TP4_1 : CLASSIFICATION (REGRESSION LIGISTIQUE)

Données : 'Social_Network_Ads.csv'

A. Régression Linéaire Simple

#################### DATA PREPROCESSING ####################

 # Regression Logistique

 # Pour notre modele de regression logistique on n'utilisera pas les colonnes

 # Importer les librairies

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

 # Importer le dataset

```
dataset = pd.read_csv('Social_Network_Ads.csv')
```

 # extraire la matrice des variables independante et le vecteur de variable dependante

```
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, -1].values
```

 # Diviser le dataset entre le Training set et le Test set

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25,
random_state = 0)
```

 # Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

 # Construction du modèle

```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
```

 # Faire de nouvelles prédictions

```
y_pred = classifier.predict(X_test)
```

 # Matrice de confusion

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

ar = (65 + 24)/(100)
er = (8+3)/(100)
```

 # Visualiser les résultats

```
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:,
0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:,
1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),
X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.4, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Résultats du Training set')
plt.xlabel('Age')
plt.ylabel('Salaire Estimé')
plt.legend()
plt.show()
```

 # Interpreter le resultat obtenu

 # % au resultat

 # % a la frontiere de rediction