설계 프로젝트 HW 3

제출기한 : 23년 05월 30일 (화) 23:59분

과제는 작성한 프로그램 코드 및 실행 결과물을 포함하여 word로 작성해주세요. (레포트 반드시 첨부)
실행 결과물을 반드시 '캡쳐'해서 report 안에 포함하세요.
작성한 Programming 코드 (.py 또는 .ipynb) 를 꼭 함께 제출 하세요.
본인이 작성한 코드에 대해서는 주석을 통해 설명을 달아주세요.
파일이 2개 이상 일 경우 하나의 .zip파일로 업로드 하세요.
 # Copy 엄금 (ex. 코드공유 & 족보 참고 등) 적발 시 0점

[참고 1]
np.ones 및 np.zeros() 함수를 통해 행렬 또는 벡터를 만들어 줄 수 있다.

```
import numpy as np
np_ones = np.ones((2,2))     #1 으로만 이루어진 2X2 행렬을 초기화 한다.
np_zeros = np.zeros((2,2)) #0 으로만 이루어진 2X2 행렬을 초기화 한다.

np_ones_vec = np.ones((2,1))  #1으로만 이루어진 2 차원 벡터를 초기화 한다.
np_zeros_vec = np.zeros((2,1))  #1으로만 이루어진 2 차원 벡터를 초기화 한다.

print(np_ones_vec)
```

실행 예)

```
[[1.]
 [1.]]
```

[참고 2]
np.arange 함수를 통해 다음과 같은 numpy형 array list를 만들 수 있다.

```
import numpy as np
np_list_1 = np.arange(0,100,1) #0(첫번째 인자)부터 100(두번째 인자)까지 1(세번째 인자)씩 증가하는 리스트 생성
np_list_2 = np.arange(-5,5,0.1) #-5(첫번째 인자)부터 5(두번째 인자)까지 0.1(세번째 인자)씩 증가하는 리스트 생성

print(np_list_1)
print(np_list_2)
```

실행 예)
[0 1 2 … 98 99]
[-5.0 -4.9 … 4.8 4.9]

1. Sigmoid 함수 [15점]

1-1) sigmoid(x)를 python 함수로 작성하라. [5점]

(참고: exponential 함수는 math.exp()를 활용하시오)

$$f(x) = \frac{1}{1+e^{-x}}$$

```
# sigmoid function
import math
def sigmoid(x):

    y = ### 이부분 작성

    return y
```

실행 예)

```
>>> sigmoid(0.5)
>>> 0.6224593312018546
```

1-2) 리스트를 입력으로 받아서 sigmoid 연산을 계산해주는 sigmoid_list 함수를 작성하라. [5점]

([참고 1]의 numpy 모듈 예제를 참고하세요)

```
# sigmoid_list function
import numpy as np
import math
def sigmoid_list(x_list):

    y = np.zeros((len(x_list),1))
    ### 이부분 작성

    return y
```

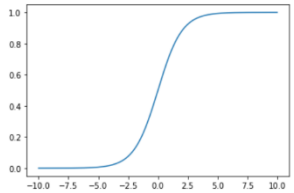
실행 예)

```
>>> print(sigmoid list([0,1,2]))
>>> [[0.5],[0.731],[0.8807]]
```

1-3) 문제 1-2 에서 설계한 sigmoid_list(x)를 이용해서, $-10 \leq x \leq 10$ 인 구간에 대해서 sigmoid 함수를 시각화 하라. [5 점]

단, x의 간격은 '0.01'로 촘촘하게 설정하라 ([참고 2]의 np.arange 활용법을 참고하세요)

```
from matplotlib import pyplot as plt
x_list = np.arange(###내부 인자값 작성)
plt.plot(x_list,sigmoid_list(x_list))
```



2. 인공 신경망 (Artificial Neural Networks) [40 점]

2-1) ann_data.csv 파일을 pandas 파일 입출력 객체를 활용하여 읽으시오 [df 라는 객체 명으로 읽으세요] [5 점]

샘플의 개수는 아래 실행예와 다를 수 있습니다.

실행 예)

```
>>> df
```

	x1	x2	class
0	0.4	0.3	1.0
1	0.3	0.8	-1.0
2	0.2	0.7	-1.0
3	0.7	0.4	1.0
4	0.6	0.2	1.0
5	0.1	0.6	-1.0

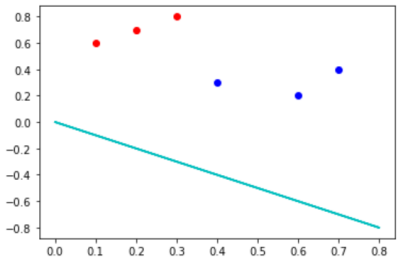
[참고 3] 인공 신경망의 학습 파라미터 $\mathbf{w} = (w_1, w_2)$와 b는 다음과 같이 임의로 초기화 했다.
학습 샘플들과 초기의 인공 신경망의 분류는 아래 함수 (visualize)를 통해 다음과 같이 출력 된다.

```
w = [0.5, 0.5]
b = 0
```

```
def visualize(df,w,b):
    for idx in range(len(df)):

        x1_1 = 0
        y1_1 = (-w[0]*x1_1-b)/w[1]

        x1_2 = 0.8
        y1_2 = (-w[0]*x1_2-b)/w[1]

        # plot samples
        if df.iloc[idx,2] == 1:
            plt.plot(df.iloc[idx,0],df.iloc[idx,1],'bo')
        else:
            plt.plot(df.iloc[idx,0],df.iloc[idx,1],'ro')

        #plot classifier

        plt.plot([x1_1,x1_2],[y1_1,y1_2])
```

실행 예) # 입력 데이터가 달라졌으므로 실행 예는 다를 수 있음.

```
>>> visualize(df,w,b)
```



2-2) 초기의 인공신경망이 주어진 학습데이터를 잘 분류 하는가? 그렇지 않다면 왜 그런가? 구체적으로 설명하시오. [5 점]

[참고 4] 각 파라미터 (w_1, w_2, b)의 update rule 은 다음과 같다.

Update rules

$$w_1{}^{+} = w_1 - \mu\frac{\partial C}{\partial w_1} = w_1 + \mu\sum_{n=1}^{N}(t_n - y_n)y_n(1-y_n)x_{n,1} \quad (1)$$

$$w_2{}^{+} = w_2 - \mu\frac{\partial C}{\partial w_2} = w_2 + \mu\sum_{n=1}^{N}(t_n - y_n)y_n(1-y_n)x_{n,2} \quad (2)$$

$$b^{+} = b - \mu\frac{\partial C}{\partial b} = b + \mu\sum_{n=1}^{N}(t_n - y_n)y_n(1-y_n) \quad (3)$$

$$C = \frac{1}{2}\sum_{n=1}^{N}(t_n - y_n)^2$$

$$y = sigmoid(z)$$

$$z = \sum_{i}^{m} w_i x_i + b$$

[참고 5] 학습 데이터들의 각 특징인 x_1과 x_2는 다음과 같이 list 데이터로 참조 한다 (각각 6 X 1 의 벡터)
또한 학습 데이터의 클래스 tn 또한 아래와 같이 list 데이터로 참조한다.

```
x1 = np.array(df.iloc[:,0:1])
x2 = np.array(df.iloc[:,1:2])
tn = np.array(df.iloc[:,2:3])
```

2-3) 입력으로 w,b, x_1, x_2를 받아 z 를 계산하는 zeta 함수를 작성하라. 그리고 아래 실행 예와 같이 1-2)에서 작성한 sigmoid_list 함수에 zeta 함수의 출력 값을 넣어 실행시켜라. (그 결과가 [참고 4]수식의 y 가 된다.) **[10 점]**

** 아래 수식은 학습 샘플이 6 개일때의 예시이며, ann_data.csv 의 파일 개수에 맞게 동작하도록 수정이 필요함*

```
def zeta(w,b,x1,x2):
    z = np.zeros((6,1))

    for idx in range(6):
        z[idx] = ## 이부분 작성
    return z
```

실행 예) # 입력 데이터가 달라졌으므로 실행 예는 다를 수 있음.

```
>>> y = sigmoid_list(zeta(w,b,x1,x2))
>>> print(y)
[[0.58661758]
 [0.63413559]
 [0.61063923]
 [0.63413559]
 [0.59868766]
 [0.58661758]]
```

2-4) 경사하강법을 이용하여 학습 파라미터 W 와 b 를 학습하라. μ(learning rate)는 0.2 로 설정하고 파라미터 업데이트를 5,000 번 반복하라. [15 점]

[참고] `np.ones((10,1))-y # 1-y 의 계산`

```
w = [0.5, 0.5] # initial weights
b = 0 # initial bias
mu = 값 설정
x1 = np.array(df.iloc[:,0:1])
x2 = np.array(df.iloc[:,1:2])
tn = np.array(df.iloc[:,2:3])

for i in range(반복횟수 설정):
    y =  sigmoid_list(zeta(w,b,x1,x2))

    ## 각 파라미터의 경사하강법 계산 및 업데이트 [참고 4] 수식 참조

    ## 파라미터 업데이트  (위의 계산이 모두 끝난 다음에 한번에 업데이트 한다)
```

2-5) 최종 학습 완료된 파라미터 w,b 를 이용하여 학습된 분류기 및 학습데이터들을 시각화 하라. [참고 3 코드 활용]
학습된 결과는 주어진 데이터들을 잘 분류하는가? [5 점]

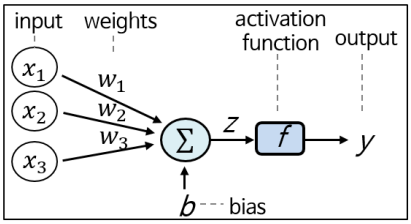
3. 유전알고리즘 (Genetic Algorithm) [45 점]

[참고 1]
데이터 로드 및 평면 출력

```
%matplotlib notebook
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import numpy as np
import pandas as pd
import random

random.seed(10) # Random 함수의 seed 고정
np.random.seed(10) # numpy.random 함수의 seed 고정

data = pd.read_csv("p1_training_data.csv") # 데이터 읽기
np_data = np.array(data)

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

## positive samples
x_1 = np_data[0:50,0]
y_1 = np_data[0:50,1]
z_1 = np_data[0:50,2]

## negative samples
x_0 = np_data[50:,0]
y_0 = np_data[50:,1]
z_0 = np_data[50:,2]

## Generation 1의 fittest gene
w1 = 0.28645574
w2 = -0.43628723
w3 = 0.30481866
b = -14.39337271

ax.plot(x_1, y_1, z_1, linestyle="none", marker="o", mfc="none", markeredgecolor="b") #샘플 출력
ax.plot(x_0, y_0, z_0, linestyle="none", marker="o", mfc="none", markeredgecolor="r") #샘플 출력

X = np.arange(0, 2, 0.1)*100
Y = np.arange(0, 2, 0.1)*100
X, Y = np.meshgrid(X, Y)

Z = (-float(w1)/w3 * X) + (-float(w2)/w3 * Y) - float(b)/w3 # 평면의 방정식

ax.plot_surface(X, Y, Z, rstride=4, cstride=4, alpha=0.4, cmap=cm.Blues) # 평면 출력
plt.show()
```

1 유전 알고리즘을 통해 주어진 데이터를 선형 분류하는 single-layer neural network를 학습하라.
아래 세부 사항을 확인하여 문제를 해결 하시오.

- 학습 데이터: p1_training_data.csv (100개x4차원데이터, 1~3차원 데이터는 입력데이터 (x1,x2,x3), 4번째 데이터는 클래스 y(1 또는 0))
- 학습 파라미터: **w** (3차원), b (1차원) [아래 그림 참조]



- Activation function: Sigmoid function 을 사용할 것.
- Population수는 100 이하로 설정하라.
- 위 조건 이외의 유전알고리즘 동작을 위한 모든 설계 조건은 자율적으로 설계 할 것 (**단, 레포트에 설계 내용을 모두 기입**)
- (fitness계산방법, Selection방법, crossover방법, mutation방법, 유전자(학습 파라미터)초기화 방법, 알고리즘 종료 조건 등)

1-1) 1세대 유전자를 초기화 하라. 어떤 방식으로 유전자를 초기화 하였는가? Population은 얼마로 세팅하였는가? [5점]
(hint. 각각의 weight는 -1~1사이로, bias는 -100~100 사이로 초기화 한다.)

1-2) 각 유전자에 대한 fitness 계산법을 설계하고 가장 fitness가 높은 유전자들을 선별하라. [10점]
fitness 계산은 어떻게 하였는가?
1세대에서 가장 fitness가 높은 상위 4개의 유전자에 대해 fitness score를 적고, 분류 평면을 도시화 하라 [실행 예 참고]

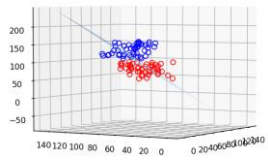
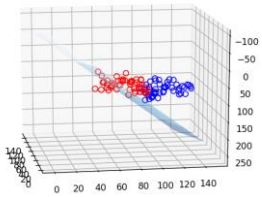
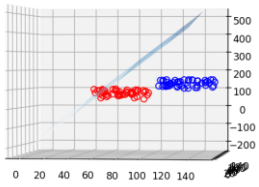
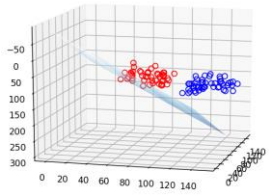
Ranking1, Fitness score: 30.48	Ranking2, Fitness score: 36.14	Ranking3, Fitness score: 44.03	Ranking4, Fitness score: 45.89
			

표1. [실행 예] 1세대 유전자 (Rank1~4)

1-3) 선택된 유전자들에 대해 Crossover와 mutation을 수행하여 2세대 유전자들을 생성하시오. 이때 생성된 자식세대의 population 수는 1세대와 동일하게 설계 하시오.

Crossover는 어떻게 수행하였는가? 상세히 작성하시오. [10점]

Mutation은 어떻게 수행하였는가? 상세히 작성하시오. [5점]

1-4) (2,3)과정을 반복하여 2세대, 3세대에서 가장 fitness가 높은 상위 4개의 유전자에 대해 fitness score를 적고, 분류 평면을 도시화 하라. [10점]

Ranking1, Fitness score:	Ranking2, Fitness score:	Ranking3, Fitness score:	Ranking4, Fitness score:

표2. 2세대 유전자 (Rank1~4)

Ranking1, Fitness score:	Ranking2, Fitness score:	Ranking3, Fitness score:	Ranking4, Fitness score:

표3. 3세대 유전자 (Rank1~4)

1-5) 최종적으로 유전알고리즘을 통해 얻어진 유전자중 가장 fitness가 높은 유전 자에 대해 분류 평면을 도시화 하라. 또한 어떤 조건으로 유전 알고리즘을 종료 하였는지 작성하라. [5점]

[실행예]

3세대 fittest gene. Fitness score : 0.000012