جامعة عبد المالك السعدي
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵄⴰⴱⴷⵍⵎⴰⵍⵉⴽ ⵙⵙⴰⵄⴷⵉ
Université Abdelmalek Essaâdi

كلية العلوم و التقنيات بطنجة
ⵜⴰⵖⵣⵡⵜ ⵏ ⵜⵣⵔⴼⵜ ⴷ ⵜⵓⵙⵙⵙⵉⵍⵜ ⵏ ⵟⴰⵏⵊⴰ
Faculté des Sciences et Techniques de Tanger

Département Génie Informatique

Cycle Ingénieur: LSI s3

Pr. *EL AACHAK LOTFI*

2024/2025

Atelier 1 : Régression linéaire

Élaborer par : *MOHAMED ILIASS KADDAR*

Module : Machine Learning

Introduction

L'objectif principal de cet atelier est de se familiariser avec les méthodes de régression en apprentissage automatique et d'appliquer ces techniques sur des jeux de données réels. Le but est d'apprendre à explorer des données, à préparer un jeu d'entraînement, à construire et comparer différents modèles de régression, puis à évaluer leurs performances à l'aide d'indicateurs quantitatifs. Ce travail permet de comprendre comment les modèles prédisent des valeurs continues et d'analyser leurs limites dans des contextes réels.

Dans le cadre de cet atelier, nous étudions trois familles de modèles de régression :

Régression linéaire simple : ce modèle cherche à établir une relation linéaire entre une seule variable explicative (feature) et une variable cible (target). Mathématiquement, il s'agit d'ajuster une droite qui minimise l'erreur entre les valeurs réelles et les valeurs prédites. Ce modèle est simple, facile à interpréter et constitue souvent une première approche pour vérifier si une relation linéaire existe entre deux variables.

Régression linéaire multiple : extension de la régression linéaire simple, ce modèle prend en compte plusieurs variables explicatives simultanément. La régression multiple permet de modéliser des phénomènes dont la cible dépend de plusieurs facteurs et nécessite souvent un prétraitement (encodage des variables catégorielles, standardisation des variables numériques, sélection de features).

Régression polynomiale : lorsque la relation entre les variables n'est pas linéaire, on peut transformer les features en fonctions polynomiales (puissances, interactions) et appliquer une régression linéaire sur ces nouvelles variables. La régression polynomiale permet de modéliser des courbes (non-linéaires) tout en conservant la formulation de régression linéaire sur les features transformées. Attention toutefois au risque de surapprentissage (overfitting) si le degré polynomial est trop élevé.

Les jeux de données utilisés pour cet atelier sont les suivants (sources) :

- **Expérience et Salaire** — dataset servant d'exemple classique pour la régression linéaire simple (Years of Experience vs Salary).

  Source : Kaggle.

- **Assurance** — dataset contenant des caractéristiques démographiques et médicales (age, sex, bmi, children, smoker, region) et la variable cible charges (coûts d'assurance). Utilisé pour la régression linéaire multiple.

  Source : Kaggle.

- **China GDP** — série temporelle du produit intérieur brut (GDP) de la Chine par année, utilisée pour comparer une régression linéaire simple et une régression polynomiale (modélisation de croissance non linéaire).

  Source : CognitiveClass / IBM

Les outils employés pour réaliser cet atelier sont : **Python**, avec les bibliothèques **Pandas** pour la manipulation des données, **Matplotlib** et **Seaborn** pour la visualisation, et **Scikit-Learn** pour la mise en œuvre, l'entraînement et l'évaluation des modèles.

Partie 1 : Exploration & Visualisation des Données (EDA)

1.1. Chargement des données et Description statistique

On utilise Pandas pour charger les données depuis les datasets.

```
1  import pandas as pd
2
3  df1 = pd.read_csv(r'C:\Users\imk\Desktop\LSI\Maching learning\data sets\Salary_Data.csv')
4  df3 = pd.read_csv(r'C:\Users\imk\Desktop\LSI\Maching learning\data sets\insurance.csv')
```

Interprétation du dataset *df1 : Years & Salary*

Le dataset **df1** contient 30 observations et deux variables numériques : les années d'expérience et le salaire correspondant. L'analyse exploratoire montre un jeu de données propre : aucune valeur manquante, aucune ligne dupliquée, et les deux variables sont parfaitement adaptées à une régression linéaire simple. Les statistiques descriptives indiquent une progression globale du salaire avec l'expérience, avec un salaire moyen d'environ **76 000 $** et des valeurs comprises

entre **37 731 $** et **122 391 $**, ce qui suggère une relation positive cohérente avec la théorie économique.

En résumé, **df1 est un dataset propre, simple et idéal pour modéliser la relation linéaire entre expérience et salaire**. La petite taille du dataset (30 lignes) limite la puissance prédictive d'un modèle.

Interprétation du dataset *df3 : Insurance dataset*

Le dataset **df3** contient 1338 individus décrits par sept variables, mélangeant numériques (âge, BMI, enfants, charges) et catégorielles (sexe, fumeur, région). Les données sont complètes : aucune valeur manquante ni doublon. Sa taille importante le rend bien adapté à l'entraînement de modèles statistiques ou de machine learning. Les statistiques montrent une grande variabilité des charges d'assurance, influencées par des facteurs médicaux et personnels — notamment le statut de fumeur, l'âge et le BMI — qui sont des déterminants connus du coût de l'assurance santé.

En somme, **df3 est un dataset propre, riche et suffisamment volumineux pour développer des modèles prédictifs fiables**. Il permet d'analyser la manière dont les caractéristiques individuelles influencent les dépenses médicales, ce qui en fait un excellent jeu de données pour les tâches de régression multivariée.

```
df1 : years & salary dataset
Aperçu des premières lignes :
   YearsExperience   Salary
0              1.1  39343.0
1              1.3  46205.0
2              1.5  37731.0
3              2.0  43525.0
4              2.2  39891.0

Informations générales :

RangeIndex: 30 entries, 0 to 29
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   YearsExperience  30 non-null     float64
 1   Salary           30 non-null     float64
dtypes: float64(2)
memory usage: 612.0 bytes
None

Nombre de lignes et colonnes : (30, 2)
Noms des colonnes : ['YearsExperience', 'Salary']
données statistiques
       YearsExperience         Salary
count        30.000000      30.000000
mean          5.313333   76003.000000
std           2.837888   27414.429785
min           1.100000   37731.000000
25%           3.200000   56720.750000
50%           4.700000   65237.000000
75%           7.700000  100544.750000
max          10.500000  122391.000000
=== Validation de la propreté du dataset ===
Nombre de lignes dupliquées : 0

Valeurs manquantes par colonne :YearsExperience    0
Salary             0
dtype: int64
```

```
df3 : assurance dataset
Aperçu des premières lignes :
   age     sex     bmi  children smoker     region      charges
0   19  female  27.900         0    yes  southwest  16884.92400
1   18    male  33.770         1     no  southeast   1725.55230
2   28    male  33.000         3     no  southeast   4449.46200
3   33    male  22.705         0     no  northwest  21984.47061
4   32    male  28.880         0     no  northwest   3866.85520

Informations générales :

RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1338 non-null   int64
 1   sex       1338 non-null   object
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64
 4   smoker    1338 non-null   object
 5   region    1338 non-null   object
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
None

Nombre de lignes et colonnes : (1338, 7)
Noms des colonnes : ['age', 'sex', 'bmi', 'children', 'smoker', 'region', 'charges']
données statistiques :
               age          bmi     children       charges
count  1338.000000  1338.000000  1338.000000   1338.000000
mean     39.207025    30.663397     1.094918  13270.422265
std      14.049960     6.098187     1.205493  12110.011237
min      18.000000    15.960000     0.000000   1121.873900
25%      27.000000    26.296250     0.000000   4740.287150
50%      39.000000    30.400000     1.000000   9382.033000
75%      51.000000    34.693750     2.000000  16639.912515
max      64.000000    53.130000     5.000000  63770.428010
Validation de la propreté du dataset
0
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

1.2 Visualisation du nuage des points des deux data sets :

Dataset 1 : Expérience vs Salaire

Graphique scatter plot qui montre la relation entre l'expérience et le salaire

- Conclusion : relation linéaire claire

Dataset 2 : Assurance

Encodage des variables catégorielles

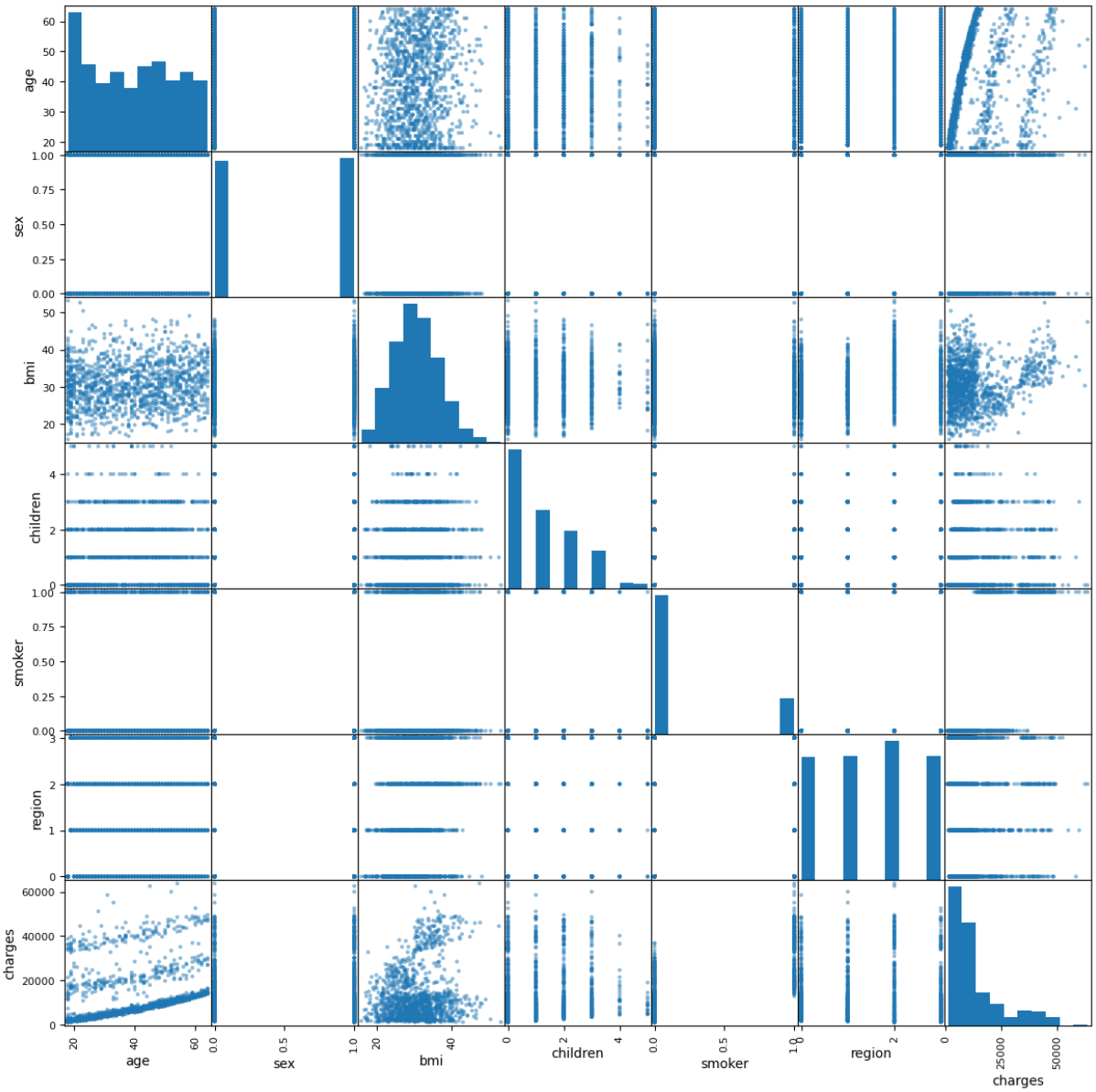
```python
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()
df3['sex'] = encoder.fit_transform(df3['sex'])
df3['smoker'] = encoder.fit_transform(df3['smoker'])
df3['region'] = encoder.fit_transform(df3['region'])
```

Certaines colonnes ne sont pas numériques, comme **sex**, **smoker**, ou **region**.
Le modèle linéaire ne peut pas traiter des chaînes de caractères : il faut donc les encoder.

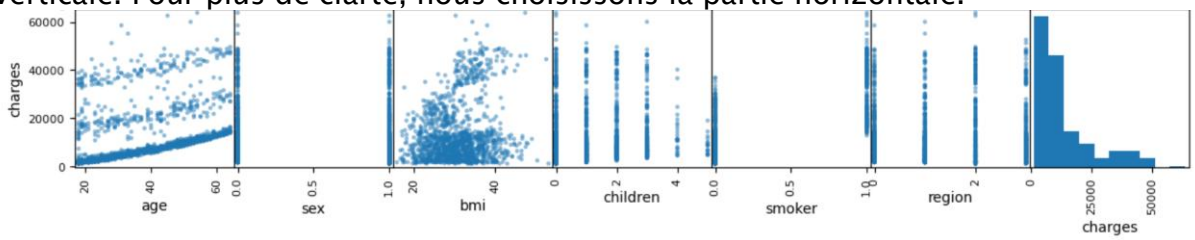
- *smoker* : yes → 1, no → 0
- *sex* : male → 1, female → 0
- *region* : 1, 2, 3, 4

Scatter matrix du dataset Insurance, pour avoir un aperçu global du dataset.

Scatter Matrix du dataset Insurance



La partie de la matrice que nous devons analyser est celle des variables en relation avec la charge. On peut prendre soit la partie horizontale, soit la partie verticale. Pour plus de clarté, nous choisissons la partie horizontale.



L'analyse montre que certaines variables ont une influence nettement plus marquée sur la charge. Le statut de fumeur (***smoker***) est le facteur le plus déterminant : les personnes qui fument (smoker = 1) présentent des charges beaucoup plus élevées que celles qui ne fument pas (smoker = 0). L'âge joue également un rôle important : plus l'âge augmente, plus la charge tend à croître. De même, l'indice de masse corporelle (*BMI*) montre une relation significative : lorsque le BMI augmente, les charges augmentent également, souvent de manière notable pour les valeurs élevées. En revanche, le sexe (homme = 1, femme = 0) ne montre pas de différence marquante sur la charge, tout comme le nombre d'enfants (*children*) et la région d'habitation (*region*), qui semblent exercer une influence faible ou négligeable sur les coûts. Ainsi, les facteurs les plus impactants restent principalement le tabagisme, l'âge et le BMI.

Partie 2 : Régression Linéaire Simple (Expérience /Salaire)

2.1 Séparation Train/Test

La séparation en ensembles *train* et *test* permet d'estimer la capacité d'un modèle à **généraliser** sur des données qu'il n'a pas vues pendant l'apprentissage. Si on entraînait et évaluait le modèle sur les mêmes données, on obtiendrait des performances optimistes (surapprentissage), et on ne saurait pas si le modèle marche sur de nouvelles observations.

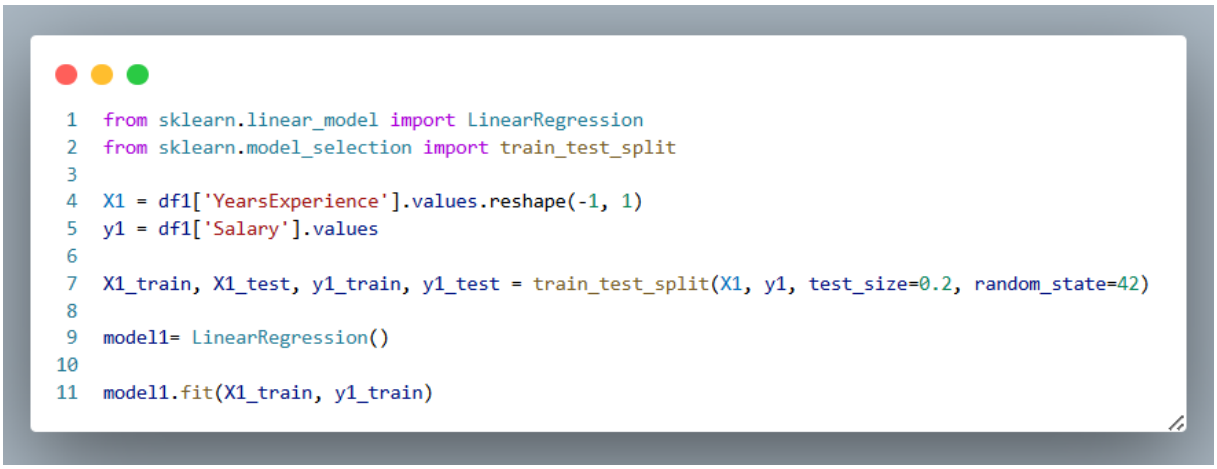
train et ***test*** :

- **Train** : sous-ensemble des données utilisé pour apprendre les paramètres du modèle.
- **Test** : sous-ensemble gardé de côté et utilisé uniquement pour évaluer la performance finale du modèle.

2.2 Entraînement du modèle :

On sépare les données en un ensemble d'entraînement et de test, puis crée un modèle de régression linéaire via ***LinearReeression***. Le modèle est ensuite entraîné pour apprendre la relation entre l'expérience et le salaire.

on utilise souvent un ratio *80/20* ou *70/30* (train/test). On fixe random_state pour rendre la séparation reproductible, ici j'ai choisi 20% pour le test, via *test_size= 0.2*



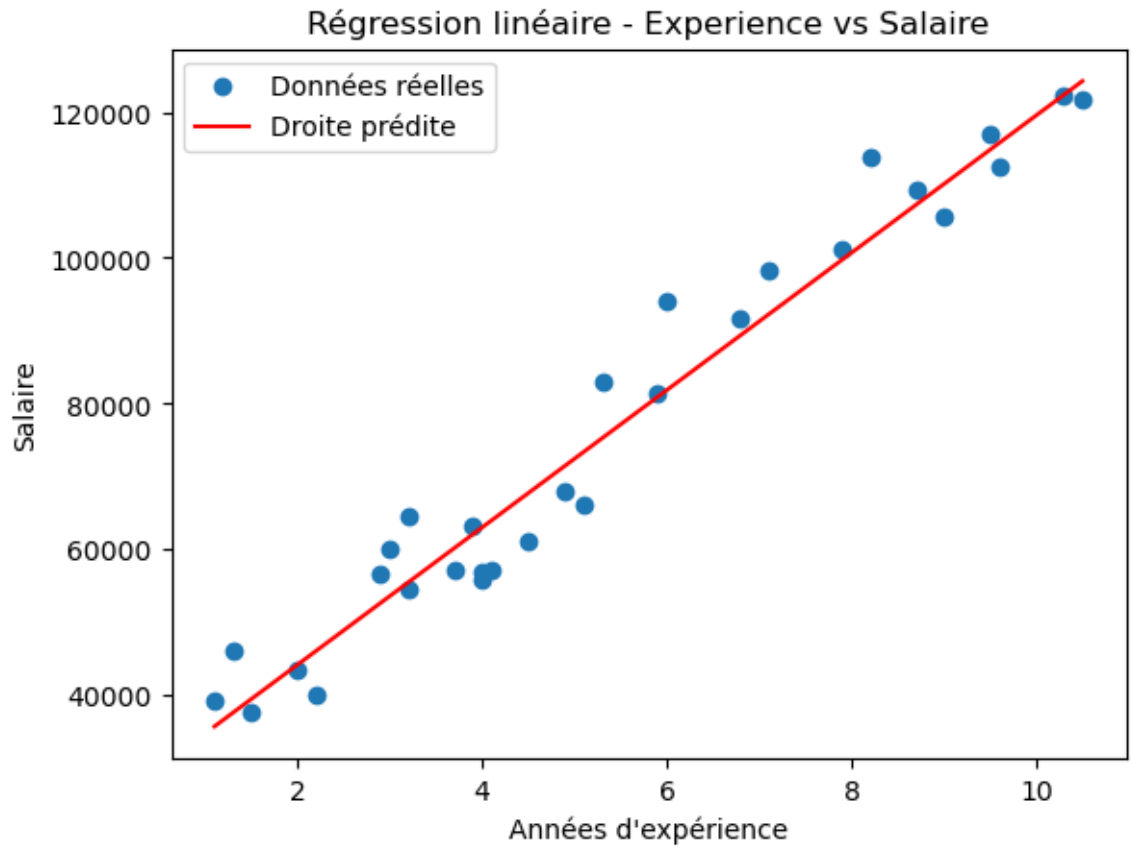
```python
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

X1 = df1['YearsExperience'].values.reshape(-1, 1)
y1 = df1['Salary'].values

X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, test_size=0.2, random_state=42)

model1= LinearRegression()

model1.fit(X1_train, y1_train)
```

2.3. Prédictions & Visualisation

On prédit sur l'ensemble de test pour mesurer la performance **réelle** du modèle : la comparaison y_test vs y_pred nous donne une estimation honnête de l'erreur sur des données invisibles lors de l'apprentissage. Si l'erreur sur le test est très supérieure à celle sur le train, c'est un signe de **surapprentissage**.

Droite de régression

Le traçage de la droite de régression sur le nuage de points permet de visualiser si le modèle capture bien la tendance générale.



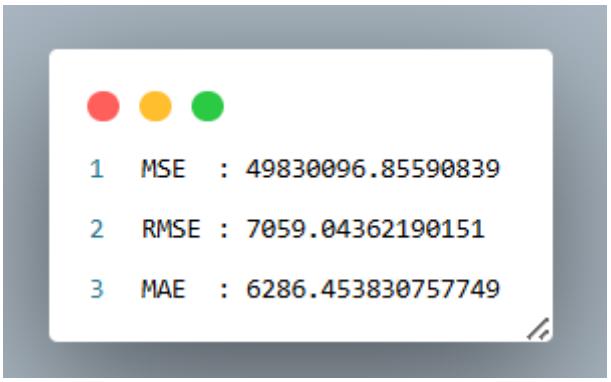
Interprétation visuelle

La droite suit bien la tendance des points, ce qui confirme la pertinence d'une relation linéaire. Avec la marge d'erreur est très faible.

2.4 Évaluation de la qualité du modèle :

Le modèle fonctionne presque correctement: Il prédit les salaires avec une erreur moyenne comprise entre **6200 et 7000 dh**.

Mais il est claire que la marge d'erreur n'est pas plus faible.



❖ MAE = 6286.45

Le modèle commet en moyenne une erreur d'environ **6286 dh** sur ses prédictions.Cette valeur reste acceptable puisque les salaires du dataset se situent généralement entre **35 000 et 120 000 dh**. De plus, avec un dataset très réduit (**30 lignes seulement**), il est normal qu'un modèle n'arrive pas à apprendre des relations très précises.

❖ RMSE = 7059.04

La racine de l'erreur quadratique moyenne indique une erreur standard d'environ **7000 dh**.
Elle est naturellement plus élevée que le MAE car :

- le **RMSE pénalise davantage les grandes erreurs**,
- ce qui suggère la présence de quelques prédictions nettement éloignées de la valeur réelle.

❖ MSE = 49,830,096

Cette valeur semble élevée, mais c'est normal puisque :

- des erreurs de **6000 à 8000 dh** deviennent logiquement des millions lorsqu'on les élève au carré.
  C'est également pour cette raison que le **RMSE est plus interprétable** que le MSE.

2.5 Conclusion

L'erreur n'est pas plus faible parce que :

- le dataset est **très petit** (30 observations),
- la régression linéaire impose une **relation strictement linéaire**,
- le salaire ne dépend pas uniquement de l'expérience (niveau d'étude, entreprise, pays, etc.),
- le dataset contient des **outliers**.

L'erreur représente environ **10 %** du salaire moyen, ce qui est raisonnable pour un dataset aussi limité.

Partie 3 : Régression Linéaire Multiple (Assurance)

3.1. Nettoyage + EDA

Dans cette partie, nous analysons le dataset *Insurance* afin de comprendre la structure des données et préparer la régression linéaire multiple

```
Résumé statistique :
                age          sex          bmi     children       smoker  \
count  1337.000000  1337.000000  1337.000000  1337.000000  1337.000000
mean     39.222139     0.504862    30.663452     1.095737     0.204936
std      14.044333     0.500163     6.100468     1.205571     0.403806
min      18.000000     0.000000    15.960000     0.000000     0.000000
25%      27.000000     0.000000    26.290000     0.000000     0.000000
50%      39.000000     1.000000    30.400000     1.000000     0.000000
75%      51.000000     1.000000    34.700000     2.000000     0.000000
max      64.000000     1.000000    53.130000     5.000000     1.000000

            region       charges
count  1337.000000   1337.000000
mean      1.516081  13279.121487
std       1.105208  12110.359656
min       0.000000   1121.873900
25%       1.000000   4746.344000
50%       2.000000   9386.161300
75%       2.000000  16657.717450
max       3.000000  63770.428010

dimensions :
(1337, 7)
```

```
Types des colones :

Index: 1337 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       1337 non-null   int64
 1   sex       1337 non-null   int64
 2   bmi       1337 non-null   float64
 3   children  1337 non-null   int64
 4   smoker    1337 non-null   int64
 5   region    1337 non-null   int64
 6   charges   1337 non-null   float64
dtypes: float64(2), int64(5)
memory usage: 83.6 KB
None

valeurs manquantes :
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64

lignes dupliquées :
0
```

- Matrice de corrélation

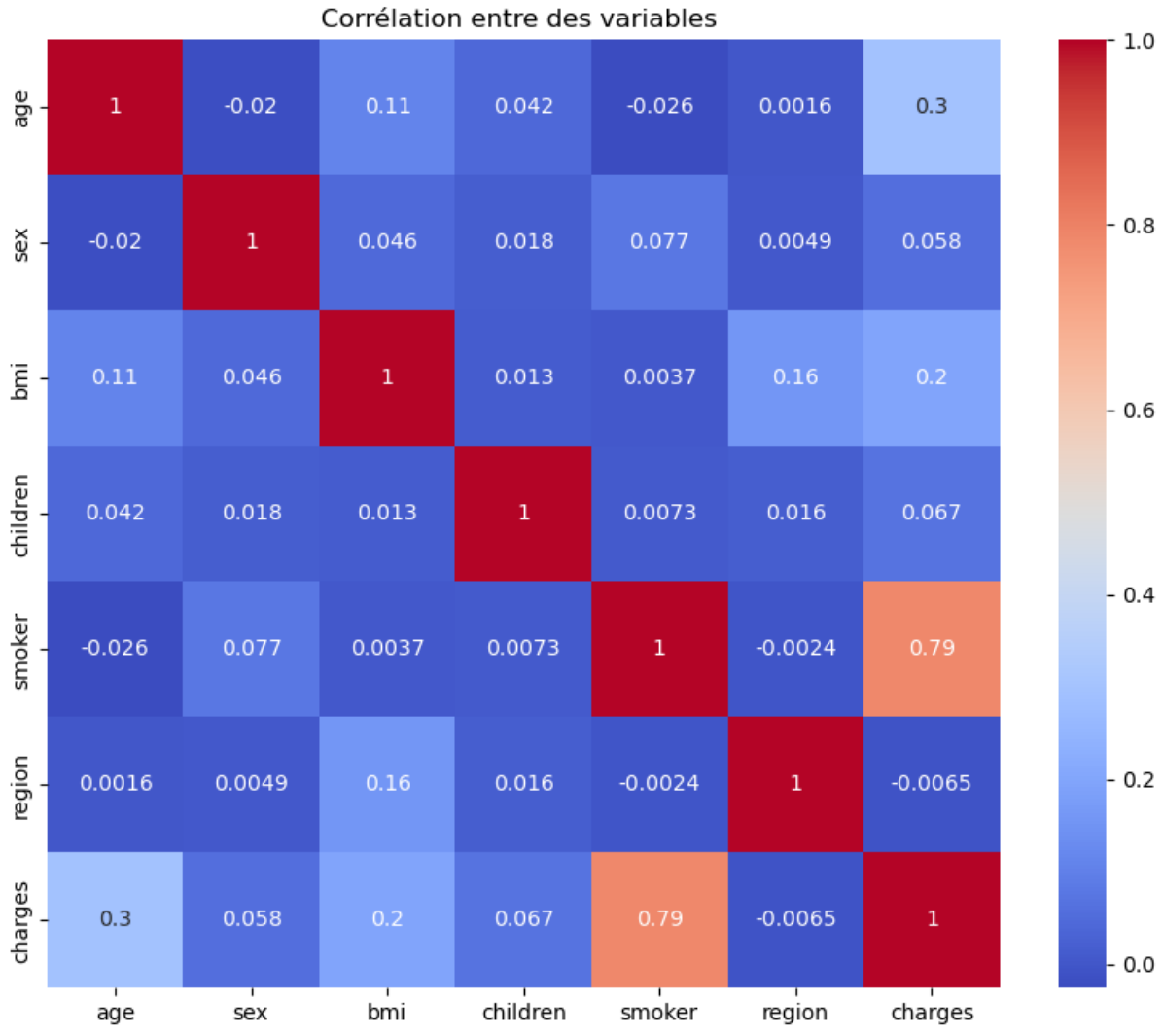
La **corrélation** permet d'identifier les variables qui ont la plus forte relation avec la target charges.
Dans notre dataset, on observe que :

- **smoker** est la variable la plus corrélée positivement avec les frais médicaux

- **bmi** montre une corrélation modérée
- **age** présente une corrélation positive importante

Ces trois caractéristiques ressortent fortement dans la matrice de corrélation.

```python
import seaborn as sns

plt.figure(figsize=(10, 8))
sns.heatmap(df3.corr(), annot=True, cmap='coolwarm')
plt.title("Corrélation entre des variables ")
plt.show()
```



•Conclusion EDA

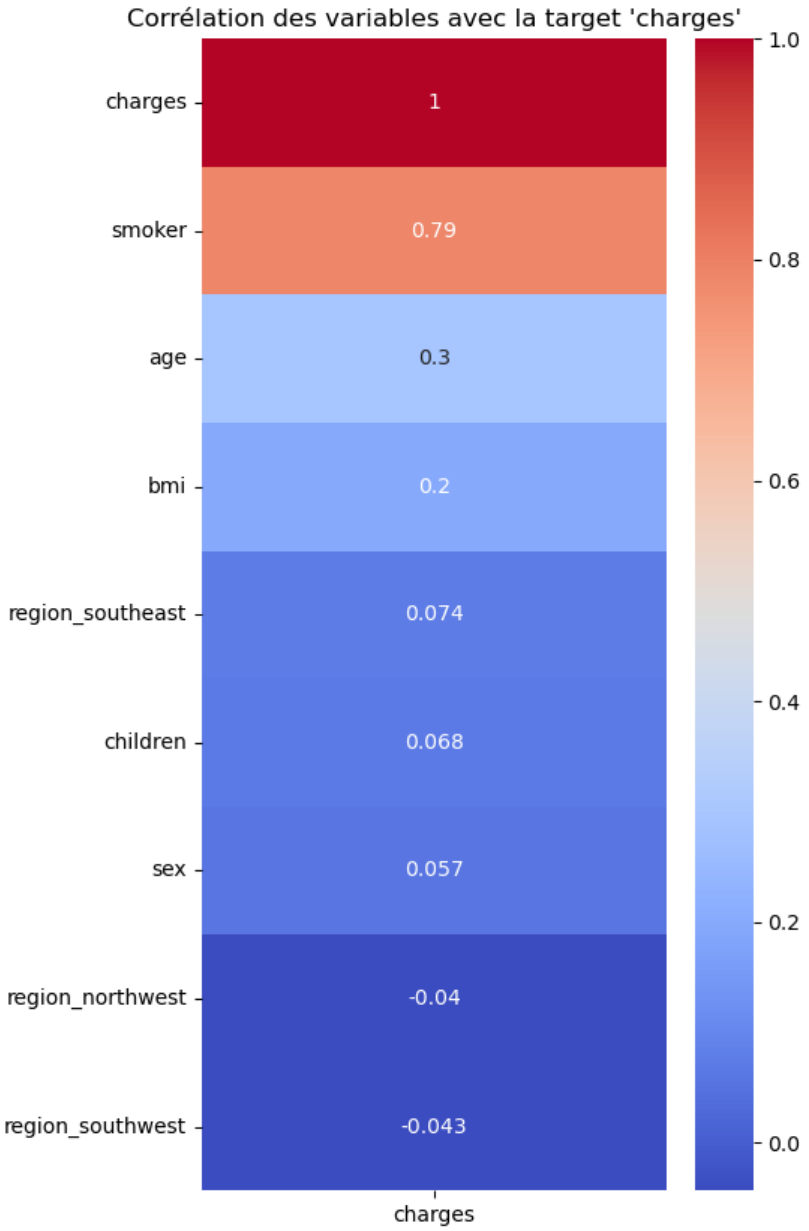
L'EDA montre clairement que certaines variables ont un impact dominant. Cela guidera la sélection des 3 features.

3.2. Sélection des 3 features :

La sélection des features est essentielle pour obtenir un modèle simple, interprétable et performant. J'ai utilisé trois techniques différentes pour faire une bonne sélection.

3.2.1 Corrélation des variables avec la ta et 'charges ' :

```python
import seaborn as sns

corr_charges = df3.corr()['charges'].sort_values(ascending=False)

print("Corrélation:")
print(corr_charges)

# Heatmap de 'charges'
plt.figure(figsize=(5,10))
sns.heatmap(corr_charges.to_frame(),
            annot=True,
            cmap="coolwarm")
plt.title("Corrélation des variables avec la target 'charges'")
plt.show()
```



Comme vu précédemment :

- **smoker** → corrélation très forte
- **age** → corrélation élevée
- **bmi** → corrélation significative

Les autres variables comme *children*, *sex et region* ont une faible influence.

Les valeurs exactes des corrélations :

```
Corrélation:
charges     1.000000
smoker      0.787234
age         0.298308
bmi         0.198401
children    0.067389
sex         0.058044
region     -0.006547
Name: charges, dtype: float64
```

3.2.2 Importance des features (RandomForestRegressor)

Pour valider le choix, nous utilisons un modèle de type Random Forest afin de mesurer l'importance relative de chaque variable.

```python
from sklearn.ensemble import RandomForestRegressor

model_rf = RandomForestRegressor()
model_rf.fit(df3[['age','sex','bmi','children','smoker', 'region']], df3['charges'])

importances = model_rf.feature_importances_
print("Importance des features")
for f, imp in zip(['age','sex','bmi','children','smoker', 'region'], importances):
    print(f"{f} : {imp}")
```

Le résultat confirme :

- **smoker** est la feature la plus importante
- **bmi** et **age** suivent juste après

```
Importance des features
age : 0.13044253833728903
sex : 0.005831893115729011
bmi : 0.2119079102794134
children : 0.020222954186407804
smoker : 0.6175890827829016
region : 0.014005621298259073
```

3.2.3 Logique métier :

D'un point de vue métier :

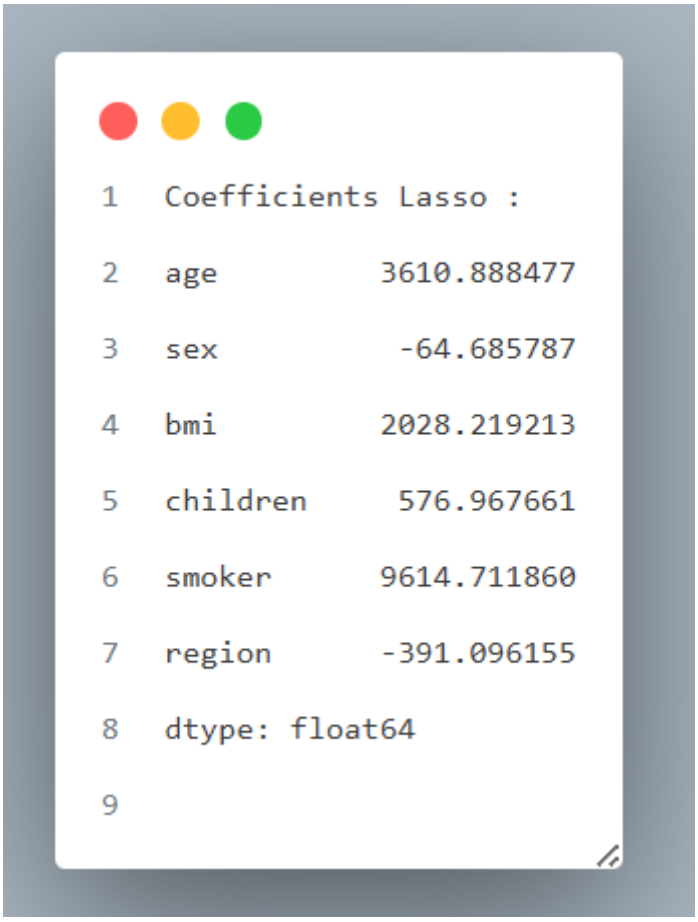
- Un **fumeur** paie naturellement beaucoup plus en assurance santé.
- L'**âge** influence fortement le risque et donc le coût.
- Le **BMI** reflète le niveau de surpoids, lié au risque médical.

3.2.4 Lasso Regression (L1 Regularization) :

Lasso applique une pénalisation L1 sur les coefficients du modèle.
Cela force certains coefficients à devenir très petits ou proches de zéro → sélection naturelle des variables importantes.

```python
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler

X = df3[['age','sex','bmi','children','smoker','region']]
y = df3['charges']

# Standardisation  pour Lasso
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Modèle Lasso
lasso = Lasso(alpha=0.01)
lasso.fit(X_scaled, y)

coef = pd.Series(lasso.coef_, index=X.columns)
print("Coefficients Lasso :")
print(coef)
```

Résultats obtenus :



```
Coefficients Lasso :
age          3610.888477
sex           -64.685787
bmi          2028.219213
children      576.967661
smoker       9614.711860
region       -391.096155
dtype: float64
```

Interprétation :

- **smoker** a de loin le coefficientplus élevé
  la variable la plus influente du dataset.
- **age** et **bmi** ont aussi des coefficients très élevés
  forts prédicteurs des charges.
- **sex** et **region** ont des coefficients très faibles
  un impact négligeable.

3.2.5 SelectKBest (F-test ou Mutual Information):

La méthode **SelectKBest utilisant le test F (f_regression)** sélectionne les variables qui ont la plus forte **relation linéaire** avec la variable cible `charges`.

```python
from sklearn.feature_selection import SelectKBest, f_regression

X = df3[['age','sex','bmi','children','smoker','region']]
y = df3['charges']

selector = SelectKBest(score_func=f_regression, k=3)
selector.fit(X, y)

scores = pd.Series(selector.scores_, index=X.columns)
print("Scores ANOVA :")
print(scores.sort_values(ascending=False))
```

Résultats obtenus :

```
Scores ANOVA :
smoker      2175.736863
age          130.402971
bmi           54.702715
children       6.090326
sex            4.513038
region         0.057217
dtype: float64
```

Interprétation

1. smoker **(score = 2175)**

Une valeur extrêmement élevée.
C'est de très loin la variable qui explique le plus les charges d'assurance.
L'effet du tabagisme sur les coûts médicaux est massivement important.

2. **age** (score = 130)

Score élevé et significatif.
Plus l'âge augmente, plus les dépenses médicales augmentent.
Relation linéaire forte.

3. bmi **(score = 54)**

Score modéré mais clair.
L'obésité est associée à de nombreux risques médicaux (cardiaques, diabète).
Influence importante mais moins que smoker et age.

3.2.4 Conclusion

Nous retenons logiquement les 3 variables les plus importantes : age, bmi, smoker

3.3. Normalisation / Standardisation

- **Pourquoi normaliser ?**

Les trois variables n'ont pas la même échelle :

- age ~ entre 18 et 65
- bmi ~ entre 15 et 50
- smoker ~ 0 ou 1

Une différence d'échelle peut déséquilibrer l'apprentissage et ralentir la convergence du modèle.
La normalisation permet :

- d'améliorer la stabilité du modèle
- parfois d'améliorer les performances
- de rendre les coefficients comparables

- **Technique choisie : Standardisation**

```python
X3_train_scaled = X3_train.copy()
X3_test_scaled = X3_test.copy()

X3_train_scaled[['age', 'bmi']] = scaler.fit_transform(X3_train[['age', 'bmi']])
X3_test_scaled[['age', 'bmi']] = scaler.transform(X3_test[['age', 'bmi']])
```

- **Pourquoi ce choix ?**

La standardisation est recommandée pour la régression linéaire car :

- elle gère bien les distributions non bornées
- elle rend les coefficients plus lisibles
- elle stabilise le calcul des moindres carrés

3.4. Entraînement & Prédiction

Nous appliquons un modèle de **régression linéaire multiple**.

```python
from sklearn.linear_model import LinearRegression
model3 = LinearRegression()
model3.fit(X3_train_scaled, y3_train)

print("\nCoefficients du modèle :")
for feature, coef in z)ip(['age','bmi','smoker'], model3.coef_):
    print(f"{feature} : {coef}")

print("Intercept :", model3.intercept_
```

De même, le modèle est entraîné sur l'ensemble d'apprentissage (80 % des données).
Puis, les prédictions sont effectuées sur l'ensemble de test.

```
1
2   Coefficients du modèle :
3   age : 3524.2627929030305
4   bmi : 1843.7537629767808
5   smoker : 23074.261734978376
```

3.5. Visualisation

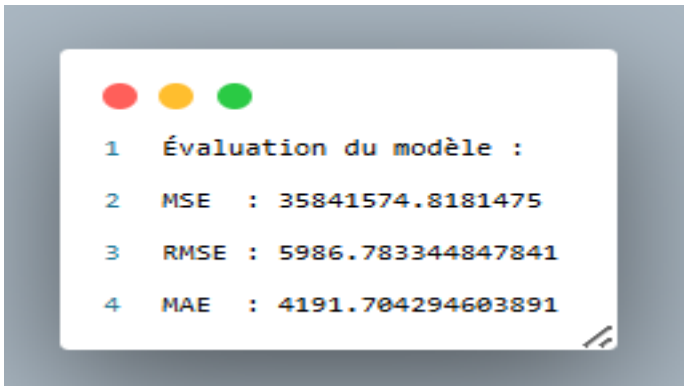
les points suivent la diagonale, cela signifie que les prédictions sont proches des valeurs réelles.
Dans notre cas, les points sont globalement bien alignés mais montrent une dispersion due à la nature complexe du dataset.

- **Résultats observés**

Les prédictions montrent que le modèle capture correctement la tendance générale du dataset, même s'il existe une certaine variabilité liée à des facteurs non inclus dans les 3 features choisies.

3.6. Évaluation

Nous utilisons les trois métriques classiques :



Interprétation des résultats

Dans notre cas :

❖ Mean Squared Error (MSE) : 35 841 574.81

- Une valeur élevée est normale dans un dataset où les charges d'assurance peuvent monter à **plus de 60 000**.
- Les grandes erreurs (cas particuliers : fumeurs + BMI élevé) ont augmenté la valeur du MSE.

❖ Root Mean Squared Error (RMSE) : 5 986.78

- Cela signifie qu'en moyenne ton modèle se trompe d'environ ± **6 000**.
- Le modèle généralise bien, mais il garde une erreur notable, logique car les dépenses médicales varient fortement entre individus.

❖ Mean Absolute Error (MAE) : 4 191.70

- Une erreur moyenne d'environ ± **4 200 $** est correcte pour un modèle simple linéaire sur ce dataset.
- Le modèle se trompe en moyenne de 4 200 $, ce qui reste raisonnable vu la variabilité du dataset.

3.7 Conclusion :

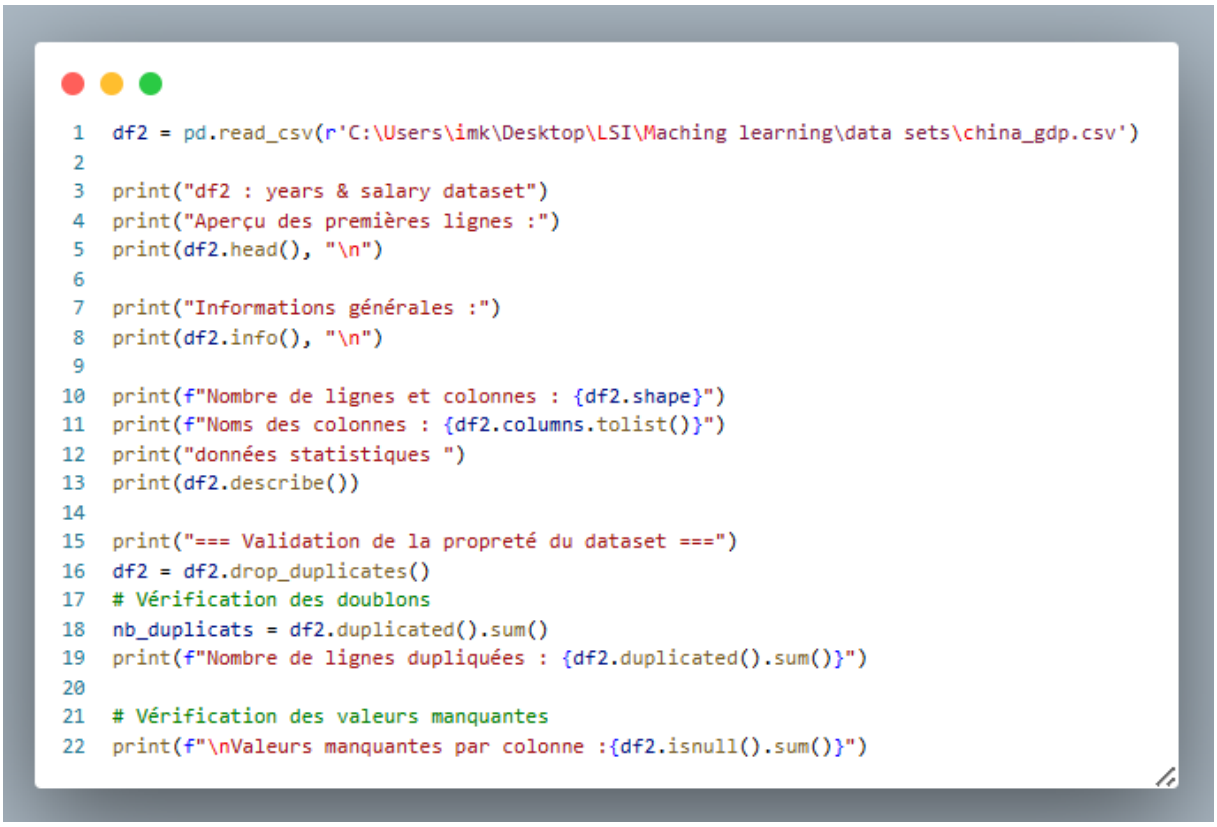
- Le modèle est **globalement cohérent**, mais :

❖ Il reste limité par la **forte variabilité des charges** (surtout les fumeurs et les personnes obèses),
❖ Les erreurs sont normales pour une régression linéaire simple avec seulement 3 features,

Partie 4 : Régression Linéaire vs Polynomiale (China GDP)

4.1 Préparation des données :

Le dataset China GDP contient l'évolution du PIB de la Chine sur plusieurs décennies.
Visuellement, la croissance du PIB ne suit **pas une droite**, mais plutôt une **courbe en forme de "S"**, typique des phénomènes économiques à long terme (croissance lente au début, accélération, puis stabilisation).



```python
df2 = pd.read_csv(r'C:\Users\imk\Desktop\LSI\Maching learning\data sets\china_gdp.csv')

print("df2 : years & salary dataset")
print("Aperçu des premières lignes :")
print(df2.head(), "\n")

print("Informations générales :")
print(df2.info(), "\n")

print(f"Nombre de lignes et colonnes : {df2.shape}")
print(f"Noms des colonnes : {df2.columns.tolist()}")
print("données statistiques ")
print(df2.describe())

print("=== Validation de la propreté du dataset ===")
df2 = df2.drop_duplicates()
# Vérification des doublons
nb_duplicats = df2.duplicated().sum()
print(f"Nombre de lignes dupliquées : {df2.duplicated().sum()}")

# Vérification des valeurs manquantes
print(f"\nValeurs manquantes par colonne :{df2.isnull().sum()}")
```

Aperçu sur la dataset et ces Informations générales :

```
df2 : years & salary dataset
Aperçu des premières lignes :
   Year         Value
0  1960  5.918412e+10
1  1961  4.955705e+10
2  1962  4.668518e+10
3  1963  5.009730e+10
4  1964  5.906225e+10

Informations générales :
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55 entries, 0 to 54
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Year    55 non-null     int64
 1   Value   55 non-null     float64
dtypes: float64(1), int64(1)
memory usage: 1012.0 bytes
None

Nombre de lignes et colonnes : (55, 2)
Noms des colonnes : ['Year', 'Value']
données statistiques
              Year         Value
count    55.00000  5.500000e+01
mean   1987.00000  1.437042e+12
std      16.02082  2.500085e+12
min    1960.00000  4.668518e+10
25%    1973.50000  1.395123e+11
50%    1987.00000  3.074796e+11
75%    2000.50000  1.268748e+12
max    2014.00000  1.035483e+13
=== Validation de la propreté du dataset ===
Nombre de lignes dupliquées : 0

Valeurs manquantes par colonne :Year     0
Value    0
dtype: int64
```

Interprétation du dataset *df2 : china_gdp*

Ce dataset contient 55 observations représentant l'évolution d'un montant économique (appelé *Value*) entre 1960 et 2014. Les données sont propres, sans doublons ni valeurs manquantes. On observe une forte croissance de la

variable *Value* au fil du temps : les valeurs passent d'environ $4{,}7\times10^{10}$ au début des années 1960 à un maximum supérieur à 1×10^{13} en 2014. Les

statistiques montrent une distribution très étalée, avec une moyenne très élevée due à l'augmentation progressive de la valeur au fil des décennies. Cette croissance suggère une tendance ascendante marquée, probablement liée à un phénomène économique cumulatif (par exemple : PIB, salaires agrégés, dépenses totales, etc.). L'ensemble du dataset est donc cohérent et prêt à être utilisé pour une analyse temporelle ou pour modéliser une tendance à long terme.

4.2 Modèles entraînés

- **Modèle 1 & 2 : Régression linéaire simple et polynomiale (degré 4)**

L'idée est de vérifier si une relation linéaire est suffisante pour expliquer l'évolution du GDP.
Comme le PIB augmente de manière très rapide sur une courte période, une simple droite ne peut pas suivre la tendance.

Ce modèle permet de représenter des courbes complexes et s'adapte beaucoup mieux à des données non linéaires comme le PIB.

Après transformation, nous entraînons un nouveau modèle de régression linéaire sur ces nouvelles features.

```
X2 = df2[['Year']]
y2 = df2['Value']

X2_train, X2_test, y2_train, y2_test = train_test_split(
    X2, y2, test_size=0.2, random_state=42
)

# model lin
lin_model = LinearRegression()
lin_model.fit(X2_train, y2_train)

# model poly
poly = PolynomialFeatures(degree=4)
X2_poly_train = poly.fit_transform(X2_train)
X2_poly_test = poly.transform(X2_test)

poly_model = LinearRegression()
poly_model.fit(X2_poly_train, y2_train)
```

5.3. Résultats et visualisation

Lorsque nous traçons les deux modèles :

- **Résultat du modèle linéaire**

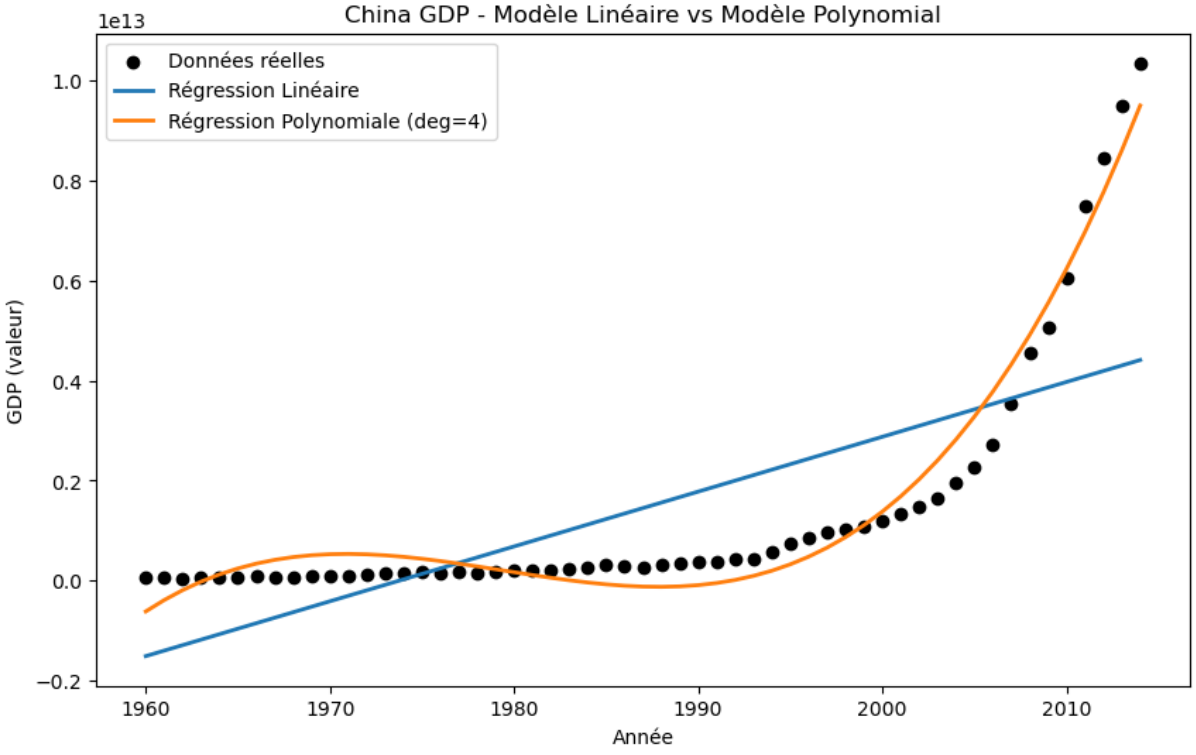
  - La droite obtenue est **presque plate**.
  - Elle ne suit pas du tout la forme réelle du GDP.
  - Elle sous-estime fortement les valeurs récentes du PIB.

Cela confirme que le modèle linéaire est **inadapté** à des données aussi non linéaires.

- **Résultat du modèle polynomial**

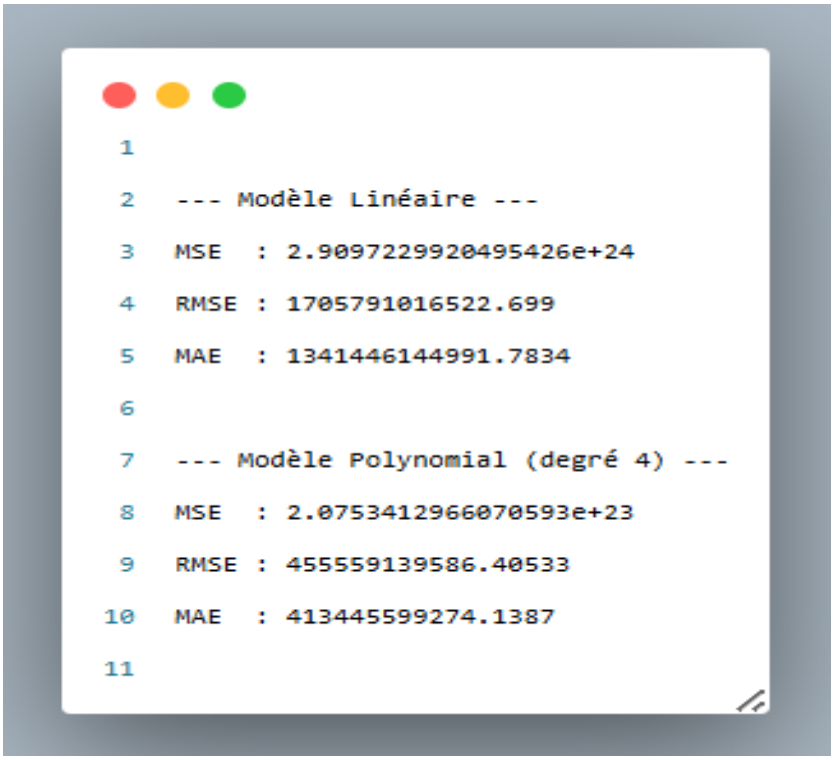
  - La courbe polynomiale suit beaucoup mieux la tendance réelle.
  - Elle reproduit la forme en "S" visible dans les données.
  - Les prédictions sont cohérentes sur l'ensemble des années.

Conclusion visuelle :



5.4. Évaluation des deux modèles

Pour comparer objectivement les performances, nous utilisons :



```
1
2   --- Modèle Linéaire ---
3   MSE  : 2.9097229920495426e+24
4   RMSE : 1705791016522.699
5   MAE  : 1341446144991.7834
6
7   --- Modèle Polynomial (degré 4) ---
8   MSE  : 2.0753412966070593e+23
9   RMSE : 455559139586.40533
10  MAE  : 413445599274.1387
11
```

Résultats obtenus :

--- Modèle Linéaire ---

- MSE très élevé
- RMSE énorme
- MAE très grand

--- Modèle Polynomial ---

- MSE **beaucoup plus faible**
- RMSE plus raisonnable
- MAE nettement inférieur au modèle linéaire

Interprétation des résultats

❖ **Modèle Linéaire**

MSE : 2.90×10^{24} – RMSE : 1.70×10^{12} – MAE : 1.34×10^{12}
Le modèle linéaire échoue totalement : il ne suit pas la forme réelle du PIB chinois, qui est clairement non linéaire (croissance logistique). La droite passe loin des points, d'où des erreurs gigantesques.

❖ Modèle Polynomial (degré 4)

MSE : 2.07×10^{23} – RMSE : 4.55×10^{11} – MAE : 4.13×10^{11}
Les erreurs restent grandes, mais nettement plus faibles que celles du modèle linéaire. Le polynôme reproduit mieux la courbure du PIB, ce qui en fait un modèle beaucoup plus adapté.

Conclusion

Le modèle linéaire est inadapté car il ne peut pas représenter une croissance non linéaire.
Le modèle polynomial de degré 4 suit beaucoup mieux la tendance réelle, bien qu'il conserve de grandes erreurs à cause de l'échelle immense du PIB (ordre de 10^{12}).

- **Linéaire** : très mauvais → forme incompatible.
- **Polynomial 4** : bien meilleur → capture la courbe réelle du PIB.

6. Conclusion générale

Dans ce TP, nous avons appliqué de manière progressive et pratique les principaux concepts liés à la **régression linéaire simple**, **régression linéaire multiple** et **régression polynomiale**, et atelcier a permis de comprendre et d'appliquer les principales techniques de régression en Machine Learning, en commençant par l'exploration et la visualisation de plusieurs jeux de données réels. Les analyses graphiques et statistiques ont montré comment les variables évoluent, comment elles interagissent entre elles et quelles sont celles qui influencent le plus la variable cible, notamment dans les cas Expérience–Salaire, Assurance et GDP de la Chine. Ces étapes d'EDA ont servi de base pour choisir les features pertinentes et justifier leur importance dans la construction des modèles.

Ensuite, les différentes formes de régression simple, multiple et polynomiale ont été mises en œuvre grâce à l'API de sklearn, avec entraînement, prédiction, visualisation des résultats et évaluation via MSE, RMSE et MAE. Les comparaisons obtenues ont montré la capacité de chaque modèle à expliquer les données et à prédire de nouvelles valeurs, tout en illustrant les limites de la régression linéaire pour certains phénomènes non linéaires, comme le GDP. Globalement, l'atelier a permis de renforcer la compréhension pratique de la régression et de la préparation de données dans un contexte de Machine Learning.

Fin.