**REPORTE DE PRÁCTICA**

**IDENTIFICACIÓN DE LA PRÁCTICA**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Práctica** | **3** | **Nombre de la práctica** | | **Regresión logística** |
| **Fecha** | **24/03/2025** | **Nombre del profesor** | | **Alma Nayeli Rodríguez Vázquez** |
| **Nombre del estudiante** | | | **Jesús Alberto Aréchiga Carrillo** | |

**OBJETIVO**

|  |
| --- |
| El objetivo de esta práctica consiste en implementar el método de regresión logística para clasificación binaria. |

**PROCEDIMIENTO**

|  |
| --- |
| Realiza la implementación siguiendo estas instrucciones. |
| Implementa el método de regresión logística en Python. Para ello, considera los siguientes requerimientos:   * Utiliza el set de datos del archivo “dataset\_RegresionLogística1.csv”. * Utiliza los siguientes valores para los parámetros iniciales:   a=vector de ceros, beta=5, iteraciones=500   * Reporta el errorJ y el valor final del vector a. Además, reporta el valor de h y la clasificación estimada para los siguientes datos de prueba:   Dato de prueba 1: x1=34.6237, x2=78.0247  Dato de prueba 2: x1=60.1826, x2=86.3086  Dato de prueba 3: x1=82.2267, x2=42.7199   * Comprueba tus resultados con los siguientes:   Dato de prueba 1: x1=34.6237, x2=78.0247. Clase correcta y=0.  Probabilidad de pertenecer a la clase 1: h= 0.0910. Predicción: clase 0  Dato de prueba 2: x1=60.1826, x2=86.3086. Clase correcta y=1.  Probabilidad de pertenecer a la clase 1: h= 0.9904. Predicción: clase 1  Dato de prueba 3: x1=82.2267, x2=42.7199. Clase correcta y=0.  Probabilidad de pertenecer a la clase 1: h= 0.5994. Predicción: clase 1 |

**IMPLEMENTACIÓN**

|  |
| --- |
| Agrega el código de tu implementación aquí. |
| import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  data = pd.read\_csv('dataset\_RegresionLogistica1.csv')  data.head()  # Separar los datos en X y Y  m,n = data.shape  array = data.to\_numpy()  X = array[:,0:n-1]  Y = array[:,n-1]  # Dibujar los datos para visualizarlos (grafico de dispersion)  class0 = np.argwhere(Y==0)  class1 = np.argwhere(Y==1)  class0 = class0[:, 0]  plt.plot(X[class0, 0], X[class0, 1], 'dc', label = 'Reprobado: clase 0')  plt.plot(X[class1, 0], X[class1, 1], 'oy', label = 'Aprobado: clase 1')  plt.xlabel('x1: Examen parcial 1')  plt.ylabel('x2: Examen parcial 2')  plt.legend()  plt.show()  # Normalizar datos  mean = X.mean(axis = 0)  std = X.std(axis = 0, ddof = 1)  X = (X - mean)/std  X = np.concatenate((np.ones((m, 1)), X), axis = 1)  # Inicializar parametros  a = np.zeros(n)  beta = 5  iteracionesMaximas = 500  # Entrenamiento  h = np.zeros(m)  J\_vector = np.zeros(m)  J\_hist = np.zeros(iteracionesMaximas)  for iter in range(iteracionesMaximas):    for i in range(m):      z = np.dot(a, X[i,:])      h[i] = 1/(1 + np.exp(-z))      J\_vector[i] = -Y[i] \* np.log(h[i]) - (1-Y[i]) \* np.log(1-h[i])    for j in range(n):      a[j] = a[j] - beta \* ((1 / m) \* np.sum((h-Y) \* X[:, j]))    J = (1/m) \* np.sum(J\_vector)    J\_hist[iter] = J  plt.plot(J\_hist)  plt.xlabel('Iteraciones')  plt.ylabel('Funcion de costo J')  plt.show()  # Imprimir los resultados  print('J: ', J)  print('a: ', a)  #Dibujar la clasificación final  class0 = np.argwhere(Y==0)  class1 = np.argwhere(Y==1)  class0 = class0[:, 0]  plt.plot(X[class0, 1], X[class0, 2], 'dc', label = 'Reprobado: clase 0')  plt.plot(X[class1, 1], X[class1, 2], 'oy', label = 'Aprobado: clase 1')  x1 = np.linspace(-2, 1.5, 2)  x2 = -(a[0] + a[1] \* x1)/a[2]  plt.plot(x1, x2, 'r', label = 'Regresion Logistica')  plt.xlabel('x1: Examen parcial 1')  plt.ylabel('x2: Examen parcial 2')  plt.legend()  plt.show()  # Dato de prueba 1  alumno = np.array([34.6237, 78.0247])  alumno = (alumno - mean)/std  alumno = np.concatenate((np.ones((1)), alumno))  z = np.dot(a, alumno)  h = 1/(1 + np.exp(-z))  print('Probabilidad de aprobar:', h)  if h >= 0.5:    print("Pertenece a la clase 1")  else:    print("Pertenece a la clase 0")  # Dato de prueba 2  alumno = np.array([60.1826, 86.3086])  alumno = (alumno - mean)/std  alumno = np.concatenate((np.ones((1)), alumno))  z = np.dot(a, alumno)  h = 1/(1 + np.exp(-z))  print('Probabilidad de aprobar:', h)  if h >= 0.5:    print("Pertenece a la clase 1")  else:    print("Pertenece a la clase 0")  # Dato de prueba 3  alumno = np.array([82.2267, 42.7199])  alumno = (alumno - mean)/std  alumno = np.concatenate((np.ones((1)), alumno))  z = np.dot(a, alumno)  h = 1/(1 + np.exp(-z))  print('Probabilidad de aprobar:', h)  if h >= 0.5:    print("Pertenece a la clase 1")  else:    print("Pertenece a la clase 0") |

**RESULTADOS**

Agrega la(s) imagen(es) con los resultados obtenidos en los espacios indicados.

|  |  |
| --- | --- |
| Gráfica de los datos con la línea de clasificación | Gráfica de convergencia |
|  |  |
| Resultados donde se despliegue el valor de J, el valor final del vector a, los datos de prueba, su clase correcta, el valor de h y la clasificación estimada para los datos de prueba. | |
|  | |

**CONCLUSIONES**

|  |
| --- |
| Escribe tus observaciones y conclusiones. |
| La regresión logística es un método estadístico utilizado principalmente para resolver problemas de clasificación binaria, es decir, cuando se busca predecir entre dos posibles resultados o categorías. Aunque comparte nombre con la regresión lineal, la regresión logística emplea una función logística (sigmoide) para transformar la salida de una combinación lineal de variables en una probabilidad que oscila entre 0 y 1, facilitando la decisión sobre la categoría a la que pertenece una observación. Es ampliamente utilizada en áreas como medicina, marketing, finanzas y aprendizaje automático debido a su facilidad interpretativa, eficiencia computacional y capacidad para proporcionar resultados probabilísticos claros sobre eventos específicos. |