Projet Long

Analyse, visualisation et étude prédictive en Machine Learning des accidents corporels de la circulation routière en France

---

Soutenu le 30/03/2022, devant la commission d'examen :

Mr Efoevi KOUDOU

Mme Sophie MEZIERES

Réalisé par :
Isslam KHATIR
Marwa MORSLI

Remerciements

Tout d'abord, nous tenons à exprimer notre plus grande reconnaissance à Monsieur
KOUDOU Efoevi Angelo - professeur d'Apprentissage à la Faculté des Sciences et Technologies - l'université de Lorraine pour sa disponibilité, ainsi que pour ses nombreux conseils qui nous ont permis d'avancer dans notre travail.

Par ailleurs, Nous remercions également toute l'équipe pédagogique de la Faculté des Sciences et Technologies et les intervenants professionnels responsables de notre formation.

On adresse également nos vifs remerciements au membre du jury Madame Sophie Mezieres pour avoir bien voulu examiner et juger ce travail.

Enfin, Nous tenons à remercier toutes les personnes qui ont contribué au succès de notre projet et qui nous ont aidés lors de la rédaction de ce rapport.

Table des matières

Introduction

Ce sujet porte sur l'étude de la sinistralité automobile en France par des méthodes de "Machine Learning" et consiste à expliquer cette sinistralité au cours des dernières années 2020 et 2021 par des méthodes de prédiction et l'analyse de données en utilisant des données libres communément décrites sous le nom d'"Open Data". Il est notifié dans la problématique que la variable à prédire par nos modèles est le niveau de gravité d'un accident subi par un individu. Les données sont récupérées sur le site www.data.gouv.fr et sont qualifiées d'Open Data.

Ce sujet est très riche, tant par la diversité des domaines étudiés lors de cette étude (comme la sinistralité automobile ou l'usage des Open Data) que par l'application de méthodes de Machine Learning introduites lors de nos cours d'Apprentissage Statistique et de Modèles Linéaires. C'est également par ailleurs un sujet d'actualité innovant. En effet, les révolutions numériques et digitales observées ces dernières années ont pour cause l'explosion de la quantité de données disponibles, mais aussi de défis qui accompagnent ces évolutions (tels que la protection des données personnelles par exemple). Ces immenses données vont propulser sur le devant de la scène les techniques de Machine Learning, qui étaient jusqu'alors difficilement applicables à cause de limitations technologiques, et qui vont permettre aujourd'hui la mise en place de modèles prédictifs performants.

Après avoir présenté les différentes notions utilisées, notre étude se décomposera en trois étapes fondamentales présentes dans toutes les études de Data Science. Dans un premier temps, il sera question de collecter, analyser et éventuellement modifier les données disponibles. La deuxième étape consistera ensuite à construire et à expliquer la sinistralité automobile par des modèles prédictifs. Enfin, il restera à évaluer la qualité de nos modèles et appliquer ces différentes étapes à plusieurs reprises si nécessaire. Nous compléterons également notre étude par la mise en place d'un outil de représentation géographique des accidents.

Chapitre 1

Open Data

Les Open Data ou données ouvertes, sont des données rendues totalement publiques et libres d'accès et de droit, pouvant être exploitées et réutilisées. Ces "données ouvertes" vont donc permettre de conduire des études permettant d'approfondir notre connaissance ou compréhension de certains sujets, et cela dans différents domaines d'activités.

1.1 Contexte général

1.1.1 Définition générale

Le terme Open Data désigne un ensemble de données auxquelles tout le monde peut accéder, utiliser ou partager sans aucune restriction.

Ces données reposent sur des critères essentiels qui sont :

— **La disponibilité et l'accès :**

Les données doivent être accessibles de façon légale et sans barrière technique quant à leur utilisation, c'est-à-dire lisibles par les machines. Elles sont préférablement disponibles sur Internet.

— **La réutilisation et la redistribution :**

Les données doivent être mises à disposition sous des conditions permettant leur réutilisation et leur redistribution, notamment la possibilité de pouvoir les combiner à d'autres données.

— **La participation universelle :**

Tout le monde doit pouvoir accéder à ces données et être en mesure de les réutiliser et les manipuler de la même manière, sans faire de discrimination sur les fins d'utilisation ou envers des personnes ou des groupes. Aucune restriction d'utilisation ne doit être faite, y compris à des fins commerciales.

Ces trois critères sont essentiels pour l'Open Data car ils permettent l'interopérabilité qui consiste en la capacité de pouvoir mélanger différents ensembles de données donnant ainsi à différentes entreprises et/ou systèmes la possibilité de travailler ensemble.

1.1.2 Sources de l'Open Data

1.1.2.1 L'intéropérabilité : essence de l'Open Data

L'intéropérabilité évoquée ci-dessus est très importante en Open Data car elle donne par la démocratisation de la donnée, l'opportunité à différentes entités de travailler ensemble créant ainsi des systèmes d'exploitation plus larges de la donnée. La mise en commun des données qui repose sur la possibilité de les mélanger librement est nécessaire afin de développer des produits et des services en plus grande quantité et surtout de meilleure qualité.

Toutefois, cette interopérabilité ne repose pas uniquement sur le partage de données. En effet, les jeux de données doivent utiliser un langage de programmation commun ou alors un intermédiaire informatique qui permet de faciliter leur exploitation.

Certaines conditions doivent alors être respectées au moment de la création de données, notamment la certification de la provenance des données publiées, l'indication des caractéristiques qui y sont liées (nous pouvons par exemple citer la date et l'heure de création des données), ou encore la garantie de la qualité de l'information...

1.1.2.2 Une grande variété de secteurs sources

L'Open Data concerne des informations en provenance de n'importe quelle source d'informations, dans n'importe quel domaine et sur n'importe quel sujet dès lors que ces données libre d'accès sont proposées pour un usage gratuit afin que les utilisateurs puissent en tirer un bénéfice, qu'il soit commercial ou non. La plupart du temps, ces Open Data sont fournies par le gouvernement ou le secteur public : il peut s'agir de données sur des budgets, des données cartographiques, ou encore de résultats découlant de recherches scientifiques. Ce ne sont néanmoins pas les seules sources d'Open Data que l'on peut trouver.

Les Open Data sont généralement des données non personnelles pour des raisons de respect de la vie privée des individus. Nous pouvons trouver de telles données dans les domaines de la science, de la cartographie, de l'environnement, de la culture, de l'éducation, de l'économie, du développement, des affaires, ou encore de la finance. En plus de données chiffrées, nous retrouvons d'autres contenus comme des images, du texte ou de la musique.

FIGURE 1.1 – Les différents secteurs de sources d'Open Data

1.2 Utilisation d'Open Data

Aujourd'hui, les principaux acteurs qui fournissent des données ouvertes sont les gouvernements et collectivités gouvernementales ainsi que les très grandes entreprises. Ils récoltent massivement tous types de données qui constituent alors de précieuses ressources. Grâce à ce recueil d'informations massives, différentes entités les utilisent de manière totalement libre et légale afin de mieux accomplir leurs tâches.
En effet, différents groupes de personnes ou organisations qui exercent des activités très diverses les utilisent, y compris les gouvernements. Nous pouvons citer comme secteurs d'activités utilisa- teurs des Open Data : la santé, les transports, le tourisme, l'environnement, la culture...

1.3 Les données ouvertes en France : Open data Gouv

En France, le site data.gouv.fr de l'open data gouv permet à tout un chacun d'accéder librement aux données publiques pour les partager, les améliorer et les réutiliser. Cette plateforme officielle permet de répartir les données ouvertes dans plusieurs catégories.

Ces catégories sont l'agriculture et l'alimentation, la culture, l'économie et l'emploi, l'éducation et la recherche, l'international et l'Europe, le logement, le développement durable et l'énergie, la santé et le social, la société, et les transports, tourisme et territoires. Grâce à cette catégorisation, les utilisateurs peuvent facilement sélectionner le domaine qui les intéresse.

Les institutions publiques telles que les ministères, les établissements scolaires ou les collectivités territoriales fournissent la plupart des données. Il est très facile pour les usagers de mettre en ligne des données ou de partager leurs réutilisations. Les meilleures réutilisations bénéficient d'une mise en avant sur le site.

En effet, le site se destine aux producteurs de données, aux réutilisateurs, et à tous les citoyens, associations et entreprises françaises. Le site répertorie plusieurs usages à ces données ouvertes. Elles peuvent permettre de répondre à des questions, de prendre des décisions et de bénéficier de services utiles au quotidien. Cela répond aux deux axes principaux de la mission d'Etalab. Le premier consiste à faciliter la transparence numérique. Le second vise à encourager l'utilisation de la plateforme par les organisations publiques.

1.4 Conclusion

Les Open Data ont d'ores et déjà permis d'étendre considérablement le savoir et les connais- sances. En effet, ce système d'exploitation très prometteur permet d'améliorer la qualité générale des produits déjà existant et d'en créer sans cesse de nouveaux. Avec cela, un large bénéfice économique a déjà été engendré ces dernières années et devrait continuer à croître considérablement durant les prochaines décennies. Cela a permis de générer plusieurs dizaines de milliards d'euros rien qu'en Europe.

Il faut toutefois faire attention quand à la sécurisation et la publication des données personnelles des consommateurs.

Chapitre 2

La sinistralité automobile en France

Outre les Open Data, le thème principal de notre étude est la sinistralité automobile en France. Il est alors nécessaire de comprendre les différentes notions de celle-ci afin d'effectuer notre étude basée sur les accidents automobiles.

Ainsi, ce chapitre donnera les définitions des différents termes utilisés par la suite, puis fera un point sur l'historique de la sinistralité automobile en France. Enfin, il discutera des multiples causes possibles des accidents accompagné de statistiques et de chiffres clés représentatifs.

2.1 Définitions

En 2004, afin de faciliter les comparaisons internationales, en France, le comité interministériel de la sécurité routière a décidé d'harmoniser les définitions de la gravité dans le fichier des accidents corporels avec les définitions adoptées par nos principaux voisins européens.

Dans notre jeu de données, la variable à prédire est la variable qui indique le niveau de gravité de chaque individu présent dans un accident. Cette variable comporte quatre modalités qui sont : Indemne, Blessé léger, Blessé hospitalisé et Tué. Définissons alors ces différents termes.

- **Un accident corporel** (mortel et non mortel) de la circulation routière relevé par les forces de l'ordre :
  - implique au moins une victime,
  - survient sur une voie publique ou privée, ouverte à la circulation publique,
  - implique au moins un véhicule.

Un accident corporel implique un certain nombre d'usagers. Parmi ceux-ci, on distingue :

- **les personnes indemnes :** impliquées non décédées et dont l'état ne nécessite aucun soin médical du fait de l'accident,
- **les victimes :** impliquées non indemnes.
  - **les personnes tuées :** personnes qui décèdent du fait de l'accident, sur le coup ou dans les trente jours qui suivent l'accident,
  - **les personnes blessées :** victimes non tuées.
    - **les blessés dits « hospitalisés » :** victimes hospitalisées plus de 24 heures,
    - **les blessés légers :** victimes ayant fait l'objet de soins médicaux mais n'ayant pas été admises comme patients à l'hôpital plus de 24 heures.

2.2 Historique de la sinistralité automobile en France

Pour information, le premier accident automobile en France eut lieu en 1770, le véhicule impliqué, appelé le fardier de Cugnot, percuta un mur à une vitesse estimée à 4 km/h ce qui provoqua la destruction de l'engin. Aucun blessé n'a été comptabilisé.

Par la suite, les véhicules en circulation sont devenus de plus en plus performants et de plus en plus rapides. De ce fait, les accidents sont aujourd'hui beaucoup plus dangereux.

Depuis 1945, les chiffres sont édifiants. Nous parlons d'au moins 500 000 décèss, soit l'équivalent d'un grand conflit majeur, ou l'équivalent de pertes humaines subies en France lors de la Seconde Guerre mondiale.

Depuis 1960, au moins 350 000 personnes sont décédés des suites d'un accident de la route en France : la seule année 1972 comptabilise 18 034 morts officielles.

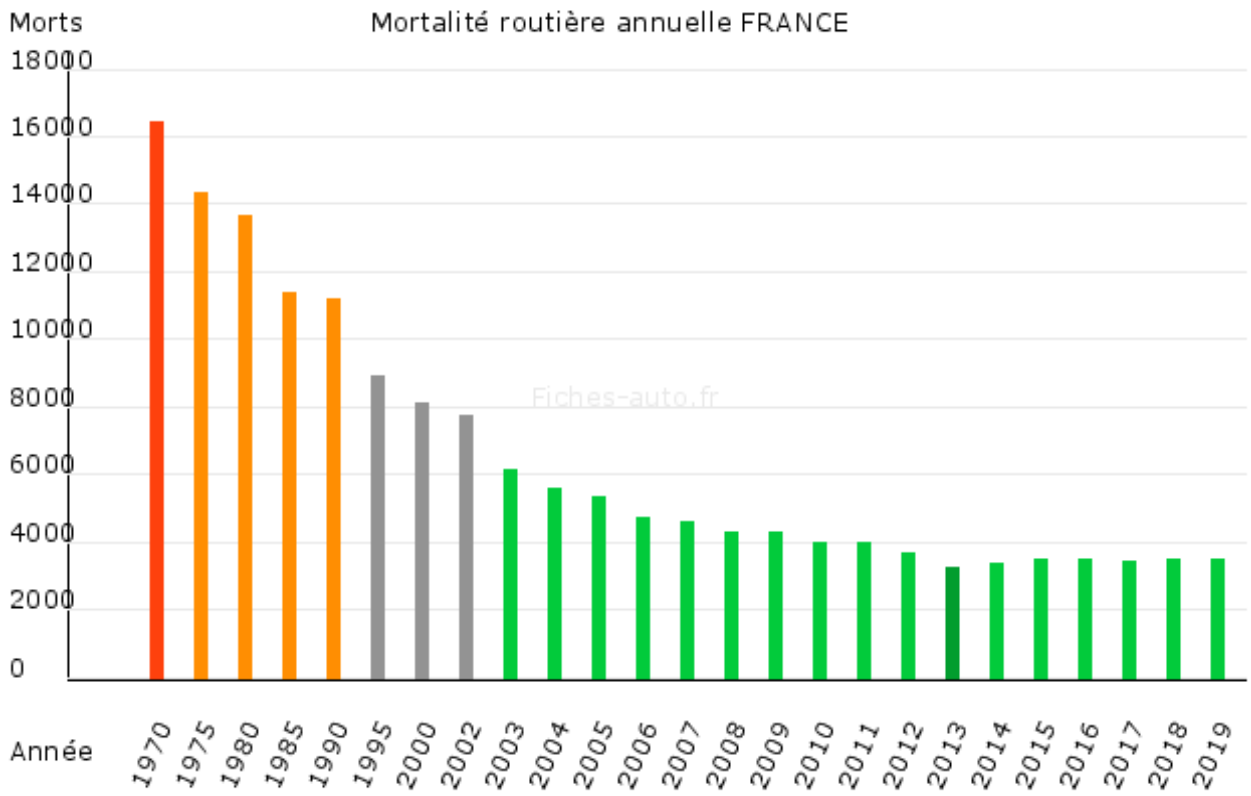

FIGURE 2.1 – Evolution de la mortalité automobile en France

Nous remarquons que le nombre de morts diminue naturellement tous les ans, et que l'ajout de système de sécurité routière comme les radars automatiques ne semblent pas contribuer à la diminution de la mortalité automobile.

Le nombre d'accidents corporels recensés par les pouvoirs publics est en baisse de 2,7% en 2015 par rapport à 2014. Alors que les blessés légers représentent la majorité des sinistres corporels (environ 71%), leur indemnisation ne représente que 5% de l'ensemble des indemnités versées. A contrario, les

blessés les plus graves ne représentent que 2% des sinistres corporels, soit environ 3 000 blessés par an pour 58% de la charge des indemnités versées.

En 2019, 3 498 personnes sont décédées sur les routes de France (métropole et outre-mer), soit 10 de plus qu'en 2018 (mortalité la plus basse enregistrée).

La France a compté 56 016 accidents corporels de la circulation. 3 244 personnes ont été tuées dans les 30 jours après leur accident, dont notamment 483 piétons, 10 usagers d'engins de déplacement personnel (tels les trottinettes électriques), 187 cyclistes, 134 cyclomotoristes, 615 motocyclistes, 1 622 automobilistes, 98 usagers de véhicules utilitaires, 36 usagers de poids lourds. 66 enfants de 14 ans ou moins sont décédés, 87 adolescents de 15-17 ans, 549 jeunes de 18-24 ans, 849 seniors de 65 ans ou plus.
Dans les départements d'Outre-mer, 162 personnes ont été tuées.
Dans les collectivités d'Outre-mer et la Nouvelle-Calédonie, on enregistre 92 tués.

2.3 Causes des accidents

Des enquêtes scientifiques menées en France entre 1983 et 2004 ont montré que dans 90% des accidents mortels, le comportement humain est en jeu ; dans 50% des accidents mortels, l'infrastructure est en jeu ; et dans 30% des accidents mortels, les facteurs liés aux véhicules sont en cause.

Certains facteurs sont relevés systématiquement par les forces de l'ordre après la survenance d'un accident : alcoolémie, choc contre un obstacle fixe, conditions météorologiques... D'autres facteurs, et non des moindres, ne sont pas systématiquement recherchés car ils sont difficiles à déterminer avec précision : vitesse, fatigue, somnolence, distraction, téléphone au volant, distances de sécurité. . .

Les causes des accidents sont multiples : facteurs humains, véhicule, conditions de circulation, état de la route, obstacles, conditions météorologiques... Mais ce sont les facteurs humains qui sont en tête dans la majorité des accidents corporels (dans 90% des cas). Parmi ceux-ci : l'alcool, la vitesse, la fatigue, la distraction et l'usage du téléphone au volant. Le cannabis multiplierait par deux la probabilité d'être responsable d'un accident mortel et par quatorze si le cannabis est associé à l'alcool. La conduite sans permis ou permis invalide est constatée dans 4% des accidents mortels.

Les causes profondes des accidents peuvent être dues au temps de trajet domicile travail et aux routes empruntées. Ainsi, les inégalités sociales seraient un facteur non négligeable dans la hausse des statistiques d'accidents mortels constatés depuis 2014.

En ville, la vitesse tue : nous estimons pour un accident entre un véhicule et un piéton qu'à 30 km/h, les blessures les plus fréquentes sont des contusions légères avec une probabilité de 15% d'être tué, alors qu'à plus de 50 km/h, la probabilité de tuer un piéton est proche de 100%. Un choc entre un véhicule et un cycliste peut avoir les mêmes conséquences, mais en plus, le cycliste chutera.

En France, en 2006, le non-port de la ceinture est présent dans 20% des accidents mortels. Un occupant portant correctement sa ceinture peut sortir indemne d'un accident, alors que le port d'une ceinture trop détendue peut conduire à heurter le volant et à des fractures de côtes dues à la collision contre la ceinture, ou encore à un traumatisme crânien avec fractures faciales et perte de connais-

sance, des fractures des genoux, fémurs ou bassin et enfin un enfoncement thoracique avec lésions sont souvent mortelles.

Jusqu'en 2006, les carrefours non giratoires sont le lieu de 23% des accidents, et 19% des accidents mortels, les virages de 40% des accidents mortels, et les obstacles sur accotement de 30% des accidents mortels. La présence d'accès riverains ou commerciaux non aménagés e représente 6%.

Chapitre 3

La base de données

Pour effectuer une étude prédictive en Machine Learning, une base de données est nécessaire.

Comme évoqué en introduction, nous utiliserons dans cette étude des données Open Data. Notre première source de données est le site www.data.gouv.fr, qui est la principale plate-forme ouverte de données publiques françaises. Lorsque l'on effectue une étude à l'aide d'Open Data, les données brutes récoltées nécessitent un traitement préalable, car elles correspondent rarement à nos besoins et ne sont pas créées pour une utilisation en particulier.

Ce traitement consiste en la recherche de valeurs manquantes, de variables ou valeurs aberrantes ou non significatives, et souvent en fonction de l'objectif final recherché, à compléter la base de donnée initiale par d'autres variables issues d'autres bases de données. Nous voyons ici un avantage non moindre des Open Data, puisque la modification des données est possible sans restriction de copyright (droits d'auteurs), à condition que ces données soient de bonne qualité.

Des analyses descriptives de notre jeu de données ont ensuite été réalisées afin de relever de premières informations sur la sinistralité automobile, avant même la mise en place de modèles de prédiction.

3.1 La base de données initiale

3.1.1 Description

Notre jeu de données répertorie l'ensemble des accidents corporels de la circulation française de 2020 et 2021. Les différentes informations relatives à chaque accident sont remplies par l'unité des forces de l'ordre intervenues sur le lieu de l'accident. Ces saisies sont rassemblées dans une fiche intitulée "Bulletin d'analyse des accidents corporels". L'ensemble de ces fiches constitue le fichier national des accidents corporels de la circulation dit "Fichier BAAC", administré par l'Observatoire National Interministériel de la Sécurité Routière "ONISR". Nos données sont accompagnées de fiches descriptives des différentes variables.
Le nombre total d'accidents enregistrés en France en 2020 est de 74665, ce qui est largement suffisant pour effectuer une étude de prédictions. Le fichier BAAC est décomposé en quatre sous-bases :

- La rubrique **USAGERS** qui décrit les individus victimes de l'accident avec 14 variables.
- La rubrique **VEHICULES** qui décrit les véhicules impliqués dans l'accident avec 10 variables.
- La rubrique **CARACTERISTIQUES** qui décrit les circonstances générales de l'accident avec 12 variables.

— La rubrique **LIEUX** qui décrit le lieu de l'accident avec 17 variables.

Chaque accident est représenté par un numéro appelé **Num_Acc** permettant de l'identifier dans les différentes bases.

3.1.2 La sous-base USAGERS

Parmi les 14 variables de la sous-base USAGERS, 13 sont qualitatives et 1 est quantitative.

— **Variables qualitatives :**

- **id_Veh :** Identifiant du véhicule repris pour chacun des usagers occupant ce véhicule (y compris les piétons qui sont rattachés aux véhicules qui les ont heurtés).

- **place :** Permet de situer la place occupée dans le véhicule par l'usager au moment de l'accident (de 1 à 9) et 10 – Piéton (non applicable).

- **catu :** Catégorie d'usager :
  1 : Conducteur
  2 : Passager
  3 : Piéton

- **grav :** Gravité de blessure de l'usager, les usagers accidentés sont classés en trois catégories de victimes plus les indemnes :
  1 : Indemne
  2 : Tué
  3 : Blessé hospitalisé
  4 : Blessé léger

- **sexe :** Sexe de l'usager.
  1 : Masculin
  2 : Féminin

- **trajet :** Motif du déplacement au moment de l'accident.
  -1 : Non renseigné
  0 : Non renseigné
  1 : Domicile – travail
  2 : Domicile – école
  3 : Courses – achats
  4 : Utilisation professionnelle
  5 : Promenade – loisirs
  9 : Autre

- **secu :** Indique l'existence et/ou l'utilisation d'un équipement de sécurité.
  -1 : Non renseigné 0 : Aucun équipement 1 : Ceinture 2 : Casque 3 : Dispositif enfants 4 : Gilet réfléchissant 5 : Airbag (2RM/3RM) 6 : Gants (2RM/3RM) 7 : Gants + Airbag (2RM/3RM) 8 : Non déterminable 9 : Autre

- **locp :** Localisation du piéton par rapport au passage piéton et description des conditions de traversée de la route.

— **Localisation du piéton :**
-1 : Non renseigné
0 : Sans objet

— **Sur chaussée :**
1 : A + 50 m du passage piéton
2 : A – 50 m du passage piéton

— **Sur passage piéton :**
3 : Sans signalisation lumineuse
4 : Avec signalisation lumineuse

— **Divers :**
5 : Sur trottoir
6 : Sur accotement
7 : Sur refuge ou BAU
8 : Sur contre allée
9 : Inconnue

- **actp :** Action du piéton :

-1 : Non renseigné
Se déplaçant
0 : Non renseigné ou sans objet
1 : Sens véhicule heurtant
2 : Sens inverse du véhicule
Divers
3 : Traversant
4 : Masqué
5 : Jouant – courant
6 : Avec animal
9 : Autre
A : Monte/descend du véhicule
B : Inconnue

- **etatp :** Précise si le piéton victime était seul ou non au moment de l'accident.
-1 : Non renseigné
1 : Seul
2 : Accompagné
3 : En groupe

— **Variable quantitative :**
- **An_nais :** Année de naissance de l'usager.

Num_Acc	id_vehicule	num_veh	place	catu	grav	sexe	an_nais	trajet	secu1	secu2	secu3	locp	actp	etatp
202000000001	154742274	B01	1	1	1	1	1983	5	1	0	-1	-1	-1	-1
202000000001	154742275	A01	1	1	3	1	1982	5	2	6	-1	-1	-1	-1
202000000002	154742273	A01	1	1	1	1	1997	1	8	0	-1	-1	-1	-1
202000000002	154742273	A01	10	3	4	1	1967	5	0	-1	-1	3	3	1
202000000003	154742271	B01	1	1	1	1	1985	0	1	0	-1	-1	-1	-1
202000000003	154742271	B01	4	2	3	1	2014	0	8	0	-1	-1	-1	-1
202000000003	154742272	A01	1	1	1	1	1963	0	1	0	-1	-1	-1	-1
202000000004	154742270	A01	1	1	1	1	1991	5	1	0	-1	0	0	-1
202000000004	154742270	A01	10	3	3	1	1991	0	0	-1	-1	2	3	1
202000000005	154742266	B01	1	1	1	2	1987	0	1	0	-1	-1	-1	-1
202000000005	154742267	A01	1	1	4	1	1983	0	1	0	-1	-1	-1	-1
202000000006	154742264	B01	1	1	3	1	2001	1	2	4	-1	-1	0	-1
202000000006	154742265	A01	1	1	1	2	1972	1	1	8	-1	-1	0	-1
202000000007	154742262	A01	1	1	1	1	1989	0	8	8	-1	-1	-1	-1
202000000007	154742263	B01	3	2	4	2	1966	3	1	0	-1	-1	-1	-1
202000000007	154742263	B01	2	2	4	2	1962	3	1	8	-1	-1	-1	-1
202000000007	154742263	B01	1	1	4	1	1987	3	1	8	-1	-1	-1	-1
202000000008	154742260	B01	1	1	1	1	1980	3	1	0	-1	-1	-1	-1
202000000008	154742261	A01	1	1	3	1	1951	5	2	6	-1	-1	-1	-1
202000000009	154742259	A01	1	1	1	2	1965	1	1	8	-1	-1	-1	-1
202000000009	154742259	A01	10	3	3	1	2009	2	0	-1	-1	3	3	1
202000000010	154742257	A01	1	1	3	1	1991	0	8	0	-1	0	0	-1
202000000010	154742258	B01	1	1	4	1	1977	1	1	0	-1	0	0	-1
202000000011	154742255	A01	1	1	1	2	1934	5	1	8	-1	-1	0	-1
202000000011	154742256	B01	1	1	4	1	2005	2	8	0	-1	-1	0	-1
202000000012	154742254	A01	1	1	1	1	1999	0	8	0	-1	0	0	-1
202000000012	154742254	A01	10	3	3	1	1988	5	8	8	-1	1	9	2

FIGURE 3.1 – Aperçu de la sous-base USAGERS

3.1.3 La sous-base VEHICULES

Parmi les 10 variables de la sous-base VEHICULES, 9 sont qualitatives et 1 est quantitative.

— **Variables qualitatives :**

- **id_vehicule :** Identifiant unique du véhicule repris pour chacun des usagers occupant ce véhicule (y compris les piétons qui sont rattachés aux véhicules qui les ont heurtés).

- **Num_Veh :** Identifiant du véhicule repris pour chacun des usagers occupant ce véhicule (y compris les piétons qui sont rattachés aux véhicules qui les ont heurtés).

- **senc :** Sens de circulation :
  -1 : Non renseigné.
  0 : Inconnu.
  1 : PK ou PR ou numéro d'adresse postale croissant.
  2 : PK ou PR ou numéro d'adresse postale décroissant.
  3 : Absence de repère.

- **catv :** Catégorie du véhicule :
  00 : – Indéterminable
  01 : – Bicyclette

02 : – Cyclomoteur <50cm3
03 : – Voiturette (Quadricycle à moteur carrossé) (anciennement "voiturette ou tricycle à moteur")
04 : – Référence inutilisée depuis 2006 (scooter immatriculé)
05 : – Référence inutilisée depuis 2006 (motocyclette)
06 : – Référence inutilisée depuis 2006 (side-car)
07 : – VL seul
08 : – Référence inutilisée depuis 2006 (VL + caravane)
09 : – Référence inutilisée depuis 2006 (VL + remorque)
10 : – VU seul 1,5T <= PTAC <= 3,5T avec ou sans remorque (anciennement VU seul 1,5T <= PTAC <= 3,5T)
11 : – Référence inutilisée depuis 2006 (VU (10) + caravane)
12 : – Référence inutilisée depuis 2006 (VU (10) + remorque)
13 : – PL seul 3,5T <PTCA <= 7,5T
14 : – PL seul > 7,5T
15 : – PL > 3,5T + remorque
16 : – Tracteur routier seul
17 : – Tracteur routier + semi-remorque
18 : – Référence inutilisée depuis 2006 (transport en commun)
19 : – Référence inutilisée depuis 2006 (tramway)
20 : – Engin spécial
21 : – Tracteur agricole
30 : – Scooter < 50 cm3
31 : – Motocyclette > 50 cm 3 et <= 125 cm 3
32 : – Scooter > 50 cm 3 et <= 125 cm 3
33 : – Motocyclette > 125 cm 3
34 : – Scooter > 125 cm 3
35 : – Quad léger <= 50 cm 3 (Quadricycle à moteur non carrossé)
36 : – Quad lourd > 50 cm 3 (Quadricycle à moteur non carrossé)
37 : – Autobus
38 : – Autocar
39 : – Train
40 : – Tramway
41 : – 3RM <= 50 cm3
42 : – 3RM > 50 cm3 <= 125 cm3
43 : – 3RM > 125 cm3
50 : – EDP à moteur
60 : – EDP sans moteur
80 : – VAE
99 : – Autre véhicule

- **obs :** Obstacle fixe heurté :
-1 : – Non renseigné
0 : – Sans objet
1 : – Véhicule en stationnement
2 : – Arbre
3 : – Glissière métallique

4 : – Glissière béton
5 : – Autre glissière
6 : – Bâtiment, mur, pile de pont
7 : – Support de signalisation verticale ou poste d'appel d'urgence
8 : – Poteau
9 : – Mobilier urbain
10 : – Parapet
11 : – Ilot, refuge, borne haute
12 : – Bordure de trottoir
13 : – Fossé, talus, paroi rocheuse
14 : – Autre obstacle fixe sur chaussée
15 : – Autre obstacle fixe sur trottoir ou accotement
16 : – Sortie de chaussée sans obstacle
17 : – Buse – tête d'aqueduc

- **obsm :** Obstacle mobile heurté :
-1 : – Non renseigné
0 : – Aucun
1 : – Piéton
2 : – Véhicule
4 : – Véhicule sur rail
5 : – Animal domestique
6 : – Animal sauvage
9 : – Autre

- **choc :** Point de choc initial :
-1 : – Non renseigné
0 : – Aucun
1 : – Avant
2 : – Avant droit
3 : - Avant gauche
4 : – Arrière
5 : – Arrière droit
6 : – Arrière gauche
7 : – Côté droit
8 : - Côté gauche
9 : – Chocs multiples (tonneaux)

- **manv :**
Manoeuvre principale avant l'accident :
-1 : – Non renseigné
0 : – Inconnue
1 : – Sans changement de direction
2 : – Même sens, même file
3 : - Entre 2 files
4 : – En marche arrière
5 : – A contresens

6 : – En franchissant le terre-plein central
7 : – Dans le couloir bus, dans le même sens
8 : - Dans le couloir bus, dans le sens inverse
9 : – En s'insérant
10 : – En faisant demi-tour sur la chaussée

Changeant de file :
11 : – A gauche
12 : – A droite

Déporté :
13 : – A gauche
14 : – A droite

Tournant :
15 : – A gauche
16 : – A droite

Dépassant :
17 : – A gauche
18 : – A droite

Divers :
19 : – Traversant la chaussée
20 : – Manœuvre de stationnement
21 : – Manœuvre d'évitement
22 : – Ouverture de porte
23 : - Arrêté (hors stationnement)
24 : - En stationnement (avec occupants)
25 : - Circulant sur trottoir
26 : - Autres manœuvres

- **motor :** Type de motorisation du véhicule :
-1 : – Non renseigné
0 : – Inconnue
1 : – Hydrocarbures
2 : – Hybride électrique
3 : - Electrique
4 : – Hydrogène
5 : – Humaine
6 : – Autre

— **Variable quantitative :**
- **occutc :** Nombre d'occupants dans le transport en commun.

Num_Acc	id_vehicule	num_veh	senc	catv	obs	obsm	choc	manv	motor	occutc
202000000001	154742274	B01	1	7	0	2	2	15	1	
202000000001	154742275	A01	1	33	0	2	1	2	1	
202000000002	154742273	A01	3	7	0	1	2	26	1	
202000000003	154742271	B01	1	7	0	2	1	1	1	
202000000003	154742272	A01	1	7	0	2	8	1	0	
202000000004	154742270	A01	3	7	0	1	3	1	1	
202000000005	154742266	B01	1	7	0	2	2	2	1	
202000000005	154742267	A01	1	7	0	2	5	2	1	
202000000006	154742264	B01	2	2	0	2	2	1	1	
202000000006	154742265	A01	1	10	0	2	1	9	1	
202000000007	154742262	A01	1	7	0	2	2	21	1	
202000000007	154742263	B01	1	7	0	2	6	2	1	
202000000008	154742260	B01	3	7	0	2	3	15	1	
202000000008	154742261	A01	2	2	0	2	1	19	1	
202000000009	154742259	A01	1	7	0	1	2	16	1	
202000000010	154742257	A01	1	7	0	2	1	13	1	
202000000010	154742258	B01	2	7	0	2	2	21	1	
202000000011	154742255	A01	2	7	0	2	2	15	1	
202000000011	154742256	B01	2	1	0	2	5	19	5	
202000000012	154742254	A01	2	7	0	1	3	26	1	
202000000013	154742253	A01	1	7	13	0	1	1	1	
202000000014	154742248	B01	2	1	0	2	4	16	5	
202000000014	154742249	A01	2	7	0	2	2	16	1	
202000000015	154742247	A01	2	7	9	0	1	19	1	
202000000016	154742243	C01	2	7	0	2	1	2	1	
202000000016	154742244	D01	2	7	0	2	1	2	1	
202000000016	154742245	A01	2	7	0	2	1	2	1	
202000000016	154742246	B01	2	7	0	2	4	2	1	
202000000017	154742242	A01	3	30	8	0	3	1	1	
202000000018	154742240	A01	3	7	0	2	6	1	1	
202000000018	154742241	B01	3	7	0	2	1	1	1	
202000000019	154742238	A01	1	7	13	0	2	1	1	

FIGURE 3.2 – Aperçu de la sous-base VEHICULES

3.1.4 La sous-base LIEUX

Parmi les 18 variables de la sous-base LIEUX, 14 sont qualitatives et 4 sont quantitatives.
— **Variables qualitatives :**

- **catr :** Catégorie de route :
  1 : Autoroute
  2 : Route nationale
  3 : Route Départementale
  4 : Voie Communales
  5 : Hors réseau public
  6 : Parc de stationnement ouvert à la circulation publique
  7 : Routes de métropole urbaine
  9 : autre

- **voie :** Numéro de la route.

- **V1 :** Indice numérique du numéro de route (exemple : 2 bis, 3 ter etc.).

- **V2 :** Lettre indice alphanumérique de la route.

- **circ :** Régime de circulation :
  -1 : Non renseigné
  1 : A sens unique
  2 : Bidirectionnelle
  3 : A chaussées séparées
  4 : Avec voies d'affectation variable

- **vosp :** Signale l'existence d'une voie réservée, indépendamment du fait que l'accident ait lieu ou non sur cette voie.
  -1 : Non renseigné
  0 : Sans objet
  1 : Piste cyclable
  2 : Bande cyclable
  3 : Voie réservée

- **prof :** Profil en long décrit la déclivité de la route à l'endroit de l'accident :
  -1 : Non renseigné
  1 : Plat
  2 : Pente
  3 : Sommet de côte
  4 : Bas de côte

- **pr :** Numéro du PR de rattachement (numéro de la borne amont). La valeur -1 signifie que le PR n'est pas renseigné.

- **plan :** Tracé en plan :
  -1 : Non renseigné
  1 : Partie rectiligne
  2 : En courbe à gauche
  3 : En courbe à droite
  4 : En « S »

- **surf :** Etat de la surface :
  -1 : Non renseigné
  1 : Normale
  2 : Mouillée
  3 : Flaques
  4 : Inondée
  5 : Enneigée
  6 : Boue
  7 : Verglacée
  8 : Corps gras – huile
  9 : Autre

- **infra :** Aménagement - Infrastructure :
  -1 : Non renseigné
  0 : Aucun

1 : Souterrain - tunnel
2 : Pont - autopont
3 : Bretelle d'échangeur ou de raccordement
4 : Voie ferrée
5 : Carrefour aménagé
6 : Zone piétonne
7 : Zone de péage
8 : Chantier
9 : Autres

- **situ :** Situation de l'accident :
-1 : Non renseigné
0 : Aucun
1 : Sur chaussée
2 : Sur bande d'arrêt d'urgence
3 : Sur accotement
4 : Sur trottoir
5 : Sur piste cyclable
6 : Sur autre voie spéciale
8 : Autres

- **vma :** Vitesse maximale autorisée sur le lieu et au moment de l'accident.

— **Variables quantitatives :**

- **nbv :** Nombre total de voies de circulation.

- **pr1 :** Distance en mètres au PR (par rapport à la borne amont). La valeur -1 signifie que le PR n'est pas renseigné.

- **lartpc :** Largeur du terre-plein central (TPC) s'il existe (en m).

- **larrout :** Largeur de la chaussée affectée à la circulation des véhicules ne sont pas compris les bandes d'arrêt d'urgence, les TPC et les places de stationnement (en m).

Num_Acc	catr	voie	v1	v2	circ	nbv	vosp	prof	pr	pr1	plan	lartpc	larrout	surf	infra	situ	vma
202000000001	4	HENRI BARBUSSE (AVENUE)	0	N/A	2	2	0	1	0	0	1		-1	1	0	1	50
202000000002	4	MOUSSEAUX(CHEMIN)	0	N/A	2	2	0	1	0	100	3		-1	1	0	1	50
202000000003	4	CARNOT(AVENUE)	0	N/A	-1	2	0	1	0	0	1		-1	1	0	1	50
202000000004	4	VICTOR HUGO (AVENUE)	0	N/A	2	2	0	1	1	0	1		-1	1	0	1	30
202000000005	3	35	0	N/A	1	1	0	1	4	674	2		-1	1	3	8	50
202000000006	3	D6015	0	N/A	2	3	0	1	66	170	1		-1	2	0	1	80
202000000007	3	DE GAULLE (AVE DU GLE)A PARTIR DES NR	0	N/A	2	6	0	1	3	600	1		-1	1	0	1	50
202000000008	4	N/A	0	N/A	2	4	0	1	(1)	(1)	1		-1	2	5	1	50
202000000009	3	928	0	N/A	2	2	0	1	6	0	1		-1	1	5	1	50
202000000010	3	941	0	N/A	2	2	0	1	134	400	1		5	1	0	1	80
202000000011	7	22	0	N/A	2	3	0	1	(1)	(1)	3		-1	1	0	1	50
202000000012	4	LECLERC (RUE DU MARECHAL)	0	N/A	2	2	0	1	(1)	(1)	1		-1	1	0	1	50
202000000013	4	N/A	0	N/A	3	2	0	1	0	0	1		-1	1	0	1	50
202000000014	4	N/A	0	N/A	2	2	1	1	(1)	(1)	3		-1	1	0	1	50
202000000015	3	GEORGES CLEMENCEAU (AVENUE)	0	N/A	2	2	0	2	0	0	3		-1	2	0	4	50
202000000016	4	VERN (DE)	0	N/A	2	2	0	1	0	0	1		-1	1	5	1	50
202000000017	4	GRIMAULT (BD LEON)	0	N/A	3	2	2	1	(1)	(1)	1		-1	1	0	5	50
202000000018	4	CHATEAUGIRON (DE)	0	N/A	2	4	0	1	0	0	1		-1	2	5	1	50
202000000019	3	5	0	N/A	2	2	0	2	21	203	3		5	2	0	3	80
202000000020	4	40 avenue du 8 Mai	0	N/A	2	2	0	1	(1)	(1)	1		-1	1	0	1	50
202000000021	4	N/A	0	N/A	2	2	0	1	0	0	1		-1	1	0	1	50
202000000022	4	LUXEMBOURG AVENUE DE	0	N/A	2	2	0	1	0	0	1		-1	1	0	1	50
202000000023	4	N/A	0	N/A	2	4	0	2	0	0	1		-1	1	2	1	50
202000000024	4	N/A	0	N/A	1	2	0	4	0	0	1		-1	1	0	1	50
202000000025	4	CANNES (ROUTE DE)	0	N/A	2	2	0	1	0	0	1		-1	1	0	1	50
202000000026	4	VAL FLEURI CHEMIN DU	0	N/A	2	2	0	1	0	0	1		-1	1	0	1	50
202000000027	4	CURIE PIERRE ET MARIE AVENUE	0	N/A	2	2	0	1	0	0	1		-1	1	0	1	50
202000000028	4	N/A	0	N/A	1	2	3	1	0	0	1		-1	1	0	1	50
202000000029	4	N/A	0	N/A	1	1	0	2	0	0	1		-1	8	0	1	30
202000000030	4	N/A	0	N/A	1	1	0	1	0	0	1		-1	2	0	1	50
202000000031	4	N/A	0	N/A	3	4	0	1	0	0	1		-1	2	0	1	50
202000000032	4	N/A	0	N/A	2	4	1	1	0	0	1		-1	1	0	5	30
202000000033	4	N/A	0	N/A	2	4	3	1	0	0	1		-1	1	0	6	50
202000000034	4	N/A	0	N/A	1	2	1	1	0	0	1		-1	1	0	5	30

FIGURE 3.3 – Aperçu de la sous-base LIEUX

3.1.5 La sous-base CARACTERISTIQUES

Parmi les 15 variables de la sous-base CARACTERISTIQUES 10 sont qualitatives et 5 sont quantitatives.

— **Variables qualitatives :**

- **hrmn :** Heure et minutes de l'accident.

- **lum :** Lumière : conditions d'éclairage dans lesquelles l'accident s'est produit :
  1 : Plein jour
  2 : Crépuscule ou aube
  3 : Nuit sans éclairage public
  4 : Nuit avec éclairage public non allumé
  5 : Nuit avec éclairage public allumé.

- **dep :** Département : Code INSEE (Institut National de la Statistique et des Etudes Economiques) du département (2A Corse-du-Sud – 2B Haute-Corse).

- **com :** Commune : Le numéro de commune est un code donné par l'INSEE. Le code est composé du code INSEE du département suivi par 3 chiffres.

- **agg :** Localisation :
  1 : Hors agglomération
  2 : En agglomération

- **int :** Intersection :
  1 : Hors intersection
  2 : Intersection en X
  3 : Intersection en T
  4 : Intersection en Y
  5 : Intersection à plus de 4 branches
  6 : Giratoire
  7 : Place
  8 : Passage à niveau
  9 : Autre intersection

- **atm :** Conditions atmosphériques :
  -1 : Non renseigné
  1 : Normale
  2 : Pluie légère
  3 : Pluie forte
  4 : Neige - grêle
  5 : Brouillard - fumée
  6 : Vent fort - tempête
  7 : Temps éblouissant
  8 : Temps couvert
  9 : Autre

- **col :** Type de collision :
  -1 : Non renseigné
  1 : Deux véhicules - frontale
  2 : Deux véhicules – par l'arrière
  3 : Deux véhicules – par le coté
  4 : Trois véhicules et plus – en chaîne
  5 : Trois véhicules et plus - collisions multiples
  6 : Autre collision
  7 : Sans collision

- **adr :** Adresse postale : variable renseignée pour les accidents survenus en agglomération.

— **Variables quatitatives :**

- **jour :** Jour de l'accident.
- **mois :** Mois de l'accident.
- **an :** Année de l'accident.
- **lat :** Latitude.
- **long :** Longitude.

Num_Acc	jour	mois	an	hrmn	lum	dep	com	agg	int	atm	col	adr	lat	long
202000000001	7	3	2020	16:55	1	91	91657	2	3	1	3	HENRI BARBUSSE (AVENUE)	48,7053500	2,4384100
202000000002	7	3	2020	08:35	2	91	91657	2	9	7	6	MOUSSEAUX(CHEMIN)	48,6900000	2,4100000
202000000003	7	3	2020	13:30	1	91	91174	2	2	1	3	CARNOT(AVENUE)	48,6106700	2,4758200
202000000004	7	3	2020	18:50	5	91	91215	2	1	1	6	VICTOR HUGO (AVENUE)	48,6978200	2,5244600
202000000005	7	3	2020	11:00	1	77	77181	1	6	1	2	LAGNY (RUE DE) - D35	48,8286457	2,7059707
202000000006	5	3	2020	07:00	2	76	76114	1	2	2	3	D6015	49,5763988	0,5101875
202000000007	6	3	2020	20:29	5	68	68297	2	1	1	2	Avenue du General de Gaulle	47,5824950	7,5563270
202000000008	7	3	2020	10:20	1	64	64024	2	3	8	3	Boulevard du Bayonne Anglet Biarritz	43,4938100	-1,5230700
202000000009	6	3	2020	08:15	1	62	62757	2	2	1	6	CALAIS (RUE DE ....19A 71 ET 28 A 110)	50,7570746	2,2325420
202000000010	4	3	2020	04:05	3	62	62540	1	1	1	1	CD941	50,4582925	2,5756715
202000000011	5	3	2020	11:50	1	59	59143	2	3	1	-1	BEAUCHAMP (RUE LEON)	50,6785000	2,8742000
202000000012	7	3	2020	05:00	4	56	56260	2	1	1	6	LECLERC(RUE DU MARECHAL)	47,6581100	-2,7513900

FIGURE 3.4 – Aperçu de la sous-base CARACTERISTIQUES

3.1.6 Analyses

La gravité des quatre sous-bases
Comme dit précédemment, notre jeu de données nécessite un traitement préalable.

En effet, certaines variables contiennent des modalités aberrantes. Par exemple, certaines variables ont des valeurs non explicitées par le descriptif de l'ONISR. Il y est précisé que les données non renseignées ont été remplacées par des zéros, des points, ou laissées vides.

Lors de l'importation des données sous python, le logiciel considère les données contenant des points ou des cases vides comme des valeurs manquantes, "NaN". Concernant les zéros aberrants, il a fallu les remplacer par des valeurs manquantes, variable par variable. De ce fait, dans certains cas, cela a significativement augmenté le nombre de valeurs manquantes pour la variable en question.

Pour certaines variables, la difficulté principale a été de remarquer que certaines modalités inexistantes ne sont pas des valeurs manquantes.

Pour la variable place qui indique la position de l'usager dans le véhicule, la modalité "0" signifie simplement que l'individu est un piéton, et inversement, la modalité "0" dans la variable actp qui décrit l'action du piéton avant l'accident, indique que l'individu est dans un véhicule.

Suite à ce travail, nous analysons maintenant la proportion des valeurs manquantes dans chacune des sous-bases.

Les graphiques suivants sont composés de deux parties : celle de gauche représente la proportion de valeurs manquantes par variable, celle de droite représente la répartition des valeurs manquantes par combinaison de variables et se lit ligne par ligne. Elle permet notamment de détecter l'existence de tendances dans la répartition des valeurs manquantes et va ainsi nous donner une information supplémentaire pour le traitement de celles-ci dans la partie gestion des valeurs manquantes.

— **La sous-base USAGERS :**

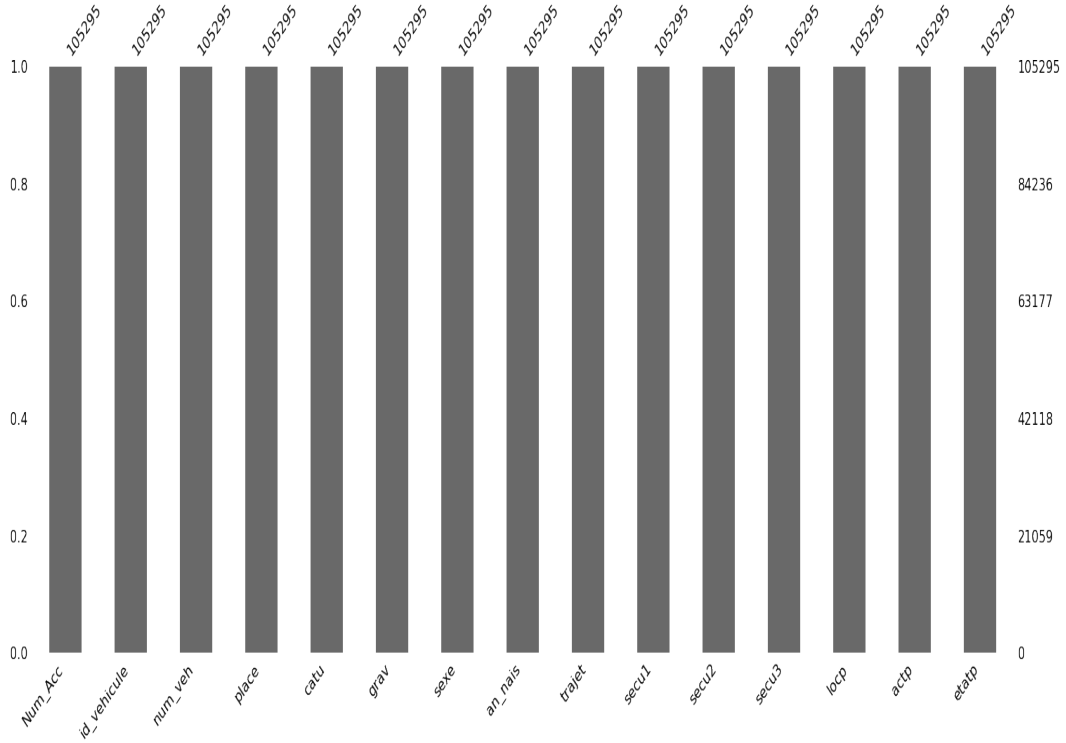

FIGURE 3.5 – Proportion de valeurs manquantes par variable de la sous-base USAGERS

— **La sous-base VEHICULES :**

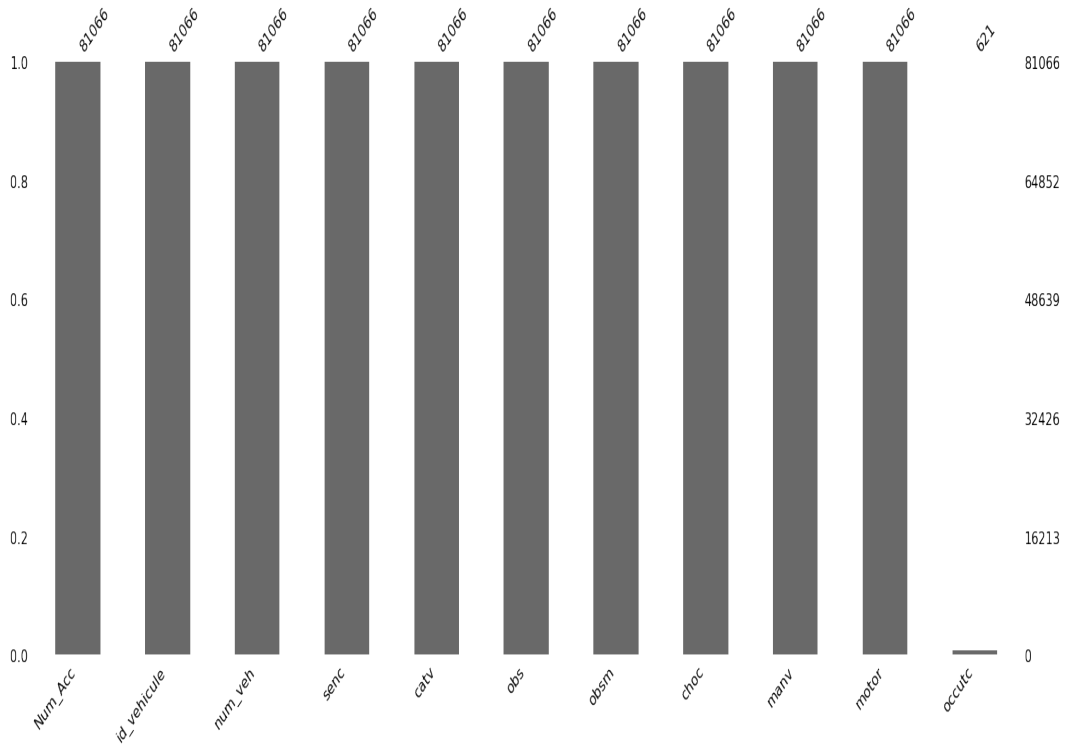

FIGURE 3.6 – Proportion de valeurs manquantes par variable de la sous-base VEHICULES

On observe essentiellement qu'il n'existe pas de valeurs manquantes dans les sous-bases USAGERS et VEHICULES.

— **La sous-base CARACTERISTIQUES :**

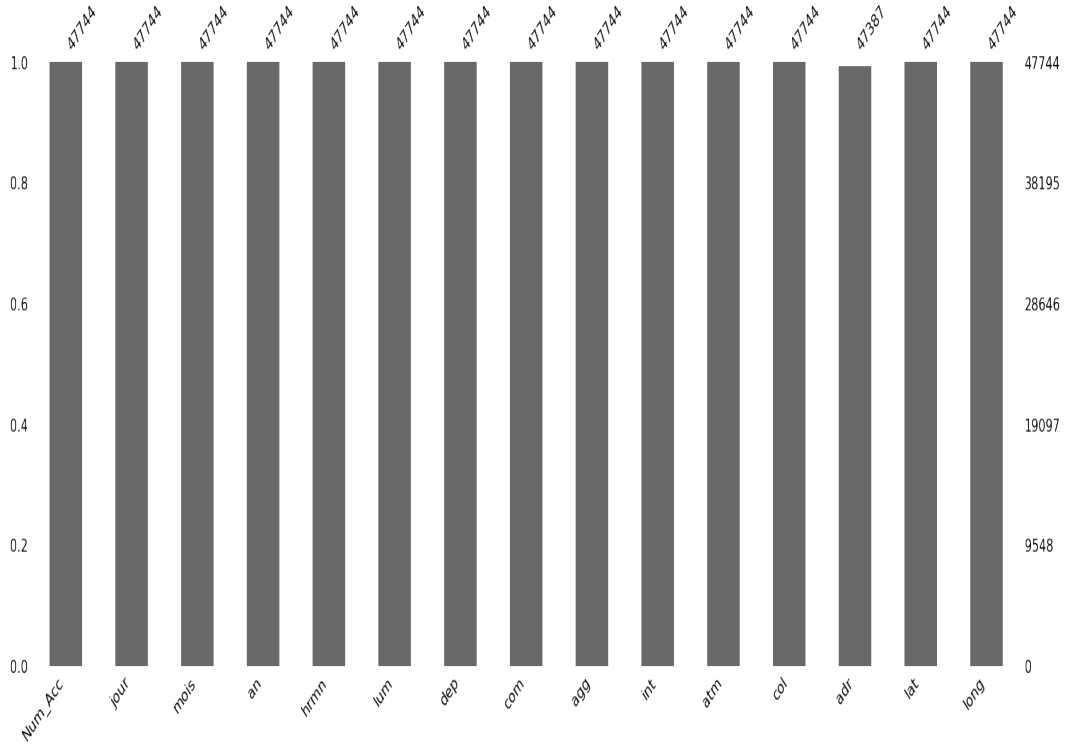

FIGURE 3.7 – Proportion de valeurs manquantes par variable de la sous-base

CARACTERISTIQUES

Nous observons principalement une variable unique qui contient 357 valeurs manquantes : adr. Elles ne correspondent qu'à 0,74% des caractéristiques.

— **La sous-base LIEUX :**

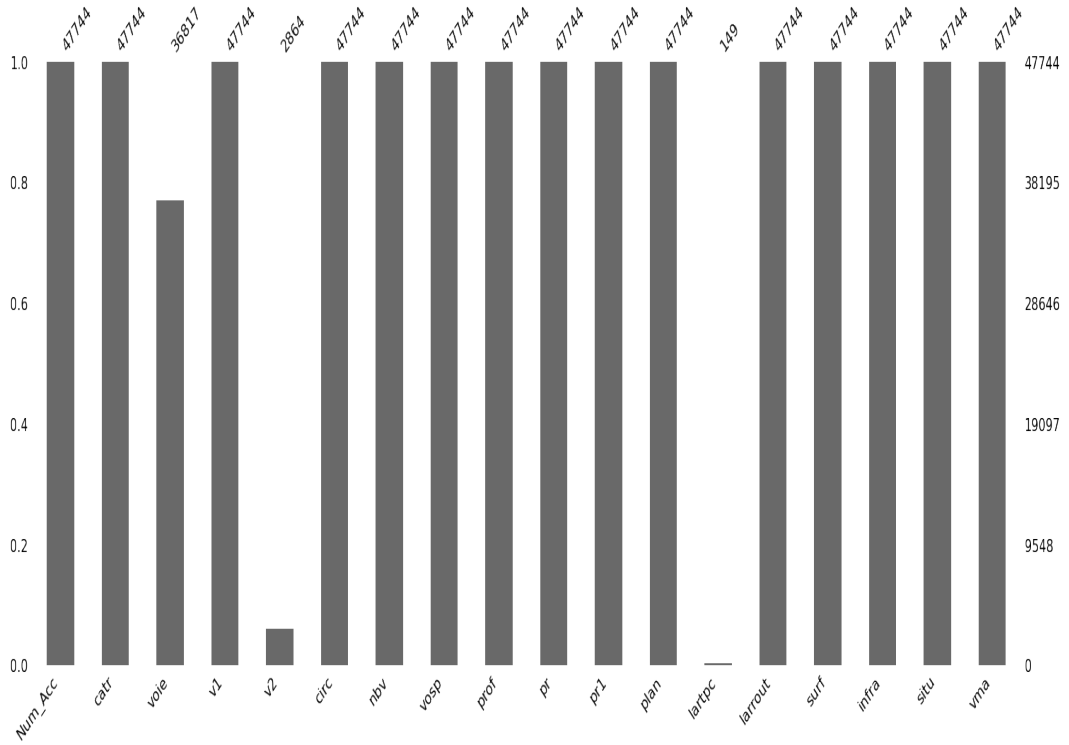

FIGURE 3.8 – Proportion de valeurs manquantes par variable de la sous-base LIEUX

Il existe principalement deux variables qui ont plus de 90% de valeurs manquantes : v2, lartpc et une variable unique qui a 22.88% de valeurs manquantes : voie.

3.1.7 Création de la base des données 2020

Dans notre situation, les données que nous souhaitons utiliser sont présentées dans plusieurs fichiers. Nous avons souvent besoin de combiner ces fichiers en un seul DataFrame pour analyser les données. Les pandas fournissent de telles fonctionnalités pour combiner facilement DataFrame avec divers types de logique d'ensemble pour les index et la fonctionnalité d'algèbre relationnelle dans le cas d'opérations de type jointure / fusion.

Il s'agit de **"Merge"** quatre tableaux partageant la même clé **"Num_Acc"**. Le résultat de la fusion est une nouvelle base de données qui combine les informations d'entrée.

— **La base 2020 :**

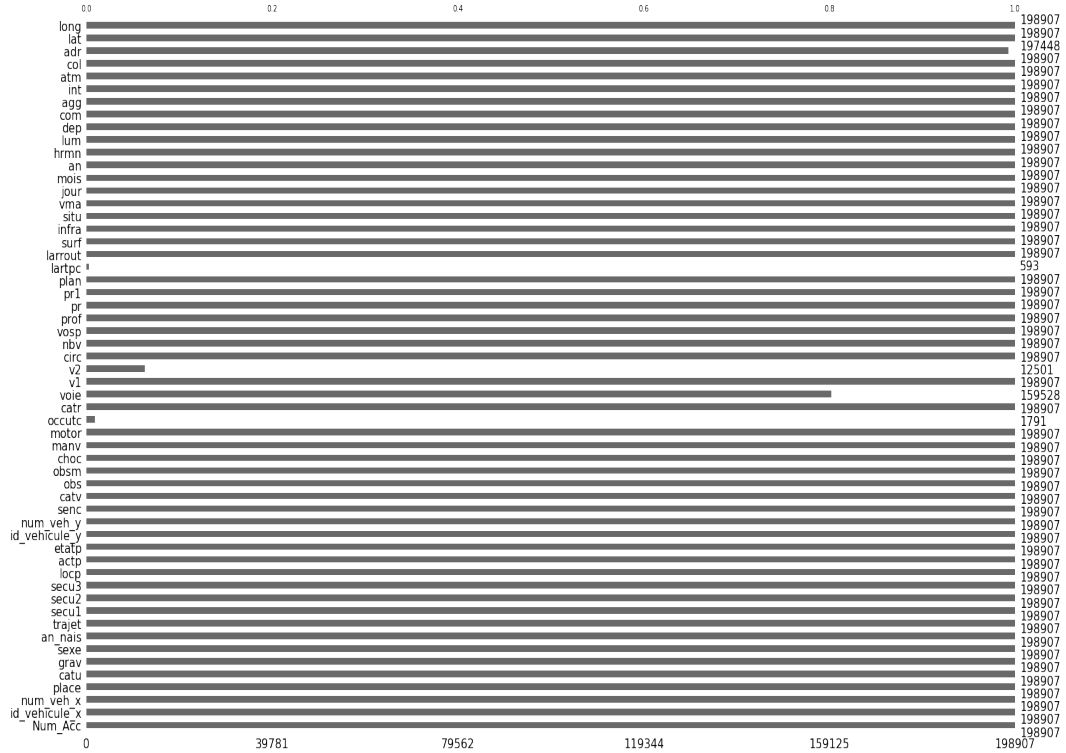

FIGURE 3.9 – Proportion de valeurs manquantes par variable de la base 2020
On observe principalement trois variables dont les valeurs manquantes sont supérieures à 90% : occutc, v2 et lartpc et deux variables dont les valeurs manquantes sont inférieures à 19% : voie et adr.

3.2 La nouvelle base de données

3.2.1 Mise en perspective

Notre objectif principal dans cette étude est d'expliquer la sinistralité automobile. Il est donc important de tenir compte de cela lors du traitement de notre base initiale. Les variables **v2** et **lartpc** ne représentent que 1% de notre jeu de données et sont difficiles à garder pour notre étude puisqu'ils engendrent beaucoup de valeurs manquantes dans les autres variables.

3.2.2 Modification de la base de données

Rappelons que dans bon nombre d'études statistiques, il est courant de voir les données manquantes d'une variable remplacées par une moyenne lorsqu'elle est quantitative et par le mode de celle-ci lorsqu'elle est qualitative. Le logiciel python, par exemple, dans la méthode d'analyse des composantes principales, adopte cette méthodologie lorsqu'elle observe des données manquantes dans un jeu de données. Cette méthode d'imputation par la moyenne ou le mode, ou encore le fait de remplacer des valeurs manquantes par des zéros, est susceptible d'altérer l'information contenue dans la base de données. En effet, la corrélation entre les variables peut être modifiée.

Ainsi, il ne nous semblait pas pertinent d'utiliser cette méthode d'imputation dans notre étude.

Le retraitement des variables contenant des valeurs manquantes sera explicité dans la partie "Gestion des valeurs manquantes".

D'autre part, Nous avons dans notre base de données 56 variables pour 198 907 lignes, c'est beaucoup et peut être un frein à la construction de bons modèles. Des fonctionnalités non pertinentes peuvent avoir un impact négatif sur les performances du modèle, tout comme la suppression de fonctionnalités significatives.

Nous souhaitons réduire le nombre de variables pour les raisons suivantes :

— Réduire les données nécessaires pour créer un modèle performant (moins d'exploration nécessaire).

— Moins de temps et d'énergie consacrés à l'exécution de l'algorithme.

— Réduire le bruit et les structures aléatoires qui semblent logiques mais peuvent favoriser le surapprentissage.

Pour finir, selon l'utilisation de la base de données, nous avons supprimé certaines variables.

Création et suppression des variables :

— **secu :** Variable créée indiquant si la moyenne est inférieure à 3 alors il y a une faible présence et utilisation d'équipement de sécurité, si la moyenne égale à 3 alors il y a une présence et utilisation moyenne d'équipement de sécurité et si la moyenne est supérieur à 3 donc il y a une forte présence et utilisation d équipement de sécurité.

— **locp :** Variable créée indiquant la localisation du piéton.
si la localisation n'est pas connu ou sans objet on donne une variable nulle.
si la localisation sur chaussée (=1 ou 2) on donne une variable égale à 1.
si la localisation sur passage piéton (=3 ou 4) on donne une variable égale à 2.
si la localisation est divers (=5 ou 6,7,8,9) on donne une variable égale à 3.

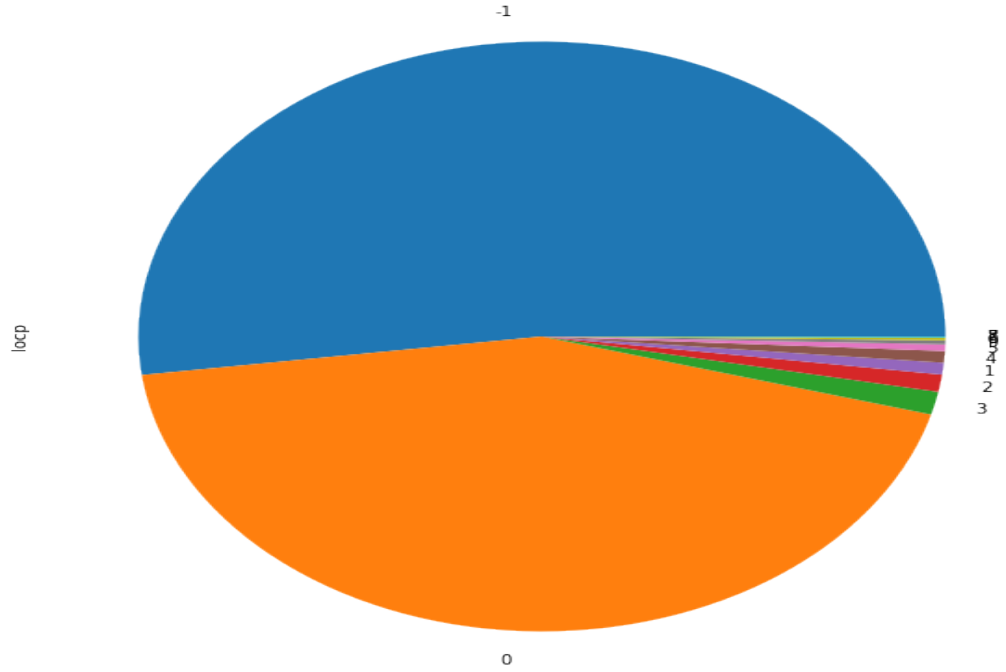

FIGURE 3.10 – Graphique circulaire de la variable locp

Nous voyons que la plupart des données concernent la variable -1. Il se traduit par le faite que la localisation du piéton est non renseigné. Si les futurs modèles se révèlent trop complexes (et ont de mauvaises performances), on pourra toujours se cantonner à cette variable. On pourrait aussi généraliser encore plus pour regrouper dans une même catégorie.

— **catv :** Variable créée indiquant le type de véhicule impliqué. En lisant la description des données, nous pouvons voir que certaines valeurs ne sont pas utilisées. Nous pouvons facilement les définir tous sur une valeur partagée utilisée pour le type NULL ou OTHER. Dans ce cas, il s'agit de la valeur 99. Pour les valeurs restantes, on peut regrouper des véhicules communs avec le même identifiant. Par exemple, nous pouvons mettre le petit cyclomoteur et le scooter à la même valeur.

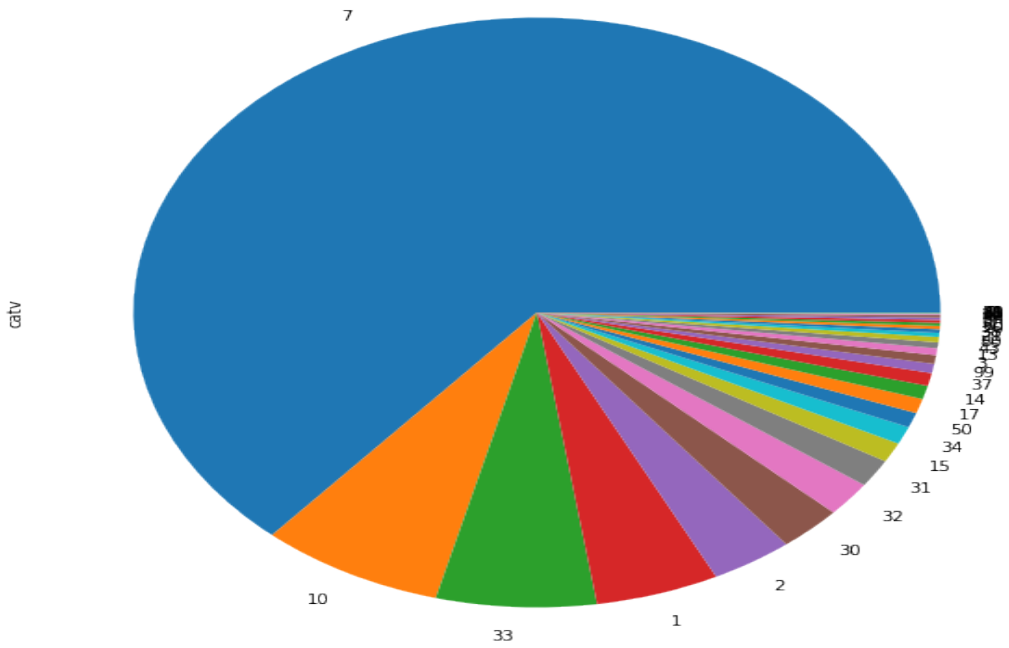

FIGURE 3.11 – Graphique circulaire de la variable catv

Nous voyons que la plupart des données concernent le type de véhicule 07. Il se traduit par véhicule léger et désigne la voiture de tous les jours. Si les futurs modèles se révèlent trop complexes (et ont de mauvaises performances), on pourra toujours se cantonner à ce type de véhicule. On pourrait aussi généraliser encore plus pour regrouper certains véhicules dans une même catégorie. Nous devons également être conscients que les futurs modèles peuvent avoir du mal à prédire correctement la classe pour les véhicules qui ne sont pas du type 07, car ce type de véhicule est omniprésent et le modèle sera incité à prédire correctement pour ce type de véhicule avec moins de soin pour autres types de véhicules. En effet, se concentrer sur le type le plus représenté générera le meilleur score en moyenne.

— **obs :** Passage de 19 à 6 modalités.

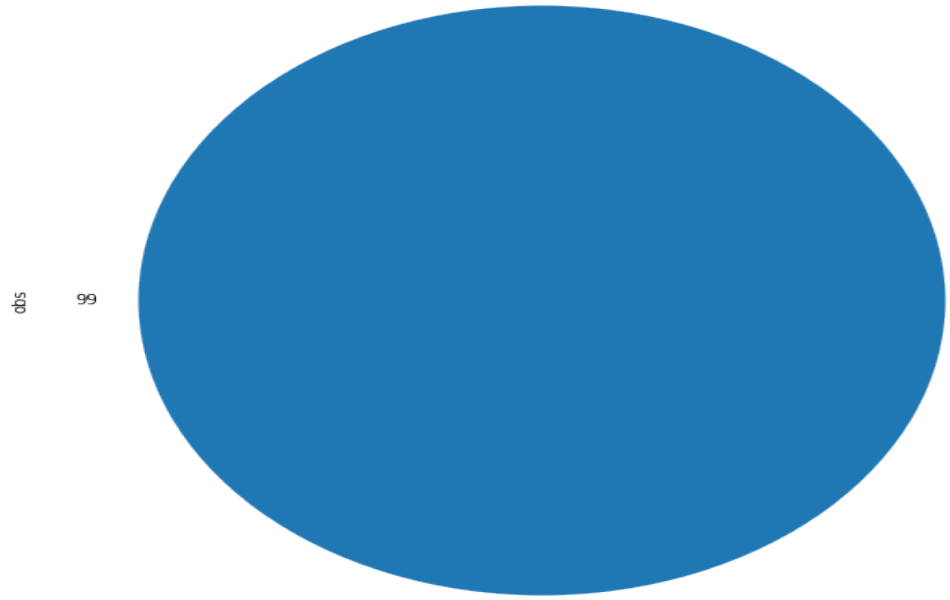

FIGURE 3.12 – Graphique circulaire de la variable obs

Nous voyons que tous des données concernent l'obstacle Buse – tête d'aqueduc, nous pouvons supprimer la variable **obs**.

— **choc :** Passage de 11 à 4 modalités.
On le mettra à 1 si son point de choc initial est (valeur = 1,2,3) et à 4 si (valeur = 3,4,5,6) et 9 si (valeur = 7,8, 9) et à 99 (valeur = -1, 0).
On peut considérer que la variable **choc** a plusieurs valeurs et la supprime.

— **grav :** Variable créée indiquant Gravité de blessure de l'usager.
Nous le mettrons à 1 si la gravité de l'accident est mortelle (valeur=2) et à 0 sinon (valeur=1,3,4).

— Nous pouvons supprimer les variables dont nous pensons qu'elles ne sont pas nécessaires pour la prédiction :
les variables **id_vehicule_x, id_vehicule_y, an_nais, actp, dep, com, etatp, larrout, an, jour, adr, lat, long, v1, vosp, pr, pr1, num_veh_x, num_veh_y, senc, situ** et **manv**.

— On peut considérer que la variable **infra** a la même valeur pour tous les attributs car plus de 82% ont une valeur de 0 que l'on peut suprimer.

3.2.3 Analyse univariée (distribution des valeurs des variables)

Dans cette partie, nous allons explorer les variables une par une. Nous devons garder à l'esprit que le fait de ne regarder qu'une seule variable à la fois peut ne pas nous donner

suffisamment d'informations pour comprendre l'ensemble de données et la relation entre les variables. C'est là que l'analyse bivariée (et multivariée) vient à la rescousse.

La méthode diffère selon que la variable est catégorielle ou numérique. Nous avons beaucoup de variables et la plupart d'entre elles sont catégorielles. Nous ne ferons pas une analyse approfondie de chacun, mais la variable catégorique peut être facilement explorée avec un diagramme circulaire.

Traçons nos variables numériques et voyons s'il est logique de calculer leur moyenne et leur variance.

Une boîte à moustaches est un graphique qui permet de visualiser comment les données (variables) est distribué à l'aide de quartiles. Il montre le minimum, le maximum, la médiane, le premier quartile et le troisième quartile de la série de données.

Diagramme en boîte, c'est la méthode pour montrer graphiquement la propagation d'une variable numérique à travers les quartiles.

3.2.3.1 La variable occutc :

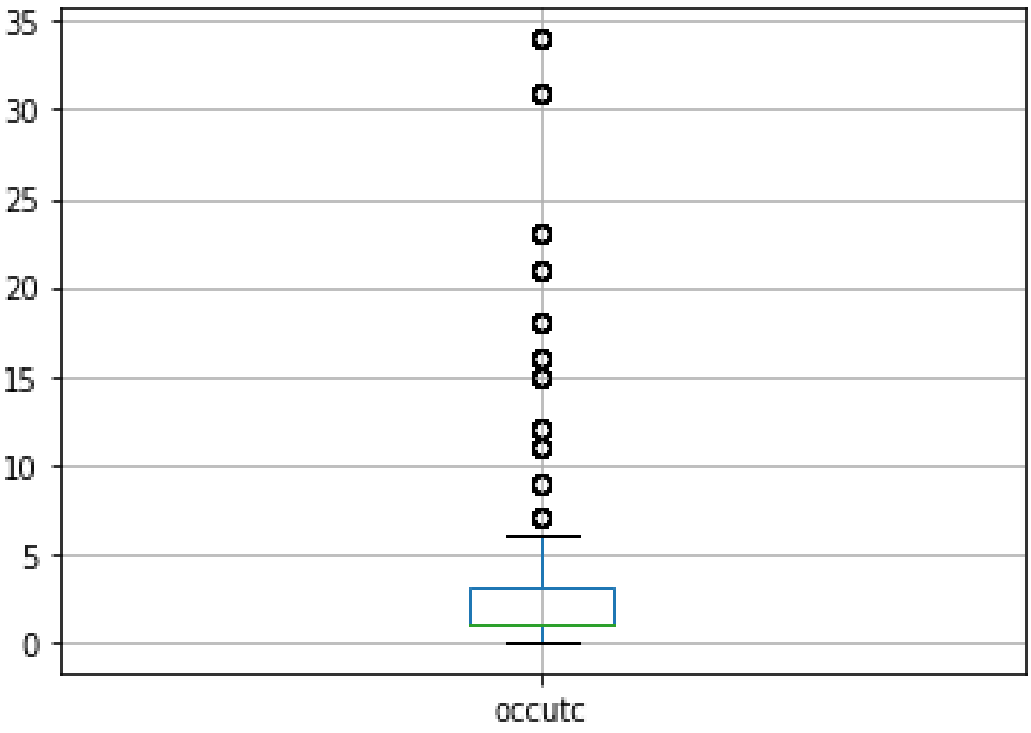

FIGURE 3.13 – une boîte à moustaches de la variable occutc

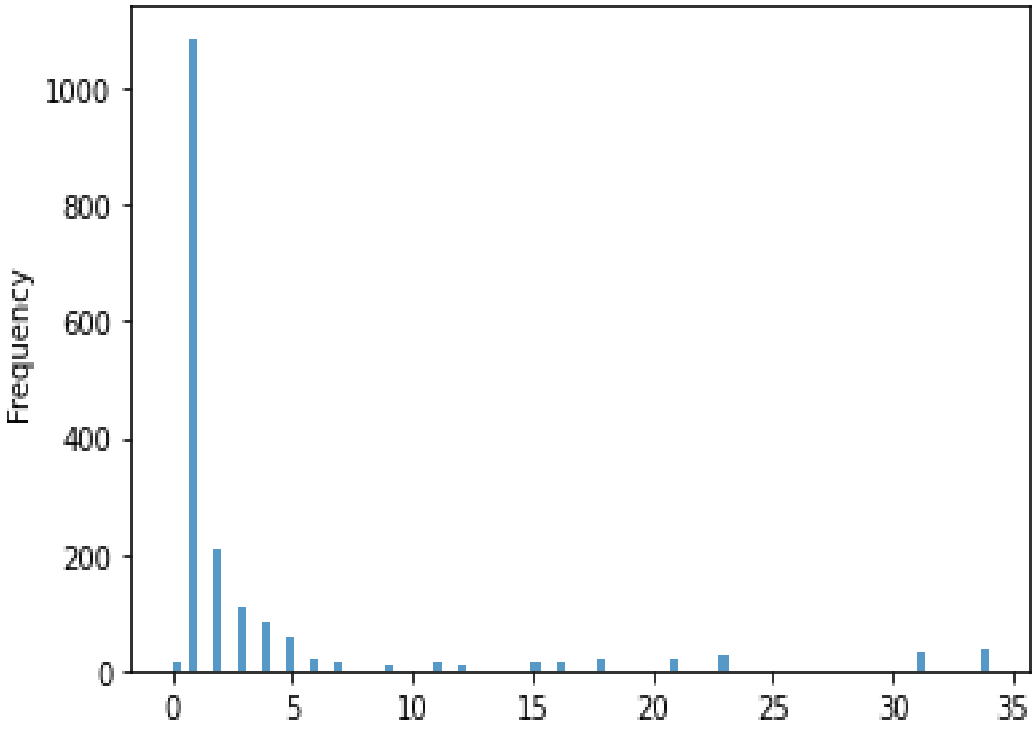

FIGURE 3.14 – l'histogramme de la variable occutc

Nous avons ici un exemple extrême d'une distribution de données déséquilibrée. Ce qui se passe, c'est que la plupart des accidents ne concernent pas le transport en commun. Donc **occutc** vaut 0. Mais lorsqu'il s'agit de transport en commun, ce nombre est élevé en raison de la nature même du transport en commun. Nous devrons aborder la répartition inégale plus tard. On peut aussi s'interroger sur l'utilité de cette variable.

3.2.3.2 La variable nbv :

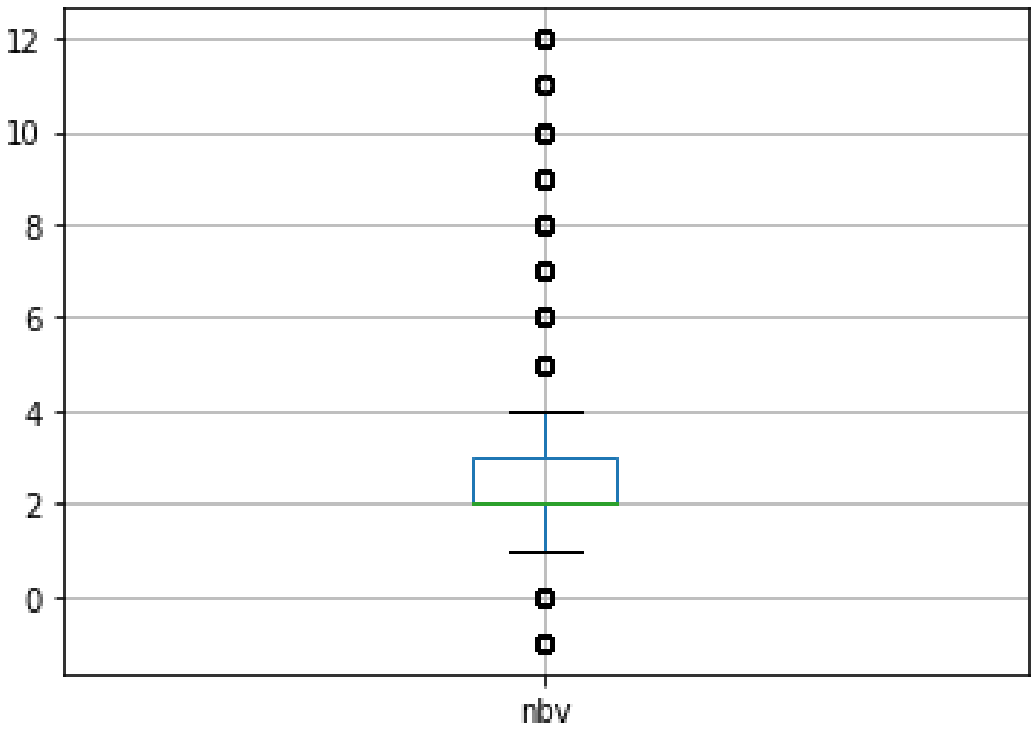

FIGURE 3.15 – une boîte à moustaches de la variable nbv

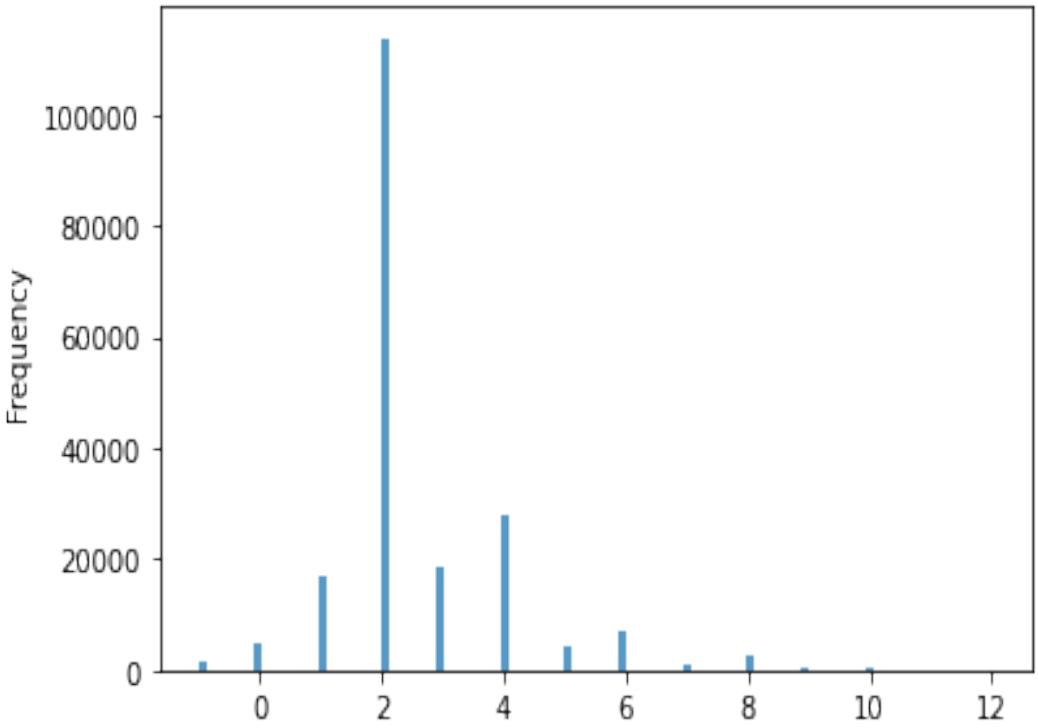

FIGURE 3.16 – l'histogramme de la variable nbv

3.2.3.3 La variable vma :

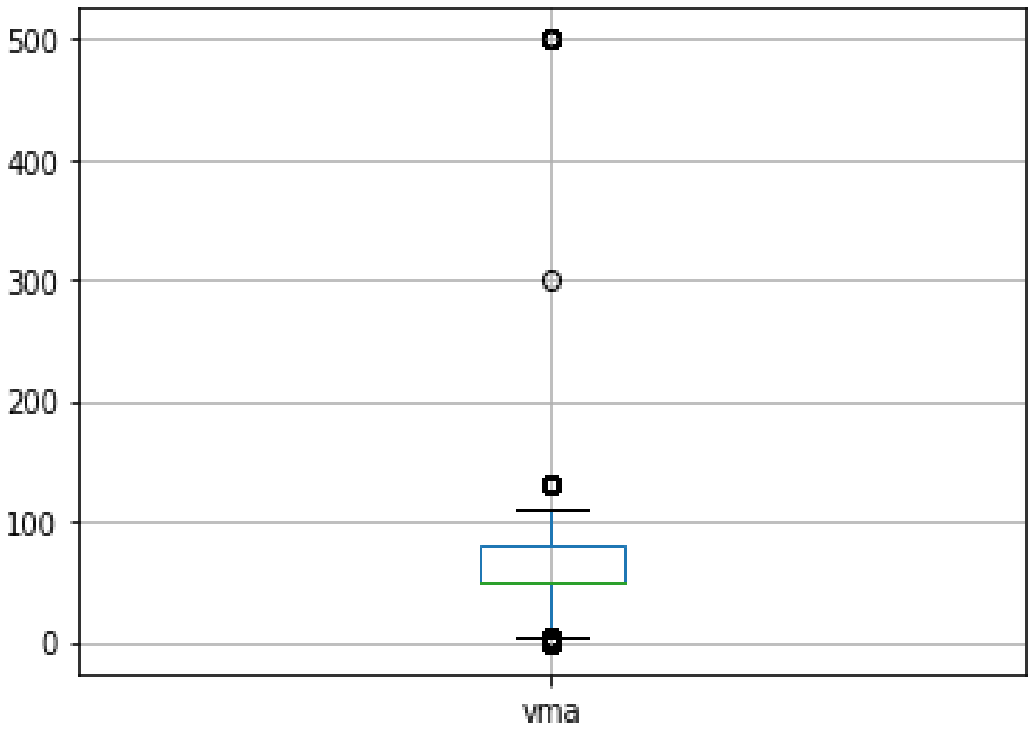

FIGURE 3.17 – une boîte à moustaches de la variable vma

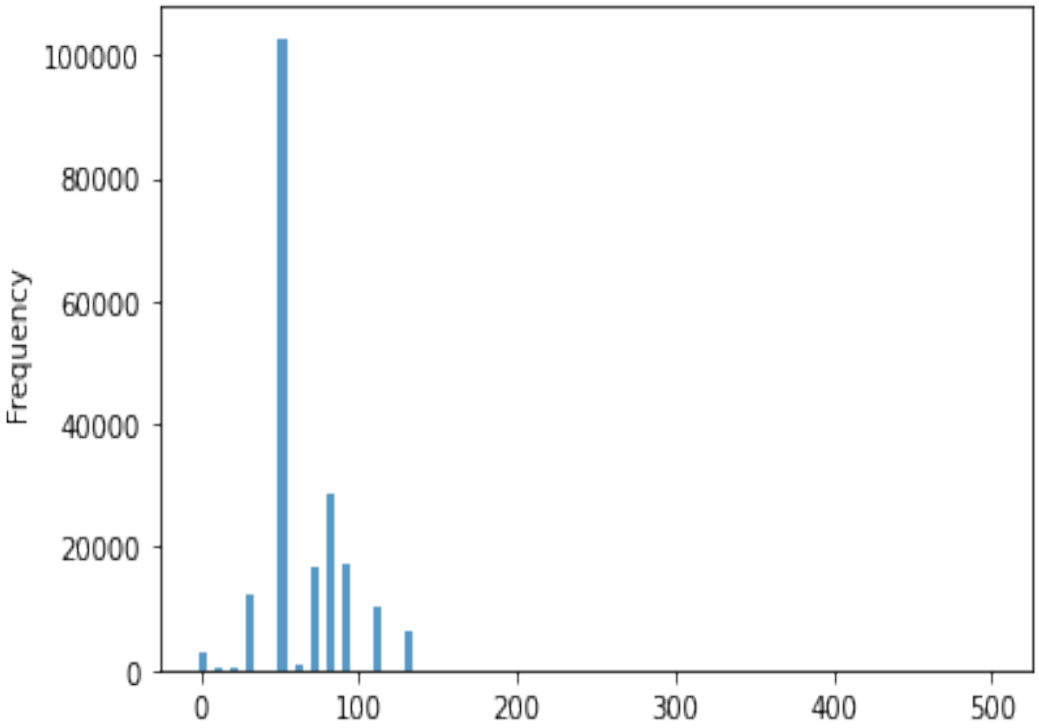

FIGURE 3.18 – l'histogramme de la variable vma

3.2.3.4 La variable grav :

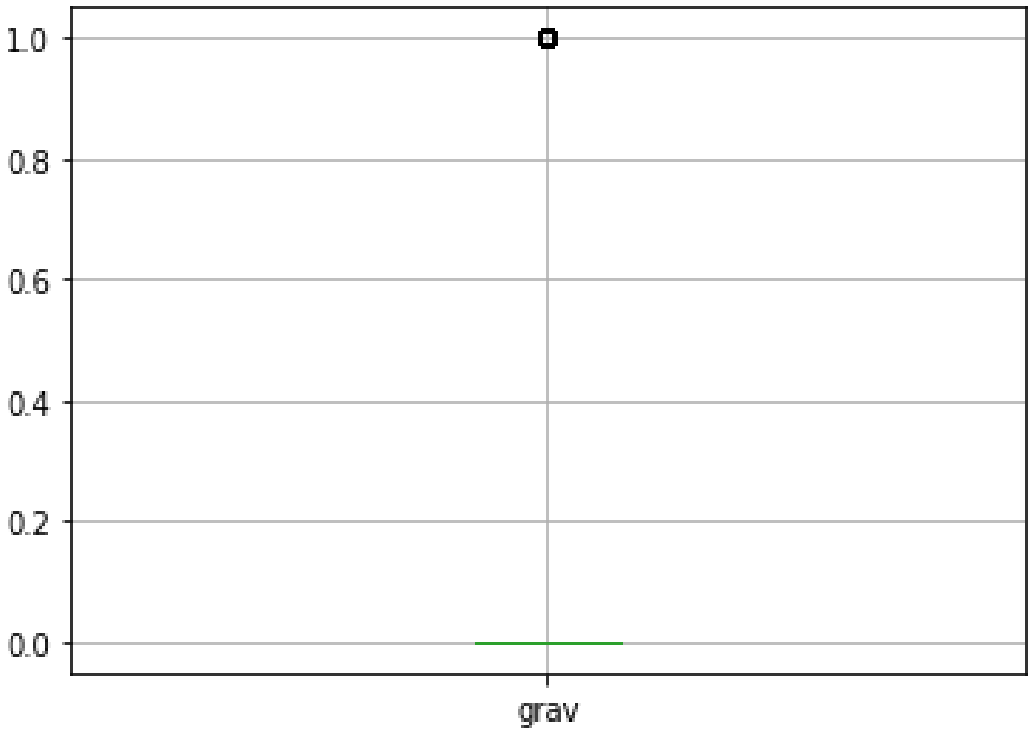

FIGURE 3.19 – une boîte à moustaches de la variable grav

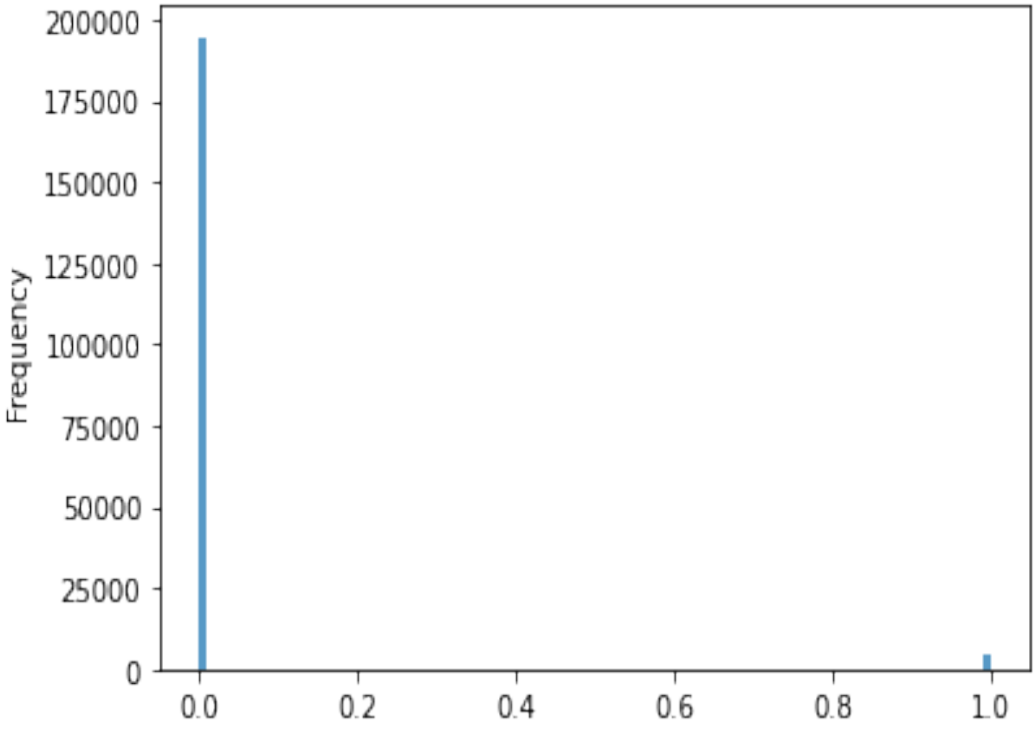

FIGURE 3.20 – l'histogramme de la variable grav

Comme vu précédemment, les accidents mortels sont sous-représentés, ce qui peut avoir un impact négatif sur les futurs modèles.

3.2.4 Analyse Bivarée (Dépendance variable)

Cette étape est cruciale, car identifier que la relation entre une variable prédictive et la variable cible est linéaire nous permettrait d'utiliser un modèle linéaire simple pour faire une prédiction de classe. Au contraire, une relation non linéaire nous obligerait à utiliser un modèle non linéaire pour obtenir les meilleures performances. Quoi qu'il en soit, il est recommandé de commencer par un modèle linéaire simple et de ne passer à un modèle plus complexe ou non linéaire que s'il ne donne pas suffisamment de résultats.

Nous avons également expliqué comment nous devrions essayer de supprimer les variables prédictives qui ne sont pas corrélées à la variable cible. L'analyse univariée peut être naïve à cet effet. Le fait que la corrélation univariée entre une variable prédictive et la variable cible soit faible ne signifie pas nécessairement que cette variable prédictive est inutile : une variable prédictive combinée à d'autres variables prédictives pourrait avoir une corrélation significative avec la variable cible.

De plus, la détection de la corrélation entre les variables prédictives signifie qu'un modèle plus simple (meilleur) avec moins de fonctionnalités peut exister. C'est là que nous pourrions utiliser la réduction de dimensionnalité comme PCA ou Autoencoders, ou la sélection de fonctionnalités comme RFE.

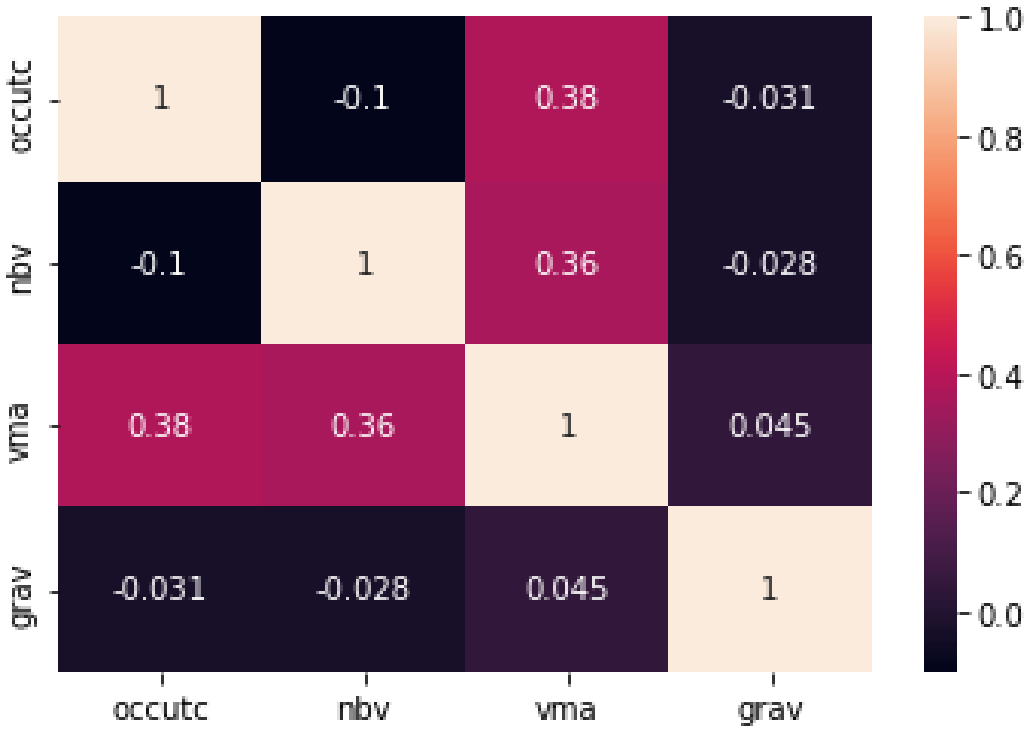

FIGURE 3.21 – Correlation Matrix

Il semble qu'il n'y ait pas de relation bivariée trés forte entre nos variables numériques.

Examinons les relations entre les variables prédictives catégorielles et la variable cible :

Nous ne montrerons que les variables les plus intéressantes, c'est-à-dire celles qui sont cor-

rélées au décès lors d'un accident.

Après avoir effectué une analyse bivariée sur chaque variable, nous avons plus d'informations sur les variables qui pourraient être inutiles. Mais cela pourrait être trompeur, car il peut y avoir des relations plus complexes que seule une analyse multivariée révélerait. Quoi qu'il en soit, nous ne le ferons pas ici.
Certaines variables prédictives sont supprimées car il semble qu'elles ne soient pas liées à la variable cible :

— **mois** (mois)

— **sexe** (genre)

— **hrmn** (heure de l'accident)

— **prof** (profil de route)

— **surf** (l'état de la route)

Certaines variables sont supprimées car il semble qu'elles soient corrélées à d'autres variables prédictives et qu'elles ne donnent pas plus d'informations que nous n'en avons déjà :

— **obsm** (obstacle mobile heurté, corrélé à la personne et au lieu de l'accident)

— **occutc** (nombre de personnes dans le véhicule, corrélé au type de véhicule)

— **nbv** (nombre de voies, corrélé au type de route)

3.2.5 La nouvelle base de données

La nouvelle base de données contient 198 907 attributs et 17 variables.

	place	catu	grav	trajet	locp	catv	motor	catr	circ	plan	vma	lum	agg	int	atm	col	secu
0	1	1	0	5	-1	7	1	4	2	1	50	1	2	3	1	3	0.0
1	1	1	0	5	-1	33	1	4	2	1	50	1	2	3	1	3	0.0
2	1	1	0	5	-1	7	1	4	2	1	50	1	2	3	1	3	2.333333
3	1	1	0	5	-1	33	1	4	2	1	50	1	2	3	1	3	2.333333
4	1	1	0	1	-1	7	1	4	2	3	50	2	2	9	7	6	2.333333
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
198902	1	1	0	5	0	7	1	3	2	1	50	1	1	1	1	2	0.0
198903	1	1	0	9	0	7	1	3	2	1	50	1	1	1	1	2	0.0
198904	1	1	0	9	0	7	1	3	2	1	50	1	1	1	1	2	0.0
198905	1	1	0	9	-1	32	1	4	-1	1	30	4	2	1	1	6	0.333333
198906	10	3	0	5	3	32	1	4	-1	1	30	4	2	1	1	6	-0.666667

FIGURE 3.22 – Aperçu de la nouvelle base

3.2.6 Gestion des valeurs manquantes

Comme dit précédemment, même après modification de la base, nos données contiennent tout de même des valeurs manquantes. Celles-ci sont néfastes au bon fonctionnement des algorithmes de prédiction, puisque certains algorithmes ne peuvent pas être compilés en leur présence.
Pour y remédier, nous pouvons appliquer une méthode consiste à enlever toutes les lignes présentant au moins une valeur manquante.

— **La méthode de suppression des données manquantes :**

Elle est relativement simple et consiste à parcourir chaque ligne du tableau de données. Si une valeur est manquante, la ligne entière est alors supprimée. Dans le cas contraire, la ligne est conservée.
Pour ce faire, nous avons utilisé la fonction **isnull()**, disponible sur python.

Avec cette méthode, On remarque que il y a pas de valeurs manquantes donc il nous reste un jeu de données comportant 198 907, ce qui représente 100% de lignes conservées. Ce résultat est satisfaisant du fait du travail effectué en amont lors de la modification de la base de données.
Ainsi, notre base nettoyée comporte une quantité de données largement suffisante pour effectuer une étude de prédiction.

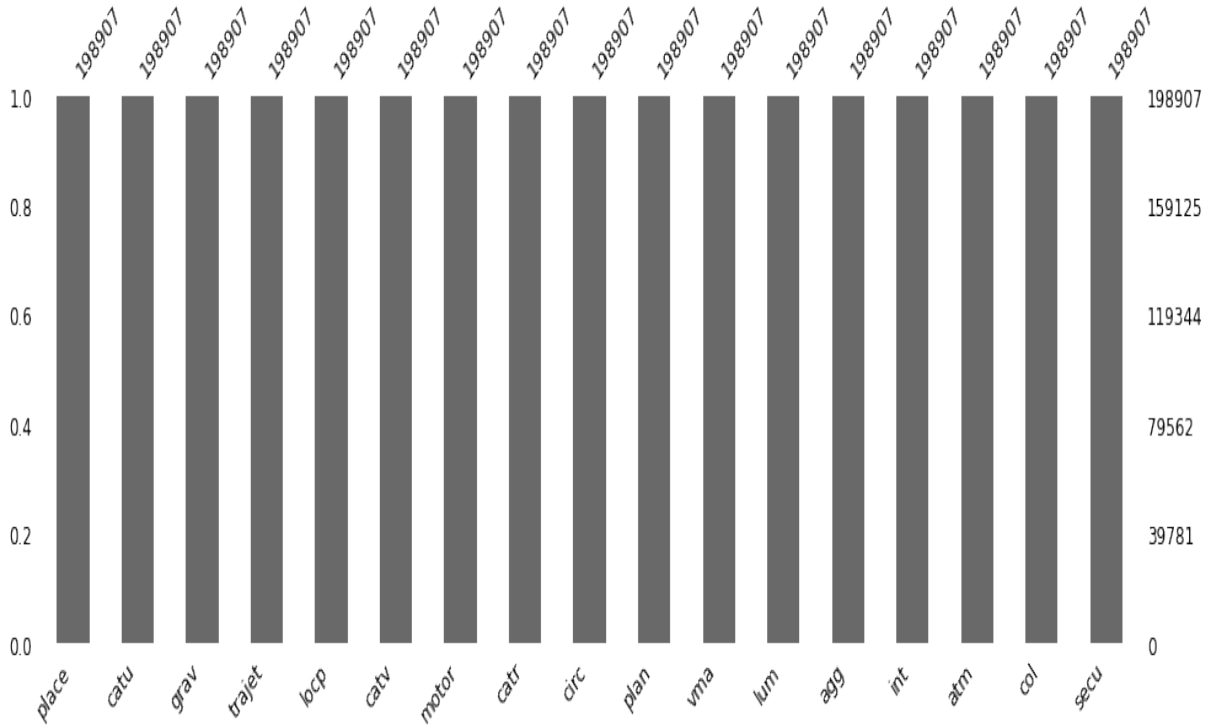

FIGURE 3.23 – la proportion de valeurs manquantes par variable de la nouvelle base

3.3 Visualisation de la base de données sur Power BI

3.3.1 Qu'est-ce que Power BI ?

Power BI est un ensemble de services logiciels, d'applications et de connecteurs qui œuvrent ensemble pour transformer des sources de données disparates en informations visuelles immersives et interactives. Vos données peuvent être sous forme de feuille de calcul Excel ou de collection d'entrepôts de données hybrides locaux ou sur le cloud. Power BI vous permet de vous connecter facilement à vos sources de données, de visualiser et de découvrir ce qui est important, et de partager ces informations avec qui vous voulez.

FIGURE 3.24 – Logo Power BI

3.3.2 Composants de Power BI

Power BI est constitué de plusieurs éléments qui fonctionnent ensemble, dont ces trois éléments de base :

— Une application de bureau Windows appelée **Power BI Desktop**.
— Un service SaaS (Software as a Service) en ligne appelé **service Power BI**.
— Des **applications mobiles** Power BI pour des appareils Windows, iOS et Android.

Ces trois éléments (Power BI Desktop, le service et les applications mobiles) sont conçus pour vous permettre de créer, partager et consommer de façon optimale des insights métier, en fonction de vos besoins ou de votre rôle.

Outre ces trois éléments, Power BI en comprend également deux autres :

— **Power BI Report Builder**, pour la création de rapports paginés à partager dans le service Power BI. Nous aborderons les rapports paginés en détail plus loin dans cet article.
— **Power BI Report Server**, qui est un serveur de rapports local dans lequel vous pouvez publier vos rapports Power BI après les avoir créés dans Power BI Desktop. Nous aborderons Power BI Report Server en détail plus loin dans cet article.

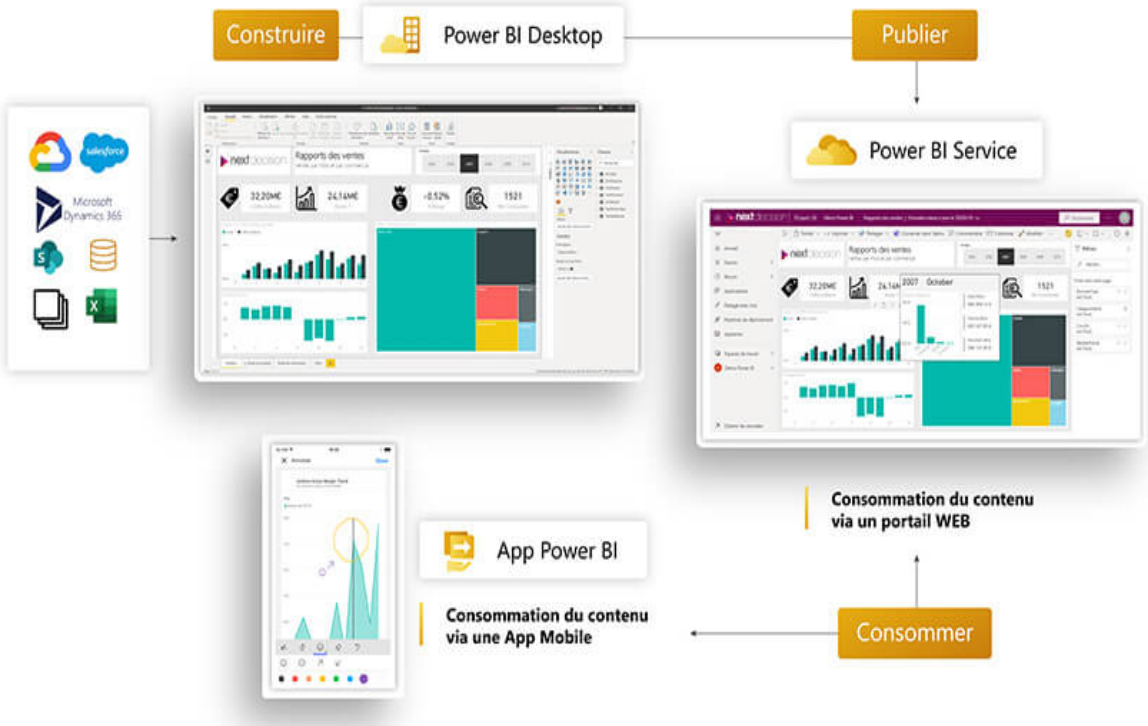

FIGURE 3.25 – schéma présentant Power BI

3.3.3 Modèle relationnel

Le modele relationnel des données des accidents de 2020 en Power BI :

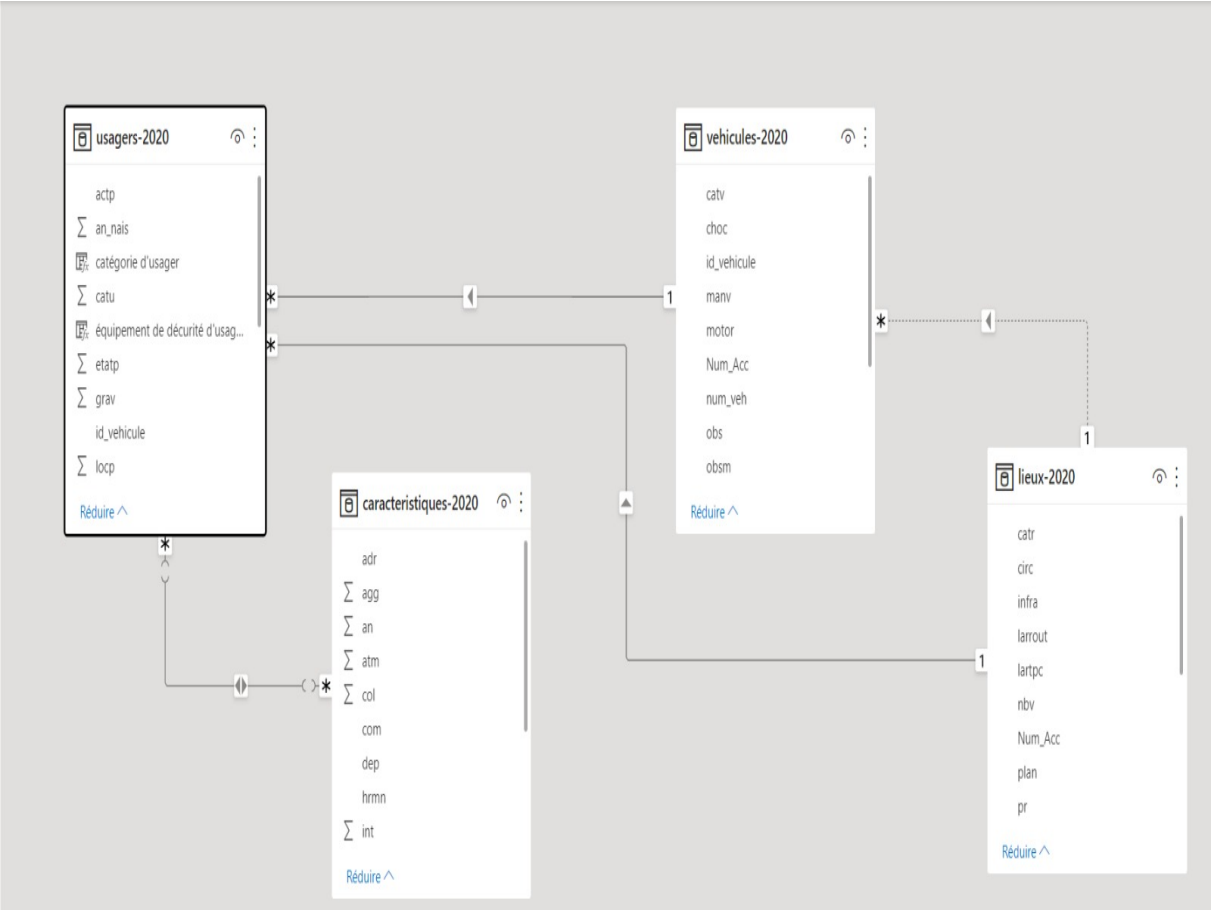

FIGURE 3.26 – Relations de modéle

3.3.4 Analyse

— On a décidé d'analyser le nombre d'accidents selon les points suivants :
Page « Général » : C'est une page de visualisation globale de données, elle contient 4 visuels et un indicateur de performance.
— Le nombre d'accidents par sexe
— La répartition des accidents par département
— Le nombre d'accident par âge
— Le nombre d'accidents par mois et gravité
Sur cette page nous avons un filtre de mois qui permet de filtrer sur des périodes précises, et un filtrer de département qui permet de filtrer par département.

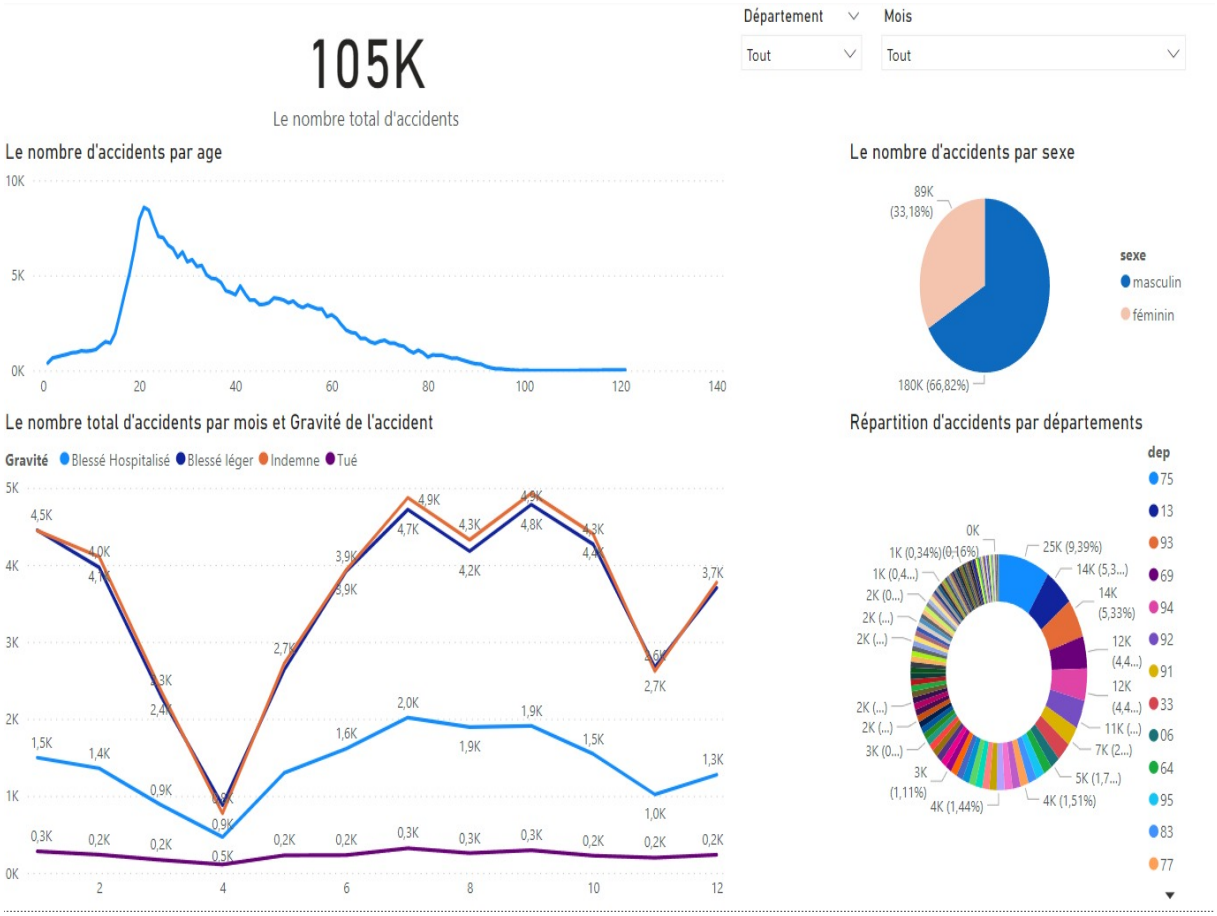

FIGURE 3.27 – Page « Général »

— **Page « Climat » :** C'est une page qui permet de visualiser le nombre d'accidents par condition climatique.

— Le nombre d'accidents par Conditions atmosphériques
— Le nombre d'accidents par type de lumière
— Le nombre d'accidents par Collision
— Le nombre total d'accidents

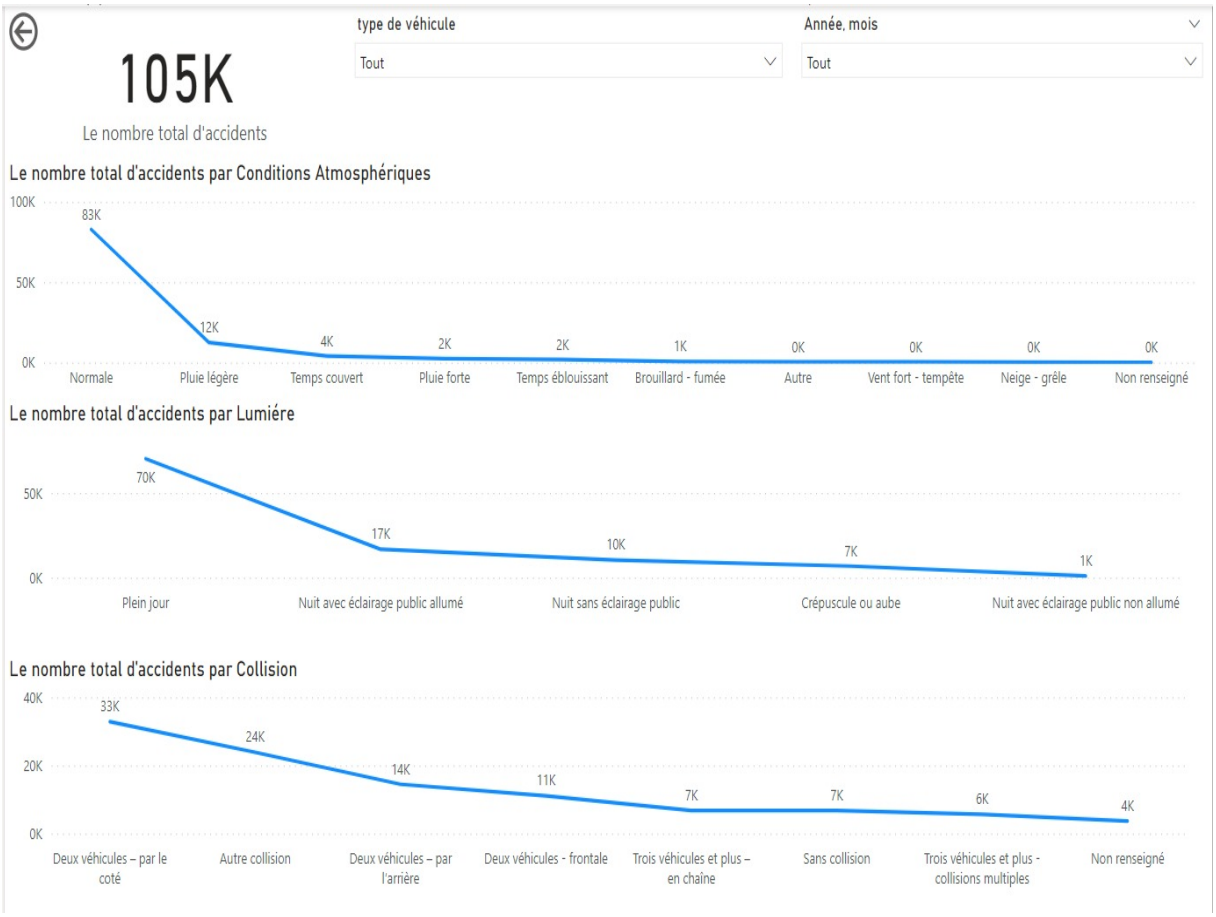

FIGURE 3.28 – Page « Climat »

— **Page « Causes » :** C'est une page qui permet de visualiser le nombre d'accidents par causes.
  — Le nombre d'accidents par type d'obstacle fixe
  — Le nombre d'accidents par type d'obstacle mobile
  — Le nombre d'accidents par type de choc initial

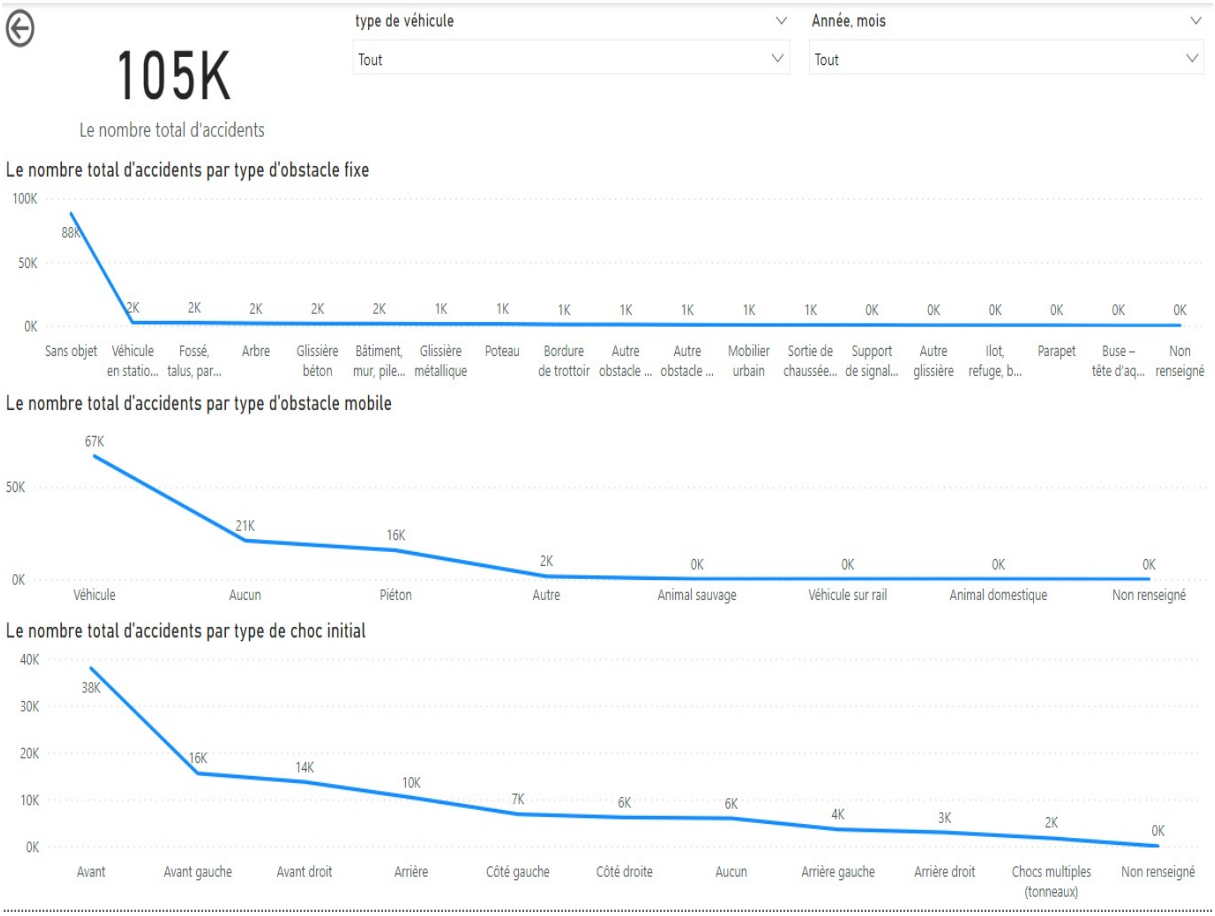

FIGURE 3.29 – Page « Causes »

— **Page « Usagers » :** C'est une page qui permet de visualiser le nombre d'accidents par usager.
  — Le nombre d'accidents par catégorie d'usager
  — Le nombre d'accidents par motif de déplacement
  — Le nombre d'accidents par équipement de sécurité de l'usager

FIGURE 3.30 – Page « Usagers »

— **Page « Carte Accidents» :** C'est une page qui permet de visualiser le nombre d'accidents par commune.

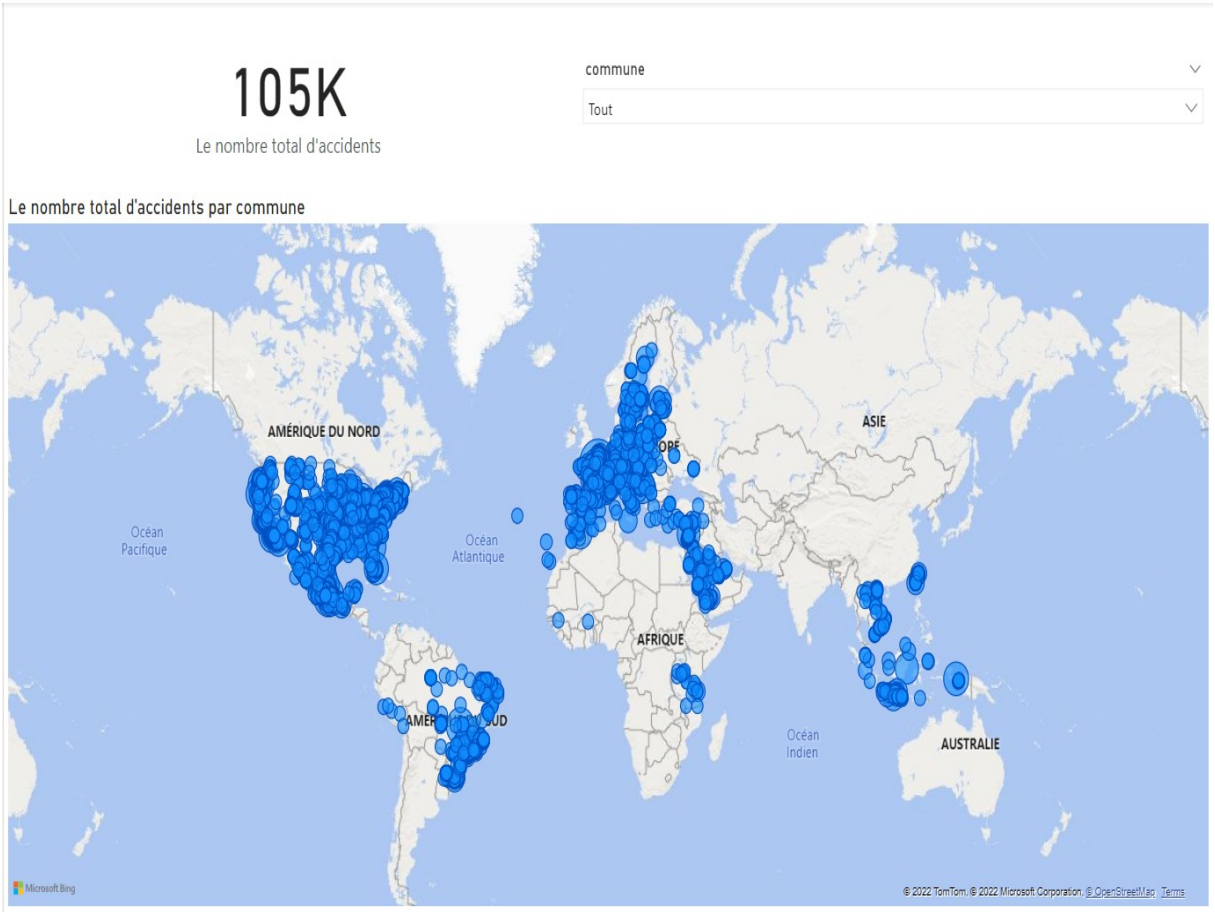

FIGURE 3.31 – Page « Carte Accidents»

Chapitre 4

La mise en place des modèles

4.1 Intérêt de la modélisation

Notre variable d'intérêt **grav** à prédire, est une variable à quatre modalités, avec une classe minoritaire : la classe de tués.

Nous avons décidé de recoder notre variable cible en une variable binaire :

— **Classe 0**, qui regroupe les blessés légers, hospitalisés, et Indemne, avec la modalité "0"

— **Classe 1**, qui regroupe les décédés, avec la modalité "1" Nous obtenons ainsi une base plus équilibrée.

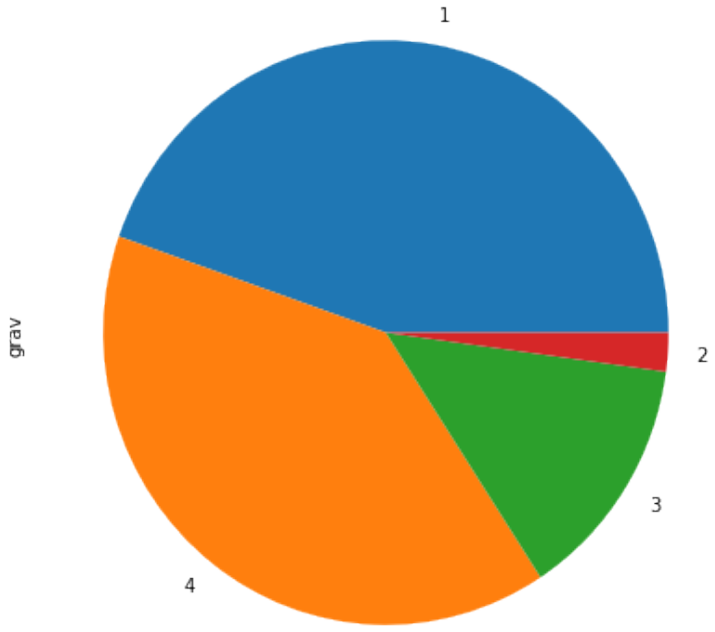

FIGURE 4.1 – Répartition de la gravité avec 4 modalités

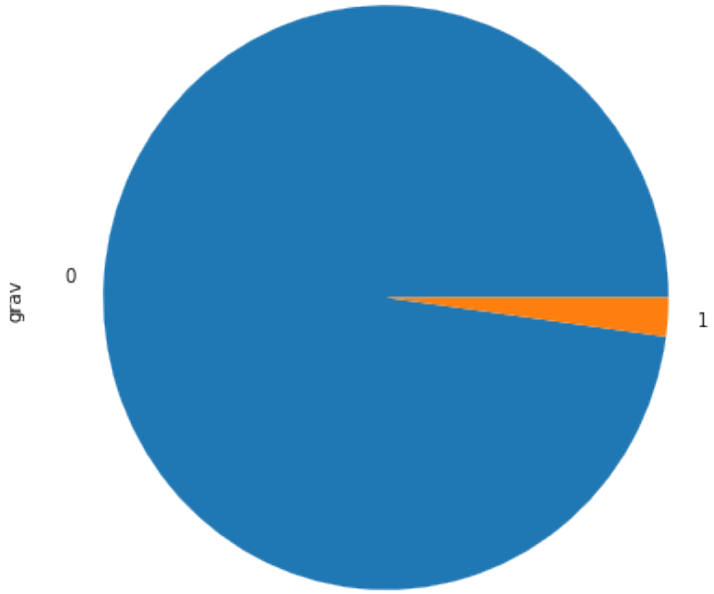

FIGURE 4.2 – Répartition de la gravité avec 2 modalités

L'intérêt purement prédictif d'un tel modèle est limité, mais réside plutôt dans la capacité du modèle à dégager les variables explicatives de la gravité et les interactions entre ces dernières, et par ailleurs à analyser la fluctuation des résultats prédictifs vis-à-vis de l'absence ou de la présence de variables connues a posteriori de l'accident.

4.2 Recherche d'informations redondantes

L'analyse de la corrélation par le coefficient de corrélation linéaire, peut nous donner une indication quant à l'existence de relations linéaires entre certaines variables, et donc à l'existence d'informations redondantes. Mais cette statistique peut se révéler insuffisante car deux variables non linéairement corrélées peuvent l'être d'une autre façon, par exemple une relation exponentielle ou polynomiale entre deux variables. La détection de ce type de relation est possible en utilisant la théorie de l'information de Shannon, dans laquelle les variables contenues dans un jeu de données constituent de l'information mesurée par l'entropie.
Dans la suite, nous n'utiliserons que l'analyse de la corrélation entre variables, le volume de données que nous avons ne permettant pas le calcul de l'entropie. Les algorithmes que nous mettrons en place par la suite ont leur propre mesure du degré d'information donnée par les variables.
Par ailleurs, nous décidons d'effectuer une analyse de données, les données catégorielles étant transformées en variables binaires.

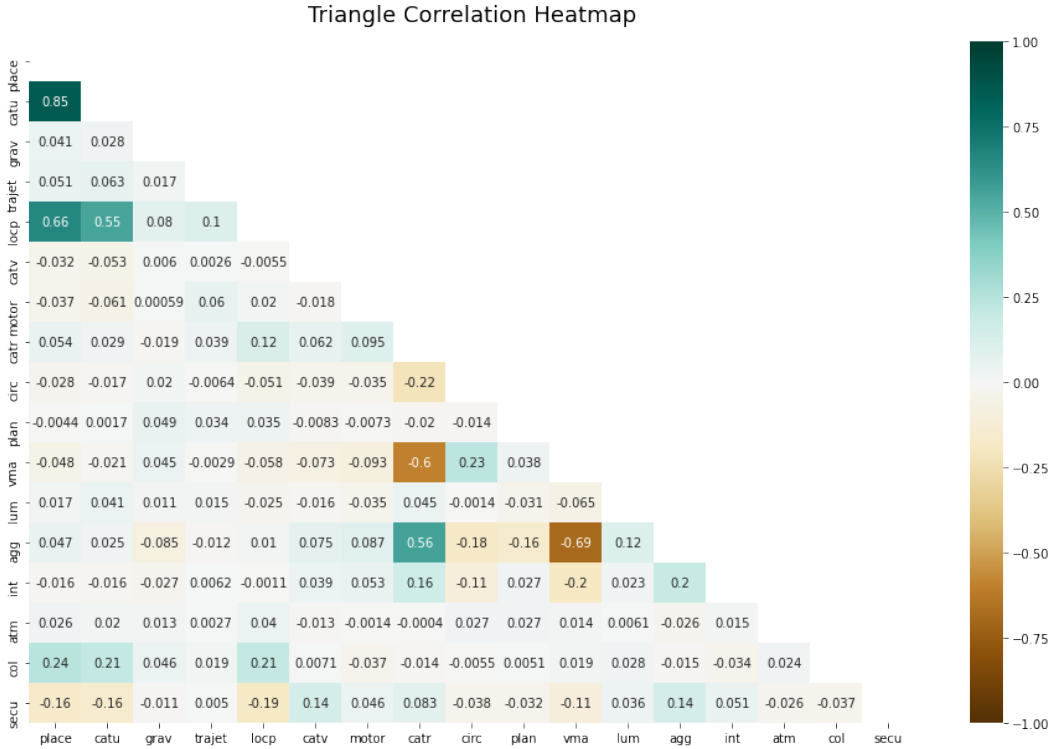

FIGURE 4.3 – Analyse de la corrélation sur la nouvelle base

Nous n'observons pas de fortes relations entre les différentes variables de notre jeu de données, sauf entre la variable décrivant la catégorie d'usager (**catu**) qui est corrélée à la variable **place** qui décrit la place occupée dans le véhicule par l'usager au moment de l'accident.De même entre localisation du piéton (**locp**) et la place (**place**), avec une corrélation autour de 66%.

4.3 Les métriques utilisées

Pour mesurer la qualité de nos modèles, nous avons utilisé plusieurs indicateurs de "performance" fréquemment utilisés pour ce type de problème, et adaptés à notre étude.
Notons qu'utiliser une seule métrique n'est pas pertinent car chaque métrique évalue un type de caractéristiques particulier. C'est donc en regardant un ensemble complémentaire de métriques bien choisies que nous pourrons évaluer la qualité de nos modèles.
Dans cette partie, nous utiliserons la notion de vrais et faux positifs, ainsi que de vrais et faux négatifs. Nous pouvons les définir de la manière suivante :

- **les vrais positifs (TP) :** ce sont les valeurs positives correctement prédites. Cela signifie que la valeur de la classe réelle est "1" et celle de la classe prédite est également "1".

- **les faux positifs (FP) :** ce sont les valeurs positives incorrectement prédites. Cela signifie que la valeur de la classe réelle est "0" et celle de la classe prédite est "1".
- **les vrais négatifs (TN) :** ce sont les valeurs négatives correctement prédites. Cela signifie que la valeur de la classe réelle est "0" et celle de la classe prédite est également "0".
- **les faux négatifs (FN) :** ce sont les valeurs négatives incorrectement prédites. Cela signifie que la valeur de la classe réelle est "1" et celle de la classe prédite est "0".

4.3.1 Exactitude (Accuracy)

L'**Exactitude** est la mesure de "performance" la plus intuitive. Il s'agit d'un rapport entre le nombre d'observations correctement prédites (les vrais positifs et les vrais négatifs) et le nombre total d'observations.

$$\text{Exactitude} = \frac{TP+TN}{TP+FP+TN+FN}$$

4.3.2 Précision (Precision)

La **Précision** est définie comme le nombre de résultats positifs corrects (les vrais positifs) divisé par le nombre de tous les résultats positifs renvoyés par le modèle de prédiction (les vrais positifs et les faux positifs).

$$\text{Précision} = \frac{TP}{TP+FP}$$

Nous n'avons pas utilisé cette métrique pour mesurer la qualité de nos modèles. Cette métrique sert à comprendre comment est construit le Score F1 (que nous définirons un peu plus loin).

4.3.3 Rappel (Recall)

Le **Rappel** est défini comme étant le nombre de résultats positifs corrects (les vrais positifs) divisé par le nombre de résultats positifs corrects ainsi que tous les échantillons qui auraient dû être identifiés comme positifs par le modèle de prédiction (les vrais positifs et les faux négatifs).

$$\text{Rappel} = \frac{TP}{TP+FN}$$

4.3.4 Score F1

Le **Score F1** (aussi appelé F-score), utilise les notions de précision et de rappel d'un test définies ci-dessus.
Il s'agit de la moyenne harmonique de la précision et du rappel, pour laquelle un Score F1 atteint sa meilleure valeur en 1 (précision et rappel parfait) et sa moins bonne valeur en 0.

$$\text{Score F1} = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Notons que le Score F1 ne tient pas compte des vrais négatifs, tout comme la précision et le rappel. Nous voyons ainsi la limite de ces trois classificateurs.

4.3.5 AUC (Area Under Curve)

Tout d'abord, définissons la **courbe ROC**, étroitement liée avec l'AUC.

Lorsque la classification est binaire ("0" ou "1"), le problème se réduit à l'obtention d'un score obtenu par le modèle prédictif mesurant la probabilité d'une observation d'appartenir à l'un des deux groupes. La question est alors de définir un seuil permettant de répartir les observations dans l'un de ces deux groupes : si l'observation obtient un score supérieur au seuil, elle sera classée dans le groupe "1", et dans le groupe "0" sinon.

Lorsque les deux groupes sont globalement équiprobables, il est légitime d'affecter une observation au groupe "1" si le modèle lui attribue un score supérieur à 0.5 et au groupe "0" sinon. Cependant, dans certains domaines comme en assurance automobile, la population des individus ayant subi un sinistre grave est peu représentée par rapport au nombre d'accidentés total, et un modèle construit par apprentissage n'affectera jamais une probabilité d'avoir un sinistre grave supérieur à 0.5. En revanche, si par expérience, la probabilité d'avoir un sinistre grave est de 0.05, alors un individu pour lequel le modèle prédit une probabilité de 0.1 pourra être jugé comme un individu à risque.

Ainsi, la courbe ROC sera d'autant plus efficiente que le seuil choisi est pertinent. Ce choix du seuil influencera donc profondément les prédictions du modèle. Il se fait en fonction des prédictions obtenues pour une base d'apprentissage d'observations pour lesquels nous connaissons le groupe d'appartenance.

Introduisons maintenant la notion de **sensibilité** et de **spécificité** :

— la sensibilité se définit comme le taux de vrais positifs

$$\text{Se} = \frac{TP}{TP+FN}$$

— la spécificité se définit comme le taux de vrais négatifs

$$\text{Sp} = \frac{TN}{TN+FP}$$

La courbe ROC représente l'évolution de la sensibilité en fonction la spécificité quand nous faisons varier le seuil. Pour évaluer la qualité d'un modèle de prédiction binaire, l'**AUC** est bon un critère. Elle est définie comme étant l'aire sous la courbe ROC. Une AUC égale à 1 signifie que le modèle prédictif est parfait, c'est-à-dire qu'il n'y a que des vrais positifs et vrais négatifs, en revanche une AUC égale à 0.5 signifie que l'affection des classes est faite aléatoirement.

4.4 La théorie des modèles de prédictions

4.4.1 Modèles linéaires généralisés (GLM)

Les modèles linéaires généralisés ont été présentés pour la première fois en 1972, par Nelder et Wedderburn. Ils permettent d'expliquer ou d'étudier la liaison entre une variable à expliquer Y et d'autres variables explicatives $(X_1, ..., X_p)$.

Ils regroupent le modèle linéaire général, le modèle log-linéaire, la régression logistique et la régression de Poisson.

Ces modèles linéaires généralisés sont principalement caractérisés par trois composantes :

— la **composante aléatoire**, qui identifie la loi de probabilités de la variable à expliquer. Nous supposons que l'échantillon statistique est constitué de n variables aléatoires indépendantes $(Y_1, ..., Y_n)$, $n \in N$, admettant des distributions issues d'une structure exponentielle, c'est-à-dire que la loi de probabilité de la composante aléatoire appartient à la famille exponentielle.

Les lois classiques utilisées en actuariat sont,$\forall i \in [1, ..., n]$, avec p le nombre de variables explicatives et n le nombre d'observations :

- la loi Normale : $Y_i \sim \mathcal{N}(\mu(x_{i,1},...,x_{i,p}),\sigma^2)$
- la loi de Bernoulli lorsque $Y_i \in \{0,1\}$: $Y_i \sim \mathcal{B}(\pi(x_{i,1},...,x_{i,p}))$
- la loi Gamma lorsque $Y_i \in \mathbb{R}^+$: $Y_i \sim \mathcal{G}(\alpha,\beta(x_{i,1},...,x_{i,p}))$
- la loi de Poisson lorsque $Y_i \in \mathbb{N}$: $Y_i \sim \mathcal{P}(\lambda(x_{i,1},...,x_{i,p}))$

— la **composante déterministe**, exprimée sous la forme d'une combinaison linéaire $\beta_0 + \beta_1 \times X_1 + ... + \beta_p \times X_p$. Ce vecteur à n composantes est appelé le "prédicteur" linéaire.

— **le lien g**, qui fait office de troisième composante, exprime une relation fonctionnelle entre la composante aléatoire et le prédicteur linéaire. Il spécifie comment l'espérance de Y est liée au prédicteur linéaire, construit à partir des variables explicatives.

$$g(E(Y_i) = \beta_0 + \beta_1 \times x_{i,1} + ... + \beta_p \times x_{i,p}$$

4.4.2 Forêts aléatoires

4.4.2.1 Les arbres de décisions

Cette méthode fut introduite par Leo Breinam en 1984.
L'apprentissage par arbre de décision est une méthode classique en apprentissage automatique. Son but est de créer un modèle qui prédit la valeur d'une variable cible depuis la valeur de plusieurs variables d'entrée.
Une des variables d'entrée est sélectionnée à chaque nœud intérieur de l'arbre selon une méthode qui dépend de l'algorithme. Chaque branche vers un nœud-fils correspond à un ensemble de valeurs d'une variable d'entrée, de manière à ce que l'ensemble des branches vers les nœuds-fils couvrent toutes les valeurs possibles de la variable d'entrée.
Il existe deux principaux types d'arbre de décision en fouille de données :

— **Les arbres de classification** *(Classification Tree)*, qui permettent de prédire à quelle classe la variable-cible appartient. Dans ce cas, la prédiction est une étiquette de classe.
— **Les arbres de régression** *(Regression Tree)*, qui permettent de prédire une quantité réelle (par exemple, le prix d'une maison ou la durée de séjour d'un patient dans un hôpital). Dans

ce cas, la prédiction est une valeur numérique.

4.4.2.2 La théorie du modèle

Les forêts aléatoires peuvent être considérées comme une agrégation d'arbres de décision. Cette méthode fut introduite par Leo Breinman en 2001, elle se base sur la technique générale du **Bagging**.

Le Bagging consiste à choisir K sous-ensembles de la base d'apprentissage avec remise, et à construire un arbre de décision sur chacun d'entre eux. Lors de la prédiction, nous prendrons ainsi la moyenne des prédictions retournées par chacun des K arbres.

Cependant, dans le Bagging, l'algorithme choisit à chaque noeud la variable qui effectue la meilleure séparation selon le critère choisi, alors que les forêts aléatoires choisissent la meilleure variable parmi une sélection aléatoire de q variables parmi les p variables initiales. C'est ce qu'on appelle le **Bootstrapping**. Le choix de q est généralement de $\sqrt{p}$ pour la classification, et de $\frac{p}{3}$ (arrondis à l'entier inférieur) pour la régression.

La méthode par forêts aléatoire consiste à faire tourner en parallèle un grand nombre d'arbres de décisions construits aléatoirement. La construction aléatoires des arbres permet d'obtenir des arbres décorrélés et donc une faible variance.

4.4.3 Machine à vecteurs de support (SVM)

Les SVMs ont été développés dans les années 1990. leur principe est simple : il ont pour but de séparer les données en classes à l'aide d'une frontière aussi « simple » que possible, de telle façon que la distance entre les différents groupes de données et la frontière qui les sépare soit maximale. Cette distance est aussi appelée « marge » et les SVMs sont ainsi qualifiés de « séparateurs à vaste marge », les « vecteurs de support » étant les données les plus proches de la frontière.

4.4.4 La Classification par la méthode des nuées dynamiques (k-means)

La classification k-means a été introduite par MacQueen en 1967. D'autres algorithmes similaires ont été développés par Forgey (1965) (centres mobiles) et Friedman (1967).

La classification k-means est une méthode itérative qui, quel que soit son point de départ, converge vers une solution. La solution obtenue n'est pas nécessairement la même quel que soit le point de départ. Pour cette raison, on répète en général plusieurs fois les calculs pour ne retenir que la solution la plus optimale pour le critère choisi.

Pour la première itération on choisit un point de départ qui consiste à associer le centre des k classes à k objets (pris au hasard ou non). On calcule ensuite la distance entre les objets et les k centres et on affecte les objets aux centres dont ils sont les plus proches. Puis on redéfinit les centres à partir des objets qui ont été affectés aux différentes classes. Puis on affecte à nouveau les objets en fonction de leur distance aux nouveaux centres. Et ainsi de suite jusqu'à ce que la convergence soit atteinte.

4.4.5 Modèle Gradient Boosting (GBM)

Comme les forêts aléatoires, le modèle Gradient boosting est aussi une méthode d'agrégation de modèles. L'algorithme est un cas particulier du Boosting.

Dans le même esprit d'approximation adaptative, Friedman (2002) a proposé sous l'acronyme MART (Multiple Additive Regression Trees) puis sous celui de GBM (Gradient Boosting Models), une famille d'algorithmes basés sur une fonction perte supposée convexe et différentiable. Le principe de base est de construire une séquence de modèles de sorte qu'à chaque étape, chaque modèle ajouté à la combinaison apparaisse comme un pas vers une meilleure solution. L'objectif est de donner un poids plus important aux individus pour lesquels la valeur a été mal prédite pour la construction du modèle suivant. Le fait de corriger les poids au fur et à mesure permet de mieux estimer les valeurs difficiles à prédire. La principale innovation est que le pas est franchi dans la direction du gradient de la fonction perte, afin d'améliorer les propriétés de convergence. Une deuxième idée consiste à approcher le gradient par un arbre de régression afin d'éviter un sur-apprentissage.
Pour résumer, l'algorithme du gradient boosting a besoin de trois éléments principaux :

- **Une fonction de perte** à optimiser qui doit être différentiable, et qui permet de résoudre le problème.
- **Un apprenant faible** pour effectuer des prédictions
- **Un modèle additif** pour combiner nos apprenants faibles afin de minimiser notre fonction de perte

4.4.6 Réseaux de neurones

Les premières réflexions sur les réseaux de neurones datent des années 1940 avec quelques noms à retenir : Von Neumann, Turing, Wiener, Mac Culloch et Pitts.

- **Réseaux neuronaux**
  Un réseau neuronal est l'association de neurones formels. Les principaux réseaux se distinguent par leur architecture (nombre de couches,...), leur niveau de complexité (le nombre de neurones, présence ou non de boucles de rétroaction dans le réseau), le type de neurones (leurs fonctions de transition ou d'activation) et enfin l'objectif visé : apprentissage supervisé ou non, optimisation, systèmes dynamiques,...
- **Le neurone formel**
  C'est en 1943 que Mac Culloch et Pitts proposent les premières notions de neurone formel.
  Le neurone réel, ou biologique, est une cellule qui se caractérise de la manière suivante :

  - des synapses : les points de connexion avec les autres neurones, fibres nerveuses ou musculaires
  - des dentrites ou entrées du neurones
  - les axones, ou sorties du neurone vers d'autres neurones ou fibres musculaires
  - le noyau qui active les sorties en fonction des stimulations en entrée

  Son fonctionnement peut se résumer par :
  - Une sommation pondérée des influx nerveux (inhibiteurs ou excitateurs) en provenance des neurones auxquels il est relié, via les dendrites et les synapses.
  - Une émission dans l'axone d'un influx si la sommation des entrées dépasse un seuil d'ac-

tivation.

En comparaison, le neurone formel de Mac Culloch et Pitts reproduit ce fonctionnement. Pour une entrée $x = (x_1, ..., x_n) \in R^n$, l'unité de poids synaptiques $\omega = (\omega_1, ..., \omega_n) \in R^n$, et un seuil d'activation , il calcule :

— la somme pondérée : $\sum_{i=0}^{n} \omega_i x_i$

— la sortie : $y(x) = \Phi(\omega.x - \theta) = 1_{\{\omega.x \geq \theta\}}$

La fonction Φ est appelé la fonction d'activation, ou fonction de réponse.

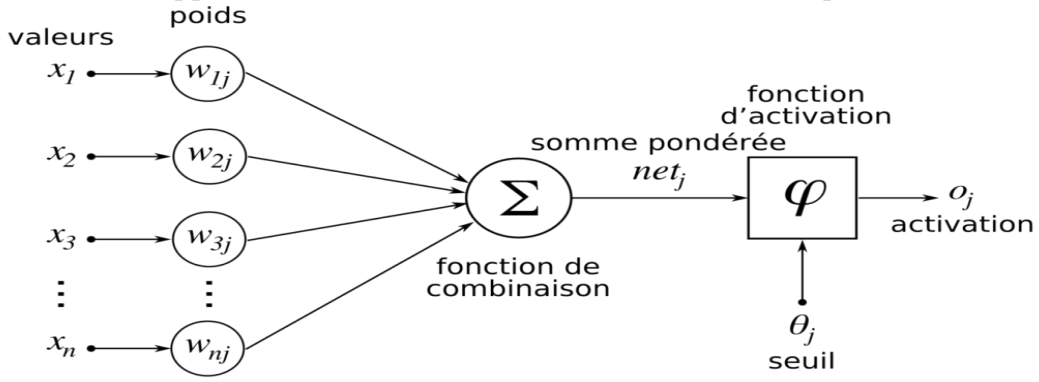

FIGURE 4.4 – Schéma explicatif du fonctionnement d'un neurone formel

Un tel neurone réalise donc une fonction de R^n dans $\{0, 1\}$.

Ces premiers travaux de Mac Culloch et Pitts ont donné naissance à l'école connexionniste. Ce concept fut ensuite mis en réseau avec une couche d'entrée et une couche de sortie par Rosenblatt (psychologue) en 1957 pour simuler le fonctionnement rétinien. C'est l'origine du perceptron.

— **Le perceptron**

Le perceptron peut être vu comme le type de réseau de neurones le plus simple. C'est un classifieur linéaire. Ainsi, la différence avec le modèle de neurone formel est que ce neurone peut « apprendre ». Le perceptron va être capable de classer des entrées et décider si elles correspondent à un côté ou bien à un autre. C'est la fonction d'activation qui distribue la réponse d'un côté ou de l'autre.

Dans sa version simplifiée, le perceptron est "simple" (mono-couche) et n'a qu'une seule sortie à laquelle toutes les entrées sont connectées. Les entrées et la sortie sont booléennes. Plus généralement, les entrées peuvent être des nombres réels.

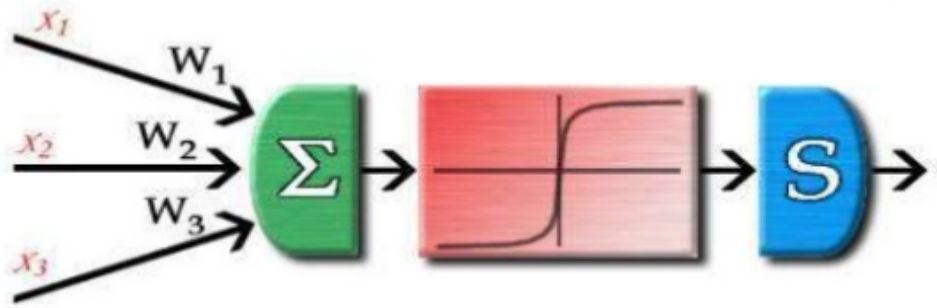

FIGURE 4.5 – Schéma explicatif du fonctionnement d'un perceptron simple

Le perceptron multicouche est un type de réseau neuronal formel organisé en plusieurs couches au sein desquelles une information circule de la couche d'entrée vers la couche de sortie uniquement. Il s'agit donc d'un réseau à propagation directe. Chaque couche est constituée d'un nombre variable de neurones, les neurones de la dernière couche (dite « de sortie ») étant les sorties du système global.

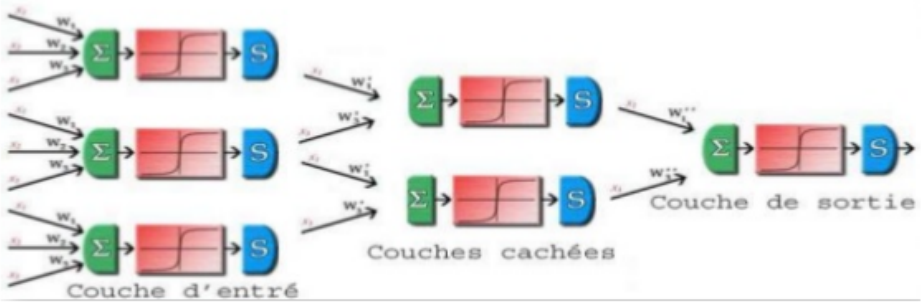

FIGURE 4.6 – Schéma explicatif du fonctionnement d'un perceptron monocouche

— **Les autres types de réseaux neuronaux**

Il existe cependant d'autres types de réseaux neuronaux que nous n'utiliserons pas dans cette étude, tels que les réseaux neuronaux convolutifs (CNN) et les réseaux neuronaux récurrents (RNN).

Les perceptrons multicouches, fonctionnent bien sur les données transactionnelles (tabulaires), toutefois, lorsque l'on désire effectuer de la reconnaissance d'images, les CNN sont un excellent choix ; ou encore avec des données séquentielles (telles que du texte, de l'audio, des séries chronologiques), les RNN sont un bon choix.

4.5 Application à notre étude

4.5.1 Échantillonnage de la base de données

Nous disposons d'un historique de données de 198 907 observations en 2020, et afin d'avoir une bonne évaluation de la qualité de nos modèles, nous décidons de la diviser en trois parties :

— **Une base d'apprentissage**, qui correspond à 67% des observations, soit 133 268 observations, et qui servira à alimenter nos modèles.
— **Une base de test**, qui correspond à 33% des observations, soit 65 639 observations et permettra de détecter du sur- ou sous-apprentissage par nos modèles.
Nous n'utiliserons pas de validation croisée, qui est aussi une méthode d'estimation de fiabilité d'un modèle fondée sur une technique d'échantillonnage.

Nous rappelons que cet échantillonnage fait suite à nos hypothèses de départ, qui sont :

— la sélection des conducteurs supposés responsables de l'accident.
— la suppression des lignes comprenant des valeurs manquantes.
— la suppression des variables aberrantes.

4.5.2 Paramétrage des modèles sous H2O

Le nombre de paramètres pour la mise en place d'un modèle est très large, même pour un modèle aussi simple que le modèle GLM. Dans la documentation des algorithmes, il est même conseillé d'utiliser les paramètres par défaut des algorithmes qui, selon le problème auquel nous faisons face, donnent des résultats satisfaisants.

Il existe des paramètres communs à tous les modèles dans H2O :
— Y : la variable à prédire, ici la variable **grav**.
— X : les variables explicatives, c'est-à-dire toutes les autres variables.
— **training_frame :** qui correspond à 80% des observations, soit 159 084 observations, et qui servira à alimenter nos modèles.
— **Une base de test** : qui correspond à 20% des observations, soit 39 823 observations et permettra de détecter du sur- ou sous-apprentissage par nos modèles.

4.5.3 Paramétrage des modèles sous Keras

Sous le modèle séquentiel, il existe plusieurs types de couches dans un réseau neuronal, et chaque couche du réseau est paramétrable. Nous avons mis en place un réseau neuronal à 4 couches, dont une couche d'entrée, 2 couches cachées, et 1 couche de sortie. Nous avons ajouté des couches de «dropout» qui permettent d'éviter le surapprentissage en ne prenant pas en compte une proportion de la base d'apprentissage à chaque cycle d'entraînement du réseau. De plus, toujours dans l'optique d'éviter le surapprentissage, il faut faire attention au nombre d'unités dans les couches cachées, qui ne doit pas être trop élevé ni trop bas si nous ne voulons pas être non plus en sous-apprentissage.

— **training_frame :** qui correspond à 90% des observations, soit 179 016 observations, et qui servira à alimenter nos modèles.
— **Une base de test** : qui correspond à 10% des observations, soit 19 890 observations et permettra de détecter du sur- ou sous-apprentissage par nos modèles.

4.5.4 Présentation des résultats

Nous présentons les résultats des différents algorithmes, selon les métriques choisies, et sur les différentes bases de données.

1. **Résultats des modèles sur la base**

Modèles	Precision	Accuracy	Recall	F-measure	AUC
Arbre de décision	0.67	0.66	0.66	0.66	0.72
Random forest	0.71	0.65	0.71	0.71	0.77
K-means	0.68	0.68	0.68	0.68	0.74
Régression logistique	0.59	0.54	0.55	0.48	0.68
SVM	0.63	0.54	0.63	0.63	___
polynomial svm	0.69	0.54	0.50	0.33	___
rbf svm	0.60	0.54	0.52	0.41	___
GridSearchCV	0.70	0.65	0.70	0.70	0.77

TABLE 4.1 – Résultats des modèles sur la base

2. **Sur la base d'apprentissage :**

Modèles	MSE	RMSE	MAE	RMSLE	Mean Residual Deviance
H2OGradientBoostingEstimator	0.02	0.14	0.04	0.10	0.02

TABLE 4.2 – Résultats sur la base d'apprentissag

3. **Sur la base de test :**

Modèles	MSE	RMSE	MAE	RMSLE	Mean Residual Deviance
H2OGradientBoostingEstimator	0.02	0.14	0.04	0.09	0.02

TABLE 4.3 – Résultats sur la base de test

4. **Deep Learning with TensorFlow**
   Nous pouvons également voir la courbe de perte :

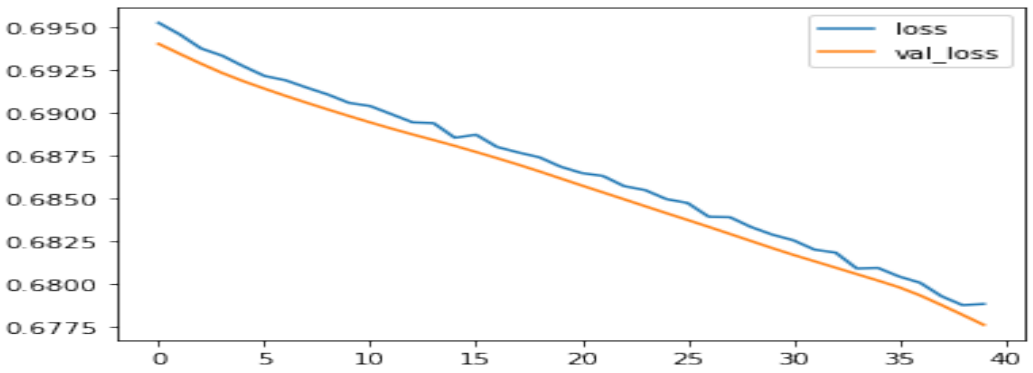

FIGURE 4.7 – la courbe de perte

La courbe bleue représente la précision des données d'entraînement et la couleur orange représente la précision des données de test. Nous notons également qu'à mesure que la précision continue de diminuer sur les données de test, la précision s'aplatit sur les données d'entraînement jusqu'à ce qu'elle progresse.

4.5.5 Interpréation des résultats

Nous notons qu'avec RandomForest, il a un meilleur score en termes de précision, de rappel, de mesure de F et d'AUC, tandis que K-mean a une meilleure Accuracy que les autres modèles et GridSearchCV a également un meilleur AUC. Mais en général les résultats sont assez proches les uns des autres.

En ce qui concerne les métriques, le modèle RandomForest donne les meilleurs résultats avec :
- critère='entropie'
- max_depth=20
- max_features='log2'
- n_estimators=500)

Sur les dix formes développées, la plus efficace est celle de RandomForest, mais les autres, plus simples, restent néanmoins acceptables. De plus, si l'on considère l'implémentation du réseau de neurones en termes de performances, on peut dire que le modèle TensorFlow est le grand perdant de cette comparaison. Il s'agit d'une analyse courante en apprentissage automatique.

En effet les modèles les plus performants ne sont pas toujours les plus complexes. Dans certaines études, les modèles simples ont parfois un rendement similaire par rapport à des modèles plus complexes, et même souvent supérieur (c'est le cas ici). Cependant l'utilité d'un modèle plus complexe peut se révéler dans son interprétabilité.

4.6 Prédiction de la gravité des accidents de l'année 2019

Étant donné que nous n'avons pas trouvé de données pour 2021, nous allons prédire la gravité des accidents sur les données de 2019.
Selon l'analyse prédictive des données d'accidents pour 2020, il a été constaté que le meilleur modèle est RandaomForest.
Nous allons maintenant prévoir les données sur les accidents de 2019 à l'aide de ce modèle.
Après une première analyse du jeu de données, nous pouvons remarquer que le jeu de données comporte de nombreuses impuretés en termes de valeurs manquantes.
Nous avons donc effectué le même gros travail de nettoyage des données, via un ensemble de processus connectés au pipeline.
Nous avons les mêmes 17 variables et nous allons faire une prédiction sur 253 488 observations. Nous avons codé normalement les données 2020 et 2019 pour utiliser le meilleur classificateur trouvé dans la partie modèle pour le prédire. Ensuite, nous avons déterminé : les données d'entraînement sont les données de 2020 et les données de test sont les données de 2019.Selon le système, on retrouve les observations d'accidents mortels et non mortels.

	Gravité
0	Indemme
1	Indemme
2	Indemme
3	Indemme
4	Indemme
...	...
253483	Indemme
253484	Indemme
253485	Indemme
253486	Indemme
253487	Indemme

FIGURE 4.8 – les observations d'accidents non mortels

	Gravité
944	Tué
1929	Tué
2302	Tué
2303	Tué
2410	Tué
...	...
248149	Tué
249053	Tué
250198	Tué
250265	Tué
251546	Tué

FIGURE 4.9 – les observations d'accidents mortels

En supposant que le gouvernement français n'ait pas analysé les accidents corporels de la route pour 2019, selon notre étude, nous pouvons prédire ce qui suit : sur 253 488 accidents corporles en 2019, la plupart des accidents (253 133) n'ont pas été mortels, alors que seulement 355 ont été mortels.

Chapitre 5

Code Python pour résoudre la problématique

5.1 Préparation des données partie I

```
#linear algebra
import numpy as np
#data processing
import pandas as pd

pep = pd.read_csv("usagers-2020.csv",delimiter=";")
vec = pd.read_csv("vehicules-2020.csv",delimiter=";")
loc = pd.read_csv("lieux-2020.csv", delimiter=";")
road = pd.read_csv("caracteristiques-2020.csv",delimiter=";")

pep.head(5)
vec.head(5)
loc.head(5)
road.head(5)

# merge all files together on the 'Num_Acc' key
df = pd.merge(pep, vec, on=['Num_Acc','Num_Acc'])
df = pd.merge(df, loc, on=['Num_Acc','Num_Acc'])
df = pd.merge(df, road, on=['Num_Acc','Num_Acc'])

df.head(5)

#La variable 'Num_Acc' nous intéresse plus
#On la supprime
df = df.drop(['Num_Acc'], axis=1)

df.shape

df.info()
```

```
import missingno as msno
msno.matrix(df)

msno.bar(df)

print(pep.isnull().sum())
import missingno as msno
msno.matrix(pep)
msno.bar(pep)
msno.heatmap(pep)

print(vec.isnull().sum())
import missingno as msno
msno.matrix(vec)
msno.bar(vec)
msno.heatmap(vec)

 print(road.isnull().sum())
 import missingno as msno
 msno.matrix(road)
 msno.bar(road)
 msno.heatmap(road)


 print(loc.isnull().sum())
 import missingno as msno
 msno.matrix(loc)
 msno.bar(loc)
 msno.heatmap(loc)

v1=df['v1']
v1
#v1 est presque vide

df.loc[df['etatp'] == -1.0]
#presque idem la variable etatp prend presque toujours -1

v2=df['v2']
v2
#Percentage of missing values of v2
totalmissingvalues=v2.isnull().sum()
Numbermissingvalues=v2.isnull().count()
Percentage=totalmissingvalues/Numbermissingvalues*100
Percentagearrondi=round(Percentage,1)
#Arrondir le nombre de chiffres après la virgule à 1
Percentagesorted=Percentagearrondi
Percentagesorted
```

```
#v2 est presque vide aussi puisque il contient plus de valeurs manquantes

lartpc=df['lartpc']
lartpc
#lartpc est presque vide

larrout=df['larrout']
larrout
#larrout est idem

pr=df['pr']
pr
#pr est presque vide

df = df.drop(['etatp','lartpc','larrout',
'an','jour','adr', 'lat','long','v1','v2',
'vosp','pr','pr1','num_veh_x','num_veh_y','senc','situ'], axis=1)

df.shape
df.dtypes # tells us the type for each column

# identifies which columns have many different values
columns = list(df)
for i in columns:
   if df[i].value_counts().size > 10:
       print (i + " " + str(df[i].value_counts().size))

df = df.copy()
#on commence par les secu
# Caculer la moyenne des 3 variables secu
df['secu'] = (df['secu1'] + df['secu2'] + df['secu3'])/3

#Test
print(df.dtypes)
df.head()

#on a plus besoin des variables secu1, secu2, secu3
df = df.drop(['secu1','secu2','secu3'], axis=1)
df

for index, row in df.iterrows():
   if row.locp == '1' or row.locp =='2': # group by "localisation sur chaussé"
       df.at[index,'locp'] = '1'
   if row.locp == '3' or row.locp== '4': # group by "localisation sur passage piéton"
       df.at[index,'locp'] = '2'
   if row.locp == '5' or row.locp=='6' or row.locp=='7'
   or row.locp== '8' or row.locp=='9': # group by "localisation est divers"
```

```
        df.at[index,'locp'] = '3'
    if row.locp == '0' or row.locp == '-1' : # group by null or other
        df.at[index,'locp'] = '99'

df.locp.value_counts().plot(kind='pie', figsize=(10, 10))
df['locp'].describe()

df = df.drop(['id_vehicule_x'], axis=1)
df = df.drop(['an_nais'], axis=1)
df = df.drop(['actp'], axis=1)
df = df.drop(['id_vehicule_y'], axis=1)
df = df.drop(['voie'], axis=1)
df = df.drop(['dep'], axis=1)
df = df.drop(['com'], axis=1)
df = df.drop(['manv'], axis=1)

df.loc[df['infra'] == 0]
#infra est idem alors on la supprime
df = df.drop(['infra'], axis=1)

for index, row in df.iterrows():
    if row.catv == '02': # group by small motorcycle
        df.at[index,'catv'] = '30'
    if row.catv == '31': # group by medium motorcycle
        df.at[index,'catv'] = '32'
    if row.catv == '33': # group by big motorcycle
        df.at[index,'catv'] = '34'
    if row.catv == '38': # group by bus
        df.at[index,'catv'] = '37'
    if row.catv == '04' or row.catv == '05' or row.catv == '06'
    or row.catv == '08' \or row.catv == '09' \
    or row.catv == '18' or row.catv == '19': # group by null or other
        df.at[index,'catv'] = '99'

df.catv.value_counts().plot(kind='pie', figsize=(10, 10))
df['catv'].describe()

#pour la variable obs:
for index, row in df.iterrows():
    if row.obs == '1'or row.obs=='2':
        df.at[index,'obs'] = '1'
    if row.obs == '3'or row.obs=='4'or row.obs=='5':
        df.at[index,'obs']= '3'
    if row.obs == '6'or row.obs=='7'or row.obs=='8'or row.obs=='9'or row.obs=='10'\
    or row.obs=='11'or row.obs=='12'or row.obs=='13':
        df.at[index,'obs'] = '6'
    if row.obs == '14'or row.obs=='15':
```

```
        df.at[index,'obs'] = '15'
    if row.obs == '16'or row.obs=='17':
        df.at[index,'obs'] = '16'
    else:
        df.at[index,'obs'] = '99'
df.obs.value_counts().plot(kind='pie', figsize=(10, 10))
df['obs'].describe()
df = df.drop(['obs'], axis=1)

#pour la variable choc:
for index, row in df.iterrows():
    if row.choc == '1'or row.choc=='2' or row.choc=='3':
        df.at[index,'choc'] = '1'
    if row.choc == '3'or row.choc=='4'or row.choc=='5' or row.choc=='6':
        df.at[index,'choc']= '4'
    if row.choc=='7'or row.choc=='8'or row.choc=='9':
        df.at[index,'choc'] = '9'
    if row.choc=='-1' or row.choc=='0':
        df.at[index,'choc'] = '99'
df.choc.value_counts().plot(kind='pie', figsize=(10, 10))
df['choc'].describe()
df = df.drop(['choc'], axis=1)

df.grav.value_counts().plot(kind='pie', figsize=(7, 7))
#binariser la variable grav
df.grav[df.grav == 1] = 0
df.grav[df.grav == 3] = 0
df.grav[df.grav == 4] = 0
df.grav[df.grav == 2] = 1
df.grav.value_counts().plot(kind='pie', figsize=(7, 7))
df['grav'].describe()

msno.heatmap(df)
df.corr()
import seaborn as sns
sns.heatmap(df.corr())
np.triu(np.ones_like(df.corr()))
import matplotlib.pyplot as plt
plt.figure(figsize=(16, 10))
# define the mask to set the values in the upper triangle to True
mask = np.triu(np.ones_like(df.corr(), dtype=np.bool))
heatmap = sns.heatmap(df.corr(), mask=mask, vmin=-1, vmax=1, annot=True, cmap='BrBG')
heatmap.set_title('Triangle Correlation Heatmap', fontdict={'fontsize':18}, pad=16)

df['place'] = df['place'].astype(object)
df['catu'] = df['catu'].astype(object)
df['sexe'] = df['catu'].astype(object)
```

```
df['trajet'] = df['trajet'].astype(object)
df['locp'] = df['locp'].astype(object)
df['catv'] = df['catv'].astype(object)
df['obsm'] = df['obsm'].astype(object)
df['motor'] = df['motor'].astype(object)
df['catr'] = df['catr'].astype(object)
df['circ'] = df['circ'].astype(object)
df['prof'] = df['prof'].astype(object)
df['plan'] = df['plan'].astype(object)
df['surf'] = df['surf'].astype(object)
df['mois'] = df['mois'].astype(object)
df['lum'] = df['lum'].astype(object)
df['agg'] = df['agg'].astype(object)
df['int'] = df['int'].astype(object)
df['atm'] = df['atm'].astype(object)
df['col'] = df['col'].astype(object)
df['secu'] = df['secu'].astype(object)
```

5.2 Analyse univariée (distribution des valeurs des variables)

```
# which variables are categorical and which are numerical?
df.dtypes
# number of people in the public transit vehicle
df.boxplot(column=['occutc'])
# number of people in the public transit vehicle
df.occutc.plot.hist(bins=100, alpha=0.75)
# number of people in the public transit vehicle
df['occutc'].describe()
# number of people in the public transit vehicle
df.occutc.value_counts()

# number of lanes on the road
df.nbv.plot.hist(bins=100, alpha=0.75)
# number of lanes on the road
df.boxplot(column=['nbv'])
df['nbv'].describe()

df.vma.plot.hist(bins=100, alpha=0.75)
df.boxplot(column=['vma'])
df['vma'].describe()

# lethality of the accident (the class we will try to predict)
df.grav.plot.hist(bins=100, alpha=0.75)
# lethality of the accident (the class we will try to predict)
df.boxplot(column=['grav'])
```

5.3 Bivariate analysis (variable dependence)

```
import seaborn as sn
import matplotlib.pyplot as plt
numericVariables = pd.DataFrame(df,columns=['occutc','nbv','vma','grav'])
corrMatrix = numericVariables.corr(method='pearson') # linear relationships
sn.heatmap(corrMatrix, annot=True)
plt.show()

numericVariables = pd.DataFrame(df,columns=['occutc','nbv','vma','grav'])
corrMatrix = numericVariables.corr(method='spearman') # monotonic relationships
sn.heatmap(corrMatrix, annot=True)
plt.show()

#Examinons les relations entre les variables prédictives catégorielles \
et la variable cible

from scipy.stats import chi2_contingency
from scipy.stats import chi2
import numpy as np

# must pass data to a contingency table first
# lighting conditions
crosstab = pd.crosstab(df['lum'], df['grav'])
#print(crosstab)
stat, p, dof, expected = chi2_contingency(crosstab)
print(p)
# we can reject the null hypothesis if the p-value is less than 0.05,
and say there is no correlation

#secu
crosstab = pd.crosstab(df['secu'], df['grav'])
#print(crosstab)
stat, p, dof, expected = chi2_contingency(crosstab)
print(p)

# type of vehicle
crosstab = pd.crosstab(df['catv'], df['grav'])
stat, p, dof, expected = chi2_contingency(crosstab)
print(p)

# whether the person is a driver (1), passenger (2), or pedestrian (3)
crosstab = pd.crosstab(df['catu'], df['grav'])
stat, p, dof, expected = chi2_contingency(crosstab)
```

```
print(p)

#agg
crosstab = pd.crosstab(df['agg'], df['grav'])
stat, p, dof, expected = chi2_contingency(crosstab)
print(p)

#int
crosstab = pd.crosstab(df['int'], df['grav'])
stat, p, dof, expected = chi2_contingency(crosstab)
print(p)

#plan
crosstab = pd.crosstab(df['plan'], df['grav'])
stat, p, dof, expected = chi2_contingency(crosstab)
print(p)

#surf
crosstab = pd.crosstab(df['surf'], df['grav'])
stat, p, dof, expected = chi2_contingency(crosstab)
print(p)

df = df.drop(['mois'], axis=1)
df = df.drop(['sexe'], axis=1)
df = df.drop(['hrmn'], axis=1)
df = df.drop(['prof'], axis=1)
df = df.drop(['surf'], axis=1)
df = df.drop(['obsm'], axis=1)
df = df.drop(['occutc'], axis=1)
df = df.drop(['nbv'], axis=1)

#final data
#contient 17 variables
df
```

5.4 Préparation des données partie II

```
#Valeurs manquantes et erronées
missingvalues=df.isnull()
missingvalues

#Total of missing values
totalmissingvaluesnosorted=df.isnull().sum()
totalmissingvaluesnosorted
```

```
import missingno as mso
plt.figure(1,figsize = (40, 5))
#ax0 = plt.subplot(1,2,1)
ax1 = plt.subplot(1,2,1)
#mso.matrix(data, ax = ax0, sparkline = False, fontsize = 15)
mso.bar(df, ax = ax1, fontsize = 12)
plt.subplots_adjust(wspace = 0.5)
```

5.5 Modéles (Machine learning)

```
# resampling
# class count
count_class_0, count_class_1 = df.grav.value_counts()
# divide by class
df_class_0 = df[df['grav'] == 0]
df_class_1 = df[df['grav'] == 1]
df_class_0_under = df_class_0.sample(count_class_1)
df_test_under = pd.concat([df_class_0_under, df_class_1], axis=0)
df_test_under.grav.value_counts().plot(kind='bar', title='Count (target)');

X = df_test_under.drop('grav', axis=1)
y = df_test_under['grav']
dfbin=pd.get_dummies(X)

import mca
#dfbin=pd.get_dummies(X)
acm=mca.MCA(dfbin)
# we can do dimensionality reduction by precising which percentage\
   of the best variables to keep
compressed=acm.fs_r(percent=0.50)
df_numeric = pd.DataFrame(data=compressed)
df_numeric.shape

df_numeric.isna().sum() # check that there are no na left
y.isna().sum() # check that there are no na left

#split data
from sklearn.model_selection import train_test_split
X = df_numeric
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, shuffle=True)

#library
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
```

```
from sklearn import tree, metrics
from sklearn.impute import SimpleImputer
from sklearn.metrics import make_scorer, accuracy_score, \
   f1_score, roc_auc_score, confusion_matrix, precision_recall_fscore_support
import matplotlib.pyplot as plt
import seaborn as sns # for data visualization

#Régression linéaire
from sklearn.linear_model import LinearRegression
regr = LinearRegression().fit(X_train, y_train)
# Make predictions using the testing set
y_pred = regr.predict(X_test)
# The coefficients
print('Coefficients: \n', regr.coef_)
# The mean squared error
print('Mean squared error: %.2f'
     % metrics.mean_squared_error(y_test, y_pred))
# The coefficient of determination: 1 is perfect prediction
print('Coefficient of determination: %.2f'
     % metrics.r2_score(y_test, y_pred))

#K-means
# n_neighbors = Number of neighbors to use by default for kneighbors queries.
param_grid={'n_neighbors': range(2,11,2)}
knn = GridSearchCV(KNeighborsClassifier(), param_grid)
knn.fit(X_train, y_train)
KNeighborsClassifier()
score = knn.score(X_test, y_test) # simple scoring
print(score)
#score = cross_val_score(knn, X, y, cv=5)  # cross validation
#print(score)
print(knn.best_params_) # best param
y_true = np.array(y_test)
y_pred = np.array(knn.predict(X_test))
#macro: calculate metrics for each label, and find their unweighted mean\
        This does not take label imbalance into account
scores = precision_recall_fscore_support(y_true, y_pred, average='macro')
print('Precision:\t' + str(scores[0]))
print('Recall:\t\t' + str(scores[1]))
print('F-measure:\t' + str(scores[2]))
cm = metrics.confusion_matrix(y_test, knn.predict(X_test))
plt.figure(figsize=(9,9))
sn.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);
```

```
import matplotlib.pyplot as plt
# Test the model using AOC-ROC Graph
y_pred_proba = knn.predict_proba(X_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
plt.legend(loc=4)
plt.show()

#Arbre de décision
from sklearn.model_selection import train_test_split, \
   cross_val_score, GridSearchCV, ParameterGrid
from sklearn import preprocessing
from sklearn.ensemble import RandomForestClassifier, \
   BaggingClassifier, AdaBoostClassifier, \
   GradientBoostingClassifier, \
   VotingClassifier
from sklearn.tree import DecisionTreeClassifier
param_grid={'max_depth': range(3,9,1)}
clf = GridSearchCV(DecisionTreeClassifier(), param_grid)
clf = clf.fit(X_train, y_train)
score = clf.score(X_test, y_test) # simple scoring
print(score)
print(clf.best_params_) # best param
y_true = np.array(y_test)
y_pred = np.array(clf.predict(X_test))
# macro: calculate metrics for each label, and find their unweighted mean \
         This does not take label imbalance into account
scores = precision_recall_fscore_support(y_true, y_pred, average='macro')
print('Precision:\t' + str(scores[0]))
print('Recall:\t\t' + str(scores[1]))
print('F-measure:\t' + str(scores[2]))
cm = metrics.confusion_matrix(y_test, clf.predict(X_test))
plt.figure(figsize=(9,9))
sn.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);

import matplotlib.pyplot as plt
# Test the model using AOC-ROC Graph
y_pred_proba = clf.predict_proba(X_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
```

```
plt.legend(loc=4)
plt.show()

#Régression logistique
from sklearn.linear_model import LogisticRegression
logRegr = LogisticRegression()
logRegr.fit(X_train, y_train)
score = logRegr.score(X_test, y_test) # simple scoring
print(score)
y_true = np.array(y_test)
y_pred = np.array(logRegr.predict(X_test))
# macro: calculate metrics for each label, and find their unweighted mean \
         This does not take label imbalance into account
scores = precision_recall_fscore_support(y_true, y_pred, average='macro')
print('Precision:\t' + str(scores[0]))
print('Recall:\t\t' + str(scores[1]))
print('F-measure:\t' + str(scores[2]))
cm = metrics.confusion_matrix(y_test, logRegr.predict(X_test))
plt.figure(figsize=(9,9))
sn.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);

import matplotlib.pyplot as plt
# Test the model using AOC-ROC Graph
y_pred_proba = logRegr.predict_proba(X_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
plt.legend(loc=4)
plt.show()

#SVM
from sklearn.svm import LinearSVC
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
lin_clf = LinearSVC(random_state=42)
lin_clf.fit(X_train, y_train)
y_pred = lin_clf.predict(X_test)
y_true = np.array(y_test)
y_pred = np.array(lin_clf.predict(X_test))
# macro: calculate metrics for each label, and find their unweighted mean \
        This does not take label imbalance into account
scores = precision_recall_fscore_support(y_true, y_pred, average='macro')
```

```
print('Precision:\t' + str(scores[0]))
print('Recall:\t\t' + str(scores[1]))
print('F-measure:\t' + str(scores[2]))
cm = metrics.confusion_matrix(y_test, lin_clf.predict(X_test))
plt.figure(figsize=(9,9))
sn.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);

# polynomial svm
poly_clf=SVC(C=1,kernel='poly',degree=2)
poly_clf.fit(X_train,y_train)
y_pred =poly_clf.predict(X_test)
y_true = np.array(y_test)
y_pred = np.array(poly_clf.predict(X_test))
# macro: calculate metrics for each label, and find their unweighted mean \
        This does not take label imbalance into account
scores = precision_recall_fscore_support(y_true, y_pred, average='macro')
print('Precision:\t' + str(scores[0]))
print('Recall:\t\t' + str(scores[1]))
print('F-measure:\t' + str(scores[2]))
cm = metrics.confusion_matrix(y_test, poly_clf.predict(X_test))
plt.figure(figsize=(9,9))
sn.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);

# rbf svm
clf_rbf=SVC(C=1,kernel='rbf',gamma=2)
clf_rbf.fit(X_train,y_train)
y_pred1 =clf_rbf.predict(X_test)
y_true = np.array(y_test)
y_pred = np.array(clf_rbf.predict(X_test))
# macro: calculate metrics for each label, and find their unweighted mean \
         This does not take label imbalance into account
scores = precision_recall_fscore_support(y_true, y_pred, average='macro')
print('Precision:\t' + str(scores[0]))
print('Recall:\t\t' + str(scores[1]))
print('F-measure:\t' + str(scores[2]))
cm = metrics.confusion_matrix(y_test, clf_rbf.predict(X_test))
plt.figure(figsize=(9,9))
sn.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
```

```
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);

#Random forest
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
from sklearn import metrics
def f1(model,x,y):
   return metrics.f1_score(y, model.predict(x),average='micro')
## tuning with RandomizeSearchCV
n_estimators=[100,500,1000]
criterion=['gini','entropy']
max_depth=[20,40,60,100]
max_feature=['sqrt','log2']
param_grid={
   'n_estimators':n_estimators,
   'criterion':criterion,
   'max_depth':max_depth,
   'max_features':max_feature
}
Randomiz_rf=RandomizedSearchCV(estimator=RandomForestClassifier(),\
    param_distributions=param_grid,n_iter=50,cv=3,scoring=f1,\
return_train_score=True,verbose=2)
Randomiz_rf.fit(X_train,y_train)
print(Randomiz_rf.best_params_)
print(Randomiz_rf.best_estimator_)

random_rf=RandomForestClassifier(criterion='entropy',max_depth=20,\
      max_features='log2', n_estimators=500)
random_rf.fit(X_train,y_train)
y_random=random_rf.predict(X_test)
y_true = np.array(y_test)
y_pred = np.array(random_rf.predict(X_test))
# macro: calculate metrics for each label, and find their unweighted mean \
        This does not take label imbalance into account
scores = precision_recall_fscore_support(y_true, y_pred, average='macro')
print('Precision:\t' + str(scores[0]))
print('Recall:\t\t' + str(scores[1]))
print('F-measure:\t' + str(scores[2]))
cm = metrics.confusion_matrix(y_test, random_rf.predict(X_test))
plt.figure(figsize=(9,9))
sn.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);
```

```
import matplotlib.pyplot as plt
# Test the model using AOC-ROC Graph
y_pred_proba = random_rf.predict_proba(X_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
plt.legend(loc=4)
plt.show()

#tuning with GridSearchCV
param_grid={'criterion': ['gini'],
'max_depth': [20,30,40],
'max_features': ['sqrt'],
'n_estimators': [900,1000,1100]}

Grid_search_rf=GridSearchCV(estimator=RandomForestClassifier(),\
   param_grid=param_grid,cv=3,scoring=f1,return_train_score=True,verbose=4)
Grid_search_rf.fit(X_train,y_train)
print(Grid_search_rf.best_params_)
print(Grid_search_rf.best_estimator_)
random_rf=RandomForestClassifier(criterion='gini',max_depth=30,\
    max_features='sqrt', n_estimators=900)
random_rf.fit(X_train,y_train)
y_random=random_rf.predict(X_test)
y_true = np.array(y_test)
y_pred = np.array(random_rf.predict(X_test))
# macro: calculate metrics for each label, and find their unweighted mean \
         This does not take label imbalance into account
scores = precision_recall_fscore_support(y_true, y_pred, average='macro')
print('Precision:\t' + str(scores[0]))
print('Recall:\t\t' + str(scores[1]))
print('F-measure:\t' + str(scores[2]))
cm = metrics.confusion_matrix(y_test, random_rf.predict(X_test))
plt.figure(figsize=(9,9))
sn.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);

import matplotlib.pyplot as plt
# Test the model using AOC-ROC Graph
y_pred_proba = random_rf.predict_proba(X_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
```

```
plt.legend(loc=4)
plt.show()

#Gradient Boosting
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.decomposition import PCA
from sklearn.metrics import mean_squared_error

sc = StandardScaler()
X_train_std = sc.fit_transform(X_train)
X_test_std = sc.transform(X_test)

# Hyperparameters for GradientBoostingRegressor
#
gbr_params = {'n_estimators': 1000,
         'max_depth': 3,
         'min_samples_split': 5,
         'learning_rate': 0.01,
         'loss': 'ls'}
# Create an instance of gradient boosting regressor
#
gbr = GradientBoostingRegressor(**gbr_params)
# Fit the model
gbr.fit(X_train_std, y_train)
# Print Coefficient of determination R^2
print("Model Accuracy: %.3f" % gbr.score(X_test_std, y_test))
# Create the mean squared error
mse = mean_squared_error(y_test, gbr.predict(X_test_std))
print("The mean squared error (MSE) on test set: {:.4f}".format(mse))

#H2O
!pip install requests
!pip install tabulate
!pip install future
!pip install -f http://h2o-release.s3.amazonaws.com/h2o/latest_stable_Py.html h2o
import h2o
h2o.init()
df_h2o=h2o.H2OFrame(df,destination_frame='df')
df_h2o.nrows
train,test= df_h2o.split_frame([0.8])
train.nrows
test.nrows
from h2o.estimators.gbm import H2OGradientBoostingEstimator
mGBM=H2OGradientBoostingEstimator()
mGBM.train([0,1],"grav",train)
```

```
p=mGBM.predict(test)
p
mGBM.model_performance(test)
h2o.cluster().shutdown()

#Réseaux neurones (avec Keras et Tenserflow)
from keras.models import Sequential
from keras.layers import Dense, Flatten,Dropout
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import cross_val_score
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import StratifiedKFold
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10, shuffle=True)
# split into input (X) and output (Y) variables
X_train = X_train.astype(float)
# encode class values as integers
encoder = LabelEncoder()
encoder.fit(y_train)
encoded_Y = encoder.transform(y_train)

model = Sequential()
model.add(Dense(16,input_dim=3, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(16,input_dim=3, activation='tanh'))
model.add(Dropout(0.1))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
loss = model.fit(X_train, y_train, batch_size=5000, epochs=40,\
     validation_split=0.3, callbacks=[], verbose=1)
score = model.evaluate(X_test, y_test, verbose=0)
print("score=", score)
import matplotlib.pyplot as plt
%matplotlib inline
plt.figure()
plt.plot(loss.history['loss'], label="loss")
plt.plot(loss.history['val_loss'], label="val_loss")
plt.legend()

#Best model : RandomForest model
#avec : criterion='entropy', max_depth=20,max_features='log2', n_estimators=500
```

5.6 Prédictions

```
#Prédiction sur les données 2019
#préparation de ces données
p = pd.read_csv("usagers-2019.csv",delimiter=";")
v = pd.read_csv("vehicules-2019.csv",delimiter=";")
l = pd.read_csv("lieux-2019.csv", delimiter=";")
r = pd.read_csv("caracteristiques-2019.csv",delimiter=";")
# merge all files together on the 'Num_Acc' key
df_pred = pd.merge(p, v, on=['Num_Acc','Num_Acc'])
df_pred = pd.merge(df_pred, l, on=['Num_Acc','Num_Acc'])
df_pred= pd.merge(df_pred, r, on=['Num_Acc','Num_Acc'])
df_pred = df_pred.drop(['Num_Acc'], axis=1)
df_pred = df_pred.drop(['etatp','lartpc','larrout','an','jour','adr',\
   'lat','long','v1','v2','vosp','pr','pr1','num_veh_x'\
     ,'num_veh_y','senc','situ'], axis=1)
df_pred['secu'] = (df_pred['secu1'] + df_pred['secu2'] + df_pred['secu3'])/3
for index, row in df_pred.iterrows():
   if row.locp == '1' or row.locp =='2': # group by "localisation sur chaussé"
       df_pred.at[index,'locp'] = '1'
   if row.locp == '3' or row.locp== '4': # group by "localisation sur passage piéton"
       df_pred.at[index,'locp'] = '2'
   if row.locp == '5' or row.locp=='6' or row.locp=='7' or row.locp== '8' \
           or row.locp=='9': # group by "localisation est divers"
       df_pred.at[index,'locp'] = '3'
   if row.locp == '0' or row.locp == '-1' : # group by null or other
       df_pred.at[index,'locp'] = '99'
df_pred = df_pred.drop(['id_vehicule_x'], axis=1)
df_pred = df_pred.drop(['an_nais'], axis=1)
df_pred = df_pred.drop(['actp'], axis=1)
df_pred = df_pred.drop(['id_vehicule_y'], axis=1)
df_pred = df_pred.drop(['voie'], axis=1)
df_pred = df_pred.drop(['dep'], axis=1)
df_pred = df_pred.drop(['com'], axis=1)
df_pred = df_pred.drop(['manv'], axis=1)
df_pred = df_pred.drop(['infra'], axis=1)

for index, row in df_pred.iterrows():
   if row.catv == '02': # group by small motorcycle
       df_pred.at[index,'catv'] = '30'
   if row.catv == '31': # group by medium motorcycle
       df_pred.at[index,'catv'] = '32'
   if row.catv == '33': # group by big motorcycle
       df_pred.at[index,'catv'] = '34'
   if row.catv == '38': # group by bus
       df_pred.at[index,'catv'] = '37'
   if row.catv == '04' or row.catv == '05' or row.catv == '06'\
```

```
        or row.catv == '08' or row.catv == '09' \
        or row.catv == '18' or row.catv == '19': # group by null or other
      df_pred.at[index,'catv'] = '99'
df_pred = df_pred.drop(['obs'], axis=1)
df_pred = df_pred.drop(['choc'], axis=1)
df_pred.grav[df_pred.grav == 1] = 0
df_pred.grav[df_pred.grav == 3] = 0
df_pred.grav[df_pred.grav == 4] = 0
df_pred.grav[df_pred.grav == 2] = 1
df_pred = df_pred.drop(['mois'], axis=1)
df_pred = df_pred.drop(['sexe'], axis=1)
df_pred = df_pred.drop(['hrmn'], axis=1)
df_pred = df_pred.drop(['prof'], axis=1)
df_pred = df_pred.drop(['surf'], axis=1)
df_pred = df_pred.drop(['obsm'], axis=1)
df_pred = df_pred.drop(['occutc'], axis=1)
df_pred = df_pred.drop(['nbv'], axis=1)

#data final
df_pred

#Total of missing values
totalmissingvaluesnosorted=df_pred.isnull().sum()
totalmissingvaluesnosorted

#Convert the categorical data into numerical before classification
from sklearn.preprocessing import LabelEncoder
data_train=df.apply(LabelEncoder().fit_transform)
data_train

#Trainig

from sklearn.model_selection import train_test_split
x=data_train.drop('grav', axis=1)
y=data_train['grav']
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=.33)


#Convert the categorical data into numerical before classification
from sklearn.preprocessing import LabelEncoder
data_test=df_pred.apply(LabelEncoder().fit_transform)
data_test

X_test=data_test.drop('grav', axis=1)

random_rf=RandomForestClassifier(criterion='entropy',max_depth=20, \
       max_features='log2', n_estimators=500)
```

```
random_rf.fit(x_train,y_train)
y_random = random_rf.predict(X_test)
y_random = pd.DataFrame({'Gravité':y_random})
map_dict = {0: 'Indemme',1:'Tué'}
y_random['Gravité'] = y_random['Gravité'].map(map_dict)

y_random.loc[y_random['Gravité'] == 'Indemme']
y_random.loc[y_random['Gravité'] == 'Tué']
```

Conclusion

Tout au long de cette étude, notre objectif était d'étudier la sinistralité automobile en France avec des méthodes de Machine Learning en utilisant des Open Data. Nous avons alors choisi d'étudier la gravité des sinistres automobiles en mettant en place des algorithmes de prédiction sur la variable grav à partir du jeu de données disponible. Pour cela, nous avons fait un certain nombre d'hypothèses et de modifications sur les données utilisées. En effet, nous avons d'abord dû retraiter certaines données, sélectionner les données à utiliser en modifiant la variable cible de départ. Les résultats obtenus sur les différents algorithmes mis en place nous ont permis de faire ressortir les différents facteurs influents de cette sinistralité avec une précision que nous pouvons qualifier de satisfaisante. Cependant, il aurait été intéressant de comparer la différence de résultats avec d'autres hypothèses, comme :
— l'imputation des valeurs manquantes au lieu d'une suppression de ces dernières, afin d'évaluer la perte d'informations résultant de cette suppression ou l'impact de l'imputation sur la prédiction
— le choix d'une modification de la variable cible en une classe combinée de blessés légers et de blessés hospitalisés d'une part, et d'autre part d'une classe combinée d'indemnes et de décédés, puisque la première est plus susceptible d'influer le portefeuille d'un assureur.
De plus les données que nous avons utilisées contiennent peu d'informations personnelles telles que le degré d'alcool des individus accidentés ou encore la vitesse des véhicules avant l'accident, et cela dans un souci de protection de la vie privée des personnes physiques. Ces données personnelles nous auraient certainement permis d'améliorer significativement la compréhension de la sinistralité car celle-ci est la combinaison de facteurs généraux et externes tels que la catégorie des routes ou les phénomènes météorologiques dont nous avons parlé dans cette étude, ainsi que de facteurs liés au conducteur et à son comportement.
Cette situation d'informations incomplètes ou partielles est l'un des inconvénients lorsque nous travaillons avec des données disponibles en Open Data car celles-ci, comme dit précédemment, ne doivent pas divulguer les informations personnelles des individus. Néanmoins, il est possible pour un assureur, en accord commun avec ses clients, d'avoir accès à ces informations dites personnelles, comme cela se fait déjà en assurance automobile même si de façon limitée, avec l'installation de capteurs, ceci permettant d'adapter la prime en fonction du comportement du conducteur.
La partie de prédiction nous a servir à savoir comment faire une prédiction sur une nouvelle Data au lieu d'utiliser une seule data et la diviser en deux parties apprentissage et test (ce qu'on fesait la plupart du temps dans notre études à l'univérsité)
Pour finir, ce bureau d'études a été une expérience très enrichissante pour nous car il nous a non seulement permis d'améliorer nos compétences informatiques, mais aussi de toucher à un sujet d'actualité qui est l'utilisation d'algorithmes de Machine Learning.

Bibliographie

[1] Documentation sur les OPEN DATA : L., B. (2017) Open Data définition : Qu'est-ce que c'est ? À quoi ça sert ?, Disponible sur : https ://www.lebigdata.fr/open-data-definition

[2] Documentation sur la SINISTRALITÉ AUTOMOBILE : Naudot, E. (2017) Statistiques des accidents de la route, Fiches-auto.fr, Disponible sur : http ://www.fiches-auto.fr/essais-tests/

Bilan 2020 de la sécurité routière, Disponile sur : https ://www.onisr.securite-routiere.gouv.fr/etat-de-l-insecurite-routiere/bilans-annuels-de-la-securite-routiere/bilan-2020-de-la-securite-routiere

Nombre de morts sur la route en France Diponible sur : https ://www.fiches-auto.fr/articles-auto/chiffres-infractions-securite-routiere/s-2317-evolution-du-nombre-de-morts-sur-la-route-france.php

Historique de la sécurité routière Disponible sur : https ://www.onisr.securite-routiere.gouv.fr/politique-de-securite-routiere/historique-de-la-securite-routiere

[3] Documentation sur les TENSORFLOW et KERAS :
Keras (2017) Keras : The Python Deep Learning library, Disponible sur : https ://keras.io/TensorFlow Site Officiel, Disponible sur : https ://www.tensorflow.org/

[4] Documentation sur les MÉTRIQUES :
Joshi, R. (2016) Accuracy, Precision, Recall F1 Score : Interpretation of Performance Measures, Disponible sur :
http ://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

[5] Documentation sur les FORÊTS ALÉATOIRES :
Vaudor, L. (2015) Classification par forêts aléatoires, R-atique, Disponible sur : http ://perso.ens-lyon.fr/lise.vaudor/classification-par-forets-aleatoires/

[6] Documentation sur le MODÈLE GRADIENT BOOSTING :
RAKOTOMALALA, R. (2016) Gradient Boosting : Technique ensembliste pour l'analyse prédictive - Introduction explicite d'une fonction de coût, Disponible sur :
http ://eric.univ-lyon2.fr/ ricco/cours/slides/gradient$_b$*oosting.pdf*

[7] Documentation sur les RÉSEAUX DE NEURONES :
Université de Toulouse (2016) Réseaux de neurones, Wiki Stat, Disponible sur : https ://www.math.univ-toulouse.fr/ besse/Wikistat/pdf/st-m-app-rn.pdf Vermet, F. (2017), Cours sur l'Apprentissage statistique

[8] Documentation sur Power BI Disponible sur :
https ://docs.microsoft.com/fr-fr/power-bi/fundamentals/power-bi-overview