ECOLE NATIONALE DE L'AVIATION CIVILE
&
ALTRAN

Année : 2020/2021

Rapport du projet IA

Présenté et soutenu par :
Théo Bouteilles, IENAC18
Denis Béréziat, IENAC17
Arnaud Lusson, IENAC18

à l'Ecole Nationale de l'Aviation Civile
4 Février 2021

Green Tech : Application de méthodes d'apprentissage artificiel à l'estimation de l'empreinte écologique d'activités

Tuteur de projet :
Thierry Dolmière Team Manager Digital chez Altran

TABLE DES MATIÈRES

I. INTRODUCTION

Aujourd'hui, l'écologie joue un rôle important dans l'économie et avec la crise sanitaire de la Covid-19, les déplacements ont dû être repensés et le télétravail s'est accru dans les entreprises.

Les entreprises ont dû revoir les emplois du temps de leurs salariés pour mettre en place des jours de télétravail, pour satisfaire aux conditions du gouvernement. Elle avait donc un nouvel enjeu économique en choisissant les jours de télétravail pour limiter l'empreinte écologique de chacun de leur employé. Pour cela, il y a plein de variables à prendre en compte comme le trajet pour venir sur le lieu du travail, la consommation des serveurs et la box internet de la maison par exemple. Le tout en essayant de prendre en compte l'évolution et l'impact des nouvelles technologies comme le déploiement de la 5G (début des essais 6G en Chine). À la vue de la complexité pour traiter la première variable qui est l'empreinte écologique du trajet habitation-travail et après discussion avec notre tuteur de projet, le sujet s'est orienté vers la problématique d'estimation de l'empreinte écologique d'activités en créant un algorithme à l'aide de méthodes d'apprentissage pouvant s'adapter aisément à différentes sources de consommation, que nous testerons en estimant la pollution carbone de la voiture au quotidien. La portabilité du projet est importante, travaillant avec Altran, ils veulent pouvoir réutiliser notre travail sur différents projets comme le projet Kaggle de réalisation d'un bateau électrique connecté avec estimation et minimisation de la consommation électrique tout au long d'une course.

L'écologie et le télétravail sont les principaux enjeux de notre sujet et ont été au cœur de nos problématiques tout au long du développement du projet, ces deux enjeux contemporains nous ont intéressés et surtout le télétravail pour connaître quels étaient ses effets sur la pollution globale mondiale. Même si le sujet s'est finalement orienté sur l'étude de la consommation des voitures, le télétravail est resté très influent en 2020 sur ce point, en modifiant par exemple la circulation et les embouteillages sur la route.

II. PRÉSENTATION DU SUJET

A. Contexte

Au sein d'Altran, Thierry Dolmière, notre tuteur de stage est Team Manager Digital c'est-à-dire qu'il est le responsable de la planification, du développement, de la mise en œuvre et de la gestion de stratégie d'équipes sur différents projets. Un des projets était l'estimation de la consommation énergétique d'un développeur/d'une équipe de développeurs, alternant entre journées en présentiel et télétravail. Ce premier projet prenait en compte tous les déplacements de l'employé, la consommation de l'ordinateur en fonction de l'utilisation et l'énergie utilisée par les serveurs par exemple. Dans un premier temps nous avons effectué un état de l'art de l'empreinte écologique des différents systèmes composants l'environnement de travail d'un développeur et avons effectué des recherches en quête de données exploitables pour pouvoir réaliser des modèles sur le sujet. Cependant nous avons été freinés par la complexité du sujet qui rendait difficile la mise en place d'un plan d'action et l'exploitation des connaissances acquises tout au long de l'année en intelligence artificielle. Nous avons donc commencé par aborder ce sujet en nous basant sur les déplacements habitat-travail et la variation de la consommation en fonction de l'heure de départ, de la distance et du mode de transport. Puis, M. Dolmière s'est vu assigné un nouveau projet sur le développement d'un bateau électrique autonome pour une course. Notre précédent travail sur la consommation des voitures a donc été développé au maximum pour avoir un algorithme adaptatif pour l'estimation de la consommation d'énergie.

B. Motivation

Dans le cadre de notre troisième année d'école d'ingénieur à l'ENAC en option Systèmes Informatiques du Transport Aérien et Intelligence Artificielle, il nous est proposé un projet sur 4 mois nous permettant de mettre en pratique nos connaissances et nos compétences professionnelles au travers d'un sujet basé sur l'élaboration et/ou l'utilisation des méthodes d'Intelligence Artificielle. Cette utilisation des méthodes d'apprentissage pour estimer l'empreinte écologique d'activités a été une formidable opportunité pour appliquer nos cours et un enjeu actuel : l'écologie.

C. Enjeux

Les enjeux du projet sont multiples : techniques, économiques et humains.

En premier lieu, il y a un défi technique lié directement au développement d'une intelligence artificielle pour estimer l'empreinte écologique d'une activité. Le développement comprend aussi bien l'algorithme et son codage que la documentation de celui-ci et sa portabilité pour d'autres activités. Cette dernière est un enjeu crucial mais difficile pour la réussite du projet.

L'enjeu est plus économique du côté d'Altran, cette estimation de consommation permettra plus tard d'optimiser l'utilisation d'énergie du bateau autonome pour la course. Le

principe de cette course est de parcourir une distance avec une consommation d'énergie limitée.
Outre la technique et l'aspect économique, il y a un aspect développement professionnel. En travaillant en collaboration avec Altran, nous appréhendons au mieux le monde et le travail en entreprises : ce que peut être l'attente des clients, la stratégie d'équipe et la mise en place concrète de méthodes comme la méthode AGILE.

D. Objectifs et contraintes

Le tout premier objectif, avant même les objectifs techniques, a été la définition des limites du sujet, de la problématique et la spécification des objectifs. Au départ, le thème du projet était vaste et le temps limité (seulement 4 mois), nous sommes donc partis sur la problématique d'estimation de la consommation d'une équipe de développement logiciel alternant télétravail et présentiel. Puis à la vue de la complexité et du nombre de facteurs influant sur la consommation du trajet habitation-travail, nous avons décidé avec notre tuteur de redéfinir la problématique plusieurs fois pour arriver à : Application de méthodes d'apprentissage artificiel à l'estimation de l'empreinte écologique d'activités, pour ensuite définir les objectifs techniques.

Pour les objectifs techniques :

- Programmer un algorithme qui, à l'aide de l'intelligence artificielle, estime la consommation d'une voiture en fonction de plusieurs paramètres (distance, heure de départ, modèle de la voiture, …)
- Produire un algorithme adaptatif, c'est-à-dire capable de fonctionner avec la même base sur d'autres calculs d'estimation ou capable de prendre en compte la découverte d'une nouvelle technologie (nouveau carburant par exemple).
- Grâce au programme, proposer des pistes ou un début de solution pour comprendre et améliorer l'empreinte carbone du système étudié.
- Écrire une documentation sur notre algorithme afin de comprendre ce qui a motivé le choix de nos différents paramètres et apporter un guide de mise en place pour des sujets similaires.

Nous avons rencontré plusieurs contraintes sur ce projet. La première est liée à une difficulté à lire voire une absence d'accès aux données de trafic routier. Le trafic est souvent en temps réel sur les sites sans sauvegarde quotidienne des données. Les sauvegardes précédentes nous informe juste sur le fait que le trafic d'une journée a été particulièrement fluide ou congestionné mais sans donner de détail sur chaque heure de la journée. Enfin, certaines données étaient payantes ou alors nous n'avions pas les droits d'accès. Une des contraintes qui a été mentionnée plus haut a aussi été le temps. 4 mois pour un sujet aussi vaste où il a fallu que nous définissions les attendus sur ce que nous étions capable de faire, nous avons donc dû gérer efficacement notre temps. La dernière contrainte est l'alternance entre distanciel et présentiel, travailler en équipe dans la même salle, c'est plus rapide que de travailler en télétravail.

III. MANAGEMENT DE PROJET

A. Choix de la méthode de travail

En accord avec notre tuteur de stage, nous avons choisi la méthode de travail AGILE et la méthodologie Scrum. Tout au long de notre projet, nous avions des réunions hebdomadaires avec notre tuteur de stage afin d'identifier et de comprendre les attentes d'Altran. Nous avons aussi utilisé la plateforme Trello pour tenir à jour nos tâches, sauvegarder nos comptes rendus de réunion et définir nos étapes (qu'on retrouve dans le diagramme de Gantt).

Concernant la partie Agile, notre tuteur représente le product owner. C'est lui qui nous précise les spécificités du produit final, ses attentes et les priorités. Nous avons ensuite essayé de suivre au mieux la méthodologie Scrum en établissant une première série de sprints pour découper le projet :

- Recherches et état de l'art.
- Récolte des données et création d'un plan d'action.
- Modélisation et test.
- Retouches et rédaction du rapport et de la soutenance.

Nous avons ensuite essayé d'organiser un maximum de réunion avec l'équipe pour se répartir les tâches et voir l'avancement de chacun. Ces réunions correspondent finalement aux séances de projet IA. Nous avons également organisé de nombreuses réunions avec le product owner, à savoir une par semaine ou une toutes les deux semaines pour faire une rétrospective du travail fourni et recadrer le contexte des sprints suivants.

<u>Avantages :</u>

L'un des principaux avantages a été d'avoir eu des retours réguliers de notre tuteur. Nous avons eu du mal à cerner les objectifs au début du projet et cette méthode a permis de faire vivre le contexte du projet, de le faire évoluer. Cela nous a également incité à mettre en place des deadlines et à essayer de les respecter.

<u>Inconvénients et difficultés :</u>

Nous avons cependant rencontré plusieurs soucis. Dans un premier temps nous avons eu des difficultés à maintenir des réunions régulières entre nous compte tenu de la charge de travail importante sur cette période pour d'autres projets ou examens. Malgré des points réguliers avec notre tuteur nous avons commencé à travailler sur un use case technique qu'assez tard, cela nous laissant que peu de temps pour faire face aux imprévus et difficultés. Le diagramme de Gantt et le Trello bien que nécessaires, n'ont pas été mis à jour au même rythme que l'évolution du projet. Nous aurions dû également tenir un backlog du produit final avec la liste des spécificités techniques listées par ordre d'importance et à faire valider par notre client. Compte tenu de la petite taille de l'équipe (3 personnes), nous avons également eu du

mal à nous répartir les tâches de manière précise et nous avons au final tous participé à l'ensemble du travail.

B. Structure temporelle et diagramme de Gantt

Le diagramme de Gantt final se trouve en annexe. Nous avons aussi essayé de planifier ce qu'il restait à faire sur le projet ou les améliorations possibles. Cependant, il a été de nombreuses fois modifié. Les principales modifications sont les suivantes :

- La récolte de données a pris plus de temps que prévu. Comme expliqué précédemment, il nous fallait des données sur le trafic routier minute par minute sur une journée, cependant les données étaient journalières et/ou payantes. Il n'a pas été question d'effectuer un relevé de mesures sur une période assez longue comme prévu initialement mais plutôt de nous adapter aux jeux de données disponibles et ce ne fut pas toujours aisé.
- Au cours du projet, sa définition et sa problématique ont été modifiées ce qui a changé quelques objectifs à faire et donc le planning des tâches. Par exemple, nous avions commencé une enquête pour connaître les logiciels utilisés par les développeurs et leurs habitudes de travail, mais suite au changement de problématique nous avons remplacé cette tâche par la récolte de données sur le trafic routier.

C. Répartition des tâches

Comme dit précédemment, nous avons utilisé la méthode AGILE et par conséquent après chaque réunion avec notre tuteur, nous répartissions les tâches à effectuer avant la prochaine réunion. Puis, chacun d'entre nous tenait les autres membres du groupe au courant de son avancement et donc chacun pouvait donner son avis et son expertise. De plus, avec les technologies collaboratives utilisées, nous avons pu travailler en même temps sur le même document. En conclusion, chaque personne du groupe a participé, suivi et commenté à l'élaboration de chaque partie de ce projet.

D. Répartition des technologies

Nous avons choisi d'écrire notre programme en langage Python. C'est le programme avec lequel nous avons fait les TP d'Apprentissage Artificiel, et nous avons donc déjà pu appréhender les différentes bibliothèques. Nous avons aussi parlé avec notre tuteur et ce choix a été accepté par Altran aussi. Python est un des langages les plus répandus dans les entreprises dans le domaine de l'Intelligence Artificielle.

Comme Integrated Development Environment (IDE), nous avons utilisé Google Colaboratory. Cela nous permettait de pouvoir avancer chacun en même temps sur une partie de l'algorithme, l'ajout de commentaire est aussi plus simple et permet de faire une table des matières. Enfin, pour l'exécution du programme, nous utilisons une puissance de calculs plus importante grâce à Google qui alloue ses ressources de calculs.

Même si le Google Colaboratory reste un choix pertinent pour présenter de manière claire le travail réalisé, nous envisageons l'utilisation d'un IDE plus classique par le client et il faudra donc réfléchir à la forme du produit final et notamment comment transmettre de manière efficace les données récoltées.

IV. DÉVELOPPEMENT DU PROJET

A. Etat de l'art

Bibliographie et sources dans la partie "A"de la bibliographie en fin de rapport.

L'étape préliminaire du projet a été de réaliser un état de l'art afin de mieux comprendre les enjeux du projet, ce qui a été réalisé et intégrer les données existantes. Nous avons essayé de collecter des informations sur l'empreinte carbone et/ou la consommation énergétique des technologies nécessaires à une équipe de développement travaillant sur un projet.

On souhaite donner ici quelques chiffres à titre informatif :

- Un calcul fait en 2009 montrait que le seul fait de complètement éteindre les ordinateurs la nuit dans une entreprise possédant 10 000 PC équivalait à une économie annuelle de 260 000 USD et 1 871 tonnes d'émissions de CO_2.
  Les techniques de l'information et de communication (TIC) consomment 13 % de l'électricité en France. Elles sont responsables de 5 % des émissions de CO_2 du pays.
- La consommation électrique des micro-ordinateurs augmente de 5 % tous les ans;
- L'électricité représente 10 % du budget des DSI;
- La facture électrique des ordinateurs (sur leur durée de vie) est désormais supérieure au coût d'achat ;
- Entre 2000 et 2005, la consommation électrique des centres d'exploitation a doublé dans le monde.
- En 2018 c'est entre 8 et 10 milliards de mails envoyés et 180 millions de recherches sur Google effectuées par heure.

1. Journée type dans une équipe de développement

Nous avons tout d'abord listé (de notre côté et par des entretiens) les différentes étapes d'une personne qui travaille soit en présentiel soit en télétravail afin de recenser tous les systèmes et technologies intéressantes à étudier :

	En présentiel	En télétravail
Transport aller retour	- Voiture - Transport en commun - Moyen non polluant (vélo…)	- Aucun
Moyens de communication	- Appels téléphoniques - Mails - Appel conférence call via internet (avec/sans vidéo)	- Les mêmes avec une plus grande utilisation des visioconférences (4G/5G ? Wifi ?)
Réunion	- Réunion en groupe, utilisation projecteur, ordinateurs personnels	- Réunion uniquement en visioconférence - Streaming d'écran/ Webcam ?
Stockage de l'information	- Papier : impressions, photocopies etc… - Disques durs, clés usb, ordinateurs - Serveurs	- Format numérique en ligne le plus souvent (Serveurs)
Travail	- Utilisation pc fixe/portable - Dual screen ? - Consommation de ressources importantes ? (Machine learning)	- Les mêmes mais la consommation d'électricité est la consommation personnelle de l'employée
Déplacements professionnels	- Voiture - Transport en commun - Avion - Train	- Aucun, on favorise les visioconférences.
Consommation énergie bureau	- Lumières - Chauffage - Refroidissement - Bureautique - Aération	- Les consommations personnelles de l'employé.

2. Sur les visioconférences

On a pu récolter quelques informations sur l'impact écologique des visioconférence en comparant notamment les différentes plateformes entre elles.

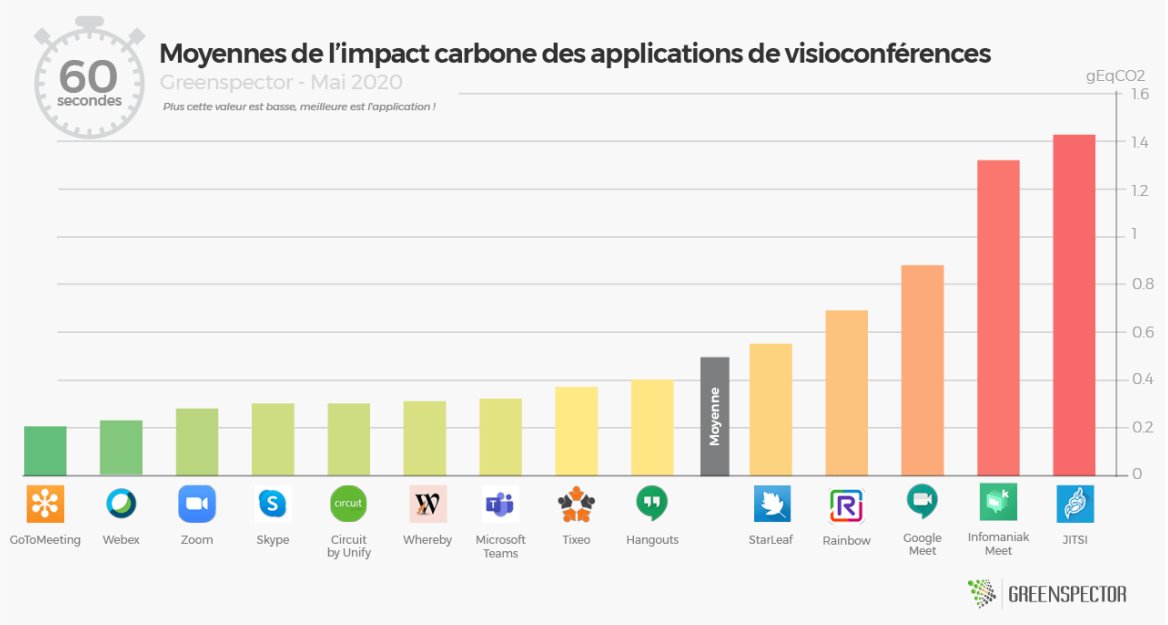


Fig 1 : Impact carbone des applications de visioconférences
source : Greenspector

On a également trouvé quelques conseils :

Une visioconférence sur mobile est en moyenne 3 fois plus impactante pour l'environnement quand on ajoute la vidéo à l'audio.

On préférera la visioconférence par rapport à un déplacement en voiture :
Comparaison pour 2 personnes qui se parlent pendant 3 heures en visio + vidéo (1 $gEqCO_2$ en moyenne par minute) alors qu'un des deux a effectué 20 kms (112 g $eqCO_2$ / km en France) aller-retour pour un face-to-face,
En visioconférence : 180*1*2= 360 g $EqCO_2$.
En voiture : 112*20= 2,4 kg $EqCO_2$. Soit 6,7 fois plus que pour la visioconférence.

3. Consommation d'un poste

On a mené des recherches sur ce qui avait déjà été fait sur le sujet de la consommation d'un ordinateur ou d'un serveur. D'après l'article *"Analysis of Energy Consumption of Software Development Process Entities"* publié sur MDPI cette consommation est en grande partie une fonction liée au nombre d'opérations effectuées par le processeur. Ils proposent donc une méthode de supervision de la consommation d'énergie lors du développement

reposant sur un programme tournant en parallèle sur le poste et mesurant le nombre d'opérations effectuées pour avoir plus tard un retour sur la consommation du projet. À partir de cette consommation électrique on peut ensuite se ramener à un équivalent en CO_2 émis en fonction de la provenance de l'énergie utilisée.

4. Pour les autres moyens technologiques

Il n'est pas tout le temps évident de trouver l'équivalent d'émission CO_2 pour toutes les technologies citées précédemment. Cependant une méthode peut consister à calculer ou déterminer la consommation de données et de trouver ensuite une équivalence en CO_2.

Par exemple :

- une heure de surf en 3G : +/- 15 Mo
- un mail sans pièce jointe : +/- 100 Ko
- un mail avec pièce jointe : +/- 1 Mo (tout dépend de la taille de la pièce jointe)
- une heure de visioconférence : +/- 150 Mo
- téléchargement d'une présentation Powerpoint : +/- 5 Mo
- téléchargement d'une application : +/- 5 Mo (certaines applications sont plus lourdes)
- visionner 1 heure de streaming vidéo en 4G : +/- 350 Mo

L'étude de référence est celle de l'ADEME de 2011, "Analyse comparée des impacts environnementaux de la communication par voie électronique" , fondée sur le choix d'1 Mo comme unité de mesure (c'est-à-dire un message équivalent à 1 Mo transmis).

- envoyer 1 Mo à 1 personne = 25 Wh = 25 min d'utilisation d'une ampoule de 60 W
- envoyer 1 Mo à 1 personne = 6 g de pétrole + 20 g de CO_2.

Exemple individu : 20 mails par jour = 1000 km parcourus en voiture/personne/an
Exemple collectivité : entreprise de 100 personnes = 13,6 tonnes d'équivalent CO_2/an = 14 allers-retours Paris-New York.

5. Consommation énergétique de l'environnement de travail

Concernant le bilan énergétique d'un poste de travail nous avons trouvé les données suivantes :

	Puissance [W/m²]	Consommation [kWh/m²]	Énergie primaire [kWh/m²]	Coût [€/m²]
	70	96	96	6,0
	100	28	78	4,5
	–	7	20	1,1
	–	(53) 12	(148) 34	(8,5) 2,0
	12	25	70	4,0
	15	31	87	5,0
	–	–	–	–
	–	**201 [kWh/m²]**	**385 [kWh/m²]**	**22,6 [€/m²]**

Fig 2 : Bilan énergétique annuel d'un poste de travail
source : energieplus-lesite.be

B. Consommation carburant et émissions CO_2

Afin de pouvoir avancer sur la partie technique et en attendant de trouver des données à exploiter nous avons simuler nos données d'émissions de CO_2 grâce à des fonctions prenant en paramètre des critère comme le modèle de la voiture, la température extérieure, la distance à parcourir ou encore l'heure de la journée. Toutes ces variables sont également simulées et ne sont pas représentatives du use case réel mais nous ont permis cependant de tester la création d'un algorithme d'apprentissage artificiel avec ses différents paramètres et de tester les méthodes citées plus loin dans le document. Les méthodes présentées dans la partie "'E. Apprentissage artificiel avec Python" ont donc été testées au préalable sur notre fonction test.

L'objectif à terme étant de créer un algorithme d'apprentissage artificiel sur le domaine de l'empreinte écologique nous essaierons de trouver des jeux de données intéressants sur lesquels mettre en pratique ce que nous avons appris.

Au final l'important n'est pas de trouver des données qui collent parfaitement au use case défini mais plutôt de trouver un jeu de données dans le thème pour mettre en pratique les méthodes du guide et montrer que l'on peut les appliquer par la suite à d'autres usages similaires.

C. Apprentissage artificiel avec Python

Dans le but de fournir une méthode de conception de modèle pour le thème de l'empreinte carbone, nous avons listé les étapes qu'il nous semblait important de détailler. Nous allons voir dans les parties suivantes comment créer notre modèle de manière cohérente en comprenant dans un premier temps de quoi celui-ci est composé.

1. Choix de la méthode utilisée : choix des hyperparamètres.

Nous allons essayer d'étudier l'influence des hyperparamètres de notre modèle en regardant le use case de la consommation/émissions des voitures. Nous nous concentrons dans ce projet sur des méthodes de régression et principalement sur l'utilisation de réseaux de neurones à plusieurs couches. L'utilisation de ces derniers peut sembler être une méthode exagérée surtout compte tenu de la faible complexité et du faible nombre de données de notre modèle de test mais on peut supposer que cela fonctionnera avec un modèle plus complexe. Dans l'idéal nous essaierons d'autres méthodes de régression afin de les comparer.

Qu'est ce qu'un hyper paramètre ?

En machine learning un hyperparamètre est un paramètre du modèle qui est fixé à l'avance et va influencer l'apprentissage de ce dernier (nombre de couches cachées par exemple). C'est donc un paramètre externe contrairement aux paramètres internes comme le poids de chaque neurone qui va évoluer au cours de l'apprentissage. Le nombre et le type de ces hyperparamètres va évoluer en fonction du modèle choisi.

Le temps requis pour entraîner et tester un algorithme d'apprentissage automatique peut dépendre du choix de ses hyperparamètres. Un hyperparamètre est généralement de type continu ou entier, conduisant à des problèmes d'optimisation de type mixte. L'existence de certains hyperparamètres est conditionnée à la valeur d'autres, par exemple la taille de chaque couche cachée dans un réseau de neurone peut être conditionnée par le nombre de couches ce celui-ci les deux ayant de l'influence l'un sur l'autre

Hyperparamètres couramment utilisés :

- Learning Rate
- Number of Epochs
- Batch size
- Hidden Layers
- Hidden Units
- Activations Functions

Ces hyperparamètres se divisent 2 catégories :

- Les paramètres de l'optimiseur.

- Les paramètres spécifiques au modèle.

1. Les paramètres de l'optimiseur

a. Le learning rate

Le taux d'apprentissage peut être vu comme la vitesse d'apprentissage du modèle. Il correspond au pas d'avancement dans la recherche du minimum de la fonction de perte. Un taux d'apprentissage trop faible permettra une recherche plus fine et plus précise mais prendra beaucoup plus de temps à converger tandis qu'un taux trop grand pourrait dépasser l'état idéal et risquerait de ne pas converger.

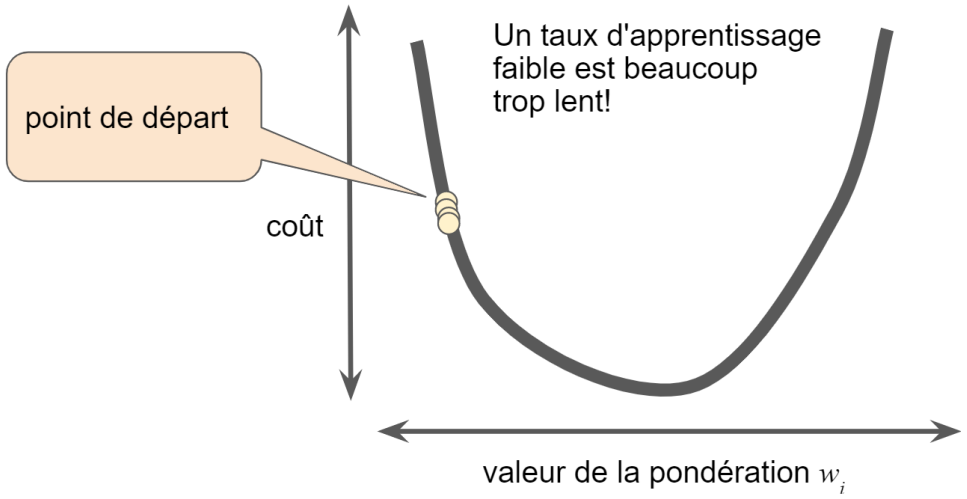


Fig 3 : Impact d'un taux d'apprentissage trop faible sur un modèle.

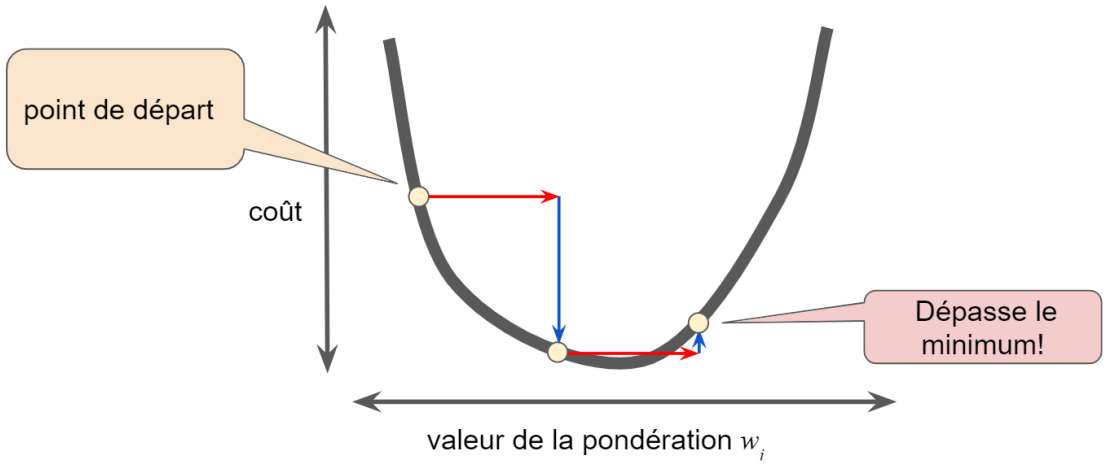


Fig 4 : Impact d'un taux d'apprentissage trop élevé sur un modèle.

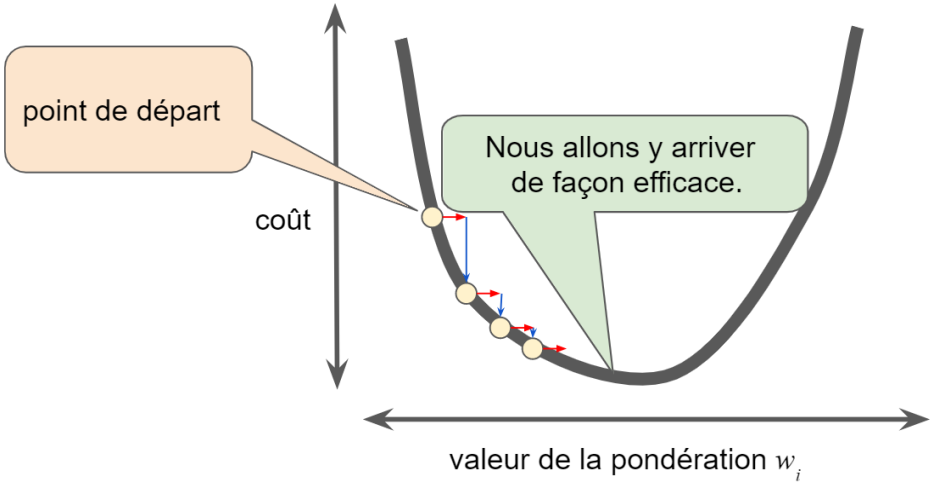


Fig 5 : Taux d'apprentissage idéal pour la convergence d'un modèle..

Afin d'obtenir une convergence plus rapide, d'éviter les oscillations et de rester coincé dans des minima locaux indésirables, le taux d'apprentissage est souvent varié pendant l'entraînement, soit en fonction d'un programme de taux d'apprentissage, soit en utilisant un taux d'apprentissage adaptatif.

Le taux d'apprentissage idéal dans une dimension est $\frac{1}{f''(x)}$ (dérivée seconde de f(x) pour x).

Le taux d'apprentissage idéal pour deux dimensions ou plus est l'inverse de la hessienne (matrice de dérivées partielles secondes).

La formule est plus complexe pour les fonctions convexes générales.

La plupart du temps le taux d'apprentissage est adaptatif et évolue au cours du temps/nombre d'étapes. Le problème est que celui-ci va varier en fonction du problème étudié et des données d'apprentissage cependant il existe de nombreux types d'algorithmes de descente de gradient adaptatifs tels que Adagrad , Adadelta, RMSprop , Adam qui sont généralement intégrés dans des bibliothèques telles que Keras .

b. Batch / Mini-batch

Le batch size représente le nombre d'exemples du dataset d'apprentissage que nous allons présenter au modèle avant de l'entraîner, i.e. de mettre à jour ses paramètres internes. Il existe 3 modes pour le choix de ce paramètre :

1. **mode batch** : où la taille du batch est égale à l'ensemble de données total.
2. **mode mini-batch** : où la taille du batch est supérieure à un mais inférieure à la taille totale du jeu de données. Généralement, un nombre qui peut être divisé en la taille totale du jeu de données.
3. **mode stochastique** : où la taille du batch est égale à un. Par conséquent, le gradient et les paramètres du réseau neuronal sont mis à jour après chaque échantillon.

Une taille de batch trop grande entraîne une mauvaise généralisation.

Pour une fonction à optimiser convexe :

D'un côté, l'utilisation d'un batch égal à l'ensemble de données garantit la convergence vers les optima globaux de la fonction objectif. Cependant, cela se fait au prix d'une convergence empirique plus lente vers ces optima. D'un autre côté, il a été démontré empiriquement que l'utilisation de batch plus petits permet une convergence plus rapide vers de «bonnes» solutions. Cela s'explique intuitivement par le fait que des tailles de batch plus petites permettent au modèle de «commencer à apprendre avant d'avoir à voir toutes les données». L'inconvénient de l'utilisation d'une taille de batch plus petite est qu'il n'est pas garanti que le modèle converge vers les optima globaux.

Pour une fonction à optimiser non convexe (la plupart du temps en réseau de neurones) :

Il a été observé empiriquement que les petites tailles de batch ont non seulement une dynamique d'entraînement plus rapide, mais également une généralisation meilleure. Mais il y a des limites : nous savons qu'une taille de batch de 1 fonctionne généralement assez mal. Il est généralement admis qu'il existe un «point idéal» pour la taille du batch entre 1 et l'ensemble de données de formation qui fournira la meilleure généralisation.

Des valeurs communément utilisées pour expérimenter : {1, 2, 4, 8, 16, 32, 64, 128, 256}.

Une taille de batch plus petite induit plus de bruit dans les calculs d'erreur, souvent plus utile pour empêcher le processus d'apprentissage de s'arrêter aux minima locaux. Une juste valeur pour la taille du batch = 32.

Exemple :

Soit un réseau de neurones pour la classification avec les paramètre suivants :

- 2 couches cachées entièrement connectées (FC), 1024 unités chacune
- Non-linéarités ReLU
- perte: probabilité log négative
- optimiseur: SGD
- taux d'apprentissage: 0,01
- époques: 30

Les graphiques ci dessous représentent l'influence des tailles de batch sur la précision de test du modèle en fonction du nombre d'époques réalisées :

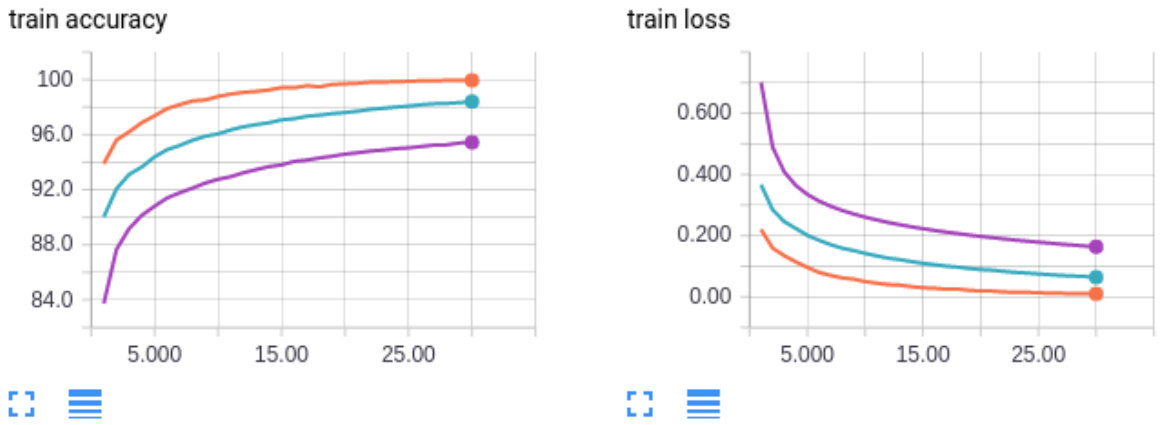


Fig 6 : Influence de la taille de batch sur la précision et la perte du set d'entraînement.

Fig 7 : Influence de la taille de batch sur la précision et la perte du set de test.

- Courbes orange: taille de lot 64
- Courbes bleues: taille de lot 256
- Courbes violettes: taille de lot 1024

Résumé :

Avantages de l'utilisation d'une taille de batch < nombre de tous les échantillons:

- Cela nécessite moins de mémoire. Étant donné que l'on entraîne le réseau en utilisant moins d'échantillons, la procédure globale de formation nécessite moins de mémoire. C'est particulièrement important si l'on ne parvient pas à insérer l'ensemble des données dans la mémoire de notre machine.

- En règle générale, les réseaux s'entraînent plus rapidement avec des mini-batch. C'est parce que nous mettons à jour les poids après chaque propagation. Si on utilise tous les échantillons pendant la propagation, on ne fait qu'une seule mise à jour pour les paramètres du réseau.

Inconvénients de l'utilisation d'une taille du batch < nombre de tous les échantillons:

Plus le batch est petit, moins l'estimation du gradient sera précise. Dans la figure ci-dessous, on peut voir que la direction du gradient du mini-batch (couleur verte) fluctue beaucoup plus par rapport à la direction du gradient du batch complet (couleur bleue).

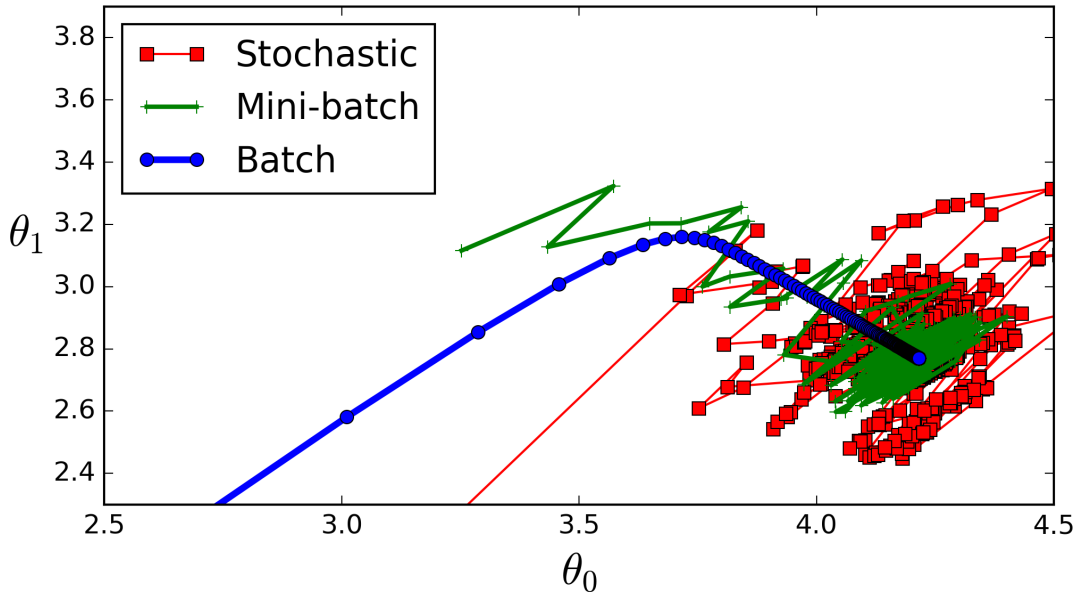


Fig 8 : Variation du gradient en fonction du type de batch utilisé.

c. Epochs

Une époque ou itération correspond à un cycle d'apprentissage complet sur l'intégralité de l'ensemble de données de manière à ce que chaque exemple ait été vu une fois. Une époque est donc composée de 1 ou plusieurs batch. Afin d'obtenir un bon résultat il faudra donc plusieurs époques (centaines ou milliers généralement).

Pour choisir le bon nombre d'époques pour l'étape d'entraînement, la métrique à laquelle nous devons prêter attention est l' erreur de validation.
La méthode *manuelle* intuitive consiste à faire entraîner le modèle sur autant d'itérations que nécessaire tant que l'erreur de validation continue de diminuer.

Il existe une technique qui peut être utilisée nommée Arrêt anticipé pour déterminer quand arrêter l'entraînement du modèle; il s'agit d'arrêter le processus de formation au cas où l'erreur de validation ne s'est pas améliorée au cours des 10 ou 20 dernières époques. Cela permet d'éviter le surapprentissage. [Fig 9]

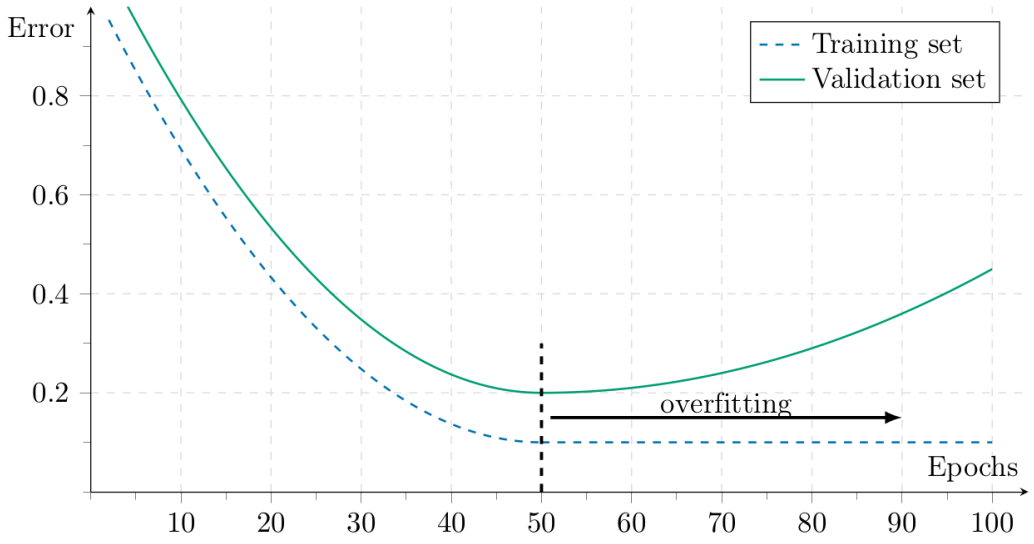


Fig 9 : Évolution de l'erreur en fonction du nombre d'époques.

Dans l'exemple précédent le nombre idéal d'itérations est 50 mais ce nombre varie en fonction du réseau, du problème et des données.

2. Les paramètres du modèle

a. Nombre de hidden units

Le nombre optimal de neurones par couche cachée est assez mystérieux. Pour une fonction simple à approximer il ne suffit que de quelques neurones le plus souvent. Pour une fonction plus complexe, on doit généralement augmenter ce nombre. Dépasser d'un peu le nombre optimal de *hidden units* n'est pas gênant mais le dépasser de beaucoup peut amener un effet de sur-apprentissage.

b. Première hidden layer

Un résultat empirique a permis de montrer que définir un nombre de *hidden units* supérieur au nombre d'entrées sur la première couche permettait d'obtenir de meilleurs résultats.

c. Nombre de hidden layers

Il arrive souvent qu'un réseau à 3 couches surclasse un réseau à 2 couches mais qu'aller au-delà n'aide pas beaucoup plus. (Sauf pour les réseaux convolutionnels où le nombre de couches cachées améliore la qualité).

3. Techniques d'optimisation des hyperparamètres :

L'objectif de l'optimisation des hyperparamètres dans l'apprentissage automatique est de trouver les hyperparamètres d'un algorithme d'apprentissage automatique donné qui renvoient les meilleures performances mesurées sur un ensemble de validation. (Les hyperparamètres, contrairement aux paramètres du modèle, sont définis avant l'entraînement.)

L'optimisation des hyperparamètres est représentée sous forme d'équation comme suit:

$$x^\star = \arg\min_{x \in \mathcal{X}} f(x)$$

Ici, $f(x)$ représente un score objectif à minimiser - tel que RMSE ou taux d'erreur - évalué sur l'ensemble de validation; x^* est l'ensemble d'hyperparamètres qui donne la valeur la plus basse du score, et x peut prendre n'importe quelle valeur dans le domaine X. En termes simples, nous voulons trouver les hyperparamètres de modèle qui donnent le meilleur score sur la métrique de l'ensemble de validation.

Les différentes métriques utilisées pour la régression et donc pour notre problème actuel sont les suivantes :

- **Erreur absolue moyenne** : moyenne de toutes les erreurs dans le modèle, l'erreur correspondant à la distance de la valeur prédite à la valeur réelle. Elle est souvent abrégée en MAE.
- **Racine de l'erreur quadratique moyenne :** Mesure la moyenne des carrés des erreurs, puis prend la racine de cette valeur. Elle est souvent abrégée en RMSE.
- **Erreur absolue relative :** Représente l'erreur sous la forme d'un pourcentage de la valeur réelle.
- **Erreur quadratique relative** : normalise l'erreur quadratique totale en la divisant par l'erreur quadratique totale des valeurs prédites.
- **Coefficient de détermination** : nombre unique qui indique la conformité des données à un modèle. La valeur 1 signifie que le modèle correspond exactement aux données. La valeur 0 signifie que les données sont aléatoires ou qu'elles ne peuvent pas être représentées par le modèle. Il est souvent appelé r 2 , R 2 ou R carré.

Il existe 4 grandes méthodes pour chercher les meilleurs hyperparamètres :

- **Recherche manuelle**: En utilisant nos connaissances sur le problème, on devine les paramètres et on observe le résultat. Sur la base de ce résultat, on modifie les paramètres. On répète cette procédure jusqu'à ce que l'on trouve des paramètres qui fonctionnent bien.

- **Grid Search:** En se servant de nos connaissances sur le problème, on identifie les plages pour les hyperparamètres. On sélectionne ensuite plusieurs points de ces plages, généralement répartis uniformément. On entraîne notre réseau en utilisant chaque combinaison de paramètres et on sélectionne la combinaison la plus performante. Sinon, on peut répéter notre recherche sur un domaine plus étroit, centré sur les paramètres les plus performants.
  Le temps de résolution de cette méthode explose avec l'augmentation des dimensions : *curse of dimensionality* [Fig 10].

- **Random Search**: Comme pour la recherche sur grille, on utilise la connaissance du problème pour identifier les plages des hyperparamètres. Cependant, au lieu de choisir

les valeurs de ces plages de manière méthodique, on les sélectionne au hasard. On répète cette procédure jusqu'à ce que l'on trouve des paramètres qui fonctionnent bien ou on utilise ce que l'on a appris pour affiner notre recherche. Cette méthode se révèle plus efficace car moins coûteuse que Grid Search [Fig 10].

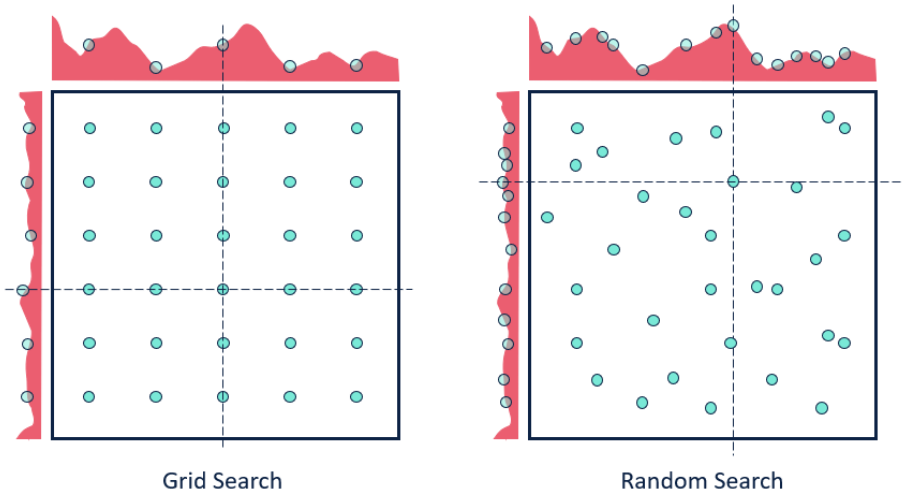


Fig 10 : Comparaison Grid Search et Random Search.

- **Optimisation bayésienne**: Des travaux plus récents ont été consacrés à l'amélioration de ces autres approches en utilisant les informations acquises lors d'une expérience donnée pour décider de la manière d'ajuster les hyper paramètres pour l'expérience suivante.

Sur l'optimisation bayésienne :

La recherche par grille et la recherche aléatoire sont légèrement meilleures que le réglage manuel, car nous configurons une grille d'hyperparamètres et exécutons automatiquement le cycle entraînement-prédiction-évaluation en boucle. Cependant, même ces méthodes sont relativement inefficaces car elles ne choisissent pas les hyperparamètres suivants à évaluer en fonction des résultats précédents. Les grilles et les recherches aléatoires ne sont pas du tout informées par les évaluations passées, et par conséquent passent souvent beaucoup de temps à évaluer les «mauvais» hyperparamètres.

Pour résumer l'optimisation bayésienne en une phrase : "*construire un modèle de probabilité de la fonction objectif et l'utiliser pour sélectionner les hyperparamètres les plus prometteurs à évaluer dans la vraie fonction objectif.*"

Approach	*ML*	*DL*	*Manual/ Auto*	*Cost*	*Space expl.*	*History*	*Parallel/ Distributed*
Babysitting	👍	👎	🐼	💰	Low	✅	No
Grid	👍	👎	🎩	💰 💰 💰 💰	High	🙈	Yes
Random	👍	👍	🎩	💰 💰 💰	Medium	🙈	Yes
Bayes SMBO	👍	👍	🎩	💰 💰	Medium - Driven	✅	*

* The SMBO by definition is sequential, however, *it's possible, but not trivial,* to build parallel/distributed solution upon these optimizations.

Fig 11 : Tableau comparatif des différentes méthodes d'optimisation des hyperparamètres.

2. Explication des métriques d'évaluation possibles.

Dans cette partie nous allons nous intéresser au choix de la métrique d'erreur. C'est-à-dire quelle fonction utiliser pour juger de la performance de notre modèle.

Le cas de la régression

Notre use-case consiste à prédire une valeur de sortie, nous sommes donc dans le cas d'une régression et les fonctions d'erreur ne seront donc pas les mêmes que pour un problème de classification par exemple [Fig 12].

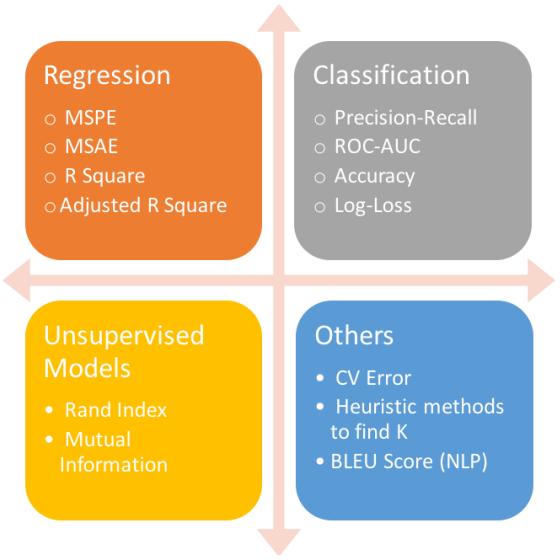


Fig 12 : Métriques d'évaluation des modèles en fonction des types de problèmes.

1. Les différentes métriques

   a. MSE (Mean Squared Error)

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)^2}_{\text{The square of the difference between actual and predicted}}$$

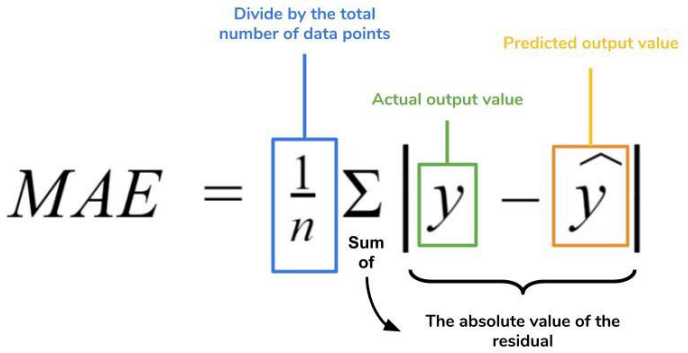
L'erreur quadratique moyenne est une erreur couramment utilisée car simple. Elle permet de donner du poids aux larges erreurs, est positive et est facilement différentiable ce qui facilite les problèmes d'optimisation. Cependant les valeurs peuvent vite devenir importantes et difficilement interprétables, c'est pourquoi on préférera souvent la métrique suivante.

   b. RMSE (Root Mean Squared Error)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

La RMSE est sensiblement la même chose que la MSE à la différence près que l'on passe le résultat à la racine. Cela permet de compenser les trop grandes erreurs et de retrouver un résultat de même dimension que la valeur calculée ce qui facilite l'interprétation.

c. MAE (Mean Absolute Error)



La MSE est plus robuste aux outliers, elle est utile quand on ne souhaite pas pénaliser les erreurs larges par rapport aux erreurs faibles. Elle n'est pas utilisable dans les cas où l'on souhaite mettre en évidence les outliers. Tout comme la RMSE, elle possède l'avantage d'exprimer l'erreur dans l'unité de la valeur de sortie.

d. R^2 et Adjusted R^2

$$R^2 = 1 - \frac{\text{MSE(model)}}{\text{MSE(baseline)}}$$

Le coefficient de détermination est une autre métrique pour évaluer les performances d'un modèle. Cela permet de comparer notre modèle avec une baseline constante et de dire de combien notre modèle est meilleur. La baseline est choisie en prenant la moyenne des données et en effectuant une MSE à cette moyenne. Cette métrique sera toujours inférieure ou égale à 1.

$$R_a^2 = 1 - \left[\left(\frac{n-1}{n-k-1}\right) \times (1 - R^2)\right]$$

where:
n = number of observations
k = number of independent variables
R_a^2 = adjusted R^2

Un des principaux inconvénients de R2 est que le résultat s'améliore avec le nombre de termes utilisés, qu'il y ait une amélioration du modèle ou non. On peut alors avoir recours au R2 ajusté qui prend en compte le nombre de termes tout en ayant la même signification. C'est une version améliorée.

Conclusion

MSE, RMSE et MAE sont les choix les plus appropriés pour comparer deux modèles entre eux.
Le choix se porte souvent sur RMSE ou MAE. On pourra choisir d'afficher les deux car la MAE permet d'interpréter le modèle plus facilement et d'avoir une idée des performances tandis que la RMSE permettra de mettre en évidence les larges erreurs et sera à utiliser dans les calculs compte tenu de sa différentiabilité. Il peut être intéressant de noter que c'est d'ailleurs la métrique retenue par Kaggle. On peut cependant tout à fait utiliser MSE si nous n'avons pas de trop grandes valeurs et MAE si on ne veut pas pénaliser les grandes erreurs.

3. Etude de l'importance des features

1. Pourquoi réduire la dimension ?

En plus d'aider à la visualisation des données, la réduction de dimension a plusieurs effets bénéfiques.

a. Réduire les coûts

Réduire le nombre de dimensions de nos données d'entrée a plusieurs avantages. Cela permet de :

- Réduire le coût en **espace mémoire** : moins d'informations à stocker.
- Réduire les **temps de calcul** : on réduit la complexité des algorithmes d'apprentissage.
- Réduire le coût **d'acquisition des données** : moins de données essentielles à acquérir donc moins de temps/ressources dépensées dans l'acquisition de ces dernières.

b. Améliorer la qualité des modèles d'apprentissage

En réduisant le nombre de variables explicatives on réduit la complexité des modèles et on évite ainsi le **surapprentissage**. En plus de cela des variables non pertinentes peuvent induire l'algorithme en erreur.

Intervient également le **fléau de la dimension (curse of dimensionality)** qui est le terme utilisé pour expliquer qu'il est difficile de faire de l'apprentissage en haute dimension car cela requiert un nombre de points beaucoup plus important pour couvrir tout l'espace.

2. L'analyse en composantes principales

a. Méthode

L'objectif de l'analyse en composantes principales est de trouver une nouvelle base orthonormée dans laquelle représenter nos données, telle que la variance des données selon ces nouveaux axes soit maximisée [Fig 13].

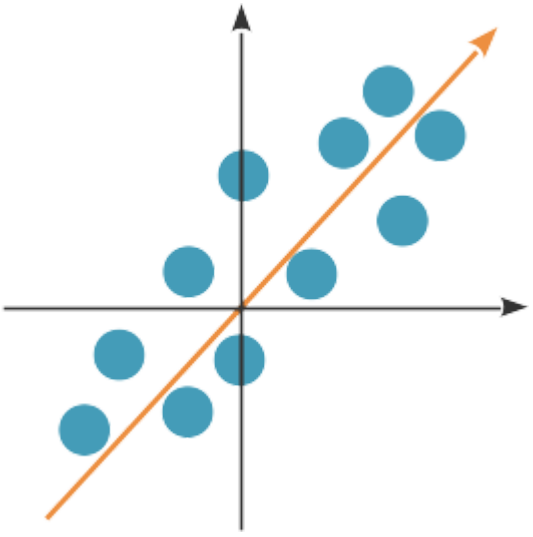


Fig 13 : Illustration d'une réduction par analyse en composante principale.

Sur la figure 13 ci-dessus on remarque que la variance des données le long de l'axe orange est grande. En utilisant cet axe comme unique dimension on réduit la dimension de nos données de 2 à 1 et on continue de pouvoir distinguer les points entre eux.

On passera rapidement sur la méthode mathématique du calcul des composantes principales, on retiendra que l'on peut construire autant de composantes principales que le nombre de dimensions initiales.

En posant :

- $X \in \Re^{pxn}$ la matrice de nos données avec n données en p dimensions.
- $\Sigma = XX^T$ la matrice de covariance des données.
- $w_1 \in \Re^p$ une direction telle que la projection des données X sur w_1 est $z_1 = {w_1}^T X$.

Trouver la première composante principale c'est trouver w1 tel que :

- ${w_1}^T \Sigma w_1$ est maximale.
- $\left|\left|w_1\right|\right| = 1$.

La deuxième composante se trouve de la même manière mais doit être orthogonale à la première et ainsi de suite.

b. Combien de composantes choisir ?

La variance totale est données par : $Tr(\Sigma) = \lambda_1 + \ldots + \lambda_p$ et la variance expliquée par les k premières composantes est donnée par : $\lambda_1 + \ldots + \lambda_k$.

S'offre à nous deux possibilité :

- Choisir le nombre de composantes qui explique un seuil de la variance totale (90%) par exemple [Fig 14].

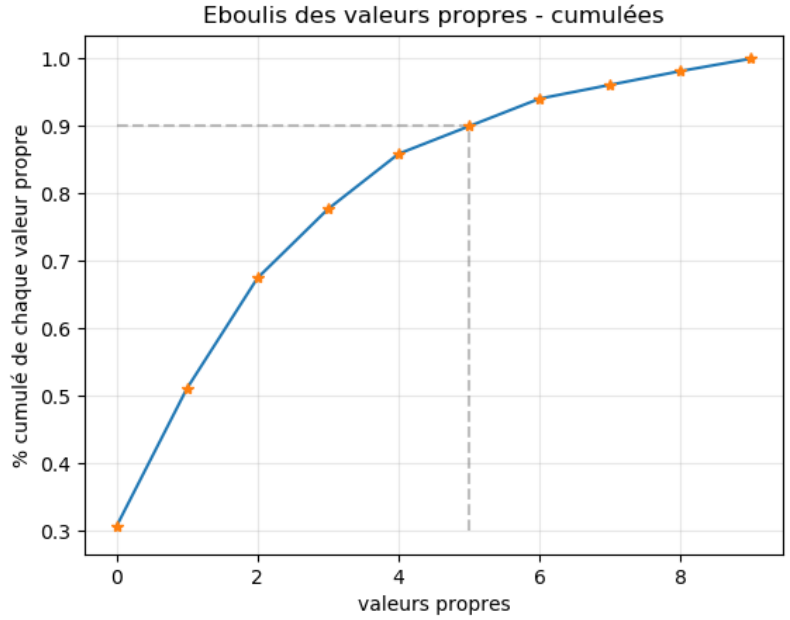


Fig 14 : Sélection du nombre de composantes principales par proportion de la variance expliquée.

- Éliminer les composantes n'ayant pas une grande implication dans l'augmentation de la variance.

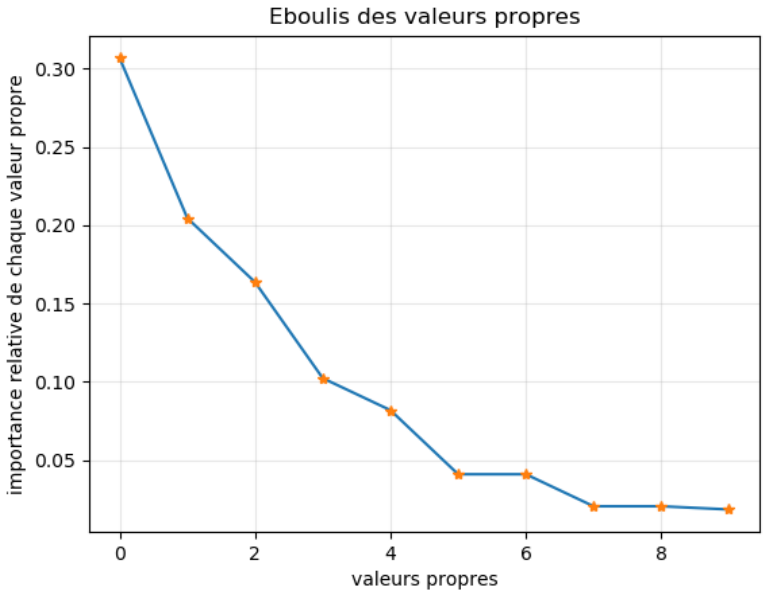


Fig 15 : Sélection des composantes les plus représentatives des données.

c. Calcul grâce à Sklearn

En utilisant le module *decomposition* de la librairie *Sklearn* on peut donc calculer ces composantes à partir de nos données [Fig 16] :

```
[ ]  pca = decomposition.PCA(n_components=5)
     pca.fit(X_data)

     print(pca.explained_variance_ratio_)
     print(pca.explained_variance_ratio_.sum())

     [0.21450415 0.20776764 0.20417998 0.1907876  0.18276063]
     0.9999999999999998
```

Fig 16 : Analyse en composantes principales du modèle simulé sans corrélations.

Ici on l'a utilisé sur notre modèle primitif d'émissions de CO_2 en cherchant 5 composantes principales parce qu'on 5 dimensions, cela nous permet ensuite d'afficher le ratio de variance que représente chacune d'elle. Malheureusement dans ce cas précis on constate que toutes les composantes se valent et qu'il est donc compliqué de réduire la dimension via cette méthode. Afin de tester la méthode on a créé une variables "Time" telle que $Time = 1.5 * Distance$ et on a effectué une analyse en composantes principales sur 6 composantes [Fig 17] :

```
[3.33873754e-01 1.75933904e-01 1.67871215e-01 1.62676047e-01
 1.59645080e-01 2.86905720e-33]
1.0
```

Fig 17 : Analyse en composantes principales du modèle simulé avec corrélations.

On constate cette fois que la variance est quasiment uniquement expliquée par les 5 premières composantes principales (85% de la variance est même expliquée en 4 composantes) et on en déduit donc la corrélation entre certaines variables.

Après avoir choisi nos composantes principales, il est également possible d'afficher la contribution de chaque variable explicative à ces composantes.

Afin d'illustrer l'intérêt de l'analyse en composantes principales voici une série de mesure de temps d'exécution de l'apprentissage d'un algorithme de régression logistique en fonction du nombre de composantes principales :

Variance Retained	Number of Components	Time (seconds)	Accuracy
1.00	784	48.94	0.9158
0.99	541	34.69	0.9169
0.95	330	13.89	0.9200
0.90	236	10.56	0.9168
0.85	184	8.85	0.9156

Fig 18 : Vitesse d'apprentissage d'un modèle en fonction du nombre de composantes principales.

Jusqu'à 85% de la variance retenue, la précision de l'algorithme ne se détériore pas et on gagne jusqu'à plus de 500% en temps de calcul.

L'analyse en composante principale sera d'autant plus utile que le nombre de variables explicatives est important. On sert en général pour passer d'un millier voir d'un million de dimensions à une centaine par exemple.

1. Local Surrogate (LIME)

a. Définition

Les modèles surrogés locaux sont des modèles interprétables utilisés pour expliquer des prédictions individuelles des modèles "boîte noire".
LIME signifie Local Interpretable Model-agnostic Explanation et la méthode LIME se concentre donc sur l'entraînement de modèles locaux pour expliquer les prédictions individuelles.

LIME teste ce qu'il survient sur les prédictions lorsque l'on effectue des variations dans le dataset fournit au modèle. Il génère donc un nouveau dataset composé de données permutées et de leurs sorties par le modèle initial. Sur ce nouveau dataset LIME entraine un modèle interprétable pondéré par la distance des nouvelles données aux données d'intérêt initiales.

Le résultat sera donc efficace sur les prédictions locales mais pas pour une estimation locale du modèle. Ce phénomène s'appelle "Local fidelity".

En résumé on suit les étapes suivantes :

- On sélectionne l'instance d'intérêt pour laquelle on souhaite avoir l'explication de sa prédiction.
- On perturbe le jeu de données et on obtient les prédictions de la boîte noire pour ces nouveaux points.
- On pondère les nouveaux échantillons en fonction de leur distance avec l'échantillon d'intérêt.
- On entraîne un modèle pondéré et interprétable sur l'ensemble des données avec variations.
- On explique la variation en interprétant le modèle local.

b. Exemple sur notre use case

On essaye d'interpréter la prédiction de la 10ème données du dataset de test [Fig 19] :

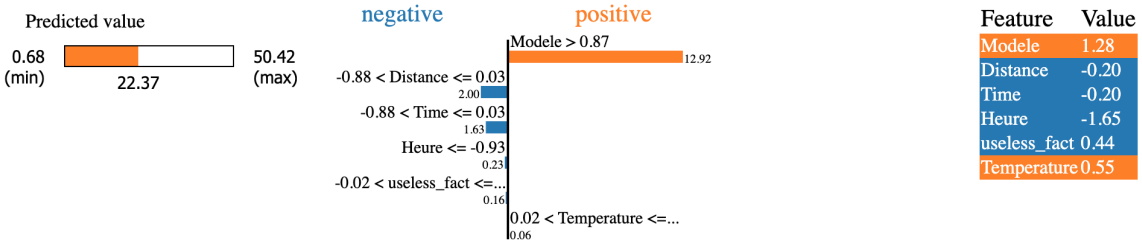


Fig 19 : Influences des variables explicatives sur la prédiction d'une entrée.

Les résultats sont sensiblement similaires à ceux du module SHAP [Fig 24], le résultat positif de notre prédiction est principalement expliqué par la valeur de la variable "Modele".

c. Avantages de LIME

- Facile d'utilisation.
- Rapide.
- On a une bonne estimation de la fiabilité de l'interprétation de par la mesure de fidélité. C'est à dire plus notre modèle LIME approche bien le modèle de base plus cette fiabilité sera élevée.

d. Inconvénients de LIME

- Fiable uniquement sur les prédictions locales.
- Moins complet que SHAP.

2. SHAP

a. Définition

SHAP est un module Python qui signifie SHapley Additive exPlanations et qui se base sur la théorie des jeux et plus particulièrement la Valeur de Shapley qui consiste à donner une répartition équitable des gains aux joueurs. L'ensemble des joueurs est appelé une coalition C. On appelle fonction caractéristique la fonction v telle que v(C) donne la valeur maximale de la coalition C. Le but est donc, à partir des valeurs de fonctions caractéristiques de toutes les sous-coalitions de C, de trouver la meilleure répartition des gains.

Exemple :

Le jeu avec la fonction caractéristique suivante:

$$v(1,2,3) = 120\ ;\ v(1,2) = 0\ ;\ v(1,3) = v(2,3) = 120\ ;\ v(1) = v(2) = v(3) = 0$$

a un noyau correspondant au point (0,0,120). Par contre, les valeurs de Shapley sont :

$$\varphi_1 = \varphi_2 = 20 \quad ; \quad \varphi_3 = 80$$

et ce point n'est pas dans le noyau.

Pour le cas du machine learning le but va être de tester toutes les combinaisons de features possibles afin de déterminer l'importance de chacune d'entre elles. En théorie il y a 2^n possibilités.

L'utilisation de SHAP permet d'obtenir deux choses importantes :

- L'interprétabilité globale : les valeurs de SHAP collectives permettent de connaître de combien chaque variable contribue, que ce soit de manière positive ou négative.

- L'interprétabilité locale : On peut obtenir les valeurs SHAP de chaque observation séparément. Cela permet de savoir pourquoi ce cas particulier reçoit telle valeur (ou classification) et quelles sont les contributions de chaque variable.

Il est important de noter que SHAP ne fournit pas d'explications de causalité mais uniquement d'interprétabilité du modèle.

b. Summary plot

Le summary plot va nous donner de l'information sur l'effet de chaque variable sur la valeur cible. On va pouvoir identifier si une variable est importante ou non et si elle affecte positivement ou non le résultat.

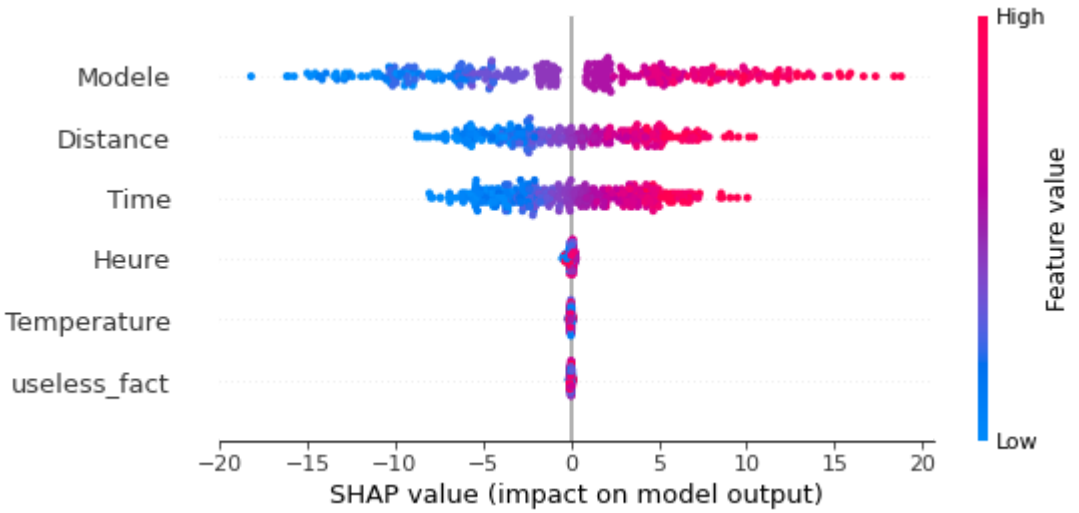


Fig 21 : Summary Plot du modèle simulé avec SHAP.

- Importance des variables : Les variables sont rangées de haut en bas dans l'ordre d'importance sur la valeur cible. Ici on constate que c'est la variable "modèle" qui aura en général le plus d'influence sur la sortie.
- Impact : La position d'un point sur l'axe horizontal indique l'amplitude de l'impact sur la valeur de sortie, de manière négative à droite et de manière positive à gauche. On constate donc que pour certaines instances, le modèle de la voiture a une un impact considérable mais que la variable température n'a jamais eu d'incidence sur la valeur de sortie.
- Valeur d'origine : la couleur d'un point détermine si la valeur de la variable était importante (en rouge) ou faible (en bleue). On en déduit par exemple que des valeurs importantes de modèle ont un impact positif sur la sortie.

c. Dependence plot

Le dependence plot est un une figure intéressante. Elle permet d'obtenir l'effet marginal d'une ou deux variables sur la valeur de sortie. Il nous indique si la relation entre la variable et la valeur prédite est linéaire, monotone ou plus complexe.
Le module SHAP inclut également la variable avec laquelle notre variable sélectionnée interagit le plus.

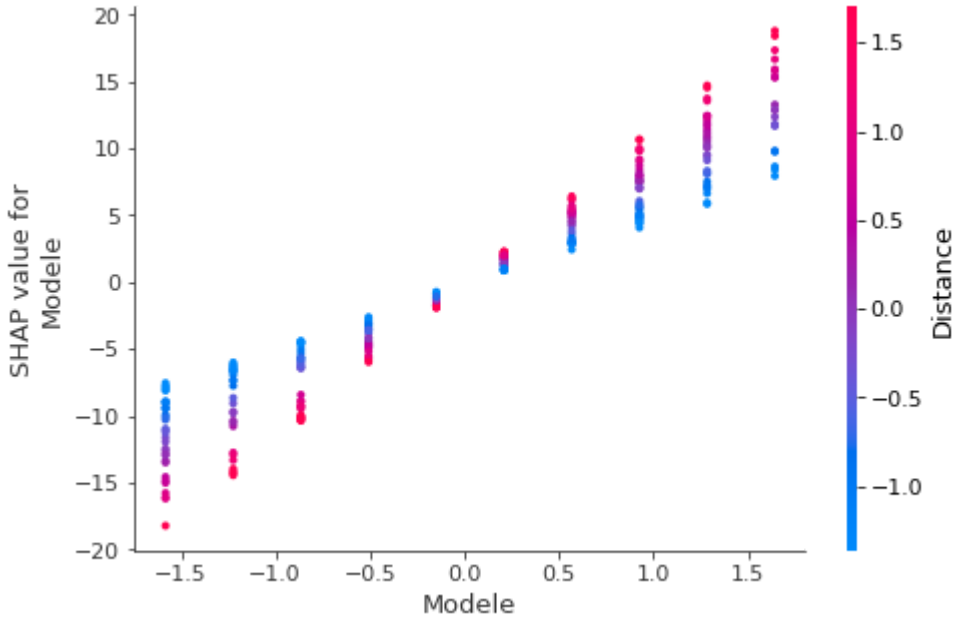


Fig 22 : Dependence Plot du modèle simulé avec SHAP.

On a par exemple ici représenté le dependence plot de “Modele” et on constate qu’il y a une tendance à peu près linéaire entre la variable “Modele” et la valeur ciblée. On constate également que la variable avec laquelle elle interagit le plus est la variable “Distance”.

d. Individual force plot

L’individual force plot est aussi très intéressant puisqu’il permet d’avoir le détails des interactions entre les variables explicatives et la sortie pour une instance particulière. Pour l'exemple, on choisit arbitrairement de s’intéresser à la dixième valeur du set de test. Il est intéressant de comparer les valeurs obtenues aux moyennes des variables de l’échantillon de test :

```
Distance       -3.367677e-17
Heure           2.220446e-17
Modele          9.992007e-17
Temperature    -5.107026e-17
useless_fact   -1.850372e-17
Time           -1.924387e-17
```

Fig 23 : Moyenne des variables explicatives du modèle simulé après normalisation.

Dans notre cas les variables ont toutes une moyenne avoisinant 0, ce qui est normal puisque nous avons normalisé les entrées.

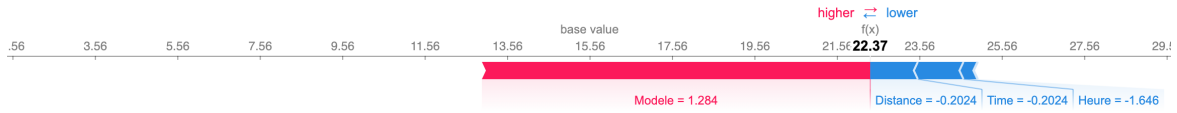


Fig 24 : Individual Force Plot de la 10ème valeur du test set du modèle simulé.

- “Base value” : cela correspond à la valeur qui aurait été prédite si aucune valeur de variable explicative n’avait été renseignée. Il s’agit tout simplement de la valeur moyenne de la sortie pour les instances de test y_test, elle est ici de 15,56.
- f(x) ou “Output value” : cela correspond à la valeur prédite par notre modèle. Ici la valeur est de 22.37 ce qui est plus élevé que la valeur moyenne de 15,56.

- Couleur des variables : En rouge ce sont les variables qui ont contribué à augmenter la valeur de sortie et en bleu celles qui ont contribué à la diminuer.
- Ici “Modele” a une valeur bien supérieure à la moyenne des données (d’autant plus qu’elles ont été normalisées) et participe donc grandement à augmenter la valeur de sortie tandis qu’une valeur de “Distance” plus faible que la moyenne tend à réduire cette sortie.

On peut également représenter les informations de la manière suivante avec un *decision plot* [Fig 25] :

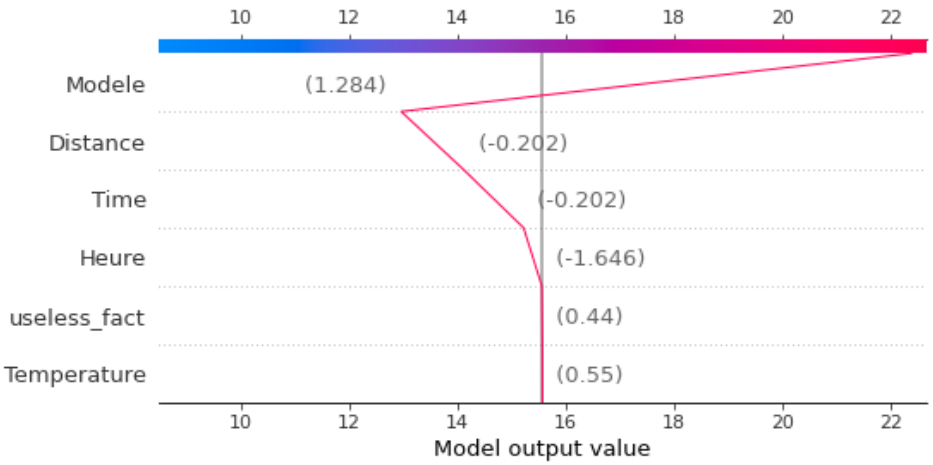


Fig 24 : Decision Plot de la 10ème valeur du test set du modèle simulé.

e. Avantages de SHAP

- SHAP a une base théorique solide sur la théorie des jeux.
- On obtient des explications contrastées entre la prédiction moyenne et la prédiction.
- Utilise les valeurs de LIME et de Shapley, utile pour comprendre les deux méthodes.
- Bonne interprétation globale du modèle.

f. Inconvénients de SHAP

- KernelSHAP bien qu’universel est très lent.
- Comme la plupart des méthodes basées sur la permutation, KernelSHAP ignore la dépendance des fonctionnalités.

4. Résultats et interprétation

a) Émissions de CO_2 et caractéristiques des véhicules

Essayons maintenant d'appliquer les méthodes vues précédemment sur un dataset concret. Compte tenu de notre difficulté à trouver des données correspondant exactement au problème soulevé, nous essayons de changer certaines parties simulées de l'exemple de base par des données réelles. Par exemple, nous allons essayer dans un premier d'analyser l'impact du modèle de la voiture sur les émissions de CO_2 pour pouvoir à terme remplacer tous ces facteurs simulés par des données réelles.

Nous utilisons pour cela le jeu de données trouvé sur Kaggle qui est extrait des données du gouvernement canadien. Il recense de très nombreux modèles de véhicules et leurs caractéristiques ainsi que leurs émissions de CO_2. On y retrouve les cylindrées, le type de carburant utilisé ou encore la consommation de carburant en ville et sur autoroute.
Certains graphiques de l'exploration des données sont disponibles en annexes. Nous pensons également séparer le code en plusieurs parties dont une dédiée à la visualisation des données.

Nous avons pu constater que le jeu de données portait principalement sur des véhicules américains [Annexe 2], moins représentatifs du parc automobile français et de surcroît plus polluant. Ce sera cependant un bon exercice pour tenter d'estimer les émissions de CO_2 depuis les caractéristiques techniques des véhicules français.

Le travail s'est décomposé en plusieurs étapes :

- La visualisation des données [Annexes 2,3,4,5,6,7] : Nous avons dans un premier temps exploré nos données pour mieux les appréhender. C'est également ici que nous avons pu vérifier qu'il n'y ait pas de données manquantes.

- Le nettoyage des données et la mise en forme : Nous nous sommes séparés des colonnes intéressantes comme la consommation combinée en miles par gallon, redondante avec celle en litres pour cent kilomètres. Nous avons également créé des sous catégories de consommation de carburant ou de taille de moteur, le modèle appréhendant mieux les catégories que les données continues. Et nous avons encodé les types de fuels, les sous catégories et les types de transmissions en valeurs numériques.

- L'entraînement du modèle [Fig 25] : Nous avons utilisé un réseau de neurones multi couches et avons fait varier les hyper paramètres avec la méthode Grid Search décrite précédemment mais nous aurions pu opter pour d'autres algorithmes, nous en essaierons d'autres si le temps le permet.

Avec une répartition train set/test set de 80/20% nous obtenons les résultats suivants sur nos métriques d'évaluation :

```
RMSE avec GridSearch
Root mean squared test error = 5.587107684030484
Mean Absolute test error = 4.306735783011713
R2 test error = 0.990954380215493
```

Fig 25 : Résultats des métriques du modèle entraîné sur les véhicules du Canada.

La moyenne d'émission de CO_2 des véhicules du dataset étant de 250 g/km les résultats obtenus sont satisfaisants.

- Analyse des features [Annexes 8,9,10,11,12] : Nous avons appliqué les méthodes décrites précédemment pour évaluer l'importance des différentes features. Les résultats sont disponibles sur le notebook. Sans grande surprise la consommation du véhicule est le facteur le plus important suivi d'assez loin par le type de fuel et la taille du moteur. Le type de transmission par exemple est quasiment négligeable pour cette étude.

- Test et mise en pratique : Afin d'évaluer la pertinence de notre modèle et d'essayer de l'exploiter nous avons décidé de rentrer les caractéristiques de plusieurs véhicules français afin de comparer la sortie de l'algorithme avec les données annoncées par le constructeur. Nous obtenons ainsi une émission de 123 g/km pour un Renault Scénic au lieu de 114 g/km annoncé et 138 g/km contre 135 g/km pour une Renault Twingo. Ces résultats sont satisfaisants mais sûrement améliorables. Les données à tester sont extraites d'un fichier CSV puis les résultats sont exportés dans un autre fichier CSV.

Conclusion :

Le travail réalisé sur ce dataset a été un bon moyen de mettre en pratique ce que nous avons appris sur les différentes méthodes décrites précédemment. De plus, cela s'inscrit dans le thème du projet et permet d'améliorer notre modèle simulé. Nous avons en effet approfondi le facteur modèle du véhicule et avons une meilleure idée de son impact sur l'émission de CO_2 d'un trajet en voiture. Nous avons pu constater cependant que consommation de carburant et émission de CO_2 étaient fortement corrélés et il serait peut être intéressant par la suite d'étudier plutôt le lien entre les caractéristiques d'un véhicule et sa consommation de carburant. En effet, un modèle capable de prédire la consommation d'un nouveau véhicule en fonction de ses caractéristiques semble être un outil intéressant. Nous n'avons également pas étudié tous les types de fuel ou bien les voitures électriques par

exemple. Il semble cependant compliqué d'intégrer les véhicules électriques à notre modèle compte tenu de la trop grande différence techniques avec les véhicules thermiques et du nombre important de facteurs polluants intervenant avant même la mise en circulation de ces derniers.

Le jeu de données utilisé pour l'entraînement de l'algorithme, l'algorithme entraîné ainsi qu'un notebook reprenant toutes les démarches depuis la visualisation des données jusqu'à l'analyse des features sont disponibles dans le livrable.

Les projets Python n'étant pas idéalement conçu pour produire des fichiers exécutables nous avons créé 4 programmes Python :

- Un programme pour visualiser et explorer les données qui exportent les graphes dans un dossier "visualisation".
- Un programme pour entraîner un modèle à partir des données du dataset et qui exporte le modèle entraîné dans le dossier "model"; les encodeurs et autres outils de processing des données sont enregistrés grâce au module Pickle dans le dossier "preprocessing".
- Un programme pour faire des prédictions depuis une liste de nouveaux véhicules "test.csv" vers un fichier "resultats.csv" contenant leurs émissions de CO_2 à l'aide du modèle entraîné.
- Un programme d'analyse de l'importance des features qui exporte les graphes dans un dossier "features".

De plus, un log de chaque programme au format texte est disponible.

b) Estimation du taux de congestion du trafic sur un trajet.

Dans le cadre de l'estimation de l'empreinte carbone d'un trajet en voiture, un des sujets importants est la congestion des voies dans les villes. En effet les bouchons des heures de points augmentent les durées de trajet mais aussi les émissions pour un même trajet.

Nous avons donc cherché à obtenir un modèle à partir des données de la ville de Paris sur la congestion des axes principaux. L'idée était ici d'avoir un modèle pouvant donner la congestion de différents zones en fonction de l'heure et du jour de l'année et de la semaine afin de pouvoir ensuite estimer par exemple les chances de traverser des zones de congestion lors d'un trajet et par exemple choisir d'autres chemins, heures ou même par exemple de proposer à l'employé de télétravailler.

Les données de donnent les taux d'occupation des capteurs (le nombre de voitures se trouvant en moyenne devant un capteur pendant un intervalle d'une heure) et les débits horaires. En combinant ces deux indicateurs on obtient un facteur de congestion :

$0\% \leq K < 15\%$	Fluide
$15\% \leq K < 30\%$	Pré-saturé
$30\% \leq K < 50\%$	Saturé
$50\% \leq K$	Bloqué

Fig 26 : Tableau explicatif des facteurs de congestion de Paris.

C'est celui-ci que nous sommes venu estimer à partir des données de Paris.
Le premier modèle entraîné prend en entrée :

- Le jour de la semaine (de 0 à 6)
- le jour de l'année (de 0 à 364)
- L'heure (0 à 23)

L'idée était ici de donner en entrée les paramètres des principaux cycles des phénomènes de congestion : Les cycles journalier et hebdomadaire. Le dernier paramètre, celui du jour de l'année visait à nous permettre de voir s' il y avait en plus de fluctuations générales au cours de l'année ou juste des points particuliers par exemple sur les jours fériés ou les vacances.
Au vu des résultats du premier modèle, il semble que les cycles hebdomadaires et journaliers soient les plus importants sur les variations du trafic, il n'y a à priori pas de fluctuations annuelles.

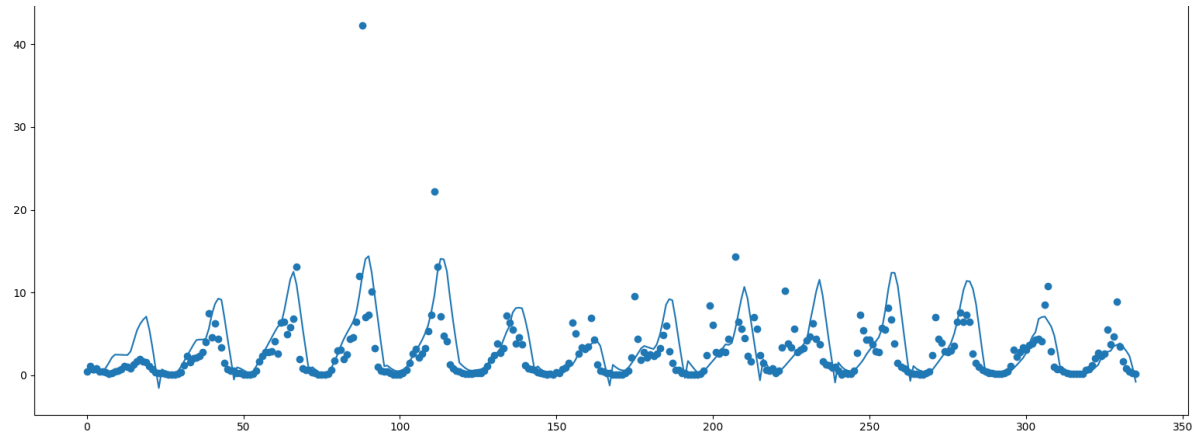


Fig 27: Estimation du taux de congestion sur les 2 premières semaines de 2018 (tracé) comparé aux vraies données (points)

Le modèle n'a pas été poussé plus loin pour le moment car il s'agissait principalement d'une piste explorée dans l'optique d'obtenir au final la consommation d'essence puis l'empreinte carbone. En effet, en connaissant la vitesse d'un véhicule on peut avoir sa consommation instantanée et donc ses émissions.

La finalité de ce modèle serait donc de fournir la vitesse de circulation sur un axe à tout moment, on pourrait ainsi intégrer cette la consommation en fonction de la vitesse sur le trajet et arriver aux émissions globales d'un trajet.

Le modèle reste cependant simple et peut être complexifié grandement afin de prendre en compte plus d'informations : il serait intéressant de prendre en compte l'état de congestion d'autres arcs au même moment et avant ainsi que les vitesses de circulation. De plus on pourrait ensuite chercher à lier les taux de congestion à la qualité de l'air ou à la quantité de bruit généré ce qui fournirait alors un outil pour appuyer les décisions des villes quand à l'ouverture et la fermeture de certains arcs, l'application de limitations de vitesses et autres outils disponibles.

c) Déploiement des modèles

Comme précisé précédemment nous ne ferons pas de fichier exécutable à partir du projet car Python ne s'y prête pas vraiment. De plus, l'utilisation des fichiers python permettra une meilleure lisibilité, réutilisation et adaptabilité du code.

Afin de faciliter la réutilisation du projet par les prochaines équipes nous fournissons un fichier "requirements.txt" contenant la liste des librairies nécessaires et permettant de paramétrer un environnement Python en installant celles-ci directement depuis le fichier.

Nous mettrons aussi à disposition des notebook sous format .ipynb reprenant le travail effectué.

Le tout sera ensuite compressé et envoyé.

D. Conclusion

Le travail accompli sur le dataset des modèles de véhicules et sur celui de la congestion du trafic sur Paris nous ont permis d'une part de mettre en pratique les méthodes apprises et d'autre part d'apporter une base pour l'élaboration d'un modèle plus complexe d'estimation d'empreinte carbone d'un trajet et de réfléchir à leurs applications comme outils de décision. A termes cela pourrait s'inscrire dans des projets de plus grandes envergures comme par exemple l'étude de l'impact du covoiturage dans le cadre d'une Green City. Il resterait à compléter l'étude sur d'autres modèles pour étendre son champ d'application. L'utilisation d'autres algorithmes avec un panel plus large de paramètres à tester permettrait sûrement d'affiner les résultats. On peut également chercher à améliorer l'affinage de l'encodage des données qui reste assez peu précis actuellement en augmentant par exemple le nombre de sous-catégories de consommation de carburant mais cela risque d'augmenter la complexité du modèle.

V. BILAN DU PROJET

A. Apports individuels et collectifs

Techniquement, l'un des grands apports de ce projet est la mise en application de nos connaissances sur l'apprentissage artificiel et l'intelligence artificielle mais d'en plus d'aller plus loin en manipulant concrètement certains paramètres empiriques. Nous avons également pu apprendre beaucoup de choses concrètes sur des parties non abordées en cours comme le nettoyage et la mise en forme des données ou bien la visualisation préliminaire de celles-ci. Les difficultés rencontrées nous ont permis d'acquérir de l'expérience et de pouvoir faire face à un plus large panel de situations.

Sur le plan professionnel, le projet nous a beaucoup apporté. Grâce à notre tuteur de stage, nous avons pu appréhender la gestion, le développement et le travail dans un projet d'entreprise. Comme, nous l'avons rapporté, le sujet a été modifié et retravaillé pour pouvoir être réalisable et concret. Nous avons pu utiliser aussi les méthodes de management apprises au cours de la scolarité et elles sont même devenues des automatismes au sein de l'équipe. Nous avons dû nous organiser avec rigueur pour remplir chacun nos tâches respectives afin de respecter les délais tout en restant investi dans notre travail au quotidien. Sans une bonne gestion et prévision de notre temps, il aurait été difficile d'être efficace dans notre action et de la mener à bien d'un bout à l'autre, avec ses contraintes et ses difficultés. Nous avons dû apprendre également à faire des compromis puisque nos points de vue, quelquefois divergents, nous ont permis d'améliorer nos capacité de travail en équipe. Ce projet nous a permis dans un autre temps de développer notre esprit d'initiative et notre sens des responsabilités. Le fait de remplir indépendamment nos fonctions à accentuer notre réactivité face aux imprévus.

En conclusion, chacun d'entre nous a pu mettre en avant ses compétences techniques sur ce projet riche en enseignement sur le monde du développement logiciel.

B. Conclusion générale sur le déroulement du projet

Au cours de ce projet, nous avons pu accroître une multitude de nos compétences car ce projet a regroupé plusieurs domaines :

- Professionnel avec, par exemple, la gestion de la relation client pour comprendre le produit final attendu par Altran.
- Informatique avec, par exemple, la conception d'un programme basé sur l'intelligence artificielle.
- Management avec, par exemple, l'utilisation de parties des méthodes AGILE et SCRUM.

Ce projet a été l'occasion d'éprouver nos capacités à nous organiser et à travailler en groupe pour que chacun puisse au mieux faire valoir sa technique, donner son avis et enrichir son panel de compétences.

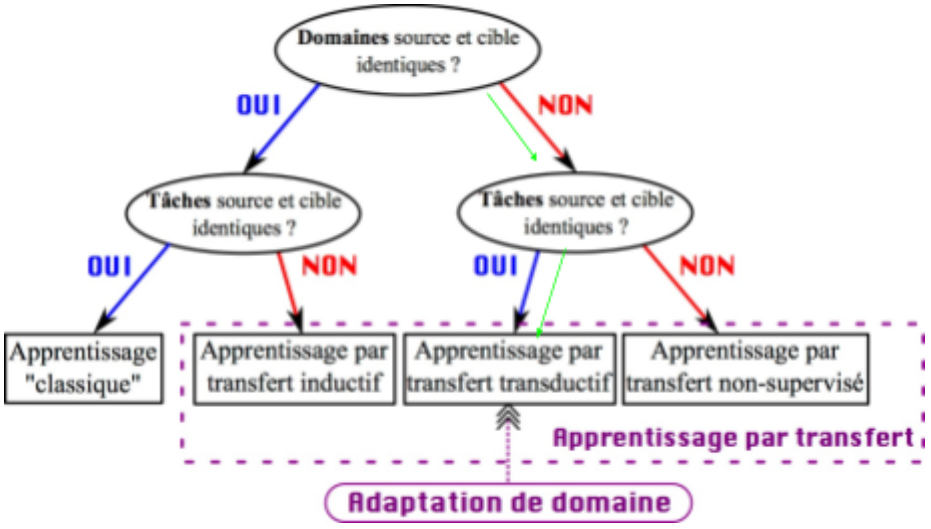
La courte durée du projet et le sujet large nous ont empêché d'approfondir certaines pistes intéressantes mais ceci nous à aussi amener à faire preuve de flexibilité et à en envisager de nouvelles qui n'auraient pas été considérées sinon. Nous avons également pu constater

l’importance des travaux préliminaires au projet que ce soit par la rédaction d’un état de l’art ou la mise en place d’un backlog avec le client/tuteur de projet. Nous aurions dû accorder du temps supplémentaire au deuxième point afin de moins nous égarer durant le projet.
Nous tenons aussi à remercier M. Dolmière à l’origine du projet et qui nous a apporté son expertise au cours de celui-ci.

VI. PERSPECTIVES

Dès le début, la perspective de ce projet de synthèse était de prendre part à un plus grand projet au sein d'Altran : la Smart City. Aujourd'hui, il y a 3 principales perspectives du projet :

- La première reste au sein du trafic routier. Le projet peut être embarqué dans un système qui optimiserait les horaires de travail et de télétravail pour réduire la pollution liée au transport. Il reprendrait tous les calculs et prédictions faites sur les données de congestions du trafic parisien. Une autre avancée peut être, l'impact du covoiturage et si oui avec quelle voiture. Vaut-il mieux être 8 dans un van plus polluant et avec plus de détours ou simplement 4 dans une plus petite voiture ? Cette partie utiliserait les estimations sur les différentes marques et les différents modèles de véhicules.
- Comme nous avons déjà parlé, notre tuteur de projet Thierry Dolmière participe à un projet Kaggle sur le thème d'une course de bateau devant parcourir une certaine distance avec une réserve d'énergie limitée. Pour ce challenge, notre projet servirait à transférer les algorithmes que nous avons utilisés pour faire de la prédiction de consommation d'énergie du bateau en fonction de la météo, de la trajectoire et des courants par exemple.
- La dernière perspective servirait à la deuxième : le transfer learning (apprentissage par transfert). Nous avions dit que l'un de nos objectifs secondaires était d'avoir un code "adaptable" comme il est possible de faire avec la reconnaissance d'objets. Le transfer learning consiste à utiliser notre modèle déjà entraîné au lieu d'en refaire un autre. Il est utilisé sur des problèmes avec beaucoup de calculs ou de données pour gagner du temps. Le principe est de trouver des similitudes entre les deux sujets. Par exemple, le transfer learning sur de notre projet peut être de l'utiliser sur une autre ville que Paris. Les intérêts sont multiples comme forcément un gain de temps de calcul mais aussi de temps de développement, il y a aussi de connaître une majoration de l'erreur sur la cible si l'on connaît bien la source et l'erreur. Concrètement, pour le traitement d'image, il n'y a qu'une seule étape qui change, entre la reconnaissance de deux objets différents, l'interprétation mais la partie détection de forme est la même par exemple. Nous avons fait des recherches sur ce mode d'apprentissage, il y en a 4 types :



Dans le cadre de notre projet, si nous voulons l'appliquer à celui du bateau, nous utiliserons, l'apprentissage par transfert non supervisé. Il y n'a pas le même domaine source mais cependant les tâches restent les mêmes : estimer une consommation. Pour finir sur cette perspective, le transfer learning est limité par la qualité des données sources et la complexité des espaces sources et cibles.

VII. BIBLIOGRAPHIE

A. Pour l'état de l'art

"Analysis of Energy Consumption of Software Development Process Entities"
https://www.mdpi.com/2079-9292/9/10/1678/pdf

Ernest Orlando Lawrence berkeley National Laboratory : "United States Data Center Energy Usage Report 2016"
https://eta-publications.lbl.gov/sites/default/files/lbnl-1005775_v2.pdf
https://dl.acm.org/doi/10.1145/3337773

David Mytton : "How much energy do data centers use?"
https://davidmytton.blog/how-much-energy-do-data-centers-use/

Uptime institute : 2019 Data Center Industry Survey
https://uptimeinstitute.com/resources/asset/2019-data-center-industry-survey

TICE-education : Consommation d'énergie et numérique
https://www.tice-education.fr/index.php/tous-les-articles-er-ressources/developpement-durable/1223-consommation-d-energie-et-numerique

Analyse de l'empreinte carbone des visioconférences sur Greenspector:
https://greenspector.com/fr/quelle-application-mobile-de-visioconference-pour-reduire-votre-impact/#:~:text=En%20visioconf%C3%A9rence%20%3A%20180*1*,one%2Dto%2Done%20physique

Consommation des données en fonction de l'utilisation :
https://anticip.paritel.fr/comprendre/mobile-enveloppe-data-choisir-selon-usages-pros/

Marie-Anne Paveau, "1 Mo = 15 g de CO_2. L'impact écologique du courrier électronique," in Technologies discursives, 15/06/2019 :
https://technodiscours.hypotheses.org/1103.

Bilan énergétique d'un lieu de travail :
https://energieplus-lesite.be/evaluer/energie-et-les-consommations2/evaluer-les-consommations-d-un-local-de-bureau/

B. Pour le sujet de l'émission de CO_2 des trajets en voiture

Dataset des émissions de CO2 des voitures du Canada :
https://www.kaggle.com/debajyotipodder/co2-emission-by-vehicles

3 Ways to Encode Categorical Variables for Deep Learning by Jason Brownlee on November 22, 2019 in Deep Learning
https://machinelearningmastery.com/how-to-prepare-categorical-data-for-deep-learning-in-python/

Données des comptages routiers de Paris :
https://www.data.gouv.fr/en/datasets/comptage-routier-historique-donnees-trafic-issues-des-capteurs-permanents-1/#_

C. Création du modèle et influence des hyper-paramètres

Hyperparameter Tuning of Keras Deep Learning Model in Python
https://onezero.blog/hyperparameter-tuning-of-keras-deep-learning-model-in-python/

Difference Between a Batch and an Epoch in a Neural Network
by Jason Brownlee on July 20, 2018 in Deep Learning :
https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

Effect of batch size on training dynamics, Kevin Shen, Jun 19, 2018 :
https://medium.com/mini-distill/effect-of-batch-size-on-training-dynamics-21c14f7a716e

Hyperparameters in Machine /Deep Learning, Jorge Leonel, Apr 6, 2019 :
https://medium.com/@jorgesleonel/hyperparameters-in-machine-deep-learning-ca69ad10b981

Optimiser les hyperparamètres du modèle - Azure Machine Learning | Microsoft Docs :
https://docs.microsoft.com/fr-fr/azure/machine-learning/algorithm-module-reference/tune-model-hyperparameters

A Guide for Making Black Box Models Explainable. Christoph Molnar, 2021-01-11 :
https://christophm.github.io/interpretable-ml-book/shap.html

Choosing the Right Metric for Evaluating Machine Learning Models — Part 1, Alvira Swalin, Apr 7, 2018 :
https://medium.com/usf-msds/choosing-the-right-metric-for-machine-learning-models-part-1-a99d7d7414e4

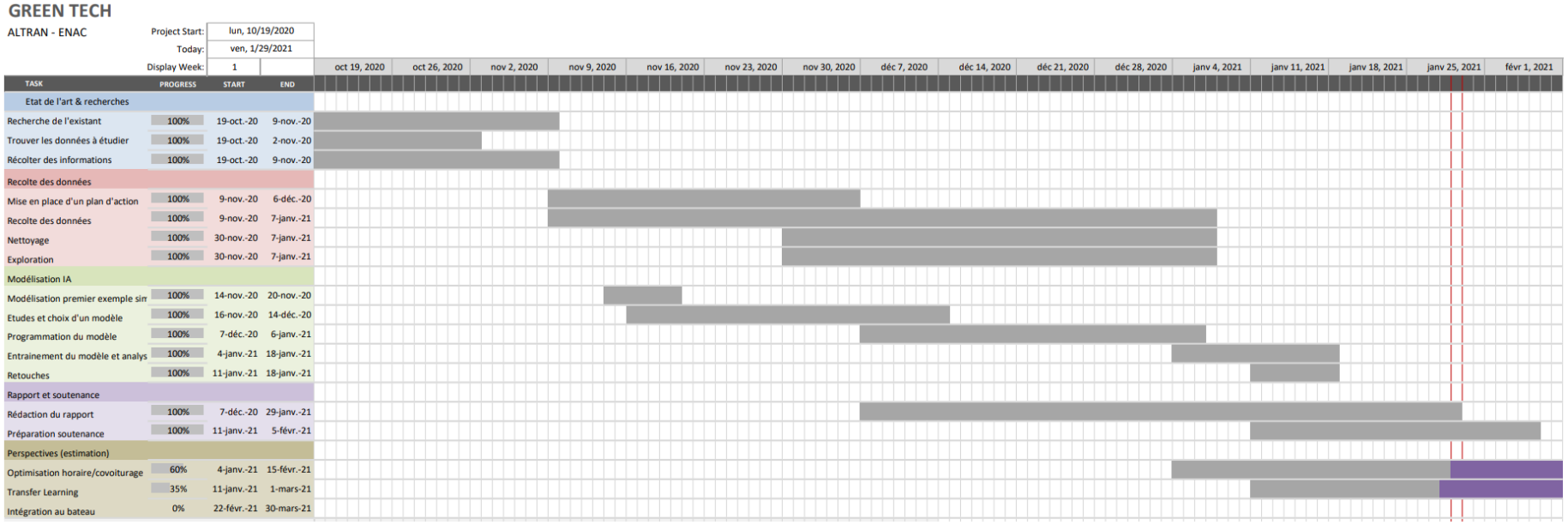
PCA using Python (scikit-learn), Michael Galarnyk, Dec 5, 2017 :
https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60

Valeur de Shapley :

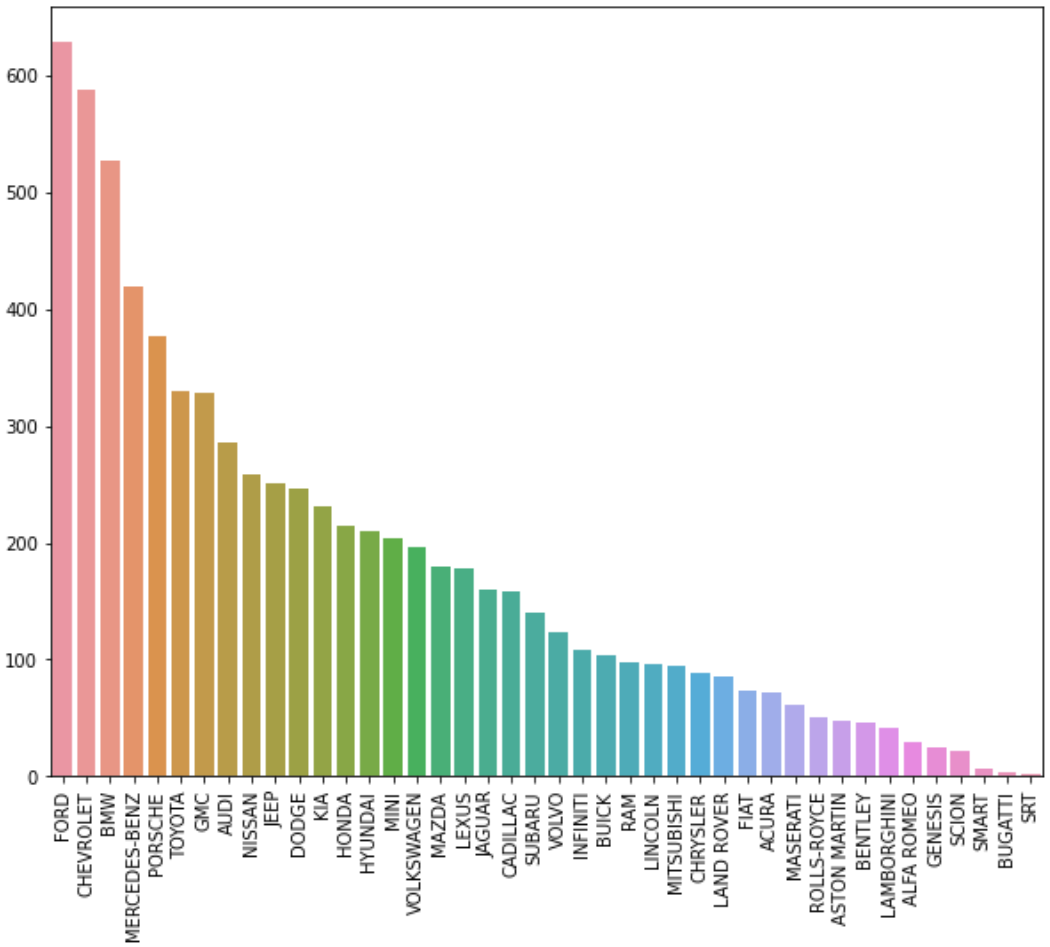
L. S. Shapley, 17. A Value for n-Person Games *:*
https://www.degruyter.com/princetonup/view/book/9781400881970/10.1515/9781400881970-018.xml

VIII. ANNEXES

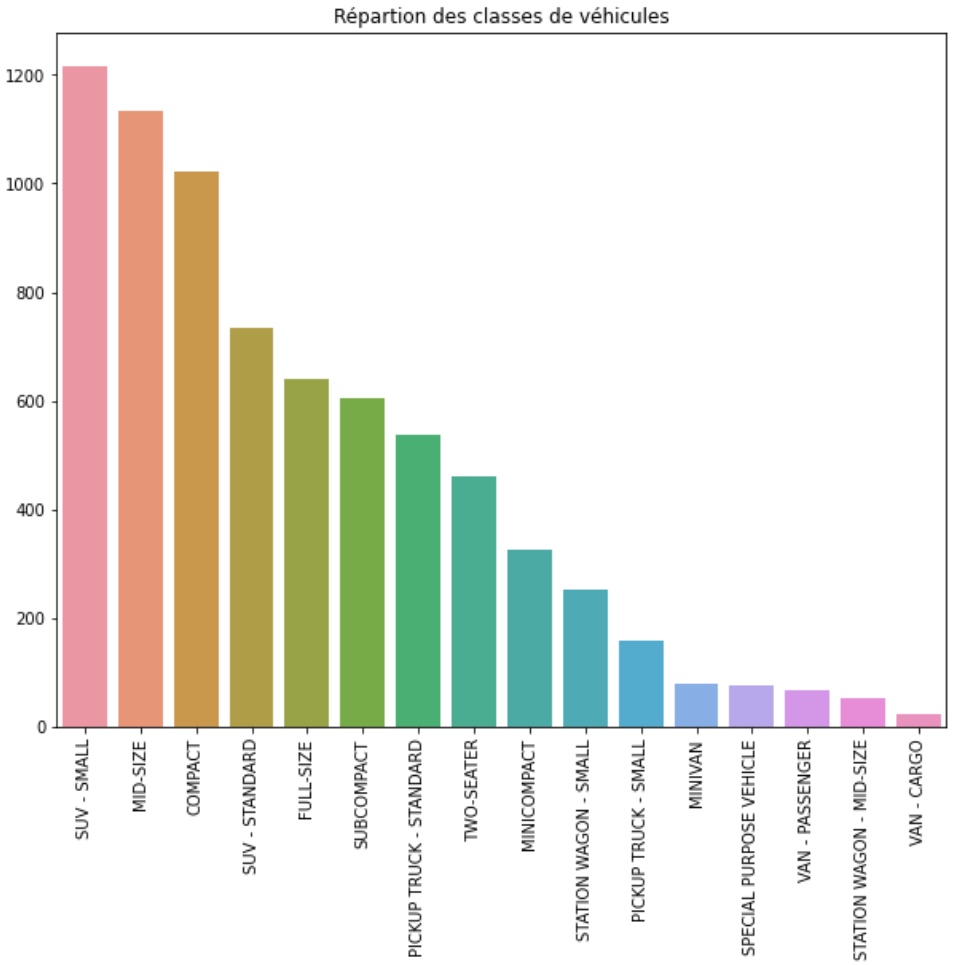
Management de projet



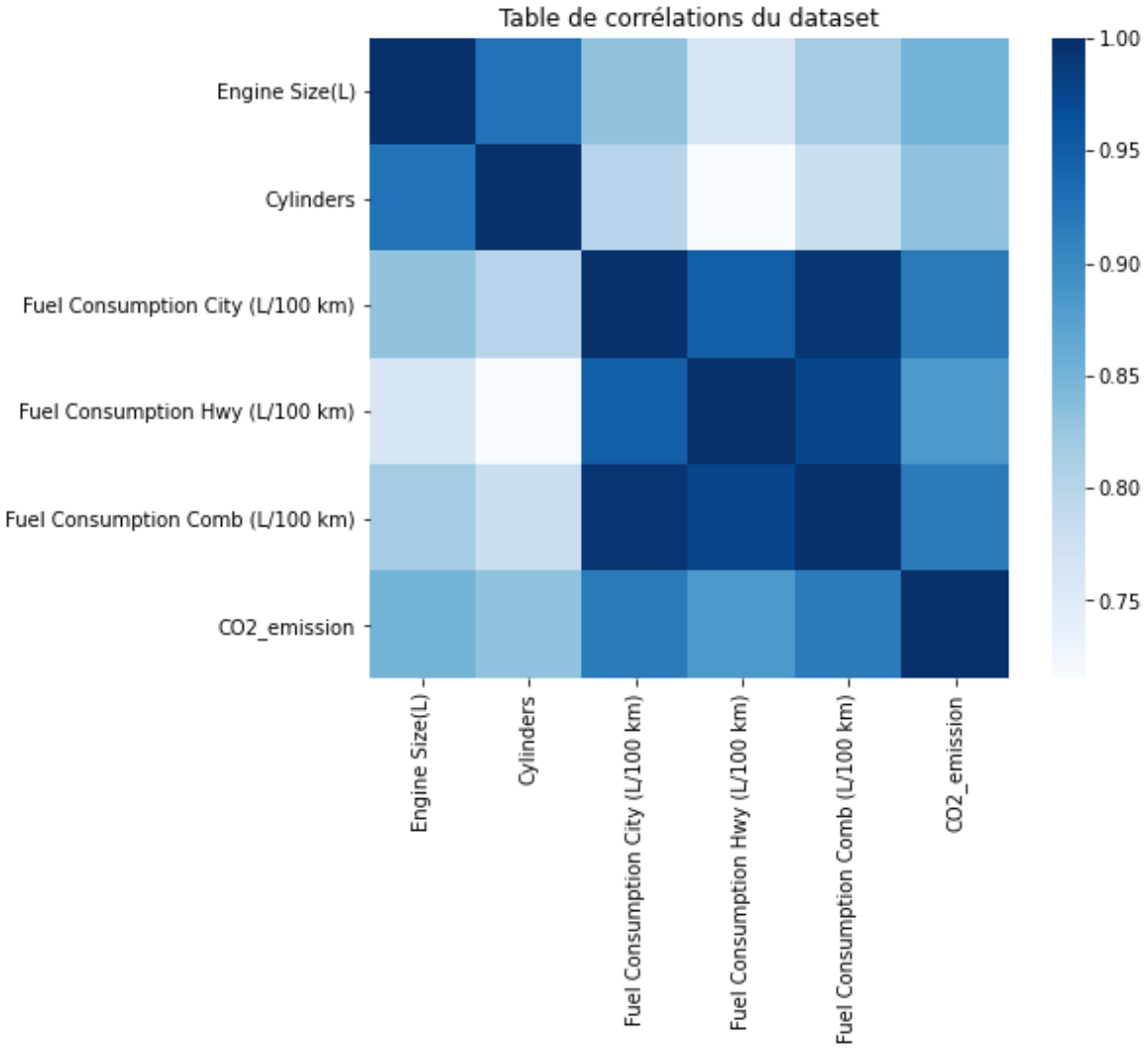
Annexe 1 : Gantt Chart du projet.



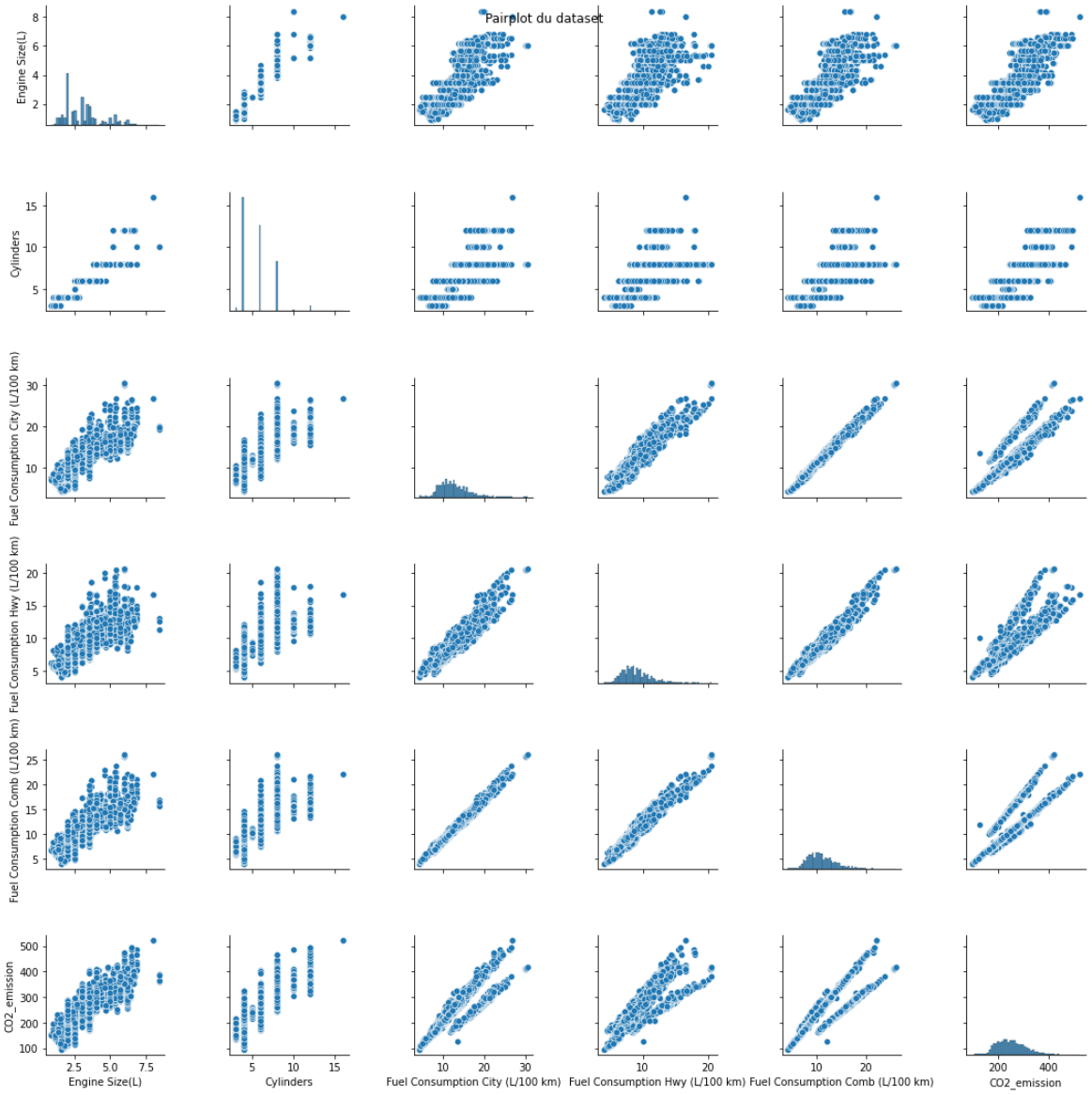
Annexe 2 : Répartition des marques de voitures du dataset du Canada.



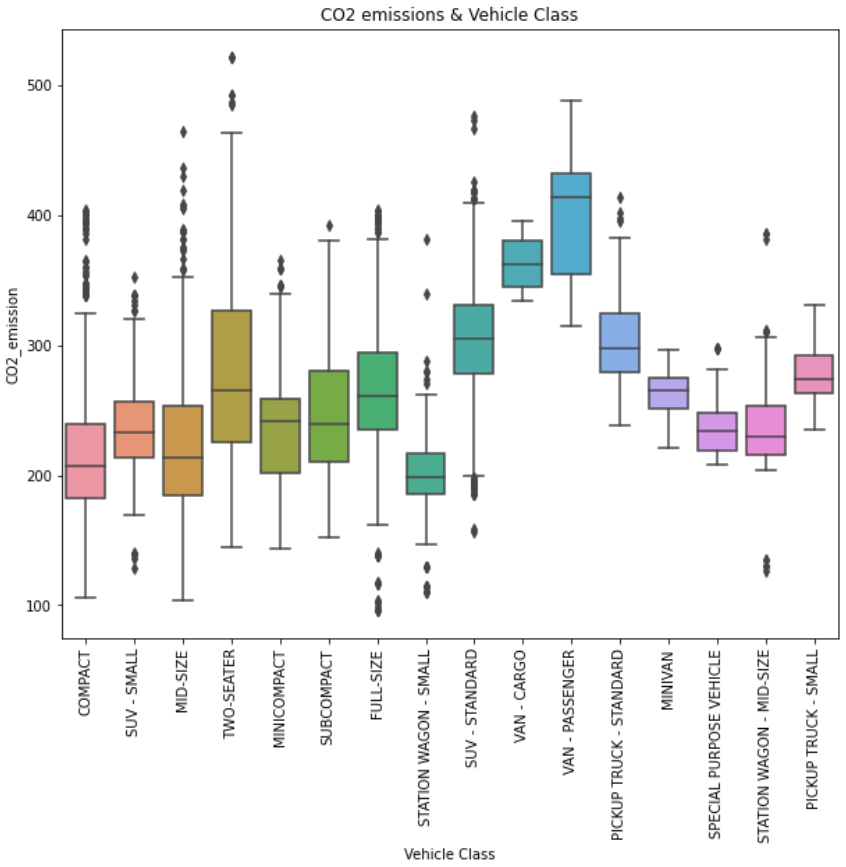
Annexe 3 : Répartition des classes de voitures du dataset du Canada.



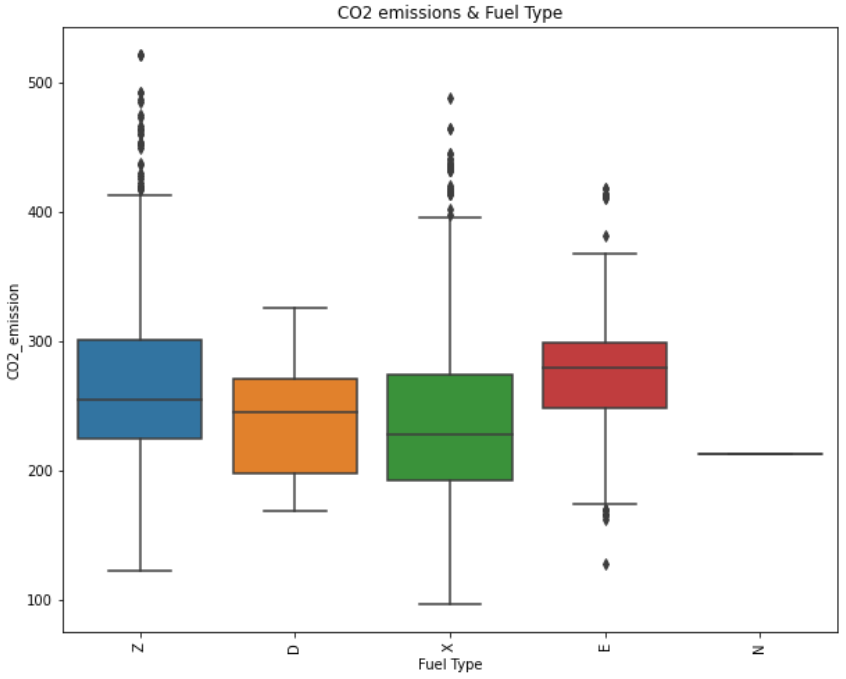
Annexe 4 : Table des corrélation des données.



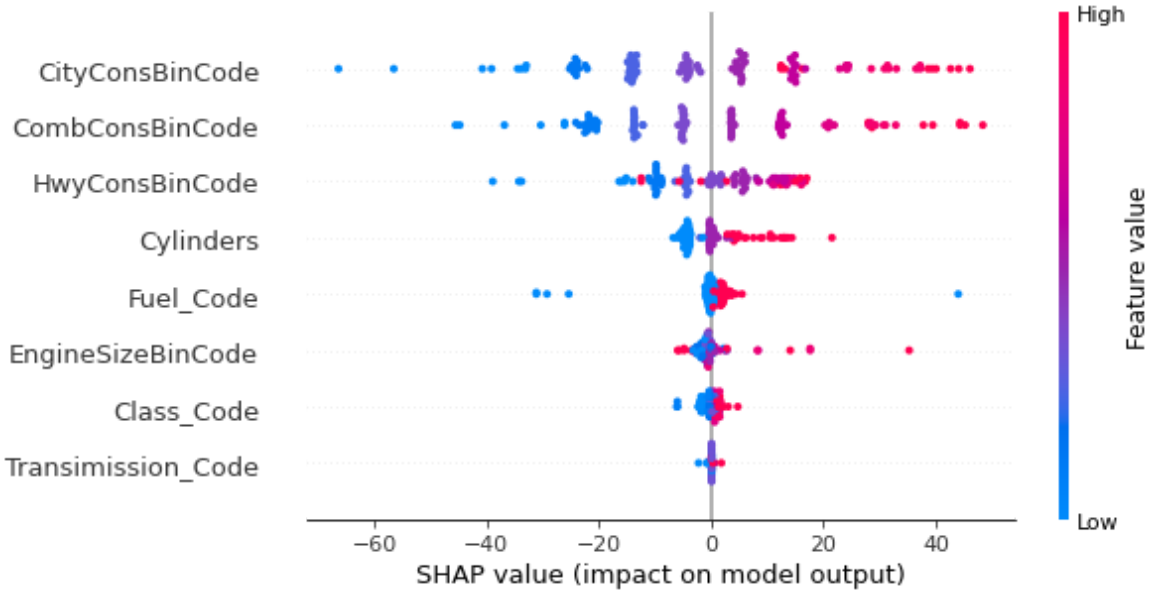
Annexe 5 : Pairplot des données du dataset.



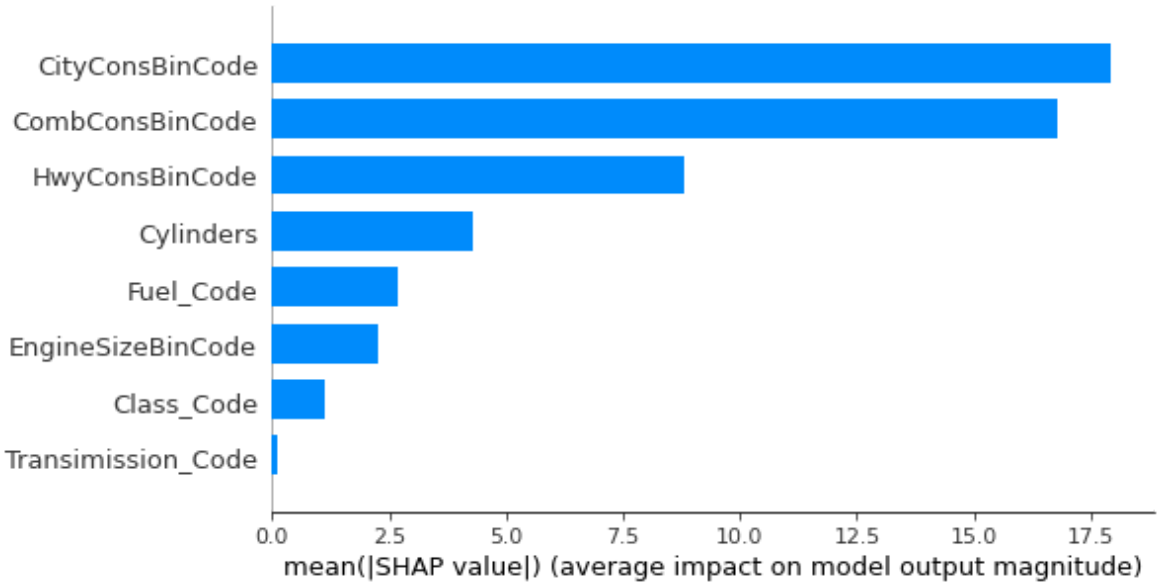
Annexe 6 : Emissions de CO2 et classes de véhicules.



Annexe 7 : Emissions de CO2 et type de fuel.



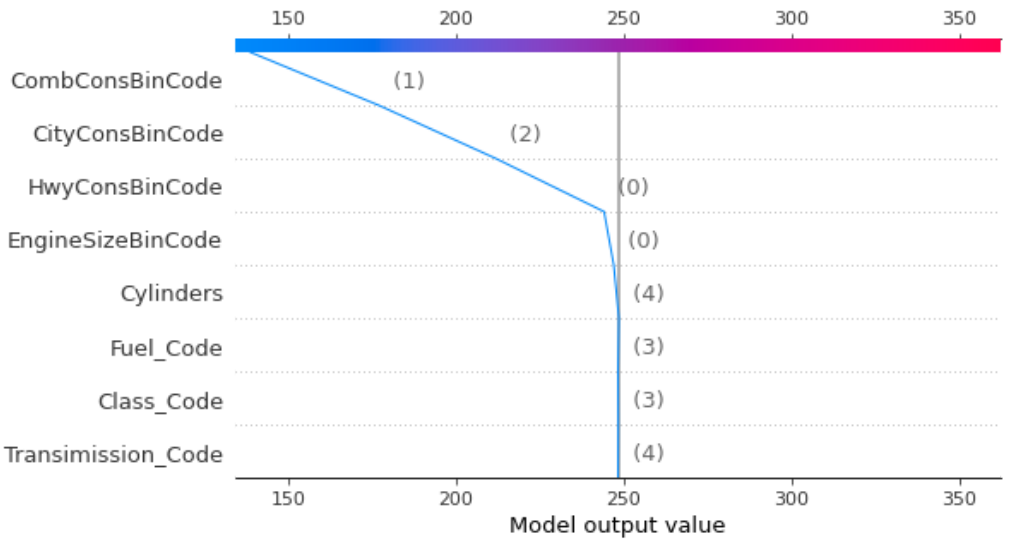
Annexe 8 : Summary plot du dataset avec encodage des données.



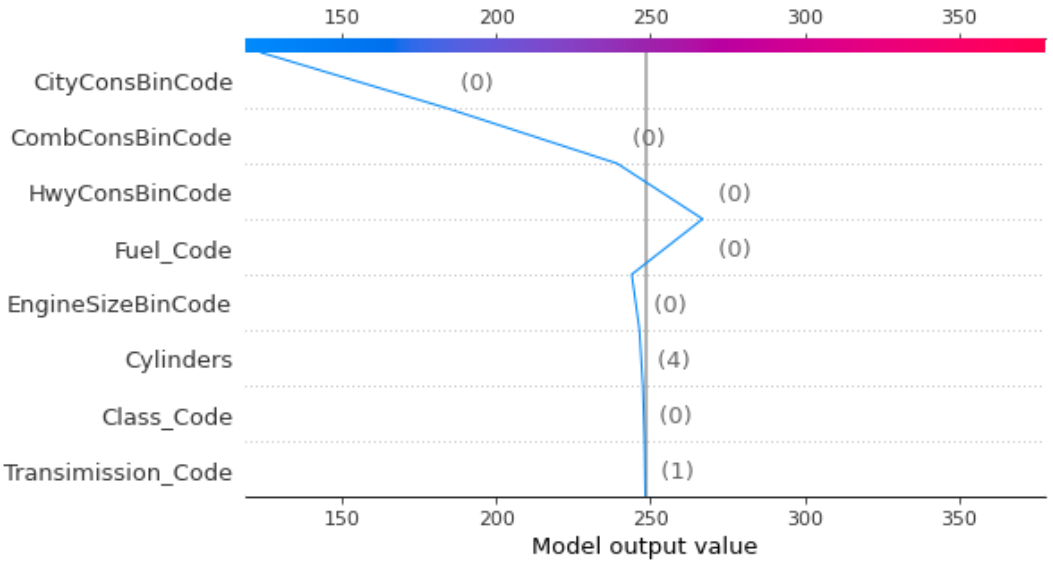
Annexe 9 : Summary bar plot du dataset avec encodage des données.

Weight	Feature
0.3750 ± 0.0056	CityConsBinCode
0.2672 ± 0.0038	CombConsBinCode
0.1737 ± 0.0077	HwyConsBinCode
0.1408 ± 0.0054	Fuel_Code
0.0954 ± 0.0162	EngineSizeBinCode
0.0363 ± 0.0051	Cylinders
0.0112 ± 0.0018	Class_Code
0.0004 ± 0.0001	Transimission_Code

Annexe 10 : Importance des features d'après le module Eli5.



Annexe 11 : Décision plot de la prédiction de CO2 de la RENAULT Twingo.



Annexe 12 : Décision plot de la prédiction de CO2 du RENAULT Scenic.