UNIVERSITÉ DE MONTPELLIER

FACULTÉ D'ÉCONOMIE
Université de Montpellier

Projet Data sous Python

Présenté par :

Mamadou DIOP

Thierno Ibrahima BALDE

Année académique 2022-2023

INTRODUCION

L'objectif de cette étude consiste en la prédiction de la sinistralité des automobiles en France. Pour ce faire nous avons à notre disposition un jeu de données qui regroupe cinq sous-bases que sont USAGERS, CARACTERISTIQUES, LIEUX, VEHICULES, PRISE EN CHARGE. Il sera alors question, dans un premier temps, d'analyser, de traiter éventuellement ces données disponibles. Ce traitement consiste en la recherche de valeurs manquantes, de valeurs aberrantes ou des variables non significatives. Il serait également possible de chercher des données externes pour compléter notre analyse. Pour avoir une vision plus large sur la sinistralité automobile, nous allons effectuer des analyses descriptives sur notre jeu de données afin d'en tirer les premières informations avant l'étape de la prédiction. Ainsi, 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.

Après cette étape d'exploration des données (EDP), la deuxième étape de cette étude s'intéressait à la construction et à la prédiction de la sinistralité automobile par des modèles prédictifs de Machine Learning. Nous allons également évaluer la qualité de nos modèles utilisés et de les comparer entre eux.

Pour rappel, il est possible que certaines de nos variables ne soient pas utiles pour la modélisation mais qui peuvent l'être dans la partie analyse exploratoire de données (univariée et multivariée).

PARTIE I

1. Analyse exploratoire des données

Comme énoncé plus haut, nous nous intéressons à l'analyse exploratoire des données. En effet, après une première visualisation graphique avec ***seaborn***, on constate que notre jeu de données contient, pour certaines variables, beaucoup de valeurs manquantes. Pour information, ces données manquantes peuvent être expliquées par le fait qu'on a fusionné cinq sous-bases de taille différente.

```
print(df1.shape)
print(df2.shape)
print(df3.shape)
print(df4.shape)
print(df5.shape)
```

```
(105295, 15)
(81066, 12)
(47744, 15)
(47773, 18)
(29570, 2)
```

Nous avons décidé de ne pas les traiter individuellement par peur de tirer des conclusions non pertinentes ou fallacieuses.

Ces valeurs manquantes sont néfastes au bon fonctionnement des algorithmes de prédiction étant donné que certains algorithmes de machine learning ne peuvent pas s'exécuter en leur

présence. Pour remédier à ce problème, nous allons procéder à éliminer toutes les variables dont les valeurs manquantes sont supérieures à un certain de seuil.

```
#Supprimer les variables qui ont dépassé certains seuils de valeurs manquantes

d = d.drop(d.columns[d.isna().sum()/d.shape[0]>0.9], axis =1)
```

Ensuite, on a scindé notre base en deux groupes (un groupe pour les variables numériques et un autre pour les variables catégorielles). Pour les variables catégorielles, nous remplaçons les valeurs manquantes par les valeurs qui répètent le plus.

```
### Pour les variables catégorielles remplaçons les NaN pour les valeurs qui répètent le plus

cat_df = cat_df.apply(lambda x : x.fillna(x.value_counts().index[0]))
cat_df.isnull().any()
```

On a pu constater également des valeurs aberrantes pour certaines variables. Par exemple dans le cas de la variable ***an_nais*** (année de naissance). On trouve également des usagers où l'année de naissance est supérieure à 2010 ce qui paraît illogique du point de vue juridique et de la loi française. Dans ce cas, nous avons décidé de supprimer toutes les variables dont l'année de naissance est supérieure ou égale à 2014 pour éviter de ne pas faire notre analyse que sur les usagers majeurs. Nous supprimons également tous les individus dont l'âge est supérieur à 85ans après avoir créé une variable ***age_conduct*** en prenant comme année de référence 2020.

```
#Supprimer tous les usagers où l'année de naissance est supérieur ou égal à 2014
df.drop(df[df['an_nais']>=2014].index, inplace =True)
```

1.1. **Visualisation des données**

Nous allons analyser notre jeu de données grâce à la visualisation des données de certaines variables afin d'en tirer des informations qui nous permettraient de mettre en place des modèles de prédiction.

1.2. Répartition des accidents par type d'usagers

Nous utilisons un graphique en barres pour montrer la répartition des accidents par type d'usagers. Les usagers les plus accidentés sont les conducteurs. Ce résultat n'est pas surprenant, dans le sens que nous étudions la sinistralité automobile. Cependant, la comparaison entre la proportion de blessés légers et d'indemnes chez les conducteurs et chez les passagers nous permet d'observer une différence entre les modalités de ces deux classes : chez les conducteurs, la classe la plus représentée est la classe des indemnes tandis que chez les passagers et piétons, ce sont les blessés légers. De plus, chez les piétons, nous pouvons remarquer une très faible proportion d'indemnes, il s'agit de la classe la moins représentée

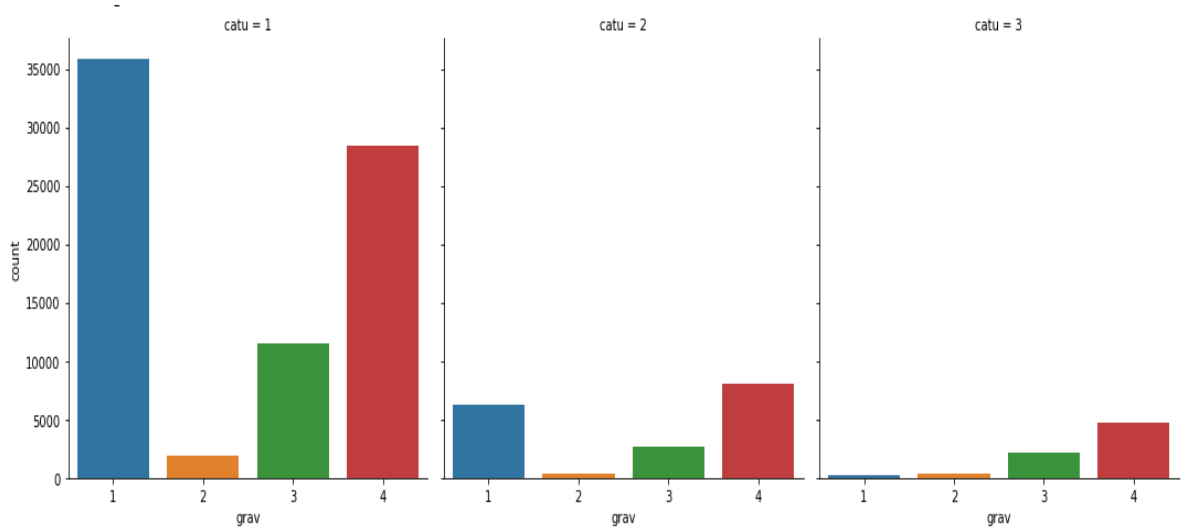


Figure 1 : Répartition des accidents par type d'usagers

1.3. Répartition des accidents par sexe

Le nombre de femmes accidentées est de moitié celui des hommes. Le nombre de blessés légers chez les femmes, est également de moitié celui des hommes. De plus, la classe la plus représentée est la classe des indemnes chez les hommes, alors que chez les femmes, ce sont les blessés légers.

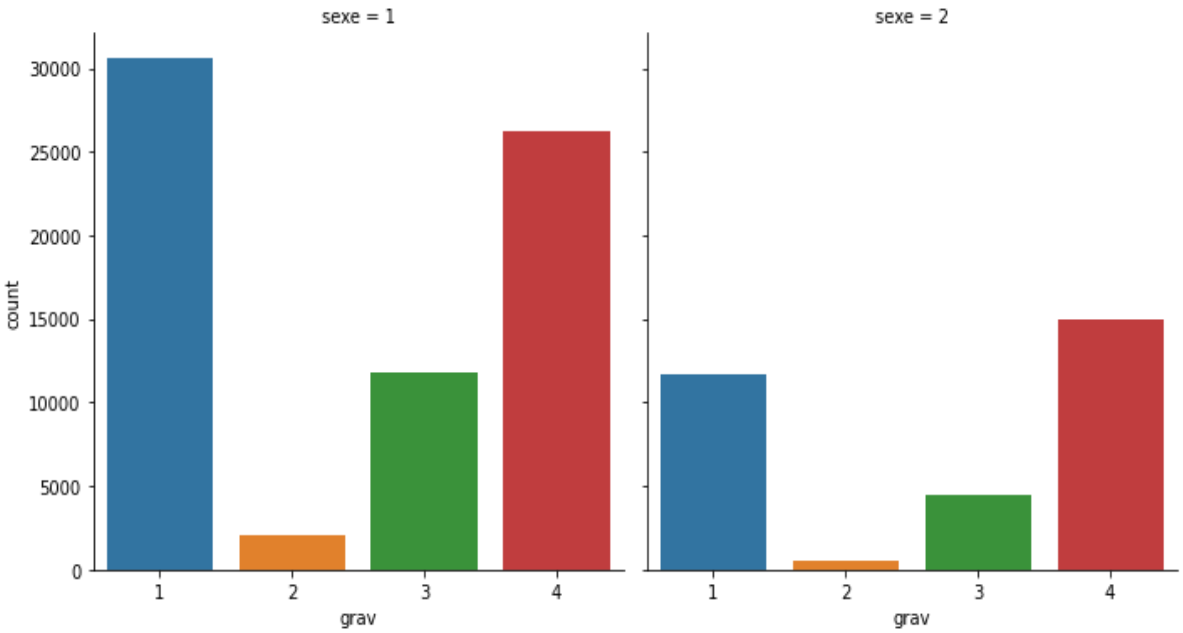


Figure 2 : Répartition des accidents par sexe

1.4. Répartition des accidents par conditions atmosphériques

Le graphique ci-dessous nous montre que la plupart des accidents soit plus de 70% ont eu lieu en temps normal et avec plus de 10% en temps de pluie légère

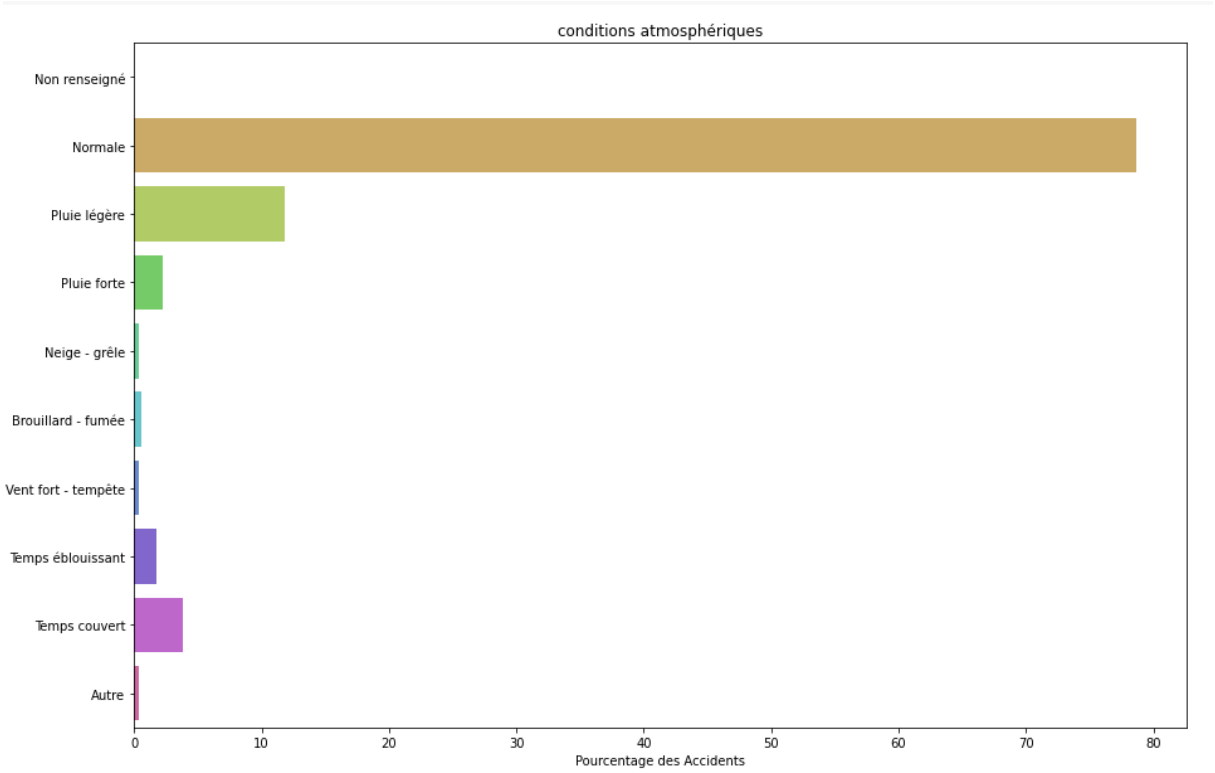


Figure 3 : Répartition des accidents par conditions atmosphériques

1.5. Répartition des accidents par année de naissance

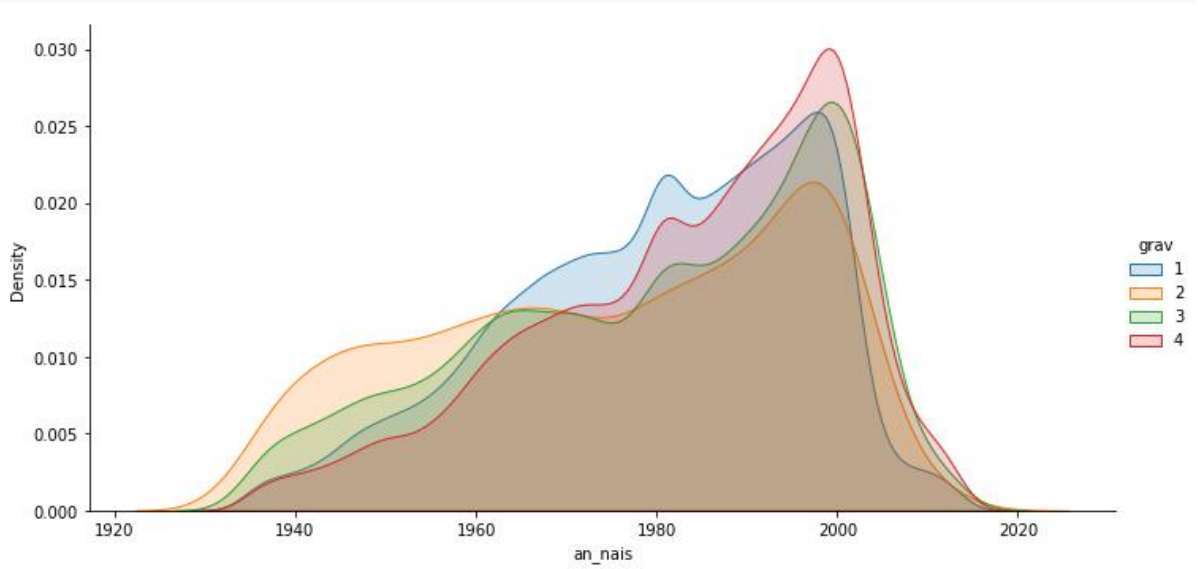


Figure 4 : Répartition des accidents par année de naissance

D'après ce graphique, on constate que les jeunes sont les plus accidentés, avec un fort taux de mortalité autour des années 2000. Ce résultat peut être mis en comparaison avec le taux de mortalité en France. En effet, la mortalité routière est une des causes importantes de mortalité en France, avec 0.5% de la mortalité due à des accidents de la route. De plus, le risque de mortalité général augmente de manière exponentielle à partir des années 90, contrairement à la mortalité routière qui baisse après cet âge.

1.6. Répartition des accidents par catégories de routes

La plupart des accidents ont lieu sur les voies communales et sur les routes départementales comme le montre le graphique ci-dessous. On peut également constater qu'il y'a moins d'accidents en dehors du réseau public par rapport autres catégories de route.

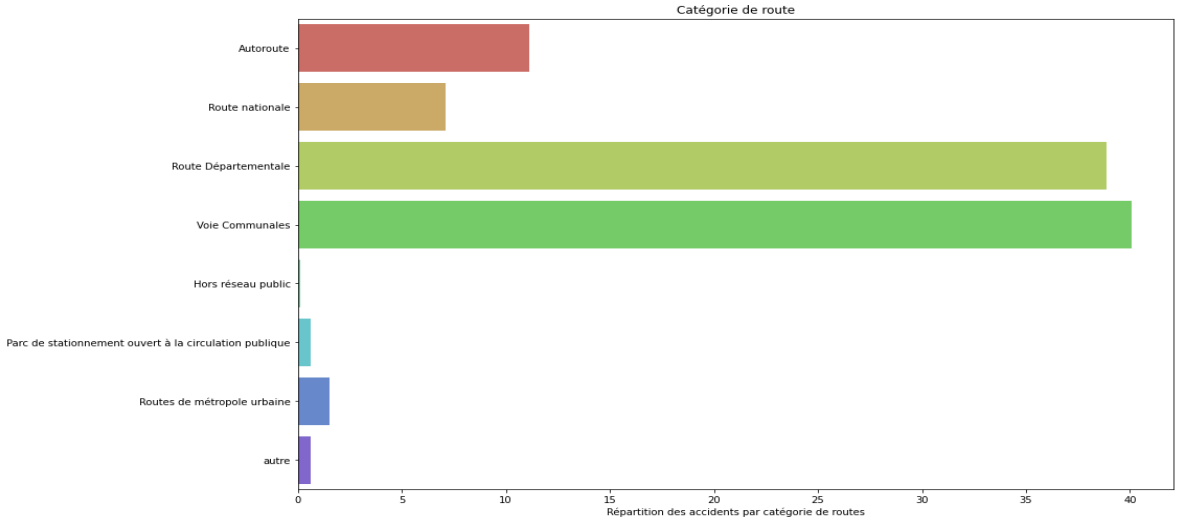


Figure 5 : Répartition des accidents par catégories de routes

- **Le pourcentage des accidents mensuels**

La plupart des accidents se produisent à la période estivale soit plus de 15% et au mois de janvier avec à peu près 11% comme l'illustre le graphique ci-dessous.

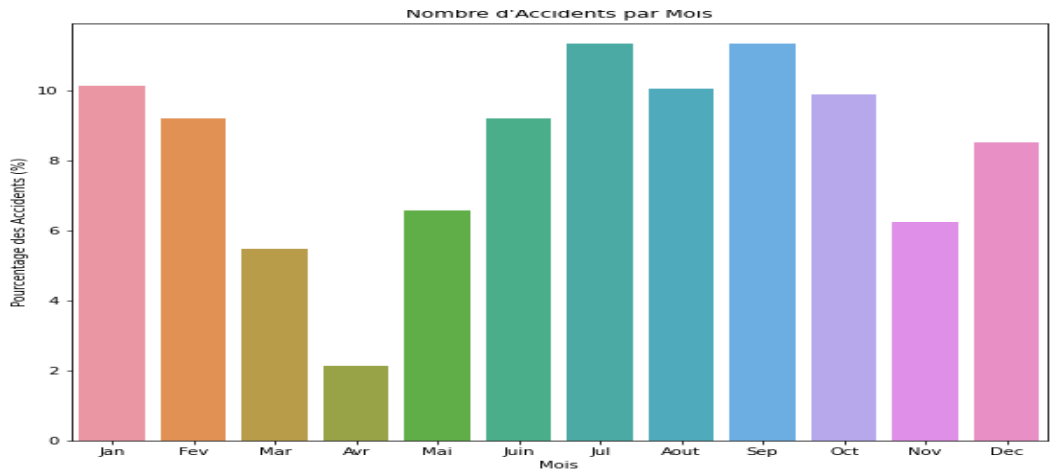


Figure 6 : Le pourcentage des accidents mensuels

Partie II : Modélisation

Notre variable target grav à prédire, est une variable à quatre modalités, avec une classe minoritaire : la classe des tués. Nous avons mis en place des modèles, dans un premier temps avec les quatre modalités et on a pu constater que les erreurs de prédictions étaient tellement élevées.

La précision était bonne sur la classe majoritaire c'est-à-dire sur la classe des indemnes, Aucun des algorithmes que nous avons mis en place lors de la tentative de construction d'algorithmes n'améliore les résultats. Nous étions donc face à un problème lié à la détection d'évènements rares, un problème classique en prédiction qui intervient généralement lorsque la modalité d'intérêt est peu représentée. Nos tentatives de rééquilibrage de la base de données n'ont donc pas permis d'améliorer ces résultats. Ainsi, nous avons décidé de recoder notre variable cible en une variable binaire en créant une fonction nommée gravité :

- Indemne, qui regroupe les indemnes, avec la modalité "1"
- Victime, qui regroupe les blessés légers, hospitalisés, et les décédés, avec la modalité "0"

```python
def gravité(df):
  if df['grav'] == 1:
    return 'Indemne'
  else:
    return 'Victime'
```

```python
df['grave'] = df.apply(gravité, axis = 1 )
```

```python
df['grave'] = df.apply(lambda x: 1 if x['grave']=='Indemne' else 0, axis = 1 )
```

2. Analyse et évaluation de nos modèles de prédiction

Nous avons utilisé la matrice de confusion et le taux d'erreur pour la plupart de nos modèles de prédiction pour évaluer leurs performances. En effet, La matrice de confusion est un outil qui permet de savoir à quel point le modèle de machine learning est « confus », ou qu'il se trompe. Il s'agit d'un tableau avec en colonne les différents cas réels et en ligne les différents cas d'usage prédits.

1.1. Les métriques utilisées

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 "0" et celle de la classe prédite est également "0".—

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

	Predict victime	Predict indemne
victime	TP	FN
Indemne	FP	TN

➢ Accuracy

L'Accuracy 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.

$$Accuarcy = \frac{TP + TN}{TP + FP + TN + FN}$$

- 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)

$$Precision = \frac{TP}{TP + FP}$$

Exemple pour le modèle random forest : $Precision = \frac{9966}{9966+1915} = 0.84$

- 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).

$$Recall = \frac{TP}{TP + FN}$$

Exemple pour le modèle random forest : $Recall = \frac{9966}{9966+2139} = 0.82$

- F1-score

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.

$$Score\ f1 = 2 * \frac{2 * TP}{2 * TP + FP + FN}$$

Exemple pour le modèle random forest : $Score\ F1 = \frac{2*9966}{2*9966+1915+2139} = 0.83$

1. Forêt aléatoire

Une forêt aléatoire ou random forest est une méthode d'apprentissage supervisé extrêmement utilisée par les data scientists. En effet, cette méthode combine de nombreux avantages dans le cadre d'un apprentissage supervisé. 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 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 (la variable **grav** dans notre cas) 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.

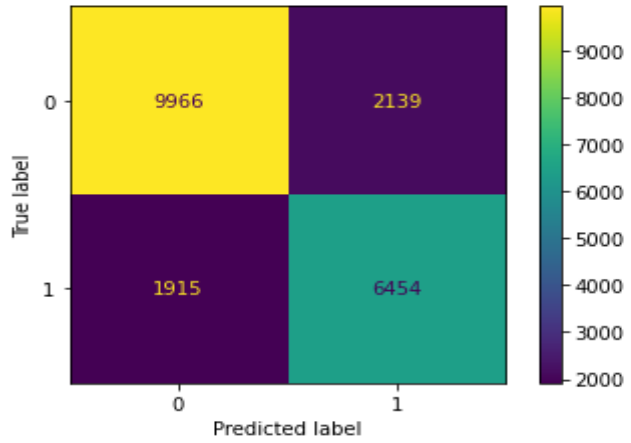
On constate que parmi les 12105 usagers victimes d'accidents, on a correctement identifié que 9966 usagers avec une précision de prédiction de 84% et 2139 usagers qui sont prédits indemnes

Cependant, parmi les 8369 usagers identifiés comme indemnes, il n'y a que 6454 usagers qui ont été correctement prédits comme indemnes par le modèle avec une précision de 75% et les 1915 sont prédits par le modèle comme victimes d'accidents.

```
### Matrice de confusion du modèle RF
prediction_rf = rf.predict(X_test)

mc = confusion_matrix(y_test,prediction_rf)
print (classification_report(y_test, prediction_rf))
```

```
              precision    recall  f1-score   support

           0       0.84      0.82      0.83     12105
           1       0.75      0.77      0.76      8369

    accuracy                           0.80     20474
   macro avg       0.79      0.80      0.80     20474
weighted avg       0.80      0.80      0.80     20474
```

```
### Matrice de confusion du modèle RF en figure

plot_confusion_matrix(rf, X_test, y_test)
plt.show()
```



Le taux d'erreur est de 19.8%

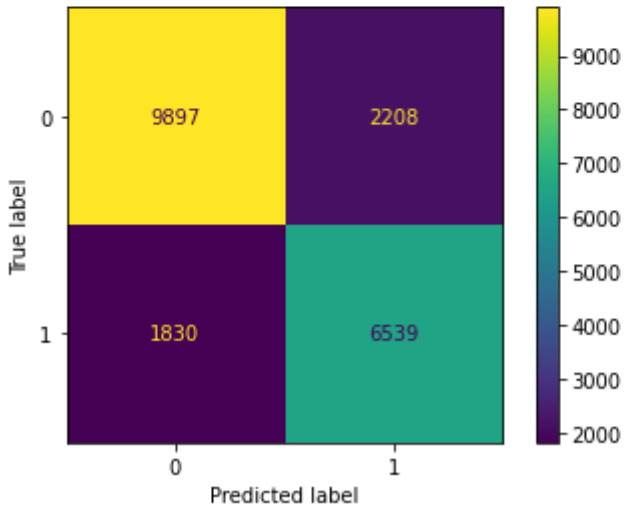
2. Modèle Grandient Boosting

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 nœud 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.

```
predict_GBC = GBC.predict(X_test)
mc_GBC = confusion_matrix(y_test,predict_GBC)
print (classification_report(y_test, predict_GBC))
```

```
              precision    recall  f1-score   support

           0       0.84      0.82      0.83     12105
           1       0.75      0.78      0.76      8369

    accuracy                           0.80     20474
   macro avg       0.80      0.80      0.80     20474
weighted avg       0.80      0.80      0.80     20474
```

```
plot_confusion_matrix(GBC, X_test, y_test)
plt.show()
```



On constate que parmi les 12105 usagers victimes d'accidents, on a correctement identifié que 9897 usagers avec une précision de prédiction de 84% et 2208 usagers qui sont prédits indemnes

Cependant, parmi les 8369 usagers identifiés comme indemnes, il n'y a que 6539 usagers qui ont été correctement prédits comme indemnes par le modèle avec une précision de 75% et les 1830 sont prédits par le modèle comme victimes d'accidents.

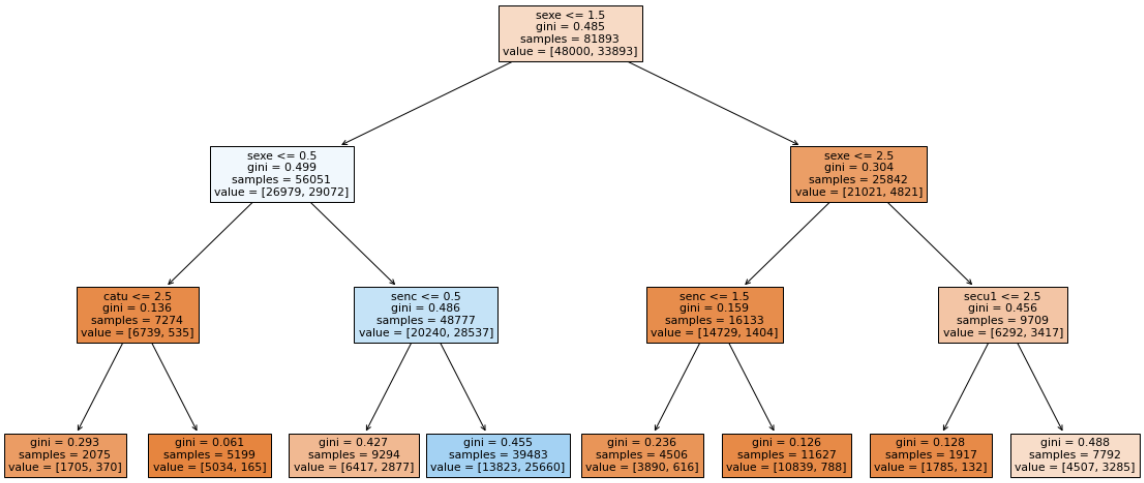
Le taux d'erreur est de 19.72%

3. Décision Tree

Cet outil d'aide à la décision ou d'exploration de données permet de représenter un ensemble de choix sous la forme graphique d'un arbre. C'est une des méthodes d'apprentissage supervisé les plus populaires pour les problèmes de classification de données.

Concrètement, un arbre de décision (Decision Tree en anglais) modélise une hiérarchie de tests pour prédire un résultat. Il existe deux principaux types d'arbre de décision : La régression et la classification.

Dans notre cas on a affaire à une classification qui permet de prédire à quelle classe la variable de sortie appartient.



On constate que parmi les 12105 usagers victimes d'accidents, on a correctement identifié que 8688 usagers avec une précision de prédiction de 81%% et 3417 usagers qui sont prédits indemnes

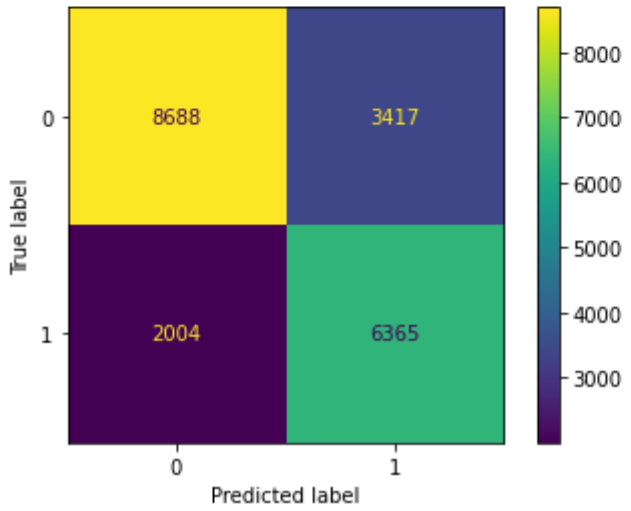
Cependant, parmi les 8369 usagers identifiés comme indemnes, il n'y a que 6365 usagers qui ont été correctement prédits comme indemnes par le modèle avec une précision de 65% et les 2004 sont prédits par le modèle comme victimes d'accidents.

Le taux d'erreur est de 26.5%

```
prediction_DT = DTree.predict(X_test)

mc = confusion_matrix(y_test,prediction_DT)
print (classification_report(y_test, prediction_DT))
```

```
              precision    recall  f1-score   support

           0       0.81      0.72      0.76     12105
           1       0.65      0.76      0.70      8369

    accuracy                           0.74     20474
   macro avg       0.73      0.74      0.73     20474
weighted avg       0.75      0.74      0.74     20474
```

```
plot_confusion_matrix(DTree, X_test, y_test)
plt.show()
```



Comparaison des modèles

Modèle	Accuracy	Recall	Score F1	Taux d'erreur
Random Forest Classifier	80%	82%	83%	19.8%
Bagging Tree	79%	82%	82%	20.8%
Gradient Boosting	80%	82%	83%	19.7%
Decision Tree Classifier	74%	72%	76%	26.5%

Parmi les modèles mis en place, on constate que les modèles les plus performants sont ceux sous Random Forest Classifier et Gradient Boosting, mais les autres modèles restent néanmoins acceptables.

Références

https://linuxtut.com/fr/d211b3bfce440acc1b19/

https://towardsdatascience.com/random-forest-in-python-24d0893d51c0

http://eric.univ-lyon2.fr/~ricco/cours/supports_data_mining.html

https://scikit-learn.org/stable/modules/ensemble.html#gradient-tree-boosting

http://eric.univ-lyon2.fr/~ricco/cours/slides/bagging_boosting.pdf

https://euria.univ-brest.fr/digitalAssets/73/73839_BE11.pdf