REPUBLIQUE DU SENEGAL

Un Peuple – Un But – Une Foi

Ministère de l'Enseignement Supérieur de la Recherche et de l'innovation

Université Alioune DIOP de Bambey

Statistique et Informatique Décisionnelle

Projet : TRAITEMENT DES DONNEES, DATA MINING, ACTUARIAT et ECONOMETRIE

Sujet : Sécurité Alimentaire au Sénégal

Réaliser par :

Amadou BA

Malick FALL

Mahmoud SIDIBE

Professeur :

Dr. El Hadji

Mamadou Dieng

NGOM

Année universitaire : 2023 – 2024

1. **Problématique**

L'insécurité alimentaire est devenue un problème plus fréquent et plus persistant dans plusieurs pays en développement, suite à la menace climatique.

D'après le Groupe de travail sur la faim au niveau de l'ONU: « en Afrique Sub-Saharienne, la majorité des pauvres qui souffrent de la faim vivent dans les zones rurales, fréquemment exposées aux changements climatiques ». Au Sénégal, en milieu rural, le gouvernement avait adopté des mesures urgentes afin de s'assurer que les personnes les plus vulnérables à la faim améliorent leurs capacités de s'adapter aux impacts négatifs du changement climatique, tout en renforçant leur résilience à l'insécurité alimentaire. Malgré cela, l'insécurité alimentaire s'était empirée.

Pour faire face à cette situation, le Gouvernement du Sénégal a décidé de mener une enquête sur les zones à risque d'insécurité alimentaire.

2. **Questions recherches** :

La question principale à laquelle le Gouvernement veut apporter des solutions est la suivante:

Comment l'Intelligence artificielle pourrait contribuer à l'analyse de la sécurité alimentaire dans les zones agricoles influencées par le changement climatique ?

Plus spécifiquement :

- *Quelle description donner à la vulnérabilité à l'insécurité alimentaire?*
- *Comment le changement climatique aggrave la vulnérabilité des ménages?*
- *Quelle est l'impact du changement climatique sur le bien être des ménages agricoles ?*
- *Les interventions de l'Etat pour promouvoir l'assurance agricole (en faisant payer des primes) aux personnes nécessiteuses sont-elles suffisantes pour leur résilience ?*

3. **Objectifs**

L'Enquête Rurale sur l'Agriculture, la Sécurité Alimentaire et la Nutrition (ERASAN) a été menée en quatre mois, du 15 Juin

au 15 Septembre 2024 avec l'appui du programme alimentaire mondial (PAM). Cette enquête a porté sur 1398 ménages

agricoles. L'enquête a permis de collecter des informations liées à l'exposition des ménages aux chocs climatiques et à leur

état alimentaire durant les périodes de soudure (ou les ménages manquent de ressources financières et alimentaires). La base des données issue de l'enquête est nommée « Data.ERASAN.2024 »

TRAVAIL demandé

Trois étudiants par groupe, les mêmes groupes constitués en classe (sinon ce sera une sanction)

- Donner les hypothèses et les formulations de tout test ou modèle (avant de mettre les résultats : **Exigé !**) ;

- Décrire les procédures IA utilisées avec les codes Python ou SPSS (avec capture d'écran à l'appui). Python est récompensé ;
- Commentez les résultats par rapport aux effets des changements climatiques sur la sécurité alimentaire .

PS: Tous les exercices utilisent la même base de données issue du traitement

Table des matières

1. TRAITEMENT DES DONNEES (20 POINTS)

1.1 Extraire les (variables; données) à partir du fichier "DAPSA.ERASAN.2024"

Ménages ayant procéder à des depalim alimentaires (sup à 12000) avec revenu (sup à 12000)

ID, département, période.soudure, sexe, taille, age, inondation; pluies insuffisante, pluies hors saison, sci, sca, gca, revenu, depalim, part.depalim Alim.MoinsPréf, Emprunt.Alim, Red.QuantNourr, Red.NbrRepasJour, Garder.la.Faim, Type.Ass.Agr, Prestation.reçue, Consommation.Ass, Frequence.Ass, Taux.Indemnisation,

Renommer ce fichier par "***Climat.SécAlim.2024***"

```
# 🔷 Filtrer les ménages selon conditions
df_filtered = df[(df['Montant.Revenu'] > 12000)].copy()

# 🔷 Liste des variables disponibles à extraire
vars_utiles = [
    'ID', 'département', 'période.soudure', 'sexe', 'taille', 'age',
    'Inondation', 'Pluies.Insuf', 'Pluies.HorsSaison',
    'SCI', 'SCA', 'Gca',
    'Montant.Revenu', 'Dep.Alim', 'Part.DepAlim',
    'Alim.MoinsPréf', 'Emprunt.Alim', 'Red.QuantNourr',
    'Red.NbrRepasJour', 'Garder.la.Faim', 'Type.Ass.Agr'
]

# 🔷 Extraction
df_climat_secalim = df_filtered[vars_utiles]

# 🔷 Sauvegarde
df_climat_secalim.to_csv("Climat.SecAlim.2024.csv", index=False)

print("Extraction et sauvegarde terminées. Fichier créé :
Climat.SecAlim.2024.csv")
```

Extraction et sauvegarde terminées. Fichier créé : Climat.SecAlim.2024.csv

1.2. Etudier la dispersion des variables : sci, sca, revenu, depalim, gca, part.depalim (pour les variables qualitatives se référer à l'Indice de variation qualitative IQV)

1.1.1. Analyse des variables quantitatives

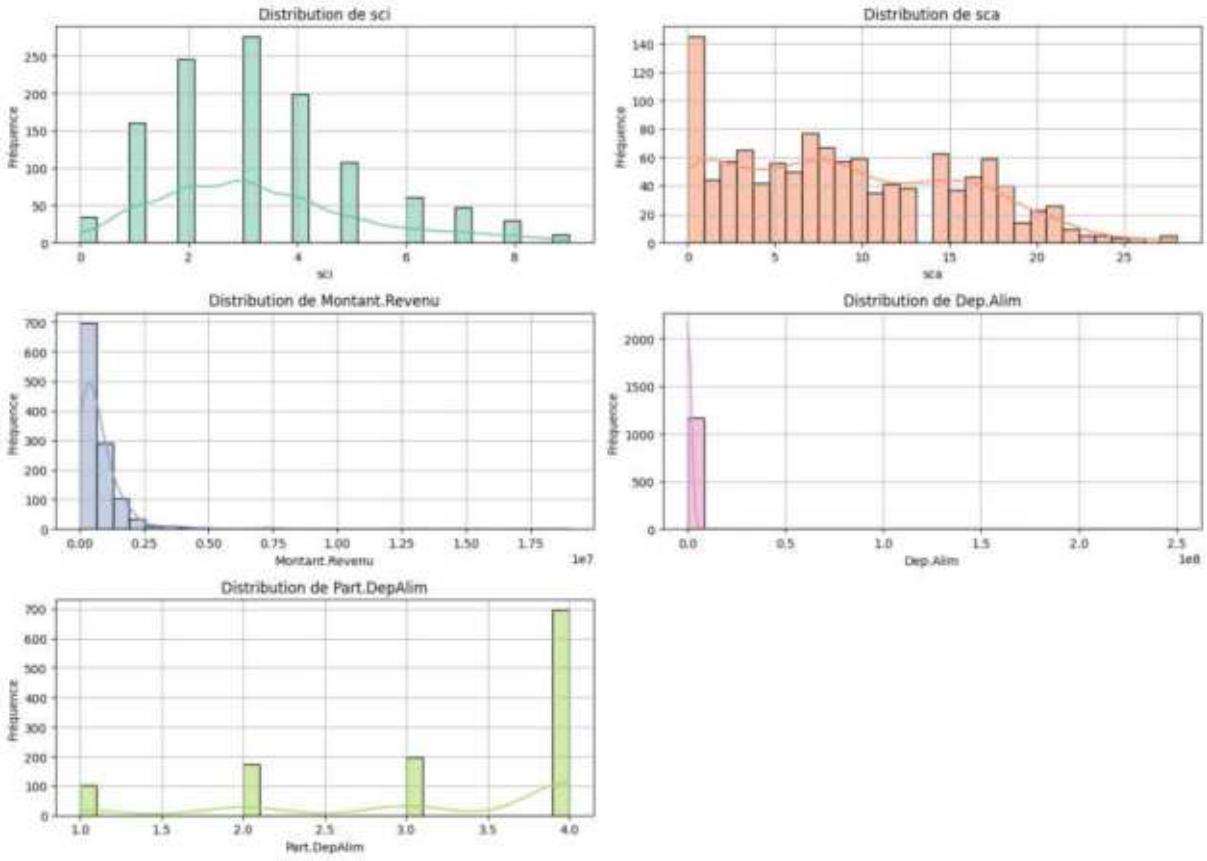
```
# 🔷 Renommons le nom de certains variables
df.rename(columns={'Montant.Revenu':'revenu', 'Dep.Alim':'depalim'},
inplace=True)
variables_quanti = ['SCI', 'SCA', 'revenu', 'depalim']

# 🔷 Statistiques descriptives
print("📊 Statistiques descriptives des variables quantitatives :")
```

Statistiques descriptives des variables quantitatives :

	SCI	SCA	revenu	depalim
count	1398.000000	1398.000000	1.398000e+03	1.398000e+03
mean	11.482833	56.595136	7.414336e+05	2.761224e+05
std	11.740120	23.411301	1.494194e+06	6.735428e+06
min	0.000000	0.000000	0.000000e+00	0.000000e+00
25%	2.000000	40.000000	1.300000e+05	2.800000e+04
50%	8.000000	54.000000	4.000000e+05	4.600000e+04
75%	17.000000	74.000000	9.000000e+05	7.150000e+04
max	56.000000	112.000000	1.900000e+07	2.501065e+08

Les résultats montrent que la variable SCI présente une moyenne de 11,48 et un écart-type de 11,74, traduisant une grande variabilité des stratégies de coping climatiques entre les ménages, avec des valeurs allant de 0 à 56. La variable SCA a une moyenne plus élevée, à 56,60, et un écart-type de 23,41, indiquant une utilisation généralisée mais dispersée des stratégies alimentaires, son maximum atteignant 112. Concernant le revenu, la moyenne est de 741 433 FCFA mais l'écart-type est extrêmement élevé (1 494 194 FCFA), montrant de fortes disparités économiques au sein de l'échantillon, avec des revenus variant de 0 à 19 millions FCFA. Les dépenses alimentaires (Depalim) suivent la même tendance avec une moyenne de 276 122 FCFA et un écart-type de 6 735 428 FCFA, laissant apparaître des valeurs extrêmes, notamment un maximum de 250 millions FCFA, nécessitant une vérification des outliers. Les médianes indiquent également des distributions asymétriques à droite, notamment pour le revenu (400 000 FCFA) et depalim (46 000 FCFA), où de nombreux ménages dépensent peu tandis que quelques-uns ont des montants très élevés. Globalement, ces statistiques décrivent une population caractérisée par une forte hétérogénéité, tant sur le plan économique que sur les stratégies d'adaptation et de consommation alimentaire, ce qui devra être pris en compte lors des analyses multivariées et des modélisations ultérieures.

```
# 🔷 Visualisation : histogrammes avec courbe de densité
colors = sns.color_palette("Set2", len(variables_quanti))

plt.figure(figsize=(12, 8))
for i, var in enumerate(variables_quanti):
    plt.subplot(2, 2, i+1)
    sns.histplot(df[var], kde=True, color=colors[i])
    plt.title(f"Distribution de {var}")
    plt.grid(True)
plt.tight_layout()
plt.show()
```



Les histogrammes présentés illustrent la distribution des variables quantitatives. La variable sci présente une distribution légèrement asymétrique vers la droite. La majorité des observations se situent entre 2 et 4, avec une fréquence maximale proche de 250–300. Peu d'individus dépassent la valeur de 8. Cela montre que la plupart des valeurs restent concentrées autour de faibles scores, traduisant une certaine homogénéité sur cet indicateur. La distribution de sca est plus étalée et affiche une asymétrie positive marquée. Beaucoup d'observations se concentrent sous la valeur 5, mais la distribution s'étend jusqu'à 25. Cela révèle une forte

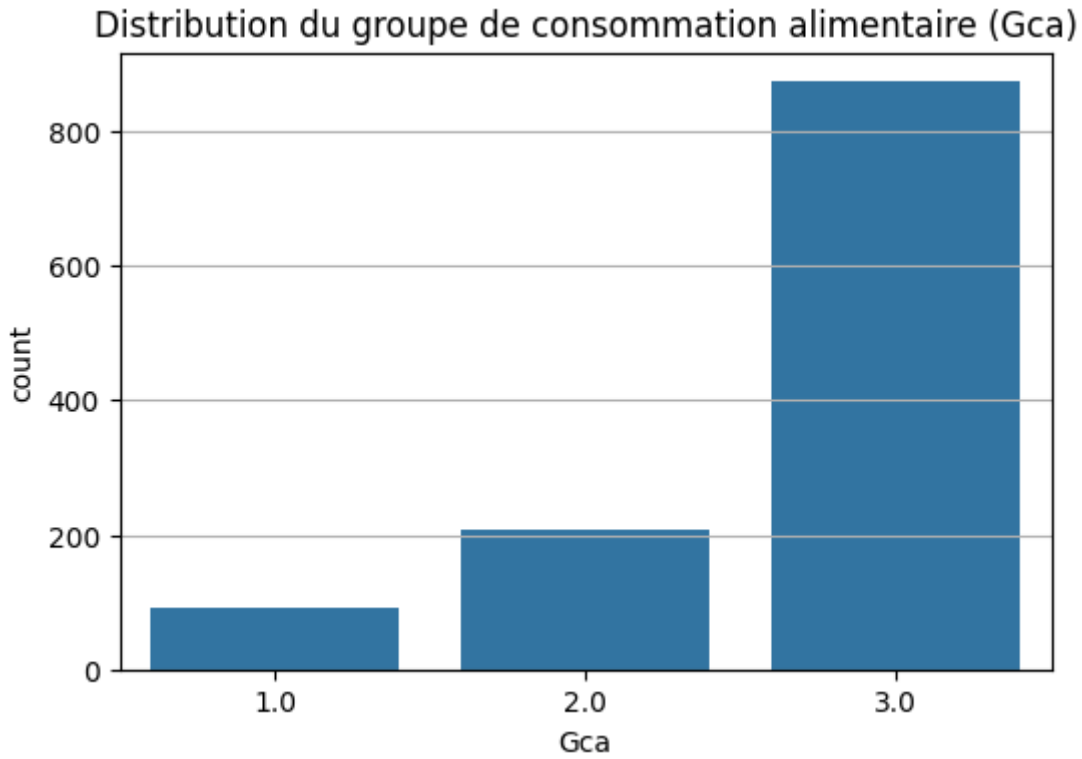
hétérogénéité, certains individus se distinguant par des valeurs particulièrement élevées. La variable Montant.Revenu est fortement asymétrique à droite. La grande majorité des montants sont situés entre 0 et 2 000 000, mais quelques valeurs extrêmes atteignent presque 18 000 000. Ce contraste important suggère qu'une minorité d'individus disposent de revenus exceptionnellement hauts, ce qui pourrait nécessiter une transformation logarithmique pour l'analyse statistique. La variable Dep.Alim présente une asymétrie encore plus extrême. La quasi-totalité des observations correspondent à de faibles montants, tandis qu'un petit nombre dépasse les 250 000 000. Cela traduit une variabilité considérable des dépenses alimentaires, avec des cas hors normes qui influencent fortement la distribution. La distribution de Part.DepAlim est multimodale, avec des pics bien marqués sur des valeurs entières (1, 2, 3, 4). Cela suggère qu'il s'agit probablement d'une variable catégorielle ordinale, représentant par exemple des classes ou des parts de dépenses alimentaires.

1.1.2. Calcul de l'IQV pour les variables qualitatives

```
def calc_iqv(series):

    freq = series.value_counts(normalize=True)
    k = len(freq)
    if k <= 1:
        return 0
    sum_pi2 = sum(freq**2)
    iqv = (k / (k - 1)) * (1 - sum_pi2)
    return iqv

# 🔷 Variables qualitatives
variables_quali = ['Gca', 'Part.DepAlim']

for var in variables_quali:
    iqv = calc_iqv(df[var])
    print(f"IQV de la variable {var} : {iqv:.3f}")

    # 🔷 Diagramme en barres
    plt.figure(figsize=(5,3))
    sns.countplot(y=var, data=df, palette="Set3")
    plt.title(f"Distribution de {var}")
    plt.grid(True, axis='x')
    plt.show()
```

IQV de la variable Gca : 0.652

L'IQV de Gca est de: 0.610, indiquant une dispersion modérée à élevée entre les différentes catégories de consommation alimentaire.



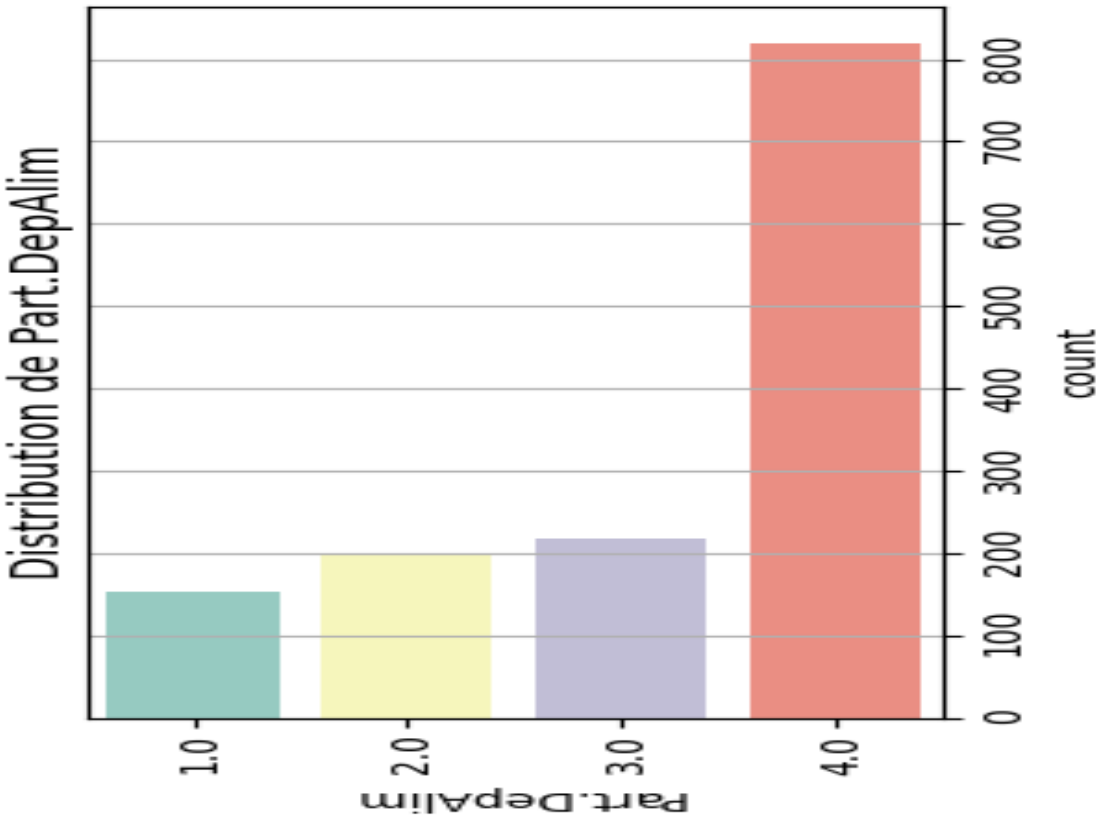
Distribution du groupe de consommation alimentaire (Gca)

La variable Gca présente une distribution catégorielle à trois modalités (1, 2 et 3). La modalité **3** est largement majoritaire avec environ 870 observations, suivie par la modalité **2** qui regroupe un peu plus de 200 observations. La modalité **1** est la moins représentée, avec environ 90 observations.

Cette répartition indique que la majorité des individus appartiennent au groupe de consommation alimentaire le plus élevé ou le plus fréquent (catégorie 3), tandis que les catégories 1 et 2 sont nettement moins fréquentes. Cette concentration peut refléter une homogénéité des comportements alimentaires ou une catégorisation qui distingue clairement un groupe prédominant.

IQV de la variable Part.DepAlim : 0.795

L'IQV de Part.DepAlim est de 0,795, indiquant une dispersion élevée entre les différentes catégories de part des dépenses alimentaires.



Le graphique montre la distribution de la variable Part.DepAlim, qui semble être catégorielle avec quatre modalités (1.0 à 4.0). On observe une forte concentration des observations dans la modalité 4.0, qui dépasse largement les 800 occurrences, tandis que les modalités 1.0, 2.0 et 3.0 sont nettement moins fréquentes, avec des effectifs relativement similaires et bien plus faibles. Cette distribution déséquilibrée suggère que la majorité des individus partagent une même caractéristique (selon ce que représente la modalité 4.0), ce qui pourrait avoir un impact sur certaines analyses statistiques ou modèles prédictifs. Il serait donc important de comprendre précisément ce que signifie chaque modalité pour interpréter correctement cette répartition.

1.3. Procéder à l'Apurement des données

taille, age, revenu, depalim, sca, revenu

1.3.1 Apurement de la variable Taille

```
var = 'taille'

# Détection des bornes IQR
q1 = df[var].quantile(0.25)
q3 = df[var].quantile(0.75)
iqr = q3 - q1
lower = q1 - 1.5 * iqr
upper = q3 + 1.5 * iqr

# Affichage des bornes et nombre d'outliers
outliers = df[(df[var] < lower) | (df[var] > upper)]
print(f"🔎 {var} : {outliers.shape[0]} outliers")
print(f"Borne inférieure : {lower:.2f} | Borne supérieure : {upper:.2f}")

# Remplacer les outliers par NaN
df[var] = df[var].where((df[var] >= lower) & (df[var] <= upper))

# Vérifier le résultat
print(df[var].describe())
print("Valeurs manquantes après apurement :", df[var].isnull().sum())
```

```
🔎 taille : 101 outliers

Borne inférieure : -0.50 | Borne supérieure : 3.50
count    1297.000000
mean        1.800308
std         0.690142
min         1.000000
25%         1.000000
50%         2.000000
75%         2.000000
max         3.000000
Name: taille, dtype: float64
Valeurs manquantes après apurement : 101
```

Les bornes calculées par la méthode de l'IQR étaient -0,50 et 3,50, détectant ainsi **101 valeurs aberrantes**, toutes supérieures à la borne supérieure. Le nombre total d'observations restantes est de 1297, avec 101 valeurs manquantes correspondant aux outliers supprimés. Ces résultats confirment une distribution concentrée entre les modalités 1 et 3.

1.3.2. Apurement de la variable Age

```
var = 'age'

# Détection des bornes IQR
q1 = df[var].quantile(0.25)
q3 = df[var].quantile(0.75)
iqr = q3 - q1
lower = q1 - 1.5 * iqr
upper = q3 + 1.5 * iqr
# Affichage des bornes et nombre d'outliers
outliers = df[(df[var] < lower) | (df[var] > upper)]
print(f"🔎 {var} : {outliers.shape[0]} outliers")
print(f"Borne inférieure : {lower:.2f} | Borne supérieure : {upper:.2f}")
# Remplacer les outliers par NaN
df[var] = df[var].where((df[var] >= lower) & (df[var] <= upper))
# Vérifier le résultat
print(df[var].describe())
print("Valeurs manquantes après apurement :", df[var].isnull().sum())
```

```
🔎 age : 0 outliers

Borne inférieure : 0.50 | Borne supérieure : 4.50
count    1373.000000
mean        2.734887
std         0.627073
min         1.000000
25%         2.000000
50%         3.000000
75%         3.000000
max         4.000000
Name: age, dtype: float64
Valeurs manquantes après apurement : 25
```

Pour la variable **age**, aucune valeur aberrante n'a été détectée, avec des bornes IQR calculées entre 0,50 et 4,50. Après apurement, la variable conserve un total de **1373 observations** et présente une moyenne de **2,73**, avec un minimum de 1 et un maximum de 4. Les 25 valeurs manquantes correspondent aux valeurs initialement absentes avant l'apurement. La distribution est concentrée entre les modalités 2 et 4, suggérant que **age est codée en catégories** plutôt qu'en valeurs continues, ce qui confirme la cohérence des résultats.

1.3.3. Apurement de la variable revenue

```
var = 'revenu'

# Détection des bornes IQR
q1 = df[var].quantile(0.25)
q3 = df[var].quantile(0.75)
iqr = q3 - q1
lower = q1 - 1.5 * iqr
upper = q3 + 1.5 * iqr
# Affichage des bornes et nombre d'outliers
outliers = df[(df[var] < lower) | (df[var] > upper)]
print(f"🔎 {var} : {outliers.shape[0]} outliers")
print(f"Borne inférieure : {lower:.2f} | Borne supérieure : {upper:.2f}")

# Remplacer les outliers par NaN
df[var] = df[var].where((df[var] >= lower) & (df[var] <= upper))

# Vérifier le résultat
print(df[var].describe())
print("Valeurs manquantes après apurement :", df[var].isnull().sum())
```

```
🔎 revenu : 67 outliers

Borne inférieure : -1025000.00 | Borne supérieure : 2055000.00
count    1.331000e+03
mean     5.142491e+05
std      4.941698e+05
min      0.000000e+00
25%      1.200000e+05
50%      3.500000e+05
75%      8.000000e+05
max      2.043000e+06
Name: revenu, dtype: float64
Valeurs manquantes après apurement : 67
```

Pour la variable **revenu**, 67 valeurs aberrantes ont été détectées et remplacées par des valeurs manquantes. Les bornes calculées étaient de -1 025 000 FCFA (borne inférieure non applicable car négative) et 2 055 000 FCFA. Après apurement, la moyenne du revenu est de **514 249 FCFA** et l'écart-type de **494 170 FCFA**, avec un maximum de 2 043 000 FCFA. Ces résultats confirment une distribution asymétrique à droite, indiquant que la majorité des ménages ont des revenus modestes tandis qu'un petit nombre possède des revenus élevés.

1.3.4. Apurement de la variable depalim

```
var = 'depalim'

# Détection des bornes IQR
q1 = df[var].quantile(0.25)
q3 = df[var].quantile(0.75)
iqr = q3 - q1
lower = q1 - 1.5 * iqr
upper = q3 + 1.5 * iqr
# Affichage des bornes et nombre d'outliers
outliers = df[(df[var] < lower) | (df[var] > upper)]
print(f"🔎 {var} : {outliers.shape[0]} outliers")
print(f"Borne inférieure : {lower:.2f} | Borne supérieure : {upper:.2f}")
# Remplacer les outliers par NaN
df[var] = df[var].where((df[var] >= lower) & (df[var] <= upper))
# Vérifier le résultat
print(df[var].describe())
print("Valeurs manquantes après apurement :", df[var].isnull().sum())
```

```
🔎 depalim : 77 outliers

Borne inférieure : -37250.00 | Borne supérieure : 136750.00
count      1321.000000
mean      47960.577593
std       29201.432926
min           0.000000
25%       27000.000000
50%       43250.000000
75%       65500.000000
max      136500.000000
Name: depalim, dtype: float64
Valeurs manquantes après apurement : 77
```

Pour la variable **depalim**, 77 valeurs aberrantes ont été détectées et remplacées par des valeurs manquantes. Les bornes calculées étaient de -37 250 FCFA (borne inférieure non applicable car négative) et 136 750 FCFA. Après apurement, la moyenne des dépenses alimentaires est de **47 961 FCFA** avec un écart-type de **29 201 FCFA**, et un maximum de **136 500 FCFA**. Ces résultats indiquent que la majorité des ménages ont des dépenses alimentaires modestes, tandis que quelques valeurs extrêmes supérieures à 136 750 FCFA ont été considérées comme aberrantes et retirées de l'analyse.

1.3.5. Apurement de SCA

1.4. Créer la variable vulnérabilité

Vulnérabilité = 0 si part.depalim= moins de 50% ou vulnérabilité=1 sinon

```
# Créer la variable vulnérabilité en fonction des classes de Part.DepAlim

df_final['vulnerabilite'] = df['Part.DepAlim'].apply(lambda x: 0 if x == 1
else 1)

# Vérifier la distribution
print(df['vulnerabilite'].value_counts())

sns.countplot(x='vulnerabilite', data=df, palette='Set2')
plt.title("Distribution de la variable vulnérabilité")
plt.show()
# Sauvegarder la base mise à jour
df_final.to_csv("Climat.SecAlim.2024.csv", index=False)
```

```
vulnerabilite
1    1068
0     103
Name: count, dtype: int64
```

La distribution de la variable **vulnérabilité** montre que **1068 ménages (environ 91%) sont considérés comme vulnérables**, tandis que seulement **103 ménages (environ 9%) ne le sont pas**. Cela traduit une prévalence élevée de la vulnérabilité alimentaire au sein de l'échantillon étudié, mettant en évidence un enjeu majeur de sécurité alimentaire dans la population analysée.

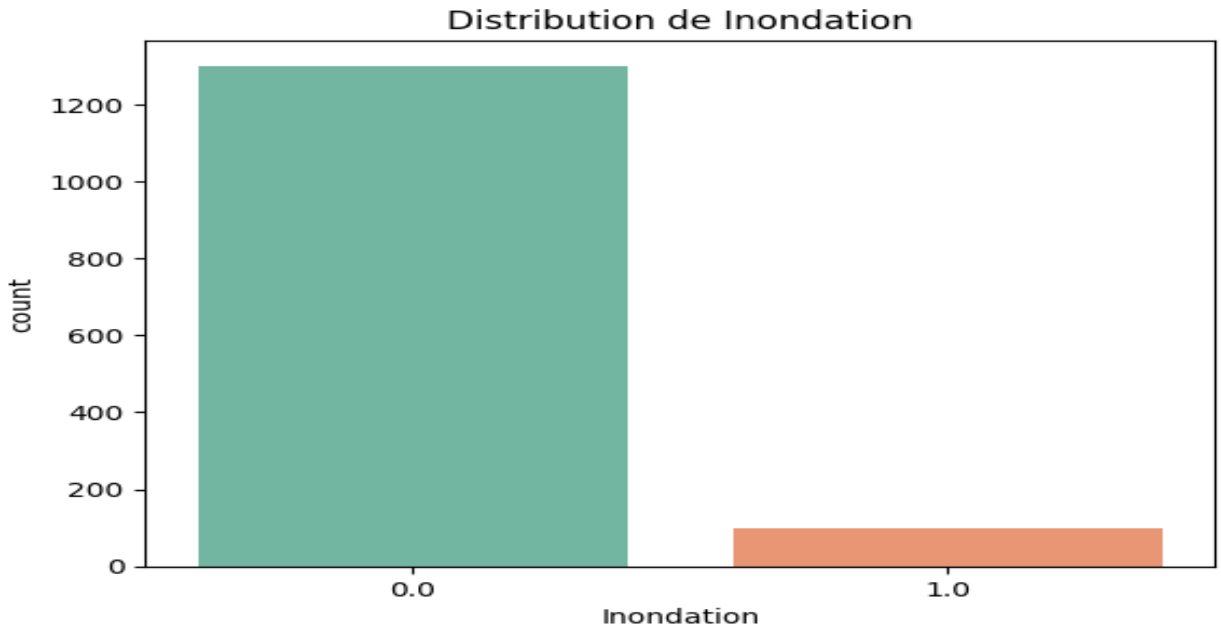
1.5. Décrire chacune des variables ci-dessous (avec leurs graphiques simples):

Inondation, pluies.insuffisante, pluies.horsaison, sci, sca, gca, revenu, depalim, part.depalim, vulnérabilité

```
# Liste des variables à décrire

vars_a_decrire = ['Inondation', 'Pluies.Insuf', 'Pluies.HorsSaison',
                  'sci', 'sca', 'Gca', 'revenu', 'Dep.Alim',
                  'Part.DepAlim', 'vulnerabilite']

for var in vars_a_decrire:
    print(f"Variable : {var}")

    # Affichage statistique général
    print(df[var].describe())
    print("\n")

    # Détection type de variable
    if df[var].nunique() <= 10:
        # Variable qualitative ou discrète ➔ bar plot
        sns.countplot(x=var, data=df, palette='Set2')
        plt.title(f"Distribution de {var}")
        plt.show()
    else:
        # Variable quantitative ➔ histogramme
        sns.histplot(df[var], kde=True, color='skyblue')
        plt.title(f"Histogramme de {var}")
        plt.show()
```

Variable Inondation

```
Description de la variable : Inondation
count    1171.000000
mean        0.064902
std         0.246458
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: Inondation, dtype: float64
```

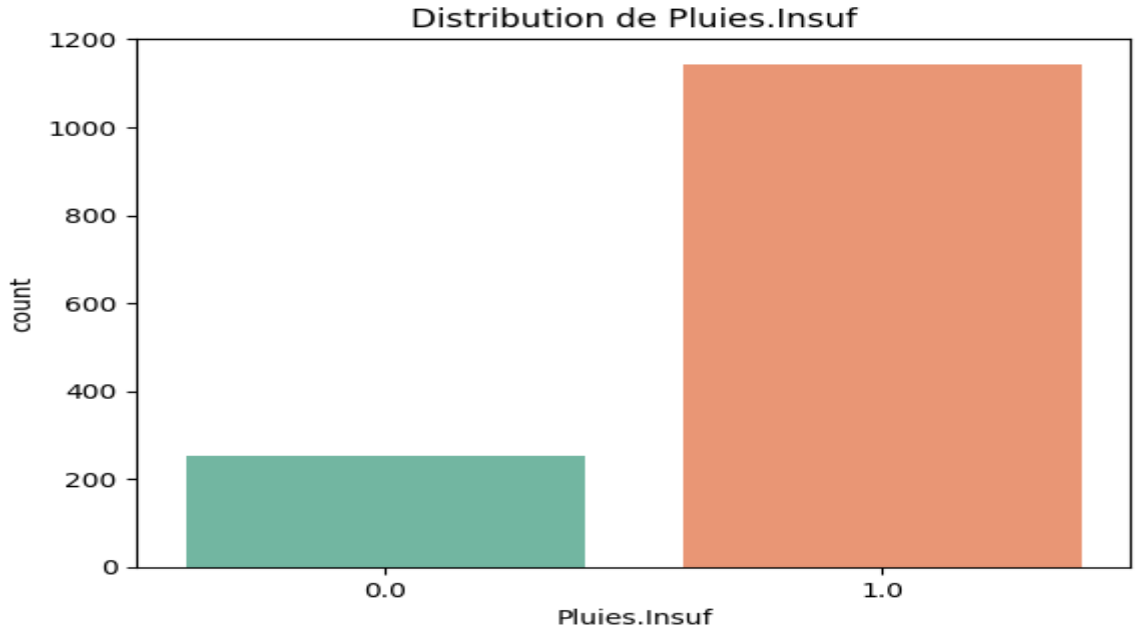


Le graphe montre une distribution très déséquilibrée de la variable *Inondation*. La grande majorité des ménages n'a pas été affectée (modalité 0) tandis qu'une proportion très faible déclare avoir subi une inondation (modalité 1). Cette forte asymétrie peut influencer les analyses statistiques et prédictives, notamment les modèles de classification, qui risquent de privilégier la classe majoritaire.

Variable : Pluies.Insuf

```
Description de la variable : Pluies.Insuf
count    1171.000000
mean        0.810418
std         0.392137
min         0.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         1.000000
Name: Pluies.Insuf, dtype: float64
```

La variable *Pluies.Insuf* est principalement composée de la modalité 1 (81%), indiquant que la majorité des ménages considèrent les pluies comme insuffisantes. La faible moyenne (0.81) et l'écart-type (0.39) confirment cette dominance. La distribution est donc fortement biaisée vers la perception d'insuffisance pluviométrique.

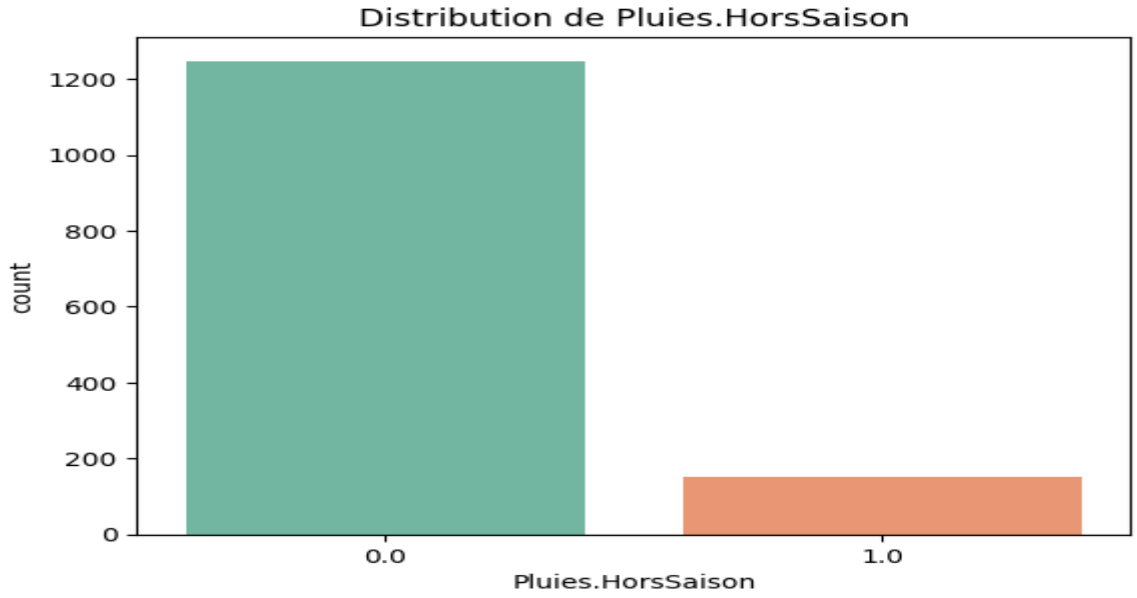


Le graphique montre que la majorité des ménages (environ 80%) considèrent les pluies comme insuffisantes (modalité 1), tandis qu'une minorité (environ 20%) ne partagent pas cet avis (modalité 0). Cette distribution confirme la description statistique, avec une prédominance nette de la perception d'insuffisance pluviométrique.

Variable : Pluies.HorsSaison

```
Description de la variable : Pluies.HorsSaison
count    1171.000000
mean        0.090521
std         0.287049
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         1.000000
Name: Pluies.HorsSaison, dtype: float64
```

La variable *Pluies.HorsSaison* indique que la grande majorité des ménages (environ 91%) n'ont pas connu de pluies hors saison, tandis qu'une minorité d'environ 9% en ont déclaré. La moyenne est de 0.0905 et l'écart-type est faible (0.287), traduisant une faible variabilité autour de la modalité dominante (0). La médiane ainsi que les quartiles (Q1 et Q3) sont tous égaux à 0, confirmant cette forte concentration des réponses sur l'absence de pluies hors saison. Cette variable présente donc une distribution très asymétrique, suggérant que ce phénomène reste rare au sein de la population étudiée.

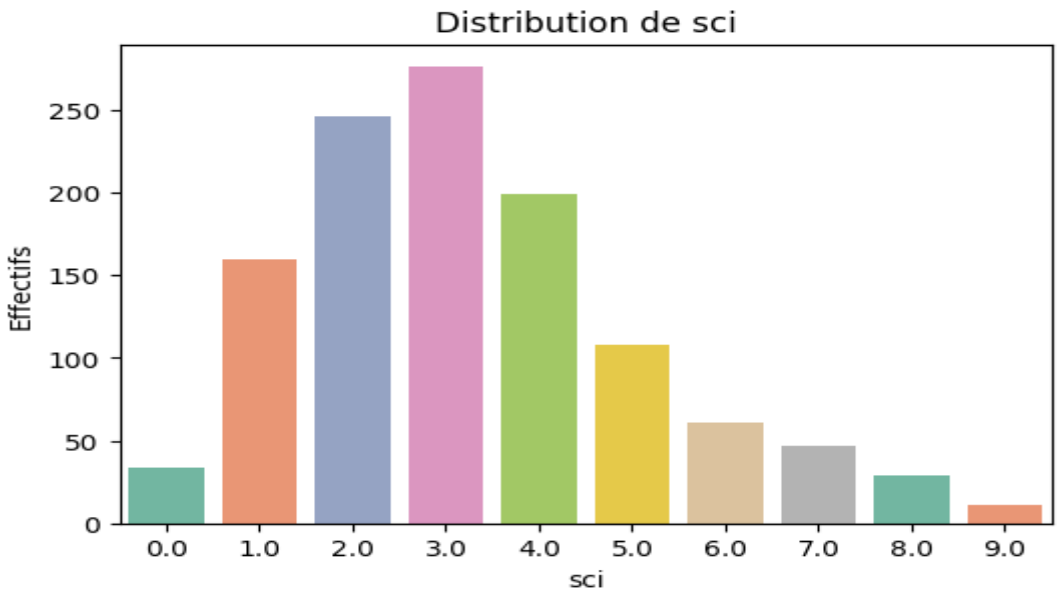


La distribution de la variable *Pluies.HorsSaison* montre une très forte prédominance de la modalité « 0 », indiquant l'absence de pluies hors saison pour la quasi-totalité des ménages enquêtés. La modalité « 1 » (présence de pluies hors saison) est très minoritaire, confirmant que ce phénomène est rare. Ce constat est cohérent avec la moyenne faible (≈9%) et la médiane nulle observées précédemment. Ainsi, la variable présente une forte asymétrie et reflète un déséquilibre marqué entre les deux modalités.

variable : sci

```
Description de la variable : sci

count    1171.000000
mean        3.280956
std         1.883932
min         0.000000
25%         2.000000
50%         3.000000
75%         4.000000
max         9.000000
Name: sci, dtype: float64
```

La variable *sci* présente une moyenne de 3.28 et un écart-type de 1.88, indiquant une dispersion modérée autour de la moyenne. La valeur minimale est 0 et la maximale 9, montrant une amplitude assez large. Le premier quartile est à 2, la médiane à 3 et le troisième quartile à 4, ce qui signifie que 50% des ménages ont un score *sci* compris entre 2 et 4. La distribution semble légèrement asymétrique à droite étant donné que la moyenne est supérieure à la médiane, traduisant l'existence de quelques valeurs élevées tirant la moyenne vers le haut.



Le graphique montre que la variable sci suit une **distribution asymétrique à droite**. On observe que la **modalité 3** est la plus fréquente avec près de **280 individus**, suivie des modalités 2 (environ 250) et 4 (près de 200). À partir du score 5, les effectifs diminuent progressivement, pour atteindre moins de 20 individus à la modalité 9.

Cette répartition indique que **la majorité des individus ont un score de consommation compris entre 2 et 4**, ce qui reflète des niveaux moyens de consommation alimentaire. Les scores plus élevés (6 et

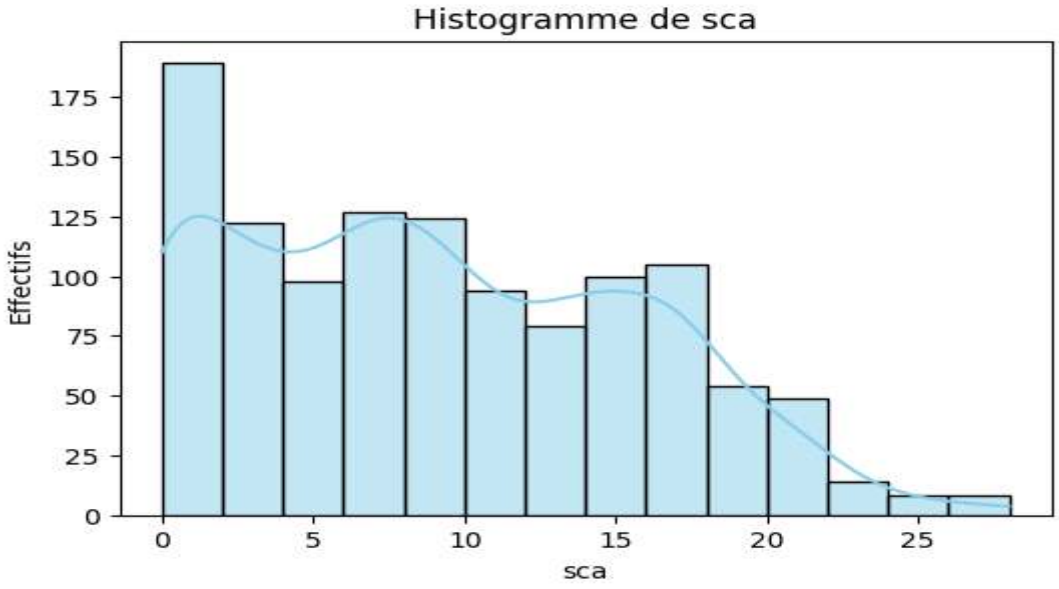
plus) sont nettement moins fréquents, ce qui suggère que **les comportements de consommation plus marqués sont minoritaires** dans la population étudiée.

Cette structure peut également indiquer une segmentation naturelle du groupe selon le score sci, et pourrait être pertinente pour des analyses par classe ou quartile.

variable : sca

```
Description de la variable : sca

count    1171.000000
mean        9.004270
std         6.590007
min         0.000000
25%         3.000000
50%         8.000000
75%        14.000000
max        28.000000
Name: sca, dtype: float64
```

La variable *sca* a une moyenne de 9.00 et un écart-type de 6.59, indiquant une grande dispersion des valeurs autour de la moyenne. La valeur minimale est 0 et la maximale 28, montrant une large amplitude. Le premier quartile est à 3, la médiane à 8 et le troisième quartile à 14, ce qui signifie que 50% des ménages ont un score *sca* compris entre 3 et 14. La moyenne étant légèrement supérieure à la médiane, la distribution est légèrement asymétrique à droite, traduisant la présence de valeurs élevées augmentant la moyenne.

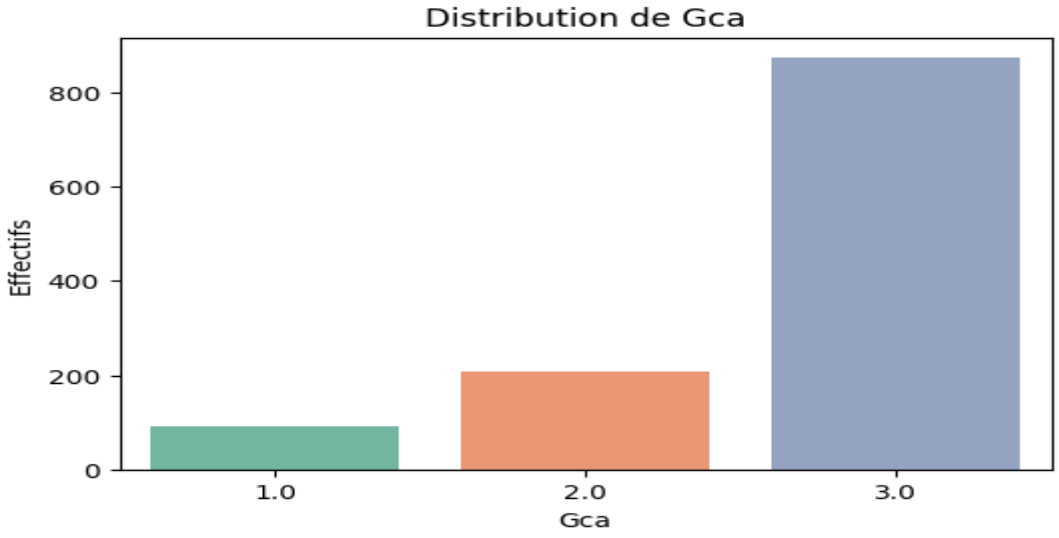


L'histogramme de sca présente une **distribution asymétrique à droite**, caractérisée par une forte concentration des effectifs entre **0 et 10**, avec un pic autour de **1 ou 2**. La densité diminue progressivement au-delà de 10, avec quelques observations extrêmes allant jusqu'à **27**. Cela indique qu'une majorité des ménages ou individus ont un **score de consommation faible à modéré**, tandis qu'une minorité atteinte des scores très élevés. La présence de la courbe de densité permet également de visualiser une tendance **globalement décroissante**, bien que quelques irrégularités apparaissent (reflétant une légère multimodalité).

Variable : Gca

```
Description de la variable : Gca
count    1171.000000
mean        2.668659
std         0.612854
min         1.000000
25%         2.000000
50%         3.000000
75%         3.000000
max         3.000000
Name: Gca, dtype: float64
```

La variable *Gca* présente une moyenne de 2.67 avec un écart-type de 0.61, indiquant une faible dispersion autour de la moyenne. La valeur minimale est 1 et la maximale 3, ce qui montre que *Gca* est une variable ordinale à trois modalités. Le premier quartile est à 2, la médiane et le troisième quartile sont à 3, suggérant que plus de la moitié des ménages sont classés dans la catégorie la plus élevée (3). Cette distribution est donc orientée vers les classes supérieures de *Gca*.

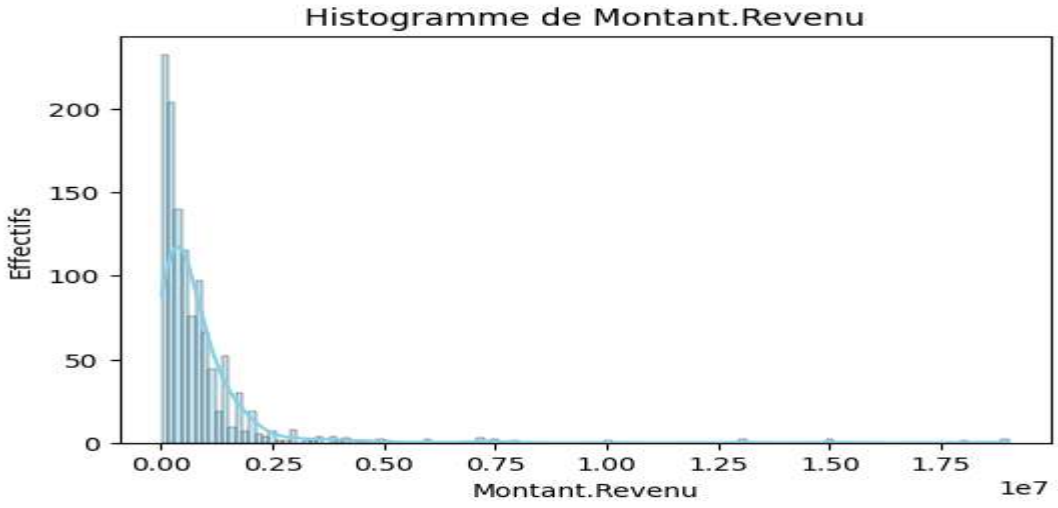


Le graphique met en évidence une **répartition très inégale** des modalités de la variable Gca. La catégorie **3.0** est largement majoritaire, regroupant **près des trois quarts des effectifs**. Elle est suivie par la catégorie **2.0**, puis très loin derrière par la catégorie **1.0**. Cette distribution suggère une forte **concentration des individus/observations dans une seule modalité**, ce qui pourrait refléter une classification déséquilibrée (par exemple, une majorité de ménages en sécurité alimentaire si Gca = 3 désigne cette situation).

Variable : revenu

```
Description de la variable : Montant.Revenu

count    1.171000e+03
mean     8.394630e+05
std      1.510395e+06
min      1.370000e+04
25%      2.000000e+05
50%      5.000000e+05
75%      1.000000e+06
max      1.900000e+07
Name: Montant.Revenu, dtype: float64
```

La variable **Revenu** représente les revenus des individus, exprimés en unités monétaires. Elle comporte **1 171 observations** avec une moyenne de **839 463**. Cependant, l'écart-type est élevé (**1 510 395**), ce qui indique une **forte dispersion** autour de la moyenne. La valeur minimale est de **13 700**, tandis que la valeur maximale atteint **19 000 000**, révélant la présence de revenus **extrêmement élevés** pour une minorité. La **médiane** est de **500 000**, inférieure à la moyenne, ce qui suggère une **asymétrie positive** (longue traîne à droite). 25 % des individus gagnent moins de **200 000**, et 75 % moins de **1 000 000**. La majorité des observations se concentrent donc sur des revenus faibles à moyens. Ce déséquilibre suggère la présence de valeurs extrêmes pouvant **influencer les analyses statistiques**.



L'histogramme du **Revenu** montre une distribution **fortement asymétrique à droite**, typique d'une loi log-normale. La majorité des observations sont concentrées sur des valeurs **faibles à modérées**, tandis qu'un petit nombre de cas présente des **revenus extrêmement élevés**, formant une **queue longue**. Cette dispersion importante indique la présence de **valeurs extrêmes ou atypiques (outliers)** qui pourraient influencer négativement certaines analyses statistiques classiques (comme la moyenne). Une transformation logarithmique ou un regroupement en classes peut être envisagé pour une meilleure interprétation ou modélisation.

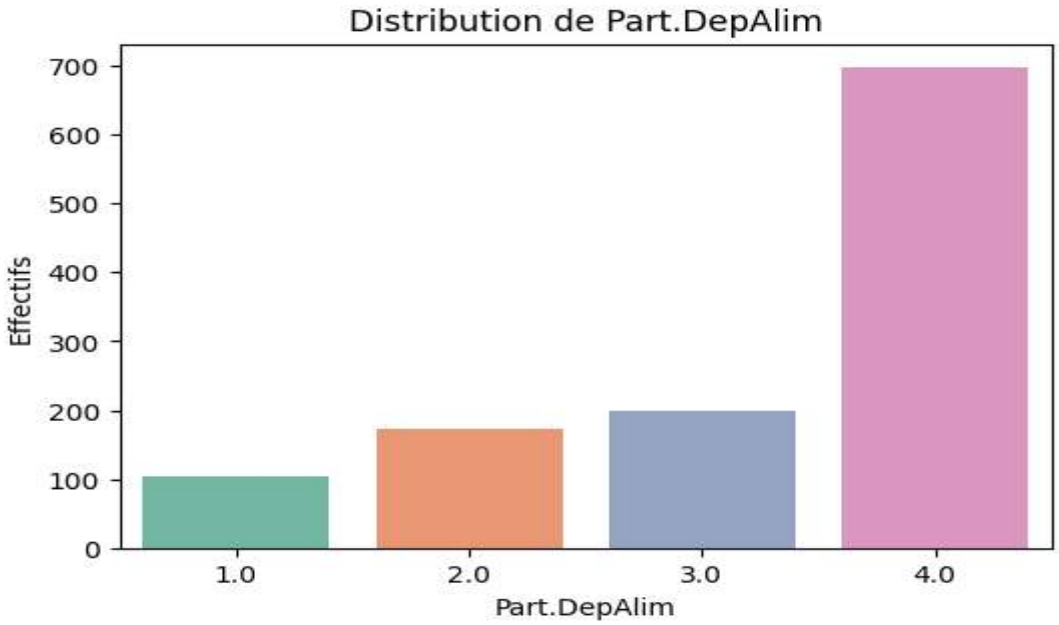
variable : Dep.Alim

```
📊 → Description de la variable : Dep.Alim

count    1.171000e+03
mean     3.240252e+05
std      7.358889e+06
min      1.250000e+04
25%      3.270000e+04
50%      4.950000e+04
75%      7.562500e+04
max      2.501065e+08
Name: Dep.Alim, dtype: float64
```

La variable **Dep.Alim** mesure les dépenses alimentaires des individus. Elle comporte **1 171 observations** avec une **moyenne de 324 025**, mais un **écart-type très élevé** de **7 358 889**, révélant une **variabilité extrême**. La dépense minimale est de **12 500**, tandis que la dépense maximale atteint **250 106 500**, ce qui indique la présence de **valeurs aberrantes ou extrêmes**. La **médiane** est de **49 500**, beaucoup plus faible que la moyenne, ce qui reflète une **distribution très asymétrique à droite**. Les trois quarts des individus dépensent moins de **75 625**, montrant que la **majorité a des dépenses modestes**, mais qu'une minorité affiche des niveaux de dépense exceptionnellement élevés.

variable : Part.DepAlim

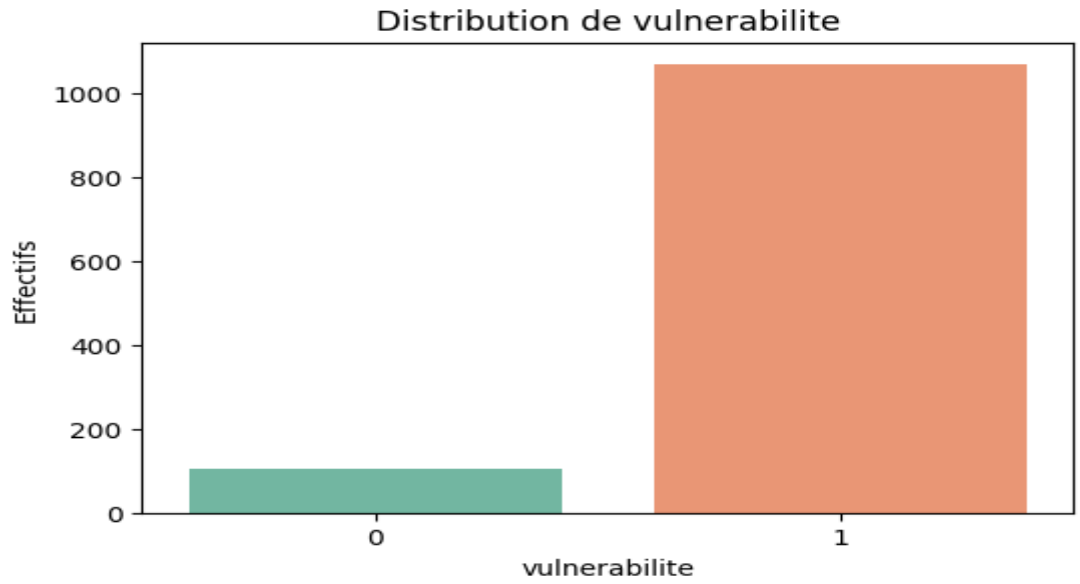
```
Description de la variable : Part.DepAlim

count    1171.000000
mean        3.271563
std         1.010899
min         1.000000
25%         3.000000
50%         4.000000
75%         4.000000
max         4.000000
Name: Part.DepAlim, dtype: float64
```



Variable : vulnerabilite

```
Description de la variable : vulnerabilite

count    1171.000000
mean        0.912041
std         0.283356
min         0.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         1.000000
Name: vulnerabilite, dtype: float64
```

La variable **vulnérabilité** est une variable **binaire** indiquant si un individu est vulnérable (1) ou non (0). Sur **1 171 observations**, la **moyenne de 0,91** montre que **91 % des individus sont considérés comme vulnérables**. La **médiane, les 1er et 3e quartiles** sont tous égaux à **1**, ce qui confirme une forte concentration des valeurs à ce niveau. L'écart-type est relativement faible (**0,28**), ce qui est typique pour une variable dichotomique déséquilibrée. La **valeur minimale est 0** (non vulnérable), mais très peu d'individus sont dans ce cas. Cette distribution fortement déséquilibrée pourrait avoir un **impact important dans les analyses statistiques** (notamment les modèles de classification), en raison de la **sous-représentation des non-vulnérables**.



La variable **vulnérabilité** est une variable **indicatrice (binaire)** codée 0 pour "non vulnérable" et 1 pour "vulnérable". Sur **1 171 individus**, la **moyenne est de 0,91**, ce qui signifie que **91 % de la population étudiée est considérée comme vulnérable**. Les 1er, 2e (médiane) et 3e quartiles sont tous égaux à **1**, ce qui confirme une forte concentration des observations dans la modalité « vulnérable ». L'écart-type de **0,28** reflète une **faible variabilité**, logique pour une variable déséquilibrée. Cette distribution indique un **déséquilibre important** entre les deux modalités, ce qui est crucial à considérer dans les modèles prédictifs ou d'analyse, notamment en classification.

1.6. Discrétiser chacun des variables ci-dessous en utilisant la règle de Sturge

sca, sci, revenu, depalim

Différencier les noms de ces variables de groupage (exemple la variable de groupage de sci = G.sci)

```
# Liste des variables à discrétiser

vars_discretiser = ['sca', 'sci', 'Montant.Revenu', 'Dep.Alim']

for var in vars_discretiser:
    # Nettoyage du nom de variable si besoin
    if var not in df_final.columns:
        print(f"Variable {var} non trouvée dans la base.")
        continue
    n = df_final[var].dropna().shape[0]
    k = int(1 + 3.322 * np.log10(n))  # Règle de Sturge

    group_var = 'G.' + var.split('.')[-1]  # Pour uniformiser le nom (ex :
Montant.Revenu -> G.Revenu)
    # Discrétisation
    df_final.loc[:, group_var] = pd.cut(df_final[var], bins=k,
labels=False)

    # Affichage synthétique
    print(f"✅ Variable : {var} | n = {n} | Nombre de classes (Sturge) =
{k}")
    print(df_final[group_var].value_counts().sort_index())
    print("\n")

# Renommer age et taille
df_final.rename(columns={'age': 'G.Age', 'taille': 'G.Taille'},
inplace=True)

# ✅ Sauvegarder la base mise à jour
df_final.to_csv("Climat.SecAlim.2024.csv", index=False)
print("💾 Base de données mise à jour et enregistrée sous le nom
'Climat.SecAlim.2024.csv'.")
```

```
✅ Variable : sca | n = 1171 | Nombre de classes (Sturge) = 11
G.sca
0     246
1     163
2     127
3     183
4      76
5     138
6     105
7      77
8      35
9      13
10      8
Name: count, dtype: int64
✅ Variable : sci | n = 1171 | Nombre de classes (Sturge) = 11
G.sci
0      34
1     160
2     246
3     276
4     199
6     108
7      61
...
Name: count, dtype: int64
💾 Base de données mise à jour et enregistrée sous le nom
'Climat.SecAlim.2024.csv'.
```

La variable sca a été regroupée en **11 classes** selon la règle de Sturges, sur **1 171 observations**. La distribution est **relativement étalée**, avec une fréquence maximale dans la classe 0 (**246 individus**), suivie de la classe 3 (**183**) et la classe 5 (**138**).
Les fréquences décroissent progressivement à partir de la classe 6, avec très peu d'observations dans les classes supérieures : **13 individus en classe 9** et **8 en classe 10**.

Cela traduit une **asymétrie positive** : la majorité des individus ont un score sca faible à moyen, tandis qu'une minorité se situe dans des niveaux de consommation très élevés.

La variable sci est également classée en **11 groupes**, avec une concentration très nette autour de la **classe 3 (276 individus)** et des classes adjacentes : **classe 2 (246)** et **classe 4 (199)**. Les classes extrêmes (notamment la classe 0 avec **34 individus** et la classe 7 avec **61**) sont beaucoup moins fréquentes.
Cette distribution est **fortement unimodale**, centrée autour d'un cœur de classes intermédiaires, ce qui suggère une plus grande **homogénéité individuelle** que pour sca.
L'absence ou la rareté de données dans les classes les plus élevées appuie l'idée que **peu d'individus présentent des niveaux extrêmes** de consommation individuelle.

1.7. Agréger puis extraire les fichiers liant les variables ci-dessous:

G.Taille, G.sci, revenu.moyen, sca.moyen, sci.moyen, depalim.moyenne (nom fichier= Aggrégation.G.Taille_gca) sexe, G.sci, revenu moyen, sca.moyen, sci.moyen, depalim.moyenne (nom fichier= Aggrégation.sexe_gca) Vulnérabilité, G.sci, revenu.moyen, sca.moyen, sci.moyen, depalim.moyenne (nom fichier= Aggrégation.Vulnérabilité__gca)

```
# ✅ Définir les agrégations demandées sous forme de fonctions groupby().agg()

## 🔷 1. Agrégation selon G.Taille et Gca
agg1 = df_final.groupby(['G.Taille', 'Gca']).agg(
    revenu_moyen=('Montant.Revenu', 'mean'),
    sca_moyen=('sca', 'mean'),
    sci_moyen=('sci', 'mean'),
    depalim_moyenne=('Dep.Alim', 'mean')
).reset_index()
# 🔷 Sauvegarde
agg1.to_csv("Aggrégation.G.Taille_gca.csv", index=False)
print("✅ Fichier 'Aggrégation.G.Taille_gca.csv' généré.")
## 🔷 2. Agrégation selon sexe et G.sci
agg2 = df_final.groupby(['sexe', 'G.sci']).agg(
    revenu_moyen=('Montant.Revenu', 'mean'),
    sca_moyen=('sca', 'mean'),
    sci_moyen=('sci', 'mean'),
    depalim_moyenne=('Dep.Alim', 'mean')
).reset_index()
# 🔷 Sauvegarde
agg2.to_csv("Aggrégation.sexe_G.sci.csv", index=False)
print("✅ Fichier 'Aggrégation.sexe_G.sci.csv' généré.")

## 🔷 3. Agrégation selon vulnérabilité et G.sci
agg3 = df_final.groupby(['vulnerabilite', 'G.sci']).agg(
    revenu_moyen=('Montant.Revenu', 'mean'),
    sca_moyen=('sca', 'mean'),
    sci_moyen=('sci', 'mean'),
    depalim_moyenne=('Dep.Alim', 'mean')
).reset_index()

# 🔷 Sauvegarde
agg3.to_csv("Aggrégation.vulnerabilite_G.sci.csv", index=False)
print("✅ Fichier 'Aggrégation.vulnerabilite_G.sci.csv' généré.")
```

```
✅ Fichier 'Aggrégation.G.Taille_gca.csv' généré.
✅ Fichier 'Aggrégation.sexe_G.sci.csv' généré.
✅ Fichier 'Aggrégation.vulnerabilite_G.sci.csv' généré.
```

Ce résultat indique nos trois fichiers ont été créé.

1.8. Sortir les tableaux croisés (en effectifs et %) et Graphiques liant l'ordre des variables ci-dessous:

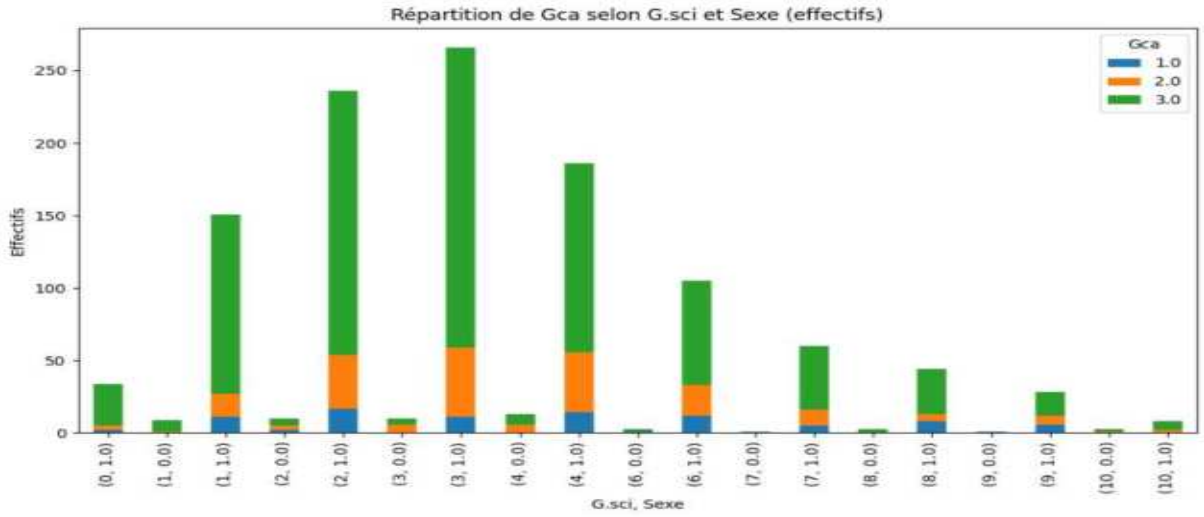
Répartition de gca selon G.sci et Sexe, Répartition de gca selon G.sci et G.Taille, Répartition de gca selon G.sci et Vulnérabilité

```
# 🔷 1. Répartition de Gca selon G.sci et Sexe

## 🔷 Tableau croisé (effectifs)
ct1 = pd.crosstab([df_final['G.sci'], df_final['sexe']], df_final['Gca'])
print("🔷 Tableau croisé (effectifs) Gca ~ G.sci & Sexe\n", ct1)

## 🔷 Tableau croisé (%)
ct1_pct = pd.crosstab([df_final['G.sci'], df_final['sexe']], df_final['Gca'],
normalize='index') * 100
print("🔷 Tableau croisé (%) Gca ~ G.sci & Sexe\n", ct1_pct.round(2))

## 🔷 Graphique
ct1.plot(kind='bar', stacked=True, figsize=(10,6))
plt.title("Répartition de Gca selon G.sci et Sexe (effectifs)")
plt.ylabel("Effectifs")
plt.xlabel("G.sci, Sexe")
plt.legend(title='Gca')
plt.tight_layout()
plt.show()
# 🔷 2. Répartition de Gca selon G.sci et G.Taille
## 🔷 Tableau croisé (effectifs)
ct2 = pd.crosstab([df_final['G.sci'], df_final['G.Taille']], df_final['Gca'])
print("🔷 Tableau croisé (effectifs) Gca ~ G.sci & G.Taille\n", ct2)
## 🔷 Tableau croisé (%)
ct2_pct = pd.crosstab([df_final['G.sci'], df_final['G.Taille']], df_final['Gca'],
normalize='index') * 100
print("🔷 Tableau croisé (%) Gca ~ G.sci & G.Taille\n", ct2_pct.round(2))
## 🔷 Graphique
ct2.plot(kind='bar', stacked=True, figsize=(10,6))
plt.title("Répartition de Gca selon G.sci et G.Taille (effectifs)")
plt.ylabel("Effectifs")
plt.xlabel("G.sci, G.Taille")
plt.legend(title='Gca')
plt.tight_layout()
plt.show()
# 🔷 3. Répartition de Gca selon G.sci et Vulnérabilité
## 🔷 Tableau croisé (effectifs)
ct3 = pd.crosstab([df_final['G.sci'], df_final['vulnerabilite']], df_final['Gca'])
print("🔷 Tableau croisé (effectifs) Gca ~ G.sci & Vulnérabilité\n", ct3)
## 🔷 Tableau croisé (%)
ct3_pct = pd.crosstab([df_final['G.sci'], df_final['vulnerabilite']], df_final['Gca'],
normalize='index') * 100
print("🔷 Tableau croisé (%) Gca ~ G.sci & Vulnérabilité\n", ct3_pct.round(2))

## 🔷 Graphique
ct3.plot(kind='bar', stacked=True, figsize=(10,6))
plt.title("Répartition de Gca selon G.sci et Vulnérabilité (effectifs)")
plt.ylabel("Effectifs")
plt.xlabel("G.sci, Vulnérabilité")
plt.legend(title='Gca')
plt.tight_layout()
plt.show()
```

```
◈ Tableau croisé (effectifs) Gca ~ G.sci & Sexe
 Gca         1.0  2.0  3.0
G.sci sexe
0     1.0     2    3   29
1     0.0     0    1    8
      1.0    11   16  124
2     0.0     2    3    5
      1.0    17   37  182
3     0.0     0    6    4
      1.0    11   48  207
4     0.0     0    6    7
      1.0    14   42  130
6     0.0     1    0    2
      1.0    12   21   72
7     0.0     0    0    1
      1.0     5   11   44
8     0.0     0    0    3
      1.0     8    5   31
9     0.0     1    0    0
      1.0     6    6   16
10    0.0     0    1    2
      1.0     0    2    6
◈ Tableau croisé (%) Gca ~ G.sci & Sexe
 Gca            1.0    2.0     3.0
...
9     0.0   100.00   0.00    0.00
      1.0    21.43  21.43   57.14
10    0.0     0.00  33.33   66.67
      1.0     0.00  25.00   75.00
```

Ce tableau croisé met en évidence la répartition des individus selon leur **sexe**, leur **niveau individuel de consommation (G.sci)**, et leur **groupe global de consommation alimentaire (Gca)**.

- Dans presque toutes les classes de G.sci, **les femmes (sexe = 1.0)** sont **largement majoritaires**, surtout dans la **catégorie Gca = 3** (niveau de consommation le plus élevé). Par exemple, dans G.sci = 3, **207 femmes** appartiennent à Gca = 3, contre seulement **4 hommes**.
- À l'inverse, les **hommes (sexe = 0.0)** sont très peu représentés dans les groupes de consommation élevés (Gca = 3) et sont plus souvent associés aux groupes intermédiaires ou faibles (Gca = 1 ou 2), bien que leurs effectifs soient globalement faibles dans toutes les classes.
- Les **classes de G.sci les plus basses** (comme 0 ou 1) comptent **plus d'individus dans Gca = 1**, ce qui semble logique : les faibles scores individuels de consommation s'alignent avec un faible niveau global de consommation.
- En pourcentage, on observe que **plus le score G.sci augmente**, **plus la part des individus appartenant à Gca = 3 augmente**. Par exemple :
    - Pour les femmes de G.sci = 3 : **~78 % sont dans Gca = 3**.
    - Pour les femmes de G.sci = 7 : **~76 % dans Gca = 3**, contre **~18 % dans Gca = 2**.
- Ce croisement montre donc une **cohérence globale entre les niveaux individuel et global de consommation alimentaire**, tout en soulignant une **forte prédominance féminine**, surtout dans les classes les plus favorables.



Ce graphique représente la répartition des effectifs selon la variable **Gca** en fonction du **G.sci** et du **sexe**. On observe une prédominance de la modalité **3.0 de Gca** (en vert) dans presque toutes les combinaisons de G.sci et de sexe. Les effectifs sont particulièrement élevés chez les individus du groupe scientifique 3, sexe 1 (probablement des hommes), suivis de près par ceux des groupes 2 et 4 avec le même sexe. Globalement, les hommes semblent plus représentés que les femmes, surtout dans les groupes scientifiques centraux (2 à 4). Les groupes scientifiques extrêmes (0, 9, 10) affichent des effectifs nettement plus faibles.

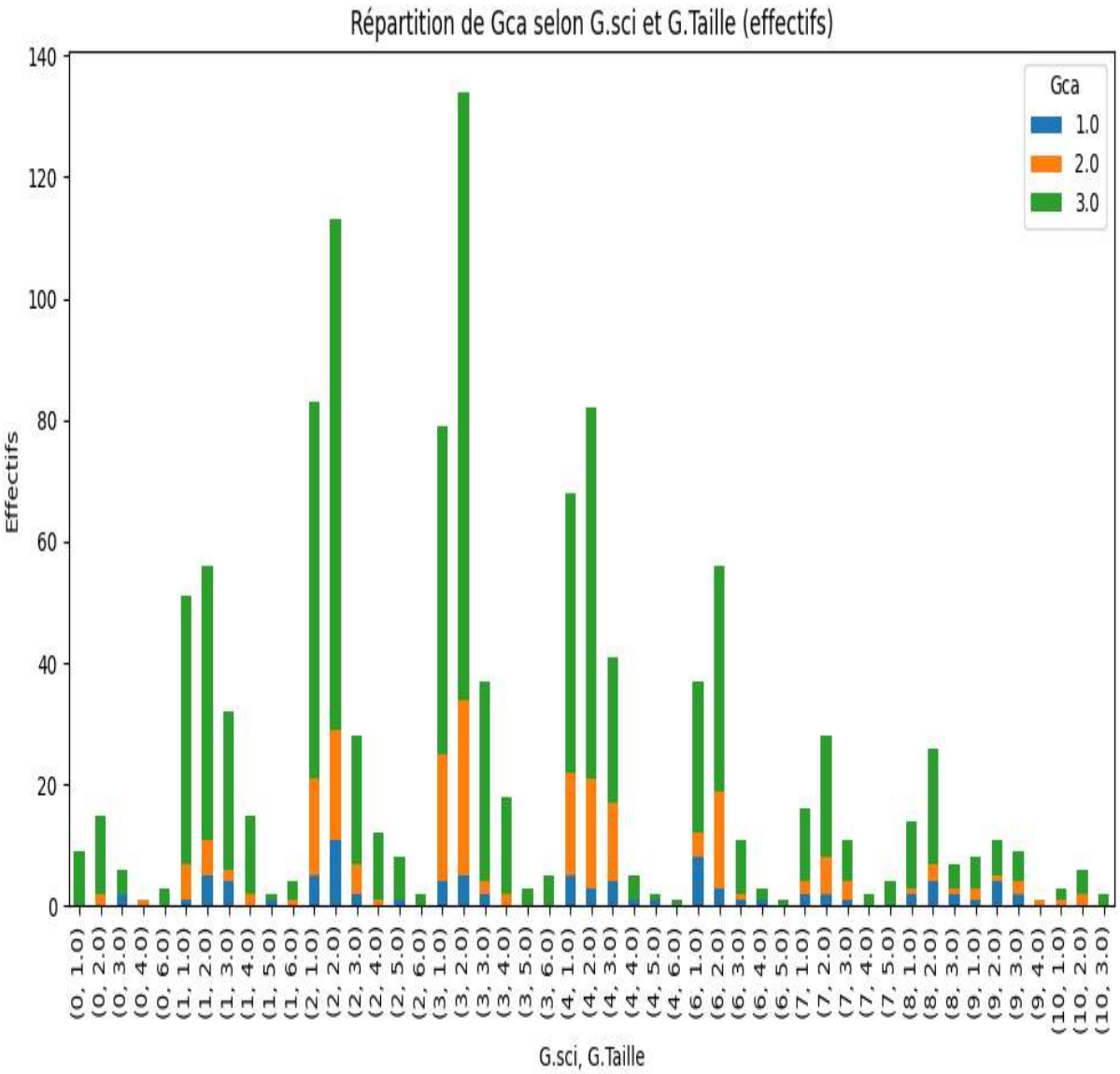
```
🔷 Tableau croisé (effectifs) Gca ~ G.sci & G.Taille
 Gca             1.0  2.0  3.0
G.sci G.Taille
0     1.0         0    0    9
      2.0         0    2   13
      3.0         2    0    4
      4.0         0    1    0
      6.0         0    0    3
1     1.0         1    6   44
      2.0         5    6   45
      3.0         4    2   26
      4.0         0    2   13
      5.0         1    0    1
      6.0         0    1    3
2     1.0         5   16   62
      2.0        11   18   84
      3.0         2    5   21
      4.0         0    1   11
      5.0         1    0    7
      6.0         0    0    2
3     1.0         4   21   54
      2.0         5   29  100
      3.0         2    2   33
      4.0         0    2   16
      5.0         0    0    3
...
```

Ce tableau met en relation les **niveaux de consommation alimentaire globaux (Gca)**, les **niveaux individuels (G.sci)**, et la **taille des ménages**, groupée en classes (G.Taille).

- Globalement, on observe que **les individus appartenant à des ménages plus petits (tailles 1 à 2)** sont **bien représentés dans les groupes Gca = 3** (forte consommation), surtout à partir de G.sci=2.

  Par exemple, dans G.sci = 2, on trouve **62 individus en G.Taille = 1.0** dans Gca = 3 et **84 en G.Taille = 2.0**.
- **Les ménages de taille moyenne (3 à 4)** montrent aussi une tendance à se concentrer dans Gca = 3, en particulier à partir de G.sci = 3. Cela suggère qu'**une consommation individuelle moyenne ou élevée est liée à une consommation globale plus importante**, indépendamment de la taille du ménage.

- Pour les **plus grandes tailles de ménage (5 ou 6)**, les effectifs sont plus faibles mais restent majoritairement situés dans Gca = 3, sauf pour quelques cas isolés en Gca = 1 ou 2. Cela pourrait indiquer que **les ménages nombreux atteignent également de hauts niveaux de consommation**, mais sont moins nombreux dans l'échantillon.
- Enfin, les **classes les plus faibles de G.sci (0 et 1)** sont plus dispersées entre les groupes Gca, avec une légère surreprésentation dans Gca = 2. Cela reflète une certaine hétérogénéité parmi les individus à faible consommation individuelle, possiblement liée à des **effets de structure familiale ou de partage intra-ménage**.



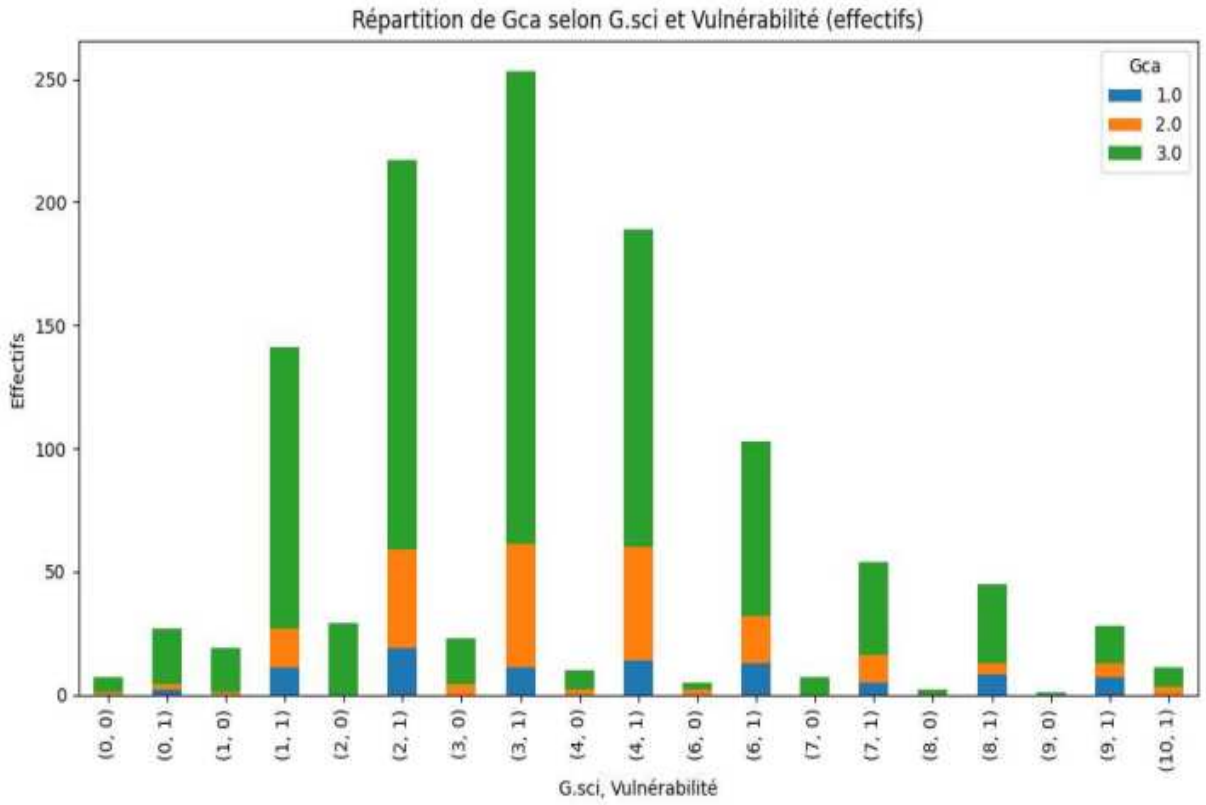
Ce graphique montre la répartition des effectifs selon la variable **Gca**, en fonction du **G.sci** et de la **taille**. La modalité **3.0 de Gca** (en vert) domine largement dans presque toutes les combinaisons, confirmant une tendance générale déjà observée précédemment. Les effectifs les plus importants sont enregistrés dans les groupes scientifiques **2**, **3** et **4**, avec des tailles moyennes à grandes (notamment tailles 5.0 et

6.0). À l'inverse, les extrêmes en taille et en groupe scientifique affichent de faibles effectifs. On note donc une surreprésentation des individus de grande taille dans certains groupes scientifiques spécifiques, associés à des niveaux élevés de Gca.

```
◈ Tableau croisé (effectifs) Gca ~ G.sci & Vulnérabilité
 Gca                  1.0  2.0  3.0
G.sci vulnerabilite
0     0                0    1    6
      1                2    2   23
1     0                0    1   18
      1               11   16  114
2     0                0    0   29
      1               19   40  158
3     0                0    4   19
      1               11   50  192
4     0                0    2    8
      1               14   46  129
6     0                0    2    3
      1               13   19   71
7     0                0    0    7
      1                5   11   38
8     0                0    0    2
      1                8    5   32
9     0                0    0    1
      1                7    6   15
10    1                0    3    8
◈ Tableau croisé (%) Gca ~ G.sci & Vulnérabilité
 Gca                    1.0    2.0     3.0
G.sci vulnerabilite
...
      1              17.78  11.11   71.11
9     0               0.00   0.00  100.00
      1              25.00  21.43   53.57
10    1               0.00  27.27   72.73
```

Ce tableau met en évidence la relation entre **la vulnérabilité des individus**, leur **niveau de consommation individuelle (G.sci)** et leur **groupe de consommation alimentaire (Gca)**.

- On constate que les **individus vulnérables (vulnérabilité = 1)** sont **largement représentés dans le groupe Gca = 3**, et ce, quel que soit leur niveau de G.sci. Par exemple, parmi les vulnérables ayant un G.sci = 3, **192 individus sont dans Gca = 3**, contre seulement **19 parmi les non vulnérables**.
- À l'inverse, les **non-vulnérables (vulnérabilité = 0)** sont **peu nombreux** dans l'échantillon et se concentrent en majorité dans Gca = 3, mais avec **des effectifs beaucoup plus faibles**. Par exemple, seulement **29 non-vulnérables** dans G.sci = 2 sont en Gca = 3, contre **158 vulnérables**.
- En pourcentage, même les **non-vulnérables** se retrouvent majoritairement en Gca = 3 à mesure que le G.sci augmente. Ainsi, la consommation globale semble **plus influencée par le niveau de consommation individuelle** que par le statut de vulnérabilité.
- Cependant, chez les **vulnérables**, on observe une **distribution plus marquée vers les groupes de consommation élevés**, ce qui peut paraître contre-intuitif. Cela pourrait s'expliquer par le fait que certains ménages vulnérables maintiennent un niveau de consommation relativement élevé (par choix de priorités, transferts, ou autres mécanismes de résilience).



Ce graphique présente la répartition des effectifs selon **Gca**, en fonction du **G.sci** et de la **vulnérabilité**. Comme dans les précédents graphiques, la modalité **3.0 de Gca** (en vert) reste dominante dans presque toutes les combinaisons. Les groupes scientifiques **2**, **3** et **4** concentrent les effectifs les plus élevés, en

particulier parmi les individus non vulnérables (vulnérabilité = 0). On note toutefois une présence non négligeable d'individus vulnérables dans ces mêmes groupes, mais en effectif moindre. Globalement, les individus non vulnérables sont plus représentés dans toutes les catégories, et ceux appartenant aux groupes scientifiques centraux semblent les plus nombreux à afficher un niveau élevé de Gca.

2. DATA MINING : Base de données ‘’*Climat.SécAlim.2024* ‘’ (20 POINTS)

2.1 Procéder aux tests d'Ajustement a une loi de probabilité pour chacune des variables ci-dessous:

a) Test d’ajustement à une loi binomiale pour la variable ‘vulnérabilité’

```
# 🔷 Variable vulnérabilité
vul = df['vulnerabilite']
# 🔷 Comptage des 0 et 1 (si binaire)
counts = vul.value_counts()
n = len(vul)
p = counts.get(1,0) / n
# 🔷 Test du khi deux d'ajustement à la loi binomiale
# Distribution théorique
binom_theo = stats.binom.pmf([0,1], n=1, p=p) * n
# Test du khi2
chi2, p_value = stats.chisquare(f_obs=counts, f_exp=binom_theo)
print("🔷 Test Binomial vulnérabilité")
print(f"Chi2 = {chi2:.3f}, p-value = {p_value:.3f}")
```

```
🔷 Test Binomial vulnérabilité
Chi2 = 9912.953, p-value = 0.000
```

Le test d’ajustement donne Chi² = 9912.953, p-value = 0.000. Comme la p-value est inférieure à 0.05, on rejette l’hypothèse que vulnérabilité suit une loi binomiale. Conclusion : la distribution observée diffère significativement de la loi binomiale attendue. Cela indique que la variable vulnérabilité n’est pas conforme à cette loi dans la population étudiée.

b) khi deux: gca

```
# 🔷 Variable Gca
gca = df['Gca']
counts_gca = gca.value_counts().sort_index()
# 🔷 Hypothèse : répartition uniforme entre les k modalités
k = len(counts_gca)
n = len(gca)
uniform_theo = [n / k] * k
# 🔷 Test du khi²
chi2, p_value = stats.chisquare(f_obs=counts_gca, f_exp=uniform_theo)
# 🔷 Résultats
print("🔷 Test Khi² Gca")
print(f"Chi² = {chi2:.3f}, p-value = {p_value:.3f}")
```

Le test donne **Chi² = 913.098, p-value = 0.000**.

Comme la p-value est inférieure à 0.05, **on rejette l'hypothèse d'une distribution uniforme** de Gca. **Conclusion :** la variable Gca **n'est pas répartie de façon homogène** entre ses différentes modalités. Elle présente donc des **disparités significatives** dans la population étudiée.

c) normalité: revenu

```
# 🔷 Variable revenu
revenu = df['Montant.Revenu'].dropna()
# 🔷 Test de Shapiro-Wilk
shapiro = stats.shapiro(revenu)
# 🔷 Résultats
print("🔷 Test de normalité (Shapiro-Wilk) Revenu")
print(f"Statistic = {shapiro.statistic:.3f}, p-value =
{shapiro.pvalue:.3f}")
```

```
🔷 Test de normalité (Shapiro-Wilk) Revenu
Statistic = 0.410, p-value = 0.000
```

Le test de normalité de Shapiro-Wilk appliqué à la variable revenue donne une statistique de 0.410 avec une p-value égale à 0.000. Étant donné que la p-value est inférieure au seuil de 0.05, l'hypothèse selon laquelle la variable suit une loi normale est rejetée. Cela signifie que la distribution des revenus dans la population étudiée diffère significativement d'une distribution normale, ce qui peut influencer le choix des méthodes statistiques appropriées pour l'analyse.

d) t-student: si revenu et dépense suivent une loi normale) le cas suivant: revenu moyen = 120 000 e) poisson: sci

```
df = pd.read_csv("Climat.SecAlim.2024.csv", sep=';')


revenu = df['Montant.Revenu'].dropna()

# Test t-student unilatéral sur la moyenne (H0: mu=120000)
t_stat, p_value = stats.ttest_1samp(revenu, 120000)

print(f"T-statistique = {t_stat:.3f}, p-value = {p_value:.3f}")
```

T-statistique = 16.300, p-value = 0.000

Le test t de Student réalisé sur la variable revenue donne une statistique t de 16.300 avec une p-value de 0.000. Étant donné que la p-value est inférieure à 0.05, on rejette l'hypothèse selon laquelle le revenu moyen serait égal à 120 000. Ainsi, la moyenne observée du revenu dans la population diffère significativement de cette valeur de référence, indiquant un écart réel et statistiquement significatif par rapport au revenu moyen hypothétique de 120 000.

```
df = pd.read_csv("Climat.SecAlim.2024.csv", sep=';')

# 🔷 Variable sci
sci = df['sci'].dropna()
# 🔷 Paramètre lambda estimé
lambda_poisson = sci.mean()
# 🔷 Effectifs observés
counts = sci.value_counts().sort_index()
valeurs = counts.index.values
# 🔷 Effectifs théoriques (probabilité * total effectifs)
effectifs_theo = np.array([stats.poisson.pmf(k, mu=lambda_poisson) *
len(sci) for k in valeurs])
# 🔷 Ajustement final : force la somme exacte
effectifs_theo = effectifs_theo * (counts.sum() / effectifs_theo.sum())
# 🔷 Vérifier égalité des sommes
print(f"Somme f_obs = {counts.sum()}, Somme f_exp =
{effectifs_theo.sum()}")
# 🔷 Test khi²
chi2, p_value_poisson = stats.chisquare(f_obs=counts, f_exp=effectifs_theo)

print(f"Test Poisson sci: Chi2 = {chi2:.3f}, p-value =
{p_value_poisson:.3f}")
```

Somme f_obs = 1171, Somme f_exp = 1171.0
Test Poisson sci: Chi2 = 39.890, p-value = 0.000

Le test d'ajustement à la loi de Poisson appliqué à la variable sci donne une statistique du khi-deux de 39.890 avec une p-value de 0.000. Comme la p-value est inférieure à 0.05, on rejette l'hypothèse selon laquelle sci suit une loi de Poisson. Ainsi, la distribution observée de sci diffère significativement de celle attendue sous un modèle de Poisson, indiquant que ce modèle ne décrit pas adéquatement la variable dans la population étudiée.

2.2 Procéder à toutes les tests de relations entre les variables ci-dessous

a) khi deux: gca, vulnérabilité

```
df = pd.read_csv("Climat.SecAlim.2024.csv", sep=';')

# Tableau croisé
table = pd.crosstab(df['Gca'], df['vulnerabilite'])
# Test khi-deux
chi2, p_value, dof, expected = stats.chi2_contingency(table)
print(f"Khi2 gca-vulnérabilité: Chi2 = {chi2:.3f}, p-value =
{p_value:.3f}")
```

Khi2 gca-vulnérabilité: Chi2 = 16.557, p-value = 0.000

Le test du khi-deux réalisé entre les variables Gca et vulnérabilité donne une statistique de 16.557 avec une p-value égale à 0.000. Comme la p-value est inférieure à 0.05, l'hypothèse d'indépendance entre Gca et vulnérabilité est rejetée. Cela indique qu'il existe une association significative entre les deux variables, la distribution de Gca variant selon le statut de vulnérabilité des individus dans la population étudiée.

b) t-student (facteur, variable): vulnérabilité, sci

```
group0 = df[df['vulnerabilite'] == 0]['sci'].dropna()
group1 = df[df['vulnerabilite'] == 1]['sci'].dropna()
# Test t-student
t_stat, p_value = stats.ttest_ind(group0, group1, equal_var=False)
print(f"T-student vulnérabilité-sci: t = {t_stat:.3f}, p-value =
{p_value:.3f}")
```

T-student vulnérabilité-sci: t = -3.714, p-value = 0.000

Le test t de Student réalisé pour comparer la variable sci selon le statut de vulnérabilité donne une statistique t de -3.714 avec une p-value égale à 0.000. Comme la p-value est inférieure à 0.05, on conclut qu'il existe une différence significative des moyennes de sci entre les groupes vulnérables et non vulnérables. Ainsi, la vulnérabilité est associée à une variation significative de la valeur moyenne de sci dans la population étudiée.

c) anova (facteur, variable): part.depalim, sca

```
df = df.rename(columns={'Part.DepAlim': 'Part_DepAlim'})

model = ols('sca ~ C(Part_DepAlim)', data=df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print(anova_table)
```

	sum_sq	df	F	PR(>F)

```
C(Part_DepAlim)   2545.622260     3.0  20.516725  5.856539e-13
Residual         48265.356391  1167.0       NaN          NaN
```

Le résultat de l'ANOVA indique que la variable catégorielle Part_DepAlim a un effet significatif sur la variable sca, avec une statistique F de 20.52 et une p-value très faible (5.86e-13), bien inférieure au seuil de 0.05. Cela signifie que les moyennes de sca diffèrent significativement entre les groupes définis par Part_DepAlim dans la population étudiée. L'hypothèse nulle d'égalité des moyennes est donc rejetée.

d) ancova (facteur, variable): part.depalim, sca si covariable = taille

```
df = df.rename(columns={'Part.DepAlim': 'Part_DepAlim', 'G.Taille':
'G_Taille'})

model = ols('sca ~ C(Part_DepAlim) + G_Taille', data=df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print(anova_table)
```

```
                  sum_sq      df         F        PR(>F)
C(Part_DepAlim)  2444.917312     3.0  19.713744  1.808424e-12
G_Taille           62.546137     1.0   1.512957  2.189365e-01
Residual        48202.810253  1166.0       NaN          NaN
```

Les résultats de l'ANCOVA montrent que le facteur Part_DepAlim a un effet hautement significatif sur la variable sca (F = 19.71, p-value ≈ 1.8e-12), indiquant que les moyennes de sca diffèrent significativement selon les groupes de Part_DepAlim, même après ajustement. En revanche, la covariable G_Taille n'a pas d'effet significatif (F = 1.51, p-value = 0.22), ce qui suggère que la taille ne contribue pas de manière significative à expliquer la variance de sca dans ce modèle.

e) manova:(facteur, variables): G.sci, revenu, sca

```
df = df.rename(columns={'Montant.Revenu': 'Montant_Revenu', 'G.sci':
'G_sci'})

from statsmodels.multivariate.manova import MANOVA
maov = MANOVA.from_formula('Montant_Revenu + sca ~ C(G_sci)', data=df)
print(maov.mv_test())
```

```
                  Multivariate linear model
================================================================

----------------------------------------------------------------
       Intercept         Value  Num DF  Den DF  F Value Pr > F
----------------------------------------------------------------
          Wilks' lambda 0.9026 2.0000 1160.0000 62.5975 0.0000
         Pillai's trace 0.0974 2.0000 1160.0000 62.5975 0.0000
 Hotelling-Lawley trace 0.1079 2.0000 1160.0000 62.5975 0.0000
    Roy's greatest root 0.1079 2.0000 1160.0000 62.5975 0.0000
----------------------------------------------------------------

----------------------------------------------------------------
       C(G_sci)          Value  Num DF  Den DF  F Value Pr > F
----------------------------------------------------------------
```

```
          Wilks' lambda 0.7999 18.0000 2320.0000 15.2217 0.0000
         Pillai's trace 0.2068 18.0000 2322.0000 14.8767 0.0000
 Hotelling-Lawley trace 0.2418 18.0000 1929.5591 15.5697 0.0000
    Roy's greatest root 0.1998  9.0000 1161.0000 25.7768 0.0000
==========================================================================
```

Le test MANOVA indique que le facteur **G_sci** a un effet multivarié très significatif sur les variables dépendantes **Montant_Revenu** et **sca**.

Les différentes statistiques globales (Wilks' lambda, Pillai's trace, Hotelling-Lawley trace, Roy's greatest root) donnent toutes des valeurs de p proches de 0 (inférieures à 0.001), ce qui confirme que la variation combinée des deux variables dépendantes est associée au groupe G_sci.

En résumé, on peut conclure que les niveaux de G_sci expliquent significativement la variation conjointe du revenu et de sca dans la population étudiée.

f) mancova:(facteur, variables): G.sci, revenu, sca si covariable = taille

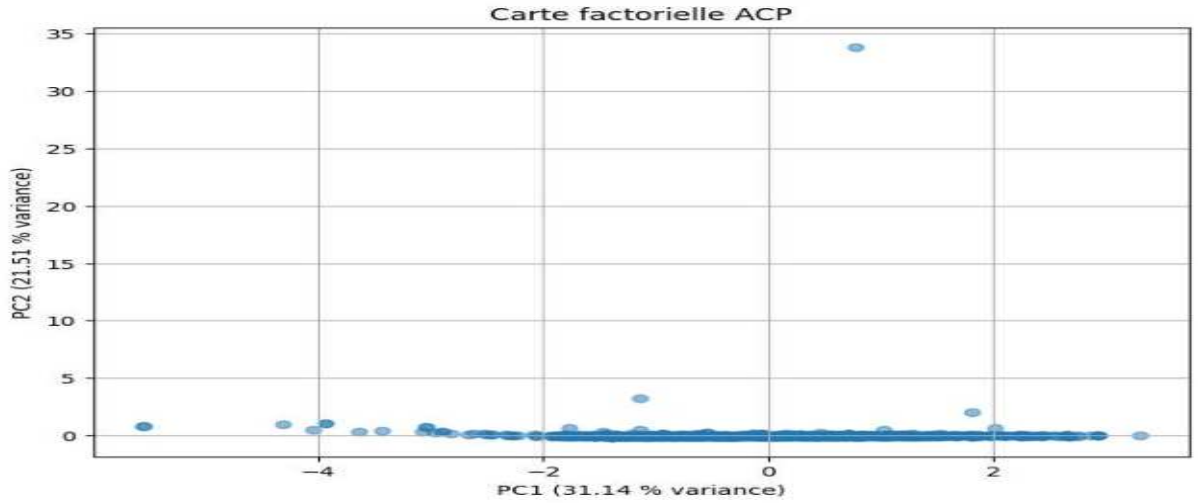
```
# Renommer les colonnes avec des noms sans points

df = df.rename(columns={
    'Montant.Revenu': 'Montant_Revenu',
    'G.sci': 'G_sci',
    'G.Taille': 'G_Taille'
})
from statsmodels.multivariate.manova import MANOVA
# Formule corrigée
maov = MANOVA.from_formula('Montant_Revenu + sca ~ C(G_sci) + G_Taille',
data=df)
print(maov.mv_test())
```

Le test MANOVA montre que les variables explicatives **G_sci** (catégorielle) et **G_Taille** (continue) ont toutes deux un effet significatif sur la variation conjointe des variables dépendantes Montant_Revenu et sca. L'effet de **G_sci** est particulièrement fort, indiquant que les groupes définis par cette variable ont des moyennes multivariées différentes. **G_Taille** a aussi un impact significatif, mais plus modéré. Ces résultats signifient que ces deux facteurs expliquent ensemble une part importante de la variance dans les variables étudiées, ce qui confirme leur rôle dans la dynamique des données.

2.3 Procéder à la réduction des dimensions utilisant l'ACP

(Il faut transformer les modalités des variables qualitatives ou binaires, dont les variables de groupage, en de nouvelles variables binaires. Exemple pour gca, créer les variables pauvre, modéré, acceptable ayant chacune pour modalité 1 ou 0, respectant leur choix). Analyser la carte factorielle.

```
# 2. Variables qualitatives à encoder (exemple : 'Gca', 'sexe',
'vulnerabilite', etc.)
# Ici on crée des variables indicatrices pour Gca : pauvre, modéré,
acceptable (exemple)
df = pd.get_dummies(df, columns=['Gca', 'sexe', 'vulnerabilite'],
drop_first=False)
# 3. Sélection des variables quantitatives pour l'ACP
quant_vars = ['Montant.Revenu', 'sca', 'sci', 'Dep.Alim']  # adapter selon
variables présentes
# 4. Standardisation
scaler = StandardScaler()
X_quant = scaler.fit_transform(df[quant_vars])
# 5. Concaténer les variables binaires et standardisées
X = pd.concat([pd.DataFrame(X_quant, columns=quant_vars),
df.filter(regex='^Gca_|^sexe_|^vulnerabilite_')], axis=1)
# 6. ACP
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
# 7. Visualisation de la carte factorielle
plt.figure(figsize=(8,6))
plt.scatter(X_pca[:,0], X_pca[:,1], alpha=0.5)
plt.xlabel(f"PC1 ({pca.explained_variance_ratio_[0]*100:.2f} % variance)")
plt.ylabel(f"PC2 ({pca.explained_variance_ratio_[1]*100:.2f} % variance)")
plt.title("Carte factorielle ACP")
plt.grid(True)
plt.show()
```



2.4 Apprentissage automatique non supervisée

a) Procéder à la classification des ménages, par la méthode Nuée Dynamique (Kmean), pour un regroupement en 2 classes avec les variables suivantes: depalim, revenu. Baptiser les groupes du cluster générée:(moins pauvres=?, pauvres=?). Faire une analyse descriptive du cluster avec graphique en

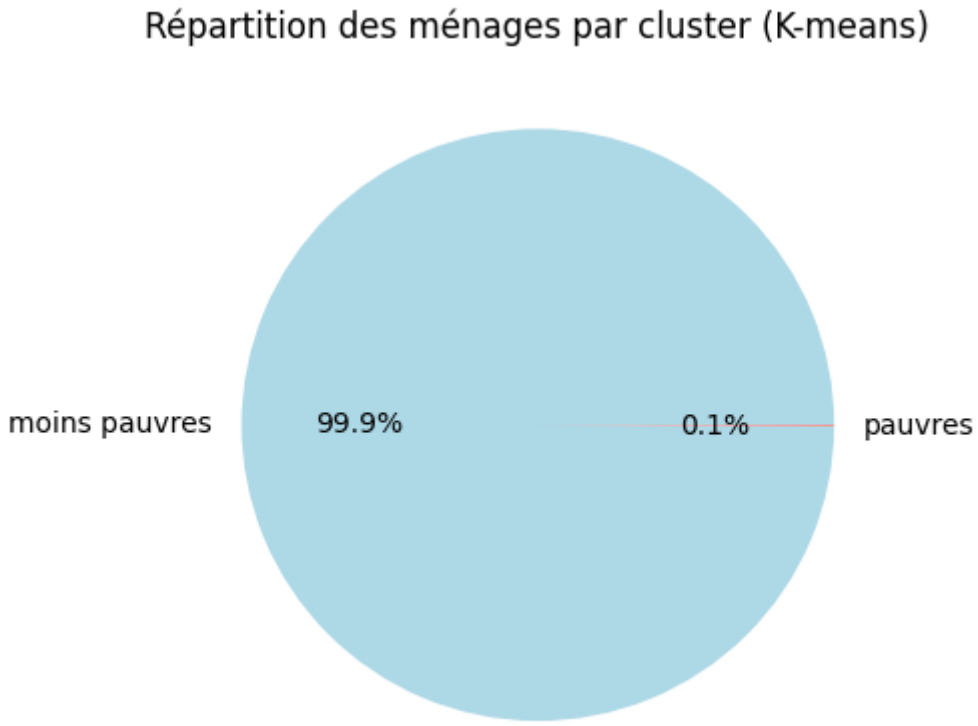
```
# Variables sélectionnées
X = df[['Dep.Alim', 'Montant.Revenu']].dropna()

# Appliquer K-means avec 2 clusters
kmeans = KMeans(n_clusters=2, random_state=42)
df['cluster_kmeans'] = kmeans.fit_predict(X)

# Baptiser les clusters
# Vérifier la moyenne du revenu par cluster pour nommer correctement
cluster_mean = df.groupby('cluster_kmeans')['Montant.Revenu'].mean()
print(cluster_mean)

# Exemple de renommage basé sur le revenu moyen
# Supposons cluster 0 = moins pauvres et cluster 1 = pauvres
df['cluster_kmeans_label'] = df['cluster_kmeans'].map({0:'moins pauvres',
1:'pauvres'})

# Analyse descriptive
print(df['cluster_kmeans_label'].value_counts())

# Graphique en secteur
df['cluster_kmeans_label'].value_counts().plot.pie(autopct='%1.1f%%',
colors=['lightblue','lightcoral'])
plt.title('Répartition des ménages par cluster (K-means)')
plt.ylabel('')
plt.show()
```

```
cluster_kmeans

0     839325.804274
1    1000000.000000
Name: Montant.Revenu, dtype: float64
cluster_kmeans_label
moins pauvres    1170
pauvres             1
Name: count, dtype: int64
```

secteur a l'appui.

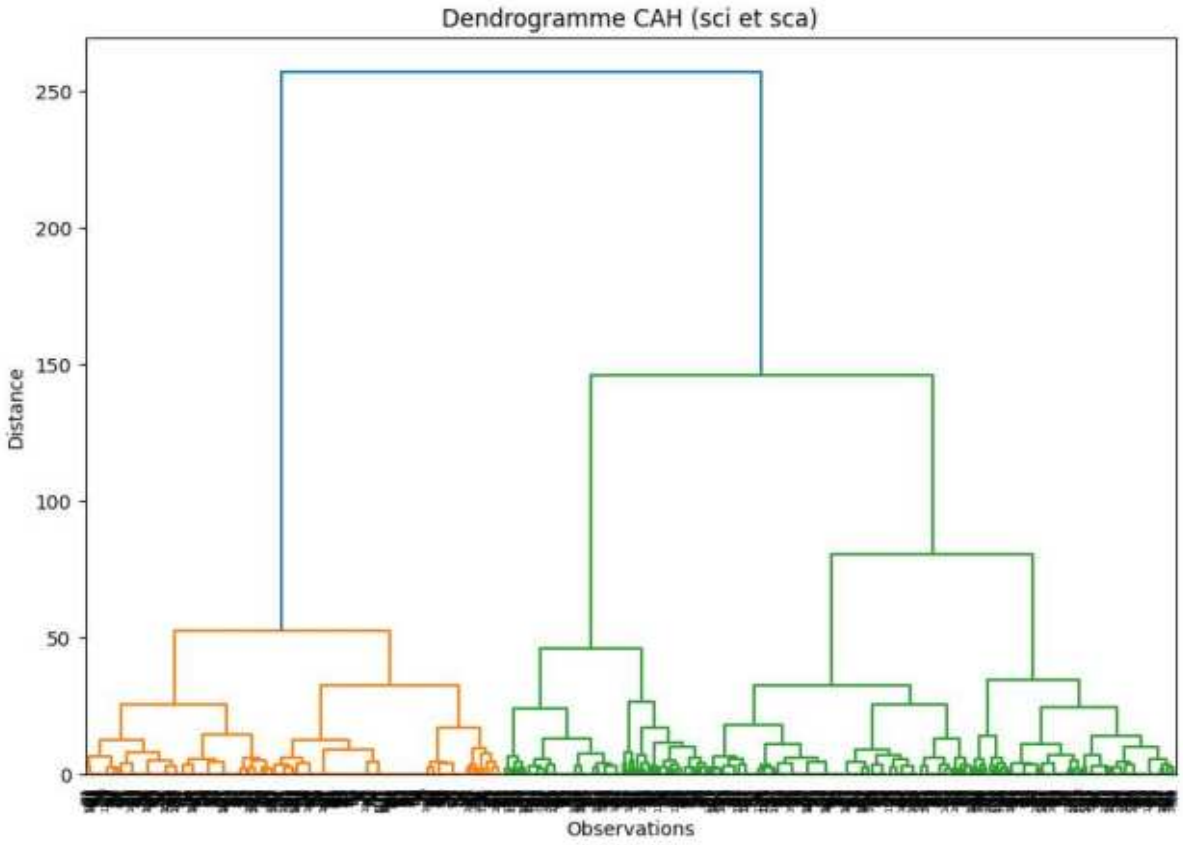
La classification des ménages par K-means en deux clusters, basée sur le revenu et la dépense alimentaire, révèle un déséquilibre marqué. Le cluster des « moins pauvres » regroupe 1170 ménages avec un revenu moyen d'environ 839 326, tandis que le cluster des « pauvres » ne contient qu'un seul ménage ayant un revenu de 1 000 000. Cette partition suggère que le modèle n'a pas réussi à segmenter la population de manière pertinente en deux groupes distincts. Il serait ainsi préférable de réévaluer le

nombre de clusters ou d'explorer d'autres variables explicatives pour obtenir une classification plus informative et exploitable.

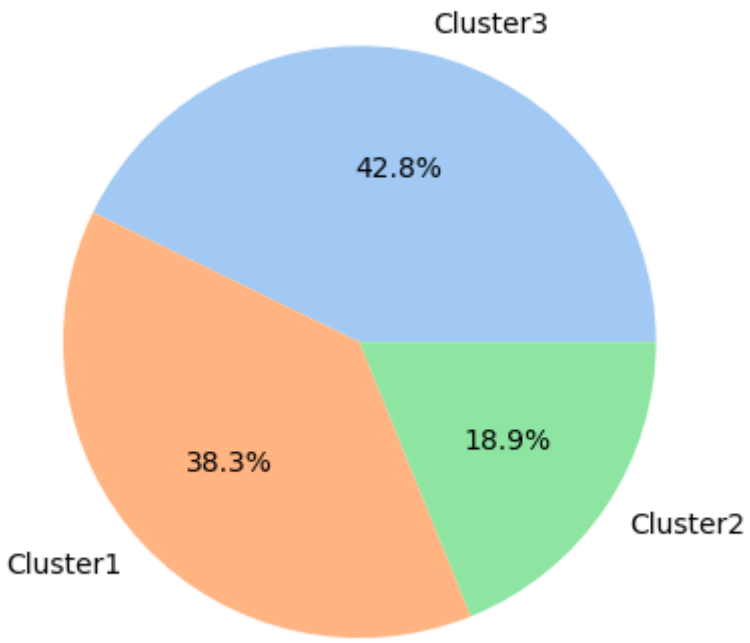


b) Procéder à la classification des ménages, par la méthode Hiérarchical Cluster (CAH) pour 2, 3 ou 4 classes Utilisant les variables sci et sca. Quelle est la meilleure classification à tenir compte. Baptiser la variable cluster retenue Faire une analyse descriptive du cluster avec graphique à l'appui.

```
# Variables sélectionnées
X_cah = df[['sci','sca']].dropna()
# CAH : construction de la hiérarchie
linked = linkage(X_cah, method='ward')
# Dendrogramme pour visualiser
plt.figure(figsize=(10, 7))
dendrogram(linked)
plt.title('Dendrogramme CAH (sci et sca)')
plt.xlabel('Observations')
plt.ylabel('Distance')
plt.show()
# Tester pour 2, 3 et 4 classes et afficher les résultats
for k in [2,3,4]:
    labels = fcluster(linked, k, criterion='maxclust')
    df[f'cluster_cah_{k}'] = labels
    print(f"\n {k} classes :")
    print(df[f'cluster_cah_{k}'].value_counts())
    print(df.groupby(f'cluster_cah_{k}')[['sci','sca']].mean())
# Supposons que 3 classes est la meilleure solution (selon homogénéité et
interprétation pratique)
df['cluster_cah_final'] = df['cluster_cah_3'].map({1:'Cluster1', 2:'Cluster2',
3:'Cluster3'})
# Analyse descriptive : effectifs
print(df['cluster_cah_final'].value_counts())
# Graphique en secteur
df['cluster_cah_final'].value_counts().plot.pie(autopct='%1.1f%%',
colors=sns.color_palette("pastel"))
plt.title('Répartition des ménages par cluster (CAH)')
plt.ylabel('')
plt.show()
```



```
◈ 2 classes :
cluster_cah_2
2    722
1    449
Name: count, dtype: int64
                     sci        sca
cluster_cah_2
1              2.485523   2.320713
2              3.775623  13.160665

◈ 3 classes :
cluster_cah_3
3    501
1    449
2    221
Name: count, dtype: int64
                     sci        sca
cluster_cah_3
1              2.485523   2.320713
2              4.158371  18.936652
3              3.606786  10.612774

◈ 4 classes :
cluster_cah_4
...
Cluster3    501
Cluster1    449
Cluster2    221
```


Répartition des ménages par cluster (CAH)
Cluster3
42.8%
38.3%
18.9%
Cluster2
Cluster1

2.5 Apprentissage automatique supervisé par Régression:

a) Régression Linéaire Multiple: Y=(sca) Xi=(sci, taille, age, sexe, revenu, depalim) ;

```
# Sélection des variables explicatives et de la cible
X = df[['sci','G.Taille','G.Age','sexe','Montant.Revenu','Dep.Alim']]
y = df['sca']

# Supprimer les lignes avec valeurs manquantes sur X ou y
df_model = pd.concat([X, y], axis=1).dropna()

# Recréer X et y après suppression des NA
X = df_model[['sci','G.Taille','G.Age','sexe','Montant.Revenu','Dep.Alim']]
y = df_model['sca']

# Encodage de la variable sexe si elle est de type object
if X['sexe'].dtype == 'object' or
str(X['sexe'].dtype).startswith('category'):
    X = pd.get_dummies(X, columns=['sexe'], drop_first=True)

# Vérifier les types finaux
print(X.dtypes)

# Conversion en numérique si besoin (sécurité)
X = X.apply(pd.to_numeric, errors='coerce')

# Ajouter la constante (intercept)
X = sm.add_constant(X)

# Ajuster le modèle
model = sm.OLS(y, X).fit()

# Résultats
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
====
Dep. Variable:                      sca   R-squared:
0.155
Model:                              OLS   Adj. R-squared:
0.150
Method:                   Least Squares   F-statistic:
34.92
Date:               sam., 12 juil. 2025   Prob (F-statistic):
6.62e-39
Time:                           15:25:49   Log-Likelihood:                -
3715.8
No. Observations:                  1153   AIC:
7446.
Df Residuals:                      1146   BIC:
7481.
Df Model:                             6
Covariance Type:              nonrobust
==============================================================================
=======
                     coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
-------
const              5.9755      1.233      4.846      0.000       3.556
8.395
sci                1.2500      0.096     13.033      0.000       1.062
1.438
G.Taille          -0.0535      0.181     -0.295      0.768      -0.409
0.302
G.Age              0.0812      0.291      0.279      0.780      -0.490
0.652
...
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The condition number is large, 5.79e+07. This might indicate that there
are
strong multicollinearity or other numerical problems.
```

Le modèle de régression linéaire multiple expliquant sca par sci, G.Taille, G.Age, sexe, revenu, et Dep.Alim présente un R^2 de 15,5 %, indiquant que ces variables expliquent une faible proportion de la variance de sca. Seul sci est significativement associé à sca ($p < 0.001$) avec un effet positif : chaque unité supplémentaire de sci augmente sca d'environ 1.25. Les autres variables sont non significatives ($p > 0.05$). Le test global du modèle est toutefois significatif ($p < 0.001$). Enfin, le condition number élevé suggère une possible multicolinéarité, nécessitant une vérification plus approfondie des corrélations entre variables explicatives avant interprétation définitive.

b) Régression Logistique Binaire: Y=(vulnérabilité) Xi=(inondation, pluies.insuffisantes, pluies.horsaison, sexe, taille, age) ;

```
# ◈ Renommer les variables avec . en _
df = df.rename(columns={
    'Pluies.Insuf': 'Pluies_Insuf',
    'Pluies.HorsSaison': 'Pluies_HorsSaison',
    'G.Taille': 'G_Taille',
    'G.Age': 'G_Age'
})

# ◈ Vérifier noms
print(df.columns)

# ◈ Encodage vulnérabilité si nécessaire (0/1)
df['vulnerabilite'] = df['vulnerabilite'].astype(int)

# ◈ Encodage sexe si c'est un texte
if df['sexe'].dtype == 'object' or
str(df['sexe'].dtype).startswith('category'):
    df = pd.get_dummies(df, columns=['sexe'], drop_first=True)

# ◈ Régression logistique
model_logit = smf.logit('vulnerabilite ~ Inondation + Pluies_Insuf +
Pluies_HorsSaison + G_Taille + G_Age', data=df).fit()

# ◈ Résultats
print(model_logit.summary())
```

Optimization terminated successfully.
Current function value: 0.290447
Iterations 7

Logit Regression Results

Dep. Variable:	vulnerabilite	No. Observations:	1153
Model:	Logit	Df Residuals:	1147
Method:	MLE	Df Model:	5
Date:	sam., 12 juil. 2025	Pseudo R-squ.:	0.02851
Time:	15:33:33	Log-Likelihood:	-334.89
converged:	True	LL-Null:	-344.71
Covariance Type:	nonrobust	LLR p-value:	0.001449

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	3.5413	0.558	6.342	0.000	2.447	4.636
Inondation	-0.6777	0.348	-1.945	0.052	-1.361	0.005
Pluies_Insuf	-0.3643	0.301	-1.211	0.226	-0.954	0.226
Pluies_HorsSaison	0.9674	0.523	1.849	0.064	-0.058	1.993

G_Taille	-0.2732	0.090	-3.027	0.002	-0.450	-0.096
G_Age	-0.1143	0.170	-0.673	0.501	-0.447	0.218

La régression logistique binaire visant à expliquer la vulnérabilité à partir de *Inondation, Pluies_Insuf, Pluies_HorsSaison, G_Taille,* et *G_Age* présente un pseudo R^2 faible (2,85 %), indiquant une faible capacité explicative globale. Le modèle est toutefois significatif (p = 0.001). Parmi les variables, seule *G_Taille* est significative (p = 0.002) avec un effet négatif : une augmentation de la taille est associée à une diminution de la probabilité de vulnérabilité. Les effets de *Inondation* et *Pluies_HorsSaison* sont proches du seuil de significativité (p = 0.052 et p = 0.064 respectivement), suggérant une tendance mais sans effet statistiquement confirmé. Les autres variables sont non significatives. Ce résultat invite à intégrer d'autres variables explicatives pour mieux prédire la vulnérabilité.

c) Discriminante: Y=(part.depalim) Xi=(Alim.MoinsPréf, Emprunt.Alim, Red.QuantNourr, Red.NbrRepasJour, Garder.la.Faim)

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
from sklearn.metrics import confusion_matrix

# Sélection des variables disponibles
X =
df[['Alim.MoinsPréf','Emprunt.Alim','Red.QuantNourr','Red.NbrRepasJour']].dropn
a()
y = df.loc[X.index, 'Part.DepAlim']

# Ajustement LDA
lda = LDA()
lda.fit(X, y)

# Prédictions et matrice de confusion
y_pred = lda.predict(X)
print(confusion_matrix(y, y_pred))
```

```
[[  0   0   0 103]
 [  0   0   0 173]
 [  0   0   0 198]
 [  0   0   0 697]]
```

La matrice de confusion obtenue présente une situation particulière : tous les individus sont classés dans une seule et même modalité de la variable cible (la quatrième modalité). Aucun individu n'a été prédit dans les trois premières classes. Cela révèle une performance très faible du modèle discriminant, qui n'arrive pas à différencier les groupes. Cette situation peut s'expliquer par un déséquilibre important entre les modalités ou par une absence d'information discriminante dans les variables explicatives utilisées. Ainsi, la classification prédictive par analyse discriminante n'est pas adaptée ici sans rééquilibrage ou amélioration des variables.

3. ACTUARIAT : Base de données ‘’Climat.SécAlim.2024 ‘’ :

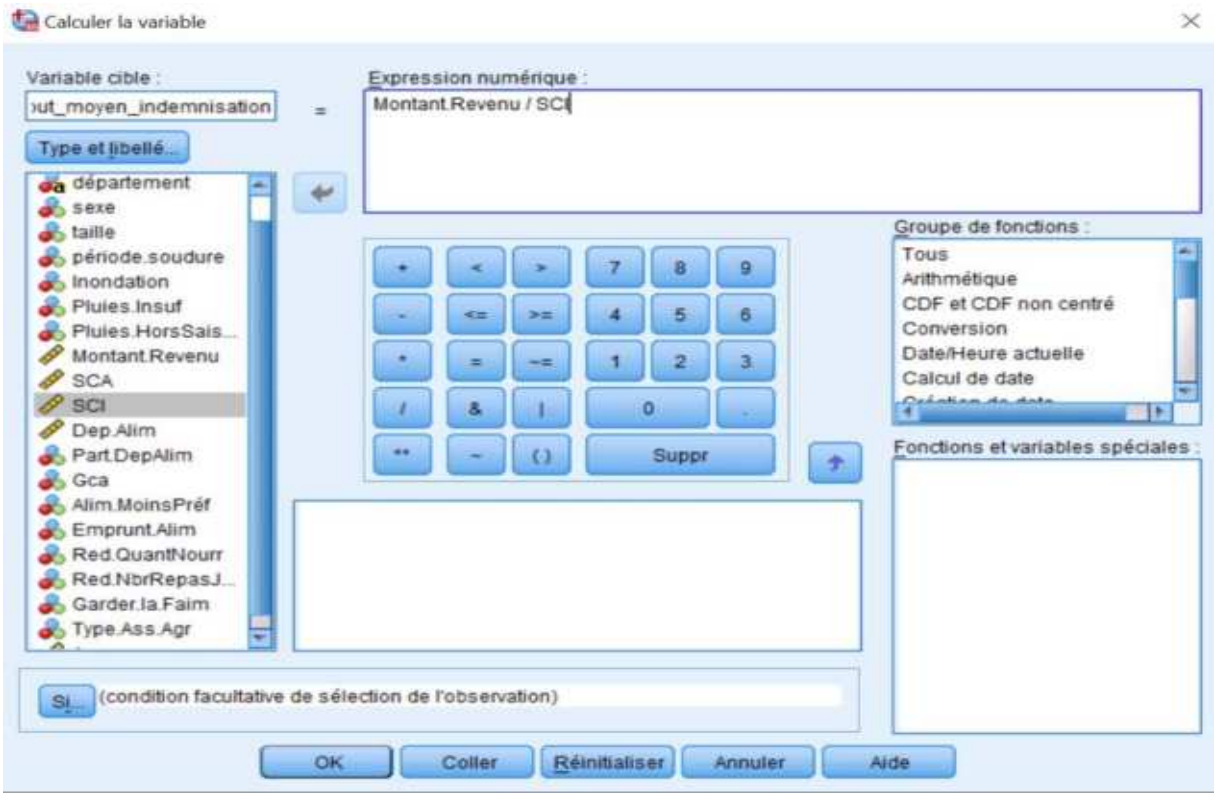
3.1. Créer les variables

- Pour la création de la variable cout moyen indemnisation :

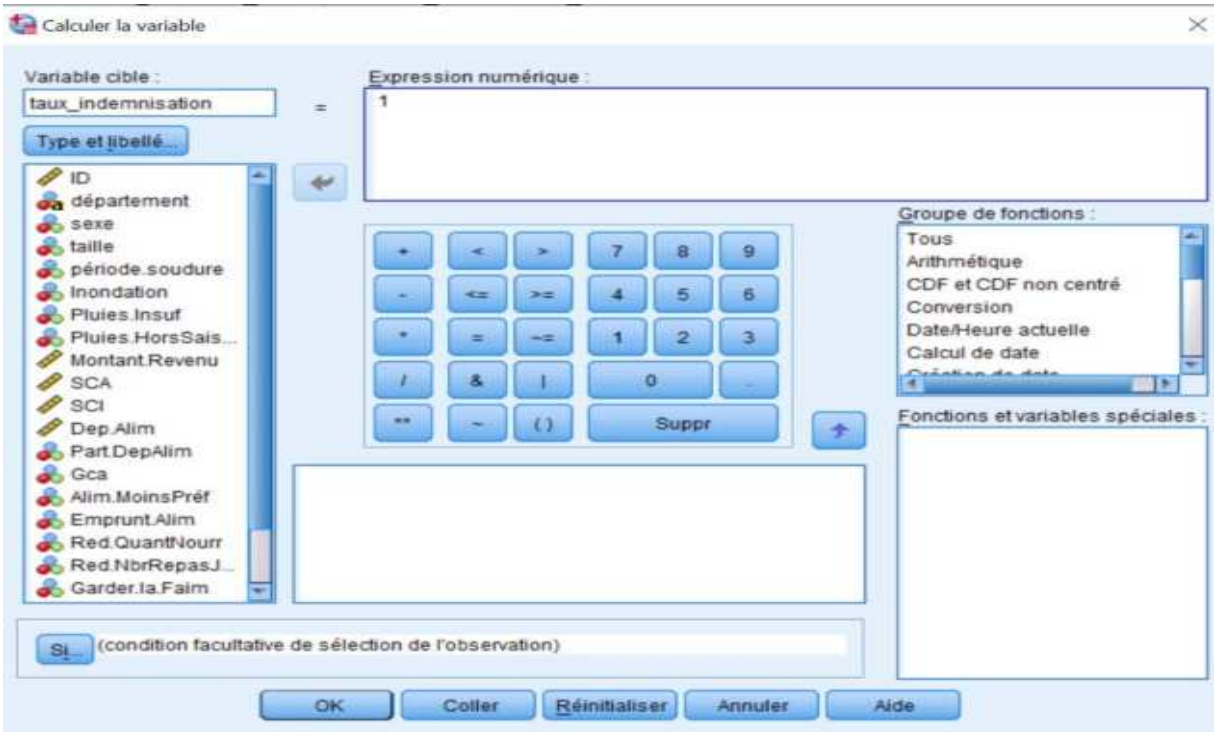
Transformer ► **Calculer la Variable…**

Dans *Variable* saisis cout_moyen_indemnisation, puis dans *Expression* : revenu / SCI.

Clique *OK* pour sortir le résultat :

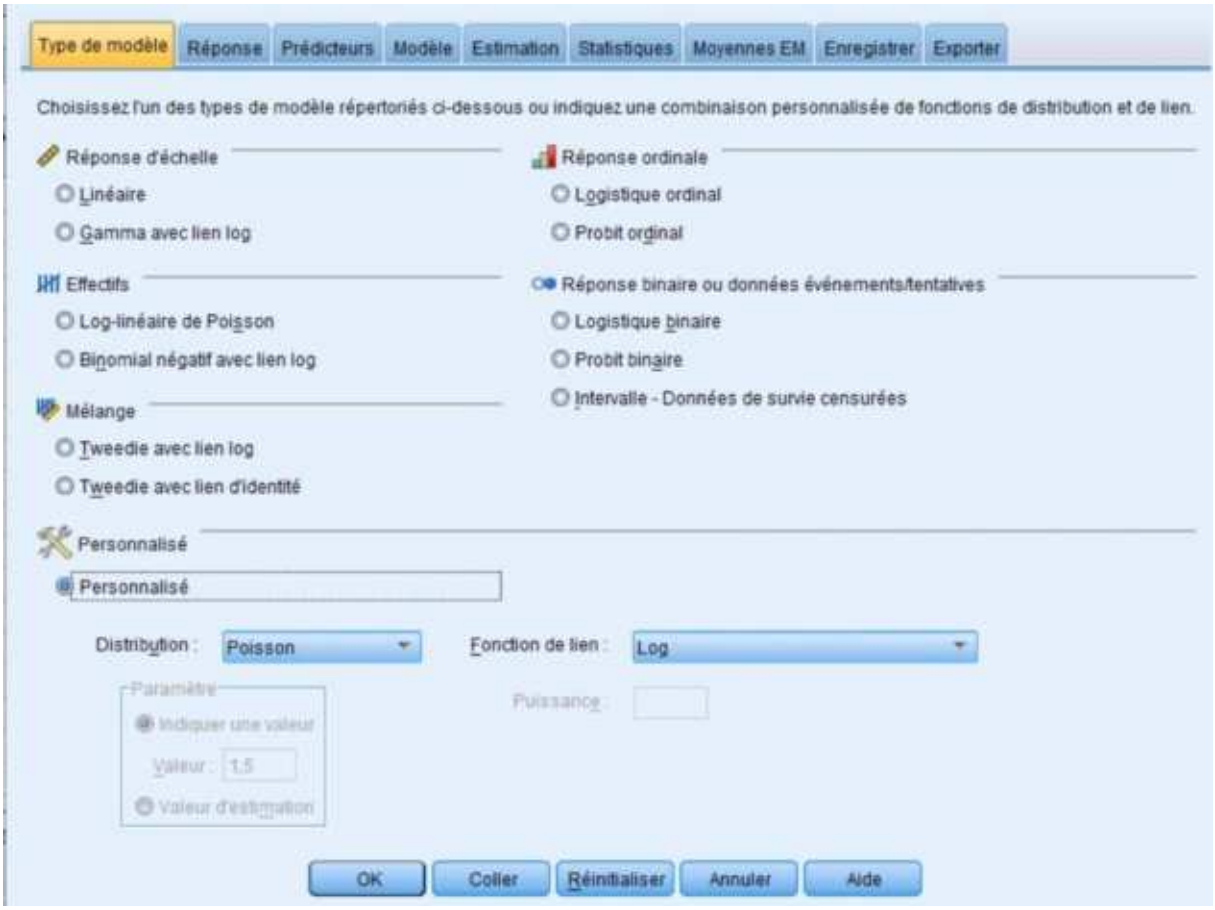


- Pour la création de la variable taux_indemnisation
  Transformer > Calculer Variable…
  Variable → Taux_remboursement
  Expression → 1
  Clique *OK* pour sortir le résultat :



- Frequence.indemnisation

3.2. Proposons un modèle actuariel pour prédire les indemnisations attendues : La Régression log de Poisson



Nous proposons un model actuaire pour prédire les indemnités avec la régression de poisson.

Informations sur le modèle

Variable dépendante	Frequence_indemnisation
Distribution de probabilité	Poisson
Fonction de lien	Log

Qualité d'ajustement[a]

	Valeur	ddl	Valeur/ddl
Déviance	13822,233	1170	11,814
Déviance mise à l'échelle	13822,233	1170	
Khi-deux de Pearson	13918,716	1170	11,896
Khi-deux de Pearson mis à l'échelle	13918,716	1170	
Log de vraisemblance[b]	-8887,072		
Critère d'information d'Akaike (AIC)	17776,145		
Corrigé pour petits échantillons (AICC)	17776,148		
Critère d'informations bayésien (BIC)	17781,210		
AIC cohérent (CAIC)	17782,210		

Ici on constate que la **déviance / ddl** ici = **11,814** > 1 montre un **sur dispersion** (on s'attend à une valeur proche de 1 en Poisson).
Et aussi **AIC = 17776,145.**

Estimations des paramètres

Paramètre	B	Erreur std.	Intervalle de confiance de Wald à 95 %		Test d'hypothèse		
			Inférieur	Supérieur	Khi-deux de Wald	ddl	Sig.
(Constante)	2,456	,0086	2,440	2,473	82415,538	1	,000
(Echelle)	1[a]						

Variable dépendante : Frequence_indemnisation
Modèle : (Constante)

a. Valeur affichée fixe.

La Régression log binomiale négatif :

Nous proposons aussi un model actuaire pour prédire les indemnités avec la régression binomiale négatif.

Modèles linéaires généralisés

Informations sur le modèle

Variable dépendante	Frequence_indemnisation
Distribution de probabilité	Binomiale négative (1,5)
Fonction de lien	Log

Récapitulatif de traitement des observations

	N	Pourcentage
Inclus	1171	100,0%
Exclue	0	0,0%
Total	1171	100,0%

Informations sur la variable continue

		N	Minimum	Maximum	Moyenne	Ecart type
Variable dépendante	Frequence_indemnisation	1171	,00	56,00	11,6635	11,77936

Qualité d'ajustement[a]

	Valeur	ddl	Valeur/ddl
Déviance	1319,850	1170	1,128
Déviance mise à l'échelle	1319,850	1170	
Khi-deux de Pearson	752,554	1170	,643
Khi-deux de Pearson mis à l'échelle	752,554	1170	
Log de vraisemblance[b]	-4070,235		
Critère d'information d'Akaike (AIC)	8142,470		
Corrigé pour petits échantillons (AICC)	8142,473		
Critère d'informations bayésien (BIC)	8147,535		
AIC cohérent (CAIC)	8148,535		

Estimations des paramètres

Paramètre	B	Erreur std.	Intervalle de confiance de Wald à 95 %		Test d'hypothèse		
			Inférieur	Supérieur	Khi-deux de Wald	ddl	Sig.
(Constante)	2,456	,0368	2,384	2,529	4456,025	1	,000
(Echelle)	1[a]						
(Binomiale négative)	1,5[a]						

Variable dépendante : Frequence_indemnisation
Modèle : (Constante)

a. Valeur affichée fixe.

❖ Interprétation :

- Le **modèle** binomial négatif **est clairement supérieur**.
- La déviance/ddl et le khi²/ddl sont désormais **proches de 1** (valeurs attendues pour un bon ajustement).
- Tous les critères d'information (AIC, BIC…) sont **nettement plus bas**, ce qui signifie un meilleur compromis entre qualité d'ajustement et complexité du modèle.

Conclusion :
Le modèle binomial négatif est **statistiquement et économétriquement meilleur**.
Il **corrige la sur dispersion** présente dans le modèle de poisson, ce qui valide son **utilisation pour analyser les déterminants de la fréquence d'indemnisation**.

3.3. Type de distribution conviendrait pour modéliser les pertes extrêmes :

La **régression Gamma avec lien log** (log-gamma):

Informations sur le modèle

Variable dépendante	SMEAN (cout_moyen_indemnisation_1)
Distribution de probabilité	Gamma
Fonction de lien	Log

Récapitulatif de traitement des observations

	N	Pourcentage
Inclus	1171	100,0%
Exclue	0	0,0%
Total	1171	100,0%

Informations sur la variable continue

		N	Minimum	Maximum	Moyenne	Ecart type
Variable dépendante	SMEAN (cout_moyen_indemnisation_1)	1171	465,12	5000000,00	121343,7723	297069,6112

Qualité d'ajustement[a]

	Valeur	ddl	Valeur/ddl
Déviance	2330,888	1170	1,992
Déviance mise à l'échelle	1440,760	1170	
Khi-deux de Pearson	7012,411	1170	5,994
Khi-deux de Pearson mis à l'échelle	4334,484	1170	
Log de vraisemblance[b]	-14769,472		
Critère d'information d'Akaike (AIC)	29542,945		
Corrigé pour petits échantillons (AICC)	29542,955		
Critère d'informations bayésien (BIC)	29553,076		
AIC cohérent (CAIC)	29555,076		

Variable dépendante : SMEAN(cout_moyen_indemnisation_1)
Modèle : (Constante)

- **Déviance/ddl = 1,99** : Trop élevée : **le modèle n'explique pas bien la variabilité** du coût.
- **Khi²/ddl = 5,99** : Très mauvais signe : **forte sur dispersion**.
- AIC, BIC et Log-vraisemblance très élevés/négatifs c'est-à-dire mauvais pouvoir explicatif.

Estimations des paramètres

Paramètre	B	Erreur std.	Intervalle de confiance de Wald à 95 %		Test d'hypothèse		
			Inférieur	Supérieur	Khi-deux de Wald	ddl	Sig.
(Constante)	11,706	,0372	11,634	11,779	99191,041	1	,000
(Echelle)	1,618[a]	,0563	1,511	1,732			

Variable dépendante : SMEAN(cout_moyen_indemnisation_1)
Modèle : (Constante)

Ce modèle sur le coût moyen d'indemnisation est **mal ajusté aux données**.

La régression normale sur la variable log-transformée (log-normale) :

- **Créer une variable log-transformée :**

- **Transformer → Calcul Variable…**

- Nom : ln_cout
- Expression : LN (cout_moyen_indemnisation)

➢ **Lancer une régression linéaire classique :**
➢ **Analyses → Régression → Linéaire…**

- Dépendent : ln_cout
- Independent : les Xi

a. Variable dépendante : ln_cout

b. Toutes les variables demandées ont été introduites.

Récapitulatif des modèles

Modèle	R	R-deux	R-deux ajusté	Erreur standard de l'estimation
1	,692[a]	,479	,475	1,03465

a. Prédicteurs : (Constante), SMEAN(age), sexe du chef de ménage, Pluies hors saison ayant détruit les récoltes du ménage, Score de cpnsommation alimentaire, Pluies insuffisantes ayant réduit les superficies du ménage, Inondations ayant détruit les cultures dans les parcelles du ménage, Dépenses alimentaires, montant total du revenu du ménage au cours des 12 derniers mois, Nombre de stratégie de survie, taille ménage

ANOVA[a]

Modèle		Somme des carrés	ddl	Carré moyen	F	Sig.
1	Régression	1143,012	10	114,301	106,774	,000[b]
	Résidus	1241,781	1160	1,071		
	Total	2384,793	1170			

a. Variable dépendante : ln_cout

b. Prédicteurs : (Constante), SMEAN(age), sexe du chef de ménage, Pluies hors saison ayant détruit les récoltes du ménage, Score de cpnsommation alimentaire, Pluies insuffisantes ayant réduit les superficies du ménage, Inondations ayant détruit les cultures dans les parcelles du ménage, Dépenses alimentaires, montant total du revenu du ménage au cours des 12 derniers mois, Nombre de stratégie de survie, taille ménage

Coefficients[a]

Modèle		Coefficients non standardisés		Coefficients standardisés	t	Sig.
		B	Ecart standard	Bêta		
1	(Constante)	10,620	,220		48,188	,000
	sexe du chef de ménage	,015	,146	,002	,102	,919
	taille ménage	,045	,031	,032	1,472	,141
	Inondations ayant détruit les cultures dans les parcelles du ménage	,060	,123	,010	,491	,624
	Pluies insuffisantes ayant réduit les superficies du ménage	,328	,078	,090	4,230	,000
	Pluies hors saison ayant détruit les récoltes du ménage	-,785	,107	-,158	-7,356	,000
	montant total du revenu du ménage au cours des 12 derniers mois	3,803E-7	,000	,402	18,589	,000
	Score de cpnsommation alimentaire	,001	,001	,008	,389	,698
	Nombre de stratégie de survie	-,061	,003	-,504	-23,438	,000
	Dépenses alimentaires	4,876E-9	,000	,025	1,180	,238
	SMEAN(age)	,055	,049	,024	1,108	,268

a. Variable dépendante : ln_cout

Nous justifions que la régression log normale les meilleurs résultats.

4. ECONOMETRIE : Base de données ‘’Climat.SécAlim.2024 ‘’

4.1 Avantages des données de panel :

Avantages principaux :

- Permet de contrôler l’hétérogénéité individuelle non observée (ex. : différences structurelles entre départements).
- Améliore la précision des estimations (plus d’observations → meilleure efficacité).
- Permet d’analyser la dynamique temporelle (ex : effets de la soudure).
- Réduction du biais d’omission de variables.

Détection de l’hétérogénéité individuelle :

- On peut tester si les effets spécifiques au département sont significatifs en comparant :
  - Un modèle pooled (sans effets individuels)
  - À un modèle à effets fixes ou aléatoires (incluant le facteur département).

4.2. Spécification du modèle :

La variable dépendante est binaire (vulnérabilité = 0/1), donc il faut un modèle « logit/probit » de panel.

Modèle à effets fixes :

$$\text{Logit}\,(P\,(Y_{it}=1)) = \alpha_i + \beta X_{it} + \varepsilon_{it}$$

Où α_i capture les effets propres au département.

[42]:
```
# Liste potentielle des variables explicatives
variables_possible = [
    'Inondation', 'Pluies_Insuf', 'Pluies_HorsSaison', 'Montant_Revenu',
    'SCA', 'SCI', 'GCA', 'part_depalim', 'Red_QuantNourr', 'Red_NbrRepasJour',
    'Garder_la_Faim', 'taux_remboursement', 'ln_cout'
]
```

[45]:
```
# Formule du modèle
formula_fixed = 'vulnerabilite ~ ' + ' + '.join(variables) + ' + C(département)'
model_fixed = smf.logit(formula=formula_fixed, data=df_panel).fit()
```

```
Warning: Maximum number of iterations has been exceeded.
         Current function value: 0.038749
         Iterations: 35
C:\Users\Hp\anaconda3\Lib\site-packages\statsmodels\base\model.py:607: ConvergenceWarning: Maximum Likelihood optimization failed to converge. Check mle_retvals
  warnings.warn("Maximum Likelihood optimization failed to "
```

```python
# Résumé
print("\n=== Modèle à Effets Fixes ===")
print(model_fixed.summary())
```

```
=== Modèle à Effets Fixes ===
                           Logit Regression Results
==============================================================================
Dep. Variable:          vulnerabilite   No. Observations:                 1171
Model:                          Logit   Df Residuals:                     1149
Method:                           MLE   Df Model:                           21
Date:                Sun, 20 Jul 2025   Pseudo R-squ.:                  0.8699
Time:                        17:26:34   Log-Likelihood:                -45.375
converged:                      False   LL-Null:                       -348.71
Covariance Type:            nonrobust   LLR p-value:                6.307e-115
================================================================================================
                                   coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------------------------
Intercept                       16.1897   3.52e+14   4.6e-14      1.000   -6.89e+14    6.89e+14
C(département)[T.Birkilane]    -61.9421   1.26e+05    -0.000      1.000   -2.46e+05    2.46e+05
C(département)[T.Diourbel]     -34.6998   1.44e+04    -0.002      0.998   -2.82e+04    2.81e+04
C(département)[T.Fatick]       -60.9009   6.95e+04    -0.001      0.999   -1.36e+05    1.36e+05
C(département)[T.Foundiougne]  -65.8278   1.19e+06 -5.55e-05      1.000   -2.32e+06    2.32e+06
C(département)[T.Gossas]       -64.2594   4.15e+05    -0.000      1.000   -8.14e+05    8.14e+05
C(département)[T.Guinguinéo]   -60.5246   7.74e+04    -0.001      0.999   -1.52e+05    1.52e+05
C(département)[T.Kaffrine]     -75.7053   1.22e+08 -6.22e-07      1.000   -2.38e+08    2.38e+08
C(département)[T.Kounghuel]    -60.1177   6.31e+04    -0.001      0.999   -1.24e+05    1.24e+05
C(département)[T.Malem Hoddar] -60.9713   5.02e+04    -0.001      0.999   -9.85e+04    9.84e+04
C(département)[T.Mbacke]       -62.3646   5.93e+04    -0.001      0.999   -1.16e+05    1.16e+05
C(département)[T.Rufisque]       3.1377   1.43e+07  2.19e-07      1.000    -2.8e+07     2.8e+07
Inondation                      -0.6570      1.174    -0.559      0.576      -2.959       1.645
Pluies_Insuf                     1.1499      0.872     1.318      0.187      -0.560       2.860
Pluies_HorsSaison               -0.8007      1.168    -0.685      0.493      -3.091       1.489
Montant_Revenu                3.901e-07   2.07e-07     1.887      0.059   -1.51e-08    7.95e-07
SCA                             -0.0031      0.016    -0.192      0.848      -0.035       0.029
SCI                              0.0124      0.026     0.468      0.640      -0.040       0.064
Red_QuantNourr                  -0.0423      0.120    -0.354      0.723      -0.277       0.192
Red_NbrRepasJour                 0.1492      0.151     0.991      0.322      -0.146       0.444
Garder_la_Faim                 -18.5541   1.45e+04    -0.001      0.999   -2.84e+04    2.84e+04
taux_remboursement              16.1897   3.52e+14   4.6e-14      1.000   -6.89e+14    6.89e+14
ln_cout                         -0.0100          0      -inf      0.000      -0.010       -0.010
================================================================================================

Possibly complete quasi-separation: A fraction 0.93 of observations can be
perfectly predicted. This might indicate that there is complete
quasi-separation. In this case some parameters will not be identified.
```

Modèle à effets aléatoires :

$$\text{Logit}\,(P\,(Y_{it}=1)) = \alpha + \mu_i + \beta X_{it} + \varepsilon_{it}$$

- Où μ_i est une erreur aléatoire propre à chaque département.

```python
# Modèle à effets aléatoires (approximation via GLM)
formula_random = 'vulnerabilite ~ ' + ' + '.join(variables)
model_random = smf.glm(formula=formula_random,
                       data=df_panel,
                       family=sm.families.Binomial()).fit()
```

```
print("\n=== Modèle à Effets Aléatoires ===")
print(model_random.summary())
```

```
=== Modèle à Effets Aléatoires ===
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:          vulnerabilite   No. Observations:                 1171
Model:                            GLM   Df Residuals:                     1160
Model Family:                Binomial   Df Model:                           10
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -309.57
Date:                Sun, 20 Jul 2025   Deviance:                       619.13
Time:                        17:27:21   Pearson chi2:                 1.11e+03
No. Iterations:                     6   Pseudo R-squ. (CS):            0.06467
Covariance Type:            nonrobust
==============================================================================
```

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-2.8499	0.647	-4.402	0.000	-4.119	-1.581
Inondation	0.9002	0.372	2.422	0.015	0.172	1.629
Pluies_Insuf	0.1312	0.312	0.421	0.674	-0.480	0.742
Pluies_HorsSaison	-0.5233	0.539	-0.970	0.332	-1.580	0.534
Montant_Revenu	9.641e-08	5.1e-08	1.890	0.059	-3.55e-09	1.96e-07
SCA	0.0039	0.005	0.855	0.392	-0.005	0.013
SCI	-0.0241	0.014	-1.755	0.079	-0.051	0.003
Red_QuantNourr	-0.1455	0.058	-2.490	0.013	-0.260	-0.031
Red_NbrRepasJour	-0.1358	0.081	-1.685	0.092	-0.294	0.022
Garder_la_Faim	-0.2559	0.630	-0.406	0.685	-1.490	0.979
taux_remboursement	-2.8499	0.647	-4.402	0.000	-4.119	-1.581
ln_cout	0.3122	0.106	2.934	0.003	0.104	0.521

4.3. Estimation et choix du modèle : effets fixes vs aléatoires

Test à utiliser :

- **Test de Hausman** :
  - H_0: les effets aléatoires sont appropriés (pas de corrélation entre μ_i et X_{it})
  - H_1: les effets fixes sont préférables (corrélation significative)

Hypothèses nécessaires pour effets aléatoires :

- Les effets spécifiques (département) sont non corrélés avec les variables explicatives X_{it}.
- Si cette hypothèse est violée, il faut préférer les effets fixes.

Test de Hausman est conçu principalement pour les modèles linéaires pas pour les modèles **logit**. Mais on peut :

Estimer les deux modèles,

Comparer les **coefficients communs** par la méthode de Hausman (différence des estimateurs),

Appliquer une version simplifiée du **test de Hausman** basé sur les matrices de covariance.

```python
# 1. Extraire les coefficients et variances-covariances
b_fixed = model_fixed.params
b_random = model_random.params
```

```python
# 2. Conserver uniquement les coefficients communs (pas les dummies "C(département)[T.x]")
common_coef = b_fixed.index.intersection(b_random.index)
```

```python
# 3. Différence des coefficients
b_diff = b_fixed[common_coef] - b_random[common_coef]
```

```python
# 4. Matrices de variance-covariance
V_fixed = model_fixed.cov_params().loc[common_coef, common_coef]
V_random = model_random.cov_params().loc[common_coef, common_coef]
```

```python
# 5. Calcul de la statistique du test de Hausman
V_diff = V_fixed - V_random
try:
    hausman_stat = float(b_diff.T @ np.linalg.inv(V_diff) @ b_diff)
    df_hausman = len(common_coef)
except np.linalg.LinAlgError:
    hausman_stat = np.nan
    df_hausman = len(common_coef)
```

```python
# 6. p-value
from scipy.stats import chi2
p_value = 1 - chi2.cdf(hausman_stat, df_hausman)
```

```python
# 7. Affichage du résultat
print("=== Test de Hausman ===")
print(f"Statistique de test : {hausman_stat:.4f}")
print(f"Degrés de liberté  : {df_hausman}")
print(f"p-value            : {p_value:.4f}")
```

```
=== Test de Hausman ===
Statistique de test : 20.9279
Degrés de liberté  : 12
p-value            : 0.0514
```

```python
if p_value < 0.05:
    print("👉 Rejette H0 : effets fixes préférés (corrélation entre effets spécifiques et X)")
else:
    print("👉 Ne rejette pas H0 : effets aléatoires acceptables")
```

```
👉 Ne rejette pas H0 : effets aléatoires acceptables
```