ACCIDENTS ROUTIERS EN FRANCE

Projet de Data Science - Octobre continu 2024

Résumé

L'objectif de ce projet est d'essayer de prédire la gravité des accidents routiers en France. Les prédictions seront basées sur les données historiques répertoriées sur le site du gouvernement.

Après avoir appliqué différentes méthodes pour nettoyer le jeu de données et extrait les caractéristiques qui nous semblaient être pertinentes pour estimer la gravité des accidents, nous avons créé un modèle de Machine Learning pour essayer de prédire la gravité des accidents routiers en France.

Problématique

Les variables que nous possédons dans notre jeu de données sont des variables catégorielles et nous cherchons à déterminer la gravité des accidents (variable cible) selon un classement qui est le suivant : blesser léger, indemne, hospitalisé et tué.

Nous cherchons donc à classer si les accidents de la route appartiennent à une de ces classes. Ainsi, compte tenu de ces éléments, il semblerait que nous cherchons à traiter une problématique de classification supervisée.

Hayate Makhfi
Hélène Deraison Lambert
Joseph Tiye
Guillaume Lavazanian

Sommaire

PARTIE 1 :
RAPPORT D'EXPLORATION, DE DATA VISUALISATION ET DE PREPROCESSING DES DONNÉES

I. Introduction au projet

1. Contexte

Pour Hayate : ce projet s'inscrit dans un contexte de reconversion professionnelle. Après avoir réalisé un bilan de compétences durant l'année 2023, je souhaiterais occuper un poste de *Data Scientist* au sein de mon entreprise (banque/assurance) à travers une mobilité interne. Ce sujet est intéressant car il me permet, d'une part, de me confronter à des problématiques que je pourrai rencontrer dans le cadre de mon futur métier au sein de mon entreprise. Et d'autre part, de mettre en pratique les compétences techniques acquises lors de cette formation.

Pour Guillaume : ce sujet n'entre pas dans le cadre de mon travail. Cependant je peux retrouver des similitudes par rapport aux projections, que met en place l'équipe de *Data Scientist* et *Data Engineer* avec qui je travaille, des commandes de pièces neuves pour les hélicoptères.

J'ai choisi ce sujet pour une démarche personnelle, puisque je suis très curieux et je conduis un deux roues. Ce sujet me donnera les connaissances nécessaires pour discuter des accidents de la route en France avec mon entourage.

Pour Joseph : ce projet n'entre pas forcément en lien avec mon métier. Cependant, ce sujet est intéressant dans la mesure où il se rapproche d'un sujet de modélisation de risque de crédit (exemple : un modèle pour accorder un crédit ou non à un particulier, ou à une entreprise en fonction de plusieurs variables et d'un large historique).

Pour Hélène : actuellement en poste dans le milieu de l'assurance, je réalise des études et des *reportings* de suivis des performances commerciales (ventes des contrats d'assurance, objectifs commerciaux) à destination des directeurs et des managers. Travailler sur ce sujet va me permettre d'élargir mon domaine de compétences en traitant une problématique de sinistralité, sujet crucial pour les compagnies d'assurance.

D'un point de vue économique, ce sujet se réfère au ratio sinistralité sur cotisation (s/c). Plus le nombre et la gravité des accidents augmentent, plus il y a de sinistres à payer pour les assureurs et plus les cotisations augmentent. C'est ce que nous observons actuellement avec l'augmentation des cotisations d'assurance habitation et les impacts climatiques qui sont plus fréquents et intenses.

Par ailleurs, le *scoring* est un projet qui se développe fortement au sein de l'entreprise dans laquelle je travaille afin d'optimiser le potentiel d'opportunités commerciales. Aussi, ce sujet va me permettre de mettre en pratique les notions de Machine Learning acquises au cours de la formation *Data Scientist* et de traiter un sujet qui fait sens dans mon parcours professionnel.

2. Objectifs

L'objectif de ce projet est d'essayer de prédire la gravité des accidents routiers en France en mettant en place un modèle de *machine Learning*.

Autrement dit, notre objectif est de prédire les accidents les plus graves (blessé hospitalisé et tué) afin de déterminer leur caractère urgent, réduire le délai d'intervention des secours, et optimiser l'allocation des ressources grâce à des techniques de *machine learning*.

Pour répondre à cet objectif, nous devons, dans un premier temps, compiler différentes sources de données historiques se référant aux caractéristiques des accidents, aux véhicules accidentés, aux usagers et aux lieux des accidents. Puis, dans un second temps, nous devons nettoyer ces données sources en sélectionnant les variables qui nous semblent les plus pertinentes pour modéliser la gravité des accidents routiers.

La majorité des variables étant des variables numériques et qualitatives, nous devrons ainsi effectuer un *preprocessing* des données avant l'étape de modélisation.

Concernant le niveau d'expertise sur la problématique adressée, l'ensemble du groupe ne possède pas de notion de sinistralité sur les accidents de la route.

Il est à noter que, par manque de temps, le groupe n'a pas pu mener d'interview avec des experts métiers pour affiner la problématique et les modèles sous-jacents.

II. Compréhension et manipulation des données

Pour mener à bien ce projet et atteindre les objectifs fixés, nous avons utilisé les bases de données historiques et annuelles des accidents corporels de la circulation routière répertoriées sur le site du gouvernement[1] pour les années 2005 à 2023. Ces données étaient accessibles librement. Nous avons pu ainsi télécharger l'ensemble des fichiers au format CSV.

Pour chaque année, ces fichiers étaient répartis en 4 rubriques : les caractéristiques, les lieux, les usagers et les véhicules. Nous avons retenu l'ensemble des fichiers sur 6 années consécutives, à savoir de 2018 à 2023.

La volumétrie de notre jeu de données sur 6 ans avant nettoyage représentait environ 910 267 lignes et 63 variables (soit 63 colonnes). Ce qui représente un nombre total d'accident égale à 331 009.

Pour notre analyse, nous avons décidé ensemble, de retenir les données sur 6 ans, de 2018 à 2023. Il nous semblait pertinent de faire une analyse à court terme pour plusieurs raisons :

- Obtenir des données plus homogènes (moins de risque de changement de réglementation et donc de méthode de collecte des données). A titre d'exemple, les données sur la qualification de blessé hospitalisé depuis l'année 2018 ne peuvent être comparées aux années précédentes suite à des modifications de méthode de saisie des forces de l'ordre.

- Analyse plus précise car les données à court terme collent à la réalité économique actuelle.

- Modèle de *machine learning* plus précis et pertinent. En effet, une analyse à long terme (10 ans) des données repose sur des données anciennes et peuvent biaiser le modèle avec notamment l'apparition de nouvelles infrastructures (développement accrue des bandes cyclables) ou encore de nouveaux comportements (la pandémie de 2019 a entraîné de nouvelles façons de travailler avec la mise en place du télétravail conduisant ainsi à la réduction de l'usage du véhicule).

[1] Site : Bases de données annuelles des accidents corporels de la circulation routière - Années de 2005 à 2021 - data.gouv.fr

III. Pre-processing et feature engineering

1. Sélection des variables explicatives

Après avoir regroupé l'ensemble des données dans un seul fichier au format CSV, nous avons ensemble procéder à une première sélection des variables pertinentes (variables explicatives) sachant que la variable cible est la gravité des accidents (noté variable « grav »).

Les variables explicatives qui nous semblaient les plus pertinentes de notre jeu de données sont les suivantes :

- **Dans le fichier caractéristique, les variables explicatives retenues sont** :

- Jour = Jour de l'accident
- mois = Mois de l'accident
- an = Année de l'accident
- hrmn = Heure et minutes de l'accident
- lum = Lumière : conditions d'éclairage dans lesquelles l'accident s'est produit
- agg = Localisation Agglomération ou hors agglomération
- int = Type d'intersection
- atm = Conditions atmosphérique au moment de l'accident
- col = Type de collision
- Dep = Département

Nous avons décidé de retirer les variables géographiques telles que : l'adresse et la commune car nous estimons que les données mentionnées dans la colonne « département » étaient suffisantes pour notre analyse. Par ailleurs, la variable département est corrélée à la variable cible selon le test de Chi-Deux. Par conséquent, nous avons choisi de garder cette variable pour notre analyse.

Nous avons également retiré la longitude, la latitude et le GPS car nous avons décidé de restreindre notre jeu de données. En effet, nous n'envisageons pas de réaliser une cartographie. Nos responsabilités respectives au travail et les délais imposés pour la réalisation de ce projet, ne nous permettent pas aujourd'hui de réaliser cette cartographie.

- **Dans le fichier lieux, les variables explicatives retenues sont** :

- Catr = catégorie de route (autoroute, route nationale, …)
- Circ = circulation (à sens unique, chaussée séparée, …)
- surf = état de la surface (mouillée, flaque, inondée, …)
- infra = aménagement - infrastructure
- situ = situation de l'accident

Sur un total de 17 variables, nous avons décidé ensemble de garder uniquement ces 5 variables car celles-ci étaient pour nous les plus pertinentes pour expliquer la gravité de l'accident. Mais aussi parce que certaines variables de ce fichier mettaient en avant des informations similaires à d'autres variables mentionnées dans la catégorie de la route. Il y avait donc une redondance des données. À titre d'exemple : la VMA (vitesse maximale autorisée) et le NBV (nombre total de voies de circulation) sont des variables liées à la catégorie de la route et donnent des informations similaires à celles de la variable « catégorie de la route ».

Dans le fichier usager, les variables explicatives retenues sont :

- Num_Acc = identifiant de l'accident
- id_usager = identifiant unique de l'usager
- id_vehicule = identifiant unique des véhicules
- num_veh = catégorie permis/ catégorie de véhicule
- place = place occupée dans le véhicule
- catu = catégorie d'usager (conducteur, passager, piéton)
- grav = gravité de blessure de l'usager
- sexe = sexe de l'usager (masculin ou féminin)
- an_nais = années de naissance de l'usager
- trajet = motif de déplacement
- secu1 = présence et utilisation d'un équipement de sécurité
- secu 2 = présence et utilisation d'un équipement de sécurité
- secu 3 = présence et utilisation d'un équipement de sécurité

Nous avons retiré les variables ACTP (action du piéton), ETATP (piéton seul ou non) et LOCP (localisation du piéton), car nous avons peu d'accidents impliquant des piétons. En effet, ces accidents représentent un volume de données insuffisant pour être exploité (8 %) comparativement aux accidents de véhicules. Ainsi, nous avons décidé de nous concentrer uniquement sur les accidents de véhicules, c'est-à-dire les usagers situés à l'intérieur d'un véhicule, afin de réduire notre jeu de données.

- **Dans le fichier véhicule, les variables explicatives sont** :

- Num_Acc = identifiant de l'accident
- id_vehicule = identifiant unique des véhicules
- num_veh = identifiant du véhicule repris pour chacun des usagers occupant ce véhicule (y compris les piétons qui sont rattachés aux véhicules que les ont heurtés)
- catv = catégorie de véhicule
- obs = obstacle fixe
- obsm = obstacle mobile heurté
- choc = point de choc initial
- manv = manœuvre principal avant l'accident

La variables OCCUTC (nombre d'occupant dans le transport en commun) a été retirée car elle présente 80 % de valeurs manquantes. Par conséquent, cette variable nous semblait non exploitable. Nous avons retiré également la variable MOTOR (type de motorisation du véhicule : Hybride, Hydrocarbures, Hybride électrique 3, Electrique 4, Hydrogène 5, Humaine 6, Autre). Nous avons choisi de nous concentrer uniquement sur la catégorie de véhicule notée CATV dans notre jeu de données, car cette variable nous semble plus pertinente pour expliquer la gravité des accidents de la route.

Cette première sélection de variables était nécessaire au vu du nombre important de variables que nous possédons dans notre jeu de données (63 variables au total). Cette pré-sélection nous a permis de réduire notre jeu de données à 27 variables.

De manière générale, les variables que nous avons retirées étaient soit non pertinentes pour notre analyse, soit redondantes avec d'autres variables, ou encore peu exploitables en raison d'un nombre trop important de valeurs manquantes. (cf la partie « nettoyage des données »).

Après cette première sélection de variable, nous avons assemblé 24 fichiers au total, que nous avons regroupés dans un seul fichier au format CSV, compilant ainsi l'ensemble des données. Pour regrouper les fichiers, nous avons utilisé la concaténation.

2. Nettoyage des données

Pour commencer notre analyse et obtenir un aperçu de notre jeu de données, nous avons utilisé différentes méthodes.

- La méthode *info* pour obtenir des informations sur le nombre d'entrées, le nombre de variables, le type de variables (numériques, catégoriques, temporelles, …) ou encore pour identifier la présence de valeurs manquantes.
- La méthode *describe* qui dans notre *Dataframe* n'est pas très significative compte tenu de la nature de nos variables. Toutefois, cette méthode nous a permis aussi de détecter les valeurs manquantes et valeurs non renseignées (notées « -1 » dans notre jeu de données).

Ensuite, nous avons nettoyé les données de la manière suivante :

(1) Les valeurs manquantes et valeurs non renseignées (notées « -1 » dans notre jeu de données).

Concernant les valeurs manquantes : elles représentaient un pourcentage peu élevé alors nous avons décidé de les supprimer (suppression de lignes) car nous pensons que cette solution n'aura pas d'impact dans l'analyse de notre jeu de données. Ci-dessous le taux des valeurs manquantes pour chacune des variables gardées :

CARACTERSITQIEUS	
Variable	% valeurs manquantes
Num_Acc	0,0000
an	0,0000
mois	0,0000
jour	0,0000
hrmn	0,0000
lum	0,0000
agg	0,0000
int	0,0000
dep	0,0000
heure	0,0000
jour_semaine	0,0000
col	0,0006
atm	0,0015

LIEUX	
Variable	% valeurs manquantes
Num_Acc	0,00
catr	0,00
circ	0,12
surf	0,13
infra	0,13
situ	0,14

USAGERS	
Variable	% valeurs manquantes
Num_Acc	0,00
place	0,00
catu	0,00
grav	0,00
sexe	0,00
num_veh	0,00
secu_combined	0,00
trajet	0,02
an_nais	0,04

VEHICULES	
Variable	% valeurs manquantes
Num_Acc	0,00
num_veh	0,00
manv	0,02
choc	0,02
obsm	0,02
obs	0,03
catv_modifié	0,76

Concernant les valeurs non renseignées (notées « -1 » dans notre jeu de données) :

L'analyse des fichiers CARACTERISTIQUES, LIEUX, USAGERS et VEHICULES montre la présence de valeurs notées "-1", indiquant des données non renseignées. Leur proportion varie selon les types de fichiers et les variables concernées.

Le fichier CARACTERISTIQUES présente un très faible taux de valeurs "-1" (0,5 %), principalement concentré sur la variable col (0,46 %). Ce qui justifie leur suppression sans impact significatif.

En revanche, le fichier LIEUX affiche une proportion plus importante (6,1 %), notamment sur la variable circ (4,1 %).

Pour le fichier USAGERS, les valeurs "-1" atteignent 1,4 %, avec une forte concentration sur la variable trajet (1,1 %) et la variable sexe (1 %).

Enfin, le fichier VEHICULES présente un taux de valeurs "-1" de 0,09 %, principalement dû à la variable choc (0,04 %).

Ci-dessous le pourcentage de valeurs non renseignées pour chacune de nos variables :

- **variable lum : 0,0014%**
- **variable int : 0,0034%**
- **variable atm : 0,0046%**
- **variable col : 0,46%**
- **variable circ : 4,1%**
- **variable surf : 0,069%**
- **variable infra : 0,72%**
- **variable situ : 0,083%**
- **variable catv : 0,0023%**
- **variable obs : 0,033%**
- **variable obsm : 0,035%**
- **variable choc : 0,04%**
- **variable manv : 0,026%**
- **variable place : 0,003%**
- **variable grav : 0,05%**
- **variable sexe : 1%**
- **variable trajet : 1,1%**

(2) Le traitement de doublons des données.

En effet, lorsque nous croisons les données caractéristiques avec les données usagers et véhicules, puisqu'un accident peut impliquer plusieurs véhicules et plusieurs usagers, cela engendre la création de doublons : l'accident est alors répété autant de fois qu'il y a de véhicules ou d'usagers. Pour éviter la redondance des données, nous avons fait le choix de supprimer les doublons. Il est à noter que nous avons recensé 17 588 lignes de doublons, soit un pourcentage égal à 2,1 %.

(3) La transformation de colonnes.

La variable heure noté « hrmn » ne nous semblait pas interprétable en l'état car elle était à la fois exprimée en heure, minutes et secondes. Nous l'avons donc transformé en ne gardant que les heures. Cette transformation nous permettait de réaliser des graphiques plus lisibles mais aussi de réaliser par la suite un test de corrélation avec la variable cible.

De la même manière, les variables secu1, secu2 et secu3 ont été agrégées en une seule variable (nommé « secu_combined ») utilisable et interprétable car chacune de ces trois variables présentait à elles plus de 20% de valeurs manquantes. Le fait d'avoir agréé ces variables en une seule permettait de réduire le pourcentage de valeur manquante. En effet, on est passé de 20% de valeurs manquantes avant transformation à 0% après transformation. Cette transformation nous a donc permis d'exploiter les données de sécurité.

Nous avons également procédé à une transformation de la variable catégorie du véhicule (catv) notée dans notre jeu de données « catv_modifié ». Cette transformation vise à simplifier la variable, qui représente plusieurs dizaines de catégories de véhicules, en regroupant les différentes valeurs en 10 grandes catégories plus homogènes. Pour ce faire, un dictionnaire de correspondance est utilisé afin d'associer chaque code spécifique de véhicule à une nouvelle catégorie plus générale. Cette transformation facilite l'analyse et l'interprétation des données en réduisant le nombre de modalités, tout en conservant la pertinence des catégories de véhicules.

Enfin, nous avons également transformé la variable « jour » en « jour de semaine » pour une meilleure interprétation des données.

Après la sélection des variables les plus pertinentes selon nous, le nettoyage des données (suppressions des valeurs manquantes et valeurs non renseignées), ainsi que le traitement des doublons, notre jeu de données contient actuellement environ 765 000 lignes et 27 variables. Ce qui représente environ 330 000 accidents de la route.

Il est à noter que ce nettoyage de données et ces transformations permettront d'améliorer la qualité de nos données et de les rendre ainsi exploitables, mais aussi d'optimiser la performance des modèles de *machine learning* choisis, et par conséquent, d'obtenir des prédictions plus précises.

Après cette étape, il nous paraissait important de détecter dès à présent, parmi les 27 variables restantes, lesquelles d'entre elles étaient corrélées à la gravité de l'accident, avant de passer à l'étape suivante, qui est le ré-encodage.

3. Test statistique : test du Chi-Deux

Nous sommes confrontés à des variables catégorielles qui sont présentées sous forme de labels. Nous pouvons donc exclure le test de Pearson car cela ne serait pas interprétable.

En effet, pour estimer la relation entre deux variables catégorielles, il faut réaliser un test de Chi-Deux (*Chi-Square Test*). Le test de Chi-deux est un test statistique utilisé pour évaluer la relation entre deux variables catégorielles dans un ensemble de données. Ce test permet de savoir si deux variables sont indépendantes ou s'il y a une relation significative entre elles.

Pour mesurer cette relation, il faut calculer une probabilité appelée p-value. Cette probabilité détermine ou non si les variables qualitatives sont indépendantes en fonction d'un seuil prédéfini qui est égale à 5%.

L'objectif est de montrer si les variables explicatives retenues sont corrélées à la variable cible (gravité des accidents routiers en France).

Signification et interprétation de p-valeur :

- Si $p < 0.05$, on rejette l'hypothèse nulle : il existe une relation significative entre la variable explicative et la gravité des accidents.
- Si $p \geq 0.05$, on ne peut pas rejeter l'hypothèse nulle : la variable explicative n'a pas d'effet significatif sur la gravité des accidents.

Résultat du test :

Variable	Chi2	p-value
obsm	90321.29	0.0
catv_modifié	83873.66	0.0
obs	59918.72	0.0
dep	57856.62	0.0
col	51821.87	0.0
manv	49858.81	0.0
catr	30550.32	0.0
choc	30205.76	0.0
agg	25539.26	0.0
age	25159.48	0.0
trajet	23759.95	0.0
circ	19177.28	0.0
lum	10316.16	0.0
situ	9655.17	0.0
place	8487.98	0.0
heure	8058.66	0.0
secu_combined	6998.61	0.0
sexe	6257.46	0.0
catu	6055.06	0.0
int	5053.66	0.0
jour_semaine	2868.52	0.0
infra	2615.97	0.0
atm	2337.84	0.0
surf	1571.35	0.0
mois	1248.60	0.0
an	767.76	0.0

On observe que toutes les variables explicatives retenues c'est-à-dire nos 27 variables sont corrélées à la variable cible. En effet, toutes les variables affichent une p-valeur de 0.0, indiquant ainsi une relation significative avec la variable cible.

Les variables ayant un **impact majeur** sur la gravité de l'accident sont l'obstacle fixe (obs) et l'obstacle mobile heurté (obsm), la catégorie du véhicule (catv_modifié), la manœuvre (manv), le type de collision (col), le département (dep), la catégorie de route (catr) et le point du choc initial (choc).

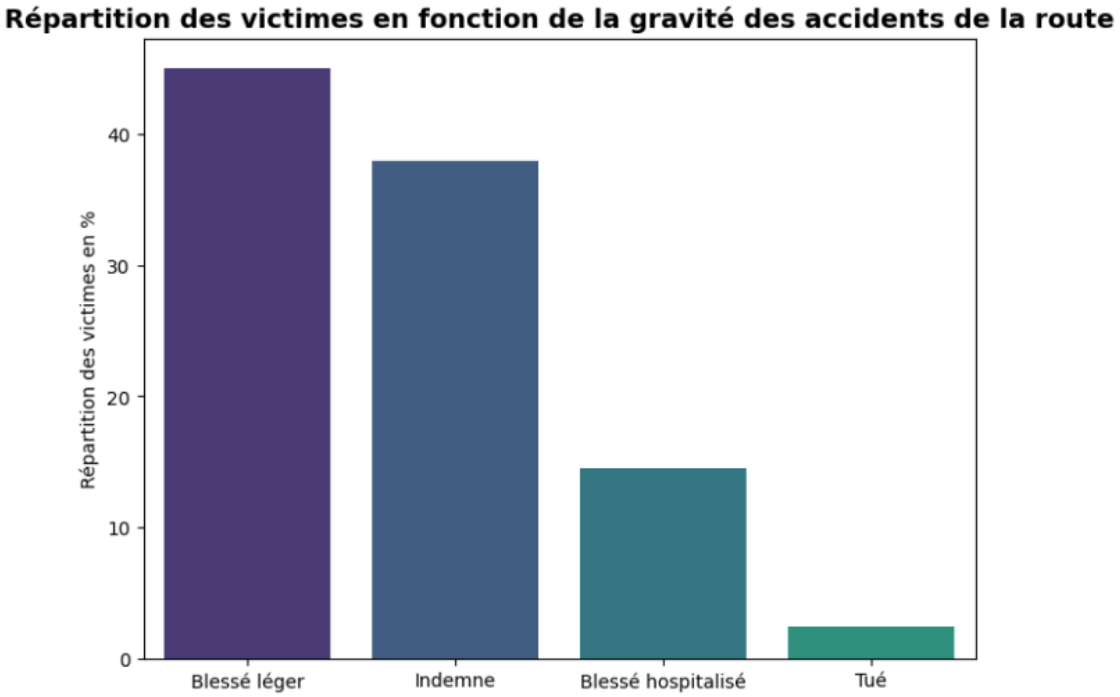
Les variables exerçant une **influence moins forte** sur la gravité de l'accident sont :

- Localisation de l'accident : en agglomération ou hors agglomération (agg)
- Âge de l'utilisateur au moment de l'accident (age)
- Trajet : le type de trajet effectué a un effet significatif sur la gravité de l'accident
- Circulation (circ) et catégorie de route (catr), ont également une forte influence.
- Luminosité (lum), l'heure, l'infrastructure (infra) exercent une influence importante.
- La surface (surf) c'est-à-dire l'état de la chaussée, a un impact plus modéré sur la gravité de l'accident.

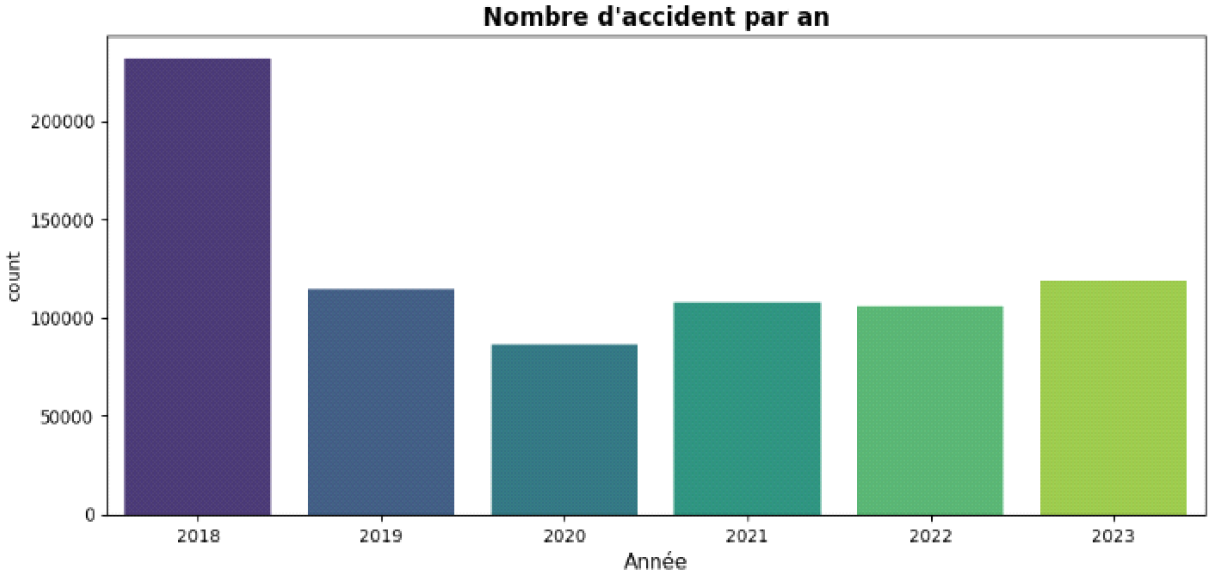
Ces résultats permettent d'identifier les variables clés pour mieux comprendre et prévenir les accidents de la route.

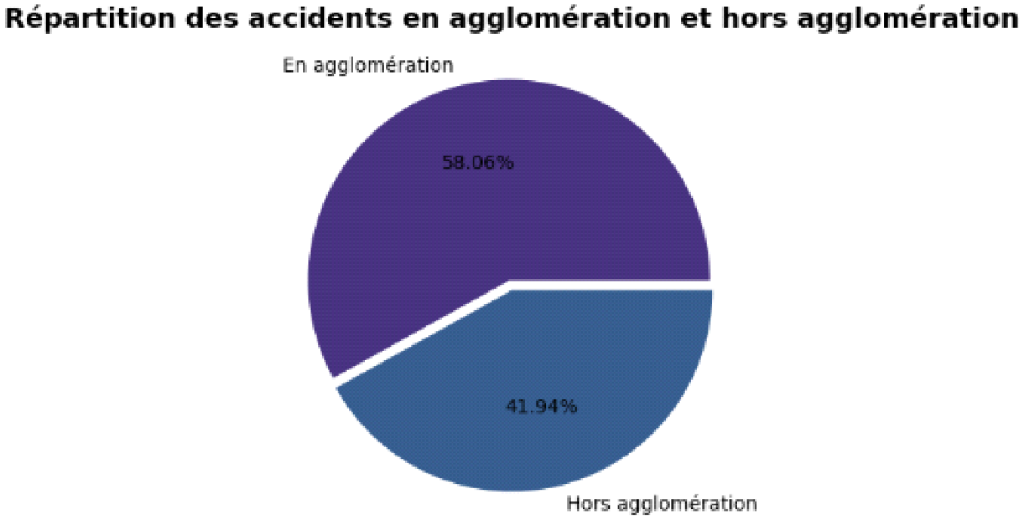
4. Visualisations graphiques

Dans un premier temps, nous avons mis en avant trois graphiques qui permettent de visualiser dans son ensemble notre jeu de données en fonction de trois critères : la répartition du nombre d'accidents selon leur gravité, le nombre d'accidents par an et la répartition des accidents en fonction du lieu (agglomération ou hors agglomération).



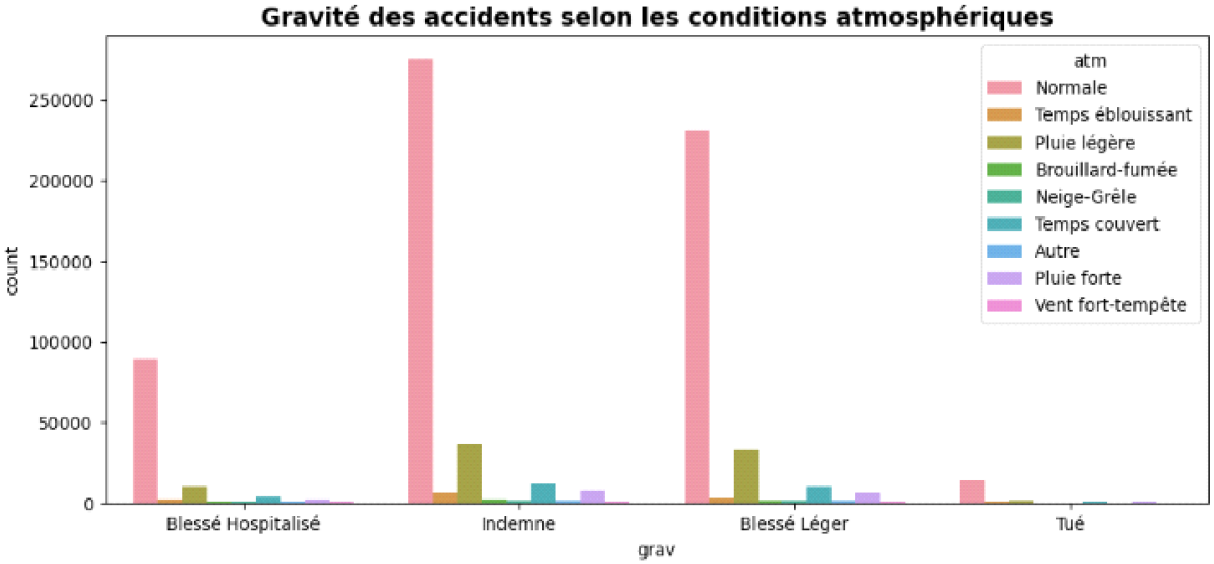
On observe que la majorité des victimes sont soit blessées légèrement (un peu plus de 40 %) soit indemnes (environ 40 %). Les blessés hospitalisés représentent une proportion nettement plus faible, tandis que les cas de décès sont rares. Ces données suggèrent que la plupart des accidents ont entrainé des conséquences relativement mineures, tandis que les accidents graves restent marginaux.



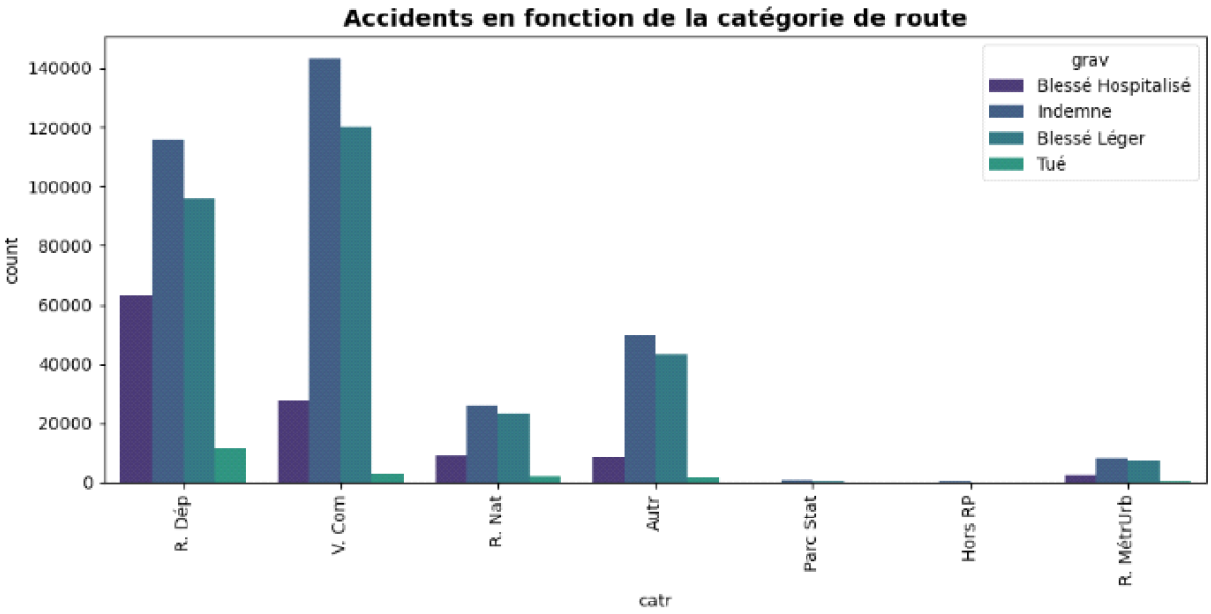


Il est observé une baisse significative du nombre d'accidents routiers entre 2018 et 2022. Ces accidents ont lieu essentiellement en agglomération et représentent 61,53 %.

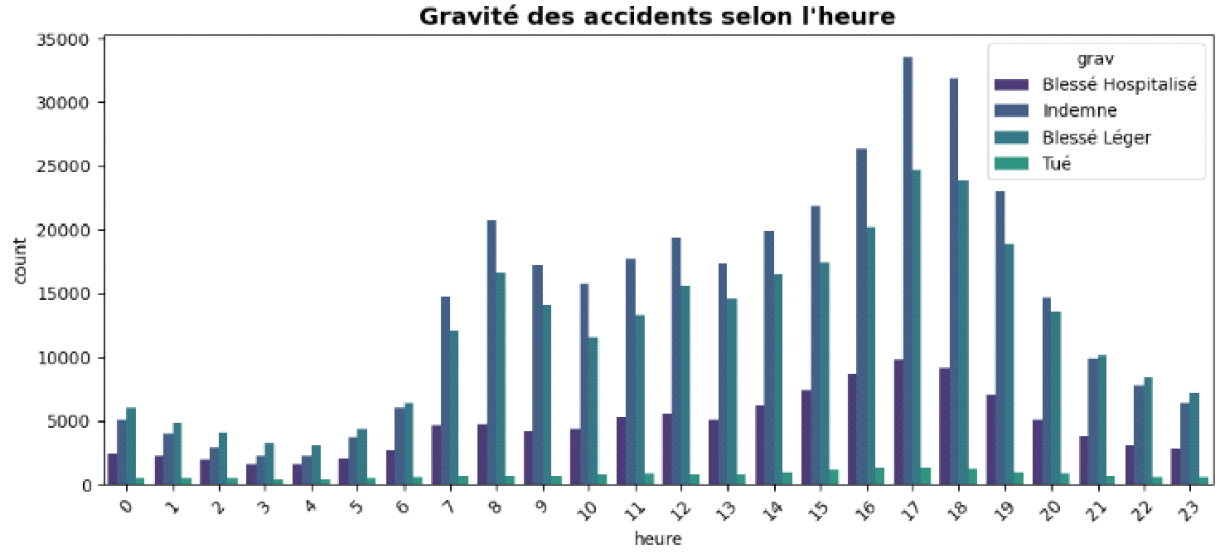
Dans un second temps, nous avons réalisé plusieurs graphiques démontrant la relation entre la gravité des accidents routiers en France (variable cible) et certaines variables explicatives, c'est-à-dire celles qui présentent une influence importante sur la gravité de l'accident. Nous avons décidé de présenter les graphiques les plus pertinents pour notre analyse.



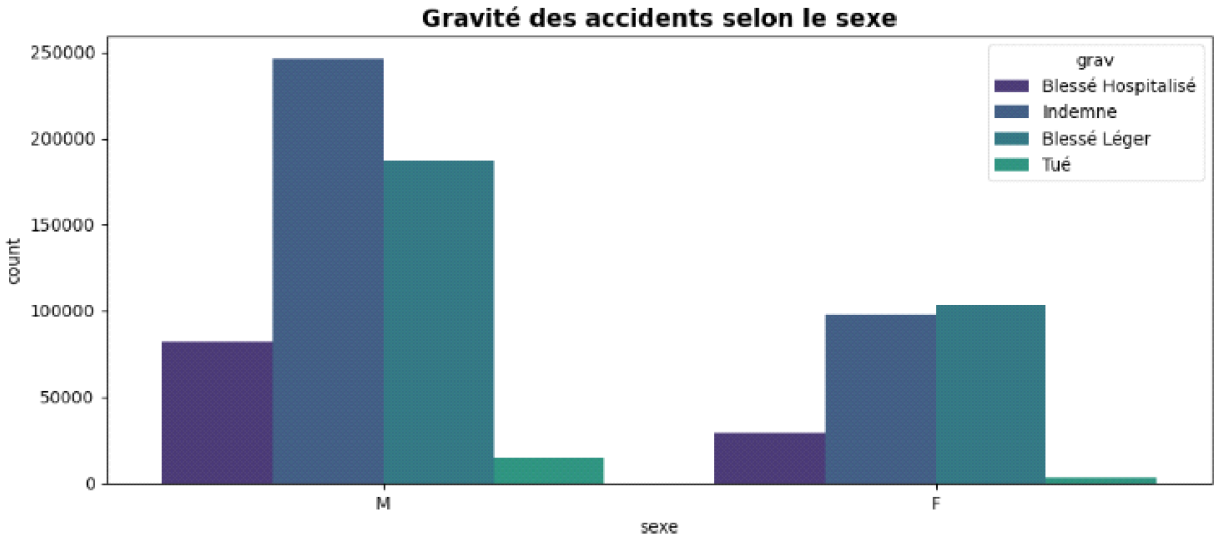
Ce graphique montre que, quelle que soit la gravité des accidents, ces derniers surviennent majoritairement lorsque les conditions atmosphériques sont normales. Les usagers accidentés par temps dit « normal » sont principalement des victimes indemnes ou des blessés légers.



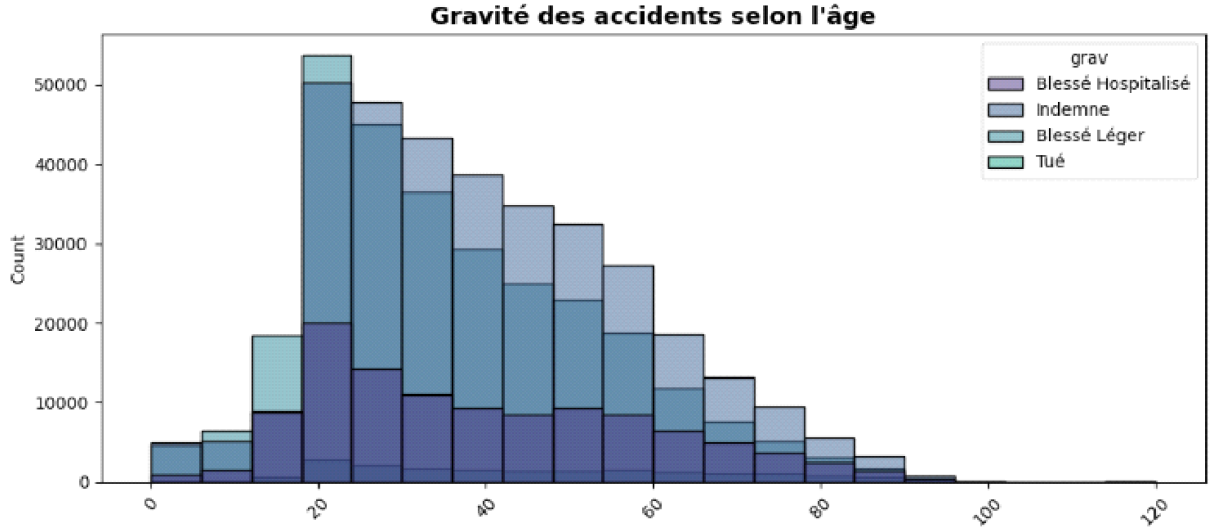
On observe que les catégories de route départementales et communales ont le plus d'accidents avec essentiellement des blessés hospitalisés ou des victimes indemnes. Les autoroutes et les routes nationales concentrent moins d'accidents routiers mais comptabilisent les cas les plus graves. Les autres catégories de routes concentrent peu d'accidents routiers.



Ce graphique montre la gravité des accidents en fonction de l'heure dans la journée. Les usagers accidentés sont essentiellement des victimes indemnes et des blessés légers avec une forte augmentation de cette catégorie pendant les heures de pointe c'est-à-dire entre 8 et 9 heures ainsi qu'entre 16 et 19 heures. Ces deux plages horaires respectives concentrent un grand nombre d'accidents. Notons également que les accidents nocturnes, sont moins fréquents, mais restent souvent les plus graves.



Il est observé que le sexe exerce une influence sur la gravité des accidents. En effet, pour chaque catégorie de gravité d'accident, le sexe masculin enregistre plus d'accidents routiers en France entre 2018 et 2023 que le sexe féminin.



Ce graphique montre que la majorité des accidents touchent les personnes âgées entre 15 et 25 ans avec une forte hausse des accidents autour des 20 ans.

Ces accidents diminuent après l'âge de 25 ans. Autrement dit, les jeunes adultes sont les plus impliqués dans les accidents certainement parce qu'ils sont moins expérimentés.

On peut également observer que les accidents des personnes âgées sont les plus graves. En effet, on voit sur ce graphique que la proportion des blessés "hospitalisés" et "tués", est plus importante comparativement aux proportions des autres classes de gravité sur les autres tranches d'âges.

On pourrait conclure qu'il serait important de prendre des mesures de prévention et de sensibilisation pour les jeunes conducteurs mais aussi les seniors.

5. Séparation du jeu de données

Nous avons réalisé cette étape avant encodage des variables pour éviter la fuite des données en évitant que des informations de l'ensemble test n'influencent involontairement le modèle.

Ce processus garantit une évaluation précise du modèle tout en maintenant l'intégrité des données et en simulant un contexte de production réaliste.

Ainsi, nous avons séparé les données en variables explicatives (*dataframe* X) et en variable expliquée (*dataframe* y).

Puis, nous avons utilisé la fonction *train_test_split*, de la librairie *Scikit-Learn* de Python afin de séparer X et y en un ensemble d'entraînement et un ensemble test.

Nous avons également utilisé le paramètre *random_state* afin de permettre la reproductibilité de découpage des données et éviter ainsi des résultats incohérents.

Il est à noter que nous avons divisé aléatoirement les variables explicatives en un ensemble d'entraînement et de test, respectivement à 80% et 20% de la quantité totale des données disponibles.

6. Encodage des variables explicatives sur les données d'entraînement

Notre jeu de données est composé de variables qualitatives qui possèdent plus de deux catégories, et chaque catégorie se voit attribuer une valeur numérique qui ne leur confère pas d'importance relative.

Autrement dit, notre *dataframe* est composé de variables déjà encodées par labels.
A titre d'exemple, la variable surface (noté surf) est présentée de la manière suivante :

-1 – Non renseigné

1 – Normale

2 – Mouillée

3 – Flaques

4 – Inondée

5 – Enneigée

6 – Boue

7 – Verglacée

8 – Corps gras – huile

9 – Autre

Diverses méthodes d'encodage existent en fonction des types de données. Le *One Hot Encoding* est l'une d'entre elles, elle permet notamment d'encoder les variables sous forme binaire. Ce qui se traduit généralement par la création de colonnes supplémentaires.

Nous avons choisi de transformer nos variables avec les techniques *One Hot Encoding* ou encore en *Frequency Encoding*, qui sont des méthodes nominales, qui conviennent aux variables de notre jeu de données.

Il est à noter que lorsque la technique *One Hot Encoding* transformait la variable avec plus de 10 modalités en créant de nouvelles colonnes, nous avons alors utilisé la technique *Frequency Encoding* afin d'éviter d'obtenir un nombre important de variables explicatives.

Cette étape est primordiale car les algorithmes de *machine learning* nécessitent des valeurs numériques pour fonctionner correctement et donner des résultats cohérents.

Les variables encodées avec la technique du *One Hot Encoding sont* : lum (qui a généré 4 nouvelles variables binaires), agg (1 nouvelle variable binaire), col (6 nouvelles variables binaires), catr (7 nouvelles variables binaires), circ (4 nouvelles variables binaires), situ (7 nouvelles variables binaires), senc (3 nouvelles variables binaires), obsm (6 nouvelles variables binaires), catu (1 nouvelle variable binaire), sexe (1 nouvelle variable binaire), trajet (6 nouvelles variables binaires) et secu_combined (6 nouvelles variables binaires).

Les variables encodées avec la technique du *Frequency Encoding* sont : int (cette variable a été remplacée par 8 nouvelles catégories à la suite de l'encodage réalisé), atm (8 nouvelles catégories), surf (9 nouvelles catégories), infra (9 nouvelles catégories), obs (17 nouvelles catégories), choc (9 nouvelles catégories), manv (26 nouvelles catégories), catv_modifié (9 nouvelles catégories), place (8 nouvelles catégories) et dep (127 nouvelles catégories).

Les variables temporelles suivantes : heure, an, mois, jour_semaine, et âge sont des variables numériques quantitatives ; nous n'appliquons donc pas de réencodage.

Notons que nous appliquerons ensuite l'encodage à nos données test en utilisant la méthode appelée « .transform() » lors de la modélisation dans la deuxième partie de ce projet.

7. Standardisation des données

Les variables explicatives de notre *dataframe* ne possèdent pas toutes la même échelle de valeurs. Ainsi, pour éviter de mettre en péril l'étape de modélisation, nous devons procéder à une mise à l'échelle de nos données (appelée aussi *scaling*). Après avoir réalisé une représentation graphique de la distribution de nos variables, nous avons choisi d'utiliser la méthode de standardisation car nos variables présentaient des distributions différentes.

Pour conclure la partie 1, qui porte sur l'exploration et le *preprocessing* des données, notre jeu de données présentait plusieurs particularités :

- Un nombre important de variables (63 au départ). Après nettoyage et réalisation de tests statistiques pour mesurer la corrélation des variables explicatives à la variable cible, ce nombre a été réduit à 27 variables.
- La présence de valeurs manquantes et non renseignées que nous avons supprimées. Le pourcentage de ces données étant faible par rapport au reste de notre *dataframe*, nous avons choisi de les supprimer.
- Et l'encodage de nos variables, qui a créé plusieurs nouvelles catégories pour chacune d'elles.

Après avoir traité ces différentes particularités, nous allons procéder à la modélisation, qui fera donc l'objet de notre deuxième partie.

PARTIE 2 :

MODÉLISATION

I. Etapes de réalisation

1. Classification du problème

Notre projet s'apparente à un problème de classification supervisée.

Les variables que nous possédons dans notre jeu de données sont des variables catégorielles et nous cherchons à déterminer la gravité des accidents (variable cible) selon un classement qui est le suivant : blessé léger, indemne, hospitalisé et tué.

Nous cherchons donc à classer les accidents de la route en France en fonction de ces classes. Ainsi, compte tenu de ces éléments, nous cherchons à traiter une problématique de **classification multi-classes**.

Concernant les métriques, elles ont été choisies en fonction de notre problématique (classification multi-classe) mais aussi en cohérence avec notre objectif.

Notre objectif était de mettre en place un modèle capable de prédire les accidents les plus graves (blessé hospitalisé et tué), afin de déterminer le caractère urgent de l'accident et ainsi réduire le délai d'intervention des secours.

La métrique de performance principale utilisée pour comparer nos modèles est **le *recall* car nous souhaitions mesurer la capacité du modèle à détecter correctement les accidents ayant entraîné des hospitalisés ou des tués**. Autrement dit, nous voulions calculer, parmi tous les accidents ayant effectivement causé la mort, la proportion que le modèle a correctement prédite comme étant des hospitalisées et des tués.

Ainsi, le recall dans notre projet, mesure la proportion des personnes réellement hospitalisées et tuées que le modèle a bien reconnues comme des « hospitalisées et tuées ».

Il est à noter qu'un *recall* élevé signifie que le modèle a su correctement détecter la plupart des accidents les plus graves.

Nous avons également retenu la **précision** car cette métrique permet de répondre à la question suivante : « **parmi toutes les prédictions positives du modèle, combien sont de vrais positifs ?** »[2]

Ainsi, la précision mesure la proportion d'accidents que le modèle a prédits comme étant grave, et qui étaient réellement des accidents graves. **En d'autres termes, elle indique si les prédictions de gravité (blessés hospitalisés et tués) faites par le modèle sont fiables.**

Notons qu'une précision élevée signifie que lorsque le modèle a prédit un accident grave, il a généralement raison. Le modèle est donc fiable lorsqu'il identifie un accident grave.

Pour choisir les métriques de performances qui nous permettait de répondre à notre problématique mais surtout à notre objectif, il nous a fallu passer par la matrice de confusion et réfléchir en ayant notre objectif en tête afin de trouver les bonnes métriques de performances.

[2] *Citation du cours de DataScientest, Sprint 1, Module 101, Notebook 13 : Modèles simples de classification.*

2. Choix du modèle et optimisation

Pour répondre à la fois à notre objectif et au problème de classification multi-classe, nous avons testé plusieurs modèles, avec différents hyperparamètres, et testé différentes techniques de rééchantillonnage notamment l'*undersampling* et l'*oversampling*. Nous avons codé 3 notebooks de modélisation.

a. Notebook 1 : cible multi-classe et technique de rééchantillonnage undersampling

Dans ce notebook, la cible est représentée par 4 classes : blessé hospitalisé, blessé léger, indemne et tué. Nous avons utilisé la technique de rééchantillonnage *undersampling* afin de rééquilibrer les classes en réduisant le nombre des lignes des classes majoritaires.

Nous avons ensuite coder les modèles suivants :

- Régression logistique
- Régression logistique avec RandomizedSearchCV et hyperparamètres
- Random Forest
- Random forest avec RandomizedSearchCV et hyperparamètres
- KNN
- KNN avec RandomizedSearchCV et hyperparamètres
- XGBoost
- XGBoost avec RandomizedSearchCV et hyperparamètres

Nous avons choisi de comparer des modèles basés sur des approches variées en termes de complexité, de capacité de généralisation et de sensibilité au déséquilibre des classes.

Dans ce notebook, nous avons calculé un *recall* global sans prioriser de classes afin de tester nos modèles et évaluer le temps de calcul. Les hyperparamètres ont été choisis essentiellement de manière aléatoire. Par ailleurs, dans ce notebook, nous n'avons pas procédé à l'interprétation des résultats notamment en utilisant des techniques d'interprétabilité telle que *Shap* car les différents modèles utilisés, ne parvenaient pas à discriminer correctement les modalités les plus proches, c'est-à-dire entre les classes « blessée léger et indemne » ainsi que les classes « hospitalisée et tuée ».

L'évaluation des performances des modèles multi-classes sur les quatre catégories de gravité montre des limites importantes. Le *recall* sur la classe « tué » atteint des valeurs jusqu'à 0.73 avec le Random Forest avec GridSearchCV et hyperparamètres. Ce qui indique que la majorité des cas graves sont correctement détectés.

Cependant, le modèle *Random Forest* avec RandomizedSearchCV et hyperparamètres (*Tableau n°1*) présente une forte confusion entre les classes, en particulier celles ayant des modalités proches en termes de gravité. Par exemple, parmi les **22 035 blessés hospitalisés réels**, seulement **5 642** ont été correctement identifiés, tandis que **4 369** ont été prédits comme *blessés légers* et surtout **9 576** comme *tués*, ce qui révèle une confusion majeure entre accidents graves et décès.

De même, sur **58 290 blessés légers**, le modèle en confond **15 313** avec des *indemnes* et **9 594** avec des *tués*, ce qui traduit une mauvaise capacité à nuancer le niveau de gravité des accidents.

Les **68 965 indemnes** sont eux aussi mal classés dans **12 152 cas** comme *blessés légers* et dans **8 080 cas** comme *tués*. Le modèle a tendance à surévaluer la gravité.

Enfin, bien que **2 637 tués** soient correctement prédits sur **3 630**, cette classe présente une **faible précision (0,09)**, ce qui montre que de nombreux individus d'autres classes sont à tort prédits comme *tués*. Ces résultats soulignent la difficulté du modèle à distinguer des catégories proches, ce qui nuit à sa fiabilité globale.

Tableau N°1 : Matrice de confusion et rapport de classification du modèle Random Forest avec RandomizedSearchCV et hyperparamètres

Recall: 0.51

Matrice de confusion :

Prédit / Réel	Blessé hospitalisé	Blessé léger	Indemne	Tué
Blessé hospitalisé	5642	4369	2448	9576
Blessé léger	7159	26224	15313	9594
Indemne	4493	12152	44240	8080
Tué	473	293	227	2637

Rapport de classification :

	precision	recall	f1-score	support
Blessé hospitalisé	0.32	0.26	0.28	22035
Blessé léger	0.61	0.45	0.52	58290
Indemne	0.71	0.64	0.67	68965
Tué	0.09	0.73	0.16	3630
accuracy			0.51	152920
macro avg	0.43	0.52	0.41	152920
weighted avg	0.60	0.51	0.55	152920

Cette analyse met en évidence que la classification faite en quatre classes entraînait une confusion importante entre niveaux de gravité proches, rendant difficile la distinction précise entre accidents légers et graves.

Cette confusion ne nous permettait pas de répondre à notre objectif et impactait la performance des modèles. Ainsi, sur les conseils de notre tuteur, **nous avons transformé le problème de classification multi-classe en un problème de classification binaire**. Par conséquent, nous avons fusionné les classes ayant les modalités les plus proches afin d'en recréer deux : **la classe des blessés légers regroupant "indemne" et "blessé léger"(classe 0) et la classe des blessés graves regroupant "tué" et "blessé hospitalisé" (classe 1)**.

La nouvelle répartition se décompose comme suit :

- Classe 0 : 635 076 lignes
- Classe 1 : 129 524 lignes

En réduisant les classes de gravité des accidents, **notre objectif est de développer un modèle capable de détecter les cas les plus graves, en maximisant le rappel sur la classe 1 (blessés graves) pour ne pas manquer un cas grave.**

Nous avons également retenu la précision sur la classe 1 afin de déterminer si les cas graves détectés sont réellement des cas graves, et ainsi limiter les fausses alertes.

Enfin nous avons également observé **la précision sur la classe 0 (blessé léger) pour analyser l'équilibre global du modèle.**

Nous avons donc codé deux autres notebooks avec une cible de classe binaire et des techniques de rééchantillonnages différentes (*undersampling* et *oversampling*). Notre but était de comparer ces deux notebooks afin de retenir le modèle le plus performant en fonction des hyperparamètres utilisés mais aussi en fonction de la technique de rééchantillonnage.

b. Notebook 2 : cible classe binaire et technique de rééchantillonnage *undersampling*

Dans ce notebook, la cible est représentée par deux classes : les blessés légers (classe 0) et les blessés graves (classe 1). Nous avons utilisé la technique de rééchantillonnage *undersampling* et testé différents modèles avec et sans hyperparamètres afin de comparer les modèles pour mesurer l'impact des hyperparamètres sur la performance.

Rappelons que nous cherchons à obtenir un *recall* élevé sur la classe 1 et une précision élevée sur la classe 1. Un *recall* élevé signifie que le modèle détecte correctement la majorité des accidents avec blessés graves, c'est-à-dire qu'il ne manque que peu de cas graves réels (peu de faux négatifs). Une précision élevée sur la classe 1 signifie que la majorité des cas prédits comme graves sont réellement graves. Ce qui permet de limiter les fausses alertes (peu de faux positifs).

Par ailleurs, nous avons aussi observé la précision en classe 0 (blessé léger) pour l'équilibre global du modèle. Nous souhaitons également vérifier que parmi les accidents prédits comme légers, la proportion de cas réellement graves reste faible.

Nous avons donc choisi de comparer des modèles reposant sur des approches variées en termes de complexité, de capacité de généralisation et de sensibilité au déséquilibre des classes, afin d'identifier la meilleure approche selon le compromis entre le rappel et la précision.

Ainsi, nous avons codé les modèles suivants :

- Régression logistique
- Régression logistique avec GridSearchCV et hyperparamètres
- Random Forest
- Random forest avec RandomizedSearchCV et hyperparamètres
- KNN
- KNN avec RandomizedSearchCV et hyperparamètres
- XGBoost avec GridSearchCV et hyperparamètres

Les hyperparamètres ont été choisis dans un premier temps, de manière aléatoire, puis dans un second temps, nous avons dû affiner ces hyperparamètres pour 2 raisons. La première : optimiser la métrique principale qui est le *recall* sur la classe 1. La seconde : économiser le temps de calcul.

Pour affiner les hyperparamètres, nous avons procédé à un choix d'hyperparamètres de manière aléatoire avec la fonction *RandomizedSearch*. Ces hyperparamètres ont été définis et affinés manuellement en fonction des résultats obtenus suite à plusieurs expériences. Dans certains cas, nous avons utilisé le GridSearchCV car il était peu coûteux en temps de calcul.

A titre d'exemple, pour obtenir un *recall* élevé et donc obtenir un modèle performant, nous avons comme expliqué précédemment, utilisé la fonction *RandomizedSearchCV* (ou *la fonction GridSearchCV*), afin de tester aléatoirement plusieurs combinaisons d'hyperparamètres. Ces combinaisons, initialement définies puis affinées manuellement, nous ont permis de choisir celle qui optimise le *recall* sur la classe 1.

Pour économiser le temps de calcul et à titre d'exemple, nous avons utilisé la fonction *RandomizedSearchCV* à la place du *GridSearchCV*.

Nous avons également, sur certains modèles, revus à la baisse le paramètre cross-validation égale à 3 plis au lieu d'un cross-validation à 5 plis, mais aussi le nombre d'itération, etc.

Comme expliqué précédemment nous avons testé différents modèles, avec et sans optimisation des hyperparamètres. Aucune amélioration significative n'a été observée, ou seulement une amélioration très légère dans certains cas (*Tableaux N°2, 3, 4 et 5*). Ainsi, pour l'analyse de la matrice de confusion et du rapport de classification, nous retiendrons les modèles optimisés avec hyperparamètres.

Nous vous présentons ci-dessous les matrices de confusion et les rapports de classification de chaque modèle.

Tableaux N°2 : Matrice de confusion et rapport de classification du modèle de Régression Logistique

Régression logistique

Matrice de confusion

Classe prédite / Classe réelle	0 - Léger	1 - Grave
0 - Léger	90844	36411
1 - Grave	5791	19874

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.94	0.71	0.81	127255
1 - Grave	0.35	0.77	0.49	25665
accuracy			0.72	152920
macro avg	0.65	0.74	0.65	152920
weighted avg	0.84	0.72	0.76	152920

Régression logistique avec GridSearchCV et hyperparamètres

Matrice de confusion

classe prédite / classe réelle	0 - Léger	1 - Grave
0 - Léger	90839	36416
1 - Grave	5807	19858

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.94	0.71	0.81	127255
1 - Grave	0.35	0.77	0.48	25665
accuracy			0.72	152920
macro avg	0.65	0.74	0.65	152920
weighted avg	0.84	0.72	0.76	152920

Le modèle de régression logistique avec *GridSearchCV* (*Tableau N°2*) présente un ***recall* élevé pour la classe 1 (blessé grave) de 0.77**, ce qui est satisfaisant : le modèle détecte correctement la majorité des cas graves, avec peu de cas graves oubliés.

En revanche, **il prédit souvent qu'un cas est grave alors qu'il ne l'est pas réellement**, ce qui entraîne **beaucoup de faux positifs**. Cela se reflète dans la précision de la classe 1, qui est de 0.35. Ce qui signifie que **seulement 35 % des cas prédits comme graves le sont vraiment**. **Autrement dit, 65 % des alertes "graves" sont fausses.** Ce déséquilibre peut poser problème si chaque alerte déclenche des actions coûteuses ou urgentes inutilement.

En parallèle, **la précision pour la classe 0 (blessé léger) atteint 0.94**, ce qui montre que le modèle est performant pour détecter les cas non graves. Mais cela signifie également, qu'**il prédit dans 6% des cas des accidents légers alors qu'en réalité ils étaient graves.**

Tableaux N°3 : Matrice de confusion et rapport de classification du modèle Random Forest

Random Forest

Matrice de confusion

classe prédite	0 - Léger	1 - Grave
classe réelle		
0 - Léger	97788	29467
1 - Grave	4728	20937

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.95	0.77	0.85	127255
1 - Grave	0.42	0.82	0.55	25665
accuracy			0.78	152920
macro avg	0.68	0.79	0.70	152920
weighted avg	0.86	0.78	0.80	152920

Random Forest avec RandomizedSearchCV et hyperparamètres

Matrice de confusion

classe prédite	0 - Léger	1 - Grave
classe réelle		
0 - Léger	96922	30333
1 - Grave	4647	21018

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.95	0.76	0.85	127255
1 - Grave	0.41	0.82	0.55	25665
accuracy			0.77	152920
macro avg	0.68	0.79	0.70	152920
weighted avg	0.86	0.77	0.80	152920

Le modèle *Random Forest* avec *RandomizedSearchCV (Tableau N°3)* présente un ***recall* élevé pour la classe 1 (blessé grave) de 0,82**, ce qui est aussi satisfaisant. Le modèle détecte correctement la majorité des cas graves, avec peu de cas graves oubliés.

Cependant, **il prédit souvent qu'un cas est grave alors qu'il ne l'est pas réellement**, ce qui entraîne une nouvelle fois **beaucoup de faux positifs**. Cela se reflète dans la précision de la classe 1, qui est de 0,41. Donc **seulement 41 % des cas prédits comme graves le sont vraiment**. **En d'autres termes, on observe 59% des alertes "graves" sont fausses.**

Concernant, la précision pour la classe 0 (blessé léger) atteint 0.95. Ainsi, le modèle est performant pour détecter les cas non graves. Mais il prédit dans **5% des cas des accidents légers alors qu'en réalité ils étaient graves.**

Tableaux N°4 : Matrice de confusion et rapport de classification du modèle KNN

KNN

Matrice de confusion :

Classe prédite	0.0	1.0
Classe réelle		
0.0	89702	37553
1.0	6252	19413

Rapport de classification :

	precision	recall	f1-score	support
0 - Léger	0.93	0.70	0.80	127255
1 - Grave	0.34	0.76	0.47	25665
accuracy			0.71	152920
macro avg	0.64	0.73	0.64	152920
weighted avg	0.84	0.71	0.75	152920

KNN avec RandomizedSearchCV et hyperparamètres

Matrice de confusion

classe prédite	0 - Léger	1 - Grave
classe réelle		
0 - Léger	91919	35336
1 - Grave	5467	20198

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.94	0.72	0.82	127255
1 - Grave	0.36	0.79	0.50	25665
accuracy			0.73	152920
macro avg	0.65	0.75	0.66	152920
weighted avg	0.85	0.73	0.76	152920

Le modèle *KNN* avec *RandomizedSearchCV (Tableau N°4)* présente un ***recall* élevé de 0,79 pour la classe 1**.

La précision de la classe 1 est encore basse (0,36). Donc **seulement 36% des cas prédits comme graves le sont vraiment**. **En d'autres termes, on observe 64% des alertes "graves" sont fausses.**

La précision pour la classe 0 est de 0.94. Ce modèle est également performant pour détecter les cas non graves. Mais il prédit dans **6% des cas des accidents légers alors qu'en réalité ils étaient graves** comme dans le modèle de la régression logistique**.**

Tableaux N°5 : Matrice de confusion et rapport de classification du modèle XGBoost

XGBoost avec GridSearchCV et hyperparamètres

Matrice de confusion

classe prédite	0 - Léger	1 - Grave
classe réelle		
0 - Léger	52753	74502
1 - Grave	2218	23447

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.96	0.41	0.58	127255
1 - Grave	0.24	0.91	0.38	25665
accuracy			0.50	152920
macro avg	0.60	0.66	0.48	152920
weighted avg	0.84	0.50	0.55	152920

Le modèle *XGBoost* avec *GriSearchCV* présente le ***recall*** **le plus élevé avec une valeur de 0,90 pour la classe 1**.

En revanche, la précision de la classe 1 est la plus basse observée avec 0,24. Donc **seulement 24% des cas prédits comme graves le sont vraiment**. **En d'autres termes, on observe que 76% des alertes "graves" sont fausses.**

La précision pour la classe 0 est de 0.96. Ce modèle est également performant pour détecter les cas non graves. Mais il prédit dans **4% des cas des accidents légers alors qu'en réalité ils étaient graves.**

Pour plus de clarté, nous avons ci-dessous un résumé des résultats du rappel et de la précision des modèles utilisés avec la technique d'échantillonnage *undersampling (Tableau N°6).*

Tableau N°6 : récapitulatif des métriques des modèles testés avec la technique d'échantillonnage undersampling

Modèles avec technique d'échantillonnage undersampling	Recall sur la classe 1 : "grave"	Précision sur classe 1 : "grave"	Précision sur classe 0 : "léger"
Régression logistique avec GridSearchCV et hyperparamètres	0,77	0,35	0,94
Random Forest avec RandomizedSearchCV et hyperparamètres	0,82	0,41	0,95
KNN avec RandomizedSearchCV et hyperparamètres	0,79	0,36	0,94
XGBoost avec GridSearchCV et hyperparamètres	0,91	0,24	0,96

On observe que le *Random Forest* présente le meilleur équilibre entre les trois métriques étudiées pour ce projet : le rappel et la précision sur la classe 1, ainsi que la précision sur la classe 0.

Toutefois avant de conclure sur ce modèle, nous allons poursuivre notre analyse avec le notebook 3, qui présente les mêmes modèles avec les mêmes hyperparamètres mais avec une technique d'échantillonnage différente. Nous avons utilisé pour ce notebook, la technique d'échantillonnage *oversampling* afin d'observer si cette technique pouvait avoir un impact significatif sur la performance de nos modèles.

c. Notebook 3 : cible classe binaire et technique de rééchantillonnage *oversampling*

Dans ce notebook, la cible est représentée par 2 classes, les blessés légers (classe 0) et les blessés graves (classe 1). Nous avons utilisé la technique de rééchantillonnage *oversampling* pour équilibrer les classes en ajoutant des lignes dans la classe 1 minoritaire.

Par ailleurs, nous avons codé les mêmes modèles avec les mêmes hyperparamètres que le précédent notebook et également utiliser les mêmes fonctions d'optimisation à savoir *RandomizedSearchCV* et *GridSearchCV.*

Tableau N°7 : Matrice de confusion et rapport de classification du modèle de Régression Logistique

Matrice de confusion

classe prédite	0 - Léger	1 - Grave
classe réelle		
0 - Léger	90902	36353
1 - Grave	5802	19863

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.94	0.71	0.81	127255
1 - Grave	0.35	0.77	0.49	25665
accuracy			0.72	152920
macro avg	0.65	0.74	0.65	152920
weighted avg	0.84	0.72	0.76	152920

Le modèle de régression logistique avec *GridSearchCV (Tableau N°7)* met en avant de **bonnes performances sur la classe 1 (blessé grave), avec un *recall* de 0,77**. Cela signifie qu'il parvient à détecter 77 % des cas réellement graves.

Cependant **la précision sur la classe 1 (blessé grave) n'est pas très élevée (35 % de précision sur la gravité)**, ce qui signifie que notre modèle rencontre quelques difficultés à prédire correctement la gravité des accidents puisque seulement 35 % des cas prédits comme graves le sont vraiment. **Autrement dit, 65 % des alertes "graves" sont fausses.**

Par ailleurs, ce modèle conserve une précision élevée sur la classe 0 (blessé léger), avec une valeur 0,94. Cela indique que lorsqu'il prédit qu'un accident est léger, le modèle est fiable (6% d'erreurs : **6% des accidents sont prédits légers alors qu'en réalité ils étaient graves).**

Tableau N°8 : Matrice de confusion et rapport de classification du modèle Random Forest

Matrice de confusion

classe prédite	0 - Léger	1 - Grave
classe réelle		
0 - Léger	118518	8737
1 - Grave	12631	13034

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.90	0.93	0.92	127255
1 - Grave	0.60	0.51	0.55	25665
accuracy			0.86	152920
macro avg	0.75	0.72	0.73	152920
weighted avg	0.85	0.86	0.86	152920

Parmi les différents modèles testés, le modèle *Random Forest* avec RandomizedSearchCV (*Tableau N°8*) se révèle être le moins performant pour identifier correctement les accidents graves. **En effet, son *recall* sur la classe 1 (blessé grave) est de seulement 0,51**. Ce qui signifie qu'il a identifié correctement 51 % des cas réellement graves.

La précision sur la classe 1 (blessé grave) est en revanche la meilleure de tous les modèles que nous avons pu essayer avec une valeur de 0,60. Ainsi, **60 % des cas prédits comme graves le sont vraiment et 40% des alertes "graves" sont fausses**.

De plus, on observe également un score élevé pour la précision sur la classe 0 (blessé léger) avec 0.90. **10 % des accidents prédits comme légers sont en réalité des accidents graves**. Cela signifie qu'il y a beaucoup d'accidents graves pour lesquels le modèle ne préconise pas d'intervention.

Tableau N°9 : Matrice de confusion et rapport de classification du modèle KNN

Matrice de confusion

classe prédite	0 - Léger	1 - Grave
classe réelle		
0 - Léger	97404	29851
1 - Grave	7123	18542

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.93	0.77	0.84	127255
1 - Grave	0.38	0.72	0.50	25665
accuracy			0.76	152920
macro avg	0.66	0.74	0.67	152920
weighted avg	0.84	0.76	0.78	152920

Le modèle *KNN* avec *RandomizedSearchCV* (*Tableau N°9*) met en avant de bonnes performances sur la classe 1 (blessé grave), avec un *recall* de 0,72. **Cela signifie qu'il parvient à détecter 72 % des cas réellement graves**.

En revanche seulement 38% des accidents prédit comme étant grave par le modèle le sont réellement (précision sur la classe 1).

De plus, ce modèle possède une bonne précision sur la classe 0 (blessé léger), avec une valeur de 0,93. **Cela indique que lorsqu'il prédit qu'un accident est léger, il y a 7 % d'erreurs.**

Tableau N°10 : Matrice de confusion et rapport de classification du modèle XGBoost

Matrice de confusion

classe prédite / classe réelle	0 - Léger	1 - Grave
0 - Léger	53388	73867
1 - Grave	2335	23330

Rapport de classification

	precision	recall	f1-score	support
0 - Léger	0.96	0.42	0.58	127255
1 - Grave	0.24	0.91	0.38	25665
accuracy			0.50	152920
macro avg	0.60	0.66	0.48	152920
weighted avg	0.84	0.50	0.55	152920

Enfin, le modèle *XGBoost* avec *GridsearchCV* se démarque comme étant le meilleur modèle sur le *recall* avec une valeur de 0,91 sur la classe 1 (blessé grave), **ce qui signifie qu'il identifie près de 9 accidents graves sur 10**.

La précision de la classe 1 (blessé grave) de notre modèle XGBoost est très faible (0,24). **Seulement 24 % des cas prédits comme graves par notre modèle le sont réellement.**

Par ailleurs, il présente également une précision forte de 0,96 sur la classe 0 (blessé léger), **indiquant qu'il commet peu d'erreurs critiques consistant à prédire à tort un accident comme léger alors qu'il est en réalité grave**.

Tableau N°11 : récapitulatif des métriques des modèles testés avec la technique d'échantillonnage oversampling

Modèles avec technique d'échantillonnage oversampling	Recall sur la classe 1 : "grave"	Précision sur classe 1 : "grave"	Précision sur classe 0 : "léger"
Regression logistique avec GridSearchCV	0,77	0,35	0,94
Random forest avec RandomizedSearchCV	0,51	0,60	0,90
KNN avec RandomizedSearchCV	0,72	0,38	0,93
XGBoost avec GridSearchCV	0,91	0,24	0,96

On observe que le KNN présente le meilleur équilibre entre le rappel sur la classe 1, la précision sur la classe 1 et la précision sur la classe 0.

Toutefois avant de conclure sur ce modèle, nous allons comparer les 2 modèles sélectionnés dans chaque notebook à savoir :

- *Random Forest* avec *RandomizedSearchCV* et hyperparamètres, avec la technique d'échantillonnage *undersampling* (notebook 2)
- *KNN* avec *RandomizedSearchCV* et hyperparamètres, avec la technique d'échantillonnage *oversampling* (notebook 3)

d. Choix du modèle et conclusion

Après avoir testé nos différents modèles et essayé deux techniques d'échantillonnage, nous avons décidé de retenir le ***Random Forest*** **avec** ***RandomizedSearchCV*** **et hyperparamètres avec la méthode d'échantillonnage** ***undersampling*** car il présentait le meilleur équilibre entre nos trois métriques.

Tableau N°12 : Rappel des métriques obtenues des deux meilleurs modèles retenus selon les techniques d'échantillonnage undersampling et oversampling

	Recall sur la classe 1 : "grave"	Précision sur classe 1 : "grave"	Précision sur classe 0 : "léger"
Random forest undersampling avec RandomizedSearchCV	0,82	0,41	0,95
KNN oversampling avec RandomizedSearchCV	0,72	0,38	0,93

En effet, le modèle *Random Forest* présente le meilleur équilibre entre le recall sur la classe 1 qui est maximisé, une précision sur la classe 1 acceptable comparativement aux autres modèles et une précision sur la classe 0 élevée.

Comme nous l'avons vu précédemment, le modèle ***Random Forest*** **avec** ***RandomizedSearchCV*** présente un ***recall*** **élevé de 0,82 pour la classe 1 (blessé grave)**, ce qui signifie qu'il parvient à détecter correctement la majorité des cas graves. En revanche, sa **précision sur la classe 1 reste modeste (0,41)**, ce qui indique que 59 % des cas prédits comme « graves » ne le sont pas réellement : le modèle génère donc un certain nombre de faux positifs. Cela peut être acceptable si l'objectif est d'anticiper le risque, quitte à sur-prévoir. À l'inverse, **la précision pour la classe 0 (blessé léger) est très élevée (0,95)**, ce qui témoigne d'une bonne fiabilité lorsqu'il annonce un cas non grave.

Comparativement, le modèle ***KNN*** est un peu moins performant : son ***recall*** **sur la classe grave est plus faible (0,72)**, et il fait aussi légèrement plus d'erreurs sur les cas légers (précision de 0,93).

Ainsi, **le modèle Random Forest** offre le **meilleur compromis global** entre sensibilité aux cas graves et fiabilité sur les cas légers.

L'optimisation du modèle a nécessité un compromis entre **rappel** ***(recall)*** et **précision** sur la classe 1 (blessé grave). Afin d'obtenir un modèle globalement plus performant, **nous avons volontairement renoncé à maximiser le** ***recall*** **au profit d'une meilleure précision sur la classe 1**. Autrement dit, **le modèle détecte moins de cas graves (baisse du** ***recall*****), mais lorsqu'il prédit qu'un cas est grave, il se trompe moins souvent (hausse de la précision)**.

Ainsi, maximiser le rappel permet de ne « rien rater », mais au prix d'un grand nombre de fausses alertes. **Par conséquent, ici, nous avons préféré réduire les fausses alertes (précision la classe 1), tout en gardant un niveau de sensibilité (rappel) acceptable.**

e. Interprétation des résultats

Étant donné que nous avons retenu le **modèle *Random Forest* avec hyperparamètres et technique d'échantillonnage *undersampling*,** nous avons approfondi l'analyse en identifiant les variables ayant le plus contribué à ses prédictions, notamment en ce qui concerne la gravité des accidents.

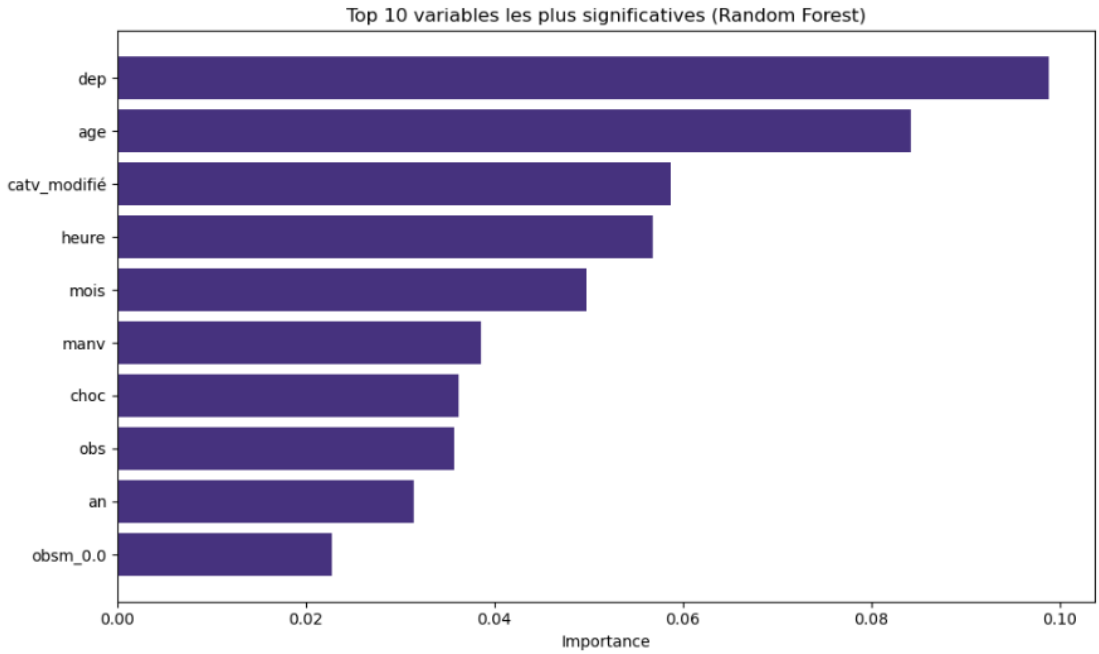


Figure N°12 : Top 10 des variables les plus influentes du modèle Random Forest

La variable la plus influente dans la prédiction de la gravité des accidents est le **département** (dep), avec une importance d'environ 0,10. Cela suggère que les différences géographiques, comme l'aménagement des routes, la densité du trafic ou les conditions locales, jouent un rôle important dans la survenue d'accidents graves.

La variable **âge** (age) influence aussi fortement le modèle dans sa prédiction des accidents graves. En effet, l'âge des personnes impliquées exerce une influence notable (0.08), ce qui reflète sans doute les différences de comportement ou de vulnérabilité selon les groupes d'âge.

La **catégorie de véhicule modifiée** (catv_modifié) et le **moment de la journée** (heure) apparaissent aussi comme des facteurs significatifs, soulignant l'importance du type de véhicule et des conditions temporelles sur la gravité des accidents. Le **mois** (mois), la **manœuvre effectuée** (manv), le type de **choc** (choc), et l'**obstacle** (obs) rencontré ont également contribué de manière non négligeable à influencer la gravité des accidents, en apportant des informations précises sur le contexte et les circonstances dans lesquels ils se produisent.

Enfin, des variables comme l'**année** (an) et la présence d'un **obstacle mobile heurté** (obsm_0.0) ont un impact plus faible, mais restent pertinentes dans la modélisation. Ces facteurs peuvent refléter des tendances évolutives ou des situations particulières d'accidents.

Ce classement des variables montre plusieurs leviers essentiels pour mieux comprendre les causes de gravité des accidents et mettre en place des actions de prévention.

f. Techniques d'optimisation utilisées

Dans tous les modèles que nous avons codés, nous avons utilisé des techniques d'optimisation de paramètres de type *GridSearchCV*, *RandomizedSearchCV* et validation croisée.

La technique ***GridSearchCV*** nous a permis de trouver une bonne combinaison d'hyperparamètres en explorant de manière exhaustive **toutes les combinaisons possibles** d'une grille de valeurs préalablement définie, afin d'améliorer les performances du modèle.

Le ***RandomizedSearchCV*** nous a permis de trouver une bonne combinaison d'hyperparamètres en explorant aléatoirement **un sous-ensemble de combinaisons** dans une grille de valeurs préalablement définie, ce qui permet d'améliorer les performances du modèle tout en réduisant le temps de recherche.

Quant à la **validation croisée**, elle nous a permis d'évaluer de manière fiable les performances des modèles. En effet, elle consiste à diviser les données en plusieurs sous-ensembles appelés plis (*k-folds* en anglais). À chaque itération, le modèle est entraîné sur $k - 1$ plis et testé sur le pli restant. Ce processus est répété k fois et chaque pli sert une fois de jeu de test. Une moyenne des performances obtenue est calculée à chaque itération. Ce qui permet d'obtenir une estimation plus fiable des performances du modèle sur de nouvelles données. Autrement dit, la validation croisée permet de mieux évaluer la capacité du modèle à se généraliser.

Ces techniques d'optimisation nous ont permis d'obtenir les meilleurs paramètres possibles dans le but d'améliorer les performances de nos modèles.

Par ailleurs, nous avons également utilisé un **algorithme de *boosting***, à savoir *XGBoost*, car il présente plusieurs avantages techniques ainsi que de bonnes performances prédictives.

Conçu pour optimiser les résultats de classification, le modèle *XGBoost* repose sur une méthode de *boosting* qui permet de combiner efficacement plusieurs arbres de décision afin d'obtenir un modèle final plus précis et plus robuste.

Il est particulièrement performant sur des jeux de données complexes, comportant des variables multiples et parfois corrélées. Tel est le cas des données liées aux accidents.

XGBoost se distingue aussi par sa capacité à réduire le surapprentissage grâce à des mécanismes de régularisation intégrés, tout en restant très rapide grâce à une exécution optimisée et répartie sur plusieurs cœurs de calcul.

Il offre également une grande robustesse face aux données bruitées ou incomplètes, ce qui en fait un choix pertinent dans des contextes réels où la qualité des données n'est pas toujours garantie.

Enfin, il offre une flexibilité importante dans le réglage des hyperparamètres, permettant d'adapter finement le modèle aux spécificités du problème.

Ainsi, les atouts du modèle XGBoost lui permettent donc de garantir des prédictions fiables et efficaces, en cohérence avec les objectifs de performance fixés pour ce projet.

II. Conclusions tirées

1. Difficultés rencontrées

a. Difficultés rencontrées lors de l'exploration et du traitement des données

Lors de l'exploration et du traitement des données, nous avons rencontré plusieurs difficultés majeures.

La première concernait l'assimilation des données sur le site data.gouv.fr. En effet, la diversité et le volume des données ont représenté une réelle réflexion sur le traitement des données. Ainsi, nous avons pris le temps de les explorer en profondeur. Cette analyse nous a permis de bien comprendre leur structure, de nous les approprier et d'identifier les variables clés pour la suite du projet.

La deuxième concernait le nettoyage des données, notamment la gestion des valeurs manquantes. Nous avons dû choisir entre supprimer ces valeurs ou les remplacer, par exemple par le mode. Ce choix était délicat, car supprimer les données manquantes risquait de réduire fortement la taille du jeu de données, tandis que leur remplacement pouvait introduire des biais.

Ensuite, la transformation de certaines variables s'est avérée complexe. Par exemple, la variable « hrmn », qui représente l'heure, les minutes et les secondes, n'était pas directement exploitable. Nous avons donc simplifié cette variable en ne conservant que les heures. Cette transformation a rendu nos graphiques plus lisibles et nous a permis de réaliser des analyses statistiques plus pertinentes, comme des tests de corrélation avec la variable cible.

Enfin, l'encodage des variables qualitatives a été un autre point délicat. Notre jeu de données contenait plusieurs variables catégorielles avec plus de deux modalités, déjà représentées par des labels numériques sans hiérarchie. Il était donc important de choisir une méthode d'encodage adaptée.

Nous avons principalement utilisé le One Hot Encoding, qui transforme chaque catégorie en une variable binaire. Cependant, pour les variables comportant plus de dix modalités, cette méthode générait trop de nouvelles colonnes, augmentant considérablement la dimension du jeu de données. Pour ces cas, nous avons choisi le Frequency Encoding, qui encode les catégories selon leur fréquence d'apparition, limitant ainsi l'augmentation du nombre de variables.

Ces étapes ont constitué les principales difficultés dans la préparation des données, nécessitant de nombreux essais pour trouver le meilleur compromis entre qualité des données et faisabilité des analyses.

b. Difficultés rencontrées lors de la modélisation

Lors de la phase de modélisation, plusieurs obstacles se sont présentés.

Tout d'abord, nous avons testé plusieurs modèles de classification multi-classe : Régression Logistique, *Random Forest*, *KNN*, *XGBoost*, ainsi que leurs versions optimisées avec *RandomizedSearchCV* ou *GridSearchCV* pour l'ajustement des hyperparamètres.

Nous avons constaté une confusion importante entre les niveaux de gravité proches. En effet, il était difficile pour les modèles de distinguer précisément les accidents légers des accidents graves. Cette confusion réduisait la performance globale des modèles et ne permettait pas de répondre à notre objectif principal. Sur les conseils de notre tuteur, nous avons donc transformé le problème de classification multiclasse en un problème de classification binaire. Nous avons fusionné les classes proches pour former deux groupes : la classe 0, regroupant « indemne » et « blessé léger », et la classe 1, regroupant « tué » et « blessé hospitalisé ». Cette simplification vise à mieux détecter les cas graves, en maximisant le rappel (*recall*) sur la classe 1, pour ne pas manquer un accident grave.

Ensuite, nous avons travaillé sur deux notebooks dédiés à cette classification binaire, en appliquant des techniques de rééchantillonnage : l'*undersampling* pour réduire la taille de la classe majoritaire, et l'*oversampling* pour augmenter la taille de la classe minoritaire.

Une difficulté majeure a été le temps de calcul très long de certains modèles. Ce problème nous a contraints à revoir leurs paramètres afin d'accélérer l'exécution, quitte à réduire légèrement leurs performances prédictives. Cette phase d'ajustement a ralenti la progression et la mise en place du projet, car il a fallu de nombreux essais pour trouver un bon compromis entre efficacité du modèle et rapidité d'exécution.

Par ailleurs, l'utilisation de SHAP a soulevé deux principales difficultés : son implémentation et l'interprétation des résultats.

La première difficulté : SHAP ne produisait aucun résultat après plus de quinze jours de calcul lorsque nous l'avons appliqué au *Random Forest* retenu. Pour contourner ce problème, nous avons créé une version allégée du modèle, réduisant le nombre d'arbres (*n_estimators* = 20 au lieu de 50 ou 100) et limitant leur profondeur (*max_depth* = 6 au lieu de 5, 10 et 20). Cette version a généré des résultats rapidement nous permettant d'obtenir un graphique *Shap*.

La deuxième difficulté : le graphique obtenu n'était pas exploitable. Les noms des variables n'apparaissaient pas, ce qui empêchait d'identifier les interactions entre elles. Ce problème d'affichage, ajouté au temps de calcul initial très long, a freiné notre progression dans le projet. Étant donné le délai imparti et malgré nos nombreuses tentatives de résolution de problème, nous avons décidé de ne pas inclure de graphique Shap.

Enfin, le choix des métriques a représenté une difficulté majeure. L'ensemble du groupe s'accordait sur l'importance de maximiser le rappel sur la classe 1, pour détecter un maximum d'accidents graves, et sur la précision de la classe 0 pour évaluer l'équilibre global du modèle. Toutefois, la prise en compte de la précision sur la classe 1 a fait débat. Certains membres ne la jugeaient pas pertinente, estimant que seul le rappel suffisait. Ces discussions ont donné lieu à plusieurs échanges très constructifs au sein du groupe, puis avec notre tuteur. Finalement, nous avons décidé ensemble de l'intégrer également, afin de limiter les fausses alertes et renforcer la robustesse globale du modèle.

Ces différentes étapes ont constitué les principales difficultés rencontrées lors de la modélisation. Elles ont nécessité de nombreux ajustements, essais et compromis pour parvenir à un modèle à la fois pertinent, interprétable et adapté aux contraintes de temps et de ressources du projet.

Pour conclure sur l'ensemble des difficultés rencontrées, celles-ci nous ont poussés à réfléchir plus en profondeur à chaque étape du projet. Elles nous ont aussi permis de progresser aussi bien dans la mise en œuvre des modèles que dans l'analyse critique des résultats, notamment l'interprétation fine des métriques et des rapports de classification. Ces difficultés ont été de véritables occasions d'apprentissage et d'enrichissement tout au long du projet.

2. Bilan

a. Contribution au projet pour chaque membre du groupe

Tout au long du projet, chaque membre du groupe a été impliqué à chaque étape, de la compréhension des données à l'élaboration du modèle, en passant par les choix méthodologiques et l'analyse des résultats. Nous avons mené des réflexions collectives régulières et chaque décision importante a été discutée et validée ensemble.

En fonction de nos contraintes personnelles et professionnelles, une répartition naturelle des tâches s'est mise en place : certains ont davantage pris en charge l'exploration des données, d'autres le développement des modèles ou encore la mise en forme des livrables.

Cependant, l'ensemble du travail a été partagé, revu et enrichi par tous, dans un esprit de coopération constante. Ce projet a ainsi été le fruit d'un véritable travail d'équipe, construit sur l'écoute, l'entraide et la complémentarité.

b. Objectifs du projet

L'objectif de ce projet était de prédire la gravité des accidents routiers en France à l'aide d'un modèle de Machine Learning, afin de déterminer le caractère urgent des situations et ainsi réduire le délai d'intervention des secours, en abordant la problématique sous l'angle de la classification.

Nous avons atteint cet objectif en développant un modèle de type *Random Forest*, capable d'estimer le niveau de gravité des accidents à partir des données historiques répertoriées sur le site du gouvernement.

Le projet nous a également permis de mettre en pratique les compétences techniques acquises tout au long de la formation, en parcourant les différentes étapes d'un projet de *Data Science* : de l'exploration des données à la modélisation, en passant par le nettoyage, l'ingénierie des variables et l'évaluation des performances. Chaque étape a fait l'objet d'une réflexion approfondie, renforçant ainsi notre capacité à aborder des cas concrets de manière rigoureuse et structurée. Cette expérience nous rend aujourd'hui plus autonomes et opérationnels pour des missions futures.

3. Suite du projet

a. Pistes d'amélioration

Nous avons répondu à notre objectif en mettant en place un modèle de prédiction de la gravité des accidents routiers en France mais nous pensons que notre modèle peut être amélioré afin d'augmenter ses performances.

La première piste d'amélioration serait de coder une technique d'**échantillonnage** plus avancée telle que la technique ***SMOTE*** afin de comparer les résultats avec les résultats obtenus des techniques d'échantillonnage ***undersampling*** et ***oversampling***. Cette approche pourrait contribuer à mieux **équilibrer le jeu de données** tout en évitant les problèmes de **surapprentissage** liés à la **répétition des mêmes observations**.

La seconde piste d'amélioration consisterait à ajuster le **seuil de classification** utilisé par le modèle. En effet, un modèle de **classification binaire** comme celui que nous avons utilisé ne renvoie pas directement une **classe** (accident grave ou non grave), mais plutôt une **probabilité** que l'accident soit grave. Par défaut, si cette probabilité est supérieure ou égale à **0,5**, le modèle classe l'exemple comme un **accident grave** (**classe 1**). En dessous, il le classe comme **non grave** (**classe 0**).

Ce seuil peut être **modifié** en fonction de nos **priorités**. Par exemple, si notre objectif est de **détecter un maximum d'accidents graves**, on peut **abaisser ce seuil**, ce qui aura pour effet d'augmenter le nombre de cas considérés comme graves. Cela permettrait au modèle d'être plus **sensible** : il identifiera plus facilement les cas graves, même si cela signifie qu'il pourrait aussi en classer certains à tort (**faux positifs**). En contrepartie, cela entraînerait généralement une **baisse de la précision** sur la **classe 1**, c'est-à-dire que parmi les cas prédits comme graves, certains ne le seront pas réellement.

Dans notre cas, cette approche est particulièrement pertinente, car dans un **contexte opérationnel**, il est souvent **préférable de générer une fausse alerte plutôt que de passer à côté d'un accident réellement grave**. En ajustant le seuil, le modèle peut donc mieux s'adapter aux **exigences du terrain**, notamment en **maximisant le rappel**, c'est-à-dire la capacité à détecter la majorité des accidents graves, **ce qui permettrait de renforcer la confiance des secours dans l'intervention des accidents graves.**

b. Evolution du projet

Dans la continuité de ce projet, plusieurs pistes pourraient être explorées pour enrichir l'analyse. Une première évolution consisterait à développer un système de scoring des accidents, en attribuant à chaque situation une probabilité de gravité plutôt qu'une classification binaire. Cela permettrait de hiérarchiser les accidents en fonction de leur criticité, et d'adapter plus finement les priorités d'intervention des secours. Ce score de risque pourrait être calibré et exploité comme un outil d'aide à la décision en temps réel.

Par ailleurs, une cartographie des accidents graves pourrait être mise en place à partir des données géographiques disponibles (comme le département, la commune ou les coordonnées GPS). En croisant les prédictions du modèle avec des données de localisation, il serait possible d'identifier des zones à forte concentration de risques et ainsi de mettre en place des plans de prévention ciblés.

Ces outils pourraient s'inscrire dans une logique de pilotage territorial de la sécurité routière, en lien avec les politiques publiques locales.

III. Bibliographie

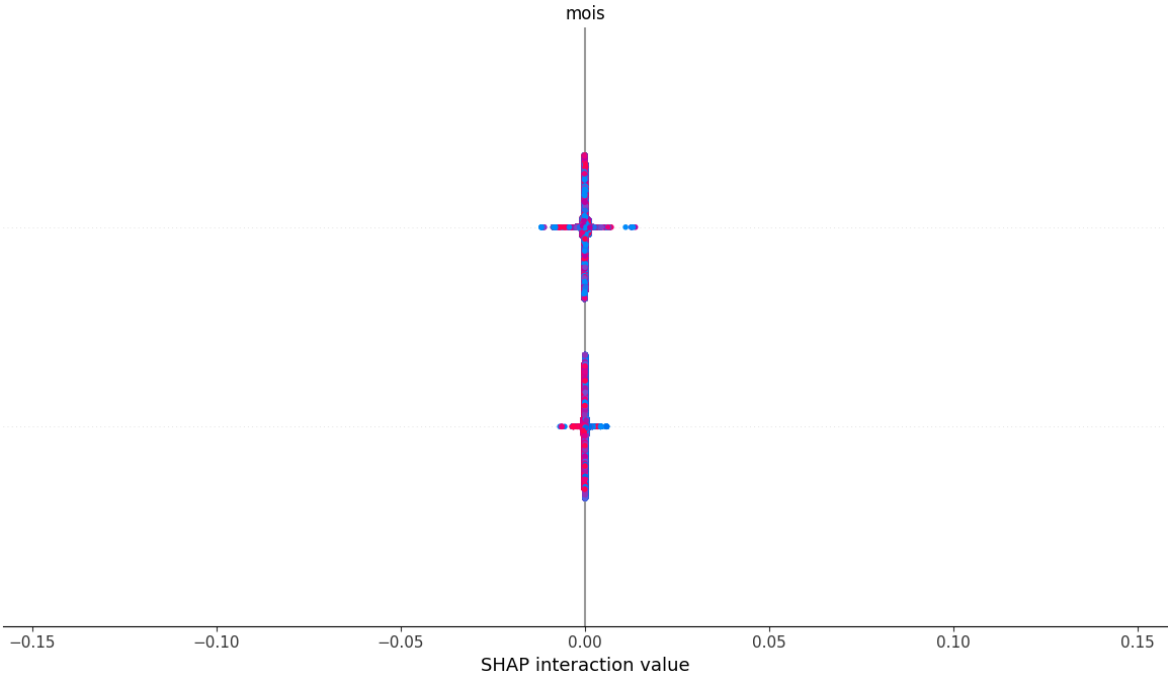
Contenus pédagogiques de la formation DataScientest :
Contenus portant sur les différents algorithmes et les étapes du projet de la Data Science.

Site data.gouv.fr :
Base de données annuelles des accidents corporels de la circulation routière - Années 2005 à 2021.

Documentation officielle Scikit-learn :
Utilisée pour l'implémentation des modèles de classification.

IV. Annexes

Premier essai du résultat du graphique *Shap* avec le problème d'affichage :



Deuxième essai du résultat du graphique *Shap* avec le problème d'affichage :