactères) ainsi que de format. L'outil DataIku nous permet de visualiser des données simplement et d'appliquer des opérations et fonctions permettant un formatage et une préparation rapide des données. Les étapes suivantes ont été réalisées sur Visual Studio Code (un environnement de développement intégré) et DataIku.

2.3.1.1. La projection Lambert93

Dans notre dataset, les coordonnées étaient reportées en projection **Lambert93** (projection géographique) ce qui rendait les coordonnées inexploitables. J'ai donc implémenté une fonction en python qui prends en entrée un fichier csv et qui transforme des données en Lambert93 en coordonnées GPS exploitables. Voici un schéma explicatif :

Figure 9 : Diagramme de la conversion des systèmes de coordonnées

Une telle opération sur un fichier csv de plus de 90 000 entrées allait prendre beaucoup de temps (complexité en O(n)). C'est pourquoi j'ai utilisé un procédé algorithmique nommé « Vectorisation ». La **vectorisation** est le processus de conversion d'un programme informatique à partir d'une implémentation scalaire, qui traite une seule paire d'opérandes à la fois, à une implémentation vectorielle qui traite une opération sur plusieurs paires d'opérandes à la fois. En appliquant ce procédé, nous sommes passé d'une dizaine de minutes d'exécution à seulement une dizaine de secondes.

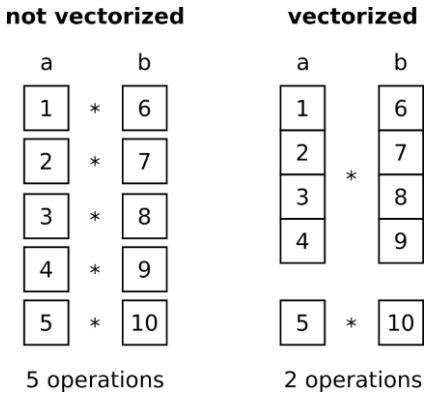

Figure 10 : Exemple de Vectorisation

Après avoir généré un fichier csv contenant les données GPS, nous avons importé ce fichier dans Dataiku. À la suite de cela nous avons joint le jeu de donné de départ et celui contenant les coordonnées GPS par une clé composée des éléments suivants : **la route, le point de repère routier, le département, la distance entre le point gps et le repère routier et enfin le coté de la route** :

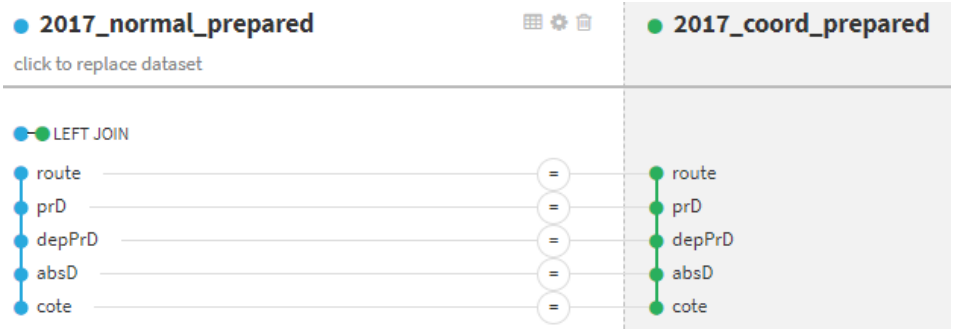

Figure 11 : Jointure selon une clé composée dans Dataiku

On voit sur l'illustration précédente que le résultat issu de la jointure entre le dataset de 2017 de base et celui contenant les coordonnées GPS associées sera constitué des lignes qui vérifient l'égalité de chaque composant de la clé. Nous avons effectué ce processus pour chaque jeu de données de base de l'année 2015 à 2019. Après s'être occupé de la préparation des données GPS, nous avons pu avancer au nettoyage des données.

2.3.1.2. Nettoyage des données

Un autre problème lié à la préparation des données est le traitement des valeurs manquantes, des lignes invalides et des colonnes inutiles. Dans l'environnement de Dataiku, nous pouvions vérifier l'intégrité des colonnes comme ceci :

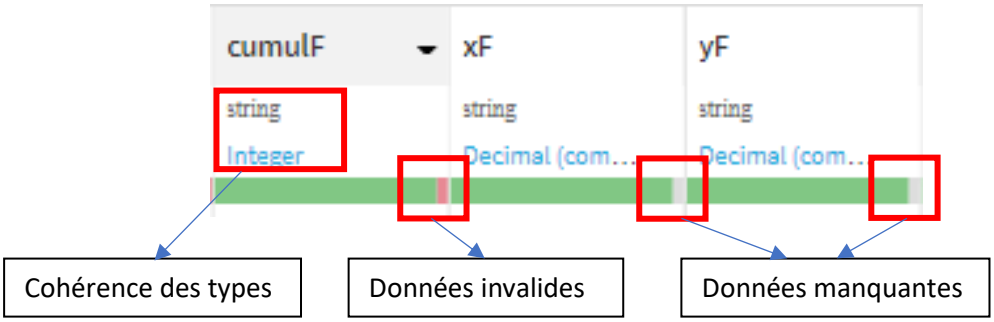

Figure 12 : Visualisation de la validité des colonnes dans Dataiku

Nous avons donc rempli les lignes invalides avec une valeur moyenne ou simplement supprimé les lignes vides. Comme l'ensemble de données avait été créé en France, les valeurs décimales avaient une virgule (,) et non un point (.) ce qui faisait planter les scripts. Heureusement, Dataiku DSS nous permet de manipuler les données très facilement en créant des recettes qui formateront et modifieront nos données :

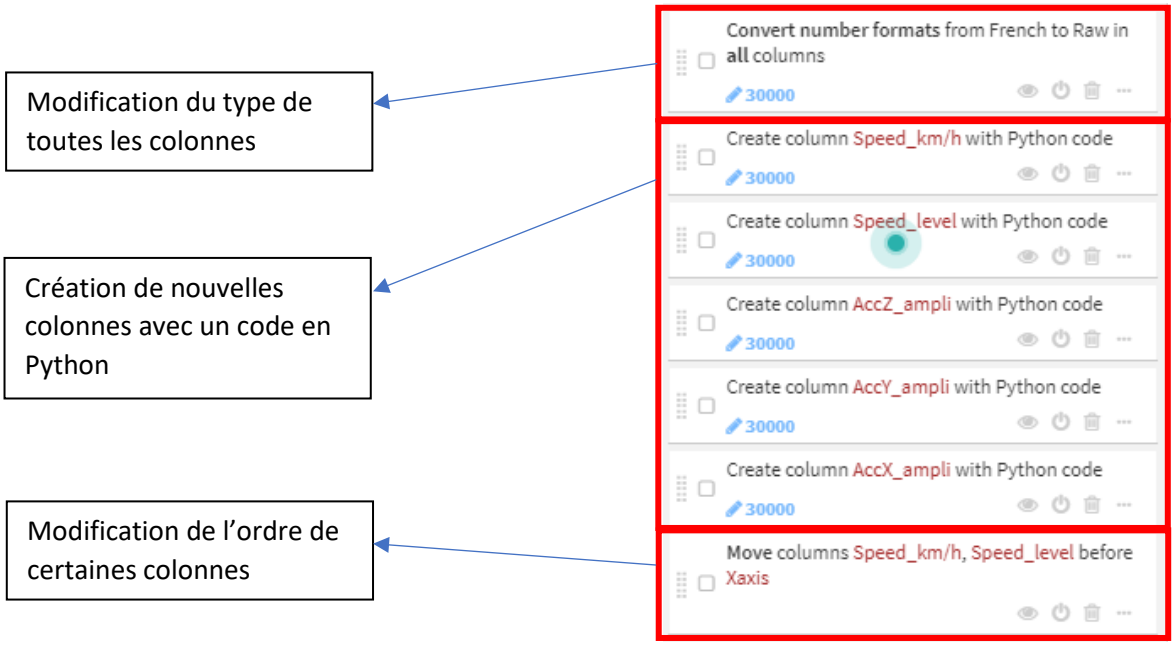

Figure 13 : Recette de préparation d'un jeu de donnée sur Dataiku

Ces deux étapes ont permis de résoudre les problèmes principaux dans la préparation des ensembles de données de départ. Après s'être assurés que notre jeu de données de départ était propre et prêt à être utilisé, nous nous sommes concentrés sur les points suivants : **quels jeux de données externes allaient pouvoir améliorer la qualité de nos données ?**

2.4. Enrichissement des données

L'enrichissement des données est un processus qui nous permet de fournir des données supplémentaires à notre jeu de données de base pour essayer d'obtenir des facteurs de prédiction plus précis. Nous avions en tête quelques données telles que : l'altitude moyenne de la route, le trafic moyen de voitures ou de poids lourds sur une route (regroupés par départements), des données météorologiques (pluie, température, etc.) et des informations sur la façon dont une route a été construite (âge et matériaux utilisés).

2.4.1. API d'altimétrie IGN

Le premier élément de données que nous avons ajouté était l'altitude (en mètres) pour un couple de coordonnées donné. Pour cela, nous avons utilisé l'API d'élévation de l'IGN (Institut national de l'information géographique et forestière). Nous avons contacté l'entreprise par mail afin d'obtenir une clé API (clé nous permettant d'utiliser leurs ressources). Une fois la clé reçue, nous avons effectué une requête pour chaque couple de coordonnées des jeux de données des années 2015 à 2019.

Après avoir récolté toutes les données nous avons exporté le résultat dans un fichier CSV contenant les coordonnées X, Y et Z et la clé composé qui nous permettra de joindre ce dataset à celui dans DataIku.

Voici un schéma explicatif de la démarche entreprise :

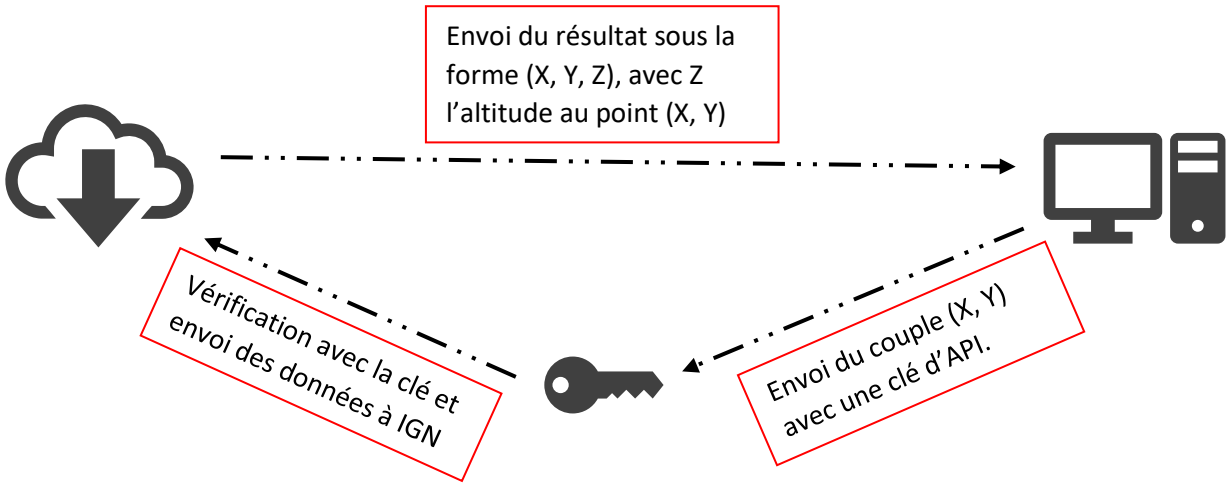

Figure 14 : Fonctionnement de l'API IGN.

Avoir l'altitude d'un couple de coordonnées nous permettra d'extraire de nouvelles informations comme :

- La route est-elle en zone montagneuse / littorale ?
- La météo associée à des hautes altitudes

2.4.2. Le trafic moyen journalier

Savoir combien de voitures et de poids lourds passent sur une route est un facteur important dans l'évaluation de l'état d'une route. C'est pourquoi nous avons utilisé ces jeux de données trouvés sur https://www.data.gouv.fr/: Le trafic moyen journalier annuel sur le réseau routier national de 2015 à 2017. Les caractéristiques de ce jeu de données sont assez similaires à celles des jeux de données de base. Cependant ils disposent d'informations très intéressantes :

- TMJA = trafic moyen journalier annuel
- RatioPL = pourcentage de poids-lourds du TMJA

Nous avons donc préparé ce jeu de données en ne gardant que la clé composée pour l'intégrer à nos données dans DataIku :

- Route = le nom de la route
- pr = point de repère routier
- depPr = Département où se trouve le PR
- absD = abscisse ou distance (en mètres) séparant le point (x;y) du PR auquel il est rattaché.

Ensuite, nous avons empilé les jeux de données pour avoir les valeurs de 2015 à 2017 et n'avons gardé que les colonnes importantes. Voici une illustration du procédé :

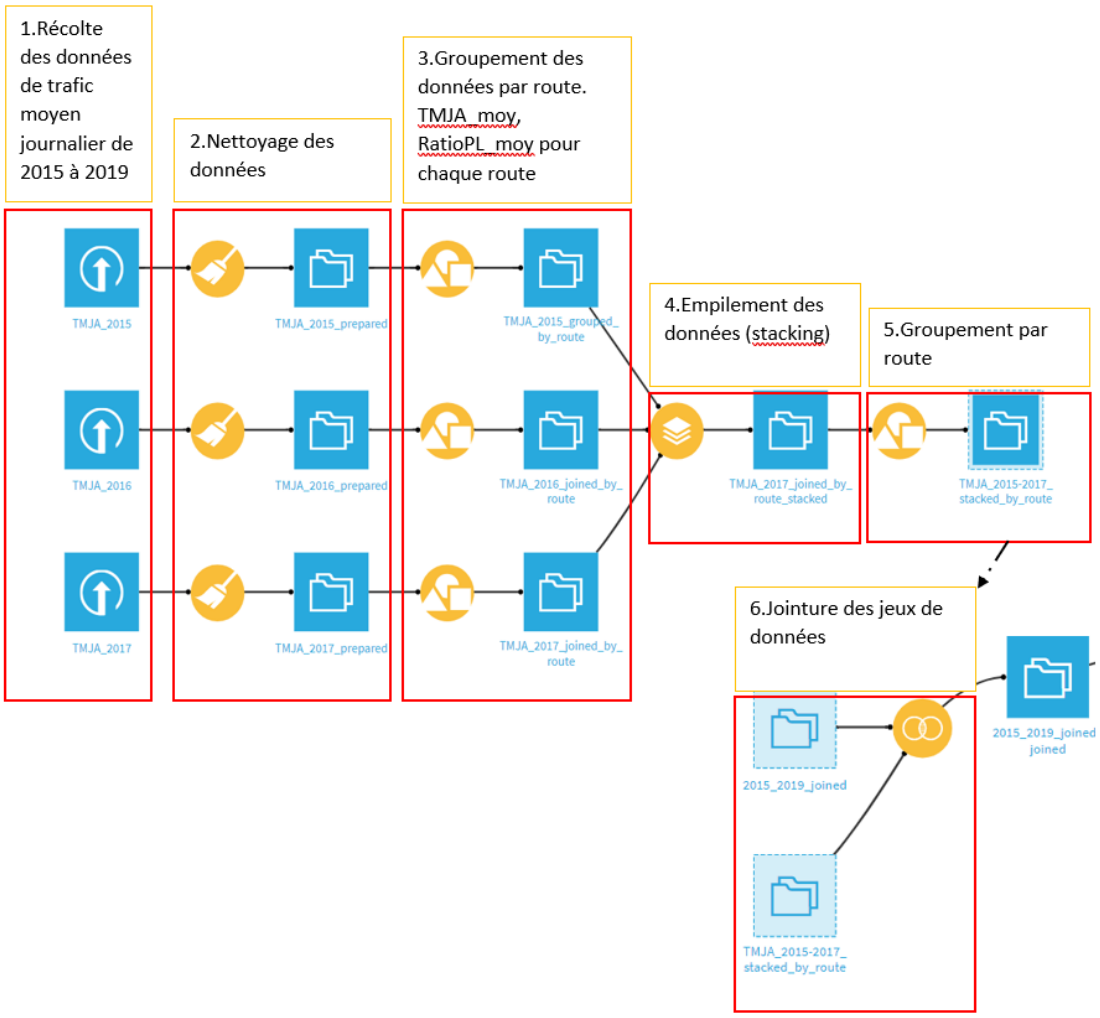

Figure 15 : Diagramme du processus d'enrichissement des données

L'intérêt de cet enrichissement est d'avoir de nouveaux facteurs de prédiction. Ici, le fait de savoir combien de voitures et poids lourds passent sur une route tous les ans nous donne une information importante sur l'état de cette route. En effet, le trafic est l'une des principales causes de dégradation des axes routiers.

2.4.3. Les données météorologiques

Dans le point précédent nous avons présenté l'utilité de connaître des informations qui ont un lien avec l'état de la route afin de mieux le prédire. La première cause de dégradation des routes en France est : les variations de météo et les intempéries. Disposer d'informations météorologiques liées à une route ou un département serait alors un bon facteur de prédiction.

Le gel par exemple est formé sur les routes lorsque la température est inférieure à 0° à la surface de la route et quand de petites fissures peuvent stocker l'eau. Si l'eau ne s'évapore pas avec le passage des véhicules et que la température le permet, elle va geler, augmentant son volume et élargissant la fissure. Ce même mécanisme de dégradation marche aussi avec les couvertures végétales (racines).

Figure 16 : Exemple de dégradation des routes (gel et dégel à gauche, végétation à droite)

Nous avons donc recherché des datasets contenant des informations météorologiques et des informations sur la couverture végétale des routes. Nous avons visité plusieurs sites d'open data comme https://www.google.com/publicdata/directory, https://data.europa.eu/data/datasets ou encore https://www.opendatasoft.com/. Nous n'avons pas réussi à récupérer des données assez précises concernant la couverture végétale des routes de France. Cependant nous avons pu extraire du site https://donneespubliques.meteofrance.fr/ ces données :

- Observation historique de la météo France 2015, 2016, 2017, 2019

Ce dataset contenait un grand nombre d'informations très complexes (analyses complexes météorologiques) mais qui, avec un travail assidu de data engineering, nous permettrait d'obtenir de bons facteurs de prédiction. Le seul problème était le manque de données dans certains départements (25% des départements). Ces départements manquants n'avaient pas participé à l'étude regroupant les données des observatoires météorologiques français. Ce manque de données voudrait dire que pour adapter notre jeu de donnée de base au jeu de données météorologiques il fallait se séparer de 25% des départements, ce qui est une trop grosse perte de données.

Nous avons donc opté pour des données moins complexes mais mieux référencées. Sur le site https://www.data.gouv.fr/ nous avons trouvé deux datasets :

- Taux de précipitation moyen des départements français de 2015 à 2017
- Analyse quotidienne des températures régionales de 2015 à 2017

Cependant, l'échelle des deux datasets n'était pas la même, l'un était à l'échelle régionale, l'autre à l'échelle départementale. Nous avons donc téléchargé un fichier csv contenant le nom et le code de chaque département pour chaque région métropolitaine française.

Le procédé suivi est le suivant :

Dataset contenant les départements pour chaque région : **dep_reg**

Dataset contenant les données de température : **temp_dts**

Dataset contenant les données de pluviométrie : **pluvio_dts**

Dataset de d'analyse des routes (dit « de base ») : **base_dts**

I. **Préparation** des données de tous les datasets en supprimant les lignes non valides
II. **Groupement des températures quotidiennes** par **code de région** dans **temp_dts**
III. **Jointure** entre **dep_reg** et **temp_dts** avec comme clé le **nom de la région**
IV. **Groupement** des données par **code de département** dans le dataset issu de l'étape 3) (temp_joined)
V. **Empilement** des données de pluviométrie de 2015 à 2017 donnant **pluvio_dts**
VI. **Jointure** de **pluvio_dts** et **temp_joined** avec comme clé le **nom des départements**

A la fin de ce procédé nous avons obtenu un jeu de données référençant la **température maximale, minimale, moyenne et le taux de pluviométrie (en mm) par département de 2015 à 2017**.

Maintenant que nous disposons de données enrichies, nous pouvons travailler sur l'optimisation et le déploiement d'un modèle de machine learning dans DataIku.

2.5. Machine Learning dans DataIku

DataIku DSS nous donne accès à de nombreux algorithmes d'apprentissage automatique ou même des programmes prêts à l'emploi que vous pouvez modifier comme vous le souhaitez dans un environnement appelé « the Lab ». Dans notre cas, nous utiliserons les algorithmes de machine learning fournis par DataIku. Cependant, avant d'entraîner un algorithme de machine learning il faut préparer des données faciles à interpréter via un processus appelé « l'ingénierie des données ».

2.5.1. Ingénierie des données

L'ingénierie des données est l'aspect de la science des données qui se concentre sur les applications pratiques de la collecte et de l'analyse des données. Pour tout le travail que les scientifiques des données font pour répondre à des questions en utilisant de grands ensembles d'informations, il doit y avoir des mécanismes pour collecter et valider ces informations. Ici, nous allons modifier les données dont nous disposons pour faciliter la compréhension de notre modèle. Nous pouvons le faire en normalisant des colonnes ou en générant des colonnes binaires à partir de colonnes numériques processus de « simplification de colonnes numériques »).

2.5.1.1. Extraction de données

Nous pouvons extraire des données de colonnes comme le nom de la route. Comme cette colonne n'est pas numérique et a beaucoup de valeurs distinctes, nous pouvons la diviser en 2 classes -> routes nationales et autoroutes. A partir de cela, nous créons 2 colonnes "estNationale" et "estAutoroute". Ces colonnes ont deux valeurs : Vrai ou Faux (1 ou 0).

Nous pouvons également copier le processus sur la liste des départements, puisqu'ils n'ont pas de valeur logique (ce sont juste des codes) nous pouvons créer 2 colonnes "est_zone_montagne" et "est_zone_littorale". Le code pour l'extraction des colonnes « est_zone_montagneuse » et « est_zone_littorale » est le suivant :

```
#This function is intern to DataIku DSS.
#The process function takes each the dataset as a DataFrame and itterates over each rows.
#This function checks if a department is in the list of the departments that are close to the sea.
def process(row):
    liste = [59, 62, 80, 76, 27, 14, 50, 35, 22, 29, 56, 44, 85, 17, 33, 40, 64, 66, 11, 34, 30, 13, 83, 6]
    ret = False
    if str(row['depPrD']) in str(liste):
        ret = True
    else:
        ret = False
    return ret
```

Figure 17 : Fonction python utilisée pour la génération de colonnes booléennes

Dans DataIku DSS, un ensemble de colonnes générées en suivant ce processus d'ingénierie des données est représenté comme ceci :

est_autoroute	est_nationale	alti_D	en_zone_montagneuse	en_zone_littorale
boolean	boolean	double	boolean	string
Text	Text	Decimal	Boolean	Boolean
True	False	26.86	false	True
True	False	26.95	false	True
True	False	27.78	false	True
True	False	27.73	false	True
True	False	28.12	false	True
True	False	28.02	false	True
True	False	38.34	false	False
True	False	38.51	false	False

Figure 18 : Extrait du dataset en préparation au machine learning

Pour les colonnes numériques trop complexes pour que des données soient générées, nous avons utilisé la normalisation pour limiter l'ensemble des valeurs dans un intervalle entre 0 et 1. Par exemple, la colonne d'altitude contient des valeurs variant de 0m à 800m, il est donc judicieux de normaliser cette colonne pour que les valeurs ne soient comprises qu'entre 0 et 1. La formule mathématique de la normalisation est la suivante :

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

Figure 19 : Formule de normalisation de mise à l'échelle

En appliquant les différentes méthodes de Data Engineering à nos données, nous avons réussi à :

- Préparer des données issues du domaine publique pour les intégrer à un modèle de machine learning
- Simplifier des données pour faciliter leur intégration au modèle

2.5.2. Sessions d'entraînement des modèles

Au sein du laboratoire de machine learning de DataIku nous avons un environnement de sessions d'entraînement des modèles. Ces sessions permettent d'évaluer le poids de chaque feature dans les modèles mais aussi d'avoir une analyse détaillée du modèle ainsi qu'un suivi du score en fonction du temps.

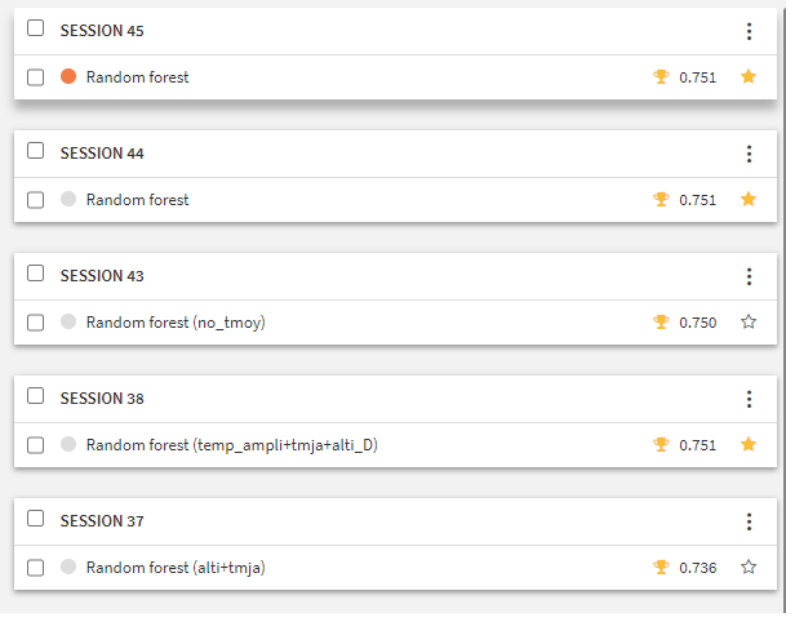

Figure 20 : Plusieurs sessions d'entraînement de modèle

2.5.2.1. Le suivi des scores et résumé de la session

Tout au long d'une session, l'utilisateur peut suivre l'évolution des scores (metrics) du modèle. Parmi ces scores on trouve :

L'AUC : (Area under ROC Curve) Aire en dessous de la fonction d'efficacité du récepteur, plus fréquemment désignée sous le terme « courbe ROC » dite aussi caractéristique de performance (d'un test) ou courbe sensibilité/spécificité, est une mesure de la performance d'un classificateur binaire (notre modèle). C'est-à-dire d'un système qui a pour objectif de catégoriser des éléments en plusieurs groupes distincts (les états de la route) sur la base d'une ou plusieurs des caractéristiques de chacun de ces éléments (la météo, le trafic, l'altitude …)

La précision : En reconnaissance des formes, en recherche d'informations et en classification (apprentissage automatique), la précision (également appelée valeur prédictive positive) est la fraction d'instances pertinentes parmi les instances extraites. La précision est donc basée sur la pertinence.

Au terme de la session, le suivi du score en fonction du temps est disponible ainsi qu'un court résumé du modèle. Ce résumé permet de connaître des informations importantes quant à l'efficacité du modèle et son fonctionnement. Par exemple, nous pourrons observer l'importance de chaque facteur de prédiction pendant l'entraînement. Ces données permettent de voir si l'utilisation des facteurs de prédiction est faite de façon logique. Si jamais le modèle utilise en grande majorité le point de repère routier pour prédire l'état de la route, alors c'est une erreur de configuration car des facteurs comme la météo ou le trafic devraient être utilisés le plus.

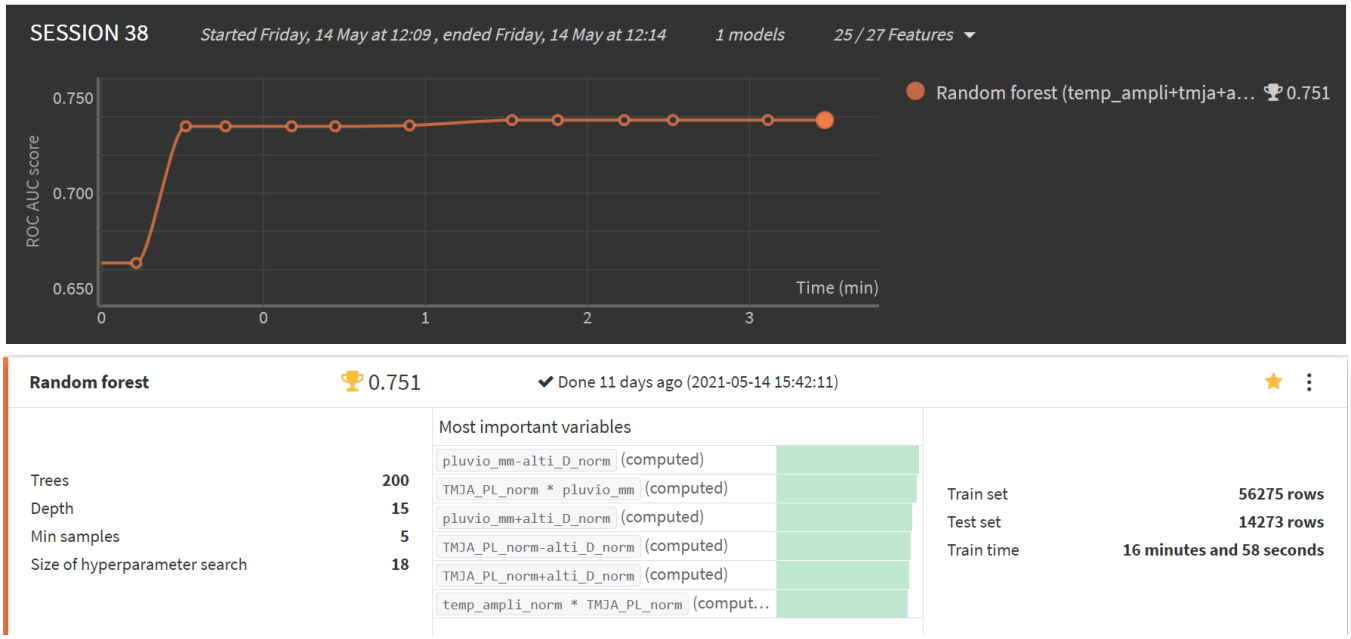

Figure 21 : Suivi du score (AUC) en fonction du temps et résumé d'une session

2.5.3. Les algorithmes de machine learning utilisés

Comme énoncé précédemment, le Lab permet de garder une trace des sessions d'entraînement de modèles de machine learning. Cependant, il a été expliqué qu'un modèle de machine learning est le résultat de données appliquées à un algorithme de machine learning. Il existe plusieurs types d'algorithmes de machine learning en classification et en régression.

La principale différence entre les algorithmes de régression et de classification est que les algorithmes de régression sont utilisés pour prédire les valeurs continues telles que le prix, le salaire, l'âge, etc. et que les algorithmes de classification sont utilisés pour prédire/classer les valeurs discrètes telles qu'Homme ou Femme, Vrai ou Faux, Spam ou Non Spam, etc.

Les parties suivantes ont pour but de vous présenter différents algorithmes de Machine Learning et la façon dont nous avons procédé pour choisir le plus adapté.

2.5.3.1. L'algorithme Random Forest

La forêt aléatoire (Random Forest) est un algorithme d'apprentissage supervisé. La "forêt" qu'il construit est un ensemble d'arbres de décision, généralement formés avec la méthode "bagging" ou « si-alors ». L'idée générale de cette méthode est qu'une combinaison de modèles d'apprentissage augmente le résultat global.

L'un des grands avantages de la forêt aléatoire est qu'elle peut être utilisée pour les problèmes de classification et de régression, qui constituent la majorité des systèmes actuels d'apprentissage automatique. Examinons la forêt aléatoire en classification, puisque la classification est parfois considérée comme la brique de base de l'apprentissage automatique. Vous pouvez voir ci-dessous à quoi ressemblerait une forêt aléatoire avec deux arbres :

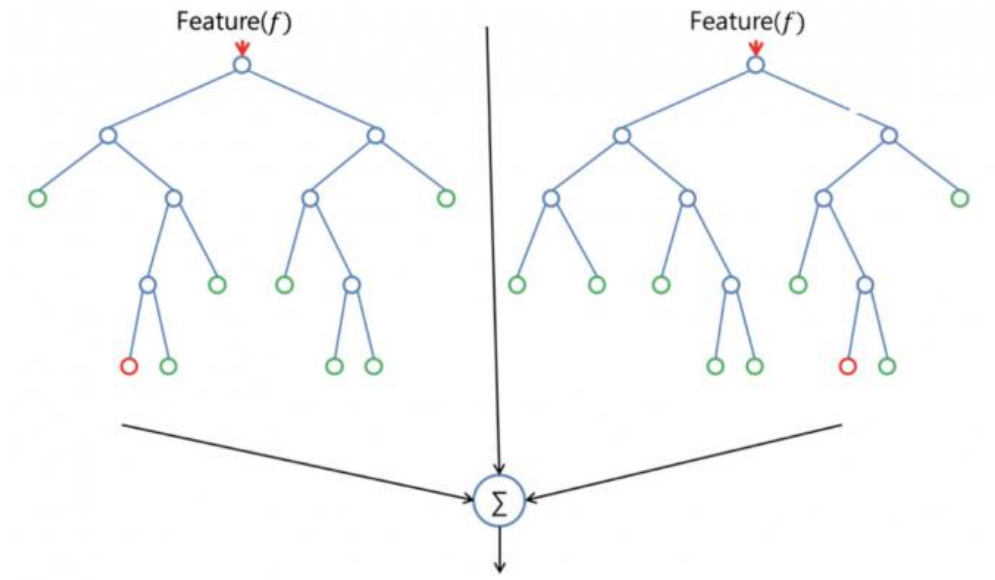

Figure 22 : Schéma d'une forêt aléatoire avec 2 arbres

Voici une analogie avec la vie réelle qui m'a été donnée pour faciliter la compréhension de l'algorithme de Random Forest :

André veut décider où aller pendant ses vacances d'un an, il demande donc des suggestions aux personnes qui le connaissent le mieux. Le premier ami qu'il sollicite lui demande ce qu'il aime et n'aime pas dans ses voyages passés. En fonction des réponses, il donne des conseils à André.

Il s'agit d'une approche typique de l'algorithme de l'arbre de décision. L'ami d'André a créé des règles pour guider sa décision sur ce qu'il devrait recommander, en utilisant les réponses d'André.

Ensuite, André demande à de plus en plus de ses amis de le conseiller et ceux-ci lui posent à nouveau différentes questions dont ils peuvent tirer des recommandations. Enfin, André choisit les endroits qui lui ont été le plus recommandés, ce qui correspond à l'approche typique de l'algorithme de la forêt aléatoire.

Une autre grande qualité de l'algorithme de forêt aléatoire est qu'il est très facile de mesurer l'importance relative de chaque caractéristique sur la prédiction.

Dans l'environnement du Lab de Dataiku il y a un excellent outil pour cela, qui mesure l'importance d'une caractéristique en examinant dans comment les autres caractéristiques sont utilisées pour voir si elles réduisent l'impureté dans tous les arbres de la forêt (la précision de la prédiction). Il calcule ce score automatiquement pour chaque caractéristique après la formation et met les résultats à l'échelle afin que la somme de toutes les importances soit égale à un.

En examinant l'importance des caractéristiques, nous pouvons décider quelles sont les caractéristiques à abandonner parce qu'elles ne contribuent pas suffisamment (ou parfois pas du tout) au processus de prédiction. C'est important car une règle générale de l'apprentissage automatique veut que plus le nombre de caractéristiques est élevé, plus le modèle risque d'être surajusté (over-fitting) et vice versa.

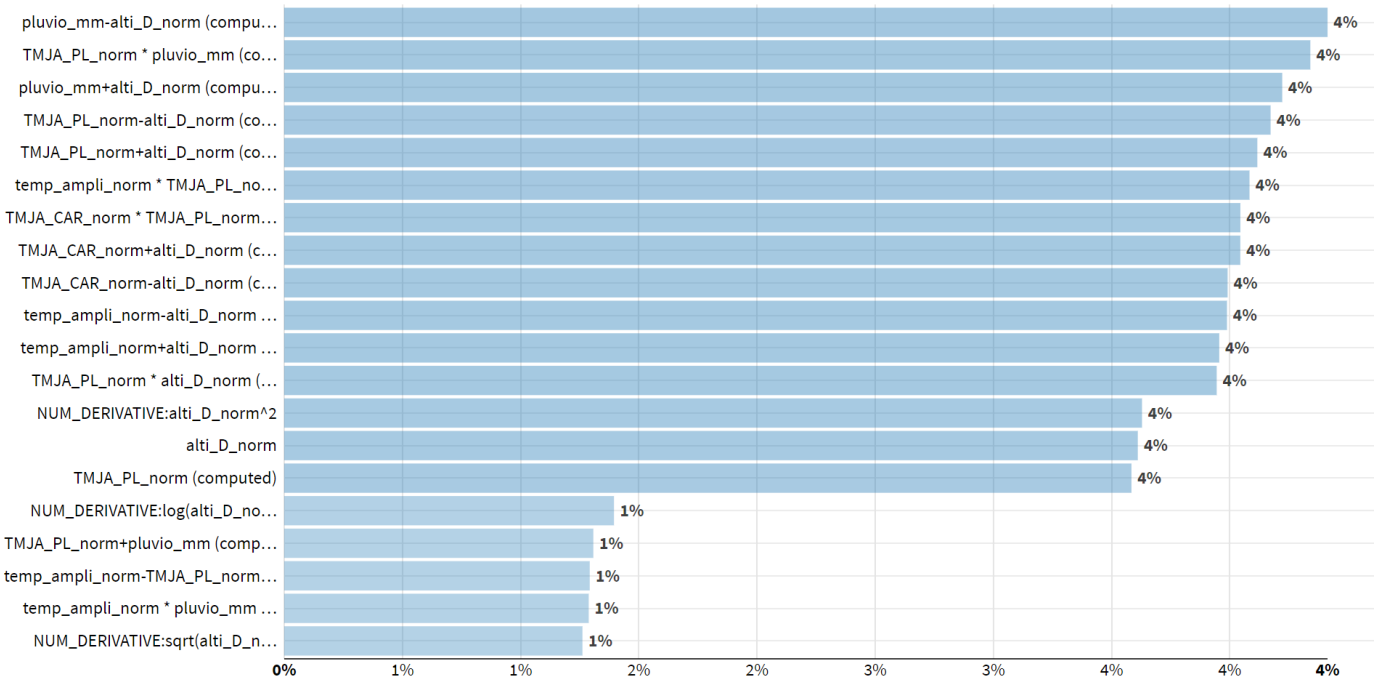

Figure 23 : Graphe représentant l'importance de chaque feature après une session d'entraînement sur une forêt aléatoire

On voit sur le graphe précédent que des combinaisons de colonnes comme pluvio_mm et alti_D_norm (la pluviométrie et l'altitude normalisée) ont une grande importance dans la prédiction. C'est-à-dire que le modèle va majoritairement se baser sur leurs valeurs pour prédire l'état de la route.

Une grande partie de nos analyses ont été réalisées sur des algorithmes de forêt aléatoire en modifiant différents paramètres.

2.5.3.2. L'algorithme SVM (Support Vector Machine)

SVM est utilisé pour classer les entrées dans une des classes prédéfinies (comme Oui / Non ou Bon Etat / Etat Moyen / Mauvais Etat). Si le SVM est utilisé pour classer deux classes comme tête / queue, alors un tel classificateur est un classificateur binaire, sinon il devient un classificateur multi classes. SVM est donc à la fois un classificateur binaire et multi classe.

L'objectif de l'algorithme SVM est de trouver un hyperplan dans un espace à N dimensions (N - le nombre de caractéristiques) qui classe distinctement les points de données.

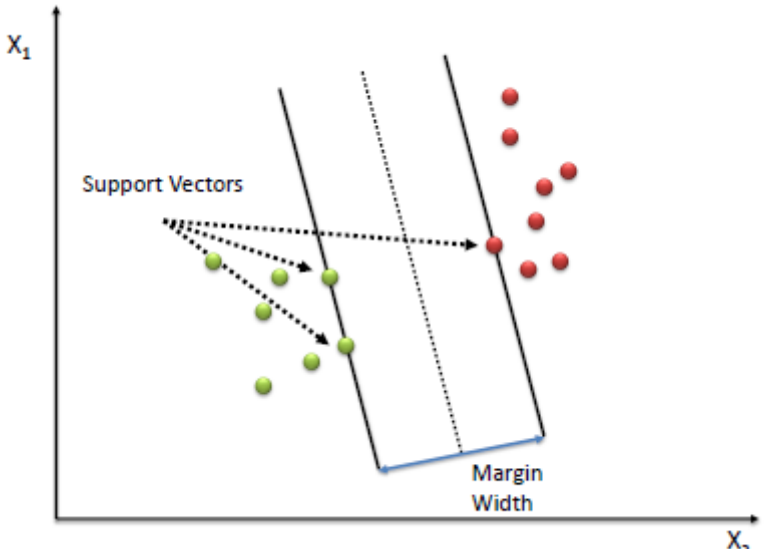

Figure 24 : Graphe représentant la classification SVM

Sur le graphe précédent, la ligne centrale est notre limite de décision (hyperplan). Les points de données qui passent par l'hyperplan pour leurs classes respectives sont les vecteurs de support. Les vecteurs de support sont les points les plus proches de l'hyperplan. Ils sont utilisés pour définir les marges de chaque classe et affectent également la position de l'hyperplan. La distance entre ces hyperplans est la largeur de la marge.

Un hyperplan est un sous-espace dont la dimension est inférieure d'une unité à celle de l'espace environnant. Par exemple, pour un plan 2-D, cet hyperplan serait une ligne. Dans les SVM, cet hyperplan agit comme le meilleur séparateur possible (ou frontière de décision) pour les différentes classes.

L'algorithme SVM présente plusieurs avantages :

- Convient aux ensembles de données comportant plus de caractéristiques que les exemples d'apprentissage.
- Gestion des cas où une ségrégation linéaire n'est pas possible notamment grâce aux noyaux.
- Efficace dans une dimension supérieure (volume de données important).
- Fonctionne bien avec les données non structurées comme le texte et les images (pour nous, des données numériques non structurées).

Cependant certaines limitations font qu'il n'est pas forcément adapté à notre étude :

- Difficile de choisir le noyau approprié (Il s'agit de fonctions mathématiques qui sont utilisées pour traiter l'entrée et la convertir sous une forme qui facilite la classification)
- Une mise à l'échelle des caractéristiques est nécessaire
- Difficulté d'interprétation du modèle final.

De ce fait, les sessions d'entraînement de l'algorithme SVM n'ont pas été très concluante. La précision de la classification ne dépassait pas 43% de bonnes réponses.

2.5.3.3. L'algorithme des KNN (K-nearest neighbors)

L'algorithme des K plus proches voisins ou K-nearest neighbors (KNN) est un algorithme de Machine Learning qui appartient à la classe des algorithmes d'apprentissage supervisé simple et facile à mettre en œuvre qui peut être utilisé pour résoudre les problèmes de classification et de régression.

Voici le processus qu'entreprends l'algorithme des KNN lors d'une classification :

I. Sélection du nombre de k-voisins
II. Calcul de la distance des voisins
III. Stockage des K-voisins les plus proches en fonction de la distance calculée
IV. Parmi les K-voisins, compter le nombre de ces voisins appartenant à chaque catégorie de prédiction
V. Attribuer au nouveau voisin la catégorie la plus présente parmi celles des K-voisins les plus proches
VI. Le modèle est prêt

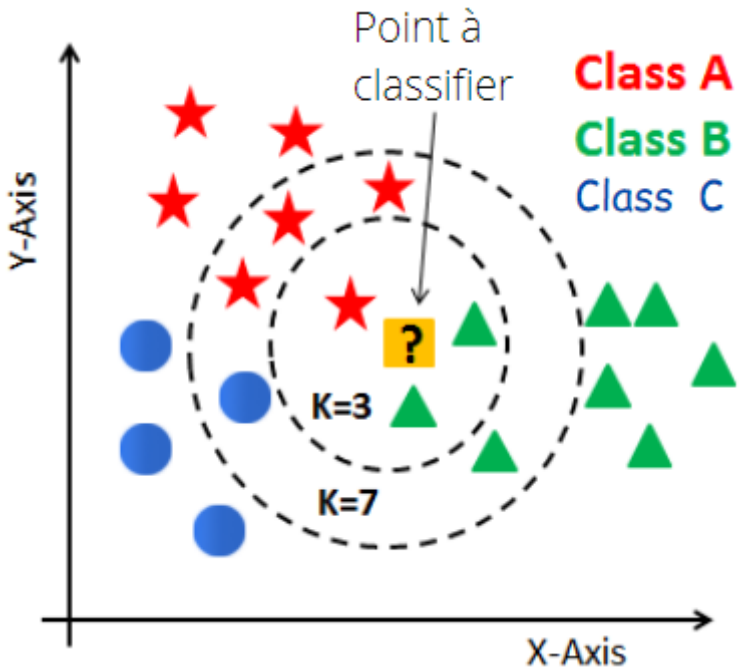

Figure 25 : Principe de l'algorithme des KNN sur 3 classes

Dans l'illustration précédente, nous devons classifier le point jaune en une des 3 classes A, B ou C (pour notre projet -> Bon Etat, Etat Moyen, Mauvais Etat). En suivant le procédé des K-voisins les plus proches, le modèle va classifier le point jaune en Classe B car il y à plus de voisins de la classe B dans le cercle des 3 voisins (K=3).

Cet algorithme présente beaucoup d'avantages comme :

- Une grande simplicité lors de la mise en œuvre.
- Il n'est pas nécessaire de créer un modèle, de régler plusieurs paramètres ou de formuler des hypothèses supplémentaires.
- L'algorithme est polyvalent. Il peut être utilisé pour la classification ou la régression.

Cependant, l'algorithme devient très lent à mesure que le nombre de variables indépendantes augmente. Ayant dans nos jeux de données un nombre important de facteurs de prédiction, l'algorithme n'était pas envisageable. Aussi, la prédiction est très dépendante du nombre de voisins.

Pour conclure, après avoir testé les algorithmes précédemment présentés sur plusieurs sessions, en classification et en régression nous avons décidé de garder l'algorithme de la forêt aléatoire (Random Forest) en classification. En effet, la régression étant plus adaptée à la prédiction de valeur continues comme un score, la mettre en place sur 3 classes n'aurait pas été adapté. Le choix du modèle étant fait, il nous a donc fallu l'optimiser au maximum.

2.5.4. L'optimisation de l'algorithme de Random Forest

Afin d'obtenir le meilleur score possible il était important de modifier certains paramètres de l'algorithme de Random Forest. Il s'agit du process d'« Hyperparameter Tuning » (réglage des hyperparamètres).

La meilleure façon de comprendre les hyperparamètres est de les comparer aux paramètres d'un algorithme qui peuvent être ajustés pour optimiser les performances, tout comme on peut tourner les boutons d'une radio FM pour obtenir un signal clair. Alors que les paramètres du modèle sont appris pendant les sessions, les hyperparamètres doivent être définis par le programmeur avant la session d'entraînement. Dans le cas d'une forêt aléatoire, les hyperparamètres comprennent le nombre d'arbres de décision dans la forêt et le nombre de caractéristiques prises en compte par chaque arbre lors de la division d'un nœud.

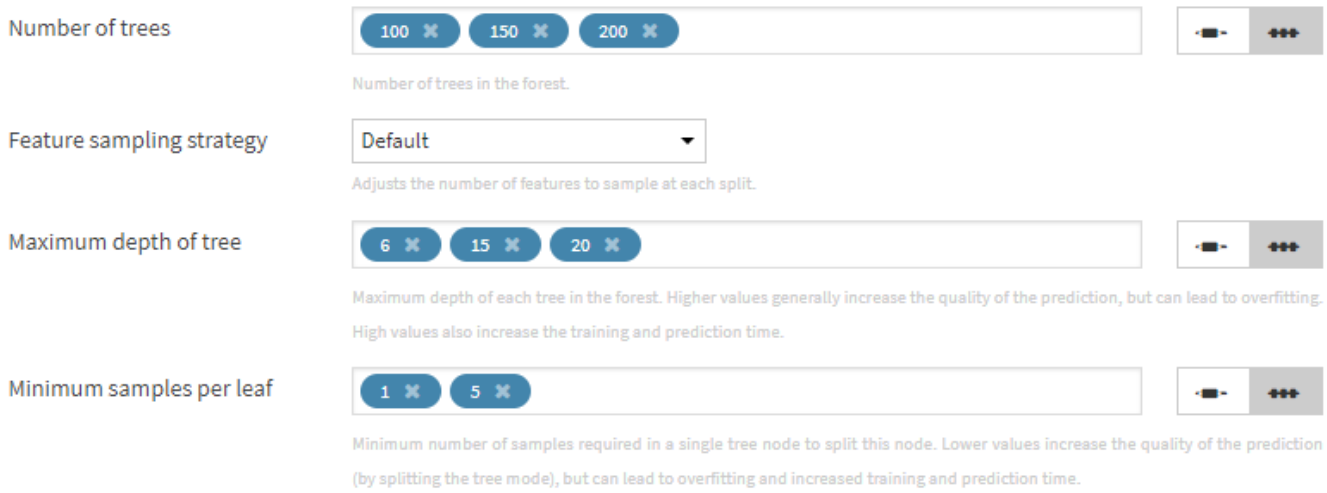

Figure 26 : Panneau de réglage des hyperparamètres de l'algorithme de Random Forest dans DataIku.

Dans l'illustration précédente, on peut voir les différentes valeurs données aux hyperparamètres (Number of tree, Maximum Depth of Tree, Minimum samples per leaf). Afin de trouver les paramètres les plus optimisés, il faut lancer un grand nombre de sessions et garder la trace des scores pour des hyperparamètres donnés. Certains paramètres doivent être manipulés avec précaution car le temps d'entraînement peut être considérablement augmenté et n'assure pas forcément une bonne précision.

Après avoir trouvé les paramètres les plus optimisés nous pouvions donc nous pencher sur l'utilisation des features et leur importance lors de l'apprentissage du modèle.

2.5.4. Analyse de l'importance des facteurs de prédiction

Le but de cette étape est de révéler les features/facteurs de prédictions qui ne sont pas utiles ou qui n'ont pas d'apport particulier dans l'amélioration du modèle.

Pour cela nous avons réalisé plusieurs analyses utilisant le même algorithme de Random Forest en classification. **Note : la proportion de base des classes est (BE, EM, ME) -> (41%, 31%, 28%). Les classes équilibrées ont comme proportion (BE, EM, ME) -> (33%, 33%, 33%).**

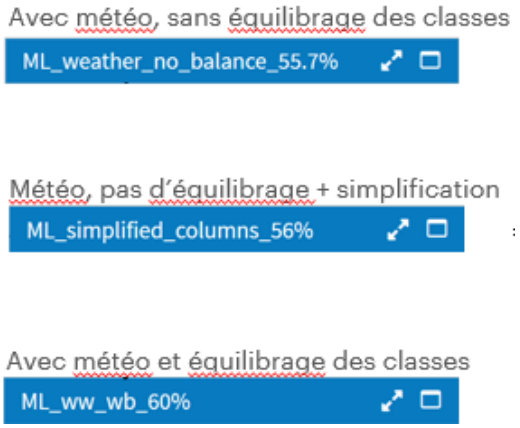

Figure 27 : 3 analyses de machine learning et les scores associés.

I. La première analyse comprend un dataset d'entraînement contenant des données météo et les données d'enrichissement dont les classes ne sont pas équilibrées. **Le score final est de 55.7% de bonnes réponses sur les données de test.**

II. La deuxième analyse comprend un dataset d'entraînement contenant les données météo et les données d'enrichissement dont les classes ne sont pas équilibrées mais les colonnes numériques sont simplifiées suivant le process décrit en partie « 2.5.1.1. Extraction de données ». **Le score final est de 56% de bonnes réponses sur les données de test.**

III. La troisième analyse comprend un dataset d'entraînement contenant des données météo et les données d'enrichissement dont les classes sont équilibrées. **Le score final est de 60% de bonnes réponses sur les données de test.**

On remarque une amélioration du score quand les classes sont équilibrées cependant ce n'est pas une condition que l'on peut se permettre de vérifier (Overfitting/surajustement). Quand plus de données seront apportées au modèles, les proportions des classes devront rester telles quelles pour que la classification ne soit pas biaisée.

On remarque aussi que lorsque les données sont passées par le processus de simplification, le score est légèrement amélioré. Etat donné que ce processus augmente le nombre de facteurs de prédiction il est important de garder une trace des features utilisés pendant chaque session d'entrainement afin de révéler le meilleur modèle.

Nous avons donc exécuté plus d'une quarantaine de sessions et référencé les features utilisés ainsi qu'une description de la session et le score final dans un tableau excel afin de garder une trace du travail s'il devait être repris plus tard.

	summary	est_autoroute	est_nationale	alti_D_norm	high_alti	low_alti	en_zone_montagneuse
	all param set -> AUC = 0.747	1	1	1	1	1	1
	only numerical -> AUC = 0.740			1			
	only binary columns -> AUC = 0.63	1	1		1	1	1
	with alti_D_norm -> AUC = 0.682	1	1	1	1	1	1
	with pluvio_mm -> AUC = 0.679	1	1		1	1	1
	no_road_boolean -> AUC = 0.728			1	1	1	1
	alti+tmjas -> AUC = 0.736	1	1	1	1	1	1
	alti+tem_ampli+tmja -> AUC = 0.7	1	1	1	1	1	1
	no_tmja_bool -> AUC = 0.743	1	1	1	1	1	1
Top score	no_tmoy -> AUC = 0.751	1	1	1	1	1	1

Figure 28 : Extrait du tableau analytique de référencement des sessions, des scores et des features utilisés.

Dans ce tableau les colonnes utilisées sont marquées par un « 1 », celles qui ne le sont pas ne sont pas marquées. Après plusieurs sessions aux scores différents en fonction des colonnes utilisées ou non, on obtient le meilleur score d'une **AUC de 0.751.** Ce score est obtenu lorsque la colonne **temperature_moyenne n'est pas utilisée.** Ceci montre aussi que tous les autres facteurs de prédiction sont importants.

Au terme de cette analyse nous avons donc étudié en détail plusieurs algorithmes de machine learning, en suivant la méthode de développement d'un modèle de machine learning. Nous avons aussi exploré des méthodes de data engineering afin de produire des scripts en Python en utilisant des librairies spécifiques nous permettant d'extraire des facteurs de prédiction de nos données.

Après avoir conduit plusieurs analyses et tests nous permettant de révéler l'importance de chaque feature et la pertinence de plusieurs modèles reposant sur la présence ou non de données, l'équilibrage ou non des classes, nous avons retenu le modèle suivant basé sur l'algorithme de la Forêt Aléatoire (RF).

Figure 29 : Résumé du modèle prédictif retenu au terme des analyses (ACC = précision)

Pour résumer les méthodes utilisées jusque-là, voici un schéma reprenant les étapes principales du développement du modèle :

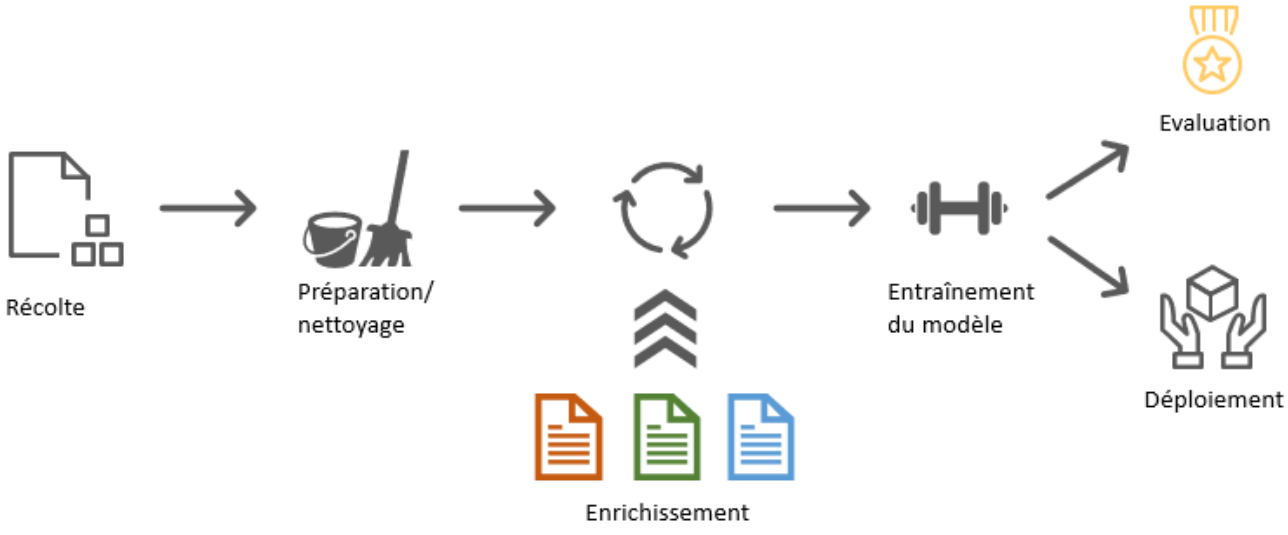

Figure 30 : Schéma explicatif de la méthode de développement de modèle de machine learning utilisée

2.6. Utilisation du modèle et ouverture

Après avoir développé le modèle et l'avoir déployé, nous nous sommes posé la question : « **Comment ce modèle allait-il être utilisé concrètement ? »**

Si ce modèle est amené à être utilisé dans une application il lui suffirait de lui donner en entrée le nom d'une route (ex : A80). Le modèle analyserait ce nom et se baserait sur son entraînement en prenant en compte les données d'enrichissement (météo, trafic, altitude etc.) et nous donnerait alors sa prédiction vraie à 56% : Bon Etat, Etat Moyen ou Mauvais Etat.

Voici un schéma représentant cette utilisation :

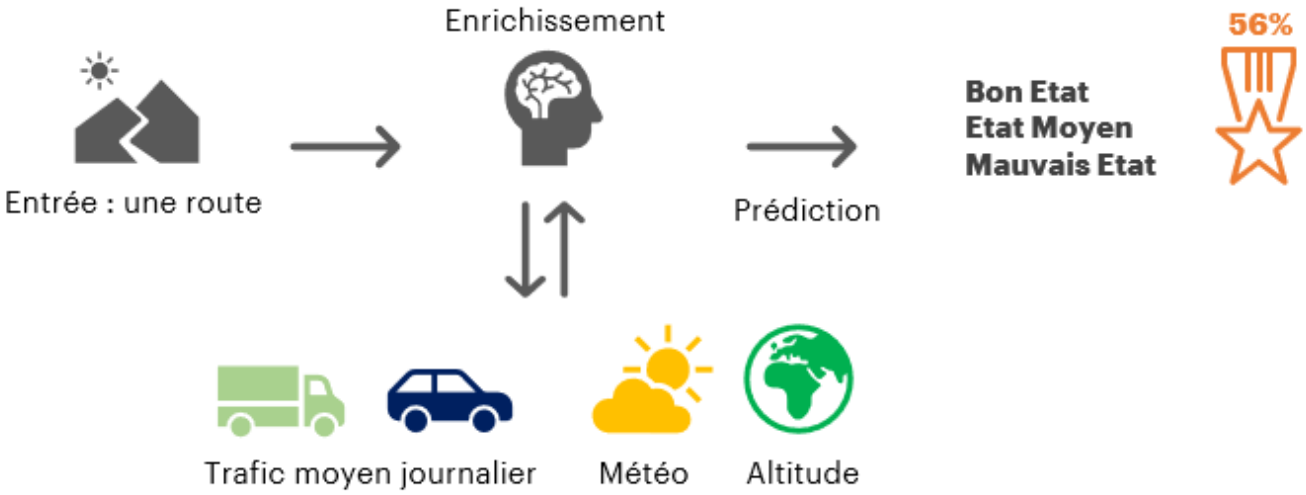

Figure 31 : Schéma explicatif de l'utilisation du modèle

Cependant de nombreuses questions se posent :

- **Pourquoi le pourcentage de bonnes réponses est-il si bas ?**
- **Est-il possible de l'améliorer ?**
- **Si c'est le cas, avec quel type de données ?**

Ces questions sont justifiées. En effet, ce modèle n'est pas encore très fiable, 56% de bonnes réponses laisse encore 44% de mauvais réponses et c'est beaucoup trop pour pouvoir proposer ce modèle dans un projet.

C'est pourquoi nous avons pensé à étudier des données en lien direct avec l'état de la route : **les vibrations.**

3. Analyse des données de vibration des routes

Après avoir fini le développement du modèle de machine learning, ce qui a occupé la moitié de la période de stage, nous avons décidé de commencer l'analyse de données de vibration récoltées lors de trajets par Valentin Villedieu, Alban Deumier et moi-même. **Toutefois, cette analyse étant très chronophage et incomplète, elle ne sera que partiellement illustrée dans ce rapport. C'est aussi une analyse très complexe et le temps ne permet pas une vulgarisation ce travail.**

Pour vous présenter rapidement cette analyse il est important de répondre aux questions suivantes :

- **Quels types de donnée** vont être récoltés ?
  - Nous allons récolter des données de **vibration** (accélération sur l'axe des Z, mouvement haut/bas de la voiture), les données de **mouvement** sur les axes X et Y (mouvement avant arrière et latéral du véhicule) ainsi que des données **GPS** pour **associer une vibration à un couple de coordonnées (donc à une route)**.
- **Comment** seront récoltées les données ?
  - Les données seront récoltées pendant des trajets sur autoroute ou route nationale **via une application** développée par Alban Deumier (Roads Reader).
  - Les données seront ensuite **centralisées dans l'outil Dataiku**.
  - Enfin nous procéderons à une **analyse numérique et graphique** de ces données pour en extraire des facteurs de prédiction
  - Puis nous intégrerons les données au modèle afin qu'il soit amélioré
- **Pourquoi** récolter ces données ?
  - Principalement pour **tenter d'améliorer le modèle** en apportant des données en lien direct avec l'état de la route.
  - Cette analyse représente **une approche concrète de l'utilisation du modèle sur système embarqué.**

Voici une illustration de ce procédé :

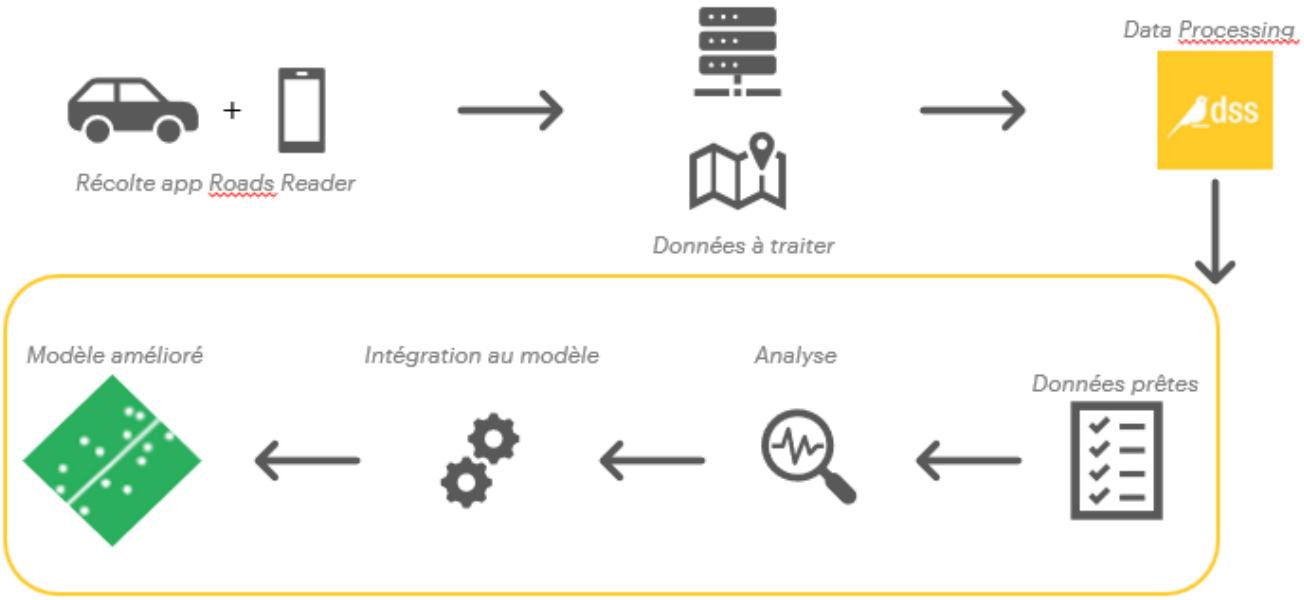

Figure 32 : Schéma explicatif du procédé d'analyse des vibrations

3.1. L'application Roads Reader

L'application utilisée est intitulée « Roads Reader ». Elle à été entièrement développée par Alban Deumier avec Unity. Cette application retourne un fichier csv contenant les données citées précédemment. Une fois installée sur un téléphone il suffit de le placer et de le fixer sur une surface dure et horizontale le smartphone écran vers le haut, le haut de l'écran coté avant de la voiture. Le smartphone doit rester actif pendant toute la captation (aucune mise en veille). Bien activer la localisation GPS sur son mobile (autoriser l'application).

3.1.1. Ecran d'accueil

Sur l'écran d'accueil il est demandé de renseigner au mieux le modèle de voiture utilisé pour l'échantillonnage car les vibrations varient en fonction du type de voiture.

Ensuite l'utilisateur peut nommer son fichier csv ici « TestProject ».

Enfin on peut initialiser le timestamp UNIX pour la fréquence d'échantillonnage.

Après avoir renseigné tous les paramètres, l'utilisateur peut initialiser l'échantillonnage en appuyant sur le bouton « Initialization »

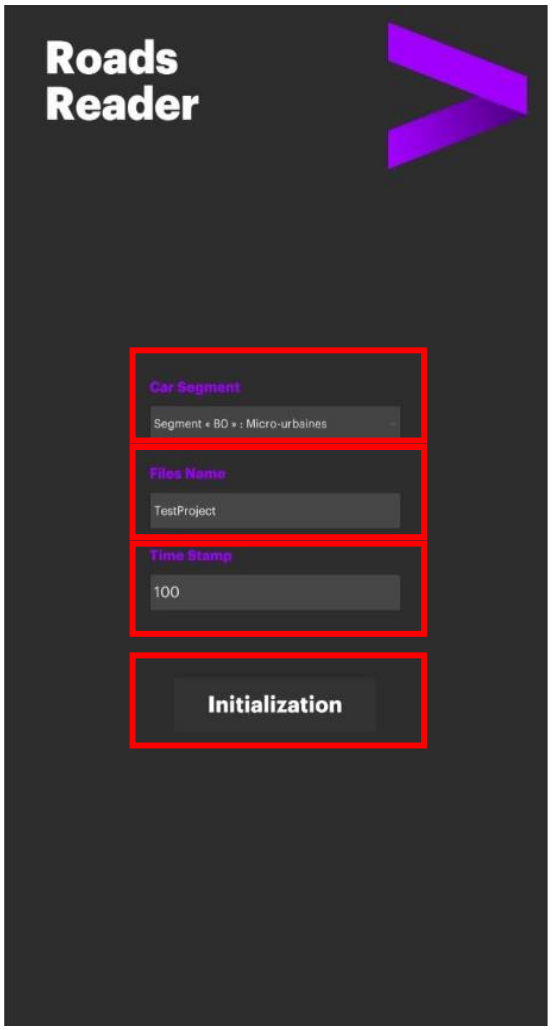

Figure 33 : Capture d'écran de la page d'accueil de Roads Reader

3.1.2. L'échantillonnage et la sauvegarde du fichier csv

Une fois que l'utilisateur est sur cette page, il doit s'assurer que sont téléphone est bien fixé et il peut appuyer sur le bouton « Start ». A ce moment, les données commencent à être enregistrées.

Figure 34 : Capture d'écran de la page d'échantillonnage de Roads Reader

Une fois le trajet terminé, l'utilisateur peut appuyer sur le bouton « Stop and Save ». Le fichier csv n'a plus qu'a être récupéré et envoyé sur DataIku pour passer à l'étape de Data Processing.

3.2. Traitement des données issues de l'application

Une fois les données de l'application importées dans DataIku nous allons les manipuler et les préparer pour extraire de nouvelles données.

Par exemple, nous avons extrait de la colonne « Speed » issue de l'application, les colonnes « Speed Km/H » et « Speed_level ». La colonne « Speed_level » à pour but de rassembler les vitesses en paliers :

- <85 km/h -> palier 1
- [85 ; 95] km/h -> palier 2
- [105 ; 115] -> palier 3
- >120 km/h -> palier 4

L'interêt de séparer les vitesses en palier est de réduire le « bruit ». C'est-à-dire, toutes les données qui n'ont pas d'importance : en dessous d'un certain seuil de vitesse, les vibrations sont négligeables.

Un autre point important à été de savoir si nous étions effectivement en train d'échantillonner une route analysée par Le Ministère de la Transition Ecologique dans les jeux de données de départ. Il fallait donc, pour chaque couple de coordonnées issus de l'application, trouver la route nationale / autoroute la plus proche pour valider l'échantillon.

Pour ce faire, nous avons implémenté une fonction en python qui, pour chaque couple de coordonnées d'un fichier issu de l'application, trouve le couple le plus proche dans le jeu de donnés de base. Etant donné que chaque jeu de donnés compte environ 100 000 lignes, il fallait utiliser un procédé de Vectorisation (expliqué dans la partie de Traitement des données) ainsi qu'une structure de données adaptée à cette recherche : un KDTree.

Un KDTree fonctionne de la même façon que l'algorithme des « K plus proches voisins » (K-nearest neighbors). De ce fait nous avons pu extraire d'un couple de coordonnées, toutes les informations sur la route la plus proche ainsi que la distance séparant les deux couples nous permettant de vérifier qu'il s'agit de la même route.

Voici un schéma du procédé :

Figure 35 : Procédé de préparation des données GPS

3.3. Prochaines étapes pour la fin de stage

Nous ne pourrons pas détailler cette analyse plus loin car elle est prévue pour la fin de stage. Cependant nous nous sommes fixés des étapes pour couvrir la dernière période :

- **Récolter des données** avec l'application Roads Reader
- **Agréger les données**, **étendre** l'analyse **à un volume de données plus important**
- **Entraîner le modèle** avec les nouvelles données / uniquement les données de vibrations

La fin du stage consistera donc à la documentation de tous ces procédés et à l'intégration complète des données de vibrations dans le modèle développé lors de la première partie.

4. Conclusion

J'ai eu la chance de faire mon stage de fin d'études dans une entreprise ayant une politique d'accueil des stagiaires permettant une réelle responsabilisation. Les missions confiées aux stagiaires sont similaires aux missions que remplissent les salariés, tout en gardant la mesure de la différence d'expérience et de compétences et en tenant compte du facteur temps.

Cette responsabilisation m'a beaucoup plu, et j'ai réellement apprécié travailler au sein d'un projet concret, dans un domaine de l'informatique en constante évolution. Je pense avoir réussi à développer un modèle de machine learning répondant aux attentes de mon tuteur de stage et je tire une certaine fierté à l'idée que mon travail sera utilisé pour d'autres projets après mon départ.

Cela dit, ce qui a fait de ce stage une réussite à mes yeux n'est pas seulement le travail accompli, mais tous les apprentissages que j'ai pu en retirer. J'ai notamment beaucoup appris sur le domaine du machine learning, les algorithmes utilisés ainsi qu'en méthodologie de travaux de « recherche » et je pense que ce sera un bénéfice considérable à l'avenir.

Je conclurais en ajoutant que j'ai beaucoup apprécié travailler chez Accenture Nantes grâce à l'ambiance chaleureuse que l'on peut y retrouver. J'ai pleinement conscience de ma chance d'avoir effectué un stage de si bonne qualité dans un environnement si agréable, et je remercie une dernière fois toutes les personnes qui ont rendu cela possible.

5. Table des figures

Figure 1 : Logo d'Accenture 3
Figure 2 : Accenture en bref 3
Figure 3 : Façade du Nantes Liberty Center 4
Figure 4 : Intérieur du Liquid Studio 5
Figure 5 : Organisation en plateformes technologiques 5
Figure 6 : Logo de DataIku 8
Figure 7 : Exemple de workflow dans DataIku 9
Figure 8 : Diagramme de développement d'un modèle de Machine Learning 13
Figure 9 : Diagramme de la conversion des systèmes de coordonnées 14
Figure 10 : Exemple de Vectorisation 15
Figure 11 : Jointure selon une clé composée dans DataIku 15
Figure 12 : Visualisation de la validité des colonnes dans DataIku 16
Figure 13 : Recette de préparation d'un jeu de donnée sur DataIku 16
Figure 14 : Fonctionnement de l'API IGN. 17
Figure 15 : Diagramme du processus d'enrichissement des données 18
Figure 16 : Exemple de dégradation des routes (gel et dégel à gauche, végétation à droite) 19
Figure 17 : Fonction python utilisée pour la génération de colonnes booléennes 21
Figure 18 : Extrait du dataset en préparation au machine learning 21
Figure 19 : Formule de normalisation de mise à l'échelle 22
Figure 20 : Plusieurs sessions d'entraînement de modèle 22
Figure 21 : Suivi du score (AUC) en fonction du temps et résumé d'une session 23
Figure 22 : Schéma d'une forêt aléatoire avec 2 arbres 24
Figure 23 : Graphe représentant l'importance de chaque feature après une session d'entraînement sur une forêt aléatoire 25
Figure 24 : Graphe représentant la classification SVM 26
Figure 25 : Principe de l'algorithme des KNN sur 3 classes 27
Figure 26 : Panneau de réglage des hyperparamètres de l'algorithme de Random Forest dans DataIku. 28
Figure 27 : 3 analyses de machine learning et les scores associés 29
Figure 28 : Extrait du tableau analytique de référencement des sessions, des scores et des features utilisés 30
Figure 29 : Résumé du modèle prédictif retenu au terme des analyses (ACC = précision) 31
Figure 30 : Schéma explicatif de la méthode de développement de modèle de machine learning utilisée 31
Figure 31 : Schéma explicatif de l'utilisation du modèle 32
Figure 32 : Schéma explicatif du procédé d'analyse des vibrations 33
Figure 33 : Capture d'écran de la page d'accueil de Roads Reader 34
Figure 34 : Capture d'écran de la page d'échantillonnage de Roads Reader 35
Figure 35 : Procédé de préparation des données GPS 36

6. Bibliographie

Les articles suivants ont été utilisés pour mettre en place notre stratégie d'étude des données de vibration. Les deux articles utilisent ce procédé d'analyse :

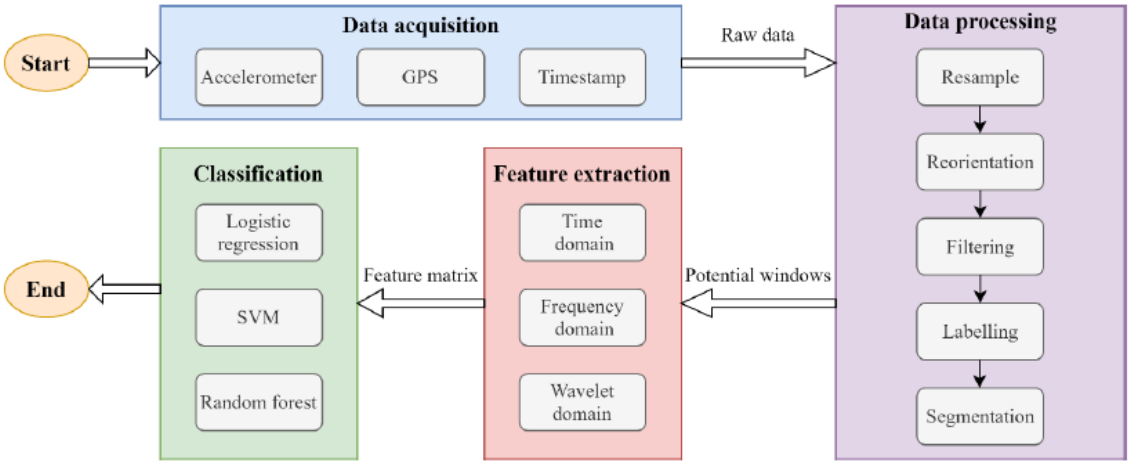

Figure 2. Workflow of the proposed methodology.

« A Machine Learning Approach to Road Surface Anomaly Assesment Using Smartphone Sensors » ***Akanksh Basavaraju, Jing Du, Fujie Zhou, and Jim Ji. University of Texas, US.***

« An Automated Machine-Learning Approach for Road Pothole Detection Using Smartphone Sensor Data» ***Chao Wu, Zhen Wang , Simon Hu , Julien Lepine , Xiaoxiang Na, Daniel Ainalis and Marc Stettler. Zhejiang University, Hangzhou, Université Laval, Quebec City and University of Cambridge***

:

Résumé

Les deux années passes dans la formation DUT doivent se terminer par un stage en entreprise de 11 semaines. Ce stage à pour but d'allier la pratique et la théorie vue pendant le parcours au sein du département informatique de l'IUT de Vannes. J'ai eu la chance de réaliser ce stage dans le centre du Nantes Advanced Technology Center chez Accenture Technology, le leader mondial dans la catégorie des SSII.

Mon rôle à été de concrétiser un projet intitulé « Développement d'un modèle prédictif de dégradation des axes routiers nationaux » à partir d'une idée venant de M. Alban DEUMIER. Ce projet consiste donc à suivre le processus de développement d'un modèle de machine learning capable de classifier l'état d'une route. J'ai eu la charge de développer entièrement le modèle.

Ce stage m'a permis de consolider mes connaissances en machine learning ainsi que de m'améliorer dans la manipulation des librairies scientifique du langage de programmation Python comme SciPy, Pandas et Numpy.

Summary

The two years spent in the Universitary Technological Institute must end with an 11 weeks internship in a company. This internship aims to combine the practical and the theoretical knowledge seen during the course within the computer science department of the IUT of Vannes. I had the chance to realize this internship in the Nantes Advanced Technology Center at Accenture Technology, the world leader in the category of IT services.

My role was to concretize a project entitled "Development of a predictive model to assess the state of national roads" from an idea coming from Mr. Alban DEUMIER. This project consists in following the development process of a machine learning model able to classify the state of a road. I oversaw the development of the whole model.

This internship allowed me to consolidate my knowledge in machine learning and in the manipulation of scientific libraries of the Python programming language such as SciPy, Pandas and Numpy.