

CORRECTION TP 2 : Prédiction Médicale

```
# -----
```

1. IMPORTATION DES BIBLIOTHÈQUES

```
# -----
```

```
import numpy as np          # Pour les calculs numériques
import pandas as pd         # Pour la manipulation de données tabulaires
import matplotlib.pyplot as plt # Pour les visualisations graphiques

from sklearn.model_selection import train_test_split # Pour diviser les données en ensembles
d'entraînement et de test

from sklearn.linear_model import LogisticRegression # Pour la régression logistique (modèle
de classification)

from sklearn.metrics import accuracy_score          # Pour évaluer la performance du modèle
```

```
# -----
```

2. CRÉATION DES DONNÉES SIMULÉES

```
# -----
```

```
# Générer un jeu de données simulé avec des informations médicales fictives
```

```
data = pd.DataFrame({
    'age': np.random.randint(20, 70, 500), # Âge des patients entre 20 et 70 ans
    'cholesterol': np.random.normal(200, 30, 500), # Niveau de cholestérol autour de 200 avec
écart-type de 30
    'blood_pressure': np.random.normal(120, 15, 500), # Tension artérielle autour de 120 avec
écart-type de 15
    'has_disease': np.random.randint(0, 2, 500) # Cible binaire : 0 (pas de maladie), 1 (malade)
})
```

```
# Afficher un aperçu des premières lignes du DataFrame
```

```
print(data.head())
```

```

# -----

# 3. ANALYSE EXPLORATOIRE

# -----

# Afficher les corrélations entre les variables

print(data.corr(numeric_only=True))


# Visualisation de la relation entre l'âge et le cholestérol, colorée selon la présence de la
maladie

plt.scatter(data['age'], data['cholesterol'], c=data['has_disease'], cmap='coolwarm')

plt.xlabel('Âge')

plt.ylabel('Cholestérol')

plt.title('Relation entre Âge, Cholestérol et Présence de Maladie')

plt.show()


# -----

# 4. PRÉPARATION DES DONNÉES

# -----

# Séparation des variables explicatives (features) et de la cible (target)

X = data[['age', 'cholesterol', 'blood_pressure']] # Variables explicatives

y = data['has_disease'] # Cible : présence de maladie


# Diviser les données en 80% d'entraînement et 20% de test

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

```