projects\project1\solution\main.py

```python
import math
from mission_test import mission1_test, mission2_test, mission3_test

########################################################################
# START Mission 1: Sigmoid 함수 만들기
# 평가 요소
# 적절한 인자를 받고, 올바른 리턴값을 돌려주는 함수를 작성할 수 있는지 평가
########################################################################
#
# Mission 1 설명
#
# 인공지능 개발에 자주 사용되는 함수 sigmoid는 다음처럼 계산된다.
#
#       1
# ----------------
# 1 + math.exp(-x)
#
# 위 식에서 x는 이 함수로 전달되는 입력값 x이며
# math.exp()는 math 모듈에서 제공하는 지수 함수를 계산해 주는 함수이다.
# 지수함수는 a**x 꼴의 함수로 math.exp()는 a가 약 2.718정도 값을 가지는 e라는 숫자
# 일때 함수값을 계산해주는 함수이다.
# x로 숫자 하나를 입력받아 sigmoid 함수 값을 계산하는 함수를 다음처럼 작성할 수 있다.
# 아래 함수는 sigmoid함수를 구현한 예시인데 이 함수는 입력으로 숫자 하나를 입력 받고
# 출력으로 숫자 하나를 반환한다.
def sigmoid_one_value(x):
    return 1 / ( 1 + math.exp(-x) )

# 첫번째 미션에서 할 코딩은 x에 숫자 여러 개가 들어 있는 리스트가 주어지면
# 리스트의 각 숫자에 대해서 sigmoid 함수값을 모두 구해 그 값을 저장한
# 리스트를 되돌리는 함수를 작성하는 것이다.

# 즉 다음처럼 함수를 실행하면
# sigmoid([1,2,3,4])
# 실행의 결과 값이 다음처럼 4개 나오게 된다.
# [0.7310585786300049, 0.8807970779778823, 0.9525741268224334, 0.9820137900379085]

# 아래 함수에 빈 곳(None)을 채우시오.
# 빈 곳을 함수의 정의대로 다시 코딩해도 좋고 위에서 만들어 놓은 sigmoid_one_value()를
# 재사용해도 좋음
def sigmoid(x):
    # x: 숫자 여러개가 들어있는 리스트

    # x안에 있는 숫자값 하나에 대해서 계산된 함숫값을 저장할 리스트를 정의
    ret = []

    # x안에 있는 숫자에 대해서 루프를 돌면서
    for x_i in x:
        # x_i에 대해서 위에 제시된 sigmoid 함숫값을 계산해서
        # ret에 추가(append) 한다.
        ret.append( 1 / ( 1 + math.exp(-x_i) ) ) # ret.append( sigmoid_one_value(x_i) )
```

```
    # 완성된 ret를 반환한다.
    return ret

# [!주의!] 다음 코드를 수정하시 마시오.
# sigmoid 함수를 완성하고 다음 mission1_test()를 실행하여
# 화면에 PASS가 나오는지 확인하시오.
MISSION_1 = mission1_test(sigmoid)
######################################################################
# END Mission 1
######################################################################


######################################################################
# START Mission 2: 파일에서 데이터 로딩
# 평가 요소
# 파일을 읽기 모드로 열고 읽어서 내용을 읽어서 알맞은 자료구조로 정리해서
# 저장할 수 있는지 평가
######################################################################
#
# Mission 2 설명
#
# 현재 파일과 같은 폴더에 있는 데이터 파일 data_x.txt, data_y.txt 파일을 읽어
# 지시대로 적절한 변수에 저장한다.
# 위 미션 1을 통화해야 미션 2가 실행됨
if MISSION_1:
    # data_x.txt에 있는 숫자를 읽어서 X_temp에 추가한다.
    # data_x.txt에는 한 줄에 숫자가 하나씩 적혀있는데 각 숫자는 꽃의 스펙을 나타낸다.
    # 구체적으로는 4개씩 끊어 꽃받침 길이, 너비, 꽃잎 길이, 너비 이다.
    # 처음 8개의 값을 예로 들면
    #
    # 4.9 <— 1번 샘플의 꽃받침 길이
    # 2.4 <— 1번 샘플의 꽃받침 너비
    # 3.3 <— 1번 샘플의 꽃잎 길이
    # 1.  <— 1번 샘플의 꽃잎 너비
    # 5.8 <— 2번 샘플의 꽃받침 길이
    # 2.7 <— 2번 샘플의 꽃받침 너비
    # 4.1 <— 2번 샘플의 꽃잎 길이
    # 1.  <— 2번 샘플의 꽃잎 너비
    #
    # 총 200라인에 걸쳐 숫자가 적혀 있으므로 4개씩 묶으면 총 50개 꽃에 대한 정보가
    # 적혀있는 셈이다.
    # 이 숫자들을 다 읽어서 4개씩 끊어서 다음과 같은 2차원 리스트 형태로
    # `X`변수에 저장하는 것이 첫번째 목표이다.
    # [
    #   [4.9, 2.4, 3.3, 1.],
    #   [5.8, 2.7, 4.1, 1.],
    #   ... 이후 계속 반복
    # ]

    # `X_temp`에 모든 숫자를 순서대로 다 담는다.
    X_temp = []
    # 파일을 오픈하고
    with open('data_x.txt', 'r') as f:
        # 오픈된 파일로부터 모든 라인을 읽어온다.
```

```
            lines = f.readlines()

            # 읽어온 한 라인을 순서대로 순회하면서
            for line in lines:
                # 읽어온 값은 문자형이므로 실수형으로 바꿔서 X_temp에 추가한다.
                X_temp.append( float(line) )

        # `X_temp`에 있는 숫자를 4개씩 끊어서 `X`에 담는다.
        X = []
        # for 문 순회를 하나씩 하지 않고 4개마다 하나씩 하고
        for i in range(0, len(X_temp), 4):
            # i부터 4개씩 X_temp를 슬라이싱 한다.
            X.append(X_temp[i:i+4])

        # 두번째 작업은 첫번째 작업과 유사하게 data_y.txt에 있는 자료를
        # y에 저장한다.
        # data_y.txt에는 50개 꽃 샘플에 대한 품종이 적혀있다.
        # 품종 구분은 'verginica'와 'not verginica'로 구분되어 있다.
        # `y`에 50 라인을 읽어 저장한다.
        # [주의] 읽어온 문자열 좌우에 공백이 없도록 가공해서 y에 저장할것
        y = []
        with open('data_y.txt', 'r') as f:
            lines = f.readlines()
            for line in lines:
                y.append(line.strip())

        # `X`, `y` 에 알맞게 데이터가 저장되었다면 MISSION_2가 True로 설정되면서
        # PASS가 출력된다.
        MISSION_2 = mission2_test(X, y)
    ##########################################################################
    # END Mission 2
    ##########################################################################

    ##########################################################################
    # START Mission 3: 인공지능 분류기 만들기
    # 평가 요소
    # 이중 for 루프를 원활하게 사용할 수 있는지
    # 변수의 연산을 올바르게 수행할 수 있는지
    # if 조건문을 바르게 사용할 수 있는지 평가
    ##########################################################################
    #
    # Mission 3 설명
    #
    # Mission 2에서 X, y에 저장된 꽃의 정보를 사용하여 꽃의 품종을 예측하는
    # 예측기를 만든다.
    if MISSION_1 and MISSION_2 :
        # 다음 설정된 숫자는 아래처럼 계산하는대 사용된다.
        # X에 들어 있는 첫번째 데이터 X[0]를 x라고 하면 x에는
        # 숫자 네 개가 들었다. 이 숫자 네 개에 대해서 다음 식을 계산한다.
        #
        # z = x[0]*weight[0] + x[1]*weight[1] + x[2]*weight[2] + x[3]*weight[3] + intercept
        #
        # 위 식은 x안에 들어 있는 꽃의 스펙 숫자 네 개를 아래 weight에 들어있는 숫자
        # 네 개와 순서대로 곱해서 다 더하고 마지막에 intercept를 더해주는 식이다.
        #
```

```
# 이런 작업을 X에 들어있는 모든 개별 꽃 샘플 스펙 X[0], X[1], ..., X[49]에 대해서 반복한다.
weight = [-0.11891001,  0.00524661,  1.9831564 ,  1.34528917]
intercept = -11.22259598

# 아래 여섯 개 step을 지시대로 완성하시오.
Z = []
# step 1. X에 들어있는 길이 4짜리 리스트를 하나씩 가져온다.
#         for 루프를 사용하고 하나씩 가져오는 리스트는 x 변수에 할당한다.
for x in X:
    # step 2. for 루프로 받아온 x안에 있는 숫자 네 개와 weight에 있는 숫자 네개를
    #         각 위치에 맞게 곱하고 모두 더한다.
    #         즉 다음 식처럼 숫자값 z를 계산한다.
    #         z = x[0]*weight[0] + x[1]*weight[1] + x[2]*weight[2] + x[3]*weight[3]
    #         이렇게 계산된 z에 intercept를 더한다.
    #         z = z + intercept
    z = 0
    for i in range(4):
        # weight에 숫자 네 개와 곱해서 더한다.
        z += x[i]*weight[i]
    # 마지막으로 intercept를 더한다.
    z += intercept

    # step 3. 계산된 z를 Z 리스트에 append 한다.
    Z.append(z)


# step 4. Z에 들어있는 계산된 모든 값(50개) 들을 sigmoid()에 입력시켜
#         변환된 값 50개를 반환 받는다.
#         sigmoid()로 부터 반환된 값을 y_preds에 저장한다.
#         이렇게 저장된 값은 모두 0에서 1사이의 숫자값으로 변환 되어 있어야 한다.
y_preds = sigmoid(Z)

# step 5. y_preds에 저장된 50개 값이
# 0.5보다 같거나 크면 y_pred_i에 'verginica'를 대입
# 0.5보다 작으면 y_pred_i에 'not verginica'를 대입
# 이렇게 해서 각 50개 샘플에 대해서 꽃의 품종을 결정한다.
incorrect = 0

# 정답이 적힌 y안에 요소들과 예측한 y_preds의 요소를
# 순서대로 하나씩 가져와 비교한다.
# 두 리스트 y, y_preds를 동시에 for 문으로 순회하시오.
for y_i, y_pred_i in zip(y, y_preds):
    if y_pred_i >= 0.5:
        y_pred_i = 'verginica'
    else:
        y_pred_i = 'not verginica'

    # step 6. 정답 꽃품종 y_i와 예측 꽃 품종 y_pred_i가 다르면
    # incorrect를 1 증가 시킨다.
    if y_i != y_pred_i:
        incorrect += 1

# 결과를 출력한다.
print("    틀린 개수: ", incorrect , "개", sep='')
print('    정확도: ',  (len(X)-incorrect) / len(X), sep='' )
```

```
    # 제대로 진행되었다면 화면에 PASS 가 출력된다.
    mission3_test(incorrect)
#########################################################################
# END Mission 3
#########################################################################
```