기계학습원론 HW4

2021312738 소프트웨어학과 김서환

1. The model is $f(x1, x2) = w2x2 + w1x1 + w0$

a) What are $\partial E/\partial w0$, $\partial E/\partial w1$, $\partial E/\partial w2$?

계산의 일관성을 위해 w0에 x0(x0=1)을 곱해져 있다고 가정하자.

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w0}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x0$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w1}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x1$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w2}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x2$$

b) Determine $w0$, $w1$, $w2$ for the logistic regression. Write the code for training of the logistic regression.

.py파일을 첨부하고 싶었으나 pdf형식이라 글로 첨부하겠습니다!

또한, Homework 4.data 파일 상대 경로로 파일을 읽어와서 데이터를 저장하였고 Homework 4.data 파일 데이터 x1, x2의 값이 커서 반복해서 학습시킬 때 max range error가 발생하여 이를 해결하면서 정답인 w값을 찾는 방법에 대해서 찾다가 데이터의 분포의 모양을 바꾸지 않는 스케일링이라는 방법을 생각했습니다. min-max 스케일링을 이용하여 구현하였고, w0, w1, w2값을 찾아냈습니다.

Logistic regression을 통해 도출한 w0 = -0.854, w1 = 13.436, w2 = -17.045 입니다.

```
[Running] python -u "c:\Users\kksh3\OneDrive\바탕 화면\5학기\기학원\HW\HW4\기학원 과제 1.py"
initial w0, w1, w2 :  -0.42642295287491283 ,  -0.37990619631058453 ,  0.4281188346160478
Solution w0, w1, w2:  -0.8544903705581934 ,  13.436394543490668 ,  -17.045424395649793
0.7610268446545655
1

[Done] exited with code=0 in 0.633 seconds

[Running] python -u "c:\Users\kksh3\OneDrive\바탕 화면\5학기\기학원\HW\HW4\기학원 과제 1.py"
initial w0, w1, w2 :  0.9782658266128335 ,  0.12224359282657415 ,  0.9153906714288926
Solution w0, w1, w2:  -0.8544848837156721 ,  13.436318033470844 ,  -17.045326385262424
0.7610258293849786
1

[Done] exited with code=0 in 0.642 seconds

[Running] python -u "c:\Users\kksh3\OneDrive\바탕 화면\5학기\기학원\HW\HW4\기학원 과제 1.py"
initial w0, w1, w2 :  0.45952758453619946 ,  -0.6559424370542428 ,  0.6353942935509993
Solution w0, w1, w2:  -0.8544752493683537 ,  13.436183689686368 ,  -17.04515428908146
0.7610240466661798
1
```

<python code>

```
import random
import math

eta = 0.01
max_iteration = 10000
limit_difference = 0.000001

def fxw(x1,x2):
    return w2*x2 + w1*x1 + w0
def Lxw(x):
    return 1/(1+math.exp(-x))

w0 = random.uniform(-1, 1)
w1 = random.uniform(-1, 1)
w2 = random.uniform(-1, 1)

data = []
file = open("Howework 4.data.txt", 'r')
lines = file.readlines()
for line in lines:
    numbers = line.split(',')
    temp = []
    for num in numbers:
        temp.append(int(num))
```

```
        data.append(tuple(temp))


x1list = []
x2list = []


for i in data:
        x1list.append(i[0])
        x2list.append(i[1])


x1min = min(x1list)
x1max = max(x1list)
x2min = min(x2list)
x2max = max(x2list)


scaleddata = []


for newx1, newx2, newy in data:
        scaledx1 = (newx1-x1min)/(x1max-x1min)
        scaledx2 = (newx2-x2min)/(x2max-x2min)
        scaleddata.append((scaledx1, scaledx2, newy))


print("initial w0, w1, w2 : ",w0,", ",w1,", ",w2)


for itr in range(max_iteration):
        g0 = 0
```

```
    g1 = 0
    g2 = 0
    for x1, x2, t in scaleddata:
        fx = fxw(x1,x2)
        derw0 = (Lxw(fx)-t)
        derw1 = (Lxw(fx)-t)*x1
        derw2 = (Lxw(fx)-t)*x2
        g0 = g0 + derw0
        g1 = g1 + derw1
        g2 = g2 + derw2

    neww0 = w0 - eta*g0
    neww1 = w1 - eta*g1
    neww2 = w2 - eta*g2

    if abs(neww0 - w0) < limit_difference and abs(neww1 - w1) < limit_difference and abs(neww2 - w2) < limit_difference:
        print("difference is too small")
        break

    w0 = neww0
    w1 = neww1
    w2 = neww2
print("Solution w0, w1, w2: ",w0,", ",w1,", ",w2)
scaledx1 = (33-x1min)/(x1max-x1min)
```

scaledx2 = (81-x2min)/(x2max-x2min)

```
classfy = Lxw(fxw(scaledx1, scaledx2))
print(classfy)
if(classfy > 0.5):
    print(1)
elif(classfy < 0.5):
    print(0)
else:
    print("unknown")
```

c) Determine the class of (33, 81).

b)의 코드 마지막 부분에 구현해두긴 했는데 스케일링을 해줬기 때문에 (33, 81)의 데이터도 스케일링을 거친 후에 f(x1,x2)에 대입해줬고, 그 결과를 다시 sigmoid함수 (= L(x,w))에 대입하여 나온 결과값이 0.5보다 크면 1로 분류, 0.5보다 작으면 0으로 분류했습니다. 따라서 실제 (33, 81)의 결과값이 0.761로, (33, 81)의 class는 1입니다.

2. The model is $f(x1, x2) = w5x2^2 + w4x1^2 + w3x2x1 + w2x2 + w1x1 + w0$

a) What are $\partial E/\partial w0$, $\partial E/\partial w1$, $\partial E/\partial w2$, $\partial E/\partial w3$, $\partial E/\partial w4$, $\partial E/\partial w5$?

계산의 일관성을 위해 w0에 x0(x0=1)을 곱해져 있다고 가정하자.

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w0}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x0$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w1}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x1$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w2}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x2$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w3}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x2\text{*x1}$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w4}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x1^2$$

$$\frac{\partial \mathrm{E}}{\partial \mathrm{w5}} = \sum_{(x,y)\in Data}\left(\frac{1}{1+e^{-w2x2-w1x1-w0}} - y\right) * x2^2$$

b) Determine *w*0, *w*1, *w*2, w3, w4, w5 for the logistic regression. Write the code for training of the logistic regression.

(1번과 동일하게).py파일을 첨부하고 싶었으나 pdf형식이라 글로 첨부하겠습니다!

또한, Homework 4.data 파일 상대 경로로 파일을 읽어와서 데이터를 저장하였고 Homework 4.data 파일 데이터 x1, x2의 값이 커서 반복해서 학습시킬 때 max range error가 발생하여 이를 해결하면서 정답인 w값을 찾는 방법에 대해서 찾다가 데이터의 분포의 모양을 바꾸지 않는 스케일링이라는 방법을 생각했습니다. min-max 스케일링을 이용하여 구현하였고, w0, w1, w2, w3, w4, w5값을 찾아냈습니다.

Logistic regression을 통해 도출한 w0 = 3.6, w1 = -1.7, w2 = -42.9, w3 = 6.7, w4 = 33.7, w5 = -3.8 입니다.

(매 실행마다 w0~w5까지의 값은 +1~-1정도의 오차가 존재)

```
[Running] python -u "c:\Users\kksh3\OneDrive\바탕 화면\5학기\기학원\HW\HW4\기학원 과제 2.py"
initial w0, w1, w2, w3, w4, w5:  -0.7071323248849257 ,  0.4817434854199034 ,  0.24586006233354096   -0.3273906925277661   -0.6746962494970596   -0.8437733745193199
Solution w0, w1, w2, w3, w4, w5:  3.6028299301675464 ,  -1.7167349760401496 ,  -42.86081541557386   6.742644354170586   33.77253350297373   -3.837048024610423
0.7845987506866494
1

[Done] exited with code=0 in 1.351 seconds

[Running] python -u "c:\Users\kksh3\OneDrive\바탕 화면\5학기\기학원\HW\HW4\기학원 과제 2.py"
initial w0, w1, w2, w3, w4, w5:  0.8605275854390946 ,  0.5040584117396332 ,  0.9869312824917105   0.8530341520782123   -0.46337667404576544   0.3274042923993665
Solution w0, w1, w2, w3, w4, w5:  3.606357499068286 ,  -1.6555790571589952 ,  -42.95349789414472   6.981836979876392   33.63019588706504   -3.8994442536306373
0.784220089574644
1

[Done] exited with code=0 in 1.358 seconds

[Running] python -u "c:\Users\kksh3\OneDrive\바탕 화면\5학기\기학원\HW\HW4\기학원 과제 2.py"
initial w0, w1, w2, w3, w4, w5:  -0.2357411477583976 ,  -0.5601512581721271 ,  0.34766454158326887   0.1184299334902641   -0.5954523988889147   -0.695970266751667
Solution w0, w1, w2, w3, w4, w5:  3.6103821466968364 ,  -1.8030325286135709 ,  -42.84523956065486   6.9158380291485235   33.855531608893486   -4.057106022425487
0.7853007812834794
1

[Done] exited with code=0 in 1.417 seconds
```

<python code>

```
import random

import math


eta = 0.01

max_iteration = 10000

limit_difference = 0.000001


def fxw(x1,x2):

    return w5*(x2**2) + w4*(x1**2) + w3*x2*x1 + w2*x2 + w1*x1 + w0
```

```
def Lxw(x):
    return 1/(1+math.exp(-x))

w0 = random.uniform(-1, 1)
w1 = random.uniform(-1, 1)
w2 = random.uniform(-1, 1)
w3 = random.uniform(-1, 1)
w4 = random.uniform(-1, 1)
w5 = random.uniform(-1, 1)

data = []
file = open("Howework 4.data.txt", 'r')
lines = file.readlines()
for line in lines:
    numbers = line.split(',')
    temp = []
    for num in numbers:
        temp.append(int(num))
    data.append(tuple(temp))

x1list = []
x2list = []

for i in data:
    x1list.append(i[0])
```

```
        x2list.append(i[1])


x1min = min(x1list)
x1max = max(x1list)
x2min = min(x2list)
x2max = max(x2list)


scaleddata = []


for newx1, newx2, newy in data:
        scaledx1 = (newx1-x1min)/(x1max-x1min)
        scaledx2 = (newx2-x2min)/(x2max-x2min)
        scaleddata.append((scaledx1, scaledx2, newy))


print("initial w0, w1, w2, w3, w4, w5: ",w0,", ",w1,", ",w2," ",w3," ",w4," ",w5)
for itr in range(max_iteration):
        g0 = 0
        g1 = 0
        g2 = 0
        g3 = 0
        g4 = 0
        g5 = 0
        for x1, x2, t in scaleddata:
                fx = fxw(x1,x2)
                derw0 = (Lxw(fx)-t)
```

```
            derw1 = (Lxw(fx)-t)*x1
            derw2 = (Lxw(fx)-t)*x2
            derw3 = (Lxw(fx)-t)*x2*x1
            derw4 = (Lxw(fx)-t)*(x1**2)
            derw5 = (Lxw(fx)-t)*(x2**2)
            g0 = g0 + derw0
            g1 = g1 + derw1
            g2 = g2 + derw2
            g3 = g3 + derw3
            g4 = g4 + derw4
            g5 = g5 + derw5

        neww0 = w0 - eta*g0
        neww1 = w1 - eta*g1
        neww2 = w2 - eta*g2
        neww3 = w3 - eta*g3
        neww4 = w4 - eta*g4
        neww5 = w5 - eta*g5

        if abs(neww0 - w0) < limit_difference and abs(neww1 - w1) < limit_difference and abs(neww2 - w2) < limit_difference and abs(neww3 - w3) < limit_difference and abs(neww4 - w4) < limit_difference and abs(neww5 - w5) < limit_difference:
            print("difference is too small")
            break

        w0 = neww0
```

```
    w1 = neww1
    w2 = neww2
    w3 = neww3
    w4 = neww4
    w5 = neww5
print("Solution w0, w1, w2, w3, w4, w5: ",w0,", ",w1,", ",w2," ",w3," ",w4," ",w5)
scaledx1 = (33-x1min)/(x1max-x1min)
scaledx2 = (81-x2min)/(x2max-x2min)


classfy = Lxw(fxw(scaledx1, scaledx2))
print(classfy)
if(classfy > 0.5):
    print(1)
elif(classfy < 0.5):
    print(0)
else:
    print("unknown")
```

c) Determine the class of (33, 81).

b)의 코드 마지막 부분에 구현해두긴 했는데 스케일링을 해줬기 때문에 (33, 81)의 데이터도 스케일링을 거친 후에 f(x1,x2)에 대입해줬고, 그 결과를 다시 sigmoid함수 (= L(x,w))에 대입하여 나온 결과값이 0.5보다 크면 1로 분류, 0.5보다 작으면 0으로 분류했습니다. 따라서 실제 (33, 81)의 결과값이 0.785로, (33, 81)의 class는 1입니다.