Práctica 3: Regresión Lineal

Ana Martín Sánchez, Nicolás Pastore Burgos

21/09/2021

1 Descripción de la práctica

En esta práctica, se pedía aplicar regresión logística multi-clase, con el fin de reconocer números escritos a mano en imágenes.

Para ello, se facilitan un total de 500 ejemplos de entrenamiento (imágenes con un número escrito a mano), en las que cada ejemplo se representa como una matriz de 20x20 números reales. Cada número indica la intensidad en escala de grises de un píxel en la imagen.

2 Solución propuesta

Con la implementación que proponemos a continuación, la precisión de la red neuronal se acerca al 97.5%.

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.optimize as opt

from scipy.io import loadmat

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

def costeReg(thetas, x, y, l):
    h = sigmoide(np.dot(x, thetas))
    return -((np.dot(np.log(h), y) + np.dot(np.log(1 - h + 1e-6), 1 - y)) /
     len(x)) + (l/(2*len(x))) * l * np.sum(thetas[1:] ** 2)

def gradienteReg(thetas, x, y, l):
    h = sigmoide(np.dot(x, thetas))
    return np.dot(x.T, h-y) / len(y) + (thetas * l) / len(y)

def oneVsAll(x, y, num_etiquetas, reg):
    theta = np.zeros((num_etiquetas, x.shape[1]))

    for i in range(num_etiquetas):
        theta[i] = opt.fmin_tnc(func=costeReg, x0=theta[i], fprime=
    gradienteReg, args=(x,(y==(i+9)%10+1)*1,reg))[0]

    return theta

def parte1():
    data = loadmat("Data/ex3data1.mat")

    x = data['X']
    y = data['y']
    yR = np.ravel(y)

    m = np.shape(x)[0]
    n = np.shape(x)[1]

    numClases = 10
    numExamples = 20

    reg = 1.0

    xNew = np.hstack([np.ones([m, 1]), x])

    theta = oneVsAll(xNew, yR, numClases, reg)

    sample = np.random.choice(xNew.shape[0], numExamples)

    correct = 0
    for i in range(m):
        result = sigmoide(np.matmul(theta, xNew[i, :]))
        id = np.argmax(result)
        if(y[i][0]%10 == (id+1)%10):
```

```
            correct+= 1

    correct = correct/m

    print("Correct values are:", (1-correct)*100, "%")

    for i in range(numExamples):
        result = sigmoide(np.matmul(theta, xNew[sample, :][i]))
        max = np.max(result)
        id = np.argmax(result)

        print("sample",i,y[sample,:][i]%10,": creemos que es un", id, "con una certeza del", max)

    plt.imshow(x[sample, :].reshape(-1, 20).T)
    plt.axis('off')

    plt.show()

"""
    ==========================================================
    ==========================================================
    ==========================================================
    ==========================================================
"""

def forwardProcess(x, num_capas, thetas):
    a = x
    m = x.shape[0]
    for i in range(num_capas):
        aNew = np.hstack([np.ones([m, 1]), a])
        a = sigmoide(np.dot(aNew,thetas[i].T))

    return a

def parte2():
    data = loadmat("Data/ex3data1.mat")

    x = data['X']
    y = data['y']
    yR = np.ravel(y)

    m = np.shape(x)[0]
    n = np.shape(x)[1]

    numExamples = 20

    pesos = loadmat("Data/ex3weights.mat")

    theta1, theta2 = pesos['Theta1'], pesos['Theta2']

    thetas = np.array([theta1, theta2], dtype='object')

```

```
1103      #sample = np.random.choice(m, 1)
1104
1105      res = forwardProcess(x, 2, thetas)
1106
1107      sample = np.random.choice(m, numExamples)
1108
1109      correct = 0
1110      for i in range(m):
1111          id = np.argmax(res[i, :])
1112          if(y[i][0]%10 == (id+1)%10):
1113              correct+= 1
1114
1115      correct = correct/m
1116
1117      print("Correct values are:", correct*100, "%")
1118
1119      for i in range(numExamples):
1120          max = np.max(res[sample, :][i])
1121          id = np.argmax(res[sample, :][i])
1122
1123          print("sample", i, y[sample,:][i]%10, ": creemos que es un", (id+1)
          %10, "con una certeza del", max)
1124
1125      plt.imshow(x[sample, :].reshape(-1, 20).T)
1126      plt.axis('off')
1127
1128      plt.show()
1129
1130  if __name__ == "__main__":
1131      parte1()
1132      parte2()
```

main.py