pima_logistic

November 29, 2025

[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
```

[2]:
```
data = pd.read_csv('pima-indians-diabetes.csv')
df = pd.DataFrame(data=data.values, columns=['Pregnancies', 'Glucose',␣
 ↪'Blood_Pressure',
                                              'Skin_Thickness', 'Insulin',␣
 ↪'Bmi', 'Diabetes_Pedigree', 'Age', 'Prediction'])
```

[5]:
```
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)
```

[6]:
```
y = df['Prediction'].values.reshape(-1, 1)

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

[7]:
```
sc = StandardScaler()
X = sc.fit_transform(X)
```

[8]:
```
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
```

[9]:
```
theta_initial = np.random.randn(X.shape[1],1)
theta_opti, cost_list = gradient_opt(X,y, theta_initial, lr=0.1, iter=25000)
```

[10]:
```
import pickle

V = {
    'theta': theta_opti,
    'ss': sc
}

# Enregistrer dans un fichier .pkl
file_name = 'classification 2.pkl'
with open(file_name, 'wb') as file:
    pickle.dump(V, file)


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

```
Objet enregistré avec succès !
```