Rapport du projet statistique 5FM

Prédiction du coût moyen des sinistres en assurance automobile

Encadré par :

- M. Fouad MARRI
- M. Saad BENJELLOUN

Réalisé par :

- Maryame KOURAM
- Hafsa AKEBLI
- Oussama ENNAJAR
- Nedson SIMO MABOU

I. Table des matières

II. Introduction 5
III. Exploration de données 6
Description des données 6
Types des variables 6
Nettoyage de données 6
Traitement des NaN : 6
Traitement de la variable « Police » 7
Traitement de la variable « Type » 7
Traitement de la variable « Age conducteur » 7
Traitement de la variable « Age du véhicule » 8
Traitement de la variable « Région » 8
Traitement de la variable « Puissance Fiscale » 8
Feature enginering 8
Traitement d'autres anomalies : 9
Suppression de features : 10
Traitement des doublons : 11
Suppression des outliers: 11
Exploration univariée des données 11
Distplots 11
La feature « Durée_effet » 13
La feature "puissance fiscale" 13
La feature "âge conducteur" 14
La feature "âge véhicule" 14
La feature "exposition" 14
La feature « charge new » 15
La feature « nombre de sinistres » 15
Fréquence des sinistres 16
Sévérité 16
La feature « Type » 17
La feature "Region2" 17
Exploration bivariée des données 17

Correlations 17
Répartition des types de véhicules sur les régions du royaume 18
Moyennes de variables numériques par type de véhicule 18
Statistiques descriptives par type de véhicule 18
Statistiques descriptives par région 19
Barplots 20
Statistiques descriptives par tranche d'âge du véhicule 21
Statistiques descriptives par tranche de puissance fiscale 21
Statistiques descriptives par tranche d'âge du conducteur 22
Catplots 23
Durée effet – sévérité – fréquence des sinistres 24
IV. Tests 25
Loi d'évolution de la sévérité 25
L'inférence statistique 25
RMSE 25
Quelques lois de probabilité 25
Loi normale 25
Loi gamma 26
Loi exponentielle 27
Loi de khi 2 27
Loi lomax 28
Implémentation sous python 29
Description 29
Résultats 30
Loi normale $N\mu, \sigma : \mu\ ,\ \sigma\ = (183.03\ , 1115.81)$ 31
Loi gamma $\Gamma k, \lambda$: 31
Loi exponentielle $E\ (\lambda) :\ \lambda = 183.03$ 32
Test de normalité : Test de D'Agostino-Pearson 32
Test d' « exponentialité »: Test d' Anderson-Darling 34
V. Modélisation de la sévérité : Modèles Linéaires Généralisés (GLM) 35
Introduction aux GLM 35
Définition mathématique d'un GLM 36

La distribution 36
Le prédicateur linéaire 36
La fonction lien 37
Estimation des coefficients 37
Qualité d'ajustement 38
La déviance 38
Le test de Pearson 38
L'Akaike Information Criterion (AIC) et le Bayesian Information Criterion (BIC) 39
Implémentation sous python 39
GLM 39
VI. Application des modèles Machine Learning 43
Régression linéaire 44
Coefficients de la régression linéaire 44
Performance et prédiction 45
Light GBM 45
Paramètres de Tuning 45
Paramètres de Fitting 46
Résultats 46
XGBoost 46
Paramètes de Tuning 46
Résultats 47
Conclusion 47

II. Introduction

L'assurance est par définition un système qui permet de prémunir un individu, une association ou une entreprise contre les conséquences financières et économiques liées à la survenance d'un risque, événement aléatoire particulier. L'assurance, un engagement inscrit dans un contrat entre assureur et assuré servant à couvrir ce dernier lors d'un événement particulier et incertain en l'échange d'une prime. Cet événement particulier et incertain s'appelle risque. Or, le risque est consubstantiel à l'activité humaine. A partir du moment où le risque est au cœur de l'activité humaine, il est normal de le retrouver dans la gestion des activités économiques et notamment des entreprises, des banques et des compagnies d'assurance. En revanche, la gestion comptable d'une compagnie d'assurance semble être très difficile compte tenu du fait que celle-ci reçoit les primes de ses clients (frais d'assurance payés régulièrement par l'assuré) avant de payer les prestations aux clients. Ceci dit, son cycle est inverse et l'oblige ainsi à définir une stratégie stricte afin de rester durable, en d'autres termes, être capable de faire face à ses obligations à l'échéance vis-à-vis des assurés à tout moment en veillant à ce que la réserve ne tombe pas en dessous de 0.

En outre, une compagnie d'assurance devrait être rentable après tout, et alors doit être capable de rémunérer ses employés et actionnaires et dégager des bénéfices. Partant, la définition de risque en assurance est la probabilité que la réserve de la compagnie d'assurance (la différence entre le total des primes reçues et le total des montants des prestations payées) devienne négative à un certain temps. C'est à ce moment-ci qu'on parle de ruine, du fait du mauvais calcul de la cotisation individuelle des clients ou trop de sinistres à couvrir. La théorie mathématique de l'assurance peut contribuer à promouvoir le développement de méthodes plus rationnelles dans la gestion des risques. Les responsables et décideurs dans les compagnies d'assurance seraient ainsi mieux à même d'intégrer dans leurs démarches le faut que le risque, bien quantifié et apprécié, constitue aussi, sinon davantage, une opportunité d'innovation, une source de création de richesse, donc de progrès pour nos sociétés.

Dans le cadre du module Statistique, nous avons travaillé sur le projet 5FM, dans lequel on était demandé de prédire le coût moyen des sinistres dans une assurance automobile en utilisant une dataset contenant 11 variables. Pour réaliser ceci, nous avons utilisés nos connaissances en statistiques acquises pendant les cours de M. Saad BENJELLOUN, et nous avons suivi les étapes indiquées par notre encadrant M. Fouad MARRI. Nous avons commencé par nettoyer les données (suppression des doublons, traitement des NaN, traiter les cas particuliers de chaque variable), ensuite on est passé au feature engineering pour construire de nouvelles variables intéressantes en fonction des anciennes variables, après nous avons fait une description univariée et bivariée des variables , ensuite nous avons fait des tests pour trouver la famille de loi la plus proche à la distribution de notre variable dépendante (à prédire) qui est dans notre cas la « sévérité », nous avons ensuite construit le modèle GLM et appliqué également les modèles machine learning pour prédire la sévérité, on a comparé entre les résultats que donnent les différents modèles pour choisir le meilleur.

Nous remercions M. Fouad MARRI pour son encadrement, et également M. Saad BENJELLOUN pour ces cours théoriques en statistiques qui nous ont été très utiles et intéressants.

III. Exploration de données

Description des données

Notre base de données contient 11 colonnes et 250000 lignes. Les différentes variables et leurs significations sont :

Police : Id
date_debut_effet : Date du début du contrat
date_fin_effet : Date de la fin début du contrat
Type : Type du véhicule
puissance_fiscale : Nombre de chevaux du véhicule (propriété intrinsèque au véhicule)
AGE_cond : L'âge du conducteur
age_vehicule : L'âge du véhicule
Region2 : Région à laquelle appartient le conducteur
charge_new : Montant total payé par l'assurance automobile pendant la période précisée
nbre_sin : Nombres d'accidents qui ont eu lieu pendant la période précisée
exposition : Période du contrat/12 mois

Types des variables

Voici les types des différentes variables :

```
police                int64
date_debut_effet      int64
date_fin_effet        int64
Type                 object
puissance_fiscale   float64
AGE_cond            float64
age_vehicule        float64
Region2              object
charge_new          float64
nbre_sin            float64
exposition          float64
dtype: object
```

Nettoyage de données

Traitement des NaN :

Cherchons tout d'abord les NaN existants dans notre base de données :

```
police                0
date_debut_effet      0
date_fin_effet        0
Type                  3
puissance_fiscale     3
AGE_cond              5
age_vehicule          3
Region2               5
charge_new       237103
nbre_sin         237103
exposition            0
dtype: int64
```

On remplace les NaN de charge_new et nbre_sin par des 0 puisqu'un conducteur qui a une charge_new vide veut dire qu'il n'a rien payé durant la durée de son contrat et donc il a payé 0, et on applique la même logique pour nbre_sin.

Et on supprime les lignes contenant tous les autres NaN. On trouve finalement une dataset contenant 249992 lignes c'est-à-dire qu'on a perdu 8 lignes.

Traitement de la variable « Police »

On a voulu nous assurer que Police est effectivement un Id, c'est-à-dire une valeur unique pour chaque ligne :

```
# unicité de chacune des valeurs
df.police.is_unique
```

```
True
```

D'où police est effectivement équivalente à un id, on peut donc nous en passer et supprimer cette variable.

Traitement de la variable « Type »

En comptant les valeurs pour chaque Type on retrouve :

```
G    184792
E     65197
W         3
Name: Type, dtype: int64
```

On remarque qu'un 3ème type de carburant s'est infiltré dans la liste qui ne devrait contenir que des 'G' pour gasoil et 'E' pour essence. Nous effacerons donc les enregistrements ayant le Type 'W'.

Traitement de la variable « Age conducteur »

On compte le nombre de lignes de la variable âge conducteur pour différents intervalles :

```
18 <= age conducteur <= 90 :  247640
age conducteur négatif :  9
0 < age conducteur < 18 :  16
age conducteur > 90 :  2340
```

On décide de supprimer donc les lignes appartenant aux 3 derniers intervalles, c'est-à-dire les lignes dont l'âge conducteur est négatif, supérieur à 90 ans, ou inférieur à 18 ans vu que les mineurs n'ont pas le droit de conduire.

Et voici la taille de notre base de données après ce petit traitement :

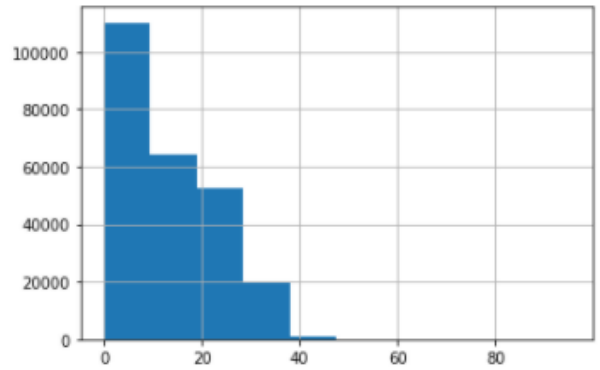
```
(247624, 11)
```

Traitement de la variable « Age du véhicule »

En visualisant l'âge véhicule dans un histogramme on retrouve :

```
df.age_vehicule.hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7bf19215d0>
```



On remarque qu'il y'a des âges de véhicule supérieurs à 30 ans. Nous décidons donc de supprimer ces enregistrements.

Traitement de la variable « Région »

On compte le nombre d'enregistrements par région pour voir les régions existantes :

```
RABAT-GHARB-MEKNES-SETTAT-FES-SUD          79642
CASA-TANGER-TETOUAN                        75558
MARRAKECH -SOUSS -ORIENTAL                 54573
DOUKKALA-KHOURIBGA-TADLA-TAZA-INDEFINIE    26031
Name: Region2, dtype: int64
```

On ne trouve donc aucun défaut de formatage majeur à corriger, à part l'espace qu'il faut supprimer pour la région 'MARRAKECH -SOUSS -ORIENTAL' pour que son écriture soit de la même forme que les autres régions.

```
df.Region2 = df.Region2.where(df.Region2 != "MARRAKECH -SOUSS -ORIENTAL", other= "MARRAKECH-SOUSS-ORIENTAL")
```

Traitement de la variable « Puissance Fiscale »

Pour la puissance fiscale on décide d'éliminer les valeurs n'appartenant pas à l'intervalle :
[moyenne -3 écart type ; moyenne + 3 écart type] qui représentent des outliers

Feature enginering

Feature Durée_effet

A partir des informations date_debut_effet et date_fin_effet l'on peut calculer la durée de contrat pour chaque enregistrement, information qui nous paraît plus pertinente pour notre modélisation.

Pour créer cette variable on convertit les colonnes date_debut_effet et date_fin_effet en type Datetime. On crée ensuite la feature « durée_effet », on la calcule en jours.

```
duree = pd.DataFrame(data=date_debut_fin.date_fin_effet - date_debut_fin.date_debut_effet, columns=['duree_effet'])
duree.reset_index(inplace=True, drop=True)
```

```
D = []
for i in range(len(duree.duree_effet)):
  D.append(duree.duree_effet[i].days)
```

```
df['duree_effet'] = D
```

Feature fréquence des sinistres

On introduit également la variable fréquence. La fréquence des sinistres correspond au rapport du nombre de sinistre par l'exposition. Cette variable donne un sens à l'exposition, qui prise seule est d'une information moins pertinente. Exposition sera donc supprimée par la suite. De même pour nbre_sinistre.

```
df.reset_index(inplace=True, drop=True)
L = []
for i in range(len(df.nbre_sin)):
  if df.nbre_sin[i] == 0 or df.exposition[i] == 0:
    L.append(0)
  else:
    L.append(df.nbre_sin[i]/df.exposition[i])
```

```
df['frequence_sin'] = L
```

Feature sévérité

Puis nous introduisons la variable « sévérité » qui correspond au coût moyen d'un sinistre pour un assuré (objet de modélisation). C'est le rapport de charge_new par le nbre de sinistre. La charge_new n'interviendra donc plus dans la suite et sera supprimée.

```
severite = []
for i in range(len(df)):
   if df['nbre_sin'][i] != 0 :
     severite.append(df['charge_new'][i] / df['nbre_sin'][i])
   else :
     severite.append(0)
```

```
df['severite'] = severite
```

Traitement d'autres anomalies :

Durée_effet nulle

On cherche à compter les enregistrements pour lesquels duree_effet = 0 c'est-à-dire date_debut_effet égale à date_fin_effet

```
# contrat à durée nulle
df[df.duree_effet == 0].count()
```

```
date_debut_effet     6
date_fin_effet       6
Type                 6
puissance_fiscale    6
AGE_cond             6
age_vehicule         6
Region2              6
charge_new           6
nbre_sin             6
exposition           6
duree_effet          6
frequence_sin        6
severite             6
dtype: int64
```

On trouve donc 6 enregistrements pour lesquels duree_effet est nulle, on va donc supprimer ces anomalies.

Anomalies par rapport à charge_new et nbre_sin

Il existe dans la dataset des enregistrements pour lesquels nbre_sin=0 et charge_new non nulle ce qui est illogique, car si une personne a eu un accident, l'assurance va bien sûr payer un montant (charge_new).

```
# verification s'il existe dans la data des enregistrements 'nbre_sin'==0 et 'charge_new'!=0
df[(df.nbre_sin == 0) & (df.charge_new != 0)]
```

On trouve 2145 enregistrements contenant cette anomalie que nous allons bien sûr supprimer.
De même on a trouvé des enregistrements pour lesquels nbre_sin non nul et charge_new=0, on va les supprimer également.

Suppression de features :

Après le feature engineering et le traitement des anomalies, on supprime les variables dont nous avons plus besoin qui sont 'exposition', 'date_debut_effet', 'date_fin_effet', 'charge_new' et 'nbre_sin'

```
df = df.drop(columns=['exposition','nbre_sin','charge_new','date_debut_effet','date_fin_effet'])
```

On obtient donc une dataset contenant les colonnes suivantes :

```
list(colonnes)
```

```
['Type',
 'puissance_fiscale',
 'AGE_cond',
 'age_vehicule',
 'Region2',
 'duree_effet',
 'frequence_sin',
 'severite']
```

Traitement des doublons :

On compte le nombre des doublons existants :

```
list(df.duplicated(subset=list(colonnes))).count(True)
```

```
64462
```

Il y a 64462 reprises de ligne à l'identique, y compris la feature 'sévérité'. L'on n'en conserve qu'un seul exemplaire pour chaque enregistrement.

Suppression des outliers:

```
df = df[df.severite <= 15000]
```

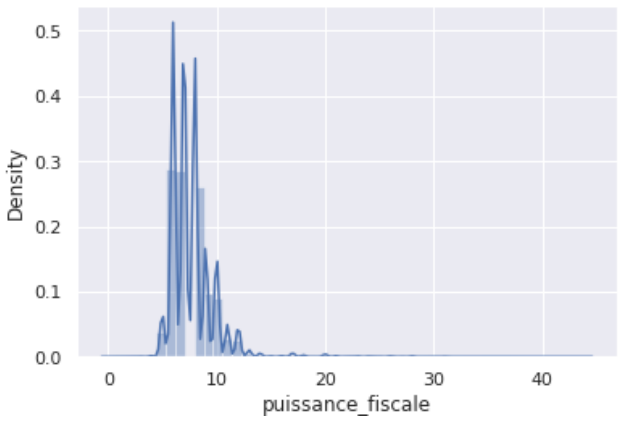
On supprime les enregistrements pour lesquels la sévérité est supérieur à 15000. C'est le même principe que nous avons appliqué pour la puissance fiscale, c'est-à-dire on supprime les valeurs n'appartenant pas à l'intervalle **[moyenne -3 écart type ; moyenne + 3 écart type],** seulement pour la sévérité l'écart type est tellement grand par rapport à la moyenne (ce qui explique le choix de 15000). En prenant les lignes dont sévérité<=15000 on prend 98.6% de la data.

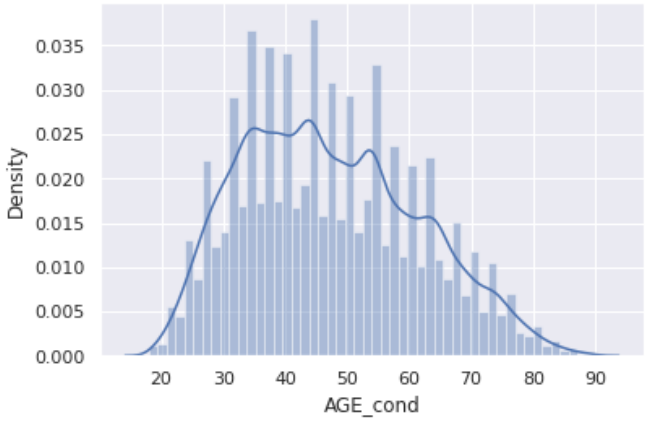
Exploration univariée des données

Grâce à l'analyse univariée et multivariée des données, on peut déduire certaines hypothèses qui pourront servir lors de la phase de modélisation.

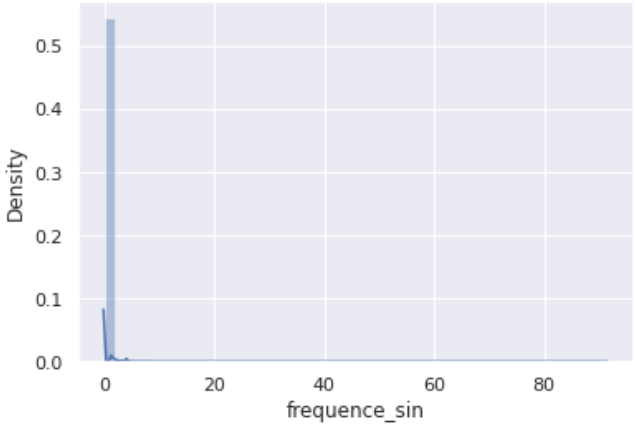
L'analyse univariée permet d'explorer une seule feature à la fois. Cette analyse se base sur les statistiques descriptives. Ces dernières permettent de tirer des indications concises sur une feature donnée. Parmi ces indicateurs, on retrouve la moyenne, la médiane ainsi que les mesures de dispersion de données.

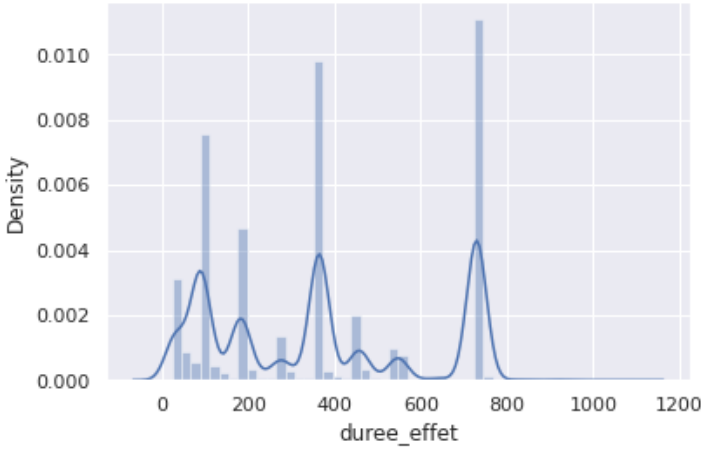
Distplots

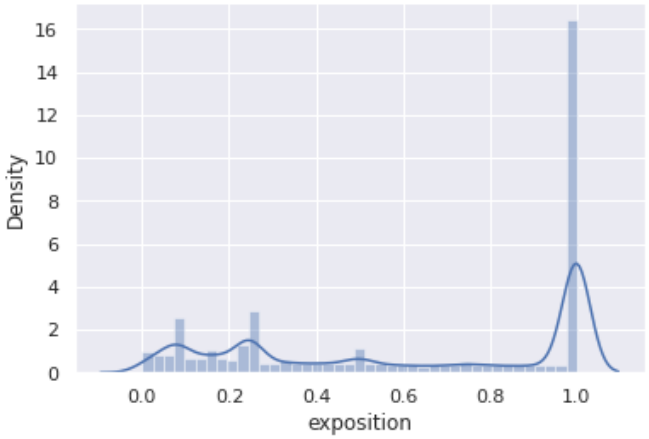


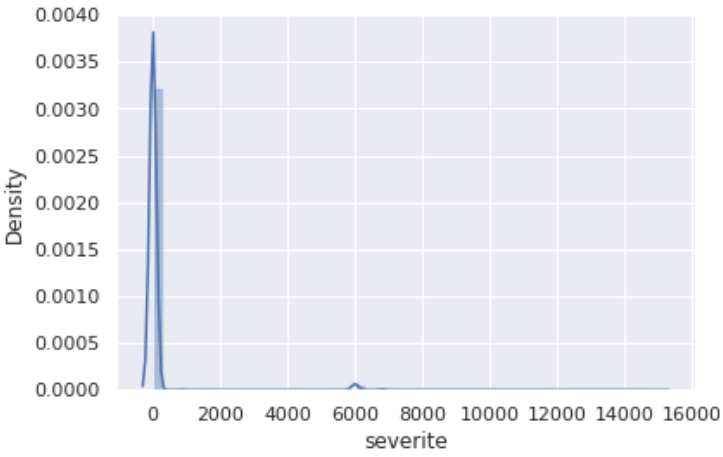


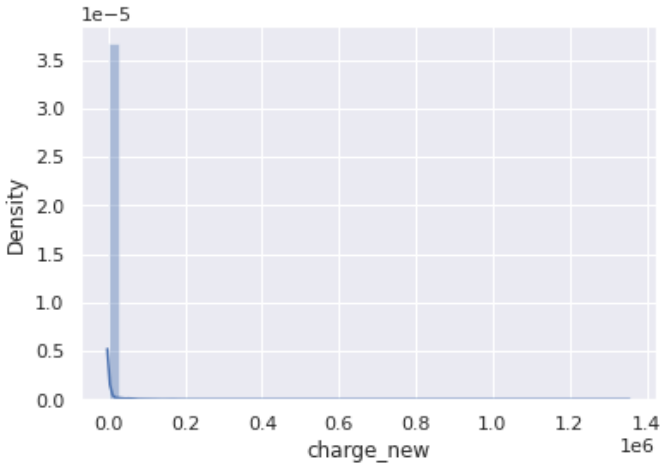

Density
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25
30
age_vehicule

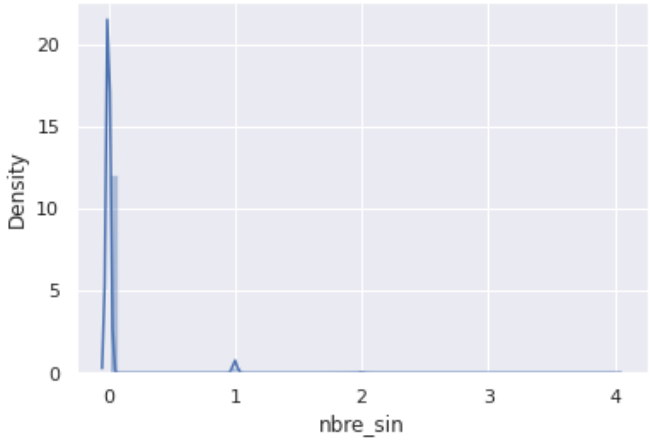

Density
0.5
0.4
0.3
0.2
0.1
0.0
0
20
40
60
80
frequence_sin


Density
0.010
0.008
0.006
0.004
0.002
0.000
0
200
400
600
800
1000
1200
duree_effet


Density
16
14
12
10
8
6
4
2
0
0.0
0.2
0.4
0.6
0.8
1.0
exposition


Density
0.0040
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
2000
4000
6000
8000
10000
12000
14000
16000
severite


1e−5
Density
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1e6
charge_new

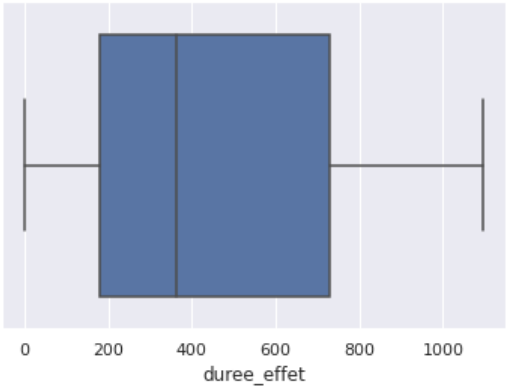


Remarquons que certaines features comme nombre de sinistres, charge new, severité et fréquence de sinistres contiennent un grand pourcentage de valeurs nuls. Cette data biaisée pourrait également nuire aux performances des modèles GLM et machine learning, sans oublier les inférences statistiques qui s'avèrent un challenge en ce qui concerne le fait de trouver une famille de loi dont la distribution est proche de la variable en question (sévérité).

La feature « Durée_effet »

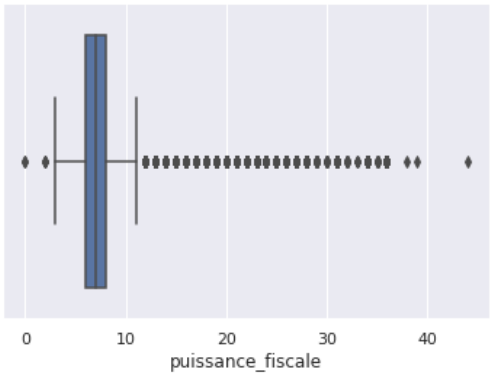
La durée d'effet représente la durée de contrat d'un client. Cette variable a une moyenne de 389.6 et une variance de 250.

```
count    233649.000000
mean        389.603807
std         250.330716
min           1.000000
25%         180.000000
50%         364.000000
75%         729.000000
max        1095.000000
Name: duree_effet, dtype: float64
```



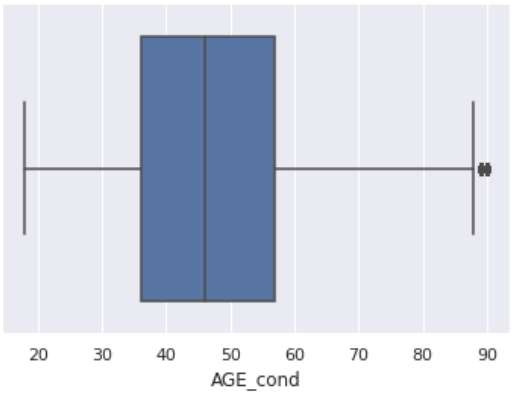
La feature "puissance fiscale"

```
puissance_fiscale
moyenne :  7.521808353555975
écart type :  1.9930148399231513
mode :  6.0
médiane :  7.0
max : 44.0
min : 0.0
```



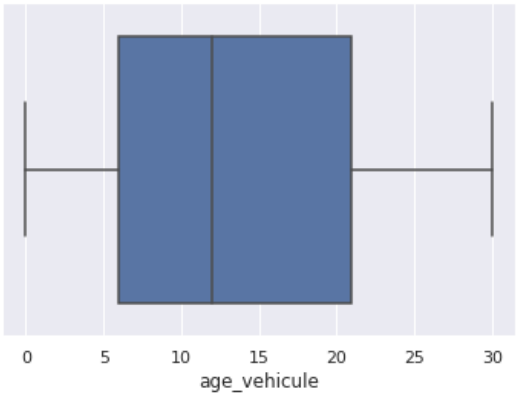
La feature “âge conducteur”

```
count    233649.000000
mean         47.199676
std          13.434573
min          18.000000
25%          37.000000
50%          46.000000
75%          56.000000
max          90.000000
Name: AGE_cond, dtype: float64
```



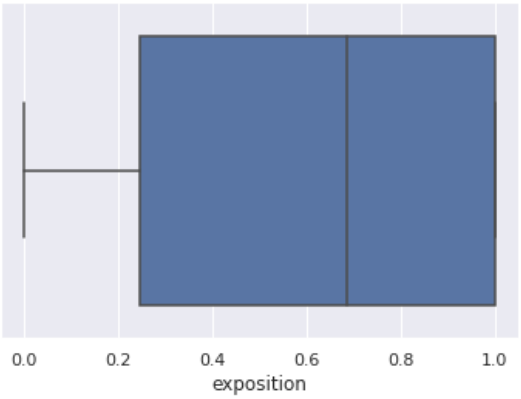
La feature “âge véhicule”

```
count    233649.000000
mean         12.321063
std           8.490791
min           0.000000
25%           5.000000
50%          10.000000
75%          19.000000
max          30.000000
Name: age_vehicule, dtype: float64
```



La feature “exposition”

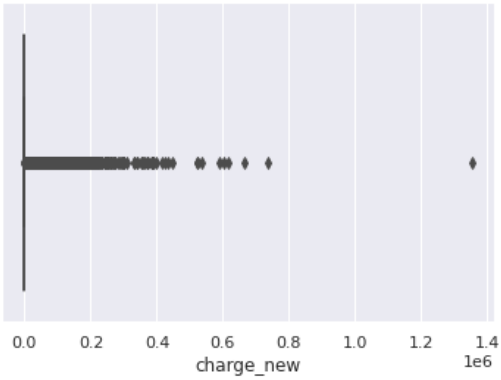
```
count    233649.000000
mean          0.614613
std           0.377609
min           0.000000
25%           0.246575
50%           0.687671
75%           1.000000
max           1.000000
Name: exposition, dtype: float64
```



La feature « charge new »

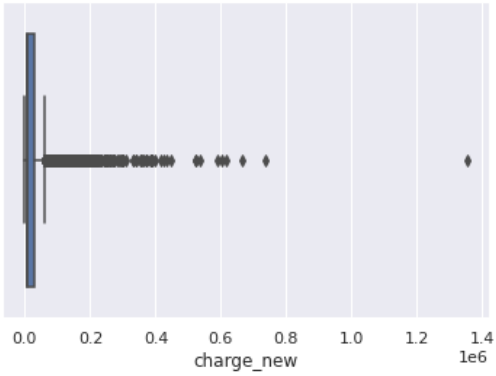
Description de l'intégralité de la distribution

```
count    2.336490e+05
mean     8.225767e+02
std      9.750926e+03
min      0.000000e+00
25%      0.000000e+00
50%      0.000000e+00
75%      0.000000e+00
max      1.353250e+06
Name: charge_new, dtype: float64
```



Description des valeurs non nuls

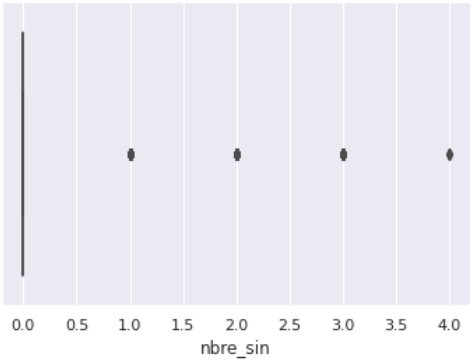
```
count    7.494000e+03
mean     2.564641e+04
std      4.825029e+04
min      1.000000e+00
25%      6.000000e+03
50%      6.000000e+03
75%      2.828646e+04
max      1.353250e+06
Name: charge_new, dtype: float64
```



La feature « nombre de sinistres »

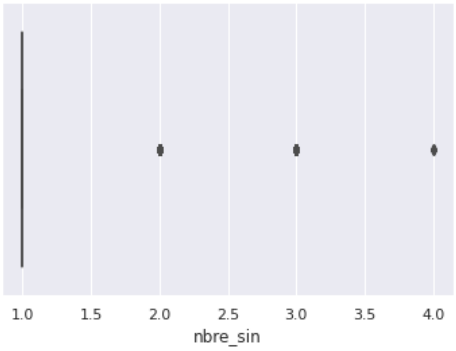
Description de l'intégralité de la distribution

```
count    233649.000000
mean          0.035802
std           0.196237
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max           4.000000
Name: nbre_sin, dtype: float64
```



Description des personnes ayant vécu au moins un sinistre

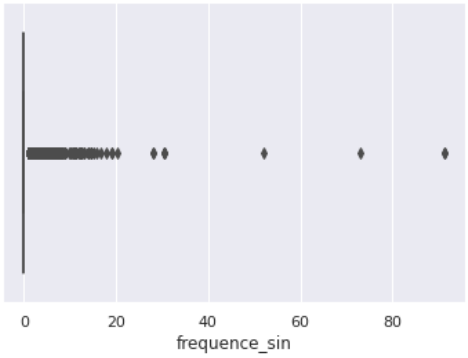
```
count    7939.000000
mean        1.053659
std         0.246707
min         1.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         4.000000
Name: nbre_sin, dtype: float64
```



Fréquence des sinistres

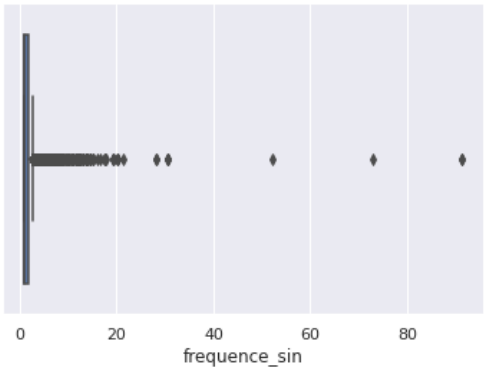
Description de toute la population

```
count    233649.000000
mean          0.058504
std           0.558680
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max          91.250000
Name: frequence_sin, dtype: float64
```



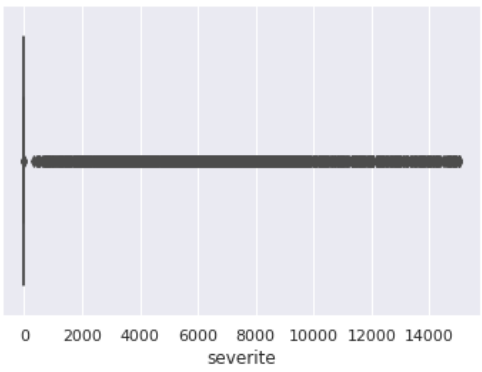
Description de la population des sinistrés

```
count    7939.000000
mean        1.721811
std         2.514520
min         1.000000
25%         1.000000
50%         1.002747
75%         1.644144
max        91.250000
Name: frequence_sin, dtype: float64
```



Sévérité

```
count    1.687420e+05
mean     1.099340e+03
std      1.124390e+04
min      0.000000e+00
25%      0.000000e+00
50%      0.000000e+00
75%      0.000000e+00
max      1.353250e+06
Name: severite, dtype: float64
```

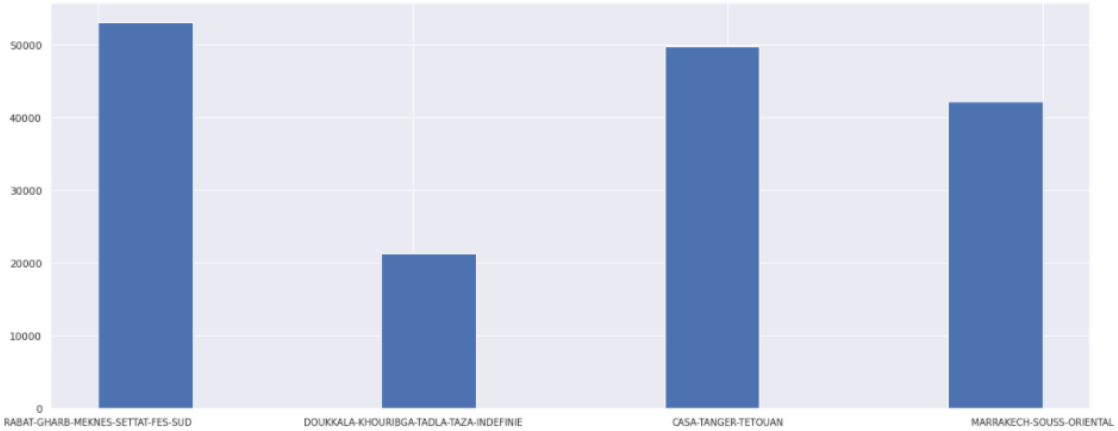


La feature « Type »

Remarquons que les clients ayant une voiture de type Gasoil sont supérieurs à ceux ayant une voiture de type Essence. Ceci peut être dû au prix du gasoil 1,454 €/l TTC le **litre** inférieur au prix de l'essence 1,607 €/l TTC. Aussi à cause de l'abondance des voitures type gasoil dans le marché par rapport aux voitures type essence.

La feature "Region2"

Ci-dessous on peut avoir une idée sur la répartition de la population des assurés dans les différentes régions.



Exploration bivariée des données

Correlations

Le tableau ci-dessous représente les facteurs de corrélations entre les différentes features. Cette matrice offre un aperçu sur les variables indépendantes significatives lors de la phase de modélisation.

	puissance_fiscale	AGE_cond	age_vehicule	duree_effet	frequence_sin	severite
puissance_fiscale	1.000000	0.083848	-0.032012	-0.006924	-0.005678	-0.014876
AGE_cond	0.083848	1.000000	-0.078558	0.173181	-0.010850	-0.002053
age_vehicule	-0.032012	-0.078558	1.000000	-0.108748	-0.043452	-0.091632
duree_effet	-0.006924	0.173181	-0.108748	1.000000	0.011835	0.096446
frequence_sin	-0.005678	-0.010850	-0.043452	0.011835	1.000000	0.467188
severite	-0.014876	-0.002053	-0.091632	0.096446	0.467188	1.000000

Répartition des types de véhicules sur les régions du royaume

Region2	CASA-TANGER-TETOUAN	DOUKKALA-KHOURIBGA-TADLA-TAZA-INDEFINIE	MARRAKECH-SOUSS-ORIENTAL	RABAT-GHARB-MEKNES-SETTAT-FES-SUD
Type				
E	0.341554	0.086469	0.269095	0.302882
G	0.281523	0.145451	0.247042	0.325983

Moyennes de variables numériques par type de véhicule

	puissance_fiscale	AGE_cond	age_vehicule	duree_effet	frequence_sin	severite
Type						
E	7.852307	45.107553	13.883663	370.041807	0.046270	153.699382
G	7.671976	48.172191	13.327792	361.755221	0.055092	195.540716

Statistiques descriptives par type de véhicule

Sévérité

	mean	std	median	amin	amax
Type					
E	153.699382	1025.138708	0.0	0.0	15000.0
G	195.540716	1152.101351	0.0	0.0	15000.0

Fréquence des sinistres

Type	mean	std	median	amin	amax
E	0.046270	0.441308	0.0	0.0	30.416667
G	0.055092	0.624704	0.0	0.0	91.250000

Statistiques descriptives par région

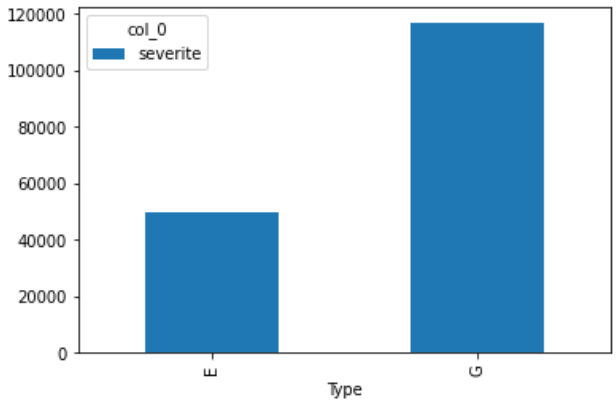
Sévérité

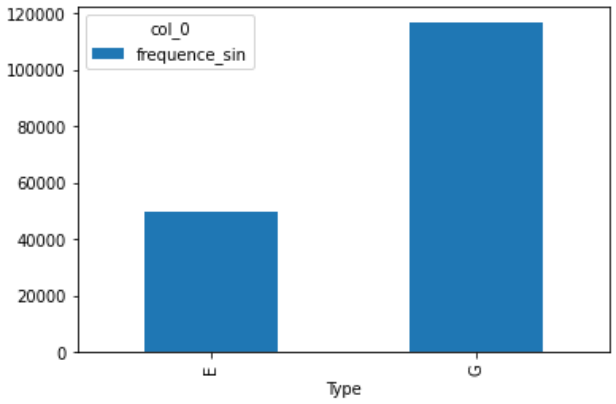
Region2	mean	std	median	amin	amax
CASA-TANGER-TETOUAN	305.102913	1415.606625	0.0	0.0	15000.0
DOUKKALA-KHOURIBGA-TADLA-TAZA-INDEFINIE	72.332654	693.637284	0.0	0.0	15000.0
MARRAKECH-SOUSS-ORIENTAL	85.715109	762.015268	0.0	0.0	15000.0
RABAT-GHARB-MEKNES-SETTAT-FES-SUD	190.157429	1157.795611	0.0	0.0	15000.0

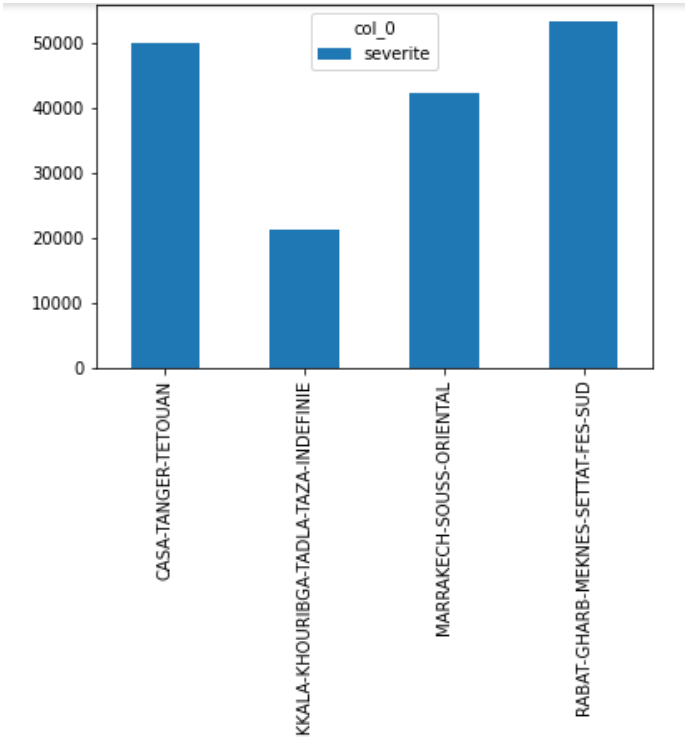
Fréquence des sinistres

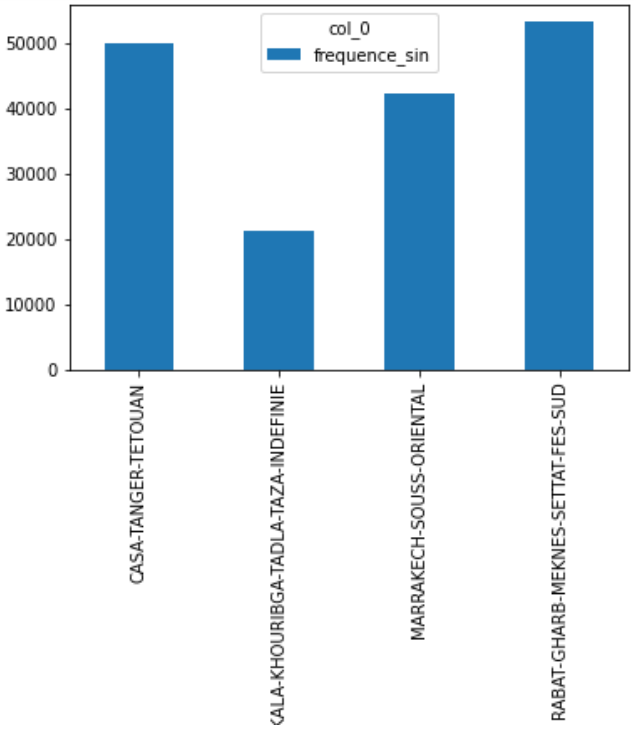
Region2	mean	std	median	amin	amax
CASA-TANGER-TETOUAN	0.084199	0.604590	0.0	0.0	73.000000
DOUKKALA-KHOURIBGA-TADLA-TAZA-INDEFINIE	0.021785	0.362291	0.0	0.0	30.416667
MARRAKECH-SOUSS-ORIENTAL	0.028982	0.575917	0.0	0.0	91.250000
RABAT-GHARB-MEKNES-SETTAT-FES-SUD	0.053602	0.615527	0.0	0.0	91.250000

Barplots


col_0
severite
120000
100000
80000
60000
40000
20000
0
E
G
Type


col_0
frequence_sin
120000
100000
80000
60000
40000
20000
0
E
G
Type


col_0
severite
50000
40000
30000
20000
10000
0
CASA-TANGER-TETOUAN
KKALA-KHOURIBGA-TADLA-TAZA-INDEFINIE
MARRAKECH-SOUSS-ORIENTAL
RABAT-GHARB-MEKNES-SETTAT-FES-SUD


col_0
frequence_sin
50000
40000
30000
20000
10000
0
CASA-TANGER-TETOUAN
CALA-KHOURIBGA-TADLA-TAZA-INDEFINIE
MARRAKECH-SOUSS-ORIENTAL
RABAT-GHARB-MEKNES-SETTAT-FES-SUD

Statistiques descriptives par tranche d'âge du véhicule

Sévérité

age_vehicule_label	mean	std	median	amin	amax
[0,10[	301.529539	1424.467944	0.0	0.0	15000.0
[10,20[	126.120511	920.285476	0.0	0.0	15000.0
[20,30]	76.470739	727.410508	0.0	0.0	15000.0

Fréquence des sinistres

age_vehicule_label	mean	std	median	amin	amax
[0,10[	0.078555	0.515264	0.0	0.0	52.142857
[10,20[	0.043115	0.775773	0.0	0.0	91.250000
[20,30]	0.025375	0.347718	0.0	0.0	28.076923

Moyennes

age_vehicule_label	puissance_fiscale	AGE_cond	age_vehicule	duree_effet	frequence_sin	severite
[0,10[	7.696040	48.039193	5.148229	395.429294	0.078555	301.529539
[10,20[	7.960970	47.931902	14.086272	352.420410	0.043115	126.120511
[20,30]	7.503658	45.364132	24.853309	332.603025	0.025375	76.470739

Statistiques descriptives par tranche de puissance fiscale

Sévérité

puissance_fiscale_label	mean	std	median	amin	amax
[0,10[	188.125364	1130.995405	0.0	0.0	15000.00
[10,20[	152.895539	1020.864767	0.0	0.0	14622.58
[20,30[	145.257257	962.139314	0.0	0.0	9789.00
[30,44]	208.763577	1382.038224	0.0	0.0	12873.92

Fréquence des sinistres

puissance_fiscale_label	mean	std	median	amin	amax
[0,10[	0.053356	0.559145	0.0	0.0	91.250000
[10,20[	0.047712	0.679245	0.0	0.0	73.000000
[20,30[	0.031965	0.229466	0.0	0.0	3.613861
[30,44]	0.039530	0.226805	0.0	0.0	1.862245

Moyennes

puissance_fiscale_label	puissance_fiscale	AGE_cond	age_vehicule	duree_effet	frequence_sin	severite
[0,10[	7.090095	46.937281	13.324002	364.679434	0.053356	188.125364
[10,20[	10.960642	49.137321	14.777518	358.888543	0.047712	152.895539
[20,30[	22.741317	49.132934	7.992814	425.009581	0.031965	145.257257
[30,44]	33.390244	49.471545	6.260163	439.577236	0.039530	208.763577

Statistiques descriptives par tranche d'âge du conducteur

Sévérité

age_cond_label	mean	std	median	amin	amax
[18,30[	176.526854	1096.873739	0.0	0.0	15000.0
[30,50[	180.963366	1116.604578	0.0	0.0	15000.0
[50,70[	200.888472	1153.590949	0.0	0.0	15000.0
[70,90]	123.627513	945.292349	0.0	0.0	15000.0

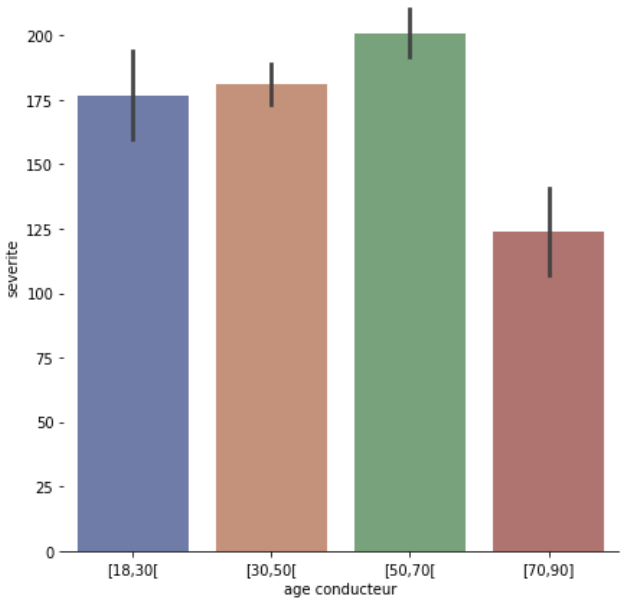
Fréquence des sinistres

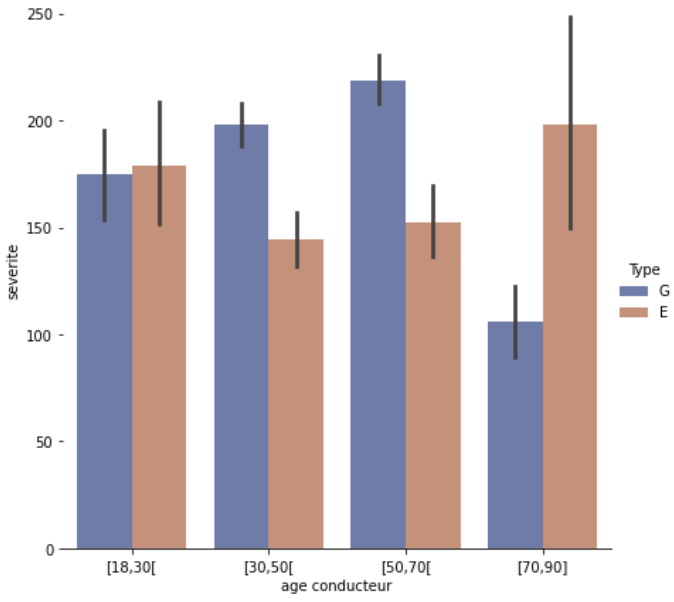
age_cond_label	mean	std	median	amin	amax
[18,30[	0.071535	0.920906	0.0	0.0	91.250000
[30,50[	0.049821	0.473478	0.0	0.0	73.000000
[50,70[	0.056303	0.625795	0.0	0.0	91.250000
[70,90]	0.026772	0.271373	0.0	0.0	19.210526

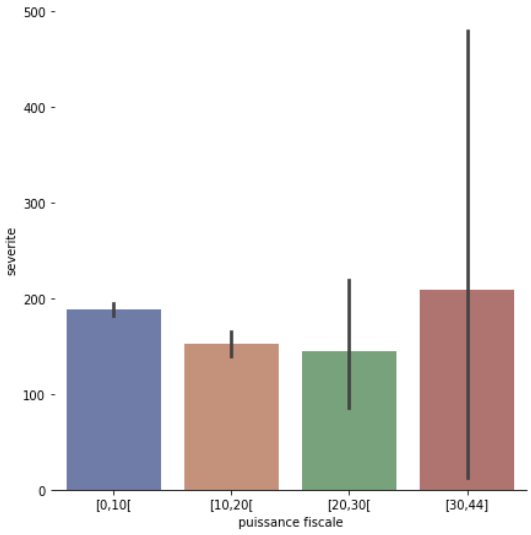
Moyennes

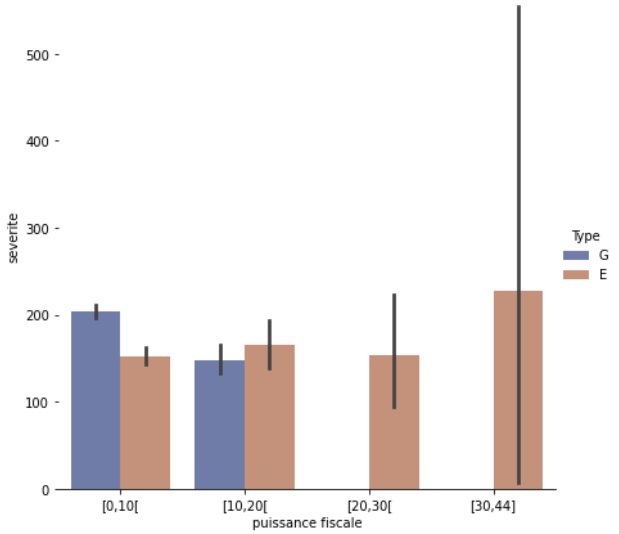
age_cond_label	puissance_fiscale	AGE_cond	age_vehicule	duree_effet	frequence_sin	severite
[18,30[	7.466659	25.968269	14.093976	264.084980	0.071535	176.526854
[30,50[	7.624456	39.592560	14.016617	346.800406	0.049821	180.963366
[50,70[	7.872411	58.198577	12.771344	409.599482	0.056303	200.888472
[70,90]	8.057374	75.120847	12.610496	402.607206	0.026772	123.627513

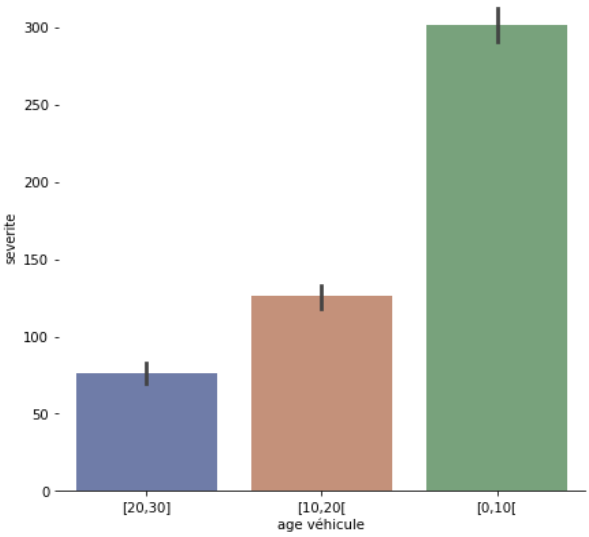
Catplots

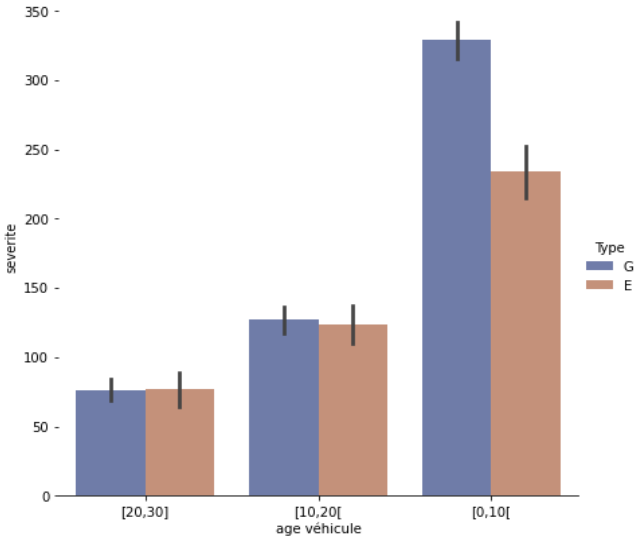




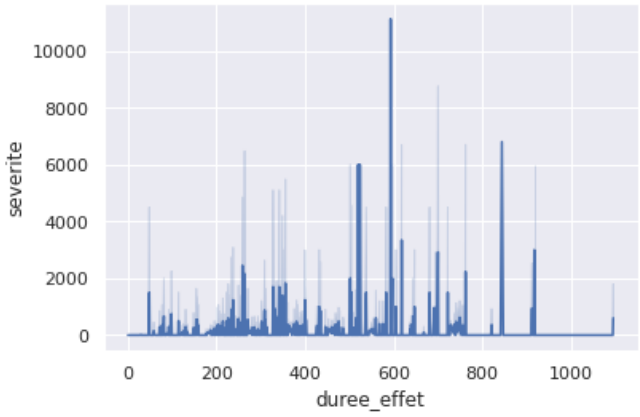


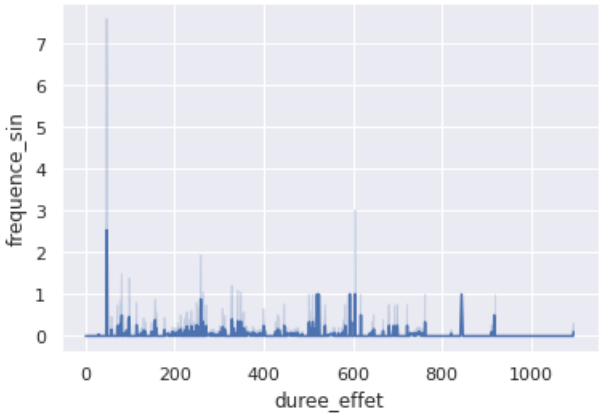






Durée effet – sévérité – fréquence des sinistres





IV. Tests

Loi d'évolution de la sévérité

Modéliser notre variable à prédire nous impose de déterminer au préalable la loi d'évolution théorique qui se rapproche le plus des observations. Pour ce faire, nous avons procédé à une inférence statistique sur des lois de probabilité continues, étant donné que la sévérité est une variable continue. Entre autres, normale, beta, gamma, Pareto, log-normale, chi, chi2, Cauchy, exponnorm, lomax, exponentielle, etc.

L'inférence statistique

L'inférence statistique est l'ensemble des techniques permettant d'induire les caractéristiques d'un groupe général (la population) à partir de celles d'un groupe particulier (l'échantillon), en fournissant une mesure de la certitude de la prédiction : la probabilité d'erreur. C'est donc un ensemble de méthodes permettant de tirer des conclusions fiables à partir de données d'échantillons statistiques.

Dans notre cas, il a s'agit de comparer le niveau d'adéquation des lois de comportement plus haut citées, avec nos observations en sévérité, les unes par rapport aux autres. La statistique discriminatoire choisie est la racine de l'erreur quadratique moyenne, en anglais « root mean squared error » (RMSE).

RMSE

On appelle Erreur Quadratique Moyenne (MSE pour Mean Squared Error en anglais) d'un estimateur $\hat{\theta}$ par rapport à un paramètre estimé θ, l'espérance $E((\hat{\theta} - \theta)^2)$. C'est la moyenne arithmétique des carrés des écarts entre prévisions du modèle et observations. La RMSE quant à elle est égale à la racine carrée de la MSE.

$$RMSE = \sqrt{MSE} = \sqrt{E((\hat{\theta} - \theta)^2)}$$

Etant donné n observations $(y_1, \dots, y_n)$ prédites par $(\hat{y}_1, \dots, \hat{y}_n)$, une formule généralement utilisée pour calculer la RMSE est :

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Quelques lois de probabilité

Loi normale

Encore appelée loi gaussienne, loi de Gauss ou lois de Laplace-Gauss, la loi normale est une loi de probabilité absolument continue dépendante de deux paramètres : son espérance, notée μ, et son écart type σ. Sa densité de probabilité est donnée par :

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

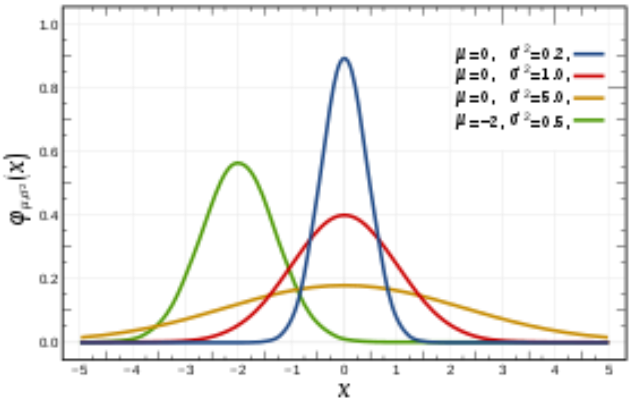


Figure 1: densité de probabilité de la loi normale

Loi gamma

Une loi Gamma est un type de loi de probabilité de variables aléatoires réelles positives. Elle est caractérisée par deux paramètres qui affectent respectivement la forme et l'échelle de sa représentation graphique.

On dit qu'une variable aléatoire X suit une loi gamma de paramètres $k > 0$ et $\lambda > 0$, notée $\Gamma(k, \lambda)$, si elle admet pour densité :

$$f(x) = \frac{\lambda}{\Gamma(k)} e^{-\lambda x} (\lambda x)^{k-1} \mathbf{1}_{\mathbb{R}_+}(x)$$

Avec Γ(k) la valeur de la fonction gamma d'Euler en k.

$E(X) = \frac{k}{\lambda}$ et $V(X) = \frac{k}{\lambda^2}$.

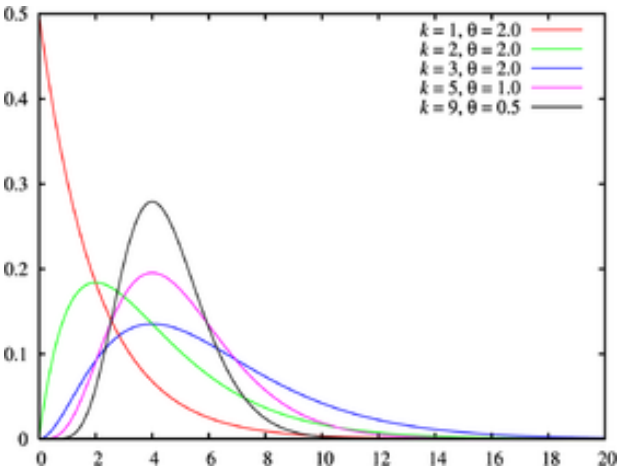


Figure 2 : Densité de probabilité de la loi gamma, $\theta = 1/\lambda$

Ses utilisations sont multiples, mais en particulier elle est très sollicitée pour modéliser les durées de vie.

Loi exponentielle

On dit que X suit la loi exponentielle de paramètre λ, si elle est absolument continue, et admet pour densité :

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{si } x > 0 \\ 0 & \text{sinon.} \end{cases}$$

$$E(X) = \frac{1}{a} \text{ et } V(X) = \frac{1}{a^2}$$

La loi exponentielle est (également) utilisée pour modéliser la durée de vie d'un phénomène sans mémoire, ou sans vieillissement, ou sans usure : la probabilité que le phénomène dure au moins s + t heures sachant qu'il a déjà duré t heures sera la même que la probabilité de durer s heures à partir de sa mise en fonction initiale.

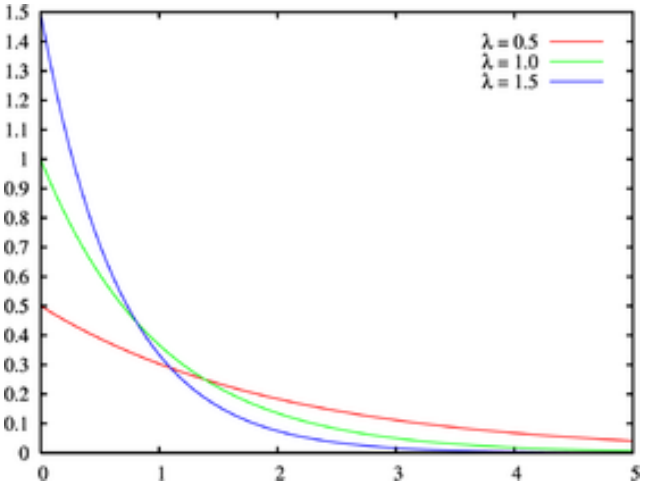


Figure 3 : densité d'une loi exponentielle

Loi de khi 2

La loi du χ^2 centrée (prononcé « khi carré » ou « khi-deux ») avec k degrés de liberté peut être perçue comme la loi de la somme de carrés de k lois normales centrées réduites indépendantes (lois normales d'espérance nulle et de variance 1). Plus généralement, On dit qu'une variable aléatoire X suit une loi de Khi 2 à k degrés de liberté a pour fonction de densité :

$$f_X(x;k) = \frac{1}{2^{\frac{k}{2}}\Gamma\left(\frac{k}{2}\right)} x^{\frac{h}{2}-1} e^{-\frac{x}{2}} \text{ , pour tout } x \text{ positif}$$

$$E(X) = \text{k} \text{ et } V(X) = 2\text{k}$$

Avec Γ la fonction gamma d'Euler.

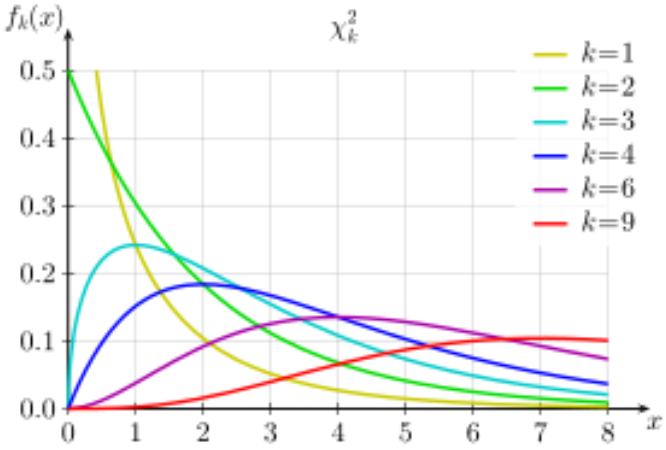


Figure 4 : densité de probabilité de la loi du Khi 2

La loi du χ^2 est généralement utilisée en inférence statistique et pour les tests statistiques notamment le test du χ^2 qui porte son nom.

Loi lomax

C'est un cas particulier de la loi de Pareto. Une variable aléatoire X continue suit une loi de Pareto de paramètres $k > 0$ et x_m si sa densité est donnée par :

$$f(x; k, x_{\mathrm{m}}) = k\frac{x_{\mathrm{m}}^k}{x^{k+1}} \quad \text{pour } x \geq x_{\mathrm{m}}.$$

$\mathbb{E}(X) = \frac{kx_{\mathrm{m}}}{k-1}$ (Espérance infinie pour k<=1) ;

$\mathrm{Var}\,(X) = \left(\frac{x_{\mathrm{m}}}{k-1}\right)^2 \frac{k}{k-2}$ (Variance infinie pour k<=2)

Dans ce cas, on dira que la variable aléatoire $X + 1$ avec $x_{\mathrm{m}} = 1$ suit une loi de Lomax.

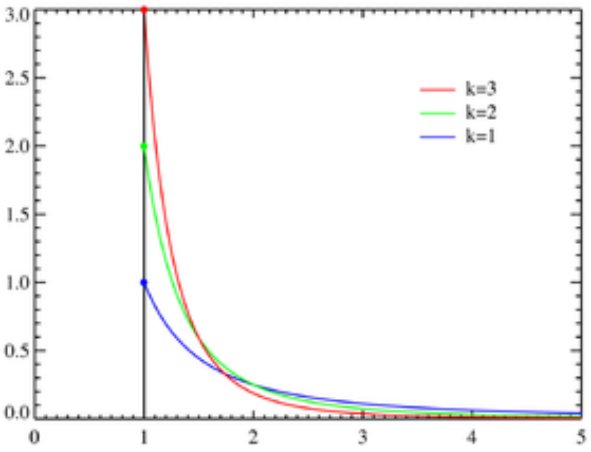


Figure 5 : Densité de probabilité de la loi de pareto pour Xm = 1

La loi de Pareto est à la base du principe des 80/20 : « agir sur 20 % de causes permet de résoudre 80 % du problème ». Le Pareto est utile pour identifier sur quelle cause agir en priorité pour améliorer de façon significative la situation et est grandement utilisée en réassurance (assurance des entreprises d'assurance).

Implémentation sous python

Description

La fonction « histogram » du module Numpy sous Python, permet d'obtenir l'histogramme d'un vecteur de données.

```
y, x = np.histogram(df.severite, bins='auto', density=True)
```

- np : module Numpy
- df.severite : La colonne sévérité est extraite de notre base de données nommée ''df''.
- bins = 'auto' : Si bins est un entier, il définit le nombre de bins de même largeur dans la plage donnée. Si bins est une chaine de caractère (notre cas), il permet selon la méthode entrée, de calculer la largeur optimale du pas de sévérité tracée en abscisses et par conséquent le nombre de bacs à partir des données comprises dans la plage demandée. La valeur 'auto' permet de choisir comme valeur de bins, le Maximum des estimateurs « Sturges » et « Freedman Diaconis ». Si n est le nombre d'observations, alors :

i. **Estimateur de Sturges** : $n_h = log_2(n) + 1$

Cet estimateur suppose la normalité des données et est trop conservatif pour des ensembles de données plus grands (point de basculement, n=1000) et non normaux.

ii. **Estimateur de Friedman Diaconis** : $h = 2\frac{IQR}{n^{\frac{1}{3}}}$

La largeur de la zone est proportionnelle à l'intervalle interquartile (IQR) et inversement proportionnelle à la racine cubique de n. Cet estimateur est souvent moins adapté pour les petits ensembles de données, mais c'est assez bon pour les grands ensembles de données. L'IQR est très robuste aux valeurs aberrantes.

Ayant donc obtenue la distribution réelle de la sévérité, nous déterminons le modèle théorique qui la décrit le mieux. Pour ce faire, le module « Scipy » est doté de la définition paramétrique de la plupart des distributions statistiques usuelles. Pour chacun de ces modèles, la fonction « fit » permet de déterminer les paramètres donnant la meilleure estimation de la sévérité si celle-ci suivant la loi de distribution dictée par le dit modèle. La qualité de ces paramètres et des modèles est mesurée à partir de la somme du carré des erreurs (Sum Squared Errors, SSE).

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

```
# listes des distributions à tester
dist_names = ['norm', 'beta','gamma', 'pareto', 't', 'loggamma', 'chi', 'c
hi2', 'cauchy', 'exponnorm', 'lomax', 'expon', 'laplace', 'gausshyper']
```

```
# optimisation des paramètres
dist = getattr(scipy.stats, name)
param = dist.fit(df.severite)

# Paramètres
loc = param[-2]
scale = param[-1]
arg = param[:-2]

# Distribution estimée
pdf = dist.pdf(x, *arg, loc=loc, scale=scale)

# SSE
model_sse = np.sum((y - pdf)**2)
```

Résultats

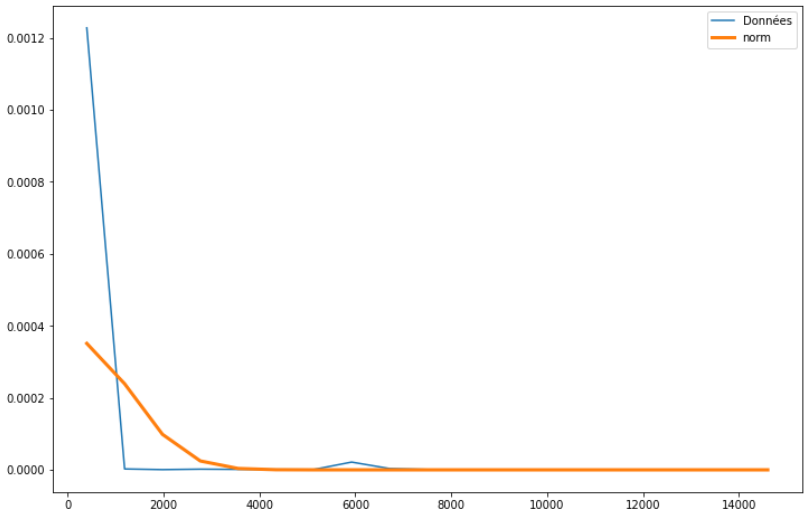
Distribution	Normale	Beta	Gamma	Pareto	T	Log-gamma	Chi	Chi 2	Cauchy
RMSE (*E-06)	2.24	2.97	2.43	2.97	3.01	2.28	1.75	1.92	3.01

Distribution	Exponentielle	exponnorm	Lomax	Laplace	Gausshyper
RMSE (*E-06)	1.46	1.46	2.92	2.23	2.39

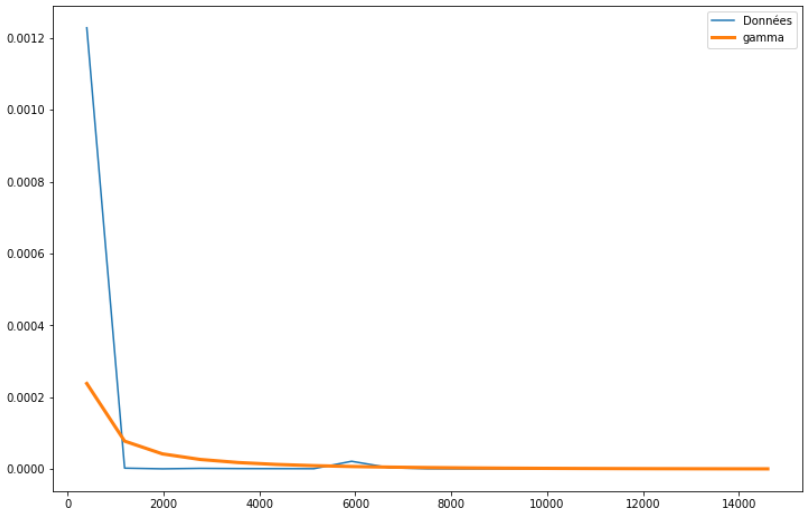
Le meilleur modèle correspond à une loi exponentielle de paramètre 183.03.

Etant donnée que nous ferons une modélisation par GLM, nous travaillerons dans la suite avec les lois normale, gamma et exponentielle (toutes les 03 sont de la famille exponentielle et ont donné parmi les meilleures valeurs de SSE).

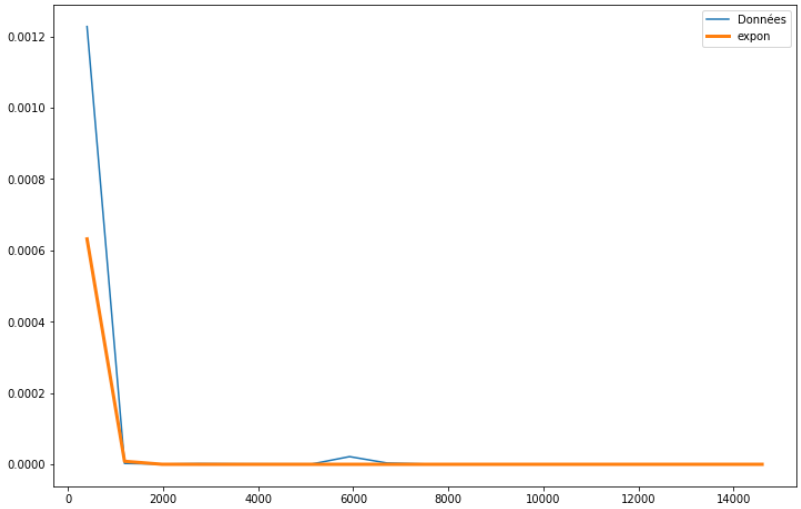
Loi normale N(μ, σ): (μ , σ) = (183.03 , 1115.81)



Loi gamma $\Gamma(k, \lambda)$:



Loi exponentielle E (λ) : $\lambda = 183.03$



Test de normalité : Test de D'Agostino-Pearson

Description

Une loi normale est caractérisée par un coefficient d'asymétrie et un coefficient d'aplatissement nuls. Ces deux indicateurs permettent donc d'apprécier de manière approximative la normalité d'une distribution.

Considérons n observations $(x_i)_{1\leq i\leq n}$.

Le coefficient d'asymétrie G_1 (skewness en anglais) est défini par :

$$G_1 = \frac{n}{(n-1)(n-2)} \sum_i \left(\frac{x_i - \bar{x}}{s}\right)^3$$

Le coefficient d'aplatissement G_2 (Kurtosis en anglais) :

$$G_2 = \frac{n(n+1)}{(n-1)(n-2)(n-3)} \sum_i \left(\frac{x_i - \bar{x}}{s}\right)^4 - \frac{3(n-1)^2}{(n-2)(n-3)}$$

$s = \sqrt{\frac{1}{n-1}\sum_i (x_i - \bar{x})^2}$ est l'écart-type de la variable x.

Pour des valeurs de G_1 et G_2 suffisamment proche de zéro, l'hypothèse de normalité de la distribution ne peut être rejetée. Il faut donc quantifier leur degré de proximité à zéro, et donc connaitre leur loi d'évolution afin de construire une statistique permettant de faire un test.

Le test de D'Agostino dit test du K carré de D'Agostino-Pearson est basé sur ces coefficient d'asymétrie et d'aplatissement. Il est d'une grande puissance pour les échantillons à grand effectif (comme notre cas). Il construit une statistique reliant les deux coefficients à travers les transformations suivantes :

Transformation du coefficient d'asymétrie :

$$G_1 = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^3}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2\right)^{3/2}}$$

$$A = G_1\sqrt{\frac{(n+1)(n+3)}{6(n-2)}}$$

$$B = \frac{3(n^2+27n-70)(n+1)(n+3)}{(n-2)(n+5)(n+7)(n+9)}$$

$$C = \sqrt{2(B-1)} - 1$$

$$D = \sqrt{C}$$

$$E = \frac{1}{\sqrt{\ln(D)}}$$

$$F = \frac{A}{\sqrt{\frac{2}{C-1}}}$$

$$\boldsymbol{z_1 = E\ln\left(F + \sqrt{F^2+1}\right)}$$

Transformation du coefficient d'aplatissement :

$$g_2 = G_2 = \frac{n(n+1)}{(n-1)(n-2)(n-3)}\sum_i\left(\frac{x_i - \bar{x}}{s}\right)^4 - \frac{3(n-1)^2}{(n-2)(n-3)}$$

$$G = \frac{24n(n-2)(n-3)}{(n+1)^2(n+3)(n+5)}$$

$$H = \frac{(n-2)(n-3)g_2}{(n+1)(n-1)\sqrt{G}}$$

$$J = \frac{6(n^2-5n+2)}{(n+7)(n+9)}\sqrt{\frac{6(n+3)(n+5)}{n(n-2)(n-3)}}$$

$$K = 6 + \frac{8}{J}\left[\frac{2}{J} + \sqrt{1+\frac{4}{J^2}}\right]$$

$$L = \left(\frac{1-\frac{2}{K}}{1+H\sqrt{\frac{2}{K-4}}}\right)$$

$$z_2 = \frac{\left(1 - \frac{2}{9K}\right) - L^{\frac{1}{3}}}{\sqrt{\frac{2}{9K}}}$$

z_1 et z_2 suivent tous deux asymptotiquement une loi normale centrée réduite. La statistique du test est donc :

$$K_2 = z_1^2 + z_2^2$$

Elle suit une loi du Khi 2 à 2 degrés de liberté. L'inadéquation de notre distribution avec une loi normale est d'autant plus marquée que K_2 est grand.

Pour un niveau de risque α, la région de critique est donnée par :

$$RC : K_2 > \chi^2_{1-\alpha}(2)$$

Implémentation

La fonction « normaltest » du module Scipy implémente le test de D'agostino-Pearson.

```
stat.normaltest(df.severite)
>>> NormaltestResult(statistic=202345.2583067654, pvalue=0.0)
```

K_2 vaut alors 202 345.25 pour notre variable sérérité, avec une valeur p nulle. Donc l'hypothèse nulle est rejetée.

Test d' « exponentialité »: Test d' Anderson-Darling

Description

Le test d'Anderson-Darling est une variante du test de Kolmogorov-Smirnov à la différence qu'il donne plus d'importance à la queue des distributions. L'hypothèse nulle pour ce test est que les données suivent une loi de distribution spécifiée (gaussienne, exponentielle, logistique, etc.). Dans notre cas, il s'agit de tester l'hypothèse d'une loi exponentielle.

La statistique de ce test est :

$A = -n - \frac{1}{n}\sum_{i=1}^{n} (2i-1)[\ln(F_i) + \ln(1 - F_{n-i+1})]$

Avec F_i est la fréquence théorique de la loi de répartition normale centrée et réduite associée à la valeur standardisée $z_i = \frac{x_i - \bar{x}}{s}$.

Les valeurs critiques du test d'Anderson-Darling dépendent de la loi considérée en hypothèse nulle. Pour un niveau de risque donné, l'hypothèse nulle est rejetée lorsque la valeur de la statistique A est supérieure à la valeur critique correspondante.

Implémentation

La fonction « anderson » du module Scipy implémente le test d'Anderson-Darling. Nous lui passons en paramètre le vecteur de données (df.severite) et la distribution servant d'hypothèse nulle (''expon'' pour ''exponentielle'').

```
stat.anderson(df.severite, dist='expon')

>>> AndersonResult(statistic=inf, critical_values=array([0.922, 1.078, 1.341,
1.606, 1.957]), significance_level=array([15. , 10. ,  5. ,  2.5,  1. ]))
```

Niveau de risque (%)	15	10	5	2.5	1
Valeur critique	0.922	1.078	1.341	1.606	1.957

On obtient une valeur de A « infinie ». De ce fait, même avec un niveau de risque très faible, l'hypothèse nulle est rejetée.

* Effectuons un test de normalité d'Anderson-Darling.

```
stat.anderson(df.severite, dist='norm')
>>> AndersonResult(statistic=60809.69814970868, critical_values=array([0.576,
0.656, 0.787, 0.918, 1.092]), significance_level=array([15. , 10. ,  5. ,
2.5,  1. ]))
```

Niveau de risque (%)	15	10	5	2.5	1
Valeur critique	0.576	0.656	0.787	0.918	1.092

La statistique A dans le cas d'un test de normalité vaut 60 809.69, très supérieure à toutes les valeurs critiques, même pour le risque 1%. Ce qui confirme le résultat du test de D'Agostino-Pearson, donc le rejet de l'hypothèse de normalité de notre variable à estimer.

V. Modélisation de la sévérité : Modèles Linéaires Généralisés (GLM)

Introduction aux GLM

Le modèle linéaire généralisé est une extension du modèle linéaire. Introduit par John Nelder et Robert Wedderburn en 1972 il permet d'étudier le lien entre une variable qualitative à expliquer et un ensemble de variables explicatives qualitatives ou quantitatives. Il englobe : modèle linéaire

général (régression multiple, analyse de la variance et analyse de la covariance), modèle log-linéaire, régression logistique, régression de Poisson.

Définition mathématique d'un GLM

Les modèles GLM sont constitués de trois composantes : la distribution, le prédicateur linéaire et une fonction lien.

La distribution

C'est une composante dite aléatoire, qui identifie la distribution de probabilités de la variable à expliquer.

On suppose que l'échantillon statistique est constitué de n variables aléatoires {Yi ; i = 1, …, n} indépendantes admettant toutes des distributions issues d'une famille exponentielle. En d'autres termes, les lois de ces variables sont dominées par une même mesure dite de référence et la famille de leurs densités par rapport à cette mesure se met sous la forme :

$$f(y_i;\theta_i,\phi) = \exp\left\{\frac{y_i\theta_i - v(\theta_i)}{u(\phi)} + w(y_i,\phi)\right\} \quad (1)$$

La plupart des lois usuelles comportant un ou deux paramètres se prêtent à cette formulation : gaussienne, gaussienne inverse, gamma, Poisson, binomiale, etc. Le paramètre θ_i est appelé paramètre naturel de la famille exponentielle.

Pour certaines lois, u est de la forme :

$$u(\phi) = \frac{\phi}{\omega_i}$$

Les poids ω_i étant les poids connus des observations, fixés ici à 1 pour simplifier ; φ est appelé alors paramètre de dispersion.

L'expression (1) peut alors être transformée en :

$$f(y_i,\theta_i) = a(\theta_i)\, b(y_i) \exp\{\, y_i\, Q(\theta_i)\,\}$$

Avec
$$Q(\theta) = \frac{\theta}{\phi}$$
$$a(\theta) = \exp\left\{-\frac{v(\theta)}{\phi}\right\},$$
$$b(y) = \exp\{w(y,\phi)\}$$

Le prédicateur linéaire

Soit X la matrice contenant les observations des variables explicatives (matrice de planification d'expérience). Supposons p variables explicatives pour prédire la grandeur Y.

$X = (\mathrm{X}_1 \cdots \mathrm{X}_p)$ où $\mathrm{X}_i \in R^n \forall i \in \{1 \ldots n\}$

X est donc une matrice $n * p$
La i -ème ligne de X correspond aux valeurs des variables explicatives pour l'observation i.
La j -ème colonne de X correspond aux valeurs de la variable explicative j.

β étant un vecteur de p paramètres, on appelle prédicateur linéaire, la composante dite déterministe η, donnée par :

$$\eta = X * \beta$$

La fonction lien

La troisième composante du GLM est une fonction qui met en relation les deux premières composantes (prédicateur linéaire et composante aléatoire). Soit $\{\mu_i = E(Y_i); i = 1, \dots, n\}$. On pose :

$$\eta_i = g(\mu_i)\ i = 1, \dots, n$$

g est appelée fonction lien, supposée monotone et différentiable. Ainsi, la valeur de la moyenne de la variable cible appartient au sous-espace engendré par les variables explicatives. C'est-à-dire :

$g(\mu_i) = \mathbf{x}_i'\beta \qquad i = 1, \dots, n$

Lorsque la fonction lien assure la liaison entre la moyenne μ_i et le paramètre naturelle θ_i, elle est dite canonique.

$$g(\mu_i) = \theta_i = x_i'\beta$$

Loi	*Nom du lien*	*Fonction lien*
Binomiale	Logit	$g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$
Poisson	Log	$g(\mu) = \log(\mu)$
Normale	Identité	$g(\mu) = \mu$
Gamma	Réciproque	$g(\mu) = -\frac{1}{\mu}$

Figure 6 : Récapitulatif des fonctions de lien canoniques des modèles usuels

La fonction lien n'est pas nécessairement canonique, donc doit faire l'objet d'un choix motivé.

Estimation des coefficients

Les coefficients β_i sont estimés par maximisation de la fonction de log-vraisemblance du GLM.

Notons $\ell(\theta_i, \phi; y_i)$ la contribution de la i -ème observation à la log-vraisemblance.

$$\ell(\theta_i, \phi; y_i) = [\psi_i\theta_i - v(\theta_i)]/u(\phi) + w(y_i, \phi)$$

La log-vraisemblance s'écrit alors :

$$\mathcal{L}(\beta) = \sum_{i=1}^{n} \ln\, f(y_i; \theta_i, \phi) = \sum_{i=1}^{n} \ell(\theta_i, \phi; y_i)$$

Les équations de résolution découlant sont :

$$\sum_{i=1}^{n} \frac{(y_i - \mu_i)x_{ij}}{\text{Var}\,(Y_i)} \frac{\partial \mu_i}{\partial \eta_i} = 0 \qquad \forall j = 1, \dots, p$$

Ces équations sont non-linéaires en β et leur résolution requiert des méthodes itératives dans lesquelles interviennent le Hessien (pour Newton - Raphson) ou la matrice d'information (pour les Scores de Fisher). Nous ne décrirons pas ici en détails ces méthodes de résolution.

Qualité d'ajustement

Après modélisation, il est absolument nécessaire de jauger sa qualité en comparant les valeurs estimées et les valeurs observées.

La déviance

La déviance D est donnée par l'expression :

$$D = -2\,(L - L_{sat})$$

Où L désigne la log-vraisemblance de notre modèle et L_{sat} la log-vraisemblance du modèle dit saturé. On appelle modèle saturé, celui qui fait l'hypothèse qu'il y'a autant de paramètres que d'observations. Un tel modèle permet donc de parfaitement estimer toutes les observations.

Plus la déviance est grande, moins le modèle considéré est proche du modèle saturé et donc moins l'estimation est de bonne qualité.

On montre que D suit asymptotiquement une loi du χ^2 à n − p degrés de liberté. Il est donc possible permet de construire un test de rejet ou d'acceptation du modèle selon que la déviance est jugée significativement ou non importante.

Le test de Pearson

Il repose sur le fait que la statistique

$$X^2 = \sum_{i=1}^{n} \frac{(y_i - \hat{\mu}_i)^2}{\widehat{\text{Var}}(\hat{\mu}_i)}$$

Suit asymptotiquement tout comme la déviance, une loi de χ^2 à n - p degrés de liberté.

Le test de Pearson donne généralement des résultats voisins de ceux du critère de la déviance. Obtenir le contraire permet donc de sonner l'alarme d'une mauvaise estimation.

Sachant que l'espérance d'une loi du χ^2 est son nombre de degrés de liberté et, connaissant les aspects approximatifs des tests construits, il est souvent d'usage de comparer les statistiques avec le nombre de degrés de liberté (ddl). Ainsi, le modèle peut être jugé satisfaisant pour un rapport D/ddl plus petit que (ou voisin de) 1.

L'Akaike Information Criterion (AIC) et le Bayesian Information Criterion (BIC)

AIC = -2L + 2 p

BIC = -2L + p log (n)

L : log vraisemblance du modèle
p : nombre de paramètres
n : nombre d'observations

L'AIC et le BIC ne sont pas à proprement parler des tests au sens de test d'une hypothèse nulle, mais des critères qui pénalisent le maximum de log-vraisemblance. Ils sont simples d'usage puisque pour chacun d'eux, le meilleur modèle est celui qui minimise leur valeur.

L'AIC est asymptotiquement optimal lorsque l'on souhaite sélectionner le modèle avec l'erreur quadratique moyenne, si l'on fait l'hypothèse que le modèle générant les données n'est pas parmi les candidats, ce qui est en fait presque toujours le cas en pratique. Également, la vitesse de convergence de l'AIC vers l'optimum est, dans un certain sens, la meilleur possible. (Yang, 2005)

Implémentation sous python

Notre objectif est de modéliser le cout moyen d'un sinistre en assurance automobile grâce aux variables explicatives :

GLM

Nous implémenterons la modélisation par GLM pour 03 distributions : gamma, exponentielle et normale.

Pour ce faire il est nécessaire de transformer les variables qualitatives « Type » et « Region » en variables numériques (grâce à LabelEncoder() du module « preprocessing » de scikitlearn) catégoriques.

```
from sklearn import preprocessing
encoder1 = preprocessing.LabelEncoder()
encoder2 = preprocessing.LabelEncoder()


df_independant['Type'] = encoder1.fit_transform(df_independant.Type)
df_independant['Region2'] = encoder2.fit_transform(df_independant.Region2)

col_cats = ['Type','Region2']
for col in col_cats:
  df_independant[col] = df_independant[col].astype('category', copy=False)
```

Rappelons qu'en GLM, la valeur de la moyenne de la variable cible appartient au sous-espace engendré par les variables explicatives. C'est-à-dire :

$g(\mu_i) = \mathbf{x}_i'\beta \qquad i = 1, \dots, n$

Ainsi, il est nécessaire d'ajouter à nos variables explicatives actuelles une « variable » constante. En l'occurrence égale à 1. De cette manière, le modèle calculera le coefficient β_0 qui la multipliera dans l'expression finale de notre variable explicative.

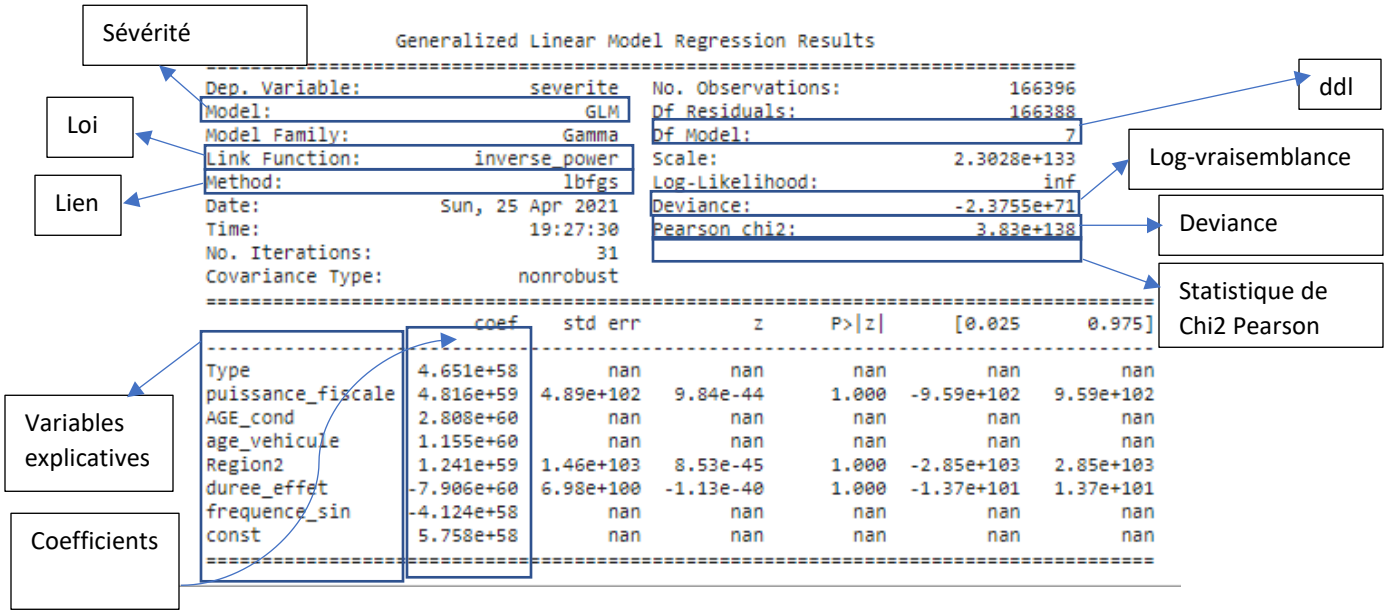
$$g(\mu_i) = \mathbf{x}_i'\beta + \beta_0$$

Sous python, le GLM est implémenté par la fonction « GLM » du module « statsmodels.api ».

Loi gamma

- **Fonction de lien : Inverse (cas canonique)**

$$g(\mathrm{x}) = \frac{1}{x}$$



Generalized Linear Model Regression Results

Dep. Variable:	severite	No. Observations:	166396
Model:	GLM	Df Residuals:	166388
Model Family:	Gamma	Df Model:	7
Link Function:	inverse_power	Scale:	2.3028e+133
Method:	lbfgs	Log-Likelihood:	inf
Date:	Sun, 25 Apr 2021	Deviance:	-2.3755e+71
Time:	19:27:30	Pearson chi2:	3.83e+138
No. Iterations:	31		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Type	4.651e+58	nan	nan	nan	nan	nan
puissance_fiscale	4.816e+59	4.89e+102	9.84e-44	1.000	-9.59e+102	9.59e+102
AGE_cond	2.808e+60	nan	nan	nan	nan	nan
age_vehicule	1.155e+60	nan	nan	nan	nan	nan
Region2	1.241e+59	1.46e+103	8.53e-45	1.000	-2.85e+103	2.85e+103
duree_effet	-7.906e+60	6.98e+100	-1.13e-40	1.000	-1.37e+101	1.37e+101
frequence_sin	-4.124e+58	nan	nan	nan	nan	nan
const	5.758e+58	nan	nan	nan	nan	nan

On obtient une log-vraisemblance « infinie » et une déviance d'environ - 2.37 e +71. Pour des valeurs si grandes, l'on peut d'ores et déjà disqualifier cette modélisation. Par ailleurs,

$$\frac{Déviance}{ddl} = -1.427e + 66 \ >> 1.$$

L'on remarquera également des valeurs d'erreurs sur les coefficients extrêmement importantes ou tout simplement non définies.

- **Fonction de lien : Identité**

$$g(\mathrm{x}) = \mathrm{x}$$

Generalized Linear Model Regression Results

Dep. Variable:	severite	No. Observations:	166396
Model:	GLM	Df Residuals:	166388
Model Family:	Gamma	Df Model:	7
Link Function:	identity	Scale:	1.0011
Method:	lbfgs	Log-Likelihood:	inf
Date:	Sun, 25 Apr 2021	Deviance:	1.1368e+07
Time:	22:21:47	Pearson chi2:	1.67e+05
No. Iterations:	94		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Type	-8.69e+04	nan	nan	nan	nan	nan
puissance_fiscale	3.212e+05	nan	nan	nan	nan	nan
AGE_cond	-4.03e+04	nan	nan	nan	nan	nan
age_vehicule	-1.976e+05	nan	nan	nan	nan	nan
Region2	-1.25e+05	nan	nan	nan	nan	nan
duree_effet	1.027e+04	nan	nan	nan	nan	nan
frequence_sin	2.15e+06	nan	nan	nan	nan	nan
const	-1.687e+04	nan	nan	nan	nan	nan

On obtient une log-vraisemblance « infinie » et une déviance d'environ 1.136 e + 07. Malgré une déviance toujours importante, l'on note une considérable amélioration par rapport au cas de la fonction de lien inverse.

$\frac{Déviance}{ddl} = 68.31 >> 1.$

Cependant le modèle n'est toujours pas satisfaisant.

- **Fonction de lien : Logarithme**

Le calcul sous python n'a pas abouti du fait d'une valeur β_{i_0} invalide lors de l'optimisation.

Loi normale

- **Fonction de lien : identité (cas canonique)**

Generalized Linear Model Regression Results

Dep. Variable:	severite	No. Observations:	166396
Model:	GLM	Df Residuals:	166388
Model Family:	Gaussian	Df Model:	7
Link Function:	identity	Scale:	9.5664e+05
Method:	lbfgs	Log-Likelihood:	-1.3818e+06
Date:	Sun, 25 Apr 2021	Deviance:	1.5917e+11
Time:	22:36:54	Pearson chi2:	1.59e+11
No. Iterations:	0		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Type	37.5804	5.276	7.122	0.000	27.239	47.922
puissance_fiscale	-6.4615	1.105	-5.850	0.000	-8.627	-4.297
AGE_cond	-1.3505	0.174	-7.757	0.000	-1.692	-1.009
age_vehicule	-8.1133	0.285	-28.494	0.000	-8.671	-7.555
Region2	-23.7482	1.980	-11.997	0.000	-27.628	-19.868
duree_effet	0.3839	0.010	39.318	0.000	0.365	0.403
frequence_sin	896.0226	4.168	214.978	0.000	887.854	904.192
const	230.9128	13.520	17.080	0.000	204.415	257.411

On obtient dans ce cas, une log-vraisemblance de -1.381 e+06 et une **déviance** d'environ **1.1591 e + 11**.

$\frac{Déviance}{ddl} = 956\,619.467 >> 1$. Le modèle n'est donc pas satisfaisant.

- **Fonction de lien : Logarithme**

Generalized Linear Model Regression Results

Dep. Variable:	severite	No. Observations:	166396
Model:	GLM	Df Residuals:	166388
Model Family:	Gaussian	Df Model:	7
Link Function:	log	Scale:	1.2786e+06
Method:	lbfgs	Log-Likelihood:	-1.4060e+06
Date:	Sun, 25 Apr 2021	Deviance:	2.1274e+11
Time:	22:41:19	Pearson chi2:	2.13e+11
No. Iterations:	2		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Type	0.0185	1.66e+05	1.11e-07	1.000	-3.25e+05	3.25e+05
puissance_fiscale	-0.0351	3.56e+04	-9.86e-07	1.000	-6.97e+04	6.97e+04
AGE_cond	-0.1252	8204.306	-1.53e-05	1.000	-1.61e+04	1.61e+04
age_vehicule	-0.0477	1.63e+04	-2.92e-06	1.000	-3.2e+04	3.2e+04
Region2	-0.0243	1.21e+04	-2.01e-06	1.000	-2.36e+04	2.36e+04
duree_effet	-0.9903	1.77e+04	-5.61e-05	1.000	-3.46e+04	3.46e+04
frequence_sin	0.0108	nan	nan	nan	nan	nan
const	7.0850	2.79e+05	2.54e-05	1.000	-5.47e+05	5.47e+05

On obtient dans ce cas, une log-vraisemblance de -1.406 e+06 et une **déviance** d'environ **2.127 e + 11**.

$\frac{Déviance}{ddl} = 1\,278\,337.38 >> 1$. Le modèle n'est donc pas satisfaisant.

- **Fonction de lien : Inverse**

Generalized Linear Model Regression Results

Dep. Variable:	severite	No. Observations:	166396
Model:	GLM	Df Residuals:	166388
Model Family:	Gaussian	Df Model:	7
Link Function:	inverse_power	Scale:	1.2764e+06
Method:	lbfgs	Log-Likelihood:	-1.4058e+06
Date:	Sun, 25 Apr 2021	Deviance:	2.1238e+11
Time:	22:44:23	Pearson chi2:	2.12e+11
No. Iterations:	69		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Type	-2.899e-08	0.000	-0.000	1.000	-0.000	0.000
puissance_fiscale	-7.925e-05	2.84e-05	-2.790	0.005	-0.000	-2.36e-05
AGE_cond	-8.416e-06	5.7e-06	-1.477	0.140	-1.96e-05	2.75e-06
age_vehicule	5.3e-05	9.38e-06	5.651	0.000	3.46e-05	7.14e-05
Region2	-4.812e-05	5.57e-05	-0.864	0.387	-0.000	6.1e-05
duree_effet	0.0003	1.78e-05	15.350	0.000	0.000	0.000
frequence_sin	-0.0003	2.08e-05	-15.469	0.000	-0.000	-0.000
const	2.075e-05	0.000	0.051	0.959	-0.001	0.001

On obtient dans ce cas, une log-vraisemblance de -1.405 e+06 et une **déviance** d'environ **2.123 e + 11**.

$\frac{Déviance}{ddl} = 1\ 276\ 414.164 >> 1$. Le modèle n'est donc pas satisfaisant.

Récapitulatif

Le tableau ci-dessus récapitule les résultats de modélisation ci-haut présenté. Il permettra de sélectionner le « meilleur » modèle.

Il comporte également la RMSE des prédictions par le modèle considéré sur les données.

Loi	Lien	Log-V (e+06)	Déviance (e+11)	Pearson (e+11)	DDL	AIC	BIC	RMSE
Gamma	Inverse	Inf.	-2.37^{E}64	3.83^{E}131	166388	-inf.	-2.37^{E}71	1130.72
Gamma	Identité	Inf.	0.000113	0.0165	166388	-inf.	9.36^{E}6	3.83^{E}6
Normale	Identité	-1.381	1.159	1.59	166388	2.763^{E}6	159.1^{E}9	978.05
Normale	Log	-1.406	2.127	2.13	166388	2.81^{E}6	212.7^{E}9	1130.72
Normale	Inverse	-1.405	2.123	2.12	166388	2.81^{E}6	212.38^{E}9	1129.76

Aucune des modélisations ci-dessus ne remplit les critères de satisfaction statistiques décrits dans la rubrique « qualité d'ajustement » de la description théorie du GLM, pour être acceptée. Cependant selon le paramètre que l'on veut mettre en exergue (AIC, BIC, RMSE, déviance/ddl, etc.), l'on peut faire un choix de moindre mal.

La loi exponentielle n'est pas testée ici car python ne la prend pas en charge dans le cadre du GLM. Chose dommage puisqu'elle est celle qui selon notre inférence statistique sur la loi, se rapproche le plus de la description réelle des observations de sévérité.

Vu sa plus grande proximité avec les données observés (RMSE minimal), nous considérerons que le modèle loi normale – lien identité est le meilleur. Il nous servira de base de comparaison avec les modèles de Machine Learning.

*** Expression du modèle Loi normale – lien identité :**

$$\boldsymbol{E}(\text{Sévérité}) = \mathbf{37.58} * \text{Type} - \mathbf{6.46} * \text{Puissance fiscale} - \mathbf{1.35} * \boldsymbol{Age\ du\ conducteur} - \mathbf{8.11} * \boldsymbol{Age\ du\ véhicule} - \mathbf{23.74} * \boldsymbol{Region} + \mathbf{0.38} * \boldsymbol{Duree\ effet} + \mathbf{896.02} * \boldsymbol{Fréquence\ des\ sinistres} + \mathbf{230.91}$$

Selon ce modèle, l'on peut donc classer les variables explicatives par ordre de prépondérance décroissante sur le cout d'un sinistre comme suit: la fréquence des sinistres, le type de véhicule, la région, l'age du véhicule, sa puissance fiscale, l'age du conduecteur et enfin la durée de son effet.

VI. Application des modèles Machine Learning

Après avoir conclu notre modèle GLM, nous procédons à la présentation des prédictions basées sur des algorithmes de Machine Learning.

L'objectif de cette partie est de comparer l'application de différents modèles de Machine Learning à notre base de données.

Avant de commencer, notons que pour évaluer la robustesse des modèles, nous avons opté pour la métrique RMSE afin d'évaluer l'erreur type de chaque modèle.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Notre variable à prédire est continue alors nous utiliserons des méthodes de régression que nous présentons par la suite.

Pour tout ce qui suit, afin de déterminer les paramètres les plus appropriés à tous nos modèles, nous subdivisons notre base de données en deux parties, 80% pour l'entrainement et pour des fins d'évaluation : 20% pour la validation.

Régression linéaire

Dans cette partie nous allons élaborer le modèle de la régression linéaire sur notre base de données

Coefficients de la régression linéaire

Nous avons trouvé les coefficients suivants, après avoir appliqué la régression sur la base de données :

	coef	std err	t	P>|t|	[0.025	0.975]
x1	-5.8118	1.255	-4.632	0.000	-8.271	-3.352
x2	-1.2787	0.198	-6.459	0.000	-1.667	-0.891
x3	-7.8370	0.324	-24.166	0.000	-8.473	-7.201
x4	0.3871	0.011	34.950	0.000	0.365	0.409
x5	799.1107	4.442	179.884	0.000	790.404	807.818
x6	158.8578	6.541	24.285	0.000	146.037	171.679
x7	-5.4139	7.989	-0.678	0.498	-21.071	10.244
x8	12.3278	7.024	1.755	0.079	-1.440	26.095
x9	85.5032	6.322	13.524	0.000	73.111	97.895
x10	103.5504	10.309	10.045	0.000	83.345	123.756
x11	147.7245	9.929	14.878	0.000	128.263	167.185

On remarque que la variable x5 « la fréquence de sinistre » est la variable la plus importante dans le model de la régression linéaire. Ce qui est tout à fait logique.

Performance et prédiction

```
# RMSE régression linéaire
rmse(Y_test,Y_pred)
```

```
919.53657245238
```

Light GBM

Le modèle que nous exploiterons dans cette partie est le Light GBM (Light Gradient Boosting Machine), un framework de boosting de gradient qui utilise un algorithme d'apprentissage arborescent.

Paramètres de Tuning

- Max_depth : C'est le seuil sur la profondeur maximale de l'arbre. (Nombre de bifurcations) Il est utilisé pour réduire l'overfitting.
- Learning_rate : définit le pas chaque itération en avançant vers un minimum de la fonction loss. Ce paramètre détermine l'impact de chaque arbre sur le modèle final.
- Num_leaves : C'est le nombre de « leaves » dans un arbre complet. Il contrôle la complexité de l'algorithme d'arborescence.
- N_estimators : Puisque le LightGBM est un Framework qui utilise plusieurs arbres, ce paramètre contrôle le nombre d'arbres utilisés dans le processus.

Nous effectuons une recherche sur grille de la manière suivante :

```
lgbm = lgb.LGBMRegressor()
param_dist = {"max_depth": [25,50, 75],
              "learning_rate" : [0.01,0.05,0.1],
              "num_leaves": [300,900,1200],
              "n_estimators": [200]
             }
grid_search = GridSearchCV(lgbm, param_grid=param_dist, cv = 4,
                                    verbose=10, n_jobs=-1)
grid_search.fit(X_train, Y_train)

grid_search.best_estimator_
```

Les paramètres du meilleur modèle trouvé sont :

```
LGBMRegressor(boosting_type='gbdt', class_weight=None, colsample_bytree=1.0,
              importance_type='split', learning_rate=0.05, max_depth=25,
              min_child_samples=20, min_child_weight=0.001, min_split_gain=0.0,
              n_estimators=200, n_jobs=-1, num_leaves=300, objective=None,
              random_state=None, reg_alpha=0.0, reg_lambda=0.0, silent=True,
              subsample=1.0, subsample_for_bin=200000, subsample_freq=0)
```

Paramètres de Fitting

Les paramètres de fitting sont des paramètres supplémentaires utilisés pour l'entrainement du modèle. Les paramètres utilisés dont nous citons les principaux :

- Early_stopping_rounds : (Nous désignons par E le paramètre early_stopping_rounds pour cette description) Ce paramètre aide à accélérer l'analyse du modèle. Le modèle arrête l'entrainement si l'une des métriques d'une validation ne s'améliore pas pour E itérations.
- Eval_metric : Désigne la métrique utilisée pour valider le modèle.
- Eval_set : Désigne la dataset d'évaluation. (Dans notre cas composé de deux colonnes)

```
# Fiting
fit_params={"early_stopping_rounds":10,
            "eval_metric" : 'neg_mean_squared_error',
            "eval_set" : [(X_test,Y_test)],
            'eval_names': ['valid'],
            'verbose': 100,
            'feature_name': 'auto',
            'categorical_feature': cat_features
           }
lgbm = lgb.LGBMRegressor(**hyper_params)
lgbm.fit(X_train, Y_train, **fit_params)
```

Les paramètres du modèle trouvé sont :

```
LGBMRegressor(boosting_type='gbdt', class_weight=None, colsample_bytree=1.0,
              importance_type='split', learning_rate=0.05, max_depth=25,
              min_child_samples=20, min_child_weight=0.001, min_split_gain=0.0,
              n_estimators=200, n_jobs=-1, num_leaves=300, objective=None,
              random_state=None, reg_alpha=0.0, reg_lambda=0.0, silent=True,
              subsample=1.0, subsample_for_bin=200000, subsample_freq=0)
```

Résultats

```
[ ]  # RMSE Light GBM
     rmse(Y_pred,Y_test,Y_pred_train,Y_train)

     (445.1200754069927, 408.9188302452841)
```

XGBoost

XGBoost est une implémentation optimisée de boosting de gradient. Cet algorithme est mis en place de telle manière à être efficace, portable et flexible.

Paramètes de Tuning

- Max_depth : C'est le seuil sur la profondeur maximale de l'arbre. (Nombre de bifurcations) Il est utilisé pour réduire l'overfitting.
- Learning_rate : définit le pas chaque itération en avançant vers un minimum de la fonction loss.
- N_estimators : Nombre d'itérations total. (Nombre d'arbres)

- Min_child_weight : nombre d'observations dans le nœud terminal de l'arbre.

```
# Hyperparameter tuning

xgb_ = xgb.XGBRegressor()
param_dist_ = {"max_depth": [10,30,50],
               "min_child_weight" : [1,3,6],
               "n_estimators": [200],
               "learning_rate": [0.05, 0.1,0.16],}
grid_search = GridSearchCV(xgb_, param_grid=param_dist_, cv = 4,
                                  verbose=10, n_jobs=-1)
grid_search.fit(X_train_, Y_train_)

grid_search.best_estimator_
```

Le meilleur modèle trouvé est de paramètres :

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0,
             importance_type='gain', learning_rate=0.05, max_delta_step=0,
             max_depth=10, min_child_weight=6, missing=None, n_estimators=200,
             n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
             silent=None, subsample=1, verbosity=1)
```

Résultats

```
[ ]  rmse(y_predX,Y_test_,y_pred_trainX,Y_train_)

     (456.41122286051046, 342.01862130273406)
```

Conclusion

Après avoir étudier les différents modèles de régression et faire une comparaison basée sur le RMSE de chaque model, Nous concluons que le model « Light GBM » présente la moindre valeur de RMSE. Celui-ci est alors le « meilleur » model des 3 modèles étudiés.

VII. Conclusion générale

En définitive, dans notre exercice de modélisation du cout moyen d'un sinistre en assurance automobile, nous avons implémenté deux méthodes : les modèles linéaires généralisés et les modèles de machine learning. En prélude de cette implémentation, il a été nécessaire d'inclure la variable « sévérité », quantité à prédire tenant lieu de cout moyen. L'exploration de notre base de données support nous a permis détecter ses anomalies et les corriger. Une analyse univariée nous a permis de caractériser chaque variable explicative, puis une analyse bivariée des données a mis en évidence les relations entre ces variables.

Le meilleur modèle GLM est obtenu pour l'hypothèse de la sévérité suivant une loi gaussienne avec la fonction identité pour fonction lien. Toutes fois, en termes de RMSE, les modèles machine learning

ont donné un meilleur résultat, en particulier le LightGBM (meilleur modèle de machin learning obtenu).

Cependant, il est important de souligner que les modèles de machine learning ont l'inconvénient d'etre des boites noires, ils ne nous donnent pas d'informations sur le poids relatif d'une variable explicative par rapport à une autre, dans une prédiction. À la différence du GLM qui explicite la relation mathématique entre sévérité et variables indépendantes.

Enfin, il a été difficile d'obtenir des scores probant sur l'ensemble de notre base de données initiale en raison d'un fort biais. En effet, 96% des enregistrements de la dataset présentent une sévérité nulle. Problème que nous avons essayé de solutionner en supprimant les outliers (severité supérieure à 15 000), soit travaillant avec 96.8% de notre base initiale (avant traitement).