**Bài 5: Supervised Learning: Classification phần 2 - Lập trình AI bằng Python**

**4. Logistic Regression**

Về cơ bản, mô hình hồi quy logistic là một trong những thành viên của họ thuật toán phân loại có giám sát. Hồi quy logistic đo lường mối quan hệ giữa các biến phụ thuộc và các biến độc lập bằng cách ước tính xác suất bằng cách sử dụng một hàm logistic.

Ở đây, nếu chúng ta nói về các biến phụ thuộc và độc lập thì biến phụ thuộc là biến lớp mục tiêu mà chúng ta sẽ dự đoán và ở mặt khác, các biến độc lập là các tính năng chúng ta sẽ sử dụng để dự đoán lớp mục tiêu.

Trong hồi quy logistic, ước tính xác suất có nghĩa là dự đoán khả năng xảy ra sự kiện. Ví dụ, chủ cửa hàng muốn dự đoán khách hàng vào cửa hàng có mua sắm thả ga (chẳng hạn) hay không. Sẽ có nhiều đặc điểm về khách hàng - giới tính, độ tuổi, v.v. sẽ được nhân viên cửa hàng quan sát để dự đoán khả năng xảy ra, tức là có mua trạm chơi hay không. Hàm logistic là đường cong sigmoid được sử dụng để xây dựng hàm với các tham số khác nhau.

**Điều kiện tiên quyết:**

Trước khi xây dựng bộ phân loại bằng cách sử dụng hồi quy logistic, chúng ta cần cài đặt gói Tkinter trên hệ thống của mình. Nó có thể được cài đặt từ https://docs.python.org/2/library/tkinter.html.

Bây giờ, với sự trợ giúp của đoạn mã dưới đây, chúng ta có thể tạo bộ phân loại bằng cách sử dụng hồi quy logistic -

Đầu tiên, chúng tôi sẽ nhập một số gói -

**import** numpy as np

from sklearn **import** linear\_model

**import** matplotlib.pyplot as plt

Bây giờ, chúng ta cần xác định dữ liệu mẫu có thể được thực hiện như sau:

X = np.array([[2, 4.8], [2.9, 4.7], [2.5, 5], [3.2, 5.5], [6, 5], [7.6, 4],

[3.2, 0.9], [2.9, 1.9],[2.4, 3.5], [0.5, 3.4], [1, 4], [0.9, 5.9]])

y = np.array([0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 3])

Tiếp theo, chúng ta cần tạo bộ phân loại hồi quy logistic, có thể được thực hiện như sau:

Classifier\_LR = linear\_model.LogisticRegression(solver = 'liblinear', C = 75)

Cuối cùng nhưng không kém phần quan trọng, chúng ta cần đào tạo classifier này  :

**Classifier\_LR**.fit(X, y)

Bây giờ, làm thế nào chúng ta có thể hình dung đầu ra? Nó có thể được thực hiện bằng cách tạo một hàm có tên Logistic\_visualize () -

Def Logistic\_visualize(Classifier\_LR, X, y):

min\_x, max\_x = X[:, 0].min() - 1.0, X[:, 0].max() + 1.0

min\_y, max\_y = X[:, 1].min() - 1.0, X[:, 1].max() + 1.0

Trong dòng trên, chúng tôi đã xác định các giá trị tối thiểu và tối đa X và Y được sử dụng trong lưới lưới. Ngoài ra, chúng tôi sẽ xác định kích thước bước để vẽ lưới ô vuông.

mesh\_step\_size = 0.02

 xác định các giá trị X và Y như sau:

x\_vals, y\_vals = np.meshgrid(np.arange(min\_x, max\_x, mesh\_step\_size),

np.arange(min\_y, max\_y, mesh\_step\_size))

output = classifier.predict(np.c\_[x\_vals.ravel(), y\_vals.ravel()])

output = output.reshape(x\_vals.shape)

plt.figure()

plt.pcolormesh(x\_vals, y\_vals, output, cmap = plt.cm.gray)

plt.scatter(X[:, 0], X[:, 1], c = y, s = 75, edgecolors = 'black',

linewidth=1, cmap = plt.cm.Paired)

Đoạn code sau sẽ xác định ranh giới của lô đất

**plt**.xlim(**x\_vals**.min(), **x\_vals**.max())

**plt**.ylim(**y\_vals**.min(), **y\_vals**.max())

**plt**.xticks((**np**.arange(**int**(**X**[:, 0].min() **-** 1), **int**(**X**[:, 0].max() + 1), 1.0)))

**plt**.yticks((**np**.arange(**int**(**X**[:, 1].min() **-** 1), **int**(**X**[:, 1].max() + 1), 1.0)))

**plt**.show()

Bây giờ, sau khi chạy mã, chúng ta sẽ nhận được kết quả sau, bộ phân loại hồi quy logistic -

