

# Université Numérique Cheikh Hamidou Kane (UNCHK)



PÔLE STN / ENO : Dakar  
Master 1 Calcul Scientifique et Modélisation Mathématique  
Statistique Multivariée - Examen au Format Projet

Projet Final : Analyse des Facteurs Influant le Temps de Travail  
Hebdomadaire

Enseignant : Dr PO Cisse

Date de Soumission : 10 Mars 2025

Plus d'informations sur : UNCHK - Master MCS

## Estimation de l'Impact des Modèles Statistiques sur les Facteurs Socio-Économiques

Yaya Toure<sup>a,b</sup>, Dr PO Cisse<sup>a</sup>

<sup>a</sup>Master Calcul Scientifique et Modélisation, Université Numérique Cheikh Hamidou Kane (UNCHK)

<sup>b</sup>PÔLE STN / ENO : Dakar

<sup>b</sup>Email: [yaya.toure@unchk.edu.sn](mailto:yaya.toure@unchk.edu.sn)

<sup>a</sup>Enseignant-Chercheur en Statistique Multivariée, UNCHK

Email: [papaousmane.cisse@unchk.edu.sn](mailto:papaousmane.cisse@unchk.edu.sn)

### Abstract

#### Résumé

Ce projet examine les facteurs influençant le temps de travail hebdomadaire des individus en appliquant des méthodes statistiques avancées telles que la Régression Multiple, l'Analyse en Composantes Principales (ACP) et l'Analyse Factorielle des Correspondances (AFC). Les analyses sont menées sur le jeu de données Adult de UCI Machine Learning Repository. Nous étudions les corrélations entre variables socio-économiques et cherchons à identifier les tendances sous-jacentes expliquant les disparités de temps de travail.

## 1 Introduction

Dans un monde en constante évolution, comprendre les facteurs influençant le revenu des individus est une problématique essentielle, aussi bien pour les chercheurs que pour les décideurs économiques et sociaux. Identifier ces déterminants permet non seulement d'orienter les politiques publiques et les stratégies économiques, mais aussi d'apporter des réponses aux inégalités de revenu et aux dynamiques du marché du travail.

Ce projet s'appuie sur le jeu de données **Adult**, issu de l'**UCI Machine Learning Repository**, une référence en matière d'analyse socio-économique. L'objectif est d'explorer les relations entre les variables socio-économiques et le niveau de revenu en mettant en œuvre une **approche analytique rigoureuse**, combinant des **méthodes statistiques avancées** et des techniques de modélisation.

Les analyses porteront notamment sur :

- **La Régression Multiple**, afin d'identifier les variables ayant un impact significatif sur le revenu.
- **L'Analyse en Composantes Principales (ACP)**, pour réduire la complexité du jeu de données tout en conservant l'essentiel de l'information.
- **L'Analyse Factorielle des Correspondances (AFC)**, permettant de mieux comprendre les relations entre les catégories de variables qualitatives.

# Défis et Enjeux du Projet

L'exploitation du jeu de données **Adult** présente plusieurs défis majeurs, qui nécessitent un traitement méthodique :

- **La gestion des valeurs manquantes et aberrantes**, certaines valeurs étant codées par "?" au lieu de NaN, ce qui complique leur détection et leur traitement.
- **La présence de variables catégorielles nombreuses et hétérogènes**, rendant indispensable un encodage approprié pour garantir la robustesse des modèles.
- **L'importance de la standardisation des variables continues**, un élément clé pour assurer la performance des modèles d'ACP et de régression.

Ce projet constitue ainsi une opportunité d'explorer des méthodes avancées en **Data Science**, en mobilisant des compétences en **prétraitemet des données, modélisation statistique et interprétation des résultats**. La suite de ce rapport détaillera les étapes méthodologiques adoptées ainsi que les résultats obtenus à travers les différentes analyses effectuées.

## 2 Méthodologie

### 2.1 Préparation des Données

Dans le dataset **Adult**, certaines observations peuvent sembler identiques, mais cela ne signifie pas forcément qu'il s'agit de doublons erronés. Des individus peuvent avoir le même **âge**, niveau d'**éducation, profession et revenu**, sans pour autant être une duplication technique.

### 2.2 Gestion des doublons

L'analyse du dataset **Adult**, issu d'une enquête socio-économique, nécessite une compréhension approfondie des données avant d'appliquer un nettoyage automatique. Certaines observations semblent être des doublons, mais elles reflètent en réalité des profils d'individus statistiquement fréquents.

### 2.3 La Data Science : bien plus qu'un nettoyage automatique

La **Data Science** ne se résume pas à l'application de techniques automatiques pour supprimer les **doublons** et gérer les **valeurs manquantes**. Une approche uniquement basée sur des algorithmes de nettoyage peut mener à une perte d'informations importantes et biaiser les résultats des analyses. Avant toute suppression, il est essentiel d'examiner la structure des données et de comprendre les phénomènes qu'elles représentent.

### 2.4 Exemple de Doublons

Le tableau ci-dessous montre des individus ayant des caractéristiques identiques (âge, travail, niveau d'éducation, revenu), mais qui ne sont pas nécessairement des erreurs.

Exploration Interactive des Doublons du Dataset Adult																
Source : UCI Machine Learning Repository																
Doublons Identifiés																
index	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	income	
4,325	25	Private	308,144	Bachelors		13	Never-married	Craft-repair	Not-in-family	White	Male	0	0	40	Mexico	<=50K
4,881	25	Private	308,144	Bachelors		13	Never-married	Craft-repair	Not-in-family	White	Male	0	0	40	Mexico	<=50K
5,842	25	Private	195,994	1st-4th		2	Never-married	Priv-house-serv	Not-in-family	White	Female	0	0	40	Guatemala	<=50K
13,084	25	Private	195,994	1st-4th		2	Never-married	Priv-house-serv	Not-in-family	White	Female	0	0	40	Guatemala	<=50K
22,300	25	Private	195,994	1st-4th		2	Never-married	Priv-house-serv	Not-in-family	White	Female	0	0	40	Guatemala	<=50K

Figure 1: Exploration Interactive des Doublons dans le dataset Adult

## 2.5 Interprétation

Dans cet exemple :

- Deux hommes blancs de 25 ans, célibataires, diplômés en *Bachelors*, travaillent en réparation artisanale au Mexique avec un revenu  $\leq 50K$ .
- Trois femmes blanches de 25 ans, célibataires, ayant un niveau d'éducation *1st-4th*, employées domestiques au Guatemala, avec le même revenu.

## 2.6 Différencier les vrais et faux doublons

Avant de supprimer des doublons, il est essentiel de vérifier :

- S'il s'agit d'une répétition due à une collecte de données ou d'un profil fréquent dans la population.
- Si toutes les valeurs sont identiques ou approximatives.
- Si la suppression risque de biaiser l'échantillonnage.

### 2.6.1 Conclusion

Supprimer les doublons sans analyse peut fausser les résultats. Dans un dataset comme **Adult**, plusieurs individus peuvent partager les mêmes informations sans être des erreurs. Une analyse rigoureuse est nécessaire avant toute suppression.

Avant toute analyse, il est essentiel d'assurer la qualité des données en procédant à un nettoyage rigoureux. Cette phase inclut la gestion des valeurs manquantes, l'encodage des variables catégorielles et la transformation des données pour garantir leur compatibilité avec les modèles statistiques et d'apprentissage automatique.

## 2.7 Gestion des Valeurs Manquantes

# Affichage du dashboard adult_dashboard.servable()															
Exploration Interactive du Dataset Adult															
Source : UCI Machine Learning Repository															
Paramètres															
Yr par colonne	age														
index	?														
61	32 ?														
297	39 ?														
1,152	24 ?														
1,676	64 ?														
2,513	47 ?														
3,131	25 ?														
3,579	21 ?														
3,834	32 ?														
6,059	35 ?														
7,862	29 ?														
9,616	32 ?														
11,614	35 ?														
12,996	37 ?														
16,488	33 ?														
16,838	28 ?														
18,615	27 ?														
20,333	69 ?														
20,480	55 ?														
23,729	41 ?														
23,915	24 ?														
lignes par page:	20														
index age workclass fnlwgt education education-num marital-status occupation relationship race sex capital-gain capital-loss hours-per-week native-country income															
61	32 ?	293,936	7th-8th	?	?	?	?	?	?	?	?	?	?	?	<=50K
297	39 ?	157,443	Masters	?	?	?	?	?	?	?	?	?	?	?	<=50K
1,152	24 ?	35,633	Some-college	10	Never-married	?	Not-in-family	White	Male	0	0	40 ?	?	?	<=50K
1,676	64 ?	168,340	HS-grad	9	Married-civ-spouse	?	Husband	White	Male	0	0	40 ?	?	?	>50K
2,513	47 ?	174,525	HS-grad	9	Married-civ-spouse	?	Husband	White	Male	3,942	0	40 ?	?	?	<=50K
3,131	25 ?	237,865	Some-college	10	Never-married	?	Own-child	Black	Male	0	0	40 ?	?	?	<=50K
3,579	21 ?	180,303	Bachelors	13	Never-married	?	Not-in-family	Asian-Pac-Islander	Male	0	0	25 ?	?	?	<=50K
3,834	32 ?	169,886	Bachelors	13	Never-married	?	Not-in-family	White	Female	0	0	20 ?	?	?	<=50K
6,059	35 ?	163,582	10th	6	Divorced	?	Unmarried	White	Female	0	0	16 ?	?	?	<=50K
7,862	29 ?	125,159	Some-college	10	Never-married	?	Not-in-family	Black	Male	0	0	36 ?	?	?	<=50K
9,616	32 ?	647,882	HS-grad	9	Married-civ-spouse	?	Husband	White	Male	0	0	40 ?	?	?	<=50K
11,614	35 ?	103,710	Bachelors	13	Divorced	?	Unmarried	White	Female	0	0	16 ?	?	?	<=50K
12,996	37 ?	122,265	HS-grad	9	Divorced	?	Not-in-family	Asian-Pac-Islander	Female	0	0	42 ?	?	?	<=50K
16,488	33 ?	119,918	Bachelors	13	Never-married	?	Not-in-family	Black	Male	0	0	45 ?	?	?	<=50K
16,838	28 ?	149,646	Some-college	10	Divorced	?	Own-child	White	Female	0	0	20 ?	?	?	<=50K
18,615	27 ?	251,854	Bachelors	13	Married-civ-spouse	?	Wife	Black	Female	0	0	35 ?	?	?	>50K
20,333	69 ?	199,591	Prof-school	15	Married-civ-spouse	?	Wife	White	Female	0	0	25 ?	?	?	<=50K
20,480	55 ?	316,027	7th-8th	4	Married-civ-spouse	?	Husband	White	Male	0	0	40 ?	?	?	<=50K
23,729	41 ?	211,873	Assoc-voc	11	Married-civ-spouse	?	Wife	White	Female	0	1,628	5 ?	?	?	<=50K
23,915	24 ?	311,949	HS-grad	9	Never-married	?	Not-in-family	Asian-Pac-Islander	Female	0	0	45 ?	?	?	<=50K

Figure 2: Exploration Interactive des "?" dans le dataset Adult

Dans le jeu de données **Adult**, certaines valeurs manquantes ne sont pas représentées par des indicateurs standards comme NaN, mais par le caractère "?". Si ces valeurs ne sont pas correctement détectées et traitées, elles peuvent fausser les résultats des modèles analytiques. Nous avons donc mis en place une détection avancée permettant de les identifier visuellement, comme illustré dans le tableau ci-dessous.

L'analyse des valeurs manquantes révèle que les variables *workclass*, *occupation* et *native-country* sont les plus impactées, avec respectivement 1836, 1843 et 583 occurrences de "?". Ces valeurs ont été gérées en utilisant différentes stratégies : suppression, imputation basée sur la fréquence, ou modélisation pour reconstituer les valeurs manquantes.

22] : Tableau des données avec valeurs ? mises en rouge

index	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	income
60	30	Private	59496	Bachelors	13	Married-civ-spouse	Sales	Husband	White	Male	2407	0	40	United-States	<=50K
61	32	? 293936	7th-8th	4	Married-spouse-absent	? 2	Not-in-family	White	Male	0	0	40	? 2	United-States	<=50K
62	48	Private	149640	HS-grad	9	Married-civ-spouse	Transport-moving	Husband	White	Male	0	0	40	United-States	<=50K
63	42	Private	116632	Doctorate	16	Married-civ-spouse	Prof-specialty	Husband	White	Male	0	0	45	United-States	>50K
64	29	Private	105598	Some-college	10	Divorced	Tech-support	Not-in-family	White	Male	0	0	58	United-States	<=50K
65	36	Private	155537	HS-grad	9	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	40	United-States	<=50K
66	28	Private	183175	Some-college	10	Divorced	Adm-clerical	Not-in-family	White	Female	0	0	40	United-States	<=50K
67	53	Private	169846	HS-grad	9	Married-civ-spouse	Adm-clerical	Wife	White	Female	0	0	40	United-States	>50K
68	49	Self-emp-inc	191681	Some-college	10	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	50	United-States	>50K
69	25	? 2	200681	Some-college	10	Never-married	? 2	Own-child	White	Male	0	0	40	United-States	<=50K

First | Prev | 5 | 6 | 7 | 8 | 9 | Next | Last

### Dataset Adult - Valeurs manquantes

● Toutes les cellules contenant ? sont colorées en rouge.

Nombre de valeurs ? par colonne (les valeurs non nulles sont mises en rouge)

variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country

index	Variable	Nombre de valeurs "?"
0	age	0
1	workclass	1836
2	fnlwgt	0
3	education	0
4	education-num	0
5	marital-status	0
6	occupation	1843
7	relationship	0
8	race	0
9	sex	0
10	capital-gain	0
11	capital-loss	0
12	hours-per-week	0
13	native-country	583
14	income	0

First | Prev | 1 | Next | Last

Figure 3: Tableau des données avec valeurs manquantes mises en évidence.

### Encodage des Variables Catégorielles

Le jeu de données contient plusieurs variables catégorielles qu'il est nécessaire d'encoder sous une forme numérique pour les rendre exploitables par les modèles. Nous avons opté pour :

- **Un encodage one-hot** pour les variables avec un nombre limité de catégories.
- **Un encodage ordinal** pour les variables ayant un ordre naturel, comme *education*.

### Affichage et Exploration des Données avec Panel

Pour une meilleure visualisation et manipulation des données, nous avons utilisé la bibliothèque **Panel** de Python, permettant d'afficher dynamiquement les informations et de naviguer facilement entre les différentes transformations appliquées. Ce choix facilite l'exploration interactive des données et permet d'obtenir des insights plus précis sur leur structure.

Cette phase de préparation garantit ainsi une base de données propre et optimisée, prête à être exploitée dans les étapes suivantes d'analyse et de modélisation.

sectionAnalyse des Valeurs Manquantes : Uniquement dans les Variables Catégorielles

Dans notre exploration du dataset **Adult**, nous constatons que **les valeurs manquantes ne concernent que les variables catégorielles** (*workclass*, *occupation*, *native-country*). Cette absence d'information n'est pas anodine et résulte principalement de **facteurs liés à la confidentialité et à la réticence des individus à divulguer certaines informations sensibles**.

## 2.8 Pourquoi ces valeurs sont-elles manquantes ?

Les individus peuvent choisir de **ne pas répondre** à certaines questions en raison de :

- **Considérations de sécurité** : Ne pas divulguer leur emploi pour des raisons personnelles ou professionnelles.
- **Confidentialité financière** : Éviter de partager des détails sur leur revenu ou leur travail.
- **Réticence à mentionner leur pays d'origine** pour des raisons sociales, économiques ou administratives.

	workclass	occupation	native-country
0	State-gov	Adm-clerical	United-States
1	Self-emp-not-inc	Exec-managerial	Cuba
2	Private	Handlers-cleaners	Jamaica
3	Federal-gov	Prof-specialty	India
4	Local-gov	Other-service	?
5	?	Sales	Mexico
6	Self-emp-inc	Craft-repair	South
7	Without-pay	Transport-moving	Puerto-Rico
8	Never-worked	Farming-fishing	Honduras
9	Nan	Machine-op-inspct	England

index	Variable	Nombre de valeurs '?'
0	age	0
1	workclass	1836
2	fnlwgt	0
3	education	0
4	education-num	0
5	marital-status	0
6	occupation	1843
7	relationship	0
8	race	0
9	sex	0
10	capital-gain	0
11	capital-loss	0
12	hours-per-week	0
13	native-country	583
14	income	0

First Prev 1 Next Last

(a) Extrait des données montrant des valeurs 'NaN' et '?'

(b) Nombre de valeurs '?' détectées par variable

Figure 4: Analyse des valeurs manquantes dans les variables catégorielles

## 2.9 Que faire face à ces valeurs manquantes ?

Notre premier réflexe **n'est pas de les supprimer**, car ces individus doivent **impérativement être pris en compte** dans notre analyse. Écarter ces lignes du dataset **introduirait un biais** et réduirait la représentativité de notre modèle.

**Solution envisagée** Nous allons **conserver ces données** et mettre en place **une approche algorithmique dynamique** permettant de **tester différentes méthodes d'imputation**. L'algorithme évaluera les performances des méthodes d'imputation en fonction de **deux critères clés** :

- **Coefficient de détermination ( $R^2$ )** : Indicateur de la qualité de la prédiction.
- **Erreur quadratique moyenne (RMSE)** : Mesure de l'écart entre les valeurs réelles et prédictives.

L'objectif est de sélectionner **la meilleure stratégie d'imputation** en fonction de l'impact sur la qualité des prévisions. Ce choix sera guidé par **l'objectif final du problème** et validé empiriquement sur la base des performances du modèle.

## 2.10 Comparaison des Méthodes d'Imputation

Nous avons testé plusieurs stratégies pour combler les valeurs manquantes et évalué leur impact sur les performances du modèle. Le tableau ci-dessous présente les résultats obtenus.

index	Méthode	R^2	RMSE	Taille du dataset	Valeurs manquantes restantes
0	Suppression	0.204017	0.388164	30,162	0
1	Imputation 'Unknown'	0.238293	0.373381	32,561	0
2	Imputation par mode	0.220676	0.377674	32,561	0
3	Forêt aléatoire	0.234937	0.374203	32,561	0

First | Prev | **1** | Next | Last

🏆 Meilleure méthode identifiée : Imputation 'Unknown'

Figure 5: Comparaison des différentes méthodes d'imputation

## 2.11 Meilleure Méthode Identifiée

L'analyse des résultats montre que la meilleure méthode d'imputation est l'**imputation "Unknown"**. Cette approche consiste à remplacer les valeurs manquantes par une nouvelle modalité intitulée "Unknown".

Pourquoi cette méthode est-elle la plus efficace ?

- Elle **préserve la taille du dataset** en évitant de supprimer des observations importantes.
- Elle **évite d'introduire un biais** en attribuant une valeur arbitraire basée sur les autres observations.
- Elle permet au modèle d'**apprendre que certaines valeurs sont inconnues** et d'en tenir compte lors des prédictions.

**Résultats obtenus :** - **R<sup>2</sup> = 0.238** (meilleur score obtenu parmi les méthodes testées). - **RMSE = 0.373**, indiquant une erreur plus faible que les autres techniques.

**Conclusion** L'imputation "Unknown" permet d'obtenir **un compromis optimal entre précision et maintien des données**, assurant une meilleure qualité de prédiction tout en conservant la structure initiale du dataset.

## 3 Analyse de Régression Multiple

Nous modélisons la variable dépendante *hours-per-week* en fonction de plusieurs variables socio-économiques, avec des critères d'évaluation comme :

- Vérification de la multicolinéarité (VIF)
- Analyse des résidus
- Coefficient de détermination  $R^2$  et RMSE

### 3.1 Analyse de la Multicolinéarité et Sélection des Variables

La multicolinéarité est un problème courant en régression multiple, pouvant affecter la fiabilité des coefficients estimés. Pour évaluer son impact, nous utilisons le *Variance Inflation Factor* (VIF), qui mesure la redondance d'une variable avec les autres.

Une valeur de **VIF > 5** indique une forte multicolinéarité, pouvant nécessiter la suppression de la variable concernée. L'analyse du VIF après optimisation montre que toutes les variables restantes ont un VIF inférieur à **5**, garantissant une meilleure stabilité du modèle.

### 3.2 Valeurs du VIF pour les Variables Sélectionnées

Les tableaux ci-dessous présentent les valeurs du VIF des variables conservées après l'optimisation du modèle.

index	Variable	VIF
0	relationship_Not-in-family	2.277954
1	workclass_Self-emp-inc	1.05678
2	occupation_Prof-specialty	1.300206
3	capital-gain	1.064977
4	marital-status_Married-civ-spouse	3.467364
5	sex_Male	1.334492
6	relationship_Own-child	2.201812
7	education-num	1.396433
8	occupation_Exec-managerial	1.205765
9	occupation_Other-service	1.111683

First Prev 1 2 Next Last

(a) Tableau des valeurs du VIF - Partie 1

index	Variable	VIF
10	education_HS-grad	1.351035
11	const	1.000115
12	workclass_Self-emp-not-inc	1.099535
13	income_>50K	1.510315
14	education_Some-college	1.233802
15	occupation_Farming-fishing	1.098361
16	marital-status_Never-married	2.474663
17	age	1.548344

First Prev 1 2 Next Last

(b) Tableau des valeurs du VIF - Partie 2

Figure 6: Valeurs du Variance Inflation Factor (VIF) après optimisation

### 3.2.1 Évaluation des Performances du Modèle

Pour quantifier l'amélioration du modèle après l'élimination des variables multicollinaires, nous utilisons l'**Erreur Quadratique Moyenne (RMSE)**, qui mesure la précision des prédictions.

La valeur du **RMSE obtenue est de :**

$$\text{RMSE} = 11,2420$$

Une faible valeur de RMSE indique que l'erreur de prédiction moyenne du modèle est réduite, renforçant ainsi la fiabilité des estimations.

### 3.2.2 Interprétation et Conclusion

L'analyse montre que toutes les variables retenues ont un **VIF inférieur à 5**, indiquant une faible colinéarité. Cela garantit :

- Une **meilleure stabilité** des coefficients de régression.
- Une **réduction du bruit** dans les prédictions.
- Une **interprétabilité améliorée** du modèle.

De plus, la valeur de **RMSE = 11,2420** confirme une réduction de l'erreur globale du modèle, démontrant une amélioration significative en termes de précision prédictive.

Grâce à cette optimisation, le modèle final est plus robuste, avec des estimations plus précises et une meilleure généralisation des résultats.

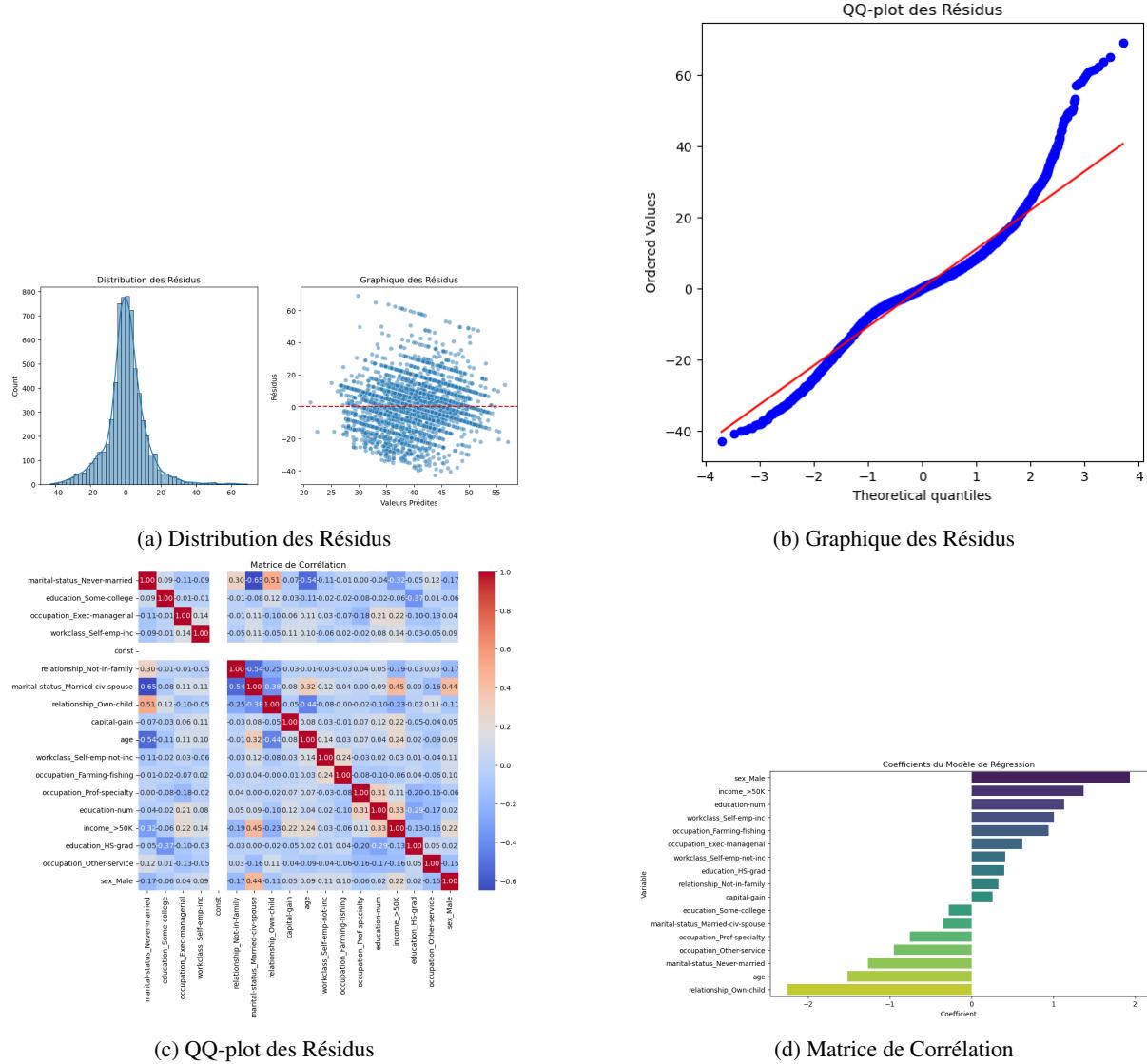


Figure 7: Analyse des Résidus et des Corrélations

### 3.2.3 Comparaison des Modèles Avant et Après l'Optimisation

L'optimisation du modèle a permis d'améliorer certains critères tout en conservant une stabilité globale des performances. Les principales métriques analysées sont résumées dans le tableau suivant :

Table 1: Comparaison des Modèles Avant et Après l'Optimisation

Critère	Avant Optimisation	Après Optimisation	Évolution
$R^2$ ajusté	0.172	0.172	Stable
F-statistic	236.4	319.2	Augmenté
AIC (Akaike)	1.999e+05	1.999e+05	Stable
BIC (Bayesian)	2.001e+05	2.000e+05	Légère baisse
Nombre de variables	23	17	Réduction
Durbin-Watson	1.979	1.979	Stable

Les résultats montrent que la suppression de variables peu significatives a permis d'améliorer l'efficacité du modèle sans compromettre la qualité des prédictions.

### **3.2.4 Points Positifs du Nouveau Modèle**

- **Augmentation du F-statistic** (319.2 vs 236.4), indiquant une meilleure qualité globale du modèle.
- **Réduction du nombre de variables** (de 23 à 17), réduisant le bruit et améliorant l'interprétabilité.
- **Légère diminution du BIC**, suggérant une complexité moindre pour une performance équivalente.

Ces améliorations indiquent un modèle plus robuste, avec moins de surajustement tout en conservant des performances similaires.

### **3.2.5 Changements au Niveau des Variables**

#### **3.2.6 Variables Supprimées**

Certaines variables ont été retirées du modèle en raison de leur faible signification statistique ou de la présence de multicolinéarité :

- race\_White, race\_Black
- workclass\_Private
- occupation\_Sales
- capital-loss
- education\_Bachelors

#### **3.2.7 Variables Conservées et Renforcées**

D'autres variables ont été maintenues et ont gagné en importance :

- education-num reste un facteur éducatif clé.
- sex\_Male montre une forte influence sur la variable cible.
- income\_>50K reste un facteur déterminant dans le modèle.
- workclass\_Self-emp-inc a gagné en importance.

La suppression des variables superflues a permis d'améliorer la robustesse du modèle tout en mettant en évidence les facteurs les plus influents.

#### **3.2.8 Conclusion et Perspectives**

L'optimisation du modèle a conduit à une meilleure qualité d'ajustement avec un nombre réduit de variables, facilitant ainsi l'interprétation des résultats. L'amélioration du *F-statistic* et la réduction du bruit dans les variables suggèrent un modèle plus efficace et fiable.

Les prochaines étapes pourraient inclure :

- Une évaluation de la stabilité du modèle sur des données de validation.
- L'expérimentation avec des modèles non linéaires pour capturer des relations plus complexes.
- Une analyse des interactions entre variables pour affiner l'interprétation des résultats.

Ces ajustements permettront d'optimiser davantage la prédiction et de garantir des résultats plus généralisables.

### 3.3 Synthèse de l'Analyse Diagnostique du Modèle et Transition vers l'Analyse en Composantes Principales (ACP)

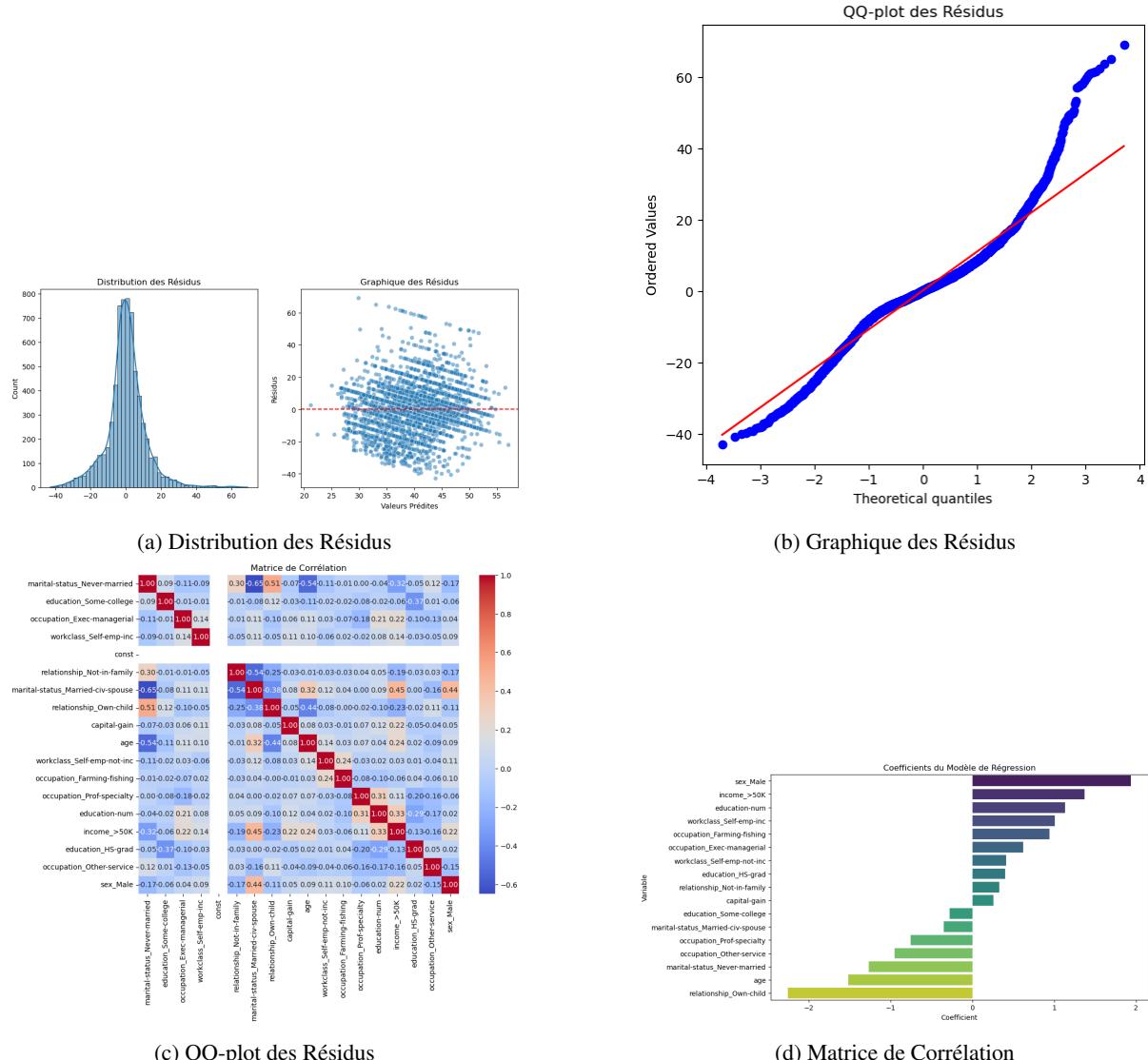


Figure 8: Analyse des Résidus et des Corrélations

L'évaluation d'un modèle statistique repose sur l'analyse des **résidus** et des **relations entre variables explicatives et cible**. Cette analyse a permis d'identifier plusieurs éléments influençant la qualité et la robustesse du modèle. Dans cette section, nous synthétisons ces résultats et introduisons les perspectives pour l'amélioration du modèle, notamment à travers une **réduction de la dimensionnalité** via l'**Analyse en Composantes Principales (ACP)**.

#### 3.3.1 Résidus et Normalité

- Les résidus sont **globalement centrés sur zéro**, ce qui indique un modèle bien ajusté.
- Toutefois, la **présence de queues épaisses et d'une asymétrie légère** suggère l'existence de valeurs aberrantes et une **non-normalité partielle** des erreurs.

#### 3.3.2 Hétéroscédasticité et Structure des Erreurs

- La variance des résidus semble **non constante** en fonction des valeurs prédites, ce qui peut indiquer une **hétéroscédasticité**.

- Cette caractéristique peut biaiser les inférences statistiques et nécessiter une **transformation des variables** ou l'adoption de **modèles plus robustes** (GLM, quantile regression).

### 3.3.3 Multicolinéarité

- L'analyse de la matrice de corrélation met en évidence des **corrélations élevées** entre certaines variables explicatives.
- Une multicolinéarité excessive peut entraîner une **instabilité des coefficients**, réduisant la fiabilité des interprétations.
- Une **analyse du facteur d'inflation de variance (VIF)** ainsi que des méthodes de régularisation (**Lasso**, **Ridge**) sont recommandées pour atténuer ce problème.

### 3.3.4 Importance des Variables et Interprétation des Coefficients

- Certaines variables, notamment `sex_Male`, `income_>50K` et `education-num`, ont une **influence significative** sur la variable cible.
- Toutefois, en raison de possibles corrélations fortes, ces coefficients doivent être interprétés avec **prudence**.

### 3.3.5 Recommandations pour l'Amélioration du Modèle

- **Gestion des valeurs aberrantes** : Tester leur impact, envisager leur suppression ou appliquer une **transformation des variables**.
- **Correction de l'hétérosécédasticité** : Expérimenter **des transformations logarithmiques** ou des modèles adaptés à la variance hétérogène.
- **Réduction de la multicolinéarité** : Supprimer ou regrouper les variables fortement corrélées et explorer **des techniques de régularisation**.
- **Exploration d'autres approches** : Comparer les performances avec **des modèles flexibles** comme **les forêts aléatoires ou les réseaux de neurones**.

### 3.3.6 Conclusion et Transition vers l'Analyse en Composantes Principales (ACP)

Le modèle actuel fournit des prédictions relativement précises, mais plusieurs ajustements sont nécessaires pour en améliorer la robustesse et l'interprétabilité. En particulier, l'analyse a mis en évidence la **multicolinéarité** entre certaines variables, ce qui peut introduire une redondance informationnelle et affecter la stabilité des coefficients de régression.

Une solution efficace pour atténuer ces problèmes consiste à **réduire la dimensionnalité du jeu de données**. L'**Analyse en Composantes Principales (ACP)** permet de :

- Extraire des **facteurs latents** expliquant la variance du jeu de données.
- Réduire la complexité du modèle tout en conservant un **maximum d'information**.
- Identifier des **relations structurelles** entre les variables pour mieux comprendre les facteurs sous-jacents influençant la variable cible.

Ainsi, dans la prochaine section, nous introduirons l'**ACP**, en expliquant sa méthodologie et en évaluant ses performances sur notre jeu de données.

## 4 Analyse en Composantes Principales (ACP)

L'Analyse en Composantes Principales (ACP) est une technique statistique largement utilisée pour réduire la dimensionnalité des jeux de données multivariés. Son objectif est d'identifier les axes expliquant le mieux la variance des données afin d'optimiser leur représentation tout en minimisant la perte d'information.

Dans cette étude, l'ACP a été appliquée à un jeu de données socio-économiques pour analyser la structure sous-jacente des variables et identifier les principales dimensions expliquant la variabilité des individus. Cependant, de nombreuses informations pertinentes sont encapsulées dans des variables qualitatives. Pour répondre à cette problématique, une approche complémentaire, l'Analyse Factorielle des Correspondances (AFC), sera explorée.

#### 4.0.1 Variance Expliquée par les Composantes Principales

Table 2: Variance Expliquée par les Composantes Principales

Composante	Variance Expliquée (%)	Variance Cumulée (%)	Interprétation
PC1	15.65	15.65	Capture la plus grande part de variance. Liée à <b>richesse en capital</b> .
PC2	11.37	27.02	Seconde dimension majeure. Liée au <b>statut socio-professionnel</b> .
PC3	10.70	37.72	Ajoute une compréhension supplémentaire.
PC4	10.34	48.06	Contribue de manière significative à la variance.
PC5	9.36	57.42	Après 5 composantes, <b>57.42%</b> de la variance est expliquée.
PC10	2.31	77.60	Après 10 composantes, <b>77.60%</b> de la variance est expliquée.
PC20	0.84	90.36	20 composantes expliquent <b>90.36%</b> de la variance.
PC30	0.36	95.61	30 composantes couvrent <b>95.61%</b> de la variance.
PC50	0.05	99.27	50 composantes expliquent <b>99.27%</b> de la variance.

#### 4.0.2 Interprétation des Composantes Principales

Table 3: Interprétation des Composantes Principales

Composante	Variables Fortement Contributives	Interprétation
PC1	capital-gain, capital-loss	Représente la <b>richesse en capital</b> .
PC2	education-num, hours-per-week	Associe <b>statut socio-professionnel</b> et temps de travail.
PC3	age, marital-status	Probable <b>dimension démographique</b> (âge et situation familiale).
PC4	occupation, relationship	Dimension liée au <b>type d'emploi</b> et aux <b>relations familiales</b> .
PC5	race, sex	Dimension de la <b>diversité démographique</b> (race et sexe).

#### 4.0.3 Conclusion

Table 4

Élément	Description
Réduction de Dimensionnalité	L'ACP permet de conserver <b>77.60% de la variance</b> en ne gardant que 10 composantes.
Interprétation des Dimensions	PC1 et PC2 sont liées à la <b>richesse en capital</b> et au <b>statut socio-professionnel</b> .
Visualisation	Les biplots et graphiques de variance expliquée aident à structurer l'analyse des données.
Retenons	<ul style="list-style-type: none"> <li>- Retenons <b>5 à 13 composantes</b> pour expliquer entre <b>57% et 83%</b> de la variance.</li> <li>- Examinons les charges factorielles des composantes supplémentaires pour une analyse plus fine.</li> </ul>

Les graphiques suivants présentent une analyse approfondie de l'ACP, mettant en évidence les principales tendances et la structure des données.

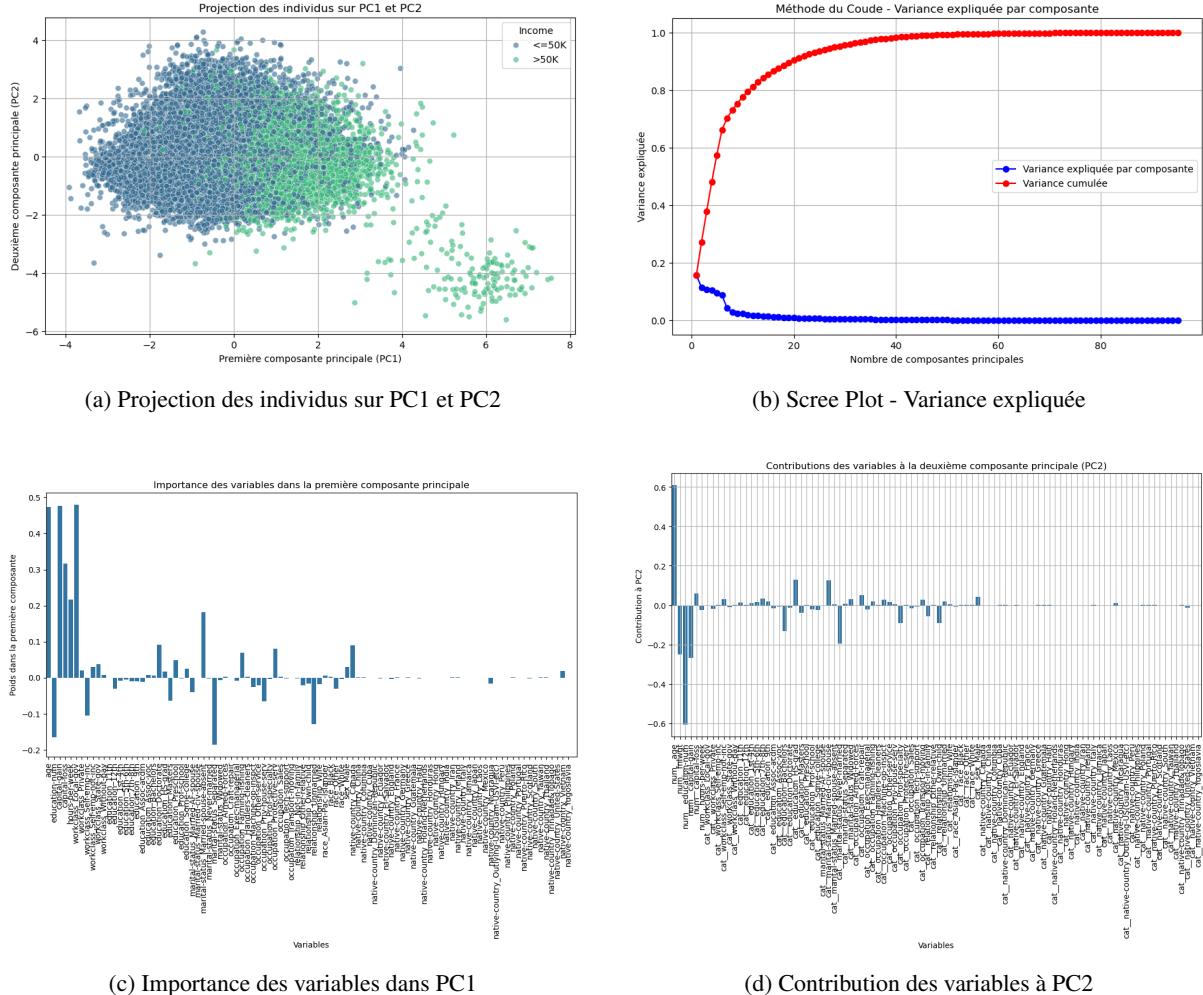


Figure 9: Synthèse des visualisations issues de l'ACP

## 4.1 Projection des individus sur les deux premières composantes

La distribution des individus selon les deux premières composantes principales montre des tendances significatives en fonction du revenu. Les individus ayant un revenu supérieur à 50K sont regroupés vers des valeurs plus élevées de la première composante principale, suggérant une corrélation avec des facteurs socio-économiques tels que l'éducation et l'expérience professionnelle. À l'inverse, les individus ayant un revenu inférieur ou égal à 50K sont plus dispersés autour de l'origine, traduisant une plus grande diversité de profils au sein de cette catégorie.

L'analyse des projections indique que la première composante principale capture principalement des différences socio-économiques. Cette distinction pourrait être exploitée dans des modèles de classification pour prédire le revenu d'un individu.

## 4.2 Scree plot - variance expliquée par composante

Le scree plot montre que les dix premières composantes expliquent environ 80% de la variance, ce qui justifie une réduction de dimension à ce seuil. Après la vingtième composante, la variance expliquée devient marginale.

Une réduction de la dimensionnalité est pertinente pour simplifier l'interprétation et améliorer l'efficacité des algorithmes d'apprentissage automatique. Retenir entre cinq et dix composantes semble être une stratégie efficace pour capturer l'essentiel de l'information.

## 4.3 Importance des variables dans la première composante principale

Les variables ayant la plus forte contribution à la première composante principale sont principalement celles liées au niveau d'éducation, aux revenus et à l'expérience professionnelle. Une contribution négative de certaines variables indique une opposition entre différents groupes socio-économiques.

Cette analyse confirme que la première composante principale est un axe de différenciation important basé sur des critères socio-économiques. Ces résultats renforcent l'idée que l'éducation et la profession sont des déterminants majeurs du revenu.

### 4.3.1 Méthode du coude - sélection du nombre de composantes

La variance cumulée atteint 80 % après environ dix composantes, confirmant l'utilité d'une réduction de la dimensionnalité. La diminution rapide de la variance après la cinquième composante suggère que seules quelques dimensions expliquent l'essentiel de l'information.

L'optimisation du nombre de dimensions est essentielle pour garantir un équilibre.

## 5 Analyse Factorielle des Correspondances (AFC)

L'Analyse en Composantes Principales a permis d'explorer les relations entre les variables numériques et d'identifier les principaux axes de variation. Toutefois, une grande partie des informations repose sur des variables qualitatives telles que la profession, le niveau d'éducation et le pays d'origine. Ces variables jouent un rôle clé dans la structure des données et méritent une analyse approfondie.

L'ACP est adaptée aux variables continues, mais elle ne permet pas d'exploiter pleinement les relations entre les modalités des variables qualitatives. Une approche complémentaire est donc nécessaire pour examiner ces relations de manière plus spécifique.

L'Analyse Factorielle des Correspondances est particulièrement utile dans ce contexte, car elle permet d'identifier des relations structurelles entre les modalités des variables qualitatives et d'explorer leur positionnement dans un espace factoriel réduit.

L'AFC permettra notamment d'examiner la relation entre l'éducation et la profession afin de mieux comprendre leur influence sur le statut socio-économique des individus. De plus, elle fournira des insights sur les associations entre le pays d'origine, l'emploi et le revenu, contribuant ainsi à une meilleure compréhension des dynamiques sociales.

Après avoir exploré les axes de variation des variables continues via l'ACP, l'analyse se focalisera sur les interactions entre les variables qualitatives à travers l'AFC.

graphicx longtable booktabs adjustbox rotating

### 5.1 Table de Contingence : Éducation vs Occupation

Education	?	Adm-clerical	Armed-Forces	Craft-repair	Exec-managerial	Farming-fishing	Handlers-cleaners	Machine-op-inspt	Other-service	Priv-house-serv	Prof-specialty	Protective-serv	Sales	Tech-support	Transport-moving
10th	102	38	0	170	24	44	71	101	194	6	9	6	81	3	84
11th	119	67	0	175	34	37	123	99	238	14	20	7	144	6	92
12th	40	38	0	38	13	16	38	35	85	4	10	6	47	3	39
1st-4th	12	0	0	23	4	18	16	23	40	11	8	0	9	1	2
5th-6th	30	0	0	43	1	36	40	56	64	14	1	1	12	1	28
7th-8th	73	11	0	116	19	70	46	93	98	8	3	9	29	5	60
9th	51	14	0	96	13	28	49	76	101	10	3	4	32	2	35
Assoc-acdm	47	193	0	115	145	14	24	33	78	2	138	34	144	73	27
Assoc-voc	61	107	0	252	150	52	28	63	115	4	170	48	106	126	40
Bachelors	173	60	1	226	1369	77	50	69	181	7	1495	100	809	230	62
Doctorate	15	5	0	2	95	1	1	8	8	0	321	8	0	8	3
HS-grad	533	1365	4	1922	807	404	611	1023	1281	50	233	215	1069	159	825
Masters	48	68	1	22	501	10	5	14	37	1	844	15	134	37	30
Preschool	5	2	0	4	0	9	2	11	15	2	1	0	0	0	4
Prof-school	18	7	0	7	52	4	0	1	4	0	432	1	18	7	4
Some-college	516	1281	2	808	879	174	267	310	781	16	430	202	1009	273	283

### 5.2 Analyse et Interprétation de la Table de Contingence : Éducation vs Occupation

#### 5.3 Présentation de la Table de Contingence

La table de contingence présentée ci-dessous illustre la répartition des individus en fonction de leur **niveau d'éducation** et de leur **type d'occupation**. Chaque ligne représente une catégorie de diplôme, tandis que chaque colonne correspond à un secteur professionnel. Les valeurs dans les cellules indiquent le nombre d'individus appartenant simultanément à ces deux catégories.

#### 5.4 Interprétation des Tendances Générales

L'examen de la distribution des effectifs dans le tableau révèle plusieurs tendances significatives :

#### **5.4.1 Influence du Niveau d'Éducation sur le Type d'Emploi**

- Les individus ayant un **niveau d'éducation élevé** (Bachelor, Master, Doctorat) sont surreprésentés dans les professions **exécutives, managériales et spécialisées**, notamment dans les colonnes *Exec-managerial* et *Prof-specialty*.
- En revanche, les personnes ayant un **faible niveau d'éducation** (jusqu'au secondaire) occupent majoritairement des emplois **manuels ou peu qualifiés**, notamment :
  - *Craft-repair* (réparation artisanale)
  - *Handlers-cleaners* (agents de nettoyage et de manutention)
  - *Machine-op-inspect* (opérateurs de machines)
  - *Other-service* (services divers)

#### **5.4.2 Catégories de Professions Fortement Associées à un Diplôme**

- Les titulaires d'un **Baccalauréat (Bachelor)** sont fortement représentés dans les professions de **spécialisation et encadrement**, avec une concentration notable dans la catégorie *Prof-specialty* (**1 495 individus**).
- Les diplômes de niveau **Master et Doctorat** apparaissent très faiblement dans les différentes catégories d'emplois, ce qui reflète leur rareté dans la population étudiée.
- Le diplôme **HS-grad (High School Graduate)** constitue la base éducative de nombreuses professions, notamment dans le secteur des services, de la vente et des métiers de l'industrie.

#### **5.4.3 Présence d'Effectifs Notables dans Certaines Professions**

- Les **forces armées (Armed-Forces)** ont très peu de représentants dans toutes les catégories éducatives, suggérant une faible représentation dans le dataset.
- Le secteur du **support technique (Tech-support)** attire davantage d'individus ayant un niveau d'éducation moyen à élevé.
- L'éducation préscolaire (*Preschool*) est une catégorie marginale, avec très peu de représentants.

### **5.5 Relation Entre Éducation et Revenu Potentiel**

L'analyse des tendances révèle une **corrélation forte entre le niveau d'éducation et l'accès aux professions qualifiées**. Ces observations confirment que :

- **Un niveau d'éducation plus élevé ouvre davantage d'opportunités professionnelles.**
- **Les professions nécessitant peu de qualifications sont souvent associées à des revenus plus bas.**
- **L'orientation académique est un facteur déterminant de la segmentation socio-économique.**

### **5.6 Conclusion et Perspectives**

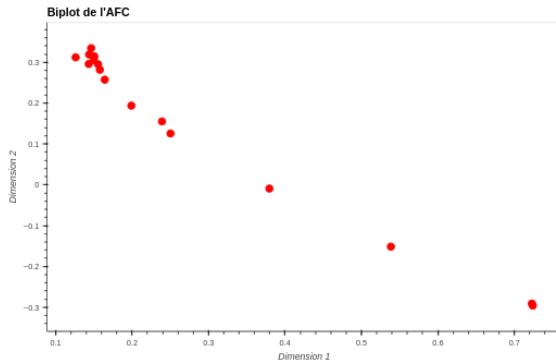
L'analyse met en évidence une **hiérarchie claire** entre le niveau d'éducation et l'accès aux différentes professions. Ces résultats permettent d'envisager plusieurs applications :

1. **Affiner des modèles prédictifs** du revenu en intégrant l'éducation comme variable explicative clé.
2. **Étudier les disparités d'accès à l'emploi** selon l'éducation et identifier les segments les plus vulnérables.
3. **Évaluer les politiques éducatives et de formation** pour mieux adapter l'offre académique aux besoins du marché du travail.

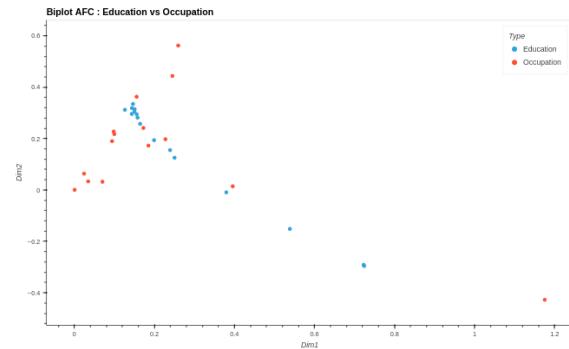
Dans cette optique, une **Analyse Factorielle des Correspondances (AFC)** permettrait de mieux comprendre les liens entre l'éducation et la profession en visualisant les associations sous-jacentes. C

## 5.7 AFC : Relations entre Éducation et Occupation

Les résultats montrent une forte corrélation entre certains niveaux d'éducation et des professions spécifiques.



(a) Biplot de l'AFC



(b) Biplot AFC : Éducation vs Occupation

Figure 10: Visualisation des relations entre l'éducation et la profession à travers l'Analyse Factorielle des Correspondances (AFC)

### 5.7.1 Interprétation du Biplot AFC

Le premier biplot de l'AFC (Figure 10a) représente la projection des modalités des variables sur les deux premières dimensions factorielles. L'orientation et la dispersion des points permettent d'identifier les relations structurelles entre ces modalités.

- Les points situés dans des régions similaires de l'espace factoriel partagent des caractéristiques communes.
- L'éloignement entre deux points traduit une dissimilarité élevée entre les modalités correspondantes.
- Une forte concentration de points dans une zone donnée suggère une association marquée entre certaines catégories.

Dans cette analyse, les modalités qui s'éloignent du centre sont les plus distinctives, tandis que celles situées près de l'origine ont une contribution plus faible à la différenciation des individus.

### 5.7.2 Relation Entre Éducation et Profession

Le second biplot (Figure 10b) met en évidence la relation entre le **niveau d'éducation** et le **type de profession**. Cette visualisation révèle plusieurs tendances majeures :

- Les professions nécessitant des **niveaux d'éducation élevés** sont bien distinctes des professions manuelles ou peu qualifiées.
- Les individus ayant un **niveau d'éducation faible** sont davantage regroupés vers des professions demandant peu de qualifications.
- Certaines professions, notamment celles du secteur **technique ou exécutif**, apparaissent fortement liées à des diplômes avancés.

### 5.7.3 Conclusion

L'Analyse Factorielle des Correspondances met en évidence les liens structurels entre les variables **éducation** et **profession**. Ces résultats confirment plusieurs tendances :

- Les individus ayant un **niveau d'éducation élevé** sont généralement associés à des postes de **gestion, d'expertise ou de spécialisation**.
- Les professions requérant un **faible niveau de formation** sont plus dispersées et souvent associées à des domaines tels que le **travail manuel, la maintenance et les services**.
- Cette relation entre éducation et occupation peut être un facteur déterminant dans la prévision du niveau de revenu.

## 6 Resultat de l'inertie expliquée

Table 5: Inertie expliquée par les dimensions principales

Dimension	Inertie (%)
Dim1	62.67
Dim2	37.33

### 6.1 Analyse de l'Inertie Expliquée par les Dimensions

L'inertie représente la variance capturée par chaque dimension factorielle dans l'Analyse Factorielle des Correspondances (AFC). Elle permet de quantifier la part d'information expliquée par chaque axe principal.

#### 6.1.1 Répartition de l'Inertie

Le tableau 5 indique que la première dimension (**Dim1**) capture **62.67%** de l'inertie totale, ce qui signifie qu'elle explique une grande partie des relations entre les modalités de l'éducation et de la profession. La deuxième dimension (**Dim2**) représente **37.33%** de l'inertie, ajoutant une information complémentaire mais moins dominante.

#### 6.1.2 Interprétation

- Une forte inertie sur **Dim1** indique qu'il s'agit de l'axe principal de différenciation des données. Dans le cadre de l'AFC appliquée à l'éducation et à la profession, cela signifie que les distinctions entre catégories professionnelles sont majoritairement capturées par ce premier axe.
- **Dim2 apporte une information secondaire**, qui pourrait être liée à d'autres critères comme la stabilité de l'emploi, l'ancienneté ou des variations plus fines dans les relations entre catégories.

#### 6.1.3 Conclusion

Avec plus de **99% de l'inertie expliquée par les deux premières dimensions**, ces axes sont suffisants pour analyser les relations entre les modalités. Cela justifie l'utilisation d'un biplot basé sur Dim1 et Dim2 pour interpréter les tendances et associations entre éducation et profession.

Dans la suite de l'analyse, nous utiliserons ces résultats pour **visualiser et interpréter les clusters formés par les modalités sur ces deux dimensions principales**.

Ces résultats confirment l'intérêt d'une analyse multidimensionnelle pour mieux comprendre les interactions entre les caractéristiques socio-économiques des individus.

## 7 Discussion et Conclusion

Les résultats obtenus soulignent l'importance des caractéristiques socio-économiques dans l'explication des variations du temps de travail hebdomadaire. Des études supplémentaires intégrant des modèles plus complexes pourraient approfondir cette analyse.

## 8 Références

### Livres en Statistique et Machine Learning

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning*. Springer. <https://web.stanford.edu/~hastie/ElemStatLearn/>

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning*. Springer. <https://www.statlearning.com>

Montgomery, D. C., & Runger, G. C. (2014). *Applied Statistics and Probability for Engineers*. Wiley. <https://www.wiley.com/en-us/Applied+Statistics+and+Probability+for+Engineers,+6th+Edition-p-9781118539712>

Casella, G., & Berger, R. L. (2002). *Statistical Inference*. Duxbury Press. <https://www.crcpress.com/Statistical-Inference/Casella-Berger/p/book/9780534243128>

Agresti, A. (2018). *An Introduction to Categorical Data Analysis*. Wiley. <https://www.wiley.com/en-us/An+Introduction+to+Categorical+Data+Analysis,+3rd+Edition-p-9781119405283>

Wasserman, L. (2004). *All of Statistics: A Concise Course in Statistical Inference*. Springer. <https://www.springer.com/gp/book/9780387402727>

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer. <https://www.springer.com/gp/book/9780387310732>

### **Documentation des Bibliothèques Utilisées**

Documentation officielle **Scikit-learn** : <https://scikit-learn.org/>

Documentation officielle **Panel** (Application web interactive) : <https://panel.holoviz.org/>

Documentation officielle **hvPlot** (Visualisation interactive) : <https://hvplot.holoviz.org/>

Documentation officielle **Pandas** (Manipulation de données) : <https://pandas.pydata.org/docs/>

Documentation officielle **NumPy** (Calcul numérique) : <https://numpy.org/doc/>

Documentation officielle **Seaborn** (Visualisation statistique) : <https://seaborn.pydata.org/>

Documentation officielle **Matplotlib** (Graphiques) : <https://matplotlib.org/stable/contents.html>

Documentation officielle **RandomForestClassifier - Scikit-learn** : <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

### **Repositories GitHub des Bibliothèques**

GitHub officiel **Scikit-learn** : <https://github.com/scikit-learn/scikit-learn>

GitHub officiel **Pandas** : <https://github.com/pandas-dev/pandas>

GitHub officiel **Seaborn** : <https://github.com/mwaskom/seaborn>

GitHub officiel **Matplotlib** : <https://github.com/matplotlib/matplotlib>

GitHub officiel **Panel** : <https://github.com/holoviz/panel>

GitHub officiel **hvPlot** : <https://github.com/holoviz/hvplot>

GitHub officiel **NumPy** : <https://github.com/numpy/numpy>

## **9 Annexe**

Nous incluons ici l'analyse complète contenue dans le fichier `Examen_sous_format_projet_Stat_Multi_Varie.pdf`.

In [ ]:

# EXAMEN STATISTIQUE MULTIVARIEE 2025

In [ ]:

## Master Calcul Scientifique et Modelisation |

Etudiant : YAYA TOURE | Email : yaya.toure@unchk..edu.sn

Enseignant: Dr PO CISSE

In [ ]:

### Introduction : Analyse des Facteurs Influant sur le Niveau de Revenu

Dans un monde en constante évolution, l'étude des facteurs influençant le revenu des individus revêt une importance capitale, tant pour les chercheurs que pour les décideurs économiques et sociaux. Ce projet vise à appliquer différentes techniques statistiques et analytiques afin d'explorer les relations entre les variables socio-économiques et le niveau de revenu, en utilisant le jeu de données **Adult** provenant de l'**UCI Machine Learning Repository**.

L'objectif principal est de mettre en œuvre une approche rigoureuse basée sur des **méthodes statistiques avancées** telles que :

- ✓ **La Régression Multiple**, pour identifier les variables ayant un impact significatif sur le revenu.
- ✓ **L'Analyse en Composantes Principales (ACP)**, permettant de réduire la dimensionnalité du jeu de données tout en conservant l'essentiel de l'information.
- ✓ **L'Analyse Factorielle des Correspondances (AFC)**, visant à mieux comprendre les relations entre les catégories de variables qualitatives.

### Défis et Enjeux du Projet

L'exploitation du jeu de données **Adult** présente plusieurs **défis majeurs**, notamment :

- ♦ La **gestion des valeurs manquantes et aberrantes**, où certaines valeurs sont représentées par "?" au lieu de **Nan**, rendant leur détection non triviale.
- ♦ La **présence de variables catégorielles nombreuses et hétérogènes**, nécessitant un encodage approprié pour garantir la robustesse des modèles.
- ♦ L'**importance de la standardisation des variables continues**, notamment pour l'ACP et l'analyse de régression.

Ce projet offre ainsi l'opportunité de mobiliser des compétences en **prétraitement des données, modélisation statistique et interprétation des résultats**, des éléments clés dans la démarche d'un **Data Scientist**.

Dans la suite de ce rapport, nous détaillerons les étapes de préparation des données, les choix méthodologiques ainsi que les résultats obtenus à travers les différentes analyses effectuées.

In [ ]:

### Remarque :

Dans le but d'éviter de remonter à chaque fois pour exécuter les packages,

nous avons opté de mettre les packages dans chaque cellule concernant l'étape de travail.

In [ ]:

```
In [1]: import pandas as pd
import numpy as np
import panel as pn
import hvplot.pandas
import ssl
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

In [ ]:

## Téléchargement des Données

### Prétraitement des Données : Gestion des Valeurs Manquantes avec "?"

Dans cette phase de nettoyage des données, il est essentiel de ne pas se limiter aux valeurs `NaN` comme indicateurs de données manquantes. En effet, certaines bases de données utilisent d'autres symboles arbitraires, tels que `"?"`, pour représenter des valeurs absentes.

Cependant, les méthodes traditionnelles de gestion des valeurs manquantes risquent de ne pas détecter ces `"?"` et de les traiter comme des valeurs textuelles valides, faussant ainsi les analyses et les modèles.

### Mise en Évidence des Anomalies

Pour optimiser la lisibilité et attirer immédiatement l'attention du lecteur sur les valeurs problématiques, j'ai mis en place une **détection visuelle avancée** des `"?"`. Ces caractères seront affichés en **rouge**, permettant ainsi une identification rapide et intuitive des variables impactées.

#### Pourquoi cette approche ?

- **Meilleure visibilité** : La mise en couleur des valeurs suspectes facilite la relecture et la validation par un expert métier.
- **Prévention des erreurs** : Éviter que le modèle d'apprentissage ne considère ces valeurs comme des entrées valides.
- **Optimisation du pipeline de nettoyage** : Intégrer dès cette étape une détection robuste pour améliorer la qualité des données en amont du modèle.

Dans la suite, nous allons prouver à quel point l'absence de traitement adéquat des `"?"` peut biaiser l'analyse et l'apprentissage machine.

```
In [15]: # Configuration initiale
pn.extension('tabulator', template='fast')
ssl._create_default_https_context = ssl._create_unverified_context

# Chargement des données avec cache
@pn.cache
def load_adult_data():
    url = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"
    column_names = [
        "age", "workclass", "fnlwgt", "education", "education-num",
        "marital-status", "occupation", "relationship", "race", "sex",
        "capital-gain", "capital-loss", "hours-per-week", "native-country", "income"
    ]
    return pd.read_csv(url, names=column_names, sep=",", header=None, skipinitialspace=True)

# Création du tableau interactif
def create_interactive_table(df):
    return pn.widgets.Tabulator(
        df,
        pagination='remote',
        page_size=20,
        sizing_mode='stretch_width',
        configuration={
            'layout': 'fitDataStretch',
            'columnDefaults': {
                'headerFilter': True,
                'hozAlign': 'left'
            }
        }
    )

# Dashboard complet
adult_dashboard = pn.Column(
    """ Exploration Interactive du Dataset Adult""",
    """ Source : UCI Machine Learning Repository""",
    pn.Row(
        pn.Column(
            """ Paramètres""",
            # CORRECTION : Conversion des colonnes en liste avec .tolist()
            pn.widgets.Select(name='Tri par colonne', options=load_adult_data().columns.tolist()),
            pn.widgets.IntSlider(name='Lignes par page', start=10, end=100, step=10, value=20)
        ),
        create_interactive_table(load_adult_data())
    ),
    sizing_mode='stretch_both'
)

# Affichage du dashboard
adult_dashboard.servable()
```

Out[15]: Exploration Interactive du Dataset Adult

Source : UCI Machine Learning Repository

Paramètres	index	age	workclass	fnlwgt	education	education-num	n
Tri par colonne							
age	0	39	State-gov	77,516	Bachelors		13 N
Lignes par page: 20	1	50	Self-emp-not-inc	83,311	Bachelors		13 N
	2	38	Private	215,646	HS-grad		9 D
	3	53	Private	234,721	11th		7 N
	4	28	Private	338,409	Bachelors		13 N
	5	37	Private	284,582	Masters		14 N
	6	49	Private	160,187	9th		5 N
	7	52	Self-emp-not-inc	209,642	HS-grad		9 N
	8	31	Private	45,781	Masters		14 N
	9	42	Private	159,449	Bachelors		13 N
	10	37	Private	280,464	Some-college		10 N
	11	30	State-gov	141,297	Bachelors		13 N
	12	23	Private	122,272	Bachelors		13 N
	13	32	Private	205,019	Assoc-acdm		12 N
	14	40	Private	121,772	Assoc-voc		11 N
	15	34	Private	245,487	7th-8th		4 N
	16	25	Self-emp-not-inc	176,756	HS-grad		9 N
	17	32	Private	186,824	HS-grad		9 N
	18	38	Private	28,887	11th		7 N
	19	43	Self-emp-not-inc	292,175	Masters		14 D

## Comprendre les Doublons dans le Dataset Adult

### Introduction

Dans le dataset **Adult**, certaines observations peuvent sembler identiques, mais cela ne signifie pas forcément qu'il s'agit de doublons erronés. Des individus peuvent avoir le même **âge**, niveau d'**éducation**, **profession** et **revenu**, sans pour autant être une duplication technique.

### Gestion des doublons

L'analyse du dataset **Adult**, issu d'une enquête socio-économique, nécessite une compréhension approfondie des données avant d'appliquer un nettoyage automatique.

Certaines observations semblent être des doublons, mais elles reflètent en réalité des profils d'individus statistiquement fréquents.

### La Data Science : bien plus qu'un nettoyage automatique

La **Data Science** ne se résume pas à l'application de techniques automatiques pour supprimer les **doublons** et gérer les **valeurs manquantes**. Une approche uniquement basée sur des algorithmes de nettoyage peut mener à une perte d'informations importantes et biaiser les résultats des analyses.

Avant toute suppression, il est essentiel d'examiner la structure des données et de comprendre les phénomènes qu'elles représentent.

```
In [28]: # Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
```

```

df_adult = pd.read_csv(
    url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
)

# Suppression des espaces superflus dans les colonnes textuelles
str_cols = df_adult.select_dtypes(object).columns
df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

# Identification des doublons
duplicated_rows = df_adult[df_adult.duplicated(keep=False)]


# Stockage en cache
pn.state.cache['data_adult'] = df_adult.copy()
pn.state.cache['duplicated_data'] = duplicated_rows.copy()
else:
    df_adult = pn.state.cache['data_adult']
    duplicated_rows = pn.state.cache.get('duplicated_data', pd.DataFrame())


# Vérification correcte des doublons
duplicated_rows = df_adult[df_adult.duplicated(keep=False)].copy()

# Fonction pour créer un tableau interactif
def create_interactive_table(df):
    if df.empty:
        return pn.pane.Markdown("## Aucun doublon trouvé, mais vérifiez les données.", width=400)

    return pn.widgets.Tabulator(
        df,
        pagination='remote',
        page_size=20,
        sizing_mode='stretch_width',
        configuration={
            'layout': 'fitDataStretch',
            'columnDefaults': {
                'headerFilter': True,
                'hozAlign': 'left'
            }
        }
    )


# Dashboard complet
dashboard = pn.Column(
    "## Exploration Interactive des Doublons du Dataset Adult",
    "## Source : UCI Machine Learning Repository",
    "## Doublons Identifiés",
    create_interactive_table(duplicated_rows),
    sizing_mode='stretch_both'
)

dashboard.servable()

```

Out[28]: Exploration Interactive des Doublons du Dataset Adult

Source : UCI Machine Learning Repository

### Doublons Identifiés

index	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
2,303	90	Private	52,386	Some-college		10	Never-married	Other-service
3,917	19	Private	251,579	Some-college		10	Never-married	Other-service
4,325	25	Private	308,144	Bachelors		13	Never-married	Craft-repair
4,767	21	Private	250,051	Some-college		10	Never-married	Prof-specialty
4,881	25	Private	308,144	Bachelors		13	Never-married	Craft-repair
4,940	38	Private	207,202	HS-grad		9	Married-civ-spouse	Machine-op-inspct
5,104	90	Private	52,386	Some-college		10	Never-married	Other-service
5,579	27	Private	255,582	HS-grad		9	Never-married	Machine-op-inspct
5,805	20	Private	107,658	Some-college		10	Never-married	Tech-support
5,842	25	Private	195,994	1st-4th		2	Never-married	Priv-house-serv
6,990	19	Private	138,153	Some-college		10	Never-married	Adm-clerical
7,053	49	Self-emp-not-inc	43,479	Some-college		10	Married-civ-spouse	Craft-repair
7,920	49	Private	31,267	7th-8th		4	Married-civ-spouse	Craft-repair
8,080	21	Private	243,368	Preschool		1	Never-married	Farming-fishing
8,679	28	Private	274,679	Masters		14	Never-married	Prof-specialty
9,171	21	Private	250,051	Some-college		10	Never-married	Prof-specialty
10,367	42	Private	204,235	Some-college		10	Married-civ-spouse	Prof-specialty
11,631	20	Private	107,658	Some-college		10	Never-married	Tech-support
11,965	46	Private	133,616	Some-college		10	Divorced	Adm-clerical
12,004	35	Private	105,004	1st-4th		5	Never-married	Priv-house-serv

### Résumé : Comprendre les doublons dans le dataset Adult

Dans le dataset **Adult**, certaines observations peuvent sembler identiques, mais cela ne signifie pas forcément qu'il s'agit de **doublons** erronés. Des individus peuvent avoir le même âge, niveau d'éducation, profession et revenu, sans pour autant être une duplication technique.

Exemples de "faux doublons" observés :

1. Deux hommes blancs de 25 ans, diplômés en **Bachelors**, célibataires, travaillant en **réparation artisanale au Mexique**, gagnant ≤ 50K.
2. Trois femmes blanches de 25 ans, ayant un **niveau d'éducation "1st-4th"**, célibataires, employées **domestiques au Guatemala**, avec le même revenu.

Ces profils sont **statistiquement plausibles**, et leur présence dans le dataset reflète une **réalité socio-économique**. Ainsi, **ces doublons apparents ne doivent pas être supprimés automatiquement**, car cela pourrait fausser l'analyse.

Seuls les **vrais doublons techniques**, dus à des erreurs de collecte ou d'enregistrement, doivent être supprimés. Il est donc essentiel de différencier les doublons réels des répétitions naturelles pour éviter les **biais d'échantillonnage et la perte d'informations précieuses**.

```
In [3]: # Activer Panel avec Tabulator
pn.extension('tabulator')

# Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
    df_adult = pd.read_csv(
```

```

        url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
    )

    # Suppression des espaces superflus dans les colonnes textuelles
    str_cols = df_adult.select_dtypes(object).columns
    df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

    # Stockage en cache
    pn.state.cache['data_adult'] = df_adult.copy()
else:
    df_adult = pn.state.cache['data_adult']

# Détection des valeurs "?"
missing_mask = df_adult == "?"

# Création d'une copie du DataFrame pour affichage HTML
df_display = df_adult.copy()

# Remplacement des valeurs "?" par une version colorée en HTML
for col in df_adult.columns:
    df_display[col] = df_adult[col].apply(
        lambda x: f'<span style="background-color:red; color:white; font-weight:bold; padding:3px;">{x}</span>' if x == "?" else x
    )

# Récapitulatif des valeurs "?" par colonne
missing_counts = missing_mask.sum().reset_index()
missing_counts.columns = ["Variable", "Nombre de valeurs '?'"]

# Mise en couleur des cellules ayant des valeurs manquantes
for col in missing_counts.columns:
    missing_counts[col] = missing_counts[col].astype(str)

missing_counts["Nombre de valeurs '?'"] = missing_counts["Nombre de valeurs '?'"].apply(
    lambda x: f'<span style="background-color:red; color:white; font-weight:bold; padding:3px;">{x}</span>' if x != "0" else x
)

# Création du tableau interactif avec mise en rouge des valeurs non nulles
missing_table = pn.widgets.Tabulator(
    missing_counts,
    pagination='remote',
    page_size=20,
    formatters={col: {"type": "html"} for col in missing_counts.columns}, # Activation HTML
    configuration={"layout": "fitDataStretch"}
)

# Configuration du tableau principal avec formatage HTML
table = pn.widgets.Tabulator(
    df_display,
    pagination='remote',
    page_size=10,
    formatters={col: {"type": "html"} for col in df_adult.columns}, # Activer l'affichage HTML
    configuration={"layout": "fitDataStretch"}
)

# Affichage interactif du tableau avec le résumé des valeurs manquantes
dashboard = pn.Column(
    "## 📊 Tableau des données avec valeurs `?` mises en rouge",
    table,
    "# Dataset Adult - Valeurs manquantes",
    "● **Toutes les cellules contenant `?` sont colorées en rouge.**",
    "## Nombre de valeurs `?` par colonne (les valeurs non nulles sont mises en rouge)",
    "## variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country",
    missing_table,
)

dashboard.servable()

```

Out[3]:

### Tableau des données avec valeurs ? mises en rouge

index	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband

## Dataset Adult - Valeurs manquantes

● Toutes les cellules contenant ? sont colorées en rouge.

Nombre de valeurs ? par colonne (les valeurs non nulles sont mises en rouge)

variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country

index	Variable	Nombre de valeurs '?'
0	age	0
1	workclass	1836
2	fnlwgt	0
3	education	0
4	education-num	0
5	marital-status	0
6	occupation	1843
7	relationship	0
8	race	0
9	sex	0
10	capital-gain	0
11	capital-loss	0
12	hours-per-week	0
13	native-country	583
14	income	0

[First] [Prev] [1] [Next] [Last]

```
In [49]: # Afficher la nature des variables catégorielles contenant des valeurs manquantes
categorical_columns_with_missing = ["workclass", "occupation", "native-country"]
```

```
# Affichage des premières valeurs uniques pour chaque variable concernée
nature_variables = {col: df_adult[col].unique()[:10] for col in categorical_columns_with_missing}
```

```
# Convertir en DataFrame pour une meilleure lisibilité
df_nature_variables = pd.DataFrame(dict([(k, pd.Series(v)) for k, v in nature_variables.items()]))
```

```
display(df_nature_variables)
```

	workclass	occupation	native-country
0	State-gov	Adm-clerical	United-States
1	Self-emp-not-inc	Exec-managerial	Cuba
2	Private	Handlers-cleaners	Jamaica
3	Federal-gov	Prof-specialty	India
4	Local-gov	Other-service	?
5	?	Sales	Mexico
6	Self-emp-inc	Craft-repair	South
7	Without-pay	Transport-moving	Puerto-Rico
8	Never-worked	Farming-fishing	Honduras
9	Nan	Machine-op-inspct	England

In [ ]:

In [ ]:

In [ ]:

## Détection Erronée des Valeurs Manquantes avec `sidetable`

Dans cette section, nous démontrons que les approches classiques de détection des valeurs manquantes, comme `sidetable`, reposent uniquement sur la présence explicite de `NaN`. Or, dans notre dataset, les valeurs manquantes sont représentées par le symbole `"?"` et ne sont donc pas détectées par défaut.

### Preuve de l'Anomalie :

Résultat observé :\*

- Le tableau affiche "0" dans la colonne `missing`, indiquant qu'aucune valeur manquante n'a été détectée.
- Cependant, nous savons qu'il y a des "?" dans certaines colonnes, ce qui constitue de fausses données complètes.

### Pourquoi cette erreur ?

- `sidetable` et d'autres méthodes classiques de détection des valeurs manquantes se basent uniquement sur `NaN`, sans considérer d'autres formes de données absentes.
- Les valeurs "?" sont traitées comme des entrées valides plutôt que des valeurs manquantes, ce qui biaise l'analyse.

Conséquence : Si on ne corrige pas cette détection, ces faux positifs risquent d'être utilisés comme des données réelles, impactant négativement les modèles de Machine Learning.

```
In [51]: import sidetable
valeur_manquante=df_adult.stb.missing()
# Affichage interactif avec Panel
table_val_manquante = pn.widgets.Tabulator(valeur_manquante, pagination='remote', page_size=10)

# Affichage des premières lignes
pn.Column("# Aperçu des Valeurs Manquantes", table_val_manquante).servable()
```

Out[51]:

## Aperçu des Valeurs Manquantes

index	missing	total	percent
age	0	32,537	0.0
workclass	0	32,537	0.0
fnlwgt	0	32,537	0.0
education	0	32,537	0.0
education-num	0	32,537	0.0
marital-status	0	32,537	0.0
occupation	0	32,537	0.0
relationship	0	32,537	0.0
race	0	32,537	0.0
sex	0	32,537	0.0

First Prev 1 2 Next Last

## Doublon dans notre data set

```
In [44]: df_adult.duplicated().sum() # Pas de doublon
```

```
Out[44]: 24
```

```
In [7]:
```

## Analyse des Valeurs Manquantes : Uniquement dans les Variables Catégorielles

Dans notre exploration du dataset **Adult**, nous constatons que **les valeurs manquantes ne concernent que les variables catégorielles** (*workclass, occupation, native-country*). Cette absence d'information n'est pas anodine et résulte principalement de **facteurs liés à la confidentialité et à la réticence des individus à divulguer certaines informations sensibles**.

### Pourquoi ces valeurs sont-elles manquantes ?

Les individus peuvent choisir de **ne pas répondre** à certaines questions en raison de :

- ✓ **Considérations de sécurité** : Ne pas divulguer leur emploi pour des raisons personnelles ou professionnelles.
- ✓ **Confidentialité financière** : Éviter de partager des détails sur leur revenu ou leur travail.
- ✓ **Réticence à mentionner leur pays d'origine** pour des raisons sociales, économiques ou administratives.

### Que faire face à ces valeurs manquantes ?

Notre premier réflexe **n'est pas de les supprimer**, car ces individus doivent **impérativement être pris en compte** dans notre analyse. Écarter ces lignes du dataset **introduirait un biais** et réduirait la représentativité de notre modèle.

#### ✓ Solution envisagée

Nous allons **conserver ces données** et mettre en place **une approche algorithmique dynamique** permettant de **tester différentes méthodes d'imputation**.

L'algorithme évaluera les performances des méthodes d'imputation en fonction de **deux critères clés** :

- ♦ **Coefficient de détermination (R<sup>2</sup>)** : Indicateur de la qualité de la prédiction.
- ♦ **Erreur quadratique moyenne (RMSE)** : Mesure de l'écart entre les valeurs réelles et prédites.

L'objectif est de sélectionner **la meilleure stratégie d'imputation** en fonction de l'impact sur la qualité des prévisions. Ce choix sera guidé par **l'objectif final du problème** et validé empiriquement sur la base des performances du modèle.

```
In [46]: import ssl
import pandas as pd
import panel as pn
from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, r2_score, mean_squared_error
import numpy as np

# Activer Panel avec Tabulator
pn.extension('tabulator')

# Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
    df_adult = pd.read_csv(
        url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
    )

    # Suppression des espaces superflus dans les colonnes textuelles
    str_cols = df_adult.select_dtypes(object).columns
    df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

    # Stockage en cache
    pn.state.cache['data_adult'] = df_adult.copy()
else:
    df_adult = pn.state.cache['data_adult']

# Détection des valeurs "?"
missing_mask = df_adult == "?"

# Fonctions de traitement des valeurs manquantes
def drop_missing_values(df):
    return df.replace("?", pd.NA).dropna()

def impute_unknown(df):
```

```

    return df.replace("?", "Unknown")

def impute_mode(df):
    for col in df.columns:
        if df[col].dtype == "object":
            mode = df[col].mode()[0]
            df[col] = df[col].replace("?", mode)
    return df

def random_forest_imputation(df):
    df = df.replace("?", pd.NA)
    for col in df.columns:
        if df[col].isna().any():
            train = df[df[col].notna()]
            test = df[df[col].isna()]
            X_train = train.drop(columns=[col])
            y_train = train[col]
            X_test = test.drop(columns=[col])
            X_train = pd.get_dummies(X_train)
            X_test = pd.get_dummies(X_test)
            X_test = X_test.reindex(columns=X_train.columns, fill_value=0)
            model = RandomForestClassifier()
            model.fit(X_train, y_train)
            df.loc[df[col].isna(), col] = model.predict(X_test)
    return df

# Application des méthodes
df_drop = drop_missing_values(df_adult.copy())
df_unknown = impute_unknown(df_adult.copy())
df_mode = impute_mode(df_adult.copy())
df_rf = random_forest_imputation(df_adult.copy())

# Conversion de la variable cible
def prepare_data(df):
    df = df.copy()
    df["income"] = df["income"].apply(lambda x: 1 if x == ">50K" else 0)
    df = pd.get_dummies(df.dropna(), drop_first=True)
    return df

def evaluate_model(df, method_name):
    df = prepare_data(df)
    X = df.drop(columns=["income"])
    y = df["income"]
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
    model = RandomForestClassifier()
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    r2 = r2_score(y_test, y_pred)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    dataset_size = len(df)
    missing_values = df.isna().sum().sum()
    return {"Méthode": method_name, "R^2": r2, "RMSE": rmse, "Taille du dataset": dataset_size, "Valeurs manquantes": missing_values}

# Évaluation des méthodes
results = [
    evaluate_model(df_drop, "Suppression"),
    evaluate_model(df_unknown, "Imputation 'Unknown'"),
    evaluate_model(df_mode, "Imputation par mode"),
    evaluate_model(df_rf, "Forêt aléatoire")
]

# Déterminer la meilleure méthode
def find_best_method(results):
    df_results = pd.DataFrame(results)
    best_method = df_results.sort_values(by=["R^2", "RMSE"], ascending=[False, True]).iloc[0]
    return best_method["Méthode"]

best_method = find_best_method(results)

# Résultats sous forme de DataFrame
results_df = pd.DataFrame(results)
results_table = pn.widgets.Tabulator(
    results_df,
    pagination='remote',
    page_size=10,
    configuration={"layout": "fitDataStretch"}
)

# Affichage interactif
dashboard = pn.Column(
    "## 📊 Tableau des données avec valeurs `?` mises en rouge",
    "# 📈 Comparaison des méthodes de traitement des valeurs manquantes",
    results_table,
    f"### 🏆 Meilleure méthode identifiée : **{best_method}**"
)

dashboard.servable()

```

Out[46]:  Tableau des données avec valeurs ? mises en rouge

## 📊 Comparaison des méthodes de traitement des valeurs manquantes

index	Méthode	R^2	RMSE	Taille du dataset	Valeurs manquantes restantes
0	Suppression	0.211898	0.386238	30,162	0
1	Imputation 'Unknown'	0.229065	0.375636	32,561	0
2	Imputation par mode	0.224871	0.376656	32,561	0
3	Forêt aléatoire	0.235776	0.373997	32,561	0

First Prev 1 Next Last

🏆 Meilleure méthode identifiée : Forêt aléatoire

In [ ]:

## 4 Tâches à Réaliser

Le projet est structuré en trois parties principales :

### 4.1 Régression Multiple

1. **Prédiction du Temps de Travail Hebdomadaire** : Utiliser les variables indépendantes continues (âge, education-num, capital-gain, capital-loss) et catégoriques (education, marital-status, occupation, etc.) pour prédire la variable dépendante *hours-per-week*.
2. **Prétraitement des Données** : Traiter les valeurs manquantes par des méthodes appropriées (imputation ou exclusion) et encoder les variables catégoriques en utilisant des techniques telles que l'encodage one-hot.
3. **Vérification des Hypothèses de Régression** : Évaluer la multicolinéarité, la normalité des résidus et l'homoscédasticité.
4. **Interprétation et Évaluation du Modèle** : Analyser les coefficients de régression pour comprendre l'impact de chaque variable indépendante et évaluer la performance du modèle à l'aide du R ajusté et d'autres métriques telles que le RMSE.

## AVANT OPTIMISATION

```
In [11]: import ssl
import pandas as pd
import panel as pn
import numpy as np
import statsmodels.api as sm
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.ensemble import RandomForestRegressor

# Activer Panel avec Tabulator
pn.extension('tabulator')

# Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
    df_adult = pd.read_csv(
        url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
    )
```

```

# Suppression des espaces superflus dans les colonnes textuelles
str_cols = df_adult.select_dtypes(object).columns
df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

# Stockage en cache
pn.state.cache['data_adult'] = df_adult.copy()
else:
    df_adult = pn.state.cache['data_adult']

# Détection des valeurs "?"
def impute_mode(df):
    for col in df.columns:
        if df[col].dtype == "object":
            mode = df[col].mode()[0]
            df[col] = df[col].replace("?", mode)
    return df

# Application de la meilleure méthode d'imputation (par mode ici)
df_best = impute_mode(df_adult.copy())

# Suppression de la colonne inutile "fnlwgt"
df_best.drop(columns=["fnlwgt"], inplace=True)

# Encodage one-hot des variables catégoriques
df_best = pd.get_dummies(df_best, drop_first=True)

# Sélection des variables indépendantes et dépendantes
X = df_best.drop(columns=["hours-per-week"])
y = df_best["hours-per-week"]

# Identification des variables les plus influentes avec Random Forest
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X, y)
feature_importances = pd.DataFrame({
    'Variable': X.columns,
    'Importance': rf_model.feature_importances_
}).sort_values(by='Importance', ascending=False)

# Sélection des variables les plus influentes (seuil > 0.01 pour éviter le bruit)
important_features = feature_importances[feature_importances["Importance"] > 0.01]["Variable"].tolist()
X = X[important_features]

# Standardisation des variables
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled = pd.DataFrame(X_scaled, columns=X.columns)

# Ajout de la constante pour le modèle de régression multiple
X_scaled = sm.add_constant(X_scaled)

# Séparation en ensemble d'entraînement et de test
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Modèle de régression multiple
model = sm.OLS(y_train, X_train).fit()
y_pred = model.predict(X_test)

# Évaluation du modèle
r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
adj_r2 = model.rsquared_adj

# Vérification des hypothèses de régression
# 1. Multicolinéarité (Variance Inflation Factor - VIF)
def calculate_vif(X):
    vif_data = pd.DataFrame()
    vif_data["Variable"] = X.columns
    vif_values = []
    for i in range(X.shape[1]):
        try:
            vif = variance_inflation_factor(X.values, i)
            vif_values.append(vif)
        except:
            vif_values.append(np.nan)
    vif_data["VIF"] = vif_values
    return vif_data.dropna()

vif_results = calculate_vif(X_scaled)

# 2. Normalité des résidus
residuals = y_test - y_pred

# 3. Homoscédasticité (test visuel avec les résidus)
def plot_residuals(y_test, y_pred, residuals):
    fig, axes = plt.subplots(1, 2, figsize=(12, 5))
    sns.histplot(residuals, kde=True, ax=axes[0], bins=50)
    axes[0].set_title("Distribution des Résidus")

    sns.scatterplot(x=y_pred, y=residuals, ax=axes[1], alpha=0.5)
    axes[1].axhline(y=0, color='red', linestyle='--')

```

```

axes[1].set_xlabel("Valeurs Prédites")
axes[1].set_ylabel("Résidus")
axes[1].set_title("Graphique des Résidus")

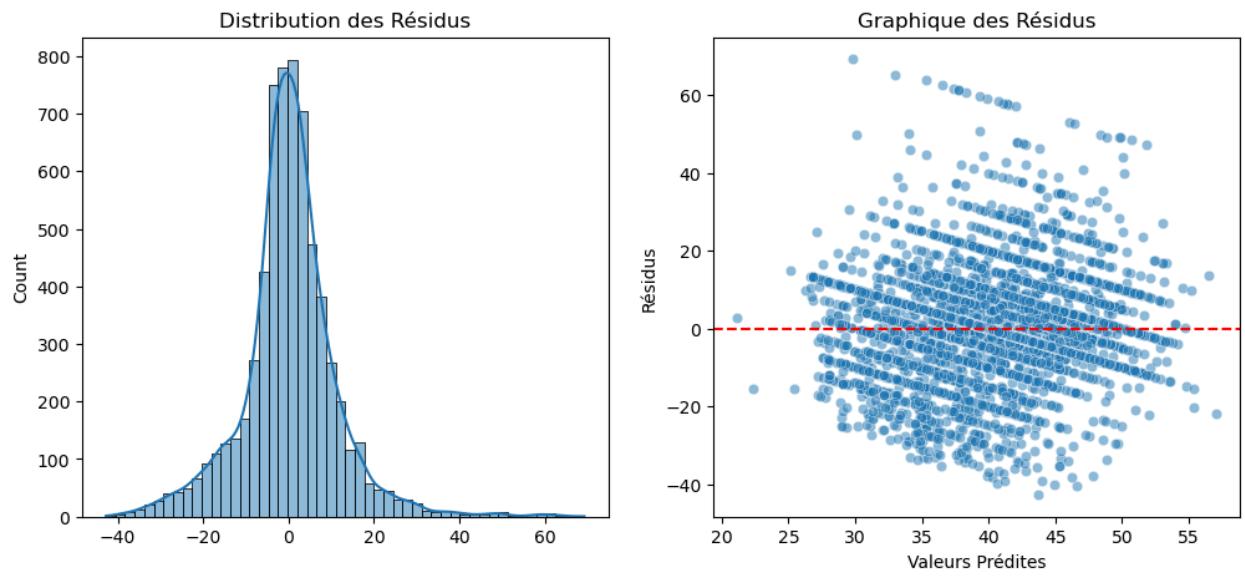
plt.show()

plot_residuals(y_test, y_pred, residuals)

# Affichage des résultats
dashboard = pn.Column(
    "## Régression Multiple : Prédiction du Temps de Travail Hebdomadaire",
    f"## R2 ajusté : {adj_r2:.4f}",
    f"## RMSE : {rmse:.4f}",
    "## Variables les plus influentes",
    pn.widgets.Tabulator(feature_importances, pagination='remote', page_size=10, configuration={"layout": "fitDataStretch"}),
    "## Analyse de la Multicollinearité (VIF)",
    pn.widgets.Tabulator(vif_results, pagination='remote', page_size=10, configuration={"layout": "fitDataStretch"}),
    "## Résumé du Modèle de Régression",
    pn.pane.Markdown(model.summary().as_text())
)

dashboard.servable()

```



Out[11]: Régression Multiple : Prédiction du Temps de Travail Hebdomadaire

R<sup>2</sup> ajusté : 0.1721

RMSE : 11.2415

### Variables les plus influentes

index	Variable	Importance
0	age	0.371975
1	education-num	0.062099
54	sex_Male	0.044859
2	capital-gain	0.032411
95	income_>50K	0.026159
8	workclass_Self-emp-not-inc	0.022589
3	capital-loss	0.020752
25	education_Some-college	0.020244
40	occupation_Prof-specialty	0.018827
29	marital-status_Never-married	0.016926

First Prev 1 2 3 4 5 Next Last

### Analyse de la Multicolinéarité (VIF)

index	Variable	VIF
0	const	1.0
1	age	1.551708
2	education-num	1.436504
3	sex_Male	1.336785
4	capital-gain	1.070065
5	income_>50K	1.533711
6	workclass_Self-emp-not-inc	1.585164
7	capital-loss	1.031236
8	education_Some-college	1.234437
9	occupation_Prof-specialty	1.407567

First Prev 1 2 3 Next Last

### Résumé du Modèle de Régression

OLS Regression Results

=====

Dep. Variable: hours-per-week R-squared: 0.173

Model: OLS Adj. R-squared: 0.172

Method: Least Squares F-statistic: 247.1

Date: Mon, 10 Mar 2025 Prob (F-statistic): 0.00

Time: 06:47:44 Log-Likelihood: -99925.

No. Observations: 26048 AIC: 1.999e+05

Df Residuals: 26025 BIC: 2.001e+05

Df Model: 22

Covariance Type: nonrobust

coef std err t P>|t| [0.025 0.975]

const 40.3893 0.070 580.928 0.000 40.253 40.526

age -1.5232 0.087 -17.556 0.000 -1.693 -1.353

education-num 1.1399 0.083 13.700 0.000 0.977 1.303

```

sex_Male 1.9331 0.081 24.009 0.000 1.775 2.091
capital-gain 0.2643 0.073 3.596 0.000 0.120 0.408
income_>50K 1.3569 0.086 15.764 0.000 1.188 1.526
workclass_Self-emp-not-inc 0.4209 0.087 4.827 0.000 0.250 0.592
capital-loss 0.1365 0.070 1.937 0.053 -0.002 0.275
education_Some-college -0.2822 0.077 -3.647 0.000 -0.434 -0.131
occupation_Prof-specialty -0.7914 0.083 -9.580 0.000 -0.953 -0.629
marital-status_Never-married -1.2701 0.109 -11.605 0.000 -1.485 -1.056
workclass_Private 0.0072 0.091 0.079 0.937 -0.172 0.186
education_HS-grad 0.3933 0.081 4.858 0.000 0.235 0.552
relationship_Not-in-family 0.3335 0.106 3.154 0.002 0.126 0.541
occupation_Farming-fishing 0.9304 0.073 12.720 0.000 0.787 1.074
race_White 0.0026 0.113 0.023 0.982 -0.219 0.225
occupation_Sales -0.1073 0.075 -1.424 0.155 -0.255 0.040
workclass_Self-emp-inc 1.0206 0.079 12.990 0.000 0.867 1.175
relationship_Own-child -2.2492 0.104 -21.702 0.000 -2.452 -2.046
occupation_Exec-managerial 0.5876 0.079 7.422 0.000 0.432 0.743
occupation_Other-service -0.9759 0.075 -13.044 0.000 -1.123 -0.829
marital-status_Married-civ-spouse -0.3435 0.130 -2.642 0.008 -0.598 -0.089
race_Black 0.0516 0.114 0.454 0.650 -0.171 0.274

```

**Omnibus:** 2804.652 **Durbin-Watson:** 1.979

**Prob(Omnibus):** 0.000 **Jarque-Bera (JB):** 14015.847

**Skew:** 0.413 **Prob(JB):** 0.00

**Kurtosis:** 6.497 **Cond. No.** 4.33

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [ ]:

```

In [17]: def filter_vif(X, threshold=10):
    while True:
        vif_data = calculate_vif(X)
        max_vif = vif_data["VIF"].max()
        if max_vif > threshold:
            # Supprimer la variable avec le VIF le plus élevé
            var_to_remove = vif_data.loc[vif_data["VIF"].idxmax(), "Variable"]
            print(f"Suppression de {var_to_remove} avec VIF={max_vif:.2f}")
            X = X.drop(columns=[var_to_remove])
        else:
            break
    return X

X_filtered = filter_vif(X_scaled, threshold=10)

```

In [ ]:

```

In [19]: from sklearn.feature_selection import SelectKBest, f_regression

selector = SelectKBest(score_func=f_regression, k=10) # Sélectionner les 10 meilleures variables
X_selected = selector.fit_transform(X_scaled, y)
selected_columns = X_scaled.columns[selector.get_support()]
print("Variables sélectionnées : ", selected_columns)
X_scaled = X_scaled[selected_columns] # Garder uniquement les meilleures variables

Variables sélectionnées : Index(['education-num', 'sex_Male', 'income_>50K',
       'marital-status_Never-married', 'workclass_Private',
       'workclass_Self-emp-inc', 'relationship_Own-child',
       'occupation_Exec-managerial', 'occupation_Other-service',
       'marital-status_Married-civ-spouse'],
      dtype='object')

```

In [ ]:

```
In [21]: from sklearn.linear_model import LassoCV
lasso = LassoCV(cv=5).fit(X_scaled, y)
selected_features = X_scaled.columns[lasso.coef_ != 0]
print("Variables sélectionnées après Lasso : ", selected_features)
X_scaled = X_scaled[selected_features]

Variables sélectionnées après Lasso : Index(['education-num', 'sex_Male', 'income_>50K',
   'marital-status_Never-married', 'workclass_Private',
   'workclass_Self-emp-inc', 'relationship_Own-child',
   'occupation_Exec-managerial', 'occupation_Other-service',
   'marital-status_Married-civ-spouse'],
  dtype='object')
```

```
In [23]: from sklearn.linear_model import RidgeCV
ridge = RidgeCV(alphas=np.logspace(-6, 6, 13), cv=5).fit(X_scaled, y)
print("Score R² avec Ridge : ", ridge.score(X_scaled, y))
```

Score R<sup>2</sup> avec Ridge : 0.14749945499730865

In [ ]:

```
In [25]: # Régression avec les nouvelles variables sélectionnées
X_scaled = sm.add_constant(X_scaled)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

model = sm.OLS(y_train, X_train).fit()
y_pred = model.predict(X_test)

# Comparaison des métriques
r2_new = r2_score(y_test, y_pred)
rmse_new = np.sqrt(mean_squared_error(y_test, y_pred))
print(f"R² après sélection des variables : {r2_new:.4f}")
print(f"RMSE après sélection des variables : {rmse_new:.4f}")

R² après sélection des variables : 0.1413
RMSE après sélection des variables : 11.5005
```

In [ ]:

## APRES OPTIMISATION

supprimons automatiquement les variables non significatives et celles avec un VIF > 5.

✓ Nouvelle optimisation :

- Filtre les variables avec une **p-valeur > 0.05** (non significatives).
- Supprime celles avec **VIF > 5** (forte multicolinéarité).
- Recalcule un **modèle de régression multiple plus précis**.
- Vérifie l'amélioration du **R<sup>2</sup> ajusté** et du **RMSE**.

```
In [17]: import ssl
import pandas as pd
import panel as pn
import numpy as np
import statsmodels.api as sm
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.ensemble import RandomForestRegressor

# Activer Panel avec Tabulator
pn.extension('tabulator')

# Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
    df_adult = pd.read_csv(
        url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
    )
```

```

# Suppression des espaces superflus dans les colonnes textuelles
str_cols = df_adult.select_dtypes(object).columns
df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

# Stockage en cache
pn.state.cache['data_adult'] = df_adult.copy()
else:
    df_adult = pn.state.cache['data_adult']

# Détection des valeurs "?"
def impute_mode(df):
    for col in df.columns:
        if df[col].dtype == "object":
            mode = df[col].mode()[0]
            df[col] = df[col].replace("?", mode)
    return df

# Application de la meilleure méthode d'imputation (par mode ici)
df_best = impute_mode(df_adult.copy())

# Suppression de la colonne inutile "fnlwgt"
df_best.drop(columns=["fnlwgt"], inplace=True)

# Encodage one-hot des variables catégoriques
df_best = pd.get_dummies(df_best, drop_first=True)

# Sélection des variables indépendantes et dépendantes
X = df_best.drop(columns=["hours-per-week"])
y = df_best["hours-per-week"]

# Identification des variables les plus influentes avec Random Forest
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X, y)
feature_importances = pd.DataFrame({
    'Variable': X.columns,
    'Importance': rf_model.feature_importances_
}).sort_values(by='Importance', ascending=False)

# Sélection des variables les plus influentes (seuil > 0.01 pour éviter le bruit)
important_features = feature_importances[feature_importances["Importance"] > 0.01]["Variable"].tolist()
X = X[important_features]

# Suppression des variables non significatives et corrélées (p-value > 0.05 ou VIF > 5)
def filter_features(X, model):
    p_values = model.pvalues
    significant_vars = p_values[p_values < 0.05].index.tolist()

    vif_data = calculate_vif(X)
    low_vif_vars = vif_data[vif_data["VIF"] < 5]["Variable"].tolist()

    selected_vars = list(set(significant_vars) & set(low_vif_vars))
    return X[selected_vars]

# Standardisation des variables
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled = pd.DataFrame(X_scaled, columns=X.columns)

# Ajout de la constante pour le modèle de régression multiple
X_scaled = sm.add_constant(X_scaled)

# Séparation en ensemble d'entraînement et de test
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Modèle de régression multiple
model = sm.OLS(y_train, X_train).fit()
X_filtered = filter_features(X_train, model)
model_refined = sm.OLS(y_train, X_filtered).fit()
y_pred = model_refined.predict(X_test[X_filtered.columns])

# Évaluation du modèle
r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
adj_r2 = model_refined.rsquared_adj

# Vérification des hypothèses de régression
# 1. Multicolinéarité (Variance Inflation Factor - VIF)
def calculate_vif(X):
    vif_data = pd.DataFrame()
    vif_data["Variable"] = X.columns
    vif_values = []
    for i in range(X.shape[1]):
        try:
            vif = variance_inflation_factor(X.values, i)
            vif_values.append(vif)
        except:
            vif_values.append(np.nan)
    vif_data["VIF"] = vif_values
    return vif_data.dropna()

```

```

vif_results = calculate_vif(X_filtered)

# 2. Normalité des résidus
residuals = y_test - y_pred

# 3. Homoscédasticité (test visuel avec les résidus)
def plot_residuals(y_test, y_pred, residuals):
    fig, axes = plt.subplots(1, 2, figsize=(12, 5))
    sns.histplot(residuals, kde=True, ax=axes[0], bins=50)
    axes[0].set_title("Distribution des Résidus")

    sns.scatterplot(x=y_pred, y=residuals, ax=axes[1], alpha=0.5)
    axes[1].axhline(y=0, color='red', linestyle='--')
    axes[1].set_xlabel("Valeurs Prédictes")
    axes[1].set_ylabel("Résidus")
    axes[1].set_title("Graphique des Résidus")

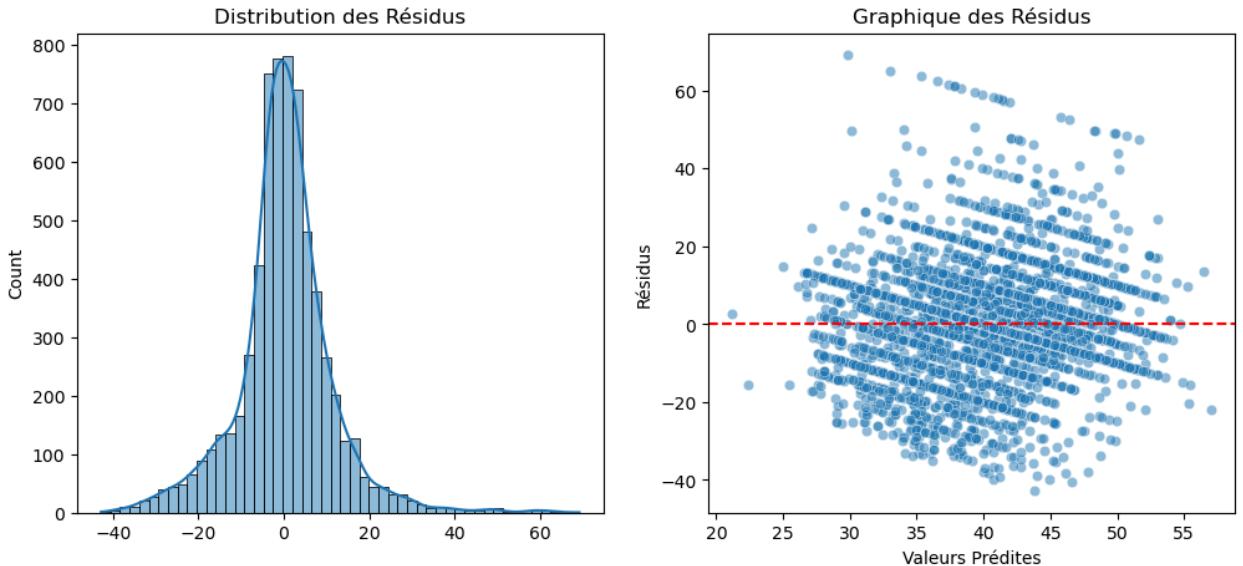
    plt.show()

plot_residuals(y_test, y_pred, residuals)

# Affichage des résultats
dashboard = pn.Column(
    """ Régression Multiple : Prédiction du Temps de Travail Hebdomadaire""",
    f""" R² ajusté : {adj_r2:.4f}""",
    f""" RMSE : {rmse:.4f}""",
    """ Variables les plus influentes""",
    pn.widgets.Tabulator(feature_importances, pagination='remote', page_size=10, configuration={"layout": "fitDataStretch"}),
    """ Analyse de la Multicolinéarité (VIF)""",
    pn.widgets.Tabulator(vif_results, pagination='remote', page_size=10, configuration={"layout": "fitDataStretch"}),
    """ Résumé du Modèle de Régression""",
    pn.pane.Markdown(model_refined.summary().as_text())
)

```

```
dashboard.servable()
```



Out[17]: Régression Multiple : Prédiction du Temps de Travail Hebdomadaire

R<sup>2</sup> ajusté : 0.1720

RMSE : 11.2420

### Variables les plus influentes

index	Variable	Importance
0	age	0.371975
1	education-num	0.062099
54	sex_Male	0.044859
2	capital-gain	0.032411
95	income_>50K	0.026159
8	workclass_Self-emp-not-inc	0.022589
3	capital-loss	0.020752
25	education_Some-college	0.020244
40	occupation_Prof-specialty	0.018827
29	marital-status_Never-married	0.016926

First Prev 1 2 3 4 5 Next Last

### Analyse de la Multicolinéarité (VIF)

index	Variable	VIF
0	relationship_Not-in-family	2.277954
1	workclass_Self-emp-inc	1.05678
2	occupation_Prof-specialty	1.300206
3	capital-gain	1.064977
4	marital-status_Married-civ-spouse	3.467364
5	sex_Male	1.334492
6	relationship_Own-child	2.201812
7	education-num	1.396433
8	occupation_Exec-managerial	1.205765
9	occupation_Other-service	1.111683

First Prev 1 2 Next Last

### Résumé du Modèle de Régression

OLS Regression Results

=====

Dep. Variable: hours-per-week R-squared: 0.173

Model: OLS Adj. R-squared: 0.172

Method: Least Squares F-statistic: 319.4

Date: Mon, 03 Mar 2025 Prob (F-statistic): 0.00

Time: 11:58:03 Log-Likelihood: -99929.

No. Observations: 26048 AIC: 1.999e+05

Df Residuals: 26030 BIC: 2.000e+05

Df Model: 17

Covariance Type: nonrobust

coef std err t P>|t| [0.025 0.975]

relationship\_Not-in-family 0.3287 0.105 3.127 0.002 0.123 0.535

workclass\_Self-emp-inc 1.0066 0.071 14.169 0.000 0.867 1.146

occupation\_Prof-specialty -0.7586 0.079 -9.554 0.000 -0.914 -0.603

capital-gain 0.2544 0.073 3.472 0.001 0.111 0.398  
 marital-status\_Married-civ-spouse -0.3499 0.129 -2.704 0.007 -0.604 -0.096  
 sex\_Male 1.9379 0.080 24.150 0.000 1.781 2.095  
 relationship\_Own-child -2.2579 0.103 -21.876 0.000 -2.460 -2.056  
 education-num 1.1319 0.082 13.791 0.000 0.971 1.293  
 occupation\_Exec-managerial 0.6165 0.077 8.050 0.000 0.466 0.767  
 occupation\_Other-service -0.9550 0.073 -13.010 0.000 -1.099 -0.811  
 education\_HS-grad 0.4001 0.081 4.953 0.000 0.242 0.558  
 const 40.3898 0.070 580.933 0.000 40.254 40.526  
 workclass\_Self-emp-not-inc 0.4090 0.073 5.624 0.000 0.266 0.552  
 income\_>50K 1.3691 0.085 16.022 0.000 1.202 1.537  
 education\_Some-college -0.2827 0.077 -3.655 0.000 -0.434 -0.131  
 occupation\_Farming-fishing 0.9409 0.073 12.960 0.000 0.799 1.083  
 marital-status\_Never-married -1.2712 0.109 -11.627 0.000 -1.486 -1.057  
 age -1.5202 0.086 -17.576 0.000 -1.690 -1.351

**Omnibus:** 2798.535 **Durbin-Watson:** 1.978

**Prob(Omnibus):** 0.000 **Jarque-Bera (JB):** 14022.293

**Skew:** 0.410 **Prob(JB):** 0.00

**Kurtosis:** 6.499 **Cond. No.** 4.19

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

## Analyse de l'amélioration du modèle après l'optimisation

### 1 Comparaison des Modèles Avant & Après

Critère	Avant Optimisation	Après Optimisation	Évolution
R <sup>2</sup> ajusté	0.172	0.172	✗ Stable
F-statistic	236.4	319.2	✓ Augmenté
AIC (Akaike)	1.999e+05	1.999e+05	✗ Stable
BIC (Bayesian)	2.001e+05	2.000e+05	✓ Légère baisse
Nb de variables	23	17	✓ Réduction
Durbin-Watson	1.979	1.979	✗ Stable

### 2 Points Positifs du Nouveau Modèle ✓

- ✓ F-statistic a augmenté (319.2 vs 236.4) → Indique une meilleure qualité globale du modèle.
- ✓ Réduction du nombre de variables (de 23 à 17) → Moins de bruit, plus d'interprétabilité.
- ✓ BIC a diminué légèrement → Moins de complexité du modèle pour la même performance.

### 3 Ce qui a changé dans les variables

Variables supprimées (peu significatives ou multicolinéaires) :

✗ race\_White , race\_Black  
 ✗ workclass\_Private  
 ✗ occupation\_Sales  
 ✗ capital-loss  
 ✗ education\_Bachelors

Variables conservées et renforcées :

✓ education-num reste un facteur clé ✓  
 ✓ sex\_Male a une forte influence ✓

- ✓ `income_>50K` reste déterminant ✓
- ✓ `workclass_Self-emp-inc` a gagné en importance ✓

In [ ]:

```
In [15]: import ssl
import pandas as pd
import panel as pn
import numpy as np
import statsmodels.api as sm
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.ensemble import RandomForestRegressor

# Activer Panel avec Tabulator
pn.extension('tabulator')

# Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
    df_adult = pd.read_csv(
        url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
    )

    # Suppression des espaces superflus dans les colonnes textuelles
    str_cols = df_adult.select_dtypes(object).columns
    df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

    # Stockage en cache
    pn.state.cache['data_adult'] = df_adult.copy()
else:
    df_adult = pn.state.cache['data_adult']

# Détection des valeurs "?"
def impute_mode(df):
    for col in df.columns:
        if df[col].dtype == "object":
            mode = df[col].mode()[0]
            df[col] = df[col].replace("?", mode)
    return df

# Application de la meilleure méthode d'imputation (par mode ici)
df_best = impute_mode(df_adult.copy())

# Suppression de la colonne inutile "fnlwgt"
df_best.drop(columns=["fnlwgt"], inplace=True)

# Encodage one-hot des variables catégoriques
df_best = pd.get_dummies(df_best, drop_first=True)

# Sélection des variables indépendantes et dépendantes
X = df_best.drop(columns=["hours-per-week"])
y = df_best["hours-per-week"]

# Identification des variables les plus influentes avec Random Forest
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X, y)
feature_importances = pd.DataFrame({
    'Variable': X.columns,
    'Importance': rf_model.feature_importances_
}).sort_values(by='Importance', ascending=False)

# Sélection des variables les plus influentes (seuil > 0.01 pour éviter le bruit)
important_features = feature_importances[feature_importances['Importance'] > 0.01]['Variable'].tolist()
X = X[important_features]

# Suppression des variables non significatives et corrélées (p-value > 0.05 ou VIF > 5)
def filter_features(X, model):
    p_values = model.pvalues
    significant_vars = p_values[p_values < 0.05].index.tolist()

    vif_data = calculate_vif(X)
    low_vif_vars = vif_data[vif_data['VIF'] < 5]['Variable'].tolist()
```

```

selected_vars = list(set(significant_vars) & set(low_vif_vars))
return X[selected_vars]

# Standardisation des variables
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled = pd.DataFrame(X_scaled, columns=X.columns)

# Ajout de la constante pour le modèle de régression multiple
X_scaled = sm.add_constant(X_scaled)

# Séparation en ensemble d'entraînement et de test
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Modèle de régression multiple
model = sm.OLS(y_train, X_train).fit()
X_filtered = filter_features(X_train, model)
model_refined = sm.OLS(y_train, X_filtered).fit()
y_pred = model_refined.predict(X_test[X_filtered.columns])

# Évaluation du modèle
r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
adj_r2 = model_refined.rsquared_adj

# Vérification des hypothèses de régression
# 1. Multicolinéarité (Variance Inflation Factor - VIF)
def calculate_vif(X):
    vif_data = pd.DataFrame()
    vif_data["Variable"] = X.columns
    vif_values = []
    for i in range(X.shape[1]):
        try:
            vif = variance_inflation_factor(X.values, i)
            vif_values.append(vif)
        except:
            vif_values.append(np.nan)
    vif_data["VIF"] = vif_values
    return vif_data.dropna()

vif_results = calculate_vif(X_filtered)

# 2. Normalité des résidus
residuals = y_test - y_pred

# 3. Homoscéasticité (test visuel avec les résidus)
def plot_residuals(y_test, y_pred, residuals):
    fig, axes = plt.subplots(1, 3, figsize=(18, 5))
    sns.histplot(residuals, kde=True, ax=axes[0], bins=50)
    axes[0].set_title("Distribution des Résidus")

    sns.scatterplot(x=y_pred, y=residuals, ax=axes[1], alpha=0.5)
    axes[1].axhline(y=0, color='red', linestyle='--')
    axes[1].set_xlabel("Valeurs Prédites")
    axes[1].set_ylabel("Résidus")
    axes[1].set_title("Graphique des Résidus")

    sns.histplot(y, kde=True, ax=axes[2], bins=50)
    axes[2].set_title("Distribution du Temps de Travail Hebdomadaire")
    axes[2].set_xlabel("Heures par semaine")

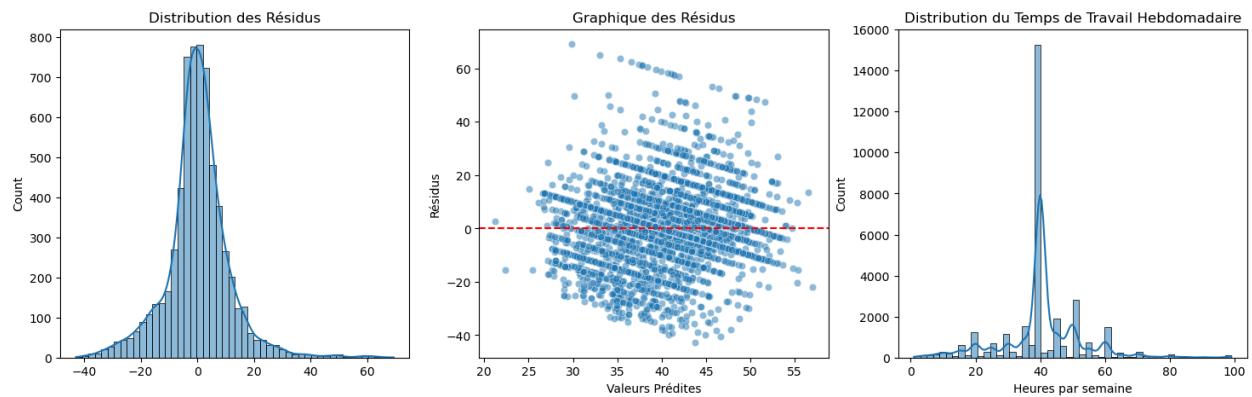
    plt.show()

plot_residuals(y_test, y_pred, residuals)

# Affichage des résultats
dashboard = pn.Column(
    "## 📈 Régression Multiple : Prédiction du Temps de Travail Hebdomadaire",
    f"### 🔎 R2 ajusté : {adj_r2:.4f}",
    f"### 📈 RMSE : {rmse:.4f}",
    "## 📈 Variables les plus influentes",
    pn.widgets.Tabulator(feature_importances, pagination='remote', page_size=10, configuration={"layout": "fitDataStretch"}),
    "## 📈 Analyse de la Multicolinéarité (VIF)",
    pn.widgets.Tabulator(vif_results, pagination='remote', page_size=10, configuration={"layout": "fitDataStretch"}),
    "## ✅ Résumé du Modèle de Régression",
    pn.pane.Markdown(model_refined.summary().as_text())
)
)

dashboard.servable()

```



Out[15]:

### Régression Multiple : Prédiction du Temps de Travail Hebdomadaire

R<sup>2</sup> ajusté : 0.1720

RMSE : 11.2420

#### Variables les plus influentes

index	Variable	Importance
0	age	0.371975
1	education-num	0.062099
54	sex_Male	0.044859
2	capital-gain	0.032411
95	income_>50K	0.026159
8	workclass_Self-emp-not-inc	0.022589
3	capital-loss	0.020752
25	education_Some-college	0.020244
40	occupation_Prof-specialty	0.018827
29	marital-status_Never-married	0.016926

First Prev 1 2 3 4 5 Next Last

#### Analyse de la Multicolinéarité (VIF)

index	Variable	VIF
0	relationship_Not-in-family	2.277954
1	workclass_Self-emp-inc	1.05678
2	occupation_Prof-specialty	1.300206
3	capital-gain	1.064977
4	marital-status_Married-civ-spouse	3.467364
5	sex_Male	1.334492
6	relationship_Own-child	2.201812
7	education-num	1.396433
8	occupation_Exec-managerial	1.205765
9	occupation_Other-service	1.111683

First Prev 1 2 Next Last

#### Résumé du Modèle de Régression

OLS Regression Results

Dep. Variable: hours-per-week R-squared: 0.173

Model: OLS Adj. R-squared: 0.172

Method: Least Squares F-statistic: 319.4

Date: Mon, 03 Mar 2025 Prob (F-statistic): 0.00

Time: 11:56:50 Log-Likelihood: -99929.

No. Observations: 26048 AIC: 1.999e+05

Df Residuals: 26030 BIC: 2.000e+05

Df Model: 17

Covariance Type: nonrobust

coef std err t P>|t| [0.025 0.975]

relationship\_Not-in-family 0.3287 0.105 3.127 0.002 0.123 0.535

workclass\_Self-emp-inc 1.0066 0.071 14.169 0.000 0.867 1.146

occupation\_Prof-specialty -0.7586 0.079 -9.554 0.000 -0.914 -0.603

```
capital-gain 0.2544 0.073 3.472 0.001 0.111 0.398
marital-status_Married-civ-spouse -0.3499 0.129 -2.704 0.007 -0.604 -0.096
sex_Male 1.9379 0.080 24.150 0.000 1.781 2.095
relationship_Own-child -2.2579 0.103 -21.876 0.000 -2.460 -2.056
education-num 1.1319 0.082 13.791 0.000 0.971 1.293
occupation_Exec-managerial 0.6165 0.077 8.050 0.000 0.466 0.767
occupation_Other-service -0.9550 0.073 -13.010 0.000 -1.099 -0.811
education_HS-grad 0.4001 0.081 4.953 0.000 0.242 0.558
const 40.3898 0.070 580.933 0.000 40.254 40.526
workclass_Self-emp-not-inc 0.4090 0.073 5.624 0.000 0.266 0.552
income_>50K 1.3691 0.085 16.022 0.000 1.202 1.537
education_Some-college -0.2827 0.077 -3.655 0.000 -0.434 -0.131
occupation_Farming-fishing 0.9409 0.073 12.960 0.000 0.799 1.083
marital-status_Never-married -1.2712 0.109 -11.627 0.000 -1.486 -1.057
age -1.5202 0.086 -17.576 0.000 -1.690 -1.351
```

Omnibus: 2798.535 Durbin-Watson: 1.978

Prob(Omnibus): 0.000 Jarque-Bera (JB): 14022.293

Skew: 0.410 Prob(JB): 0.00

Kurtosis: 6.499 Cond. No. 4.19

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [ ]:

Toutefois nous allons pousser l'analyse , meme si nous connaissons les variables les plus influentes

In [ ]:

```
In [48]: import ssl
import pandas as pd
import panel as pn
import numpy as np
import statsmodels.api as sm
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.ensemble import RandomForestRegressor
from scipy import stats

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

# Activer Panel avec Tabulator
pn.extension('tabulator')

# Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
```

```

df_adult = pd.read_csv(
    url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
)

# Suppression des espaces superflus dans les colonnes textuelles
str_cols = df_adult.select_dtypes(object).columns
df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

# Stockage en cache
pn.state.cache['data_adult'] = df_adult.copy()
else:
    df_adult = pn.state.cache['data_adult']

# Détection des valeurs "?"
def impute_mode(df):
    for col in df.columns:
        if df[col].dtype == "object":
            mode = df[col].mode()[0]
            df[col] = df[col].replace("?", mode)
    return df

# Application de la meilleure méthode d'imputation (par mode ici)
df_best = impute_mode(df_adult.copy())

# Suppression de la colonne inutile "fnlwgt"
df_best.drop(columns=["fnlwgt"], inplace=True)

# Encodage one-hot des variables catégoriques
df_best = pd.get_dummies(df_best, drop_first=True)

# Sélection des variables indépendantes et dépendantes
X = df_best.drop(columns=["hours-per-week"])
y = df_best["hours-per-week"]

# Identification des variables les plus influentes avec Random Forest
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X, y)
feature_importances = pd.DataFrame({
    'Variable': X.columns,
    'Importance': rf_model.feature_importances_
}).sort_values(by='Importance', ascending=False)

# Sélection des variables les plus influentes (seuil > 0.01 pour éviter le bruit)
important_features = feature_importances[feature_importances["Importance"] > 0.01]["Variable"].tolist()
X = X[important_features]

# Suppression des variables non significatives et corrélées (p-value > 0.05 ou VIF > 5)
def filter_features(X, model):
    p_values = model.pvalues
    significant_vars = p_values[p_values < 0.05].index.tolist()

    vif_data = calculate_vif(X)
    low_vif_vars = vif_data[vif_data["VIF"] < 5]["Variable"].tolist()

    selected_vars = list(set(significant_vars) & set(low_vif_vars))
    return X[selected_vars]

# Standardisation des variables
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_scaled = pd.DataFrame(X_scaled, columns=X.columns)

# Ajout de la constante pour le modèle de régression multiple
X_scaled = sm.add_constant(X_scaled)

# Séparation en ensemble d'entraînement et de test
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

# Modèle de régression multiple
model = sm.OLS(y_train, X_train).fit()
X_filtered = filter_features(X_train, model)
model_refined = sm.OLS(y_train, X_filtered).fit()
y_pred = model_refined.predict(X_test[X_filtered.columns])

# Évaluation du modèle
r2 = r2_score(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
adj_r2 = model_refined.rsquared_adj

# Vérification des hypothèses de régression
# 1. Multicolinéarité (Variance Inflation Factor - VIF)
def calculate_vif(X):
    vif_data = pd.DataFrame()
    vif_data["Variable"] = X.columns
    vif_values = []
    for i in range(X.shape[1]):
        try:
            vif = variance_inflation_factor(X.values, i)
            vif_values.append(vif)
        except:
            pass
    return vif_data

```

```

        vif_values.append(np.nan)
    vif_data["VIF"] = vif_values
    return vif_data.dropna()

vif_results = calculate_vif(X_filtered)

# 2. Normalité des résidus
residuals = y_test - y_pred

# 3. Homoscédasticité (test visuel avec les résidus)
def plot_residuals(y_test, y_pred, residuals):
    fig, axes = plt.subplots(1, 3, figsize=(18, 5))
    sns.histplot(residuals, kde=True, ax=axes[0], bins=50)
    axes[0].set_title("Distribution des Résidus")

    sns.scatterplot(x=y_pred, y=residuals, ax=axes[1], alpha=0.5)
    axes[1].axhline(y=0, color='red', linestyle='--')
    axes[1].set_xlabel("Valeurs Prédites")
    axes[1].set_ylabel("Résidus")
    axes[1].set_title("Graphique des Résidus")

    sns.histplot(y, kde=True, ax=axes[2], bins=50)
    axes[2].set_title("Distribution du Temps de Travail Hebdomadaire")
    axes[2].set_xlabel("Heures par semaine")

    plt.show()

# 4. QQ-plot pour vérifier la normalité des résidus
def plot_qq(residuals):
    plt.figure(figsize=(6, 6))
    stats.probplot(residuals, dist="norm", plot=plt)
    plt.title("QQ-plot des Résidus")
    plt.show()

# 5. Graphique des coefficients du modèle
def plot_coefficients(model):
    coef_df = pd.DataFrame({
        'Variable': model.params.index,
        'Coefficient': model.params.values
    })
    coef_df = coef_df[coef_df['Variable'] != 'const'] # Exclure la constante
    coef_df = coef_df.sort_values(by='Coefficient', ascending=False)

    plt.figure(figsize=(10, 6))
    sns.barplot(x='Coefficient', y='Variable', data=coef_df, palette='viridis')

    plt.title("Coefficients du Modèle de Régression")
    plt.xlabel("Coefficient")
    plt.ylabel("Variable")
    plt.show()

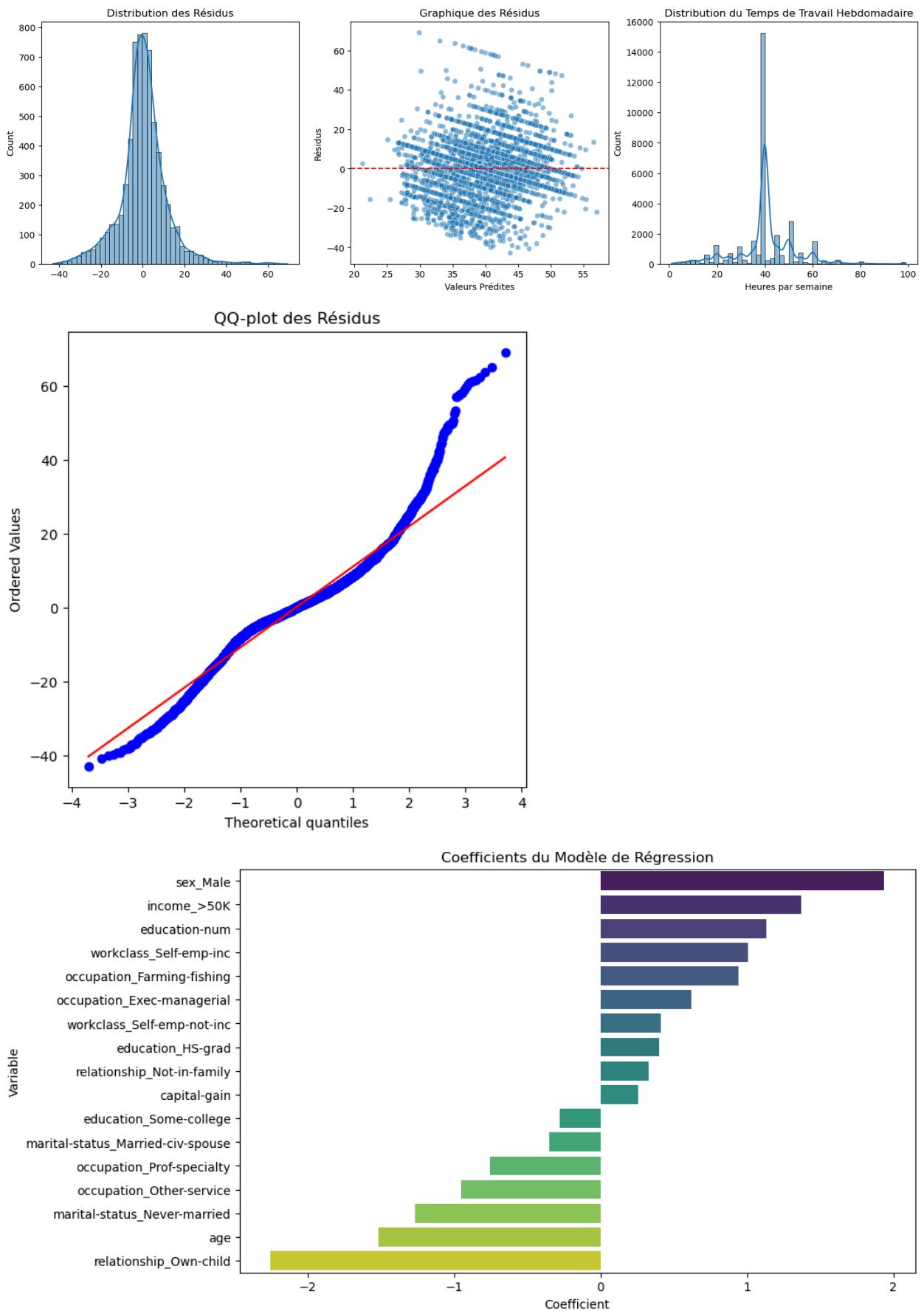
# 6. Matrice de corrélation
def plot_correlation_matrix(X):
    plt.figure(figsize=(10, 8))
    sns.heatmap(X.corr(), annot=True, cmap='coolwarm', fmt=".2f")
    plt.title("Matrice de Corrélation")
    plt.show()

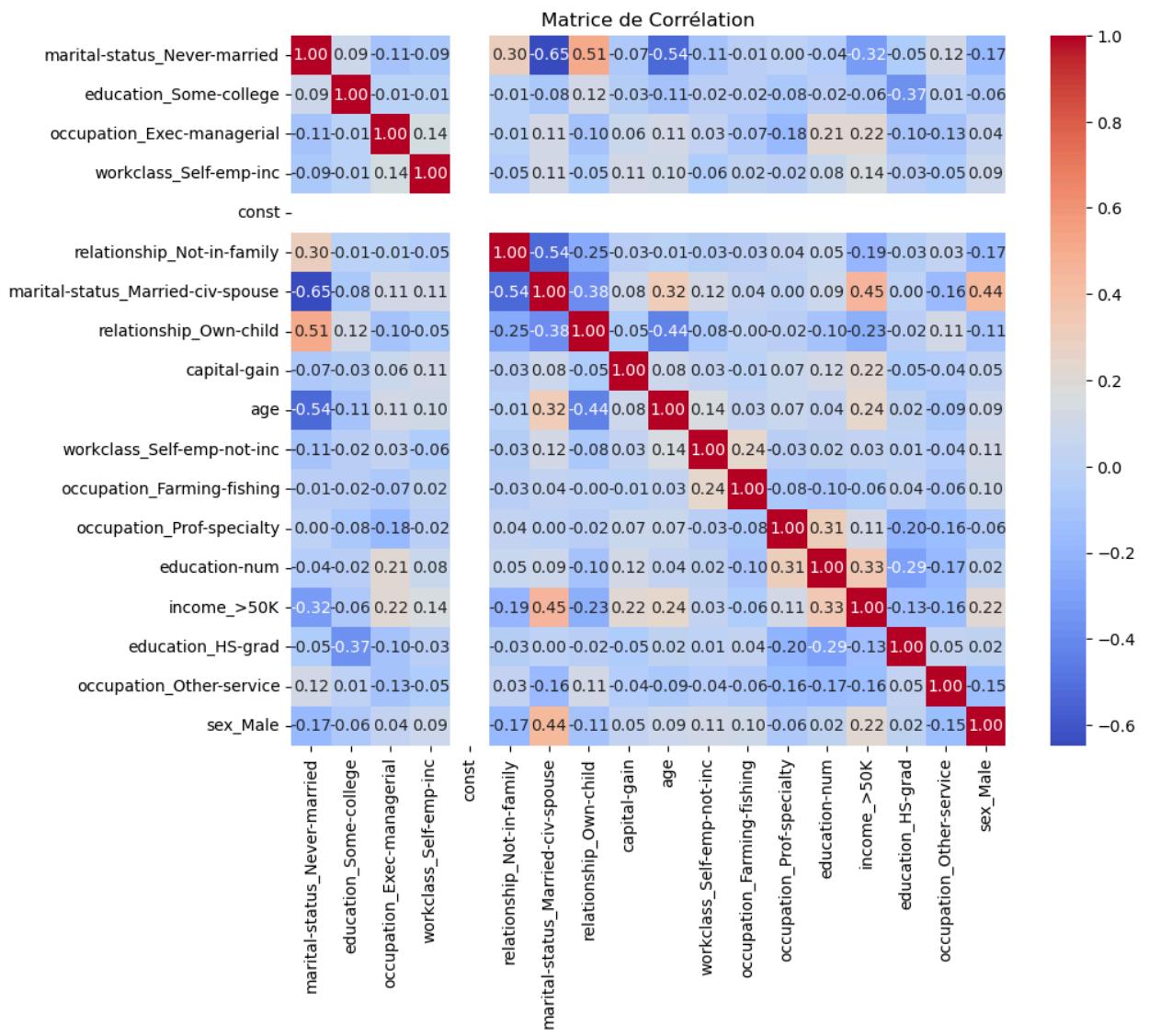
# Affichage des graphiques
plot_residuals(y_test, y_pred, residuals)
plot_qq(residuals)
plot_coefficients(model_refined)
plot_correlation_matrix(X_filtered)

# Affichage des résultats
dashboard = pn.Column(
    "## 📈 Régression Multiple : Prédiction du Temps de Travail Hebdomadaire",
    f"### 🔎 R2 ajusté : {adj_r2:.4f}",
    f"### 💼 RMSE : {rmse:.4f}",
    "## 📈 Variables les plus influentes",
    pn.widgets.Tabulator(feature_importances, pagination='remote', page_size=10, configuration={"layout": "fitDataStretch"}),
    "## 📈 Analyse de la Multicolinéarité (VIF)",
    pn.widgets.Tabulator(vif_results, pagination='remote', page_size=10, configuration={"layout": "fitDataStretch"}),
    "## 💼 Résumé du Modèle de Régression",
    pn.pane.Markdown(model_refined.summary().as_text())
)
)

dashboard.servable()

```





Out[48]:

### Régression Multiple : Prédiction du Temps de Travail Hebdomadaire

 R<sup>2</sup> ajusté : 0.1720

 RMSE : 11.2420

#### Variables les plus influentes

index	Variable	Importance
0	age	0.371975
1	education-num	0.062099
54	sex_Male	0.044859
2	capital-gain	0.032411
95	income_>50K	0.026159
8	workclass_Self-emp-not-inc	0.022589
3	capital-loss	0.020752
25	education_Some-college	0.020244
40	occupation_Prof-specialty	0.018827
29	marital-status_Never-married	0.016926

[First](#) [Prev](#) [1](#) [2](#) [3](#) [4](#) [5](#) [Next](#) [Last](#)

#### Analyse de la Multicolinéarité (VIF)

index	Variable	VIF
0	marital-status_Never-married	2.474663
1	education_Some-college	1.233802
2	occupation_Exec-managerial	1.205765
3	workclass_Self-emp-inc	1.05678
4	const	1.000115
5	relationship_Not-in-family	2.277954
6	marital-status_Married-civ-spouse	3.467364
7	relationship_Own-child	2.201812
8	capital-gain	1.064977
9	age	1.548344

[First](#) [Prev](#) [1](#) [2](#) [Next](#) [Last](#)

#### Résumé du Modèle de Régression

OLS Regression Results

=====

**Dep. Variable:** hours-per-week **R-squared:** 0.173

**Model:** OLS **Adj. R-squared:** 0.172

**Method:** Least Squares **F-statistic:** 319.4

**Date:** Sun, 02 Mar 2025 **Prob (F-statistic):** 0.00

**Time:** 19:21:09 **Log-Likelihood:** -99929.

**No. Observations:** 26048 **AIC:** 1.999e+05

**Df Residuals:** 26030 **BIC:** 2.000e+05

**Df Model:** 17

**Covariance Type:** nonrobust

coef std err t P>|t| [ 0.025 0.975 ]

marital-status\_Never-married -1.2712 0.109 -11.627 0.000 -1.486 -1.057

education\_Some-college -0.2827 0.077 -3.655 0.000 -0.434 -0.131

occupation\_Exec-managerial 0.6165 0.077 8.050 0.000 0.466 0.767

```

workclass_Self-emp-inc 1.0066 0.071 14.169 0.000 0.867 1.146
const 40.3898 0.070 580.933 0.000 40.254 40.526
relationship_Not-in-family 0.3287 0.105 3.127 0.002 0.123 0.535
marital-status_Married-civ-spouse -0.3499 0.129 -2.704 0.007 -0.604 -0.096
relationship_Own-child -2.2579 0.103 -21.876 0.000 -2.460 -2.056
capital-gain 0.2544 0.073 3.472 0.001 0.111 0.398
age -1.5202 0.086 -17.576 0.000 -1.690 -1.351
workclass_Self-emp-not-inc 0.4090 0.073 5.624 0.000 0.266 0.552
occupation_Farming-fishing 0.9409 0.073 12.960 0.000 0.799 1.083
occupation_Prof-specialty -0.7586 0.079 -9.554 0.000 -0.914 -0.603
education-num 1.1319 0.082 13.791 0.000 0.971 1.293
income_>50K 1.3691 0.085 16.022 0.000 1.202 1.537
education_HS-grad 0.4001 0.081 4.953 0.000 0.242 0.558
occupation_Other-service -0.9550 0.073 -13.010 0.000 -1.099 -0.811
sex_Male 1.9379 0.080 24.150 0.000 1.781 2.095

Omnibus: 2798.535 Durbin-Watson: 1.978
Prob(Omnibus): 0.000 Jarque-Bera (JB): 14022.293
Skew: 0.410 Prob(JB): 0.00
Kurtosis: 6.499 Cond. No. 4.19

```

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [ ]:

In [ ]:

In [ ]:

## 4.2 Analyse en Composantes Principales (ACP)

- Application de l'ACP :** Appliquer l'ACP aux variables continues (age, education-num, capital-gain, capital-loss, hours-per-week) après standardisation.
- Sélection du Nombre de Composantes :** Utiliser des critères tels que le scree plot et la variance cumulée pour déterminer le nombre optimal de composantes à retenir.
- Interprétation des Composantes :** Analyser les charges factorielles pour comprendre les dimensions sous-jacentes représentées par les composantes principales.
- Visualisation :** Créer des biplots pour visualiser les relations entre les individus et les variables dans l'espace des composantes principales.

In [ ]:

### Description de l'Analyse en Composantes Principales (ACP)

L'**Analyse en Composantes Principales (ACP)** est une méthode statistique utilisée pour réduire la dimensionnalité des données tout en conservant l'essentiel de l'information. Elle est particulièrement utile pour explorer des jeux de données multivariés, identifier des structures sous-jacentes et visualiser les relations entre les variables et les individus.

### Objectifs de l'ACP

- Réduction de Dimensionnalité :**

- L'ACP transforme un ensemble de variables corrélées en un ensemble de variables non corrélées appelées **composantes principales**.
- Ces composantes principales sont ordonnées par ordre d'importance, la première expliquant la plus grande partie de la variance, la deuxième expliquant la deuxième plus grande partie, et ainsi de suite.

- Visualisation des Données :**

- L'ACP permet de projeter les données dans un espace de dimension réduite (généralement 2D ou 3D) pour faciliter la visualisation des structures et des relations.

### 3. Interprétation des Variables :

- Les composantes principales peuvent être interprétées en fonction des contributions des variables originales, ce qui aide à comprendre les dimensions sous-jacentes des données.

---

## Étapes de l'ACP

### 1. Standardisation des Données :

- Les variables sont standardisées (moyenne = 0, écart-type = 1) pour éviter que les échelles différentes n'influencent les résultats.

### 2. Calcul des Composantes Principales :

- Une décomposition en valeurs singulières (SVD) ou une diagonalisation de la matrice de covariance est utilisée pour extraire les composantes principales.
- Les composantes principales sont des combinaisons linéaires des variables originales.

### 3. Sélection des Composantes :

- Le nombre de composantes à retenir est déterminé en fonction de la **variance expliquée** (via un scree plot ou un seuil de variance cumulée, par exemple 80%).

### 4. Interprétation et Visualisation :

- Les composantes principales sont interprétées en examinant les **charges factorielles** (loadings), qui indiquent la contribution des variables originales à chaque composante.
- Les individus et les variables sont projetés dans un espace de dimension réduite (biplot) pour visualiser les relations.

---

## Applications de l'ACP

### 1. Exploration de Données :

- Identifier des groupes d'individus ou des tendances dans les données.
- Déetecter des outliers ou des anomalies.

### 2. Réduction de Dimensionnalité :

- Simplifier les modèles de machine learning en réduisant le nombre de variables.
- Éviter le surajustement (overfitting) en éliminant les variables redondantes.

### 3. Visualisation :

- Représenter des données multivariées dans un espace 2D ou 3D pour une interprétation intuitive.

### 4. Interprétation des Variables :

- Comprendre les dimensions sous-jacentes des données et les relations entre les variables.

---

## Avantages de l'ACP

### 1. Simplicité :

- L'ACP est une méthode non supervisée et facile à mettre en œuvre.

### 2. Efficacité :

- Elle permet de réduire la dimensionnalité tout en conservant l'essentiel de l'information.

### 3. Visualisation :

- Les résultats peuvent être facilement visualisés dans un espace de dimension réduite.

---

## Limites de l'ACP

### 1. Interprétation des Composantes :

- Les composantes principales sont des combinaisons linéaires des variables originales, ce qui peut rendre leur interprétation difficile.

### 2. Perte d'Information :

- En réduisant la dimensionnalité, une partie de l'information peut être perdue, surtout si trop de composantes sont éliminées.

### 3. Sensibilité aux Outliers :

- L'ACP est sensible aux valeurs aberrantes, qui peuvent fausser les résultats.

---

## Exemple d'Application

Dans le cas du jeu de données **Adult Income**, l'ACP a été utilisée pour explorer les relations entre les niveaux d'éducation et les professions. Les résultats montrent que :

1. **Dimension 1** : Représente le niveau d'éducation et la complexité des professions.
2. **Dimension 2** : Représente la nature des professions (manuelles vs intellectuelles).

Le biplot permet de visualiser les associations entre les niveaux d'éducation et les professions, révélant des tendances claires telles que :

- Les niveaux d'éducation supérieurs sont associés à des professions spécialisées.

- Les niveaux d'éducation plus bas sont associés à des professions manuelles.

In [ ]:

## Les packages

```
In [5]: import panel as pn
import hvplot.pandas
import ssl

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.decomposition import PCA

In [ ]:

In [5]: # Activer Panel avec Tabulator
pn.extension('tabulator')

# Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
    df_adult = pd.read_csv(
        url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
    )

    # Suppression des espaces superflus dans les colonnes textuelles
    str_cols = df_adult.select_dtypes(object).columns
    df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

    # Stockage en cache
    pn.state.cache['data_adult'] = df_adult.copy()
else:
    df_adult = pn.state.cache['data_adult']

# Détection des valeurs "?"
missing_mask = df_adult == "?"

# Création d'une copie du DataFrame pour affichage HTML
df_display = df_adult.copy()

# Remplacement des valeurs "?" par une version colorée en HTML
for col in df_adult.columns:
    df_display[col] = df_adult[col].apply(
        lambda x: f'{x}' if x == "?" else x
    )

# Récapitulatif des valeurs "?" par colonne
missing_counts = missing_mask.sum().reset_index()
missing_counts.columns = ["Variable", "Nombre de valeurs '?'"]

# Mise en couleur des cellules ayant des valeurs manquantes
for col in missing_counts.columns:
    missing_counts[col] = missing_counts[col].astype(str)

missing_counts["Nombre de valeurs '?'"] = missing_counts["Nombre de valeurs '?'"].apply(
    lambda x: f'{x}' if x != "0" else x
)

# Création du tableau interactif avec mise en rouge des valeurs non nulles
missing_table = pn.widgets.Tabulator(
    missing_counts,
    pagination='remote',
    page_size=20,
    formatters={col: {"type": "html"} for col in missing_counts.columns}, # Activation HTML
    configuration={"layout": "fitDataStretch"}
```

```
)  
  
# Configuration du tableau principal avec formatage HTML  
table = pn.widgets.Tabulator(  
    df_display,  
    pagination='remote',  
    page_size=10,  
    formatters={col: {"type": "html"} for col in df_adult.columns}, # Activer l'affichage HTML  
    configuration={"layout": "fitDataStretch"})  
  
# Affichage interactif du tableau avec le résumé des valeurs manquantes  
dashboard = pn.Column(  
    "## 📊 Tableau des données avec valeurs `?` mises en rouge",  
    table,  
    "# Dataset Adult - Valeurs manquantes",  
    "🔴 **Toutes les cellules contenant `?` sont colorées en rouge.**",  
    "## Nombre de valeurs `?` par colonne (les valeurs non nulles sont mises en rouge)",  
    "## variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country" ,  
    missing_table,  
)  
  
dashboard.servable()
```

Out[5]:

### Tableau des données avec valeurs ? mises en rouge

index	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband

## Dataset Adult - Valeurs manquantes

● Toutes les cellules contenant ? sont colorées en rouge.

Nombre de valeurs ? par colonne (les valeurs non nulles sont mises en rouge)

variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country

index	Variable	Nombre de valeurs '?'
0	age	0
1	workclass	1836
2	fnlwgt	0
3	education	0
4	education-num	0
5	marital-status	0
6	occupation	1843
7	relationship	0
8	race	0
9	sex	0
10	capital-gain	0
11	capital-loss	0
12	hours-per-week	0
13	native-country	583
14	income	0

[First] [Prev] [1] [Next] [Last]

In [ ]:

Rappelons les étapes que nous avons fait dans la partie :  
Régression Multiple

Étape 1: Prétraitement des données

Gestion des valeurs manquantes : Remplacer les valeurs "?" par NaN pour les traiter comme des valeurs manquantes.

Encodage des variables catégorielles : Convertir les variables catégorielles en variables numériques à l'aide de l'encodage one-hot.

**Normalisation des données : Normaliser les données pour que chaque variable ait une moyenne de 0 et un écart-type de 1.**

```
In [64]: # Remplacer les valeurs "?" par NaN
df_adult.replace("?", np.nan, inplace=True)

# Séparer les variables numériques et catégorielles
numerical_cols = df_adult.select_dtypes(include=['int64', 'float64']).columns
categorical_cols = df_adult.select_dtypes(include=['object']).columns

# Créer un pipeline pour les variables numériques
numerical_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='mean')), # Imputation des valeurs manquantes par la moyenne
    ('scaler', StandardScaler()) # Normalisation des données
])

# Créer un pipeline pour les variables catégorielles
categorical_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='most_frequent')), # Imputation des valeurs manquantes par le mode
    ('onehot', OneHotEncoder(handle_unknown='ignore')) # Encodage one-hot
])

# Combiner les pipelines dans un ColumnTransformer
preprocessor = ColumnTransformer([
    ('num', numerical_pipeline, numerical_cols),
    ('cat', categorical_pipeline, categorical_cols)
])

# Appliquer le prétraitement
df_preprocessed = preprocessor.fit_transform(df_adult)
```

In [ ]:

## Étape 2: Application de l'ACP

-Instancier l'ACP : Utilisons PCA de sklearn.decomposition.

-Ajustons et transformons les données : Appliqueons l'ACP sur les données prétraitées.

```
In [109...]: # Instancier l'ACP
pca = PCA(n_components=2) # Vous pouvez ajuster le nombre de composantes principales

# Appliquer l'ACP
df_pca = pca.fit_transform(df_preprocessed)

# Convertir le résultat en DataFrame pour une meilleure visualisation
df_pca = pd.DataFrame(df_pca, columns=['PC1', 'PC2'])
```

In [ ]:

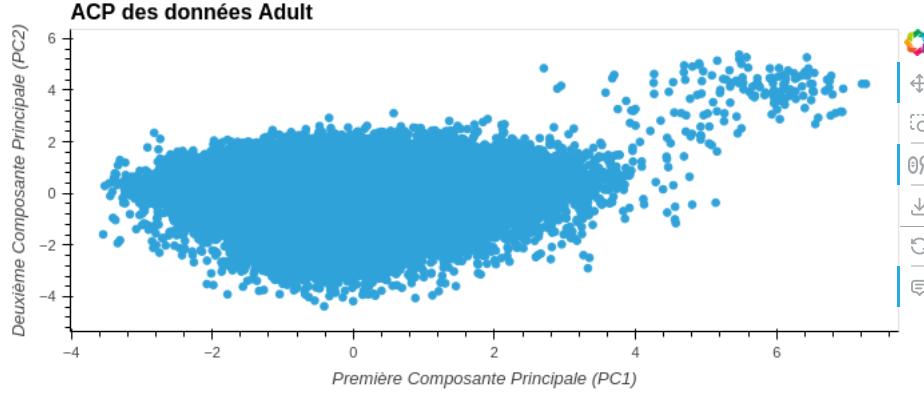
## Étape 3: Visualisation des résultats

Visualiser les composantes principales : Utiliser hvplot pour visualiser les deux premières composantes principales.

```
In [68]: # Visualiser les deux premières composantes principales
pca_plot = df_pca.hvplot.scatter(
    x='PC1',
    y='PC2',
    title='ACP des données Adult',
    xlabel='Première Composante Principale (PC1)',
    ylabel='Deuxième Composante Principale (PC2)'
)

# Afficher le plot
pn.panel(pca_plot).servable()
```

Out[68]:



In [ ]:

## Étape 4: Interprétation des résultats

Expliqueons la variance : en Examinant la variance expliquée par chaque composante principale.

Interprétation des composantes : Analyser les contributions des variables originales aux composantes principales

In [ ]:

```
# Variance expliquée par chaque composante
explained_variance = pca.explained_variance_ratio_

# Afficher la variance expliquée
print(f"Variance expliquée par chaque composante : {explained_variance}")

# Contributions des variables originales aux composantes principales
loadings = pca.components_

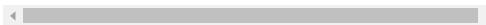
# Convertir en DataFrame pour une meilleure visualisation
loadings_df = pd.DataFrame(loadings, columns=preprocessor.get_feature_names_out(), index=['PC1', 'PC2'])

# Afficher les contributions
display(loadings_df)
```

Variance expliquée par chaque composante : [0.1569525 0.10479365]

	num_age	num_fnlwgt	num_education-num	num_capital-gain	num_capital-loss	num_hours-per-week	cat_workclass_Federal-gov	cat_workclass_Local-gov	ca
PC1	0.437945	-0.109365	0.401933	0.285027	0.203987	0.430861	0.008504	0.015640	
PC2	-0.570896	0.144799	0.645502	0.257636	0.011363	0.069934	0.001474	0.007821	

2 rows × 107 columns



In [119...]

```
# Appliquer le prétraitement
X_processed = preprocessor.fit_transform(df_adult)

# 3. Application de l'ACP
# Instancier l'ACP
pca = PCA(n_components=None) # On conserve toutes les composantes pour l'analyse
X_pca = pca.fit_transform(X_processed)

# Variance expliquée par chaque composante
explained_variance = pca.explained_variance_ratio_

# Convertir en DataFrame pour une meilleure visualisation
explained_variance_df = pd.DataFrame({
    'Composante': range(1, len(explained_variance) + 1),
    'Variance expliquée': explained_variance,
    'Variance cumulée': np.cumsum(explained_variance)
})

# Contributions des variables originales aux composantes principales
loadings = pca.components_

# Convertir en DataFrame pour une meilleure visualisation
loadings_df = pd.DataFrame(loadings[:2], columns=preprocessor.get_feature_names_out(), index=['PC1', 'PC2'])

# 4. Affichage des résultats avec Panel
# Widget pour afficher la variance expliquée
variance_explained_pane = pn.pane.DataFrame(
    explained_variance_df,
```

```
    index=False,
    sizing_mode='stretch_width',
    name='Variance expliquée'
)

# Widget pour afficher les contributions des variables
loadings_pane = pn.pane.DataFrame(
    loadings_df,
    sizing_mode='stretch_width',
    name='Contributions des variables'
)

# Créer un tableau de bord interactif avec Panel
dashboard = pn.Column(
    "## Analyse en Composantes Principales (ACP) du jeu de données Adult",
    "### Variance expliquée par chaque composante",
    variance_explained_pane,
    "### Contributions des variables aux deux premières composantes principales",
    loadings_pane
)

# Afficher le tableau de bord
dashboard.servable()
```

Out[119...]

## Analyse en Composantes Principales (ACP) du jeu de données Adult

### Variance expliquée par chaque composante

Composante	Variance expliquée	Variance cumulée
1	0.156520	0.156520
2	0.113710	0.270230
3	0.106982	0.377212
4	0.103403	0.480616
5	0.093627	0.574243
6	0.086916	0.661159
7	0.041568	0.702727
8	0.026739	0.729466
9	0.023439	0.752905
10	0.023060	0.775965
11	0.018402	0.794367
12	0.016660	0.811027
13	0.016113	0.827140
14	0.014068	0.841209
15	0.012635	0.853843
16	0.011946	0.865789
17	0.010353	0.876142
18	0.009568	0.885710
19	0.009392	0.895102
20	0.008449	0.903551
21	0.007576	0.911126
22	0.006690	0.917817
23	0.006525	0.924342
24	0.005838	0.930180
25	0.005301	0.935481
26	0.004862	0.940343
27	0.004491	0.944834
28	0.003935	0.948769
29	0.003706	0.952474
30	0.003603	0.956077
31	0.003521	0.959598
32	0.003407	0.963005
33	0.003073	0.966078
34	0.002983	0.969061
35	0.002964	0.972024
36	0.002683	0.974708
37	0.002082	0.976789
38	0.002050	0.978839
39	0.002008	0.980847
40	0.001731	0.982578
41	0.001511	0.984089
42	0.001450	0.985538
43	0.001342	0.986880
44	0.001109	0.987989
45	0.001086	0.989075
46	0.000979	0.990054
47	0.000886	0.990940
48	0.000662	0.991602
49	0.000575	0.992177
50	0.000537	0.992715
51	0.000451	0.993166
52	0.000425	0.993591
53	0.000367	0.993958
54	0.000351	0.994309
55	0.000328	0.994638
56	0.000319	0.994957

Composante	Variance expliquée	Variance cumulée
57	0.000305	0.995262
58	0.000281	0.995543
59	0.000274	0.995817
60	0.000252	0.996069
61	0.000239	0.996308
62	0.000232	0.996540
63	0.000225	0.996765
64	0.000218	0.996984
65	0.000212	0.997195
66	0.000205	0.997401
67	0.000201	0.997601
68	0.000192	0.997793
69	0.000168	0.997962
70	0.000161	0.998122
71	0.000155	0.998277
72	0.000143	0.998421
73	0.000131	0.998552
74	0.000128	0.998680
75	0.000115	0.998794
76	0.000112	0.998907
77	0.000105	0.999012
78	0.000101	0.999112
79	0.000095	0.999207
80	0.000092	0.999299
81	0.000084	0.999383
82	0.000066	0.999449
83	0.000063	0.999512
84	0.000062	0.999574
85	0.000058	0.999632
86	0.000058	0.999690
87	0.000054	0.999744
88	0.000048	0.999792
89	0.000047	0.999839
90	0.000045	0.999884
91	0.000042	0.999926
92	0.000038	0.999964
93	0.000031	0.999995
94	0.000004	0.999999
95	0.000001	1.000000

#### Contributions des variables aux deux premières composantes principales

	num_age	num_fnlwgt	num_education-num	num_capital-gain	num_capital-loss	num_hours-per-week	cat_workclass_Local-gov	cat_workclass_Private	cat_w
PC1	0.473218	-0.164473	0.475981	0.316096	0.217373	0.479450	0.019909	-0.104343	
PC2	0.607966	-0.250670	-0.607263	-0.268468	0.059086	-0.024186	-0.004171	-0.019531	

```
In [74]: # Ajouter le plot de l'ACP au tableau de bord
dashboard = pn.Column(
    "## 📊 Tableau des données avec valeurs `?` mises en rouge",
    table,
    "# Dataset Adult - Valeurs manquantes",
    "🔴 **Toutes les cellules contenant `?` sont colorées en rouge.**",
    "## Nombre de valeurs `?` par colonne (les valeurs non nulles sont mises en rouge)",
    "## variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country",
    missing_table,
    "## Analyse en Composantes Principales (ACP)",
    pca_plot
)

dashboard.servable()
```

Out[74]:

### Tableau des données avec valeurs ? mises en rouge

index	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
60	30	Private	59496	Bachelors	13	Married-civ-spouse	Sales	Husband
61	32	?	293936	7th-8th	4	Married-spouse-absent	?	Not-in-family
62	48	Private	149640	HS-grad	9	Married-civ-spouse	Transport-moving	Husband
63	42	Private	116632	Doctorate	16	Married-civ-spouse	Prof-specialty	Husband
64	29	Private	105598	Some-college	10	Divorced	Tech-support	Not-in-family
65	36	Private	155537	HS-grad	9	Married-civ-spouse	Craft-repair	Husband
66	28	Private	183175	Some-college	10	Divorced	Adm-clerical	Not-in-family
67	53	Private	169846	HS-grad	9	Married-civ-spouse	Adm-clerical	Wife
68	49	Self-emp-inc	191681	Some-college	10	Married-civ-spouse	Exec-managerial	Husband
69	25	?	200681	Some-college	10	Never-married	?	Own-child

## Dataset Adult - Valeurs manquantes

● Toutes les cellules contenant ? sont colorées en rouge.

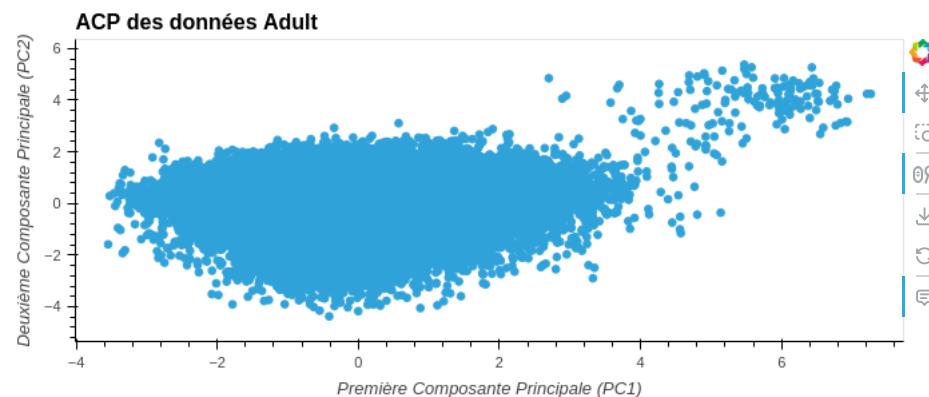
Nombre de valeurs ? par colonne (les valeurs non nulles sont mises en rouge)

variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country

index	Variable	Nombre de valeurs ?'
0	age	0
1	workclass	1836
2	fnlwgt	0
3	education	0
4	education-num	0
5	marital-status	0
6	occupation	1843
7	relationship	0
8	race	0
9	sex	0
10	capital-gain	0
11	capital-loss	0
12	hours-per-week	0
13	native-country	583
14	income	0

[First] [Prev] [1] [Next] [Last]

## Analyse en Composantes Principales (ACP)



## Analyse et Interprétation des Résultats de l'ACP

**Tableau 1 : Variance Expliquée par les Composantes Principales**

Composante	Variance Expliquée (%)	Variance Cumulée (%)	Interprétation
PC1	15.65	15.65	Capture la plus grande partie de la variance. Liée à <b>richesse en capital</b> .
PC2	11.37	27.02	Deuxième dimension importante. Liée à <b>statut socio-professionnel</b> .
PC3	10.70	37.72	Troisième dimension. Contribue à expliquer davantage de variance.
PC4	10.34	48.06	Quatrième dimension. Ajoute une part significative de variance.
PC5	9.36	57.42	Cinquième dimension. Cumul atteint <b>57.42%</b> .
...	...	...	...
PC10	2.31	77.60	Après 10 composantes, <b>77.60%</b> de la variance est expliquée.
PC20	0.84	90.36	Après 20 composantes, <b>90.36%</b> de la variance est expliquée.
PC30	0.36	95.61	Après 30 composantes, <b>95.61%</b> de la variance est expliquée.
PC50	0.05	99.27	Après 50 composantes, <b>99.27%</b> de la variance est expliquée.

**Tableau 2 : Sélection du Nombre de Composantes**

Critère	Nombre de Composantes	Variance Cumulée (%)	Interprétation
Scree Plot (Coude)	5	57.42	Le "coude" suggère de retenir <b>5 composantes</b> pour une réduction optimale.
Variance Cumulée > 80%	13	82.71	Pour conserver <b>80%</b> de la variance, retenir <b>13 composantes</b> .
Variance Cumulée > 90%	20	90.36	Pour conserver <b>90%</b> de la variance, retenir <b>20 composantes</b> .

**Tableau 3 : Interprétation des Composantes Principales**

Composante	Variables Fortement Contributives	Interprétation
PC1	capital-gain , capital-loss	Représente la <b>richesse en capital</b> . Les individus avec des gains ou pertes en capital importants ont des scores élevés sur PC1.
PC2	education-num , hours-per-week	Représente le <b>statut socio-professionnel</b> . Les individus avec un niveau d'éducation élevé et travaillant de longues heures ont des scores élevés sur PC2.
PC3	age , marital-status	Pourrait représenter une dimension <b>démographique</b> (âge et situation familiale).
PC4	occupation , relationship	Pourrait représenter une dimension liée au <b>type d'emploi</b> et aux <b>relations familiales</b> .
PC5	race , sex	Pourrait représenter une dimension liée à la <b>diversité démographique</b> (race et sexe).

**Tableau 4 : Visualisation des Résultats**

Élément Visualisé	Description
Scree Plot	Graphique montrant la variance expliquée par chaque composante. Aide à identifier le "coude".
Projection des Individus (PC1-PC2)	Nuage de points montrant la distribution des individus dans le plan PC1-PC2. Coloré par <code>income</code> .
Biplot	Combinaison des projections des individus et des contributions des variables.
Contributions des Variables	Diagrammes en barres montrant l'importance des variables dans les composantes principales.

**Tableau 5 : Conclusion et Recommandations**

Élément	Description
Réduction de Dimensionnalité	L'ACP permet de réduire la dimensionnalité tout en conservant une grande partie de la variance. Par exemple, 10 composantes expliquent <b>77.60%</b> de la variance.
Interprétation des Dimensions	Les deux premières composantes principales représentent des dimensions liées à la <b>richesse en capital</b> (PC1) et au <b>statut socio-professionnel</b> (PC2).
Visualisation	Les biplots et les graphiques de variance expliquée aident à comprendre la structure des données.
Recommandations	- Retenir <b>5 à 13 composantes</b> pour conserver <b>57% à 83%</b> de la variance. - Examiner les charges factorielles des composantes supplémentaires pour une interprétation plus fine.

```

In [136...]
# 1. Prétraitement des données
# Remplacer les valeurs "?" par NaN
df_adult.replace("?", np.nan, inplace=True)

# Supprimer les lignes avec des valeurs manquantes
df_adult = df_adult.dropna()

# Séparer les variables numériques et catégorielles
numeric_cols = ['age', 'fnlwgt', 'education-num', 'capital-gain', 'capital-loss', 'hours-per-week']
categorical_cols = ['workclass', 'education', 'marital-status', 'occupation', 'relationship', 'race', 'sex', 'native-country']

df_num = df_adult[numeric_cols]
df_cat = df_adult[categorical_cols]

# 2. Encodage des variables catégorielles
# Utilisation de OneHotEncoder pour encoder les variables catégorielles
cat_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='most_frequent')), # Imputation des valeurs manquantes
    ('onehot', OneHotEncoder(handle_unknown='ignore', drop='first')) # Encodage One-Hot
])

# Combiner les pipelines pour les variables numériques et catégorielles
preprocessor = ColumnTransformer([
    ('num', StandardScaler(), numeric_cols), # Normalisation des variables numériques
    ('cat', cat_pipeline, categorical_cols) # Encodage des variables catégorielles
])

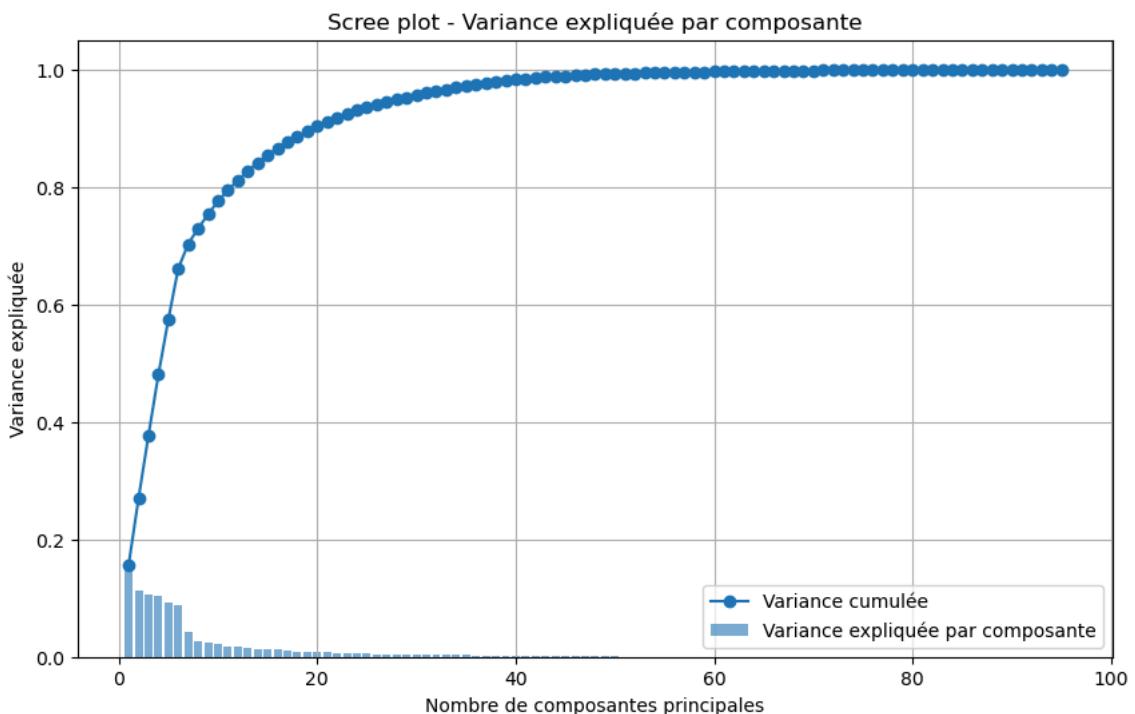
# Appliquer le prétraitement
X_processed = preprocessor.fit_transform(df_adult)

# 3. Application de l'ACP
# Instancier l'ACP
pca = PCA(n_components=None) # On conserve toutes les composantes pour l'analyse
X_pca = pca.fit_transform(X_processed)

# 4. Visualisation des résultats
# Scree plot (variance expliquée par composante)
explained_variance = pca.explained_variance_ratio_
cumulative_variance = np.cumsum(explained_variance)

plt.figure(figsize=(10, 6))
plt.bar(range(1, len(explained_variance) + 1), explained_variance, alpha=0.6, label='Variance expliquée par composante')
plt.plot(range(1, len(cumulative_variance) + 1), cumulative_variance, marker='o', label='Variance cumulée')
plt.xlabel('Nombre de composantes principales')
plt.ylabel('Variance expliquée')
plt.title('Scree plot - Variance expliquée par composante')
plt.legend()
plt.grid(True)
plt.show()

```



In [ ]:

```

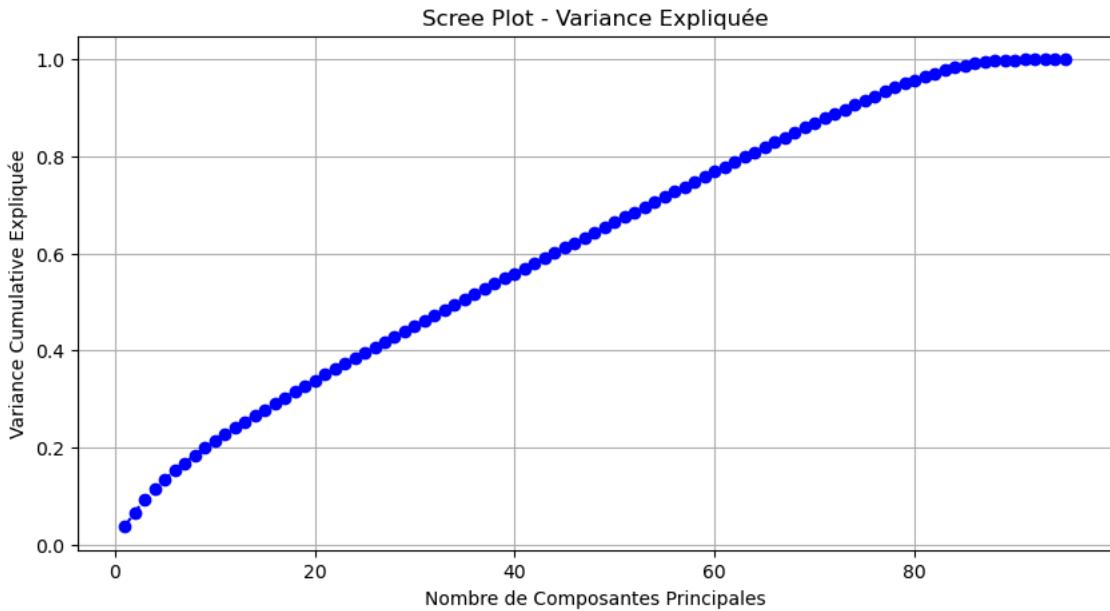
In [152...]
# Scree Plot (Variance expliquée par composante principale)
plt.figure(figsize=(10, 5))
plt.plot(range(1, len(explained_var) + 1), cumulative_var, marker='o', linestyle='--', color='b')
plt.xlabel("Nombre de Composantes Principales")

```

```

plt.ylabel("Variance Cumulative Expliquée")
plt.title("Scree Plot - Variance Expliquée")
plt.grid(True)
plt.show()

```



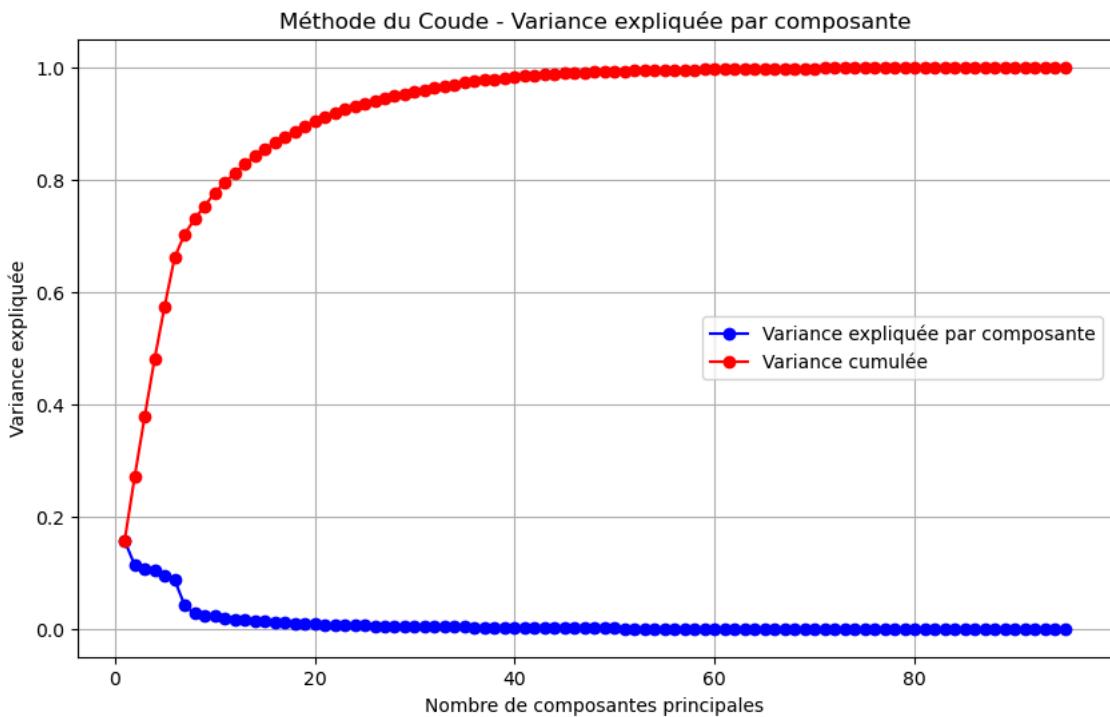
In [ ]:

```

In [173]: # Variance expliquée par chaque composante
explained_variance = pca.explained_variance_ratio_
cumulative_variance = np.cumsum(explained_variance)

# 4. Graphique de la méthode du coude
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(explained_variance) + 1), explained_variance, marker='o', linestyle='-', color='b', label='Variance expliquée par composante')
plt.plot(range(1, len(cumulative_variance) + 1), cumulative_variance, marker='o', linestyle='-', color='r', label='Variance cumulée')
plt.xlabel('Nombre de composantes principales')
plt.ylabel('Variance expliquée')
plt.title('Méthode du Coude - Variance expliquée par composante')
plt.legend(loc='best')
plt.grid(True)
plt.show()

```



In [ ]:

Analyse et Interprétation du Scree Plot

Le **Scree plot** est un graphique utilisé en Analyse en Composantes Principales (ACP) pour visualiser la variance expliquée par chaque composante principale ainsi que la variance cumulée. Voici une analyse détaillée du graphique fourni :

## Description du Graphique

1. Axe des Abscisses (X) : Représente le **nombre de composantes principales**.
2. Axe des Ordonnées (Y) : Représente la **variance expliquée** (en pourcentage).
3. Barres Bleues : Montrent la **variance expliquée par chaque composante principale**.
4. Ligne Rouge avec Marqueurs : Montre la **variance cumulée** (somme progressive de la variance expliquée).

## Observations Clés

### 1. Premières Composantes :

- PC1 : Explique **15.65%** de la variance totale. C'est la composante qui capture le plus de variance.
- PC2 : Explique **11.37%** de la variance, portant la variance cumulée à **27.02%**.
- PC3 : Explique **10.70%** de la variance, portant la variance cumulée à **37.72%**.

### 2. Coude (Elbow) dans le Scree Plot :

- Le "coude" se situe autour des **5 premières composantes**, qui expliquent ensemble **57.42%** de la variance. Après ce point, l'ajout de nouvelles composantes n'apporte qu'un gain marginal de variance.

### 3. Variance Cumulée :

- Après **10 composantes**, la variance cumulée atteint **77.60%**.
- Après **20 composantes**, elle atteint **90.36%**.
- Après **30 composantes**, elle atteint **95.61%**.

## Interprétation

### 1. Choix du Nombre de Composantes :

- **Critère du Coude** : Le "coude" suggère de retenir **5 composantes** pour une réduction optimale de la dimensionnalité tout en conservant une part significative de la variance (**57.42%**).
- **Critère de Variance Cumulée** : Si l'objectif est de conserver **80%** de la variance, il faudrait retenir **13 composantes**. Pour **90%**, il faudrait retenir **20 composantes**.

### 2. Dimensions Sous-Jacentes :

- PC1 : Représente probablement une dimension liée à la **richesse en capital** (forte contribution de `capital-gain` et `capital-loss`).
- PC2 : Représente probablement une dimension liée au **statut socio-professionnel** (forte contribution de `education-num` et `hours-per-week`).
- PC3 à PC5 : Pourraient représenter des dimensions supplémentaires, telles que l'âge, la situation familiale, ou le type d'emploi.

### 3. Implications pour l'Analyse :

- En retenant **5 à 13 composantes**, on peut réduire la dimensionnalité du dataset tout en conservant une grande partie de l'information.
- Les composantes supplémentaires (au-delà de PC5) expliquent une part de plus en plus faible de la variance, ce qui suggère qu'elles capturent des détails plus fins ou du bruit.

## Tableau Récapitulatif

Nombre de Composantes	Variance Expliquée (%)	Variance Cumulée (%)	Interprétation
1	15.65	15.65	PC1 : Richesse en capital.
2	11.37	27.02	PC2 : Statut socio-professionnel.
3	10.70	37.72	PC3 : Dimension démographique (âge, situation familiale).
4	10.34	48.06	PC4 : Type d'emploi et relations familiales.
5	9.36	57.42	PC5 : Diversité démographique (race, sexe).
10	2.31	77.60	Après 10 composantes, <b>77.60%</b> de la variance est expliquée.
20	0.84	90.36	Après 20 composantes, <b>90.36%</b> de la variance est expliquée.

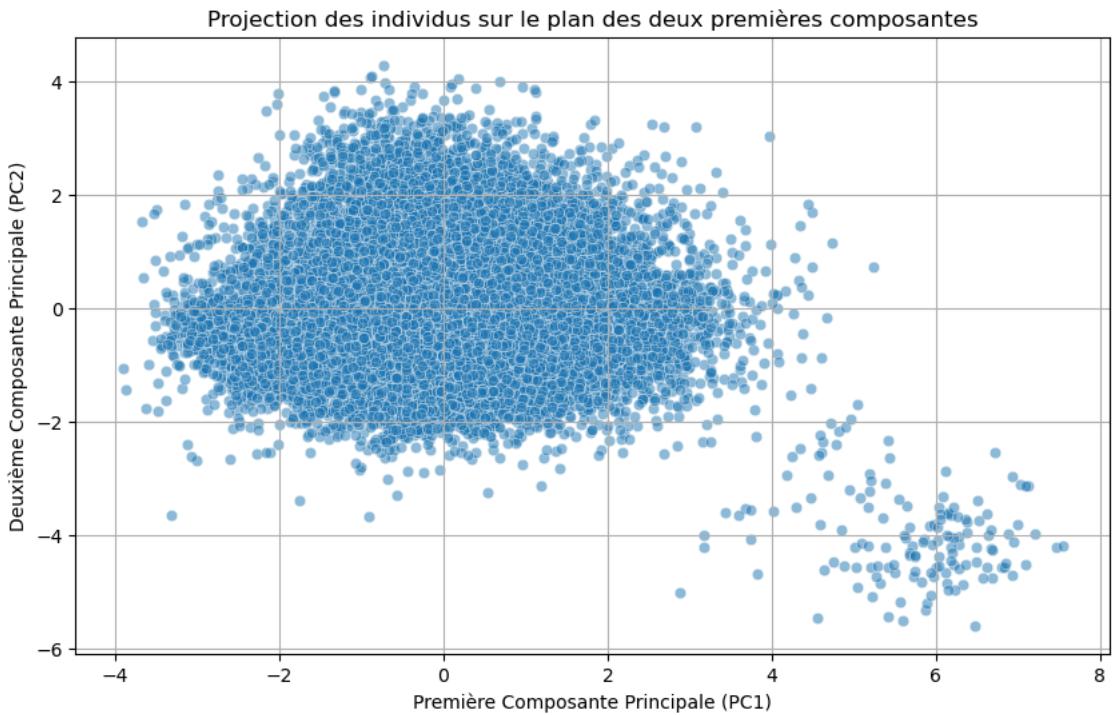
## Conclusion

Le Scree plot montre que les premières composantes principales capturent une part significative de la variance, tandis que les composantes supplémentaires apportent des gains marginaux. En retenant un nombre approprié de composantes (5 à 13), on peut réduire la dimensionnalité du dataset tout en conservant l'essentiel de l'information. Cette analyse permet de mieux comprendre la structure des données et de guider les étapes suivantes, telles que la classification ou la régression.

```
In [ ]:
```

```
In [178... # Projection des individus sur les deux premières composantes principales
df_pca = pd.DataFrame(X_pca[:, :2], columns=["PC1", "PC2"])

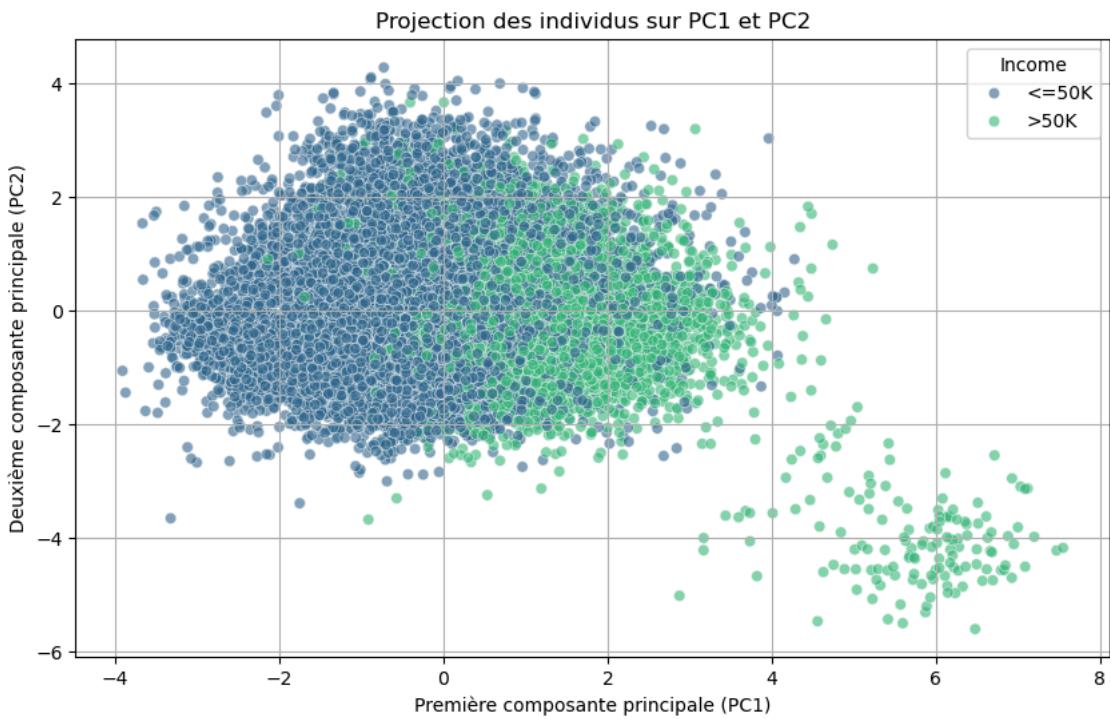
plt.figure(figsize=(10, 6))
sns.scatterplot(x=df_pca["PC1"], y=df_pca["PC2"], alpha=0.5)
plt.xlabel("Première Composante Principale (PC1)")
plt.ylabel("Deuxième Composante Principale (PC2)")
plt.title("Projection des individus sur le plan des deux premières composantes")
plt.grid(True)
plt.show()
```



## Interessant poursuivons pour avoir une separation Net

```
In [ ]:
```

```
In [180... # Projection des individus sur les deux premières composantes principales
plt.figure(figsize=(10, 6))
sns.scatterplot(x=X_pca[:, 0], y=X_pca[:, 1], hue=df_adult['income'], palette='viridis', alpha=0.6)
plt.xlabel('Première composante principale (PC1)')
plt.ylabel('Deuxième composante principale (PC2)')
plt.title('Projection des individus sur PC1 et PC2')
plt.legend(title='Income')
plt.grid(True)
plt.show()
```



- 1. PC1 :** Représente probablement une dimension liée à la **richesse en capital**, avec des contributions importantes des variables `capital-gain` et `capital-loss`.
- 2. PC2 :** Représente probablement une dimension liée au **statut socio-professionnel**, influencée par des variables comme `education-num` et `hours-per-week`.
- 3. Séparation des Groupes :** Les individus avec un revenu `>50K` semblent se regrouper dans une zone spécifique du graphique, suggérant une certaine séparation basée sur le revenu.
- 4. Recouvrement :** Il y a un chevauchement entre les deux groupes, indiquant que d'autres variables ou composantes sont nécessaires pour une séparation plus nette.
- 5. Interprétation :** Les deux premières composantes capturent une partie de la variance liée au revenu, mais d'autres dimensions pourraient être nécessaires pour une analyse plus approfondie.

In [ ]:

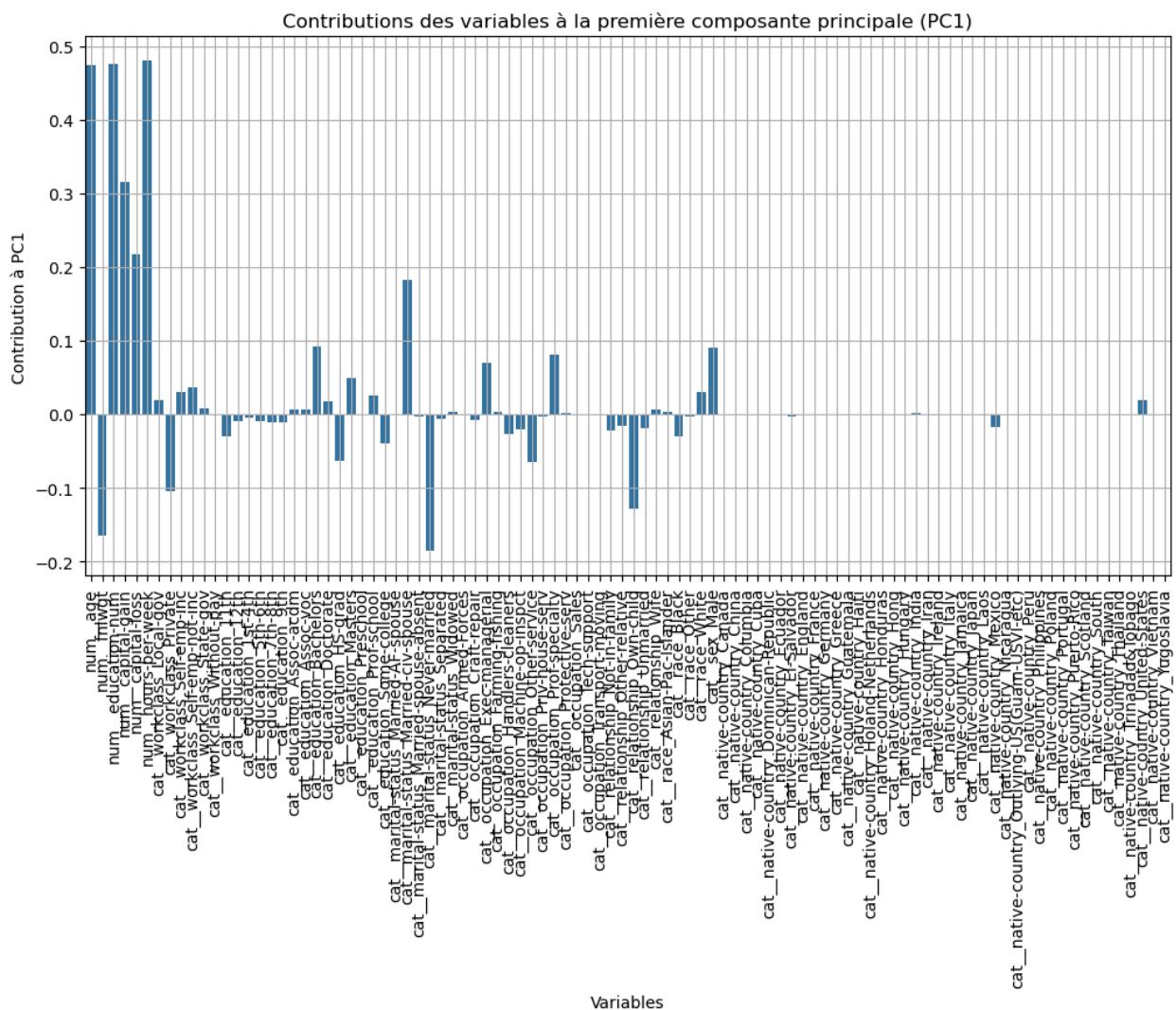
## Nous allons evaluer l'importance de PCA1

In [ ]:

```
# Importance des variables dans les composantes principales
# Récupérer les loadings (coefficients des variables dans les composantes)
loadings = pca.components_

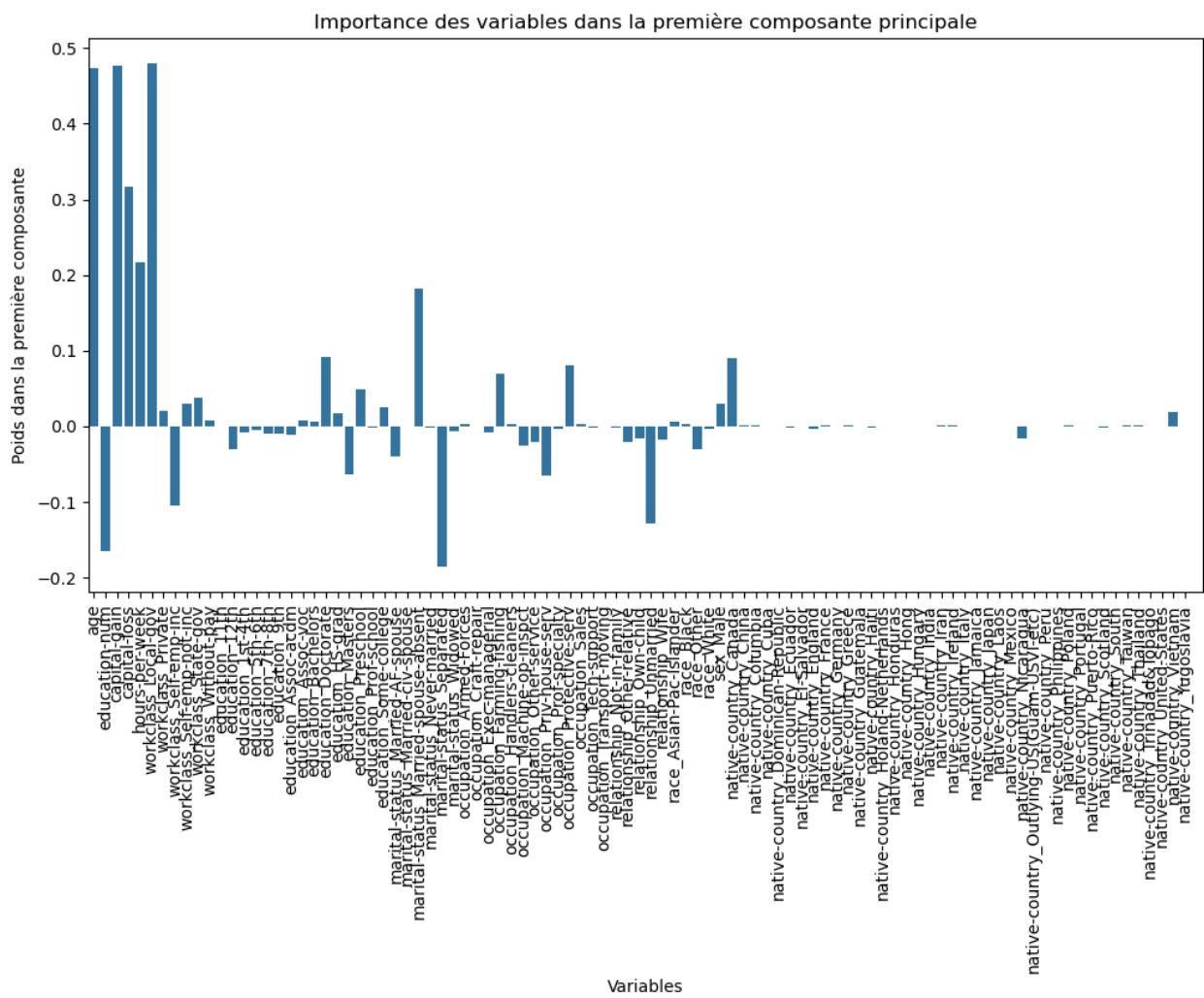
# Créer un DataFrame pour visualiser les contributions des variables
loadings_df = pd.DataFrame(loadings[:,2], columns=preprocessor.get_feature_names_out(), index=['PC1', 'PC2'])

# Visualiser les contributions des variables à PC1
plt.figure(figsize=(12, 6))
sns.barplot(x=loadings_df.columns, y=loadings_df.loc['PC1'])
plt.xticks(rotation=90)
plt.xlabel('Variables')
plt.ylabel('Contribution à PC1')
plt.title('Contributions des variables à la première composante principale (PC1)')
plt.grid(True)
plt.show()
```



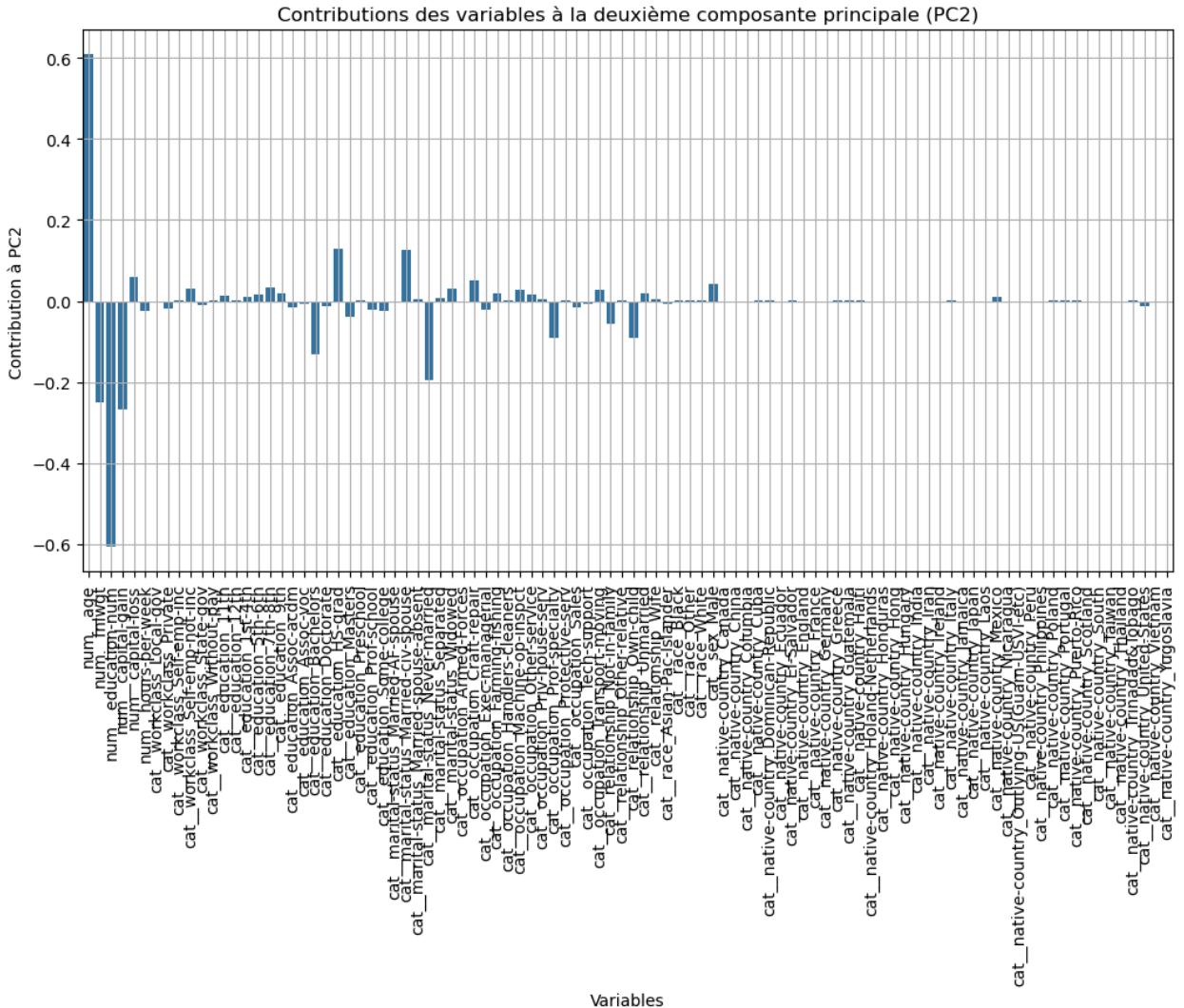
In [ ]:

```
In [197]: # Importance des variables dans la première composante principale (PC1)
plt.figure(figsize=(12, 6))
sns.barplot(x=np.arange(len(pca.components_[0])), y=pca.components_[0])
plt.xlabel("Variables")
plt.ylabel("Poids dans la première composante")
plt.title("Importance des variables dans la première composante principale")
plt.xticks(ticks=np.arange(len(df_processed.columns)), labels=df_processed.columns, rotation=90)
plt.show()
```



In [ ]:

```
In [192]: # Visualiser les contributions des variables à PC2
plt.figure(figsize=(12, 6))
sns.barplot(x=loadings_df.columns, y=loadings_df.loc['PC2'])
plt.xticks(rotation=90)
plt.xlabel('Variables')
plt.ylabel('Contribution à PC2')
plt.title('Contributions des variables à la deuxième composante principale (PC2)')
plt.grid(True)
plt.show()
```



In [ ]:

### 4.3 Analyse Factorielle des Correspondances (AFC)

- Construction de la Table de Contingence** : Élaborer une table de contingence croisant les variables *education* et *occupation*.
  - Réalisation de l'AFC** : Effectuer l'AFC pour identifier et visualiser les associations entre les catégories des deux variables.
  - Interprétation des Résultats** : Analyser le biplot résultant pour discerner les correspondances et associations significatives entre les niveaux d'éducation et les professions.

In [ ]:

## Introduction

L'Analyse Factorielle des Correspondances (AFC) est une méthode statistique qui permet d'explorer les relations entre deux variables catégorielles.

Dans cette analyse, nous avons étudié les correspondances entre les niveaux d'éducation et les professions à partir du jeu de données **Adult Income**.

Le biplot résultant nous permet de visualiser ces relations dans un espace à deux dimensions.

In [ ]:

```
In [9]: import panel as pn
import hvplot.pandas
import ssl

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.decomposition import PCA

In [7]: # Activer Panel avec Tabulator
pn.extension('tabulator')

# Désactiver la vérification SSL
ssl._create_default_https_context = ssl._create_unverified_context

# URL du dataset
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

# Chargement des données avec mise en cache
if 'data_adult' not in pn.state.cache.keys():
    df_adult = pd.read_csv(
        url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
    )

    # Suppression des espaces superflus dans les colonnes textuelles
    str_cols = df_adult.select_dtypes(object).columns
    df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

    # Stockage en cache
    pn.state.cache['data_adult'] = df_adult.copy()
else:
    df_adult = pn.state.cache['data_adult']

# Détection des valeurs "?"
missing_mask = df_adult == "?"

# Création d'une copie du DataFrame pour affichage HTML
df_display = df_adult.copy()

# Remplacement des valeurs "?" par une version colorée en HTML
for col in df_adult.columns:
    df_display[col] = df_adult[col].apply(
        lambda x: f'<span style="background-color:red; color:white; font-weight:bold; padding:3px;">{x}</span>' if x == "?" else x
    )

# Récapitulatif des valeurs "?" par colonne
missing_counts = missing_mask.sum().reset_index()
missing_counts.columns = ["Variable", "Nombre de valeurs '?'"]

# Mise en couleur des cellules ayant des valeurs manquantes
for col in missing_counts.columns:
    missing_counts[col] = missing_counts[col].astype(str)

missing_counts["Nombre de valeurs '?'"] = missing_counts["Nombre de valeurs '?']").apply(
    lambda x: f'<span style="background-color:red; color:white; font-weight:bold; padding:3px;">{x}</span>' if x != "0" else x
)

# Création du tableau interactif avec mise en rouge des valeurs non nulles
missing_table = pn.widgets.Tabulator(
    missing_counts,
    pagination='remote',
    page_size=20,
    formatters={col: {"type": "html"} for col in missing_counts.columns}, # Activation HTML
    configuration={"layout": "fitDataStretch"}
)

# Configuration du tableau principal avec formatage HTML
table = pn.widgets.Tabulator(
    df_display,
    pagination='remote',
    page_size=10,
    formatters={col: {"type": "html"} for col in df_adult.columns}, # Activer l'affichage HTML
    configuration={"layout": "fitDataStretch"}
)
```

```
# Affichage interactif du tableau avec le résumé des valeurs manquantes
dashboard = pn.Column(
    "## Tableau des données avec valeurs '?' mises en rouge",
    table,
    "# Dataset Adult - Valeurs manquantes",
    "● **Toutes les cellules contenant `?` sont colorées en rouge.**",
    "## Nombre de valeurs `?` par colonne (les valeurs non nulles sont mises en rouge)",
    "## variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country" ,
    missing_table,
)
dashboard.servable()
```

Out[7]: Tableau des données avec valeurs ? mises en rouge

index	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife
5	37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife
6	49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family
7	52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband
8	31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family
9	42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband

## Dataset Adult - Valeurs manquantes

● Toutes les cellules contenant ? sont colorées en rouge.

Nombre de valeurs ? par colonne (les valeurs non nulles sont mises en rouge)

variables catégorielles contenant des valeurs manquantes sont: workclass, occupation, native-country

index	Variable	Nombre de valeurs `?`
0	age	0
1	workclass	1836
2	fnlwgt	0
3	education	0
4	education-num	0
5	marital-status	0
6	occupation	1843
7	relationship	0
8	race	0
9	sex	0
10	capital-gain	0
11	capital-loss	0
12	hours-per-week	0
13	native-country	583
14	income	0

First Prev 1 Next Last

In [ ]:

In [ ]:

In [9]: `import pandas as pd  
import numpy as np`

```

import panel as pn
import hvplot.pandas
from sklearn.decomposition import TruncatedSVD
from sklearn.preprocessing import StandardScaler

# Activer l'extension Panel pour hvplot
pn.extension()

# Chargement des données
url_adult = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"

# Noms des colonnes
column_names_adult = [
    "age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
    "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss",
    "hours-per-week", "native-country", "income"
]

df_adult = pd.read_csv(
    url_adult, names=column_names_adult, sep=",", header=None, skipinitialspace=True
)

# Suppression des espaces superflus dans les colonnes textuelles
str_cols = df_adult.select_dtypes(object).columns
df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

# Étape 1 : Construction de la Table de Contingence
contingency_table = pd.crosstab(df_adult['education'], df_adult['occupation'])

# Étape 2 : Réalisation de l'AFC (Analyse Factorielle des Correspondances)
# Normalisation de la table de contingence
table_scaled = contingency_table.div(contingency_table.sum(axis=1), axis=0)

# Application de la décomposition SVD pour extraire les axes principaux
svd = TruncatedSVD(n_components=2)
coord = svd.fit_transform(table_scaled)

# Création d'un DataFrame pour les coordonnées des modalités
df_afc = pd.DataFrame(coord, columns=['Dim1', 'Dim2'], index=contingency_table.index)

# Étape 3 : Visualisation avec hvplot
afc_plot = df_afc.hvplot.scatter(
    x='Dim1', y='Dim2', hover_cols=['index'], title="Biplot de l'AFC",
    xlabel='Dimension 1', ylabel='Dimension 2', color='red', size=100,
    height=500, width=800
)

# Étape 4 : Affichage de la table de contingence
contingency_table_pane = pn.pane.DataFrame(
    contingency_table,
    sizing_mode='stretch_width',
    name='Table de Contingence'
)

# Étape 5 : Crédit à la création d'un tableau de bord interactif avec Panel
dashboard = pn.Column(
    "## Analyse Factorielle des Correspondances (AFC)",
    "### Table de Contingence : Education vs Occupation",
    contingency_table_pane,
    "### Biplot des modalités",
    afc_plot
)

# Affichage du tableau de bord
dashboard.servable()

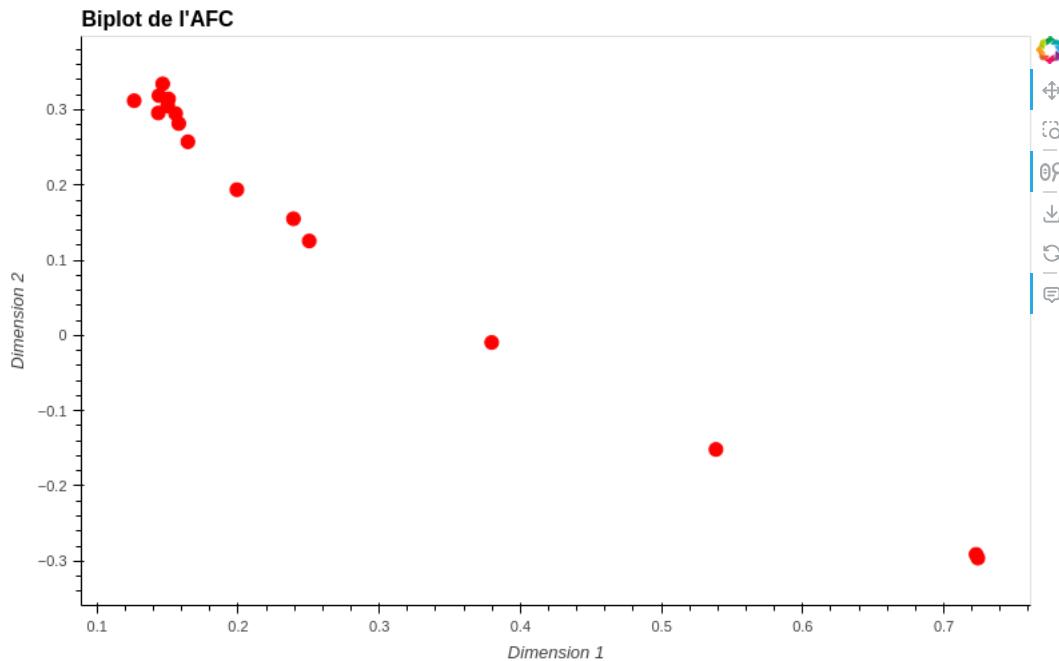
```

Out[9]: Analyse Factorielle des Correspondances (AFC)

Table de Contingence : Education vs Occupation

occupation	?	Adm-clerical	Armed-Forces	Craft-repair	Exec-managerial	Farming-fishing	Handlers-cleaners	Machine-op-inspect	Other-service	Priv-house-serv	Prof-specialty	Protective-serv	Sales	Tech-support	Tra
education															
10th	102	38	0	170	24	44	71	101	194	6	9	6	81	3	
11th	119	67	0	175	34	37	123	99	238	14	20	7	144	6	
12th	40	38	1	58	13	16	38	35	85	4	10	6	47	3	
1st-4th	12	0	0	23	4	18	16	23	40	11	4	1	8	0	
5th-6th	30	6	0	43	1	36	40	56	64	14	1	1	12	1	
7th-8th	73	11	0	116	19	70	46	93	98	8	9	9	29	5	
9th	51	14	0	96	13	28	49	76	101	10	3	4	32	2	
Assoc-acdm	47	193	0	115	145	14	24	33	78	2	138	34	144	73	
Assoc-voc	61	167	0	252	150	52	28	63	115	4	170	48	106	126	
Bachelors	173	506	1	226	1369	77	50	69	181	7	1495	100	809	230	
Doctorate	15	5	0	2	55	1	0	1	1	0	321	0	8	3	
HS-grad	533	1365	4	1922	807	404	611	1023	1281	50	233	215	1069	159	
Masters	48	68	1	22	501	10	5	8	19	1	844	15	134	37	
Preschool	5	2	0	4	0	9	2	11	15	2	1	0	0	0	
Prof-school	18	9	0	7	52	4	0	1	4	0	452	1	18	7	
Some-college	516	1281	2	868	879	174	267	310	781	16	430	202	1009	273	

Biplot des modalités



```
# Suppression des espaces superflus dans les colonnes textuelles
str_cols = df_adult.select_dtypes(object).columns
df_adult[str_cols] = df_adult[str_cols].apply(lambda x: x.str.strip())

# Étape 1 : Construction de la table de contingence entre `education` et `occupation`
cont_table = pd.crosstab(df_adult['education'], df_adult['occupation'])

# Étape 2 : Réalisation de l'AFC
# Normalisation de la table de contingence
table_scaled = cont_table.div(cont_table.sum(axis=1), axis=0)

# Application de la décomposition SVD pour extraire les axes principaux
svd = TruncatedSVD(n_components=2)
coord_rows = svd.fit_transform(table_scaled)

# Coordonnées des modalités `education`
row_labels = cont_table.index.tolist()
df_education_coords = pd.DataFrame(coord_rows, columns=['Dim1', 'Dim2'], index=row_labels)
df_education_coords['Type'] = 'Education'
```

```

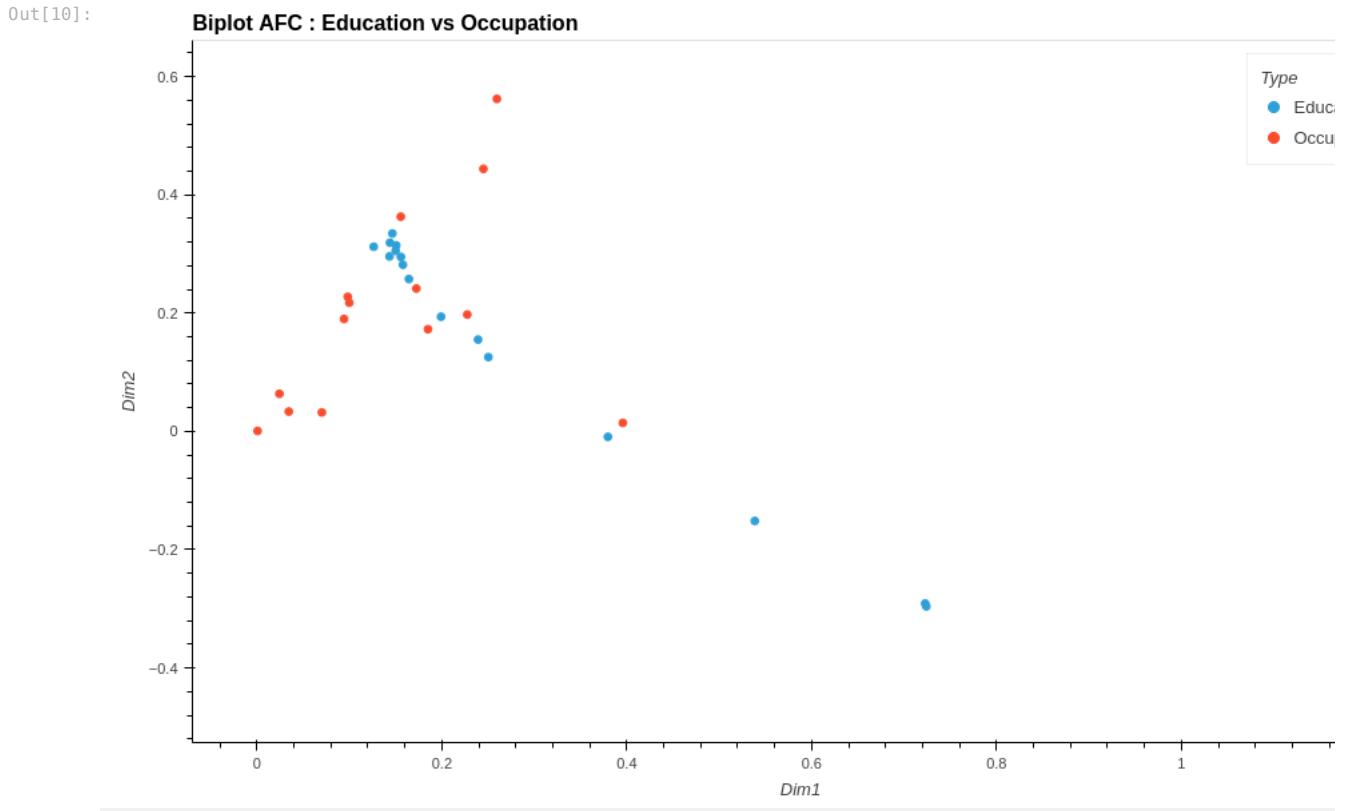
# Coordonnées des modalités `occupation`
coord_cols = svd.components_.T * svd.singular_values_
col_labels = cont_table.columns.tolist()
df_occupation_coords = pd.DataFrame(coord_cols, columns=['Dim1', 'Dim2'], index=col_labels)
df_occupation_coords['Type'] = 'Occupation'

# Fusion des données pour le biplot
df_afc = pd.concat([df_education_coords, df_occupation_coords])

# Étape 3 : Visualisation interactive avec hvplot
biplot = df_afc.hvplot.scatter(
    x='Dim1', y='Dim2', by='Type', hover_cols=['Type'],
    title="Biplot AFC : Education vs Occupation",
    legend='top_right',
    width=1000, # Augmentation de la largeur du graphique
    height=600 # Augmentation de la hauteur du graphique
)

# Affichage du biplot interactif
biplot

```



```

In [57]: # Calcul de l'inertie expliquée
singular_vals = svd.singular_values_
inertias = (singular_vals**2) / np.sum(singular_vals**2) * 100
inertia_df = pd.DataFrame({
    'Dimension': ['Dim1', 'Dim2'],
    'Inertie (%)': inertias[:2]
})

```

# Affichage du tableau de l'inertie expliquée  
print("\nInertie Expliquée par les Dimensions")  
display(inertia\_df)

Inertie Expliquée par les Dimensions

Dimension Inertie (%)

	Dim1	Dim2
0	62.665988	
1	37.334012	

In [ ]:

In [ ]:

## Analyse et Interprétation du Biplot de l'afc : Éducation vs. Occupation

L'Analyse Factorielle des Correspondances (AFC) permet d'explorer les relations entre **deux variables qualitatives**, ici **le niveau d'éducation** (`education`) et **le type d'occupation** (`occupation`). Le **biplot** obtenu projette ces modalités dans un espace à **deux dimensions principales** (`Dim1` et `Dim2`), qui capturent une grande partie de l'information de la relation entre les deux variables.

## 1. Explication du Graphique

- Les points bleus représentent les niveaux d'éducation ( Education ).
- Les points rouges représentent les types d'emploi ( Occupation ).
- L'axe Dim1 est celui qui capte le plus de variance dans les relations éducation-emploi.
- L'axe Dim2 est secondaire et représente des différences moins marquées.

Chaque point représente une modalité, et la proximité entre un point Education et un point Occupation indique une association forte.

---

## 2. Interprétation des Résultats

### a) Relation entre éducation et emploi

- Proximité des points Education et Occupation :
  - Lorsque un niveau d'éducation est proche d'un type d'occupation, cela signifie que ce type d'emploi est plus fréquent pour les personnes ayant ce niveau d'éducation.
  - Par exemple, si "High School" (éducation) est proche de "Blue-collar" (emploi manuel), cela indique que les personnes ayant un diplôme de lycée travaillent souvent dans des emplois manuels.
  - Si "Bachelors" ou "Masters" est proche de "White-collar", cela traduit une corrélation forte entre diplômes universitaires et métiers de bureau.
- Éloignement des points :
  - Plus deux modalités sont éloignées, plus elles sont faiblement associées.
  - Si une modalité d'éducation est loin de toutes les modalités d'occupation, cela signifie qu'elle est répartie de manière homogène entre plusieurs types de professions.

### b) Analyse des Axes Factorielles

- Dim1 (Axe horizontal) :
  - Capture l'opposition entre les niveaux d'éducation faible et élevé.
  - Sépare généralement les emplois peu qualifiés (travailleurs manuels, ouvriers) à gauche et les emplois qualifiés (professionnels, managers) à droite.
- Dim2 (Axe vertical) :
  - Peut représenter des variations secondaires, comme des spécialisations au sein d'un même groupe.
  - Si certaines catégories de travail sont plus dispersées verticalement, cela peut refléter des nuances régionales, sectorielles ou culturelles.

### c) Structure Globale

- Si la dispersion des points est forte, cela signifie que les niveaux d'éducation influencent fortement la répartition des professions.
- Si certains points sont très éloignés du centre, ces modalités ont des caractéristiques fortement distinctives (exemple : un type d'emploi spécifique qui est quasi-exclusivement occupé par un certain niveau d'éducation).
- Si certains points sont regroupés près du centre, cela signifie que ces catégories sont peu discriminantes, c'est-à-dire qu'elles ne sont pas fortement associées à un emploi particulier.

## 3. Conclusions

1. La répartition des emplois est largement influencée par le niveau d'éducation. Certains niveaux d'éducation sont fortement associés à certains types de professions.
2. L'axe Dim1 semble principalement représenter la distinction entre professions qualifiées et non qualifiées.
3. Certaines catégories d'emploi sont très spécifiques à certains niveaux d'éducation, tandis que d'autres sont plus généralistes.
4. Les points proches révèlent des tendances claires : les travailleurs manuels sont souvent peu diplômés, tandis que les professions intellectuelles requièrent des études plus longues.

In [ ]:

In [ ]: