

# Analyse des Salaires avec le Dataset “Adult- Census-Income”

# Plan :

- 01     **Introduction et Objectifs du Projet**
- 02     **Compréhension et Préparation des Données**
- 03     **Exploration des Données, Classification et Clustering**
- 04     **Conclusion**





# 01

# Introduction et

# Objectifs



# Cadre du projet :

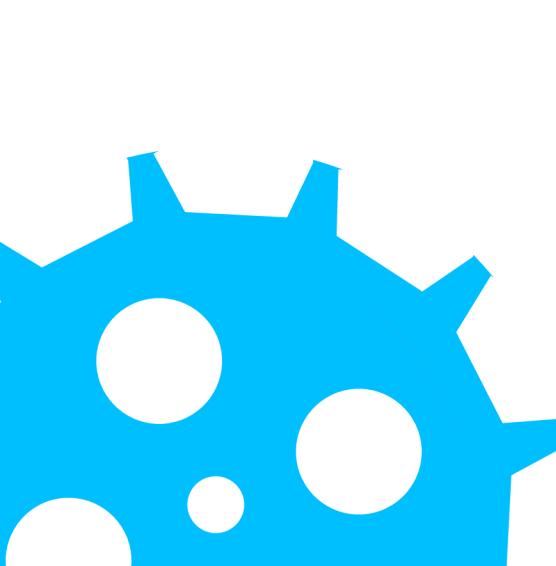
Le projet vise à prédire si le salaire d'un individu est supérieur ou inférieur à 50 000 \$ en utilisant le dataset **Adult-Census-Income**

## Objectif :

- Développer des modèles de machine learning pour classifier les individus en deux catégories : salaires supérieurs à 50K \$/an et salaires inférieurs à 50K \$/an
- Appliquer des techniques de classification et de clustering afin d'identifier les facteurs influençant les salaires
- Améliorer la précision des prédictions grâce à l'analyse des données et l'optimisation des modèles

# Défis à surmonter :

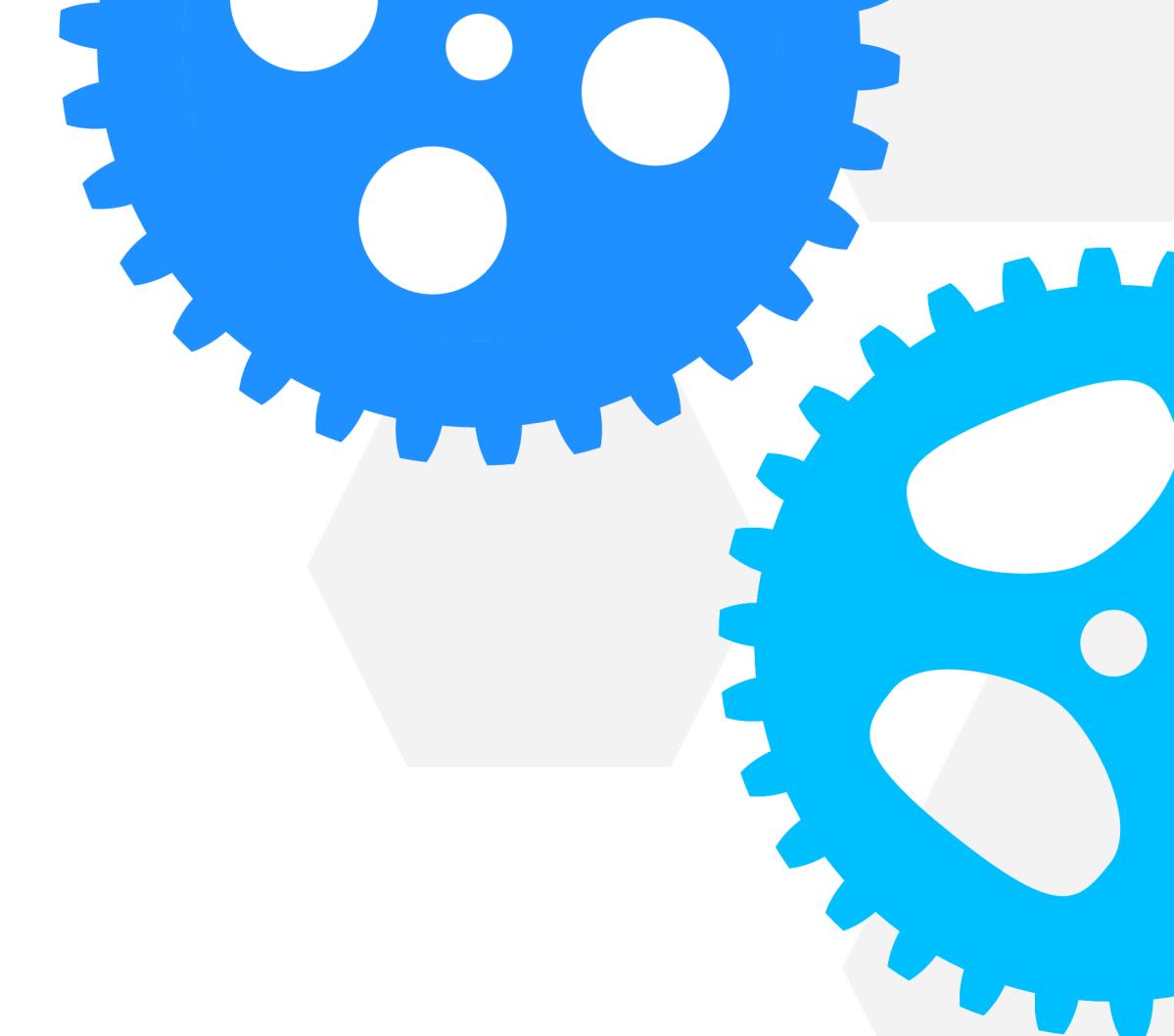
- Quels sont les features nécessaires pour juger la classe de revenu ?
- Quel algorithme de machine learning sera le plus performant ?
- Comment évaluer la performance du modèle choisi ?





02

# Compréhension et Préparation des Données

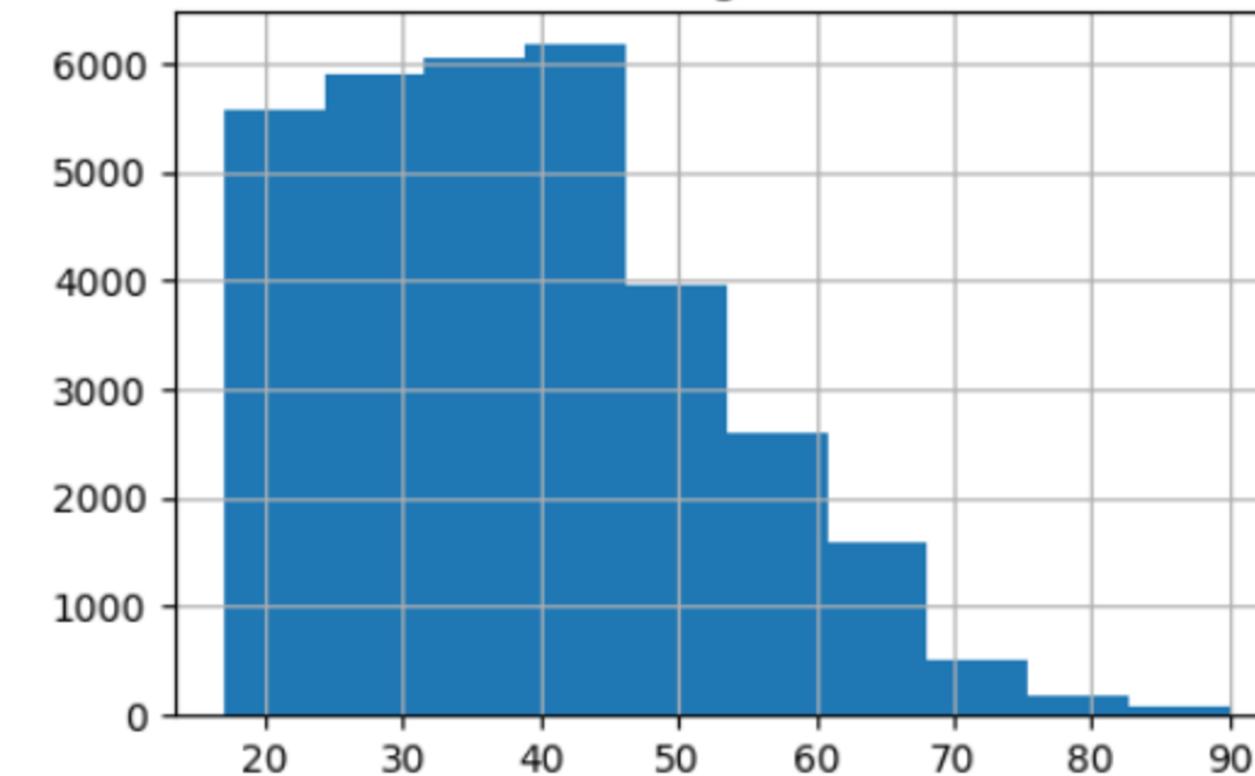


# Compréhension des Données :

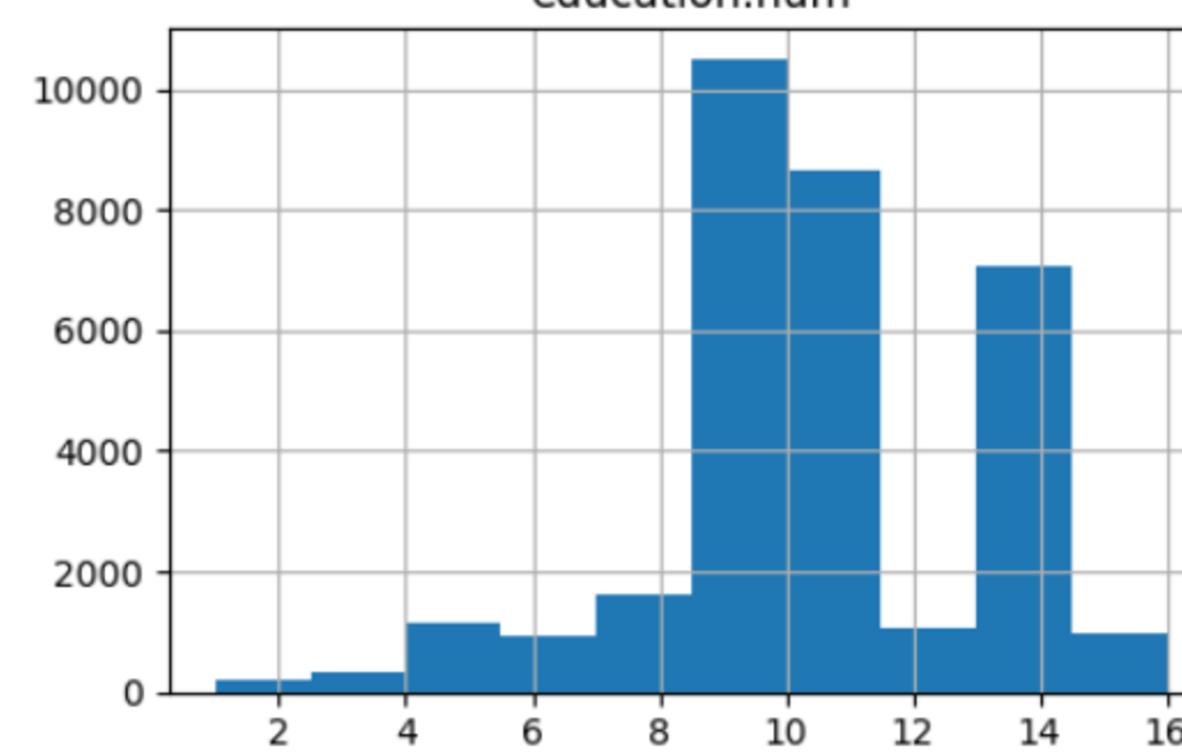
- Dataset avec 32 000 individus
- Attributs numériques : Age, Capital gain, Capital loss, Hours per week
- Attributs catégoriques : Workclass, Education, Marital status, Occupation, Race, Sex
- Distribution des classes : 75% des individus gagnent moins de 50K \$/an, 25% gagnent plus de 50K \$/an

# Compréhension des Données :

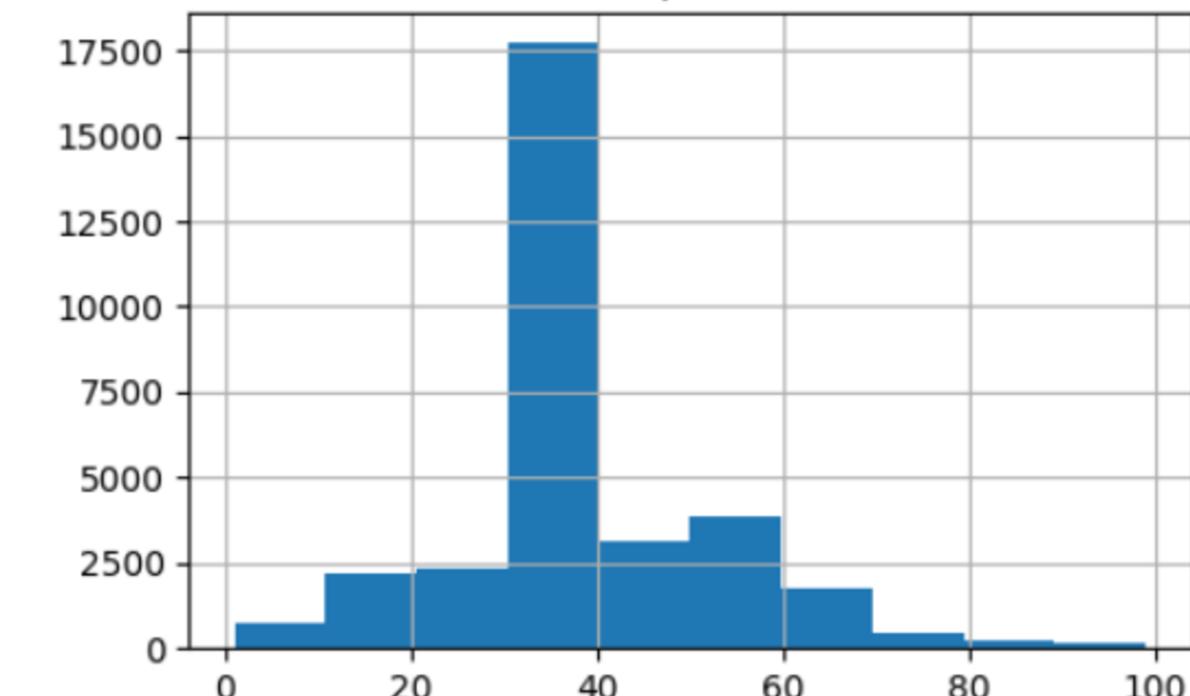
age



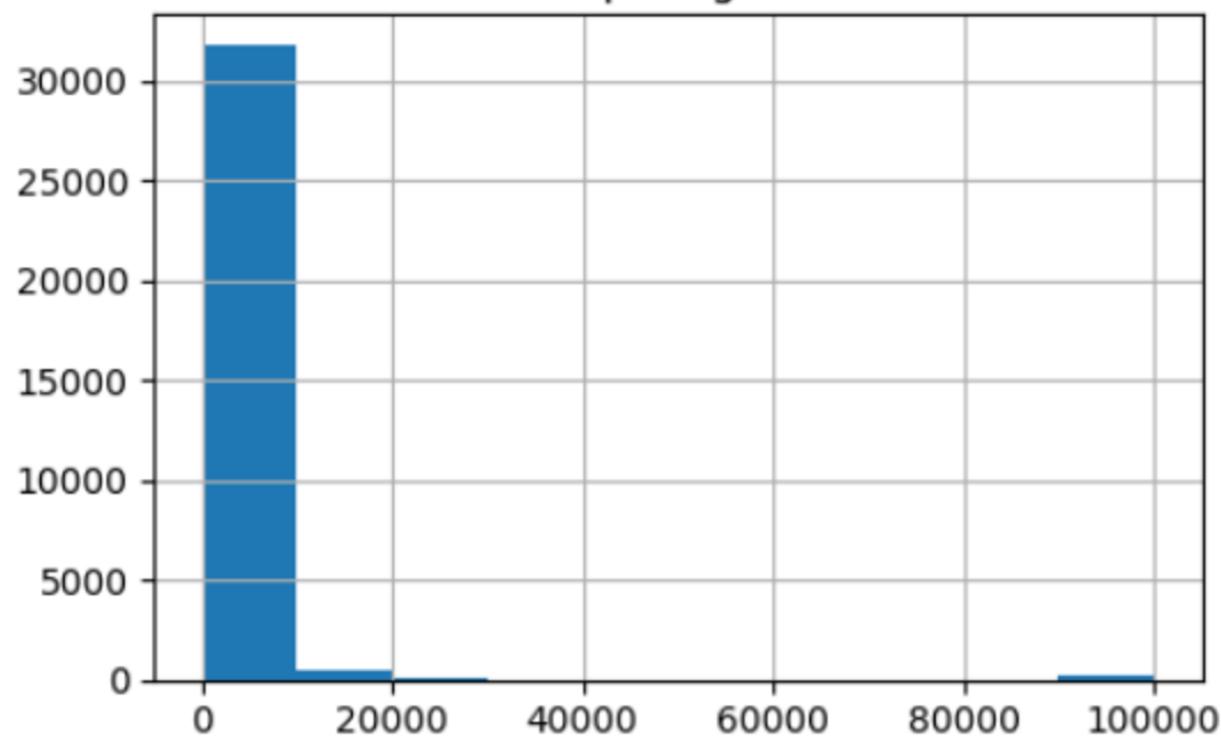
education.num



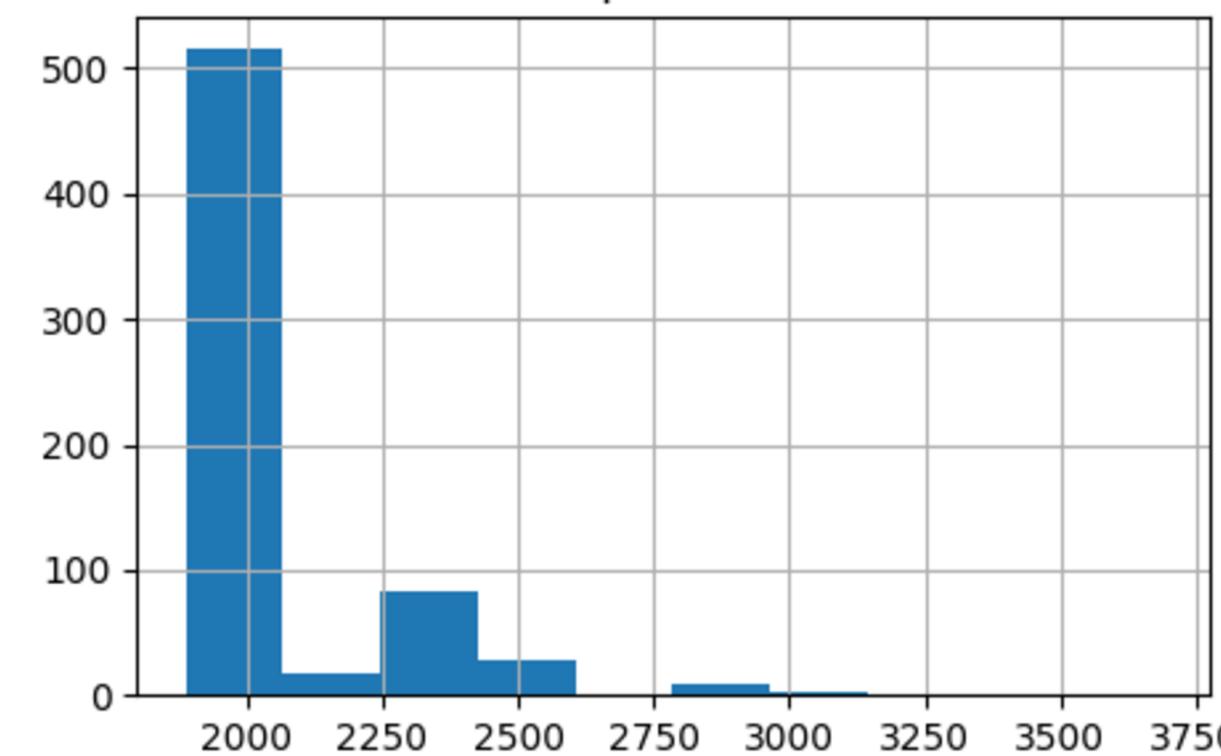
hours.per.week



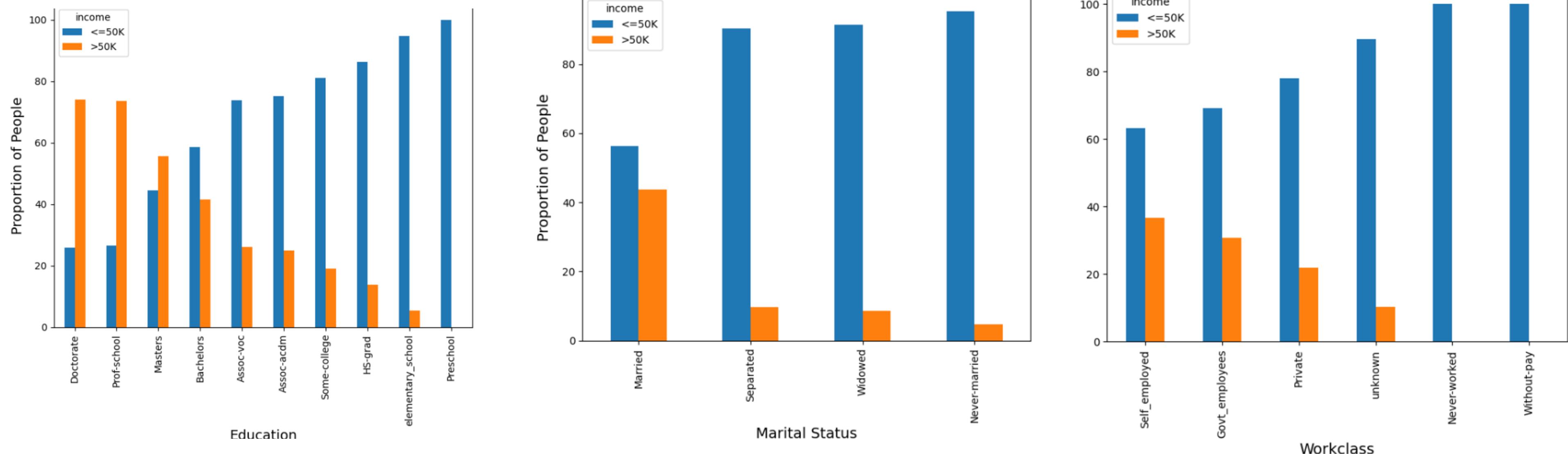
capital.gain



capital.loss



# Compréhension des Données :



# Préparation des Données :

- **Nettoyage des données** : Traitement des valeurs manquantes dans certaines colonnes (Workclass, Occupation..) en les remplaçant par des modalités ou des moyennes afin de garantir la cohérence des données pour l'apprentissage
- **Encodage des variables catégorielles** : Les variables comme Workclass, Marital status, Occupation, Race, et Sex ont été transformées en variables numériques via l'encodage One-Hot ou d'autres techniques adaptées, facilitant ainsi leur utilisation par les modèles de machine learning.

# Préparation des Données :

- **Normalisation des données** : Mise à l'échelle des variables numériques (par exemple, Age, Capital gain, Hours per week) afin de les amener à une même échelle
- **Gestion des Outliers** : Identification des valeurs extrêmes (par exemple, un capital gain de 99999 ou des heures de travail non réalistes) et prise de décision sur leur suppression ou correction pour éviter qu'elles n'influencent négativement les performances du modèle
- **Feature Engineering** : Sélection et transformation des variables importantes pour améliorer le modèle. Par exemple, la fusion de Education (variable catégorielle) et Education-num (variable numérique) permet de réduire la redondance tout en préservant l'information essentielle sur le niveau d'éducation



# 03

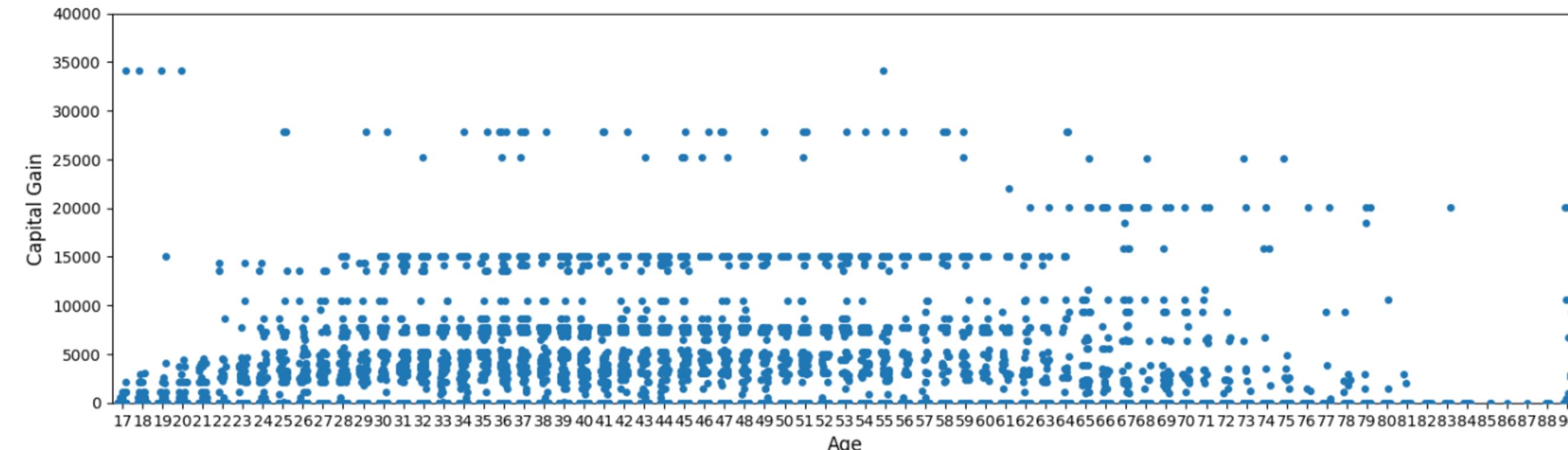
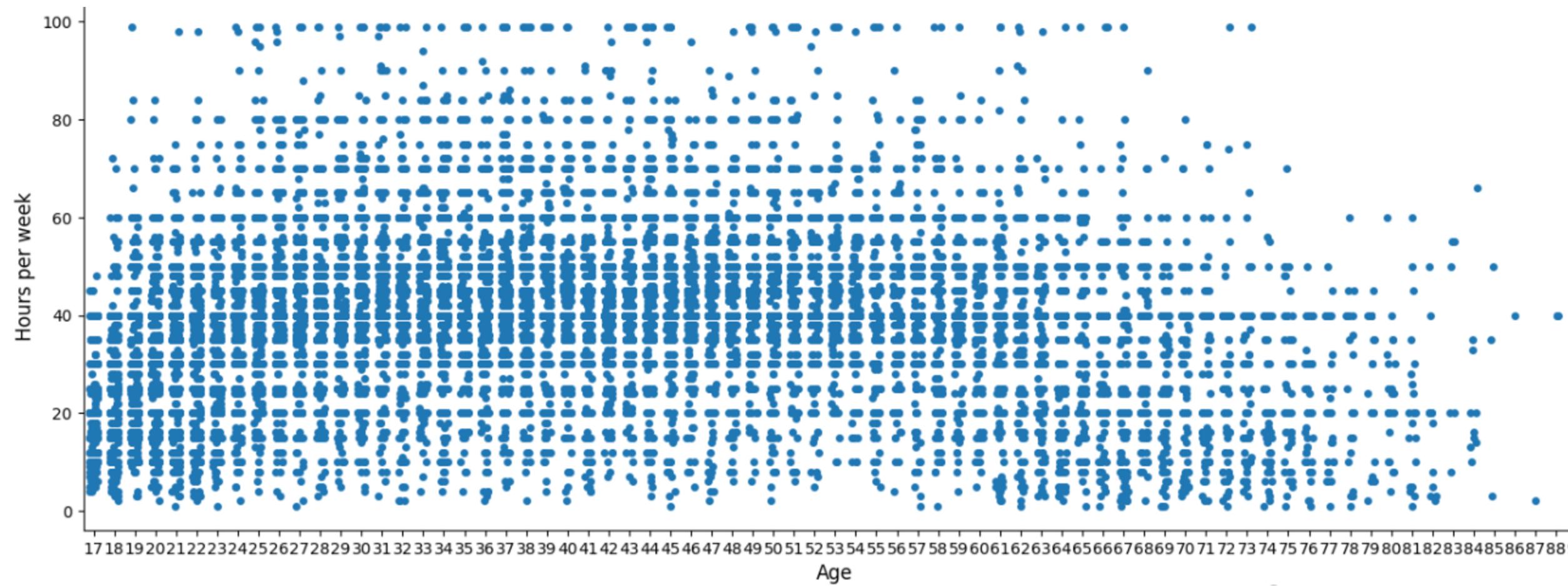
# Exploration des Données, Classification et Clustering



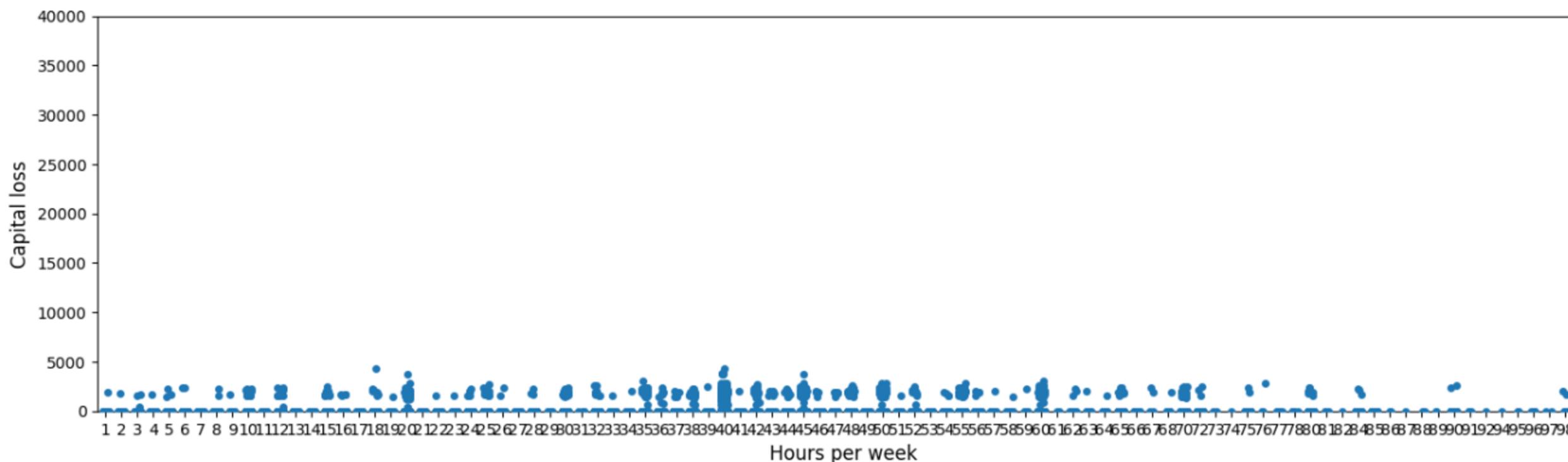
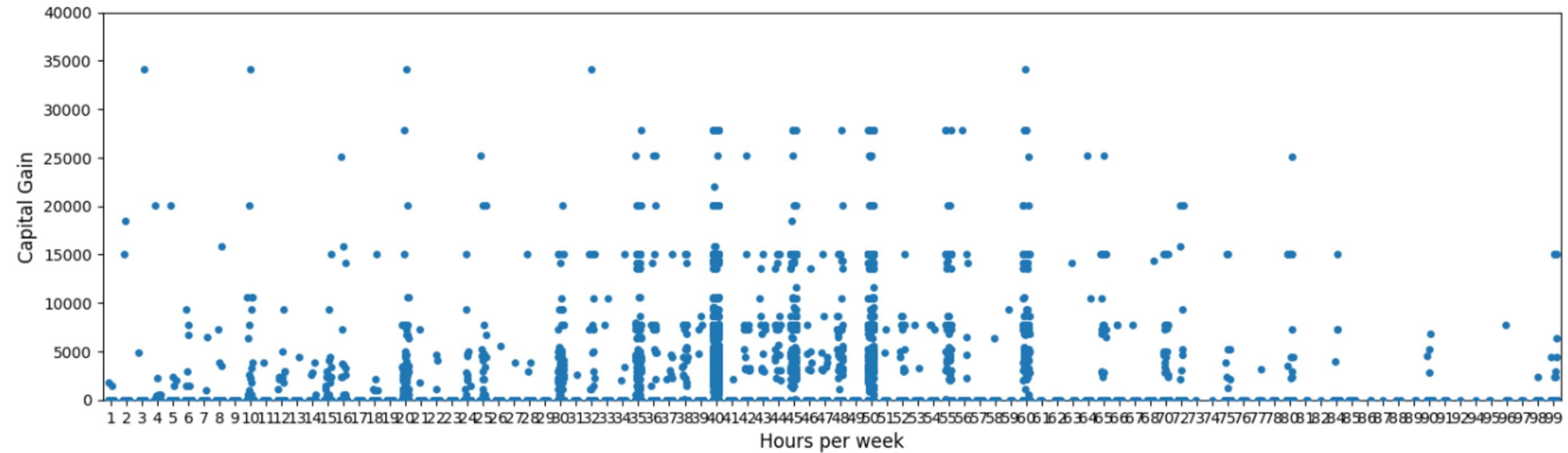
# Exploration des Données :

- **Visualisation des relations entre les features** : Utilisation de scatter plots et de pair plots pour observer les interactions entre les variables et détecter d'éventuelles relations non linéaires
- **Analyse des corrélations** : Identification des features les plus influentes sur la variable cible à l'aide de coefficients de corrélation afin de sélectionner les plus pertinentes pour les modèles.
- **Visualisation des données** : Utilisation d'histogrammes, de boxplots et de matrices de corrélation pour mieux comprendre les tendances et les relations entre les variables

# Aperçu des relations entre les Données :

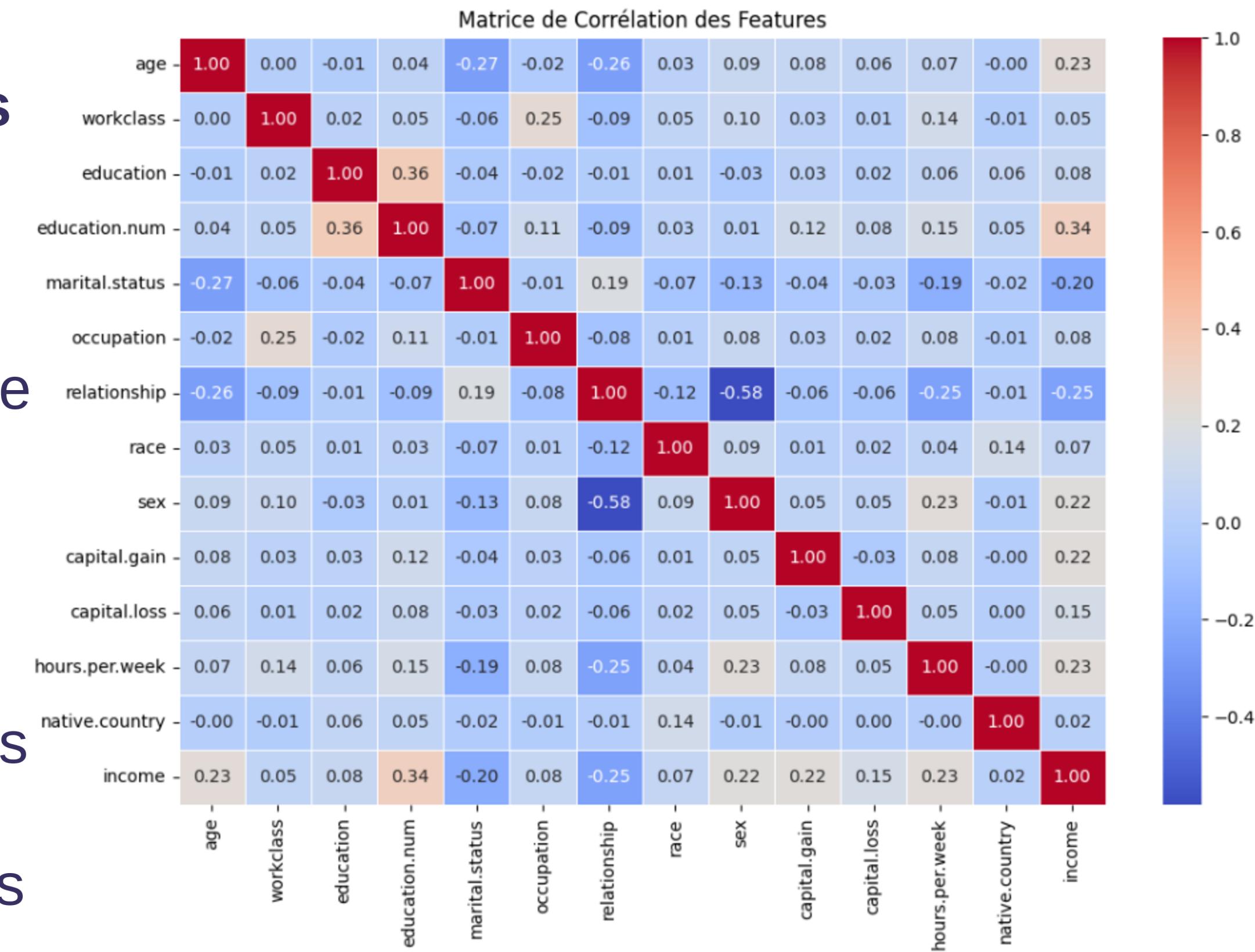


# Aperçu des relations entre les Données :



# Matrice de Corrélation :

- Une forte corrélation a été observée entre **age** et **hours per week** .
- Une forte corrélation a également aussi trouvée entre le **capital gain** et **hours per week** .
- Une corrélation quasiment faible a été observée entre les autres variables, suggérant des relations moins marquées entre elles.



# Application des algorithmes de Classification:

- **Logistic Regression** : Classification binaire.
- **Decision Tree** : Arbre hiérarchique.
- **K-Nearest Neighbors (K-NN)** : Apprentissage supervisé.
- **Support Vector Classifier (SVC)** : Séparation optimale.
- **Random Forest** : Forêt d'arbres.
- **Adaboost** : Amélioration itérative.
- **Bagging** : Rééchantillonnage bootstrap.
- **Extra Trees** : Arbres aléatoires.
- **Naive Bayes** : Probabilités conditionnelles.

# Performances des algorithmes de classification :

Modèle	Accuracy	Recall	Spécificité	FPR	Précision	F1-Score
Logistic Regression	84.28%	93.20%	55.21%	44.79%	87.14%	0.90
Decision Tree	79.86%	86.78%	57.33%	42.67%	86.88%	0.87
K-Nearest Neighbors	81.08%	90.72%	49.71%	50.29%	85.45%	0.88
Support Vector Classifier	84.08%	93.76%	52.57%	47.43%	86.55%	0.90
Random Forest	83.99%	92.26%	57.07%	42.93%	87.50%	0.90
Adaboost	84.15%	95.68%	46.64%	53.36%	85.38%	0.90
Bagging	82.11%	89.09%	59.40%	40.60%	87.72%	0.88
Extra Trees	82.31%	90.20%	56.64%	43.36%	87.14%	0.89
Naive Bayes	30.51%	9.80%	97.94%	2.06%	93.93%	0.18

# Interprétation des résultats :

- **Logistic Regression** et **SVC** sont les meilleurs modèles avec des scores élevés en précision, rappel et F1-score.
- **Adaboost** a un excellent rappel, mais une précision plus faible et un FPR élevé.
- **Naive Bayes** a un rappel faible et un F1-score bas, ce qui le rend peu performant.
- **Random Forest**, **Extra Trees**, **Bagging**, et **K-NN** offrent des résultats équilibrés avec de bonnes performances.
- **Naive Bayes** a une spécificité élevée, mais une faible capacité de détection.

# Clustering avec KMeans :

- **K-Means** n'est pas assez performant sur ce dataset car les données ne forment pas des groupes séparables dans l'espace des features ; la distribution des individus selon income est trop imbriquée, ce qui empêche l'algorithme de bien distinguer les deux classes.
- Le modèle K-Means initial, avec ses paramètres par défaut (`n_clusters=2`), a produit une inertie élevée de **403476.41**, indiquant un mauvais regroupement des données. En optimisant les paramètres via la recherche automatique (**Elbow Method** et **Score de Silhouette**), le nombre optimal de clusters a été trouvé à 7, avec l'algorithme elkan, réduisant l'inertie à **278079.44**

THE  
END



# 04 CONCLUSION

# Conclusion :

- **Préparation des données** : La compréhension des données et le nettoyage (gestion des valeurs manquantes, normalisation) sont essentiels pour améliorer les performances des modèles.
- **Évaluation des modèles** : Plusieurs modèles ont été testés, avec des performances variables. **Logistic Regression** et **SVC** ont montré de bons résultats, notamment en recall et précision.
- **Clustering avec KMeans** : Les résultats de KMeans ont été décevants, en raison de la faible séparabilité des données, malgré l'optimisation des hyperparamètres.



**MERCI POUR VOTRE ATTENTION !**