https://www.tutorialspoint.com/artificial\_intelligence\_with\_python/index.htm

Logistic Regression

It is one of the members of supervised classification algorithm family.

AGRUPARA EN 4 GRUPOS LA DATA, ESTO EJEMPLO ES PARA VER MEJOR LOS 4 GRUPOS

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

Logistic regression measures the relationship between dependent variables and independent variables by estimating the probabilities using a logistic function.

Here, if we talk about dependent and independent variables then dependent variable is the target class variable we are going to predict and on the other side the independent variables are the features we are going to use to predict the target class.

In logistic regression, estimating the probabilities means to predict the likelihood occurrence of the event.

For example, the shop owner would like to predict the customer who entered into the shop will buy the play station (for example) or not.

There would be many features of customer − gender, age, etc. which would be observed by the shop keeper to predict the likelihood occurrence, i.e., buying a play station or not.

The logistic function is the sigmoid curve that is used to build the function with various parameters.

Before building the classifier using logistic regression, we need to install the Tkinter package on our system. <https://docs.python.org/2/library/tkinter.html>.

import numpy as np

from sklearn import linear\_model

import matplotlib.pyplot as plt

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]])

# 4 GRUPOS

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

#logistic regression classifier

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

#train classifier

Classifier\_LR.fit(X, y)

To visualize the data, we defined the minimum and maximum values X and Y to be used in mesh grid. In addition, we will define the step size for plotting the mesh grid.

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

mesh\_step\_size = 0.02

Let us define the mesh grid of X and Y values as follows −

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

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

With the help of following code, we can run the classifier on the mesh grid −

output = Classifier\_LR.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)

The following line of code will specify the boundaries of the plot

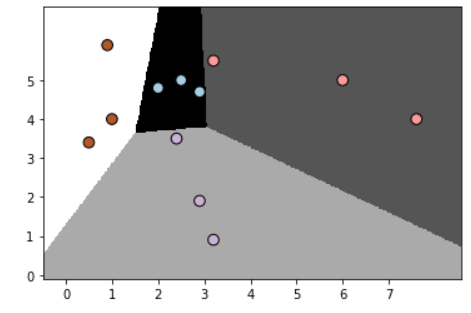
plt.xlim(x\_vals.min(), x\_vals.max())

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()



AGRUPO EN 4 GRUPOS LA DATA, ESTO SIRVE PARA VERLA MEJOR

# 4 GRUPOS

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