In [1]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
```

In [29]:

```
data = pd.read_csv('pima-indians-diabetes.csv')
df = pd.DataFrame(data=data.values, columns=['Pregnancies', 'Glucose', 'Blood_Pr
df.head()
```

Out[29]:

	Pregnancies	Glucose	Blood_Pressure	Skin_Thickness	Insulin	Bmi	Diabetes_Pedigre
0	1.0	85.0	66.0	29.0	0.0	26.6	0.35
1	8.0	183.0	64.0	0.0	0.0	23.3	0.67
2	1.0	89.0	66.0	23.0	94.0	28.1	0.16
3	0.0	137.0	40.0	35.0	168.0	43.1	2.28
4	5.0	116.0	74.0	0.0	0.0	25.6	0.20

In [30]:

```
def clean_outliers(col) :

    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)

    IqR = Q3 - Q1

    lower_bound = Q1 - 1.5 * IqR
    upper_bound = Q3 + 1.5 * IqR

    return df[col].clip(lower=lower_bound, upper=upper_bound)

x = df.columns

for i in x :
    df[i] = clean_outliers(i)
```

In [14]:

```
df.shape
```

Out[14]:

```
(767, 9)
```

In [31]:

```
y = df['Prediction'].values.reshape(-1, 1)

X = df.drop(columns='Prediction')
X = np.hstack((np.ones((X.shape[0],1)),X))
sc = StandardScaler()
X = sc.fit_transform(X)
```

In [32]:

```
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [34]:

```
def segmoind(z) :
    return (1 / (1 + np.exp(-z)))
```

```


def model(x, theta) :
    fc = x.dot(theta)
    return segmoind(fc)


def fonction_obj(X,y, theta):
    m = len(y)
    y_pred = model(X,theta)
    cost = -(1/m)*np.sum(y*np.log(y_pred)+(1-y)*np.log(1-y_pred))
    return cost

def gradient_opt(X,y, theta, lr=0.01, iter=20000):
    m = len(y)
    cost_list = []
    for i in range(iter):
        y_pred = model(X, theta)
        gradient = (1/m) * X.T.dot(y_pred - y)
        cost = fonction_obj(X,y, theta)
        theta -= lr * gradient
        cost_list.append(cost)


    return theta, cost_list
```

In [39]:

```
theta_initial = np.random.randn(x_train.shape[1],1)
theta_opti, cost_list = gradient_opt(x_train,y_train, theta_initial, lr=0.1, ite
```

In [59]:

```
ypredection = model(x_test, theta_opti)
y0 = ypredection[ypredection<0.5]
len(y0)
```

Out[59]:

```
83
```

In [ ]:

```
y_pred = model(x_test,theta_opti)
y_pred1 = (y_pred>=0.5)
performance = accuracy_score(y_test,y_pred1)
print(f"Sccuracy score : {performance*100:.4f}")
```

```
Sccuracy score : 76.6234
```

In [41]:

```
mse = fonction_obj(x_test, y_test, theta_opti)
mse
```

Out[41]:

```
np.float64(0.4989150045780876)
```

In [ ]:

In [ ]:

In [ ]:

```
import pickle

# Exemple : un dictionnaire Python


# Enregistrer dans un fichier .pkl
file_name = 'indians_diabet_logistic_regression_model.pkl'
with open(file_name, 'wb') as file:
    pickle.dump(theta_opti, file)


print("Objet enregistré avec succès !")
```

```
Objet enregistré avec succès !
```

In [ ]:

```

```